Engineering Trustworthy Intelligent Systems

ETIS Instructor Course Package

Course Design and Educational Operations

Software Engineering, Governance, and Operational Trust in the AI Era

William T. O’Connell, Ph.D.

FIRST EDITION EDUCATIONAL PRODUCT

Contents

Copyright and Use

This educational product is part of the Engineering Trustworthy Intelligent Systems educational product suite.

ETIS Educational Product Series

This document is part of the **ETIS Educational Product Series**.

The educational product series transforms *Engineering Trustworthy Intelligent Systems* from a publication into a teachable, adoptable, and stewardable engineering framework.

The series is designed for instructors, students, departments, universities, professional educators, and institutions adopting ETIS in software engineering, AI governance, responsible AI, enterprise systems, capstone, project-based, or professional-practice environments.

Product Family

The ETIS Educational Product Series includes:

Product	Primary Purpose
ETIS Educational Ecosystem Guide	Explain the full ETIS educational architecture and public product model
ETIS Instructor Course Package	Provide the instructor operating system for course design and delivery
ETIS Classroom Facilitation Guide	Help instructors run ETIS classrooms as active engineering environments
ETIS Instructor Handbook	Preserve instructor guidance, teaching judgment, and long-term stewardship practices
ETIS Student Professional Engineering Guide	Help students understand and practice professional engineering behaviors

Relationship to the ETIS Book

The ETIS book remains the authoritative doctrine.

The educational product series translates that doctrine into teaching, learning, adoption, classroom operation, and stewardship resources.

Educational products teach ETIS.

Adoption examples prove ETIS.

Educational stewardship sustains ETIS over time.

Common Educational Premise

ETIS education is built on a simple premise:

> AI can produce artifacts. Engineers create trust.

Students should not merely complete assignments.

They should develop evidence of engineering maturity.

Instructors should not merely deliver content.

They should operate educational systems that help students practice trustworthy engineering.

Shared Educational Mission

ETIS educational products teach future engineers to:

- define intent
- engineer context
- bound authority
- verify behavior
- operate reality
- explain decisions
- own outcomes

Use of This Product

This product is intended to be used as a public educational resource.

It should be read together with the ETIS book, appendices, educational ecosystem pages, instructor resources, student resources, flagship implementation guidance, and institutional adoption guidance.

ETIS Instructor Course Package

Who This Is For

This product is for instructors who want to design and operate ETIS-based courses.

It supports:

- software engineering courses
- AI governance courses
- responsible AI courses
- enterprise systems courses
- capstone courses
- project-based software courses
- graduate professional-practice courses
- institutional pilots

Purpose

The Instructor Course Package is the educational operating system for instructors.

It helps instructors design a course where students practice engineering accountability, repository-centered evidence, AI responsibility, reviewability, release defense, operational thinking, and stewardship.

Instructor Operating Premise

An ETIS course is not a conventional programming course.

It is a professional engineering environment.

Students should learn that software engineering work is complete only when it can be explained, reviewed, verified, governed, operated, and defended.

The Six Educational Engines

The Instructor Course Package is organized around six educational engines.

Engine	Directory Logic	Guiding Question
Educational Intent Engine	syllabus guidance	What will students experience?
Educational Sequencing Engine	schedule guidance	In what order will students mature?
Educational Accountability Engine	assignment sequence guidance	How will students progressively prove engineering accountability?
Educational Evaluation Engine	assessment guidance	How will we know students are becoming trustworthy engineers?
Educational Operations Engine	classroom facilitation	How do instructors run ETIS classroom experiences?
Educational Stewardship Engine	instructor notes	What should instructors know while operating ETIS educational systems over time?

These engines should not be expanded casually.

They define the instructor operating model.

Course Design Principles

ETIS course design should follow these principles:

- Educational work should resemble professional engineering work.
- Engineering accountability is the educational outcome.
- AI usage should be governed, not avoided.
- Students should preserve evidence.
- Review should occur throughout the semester.
- Assignments should operate as phase gates.
- Release defense should be explicit.
- Operational thinking should appear before the course ends.
- Every semester should leave evidence for the next instructor.

Course Experience

Students should experience the course as a staged transformation.

They begin as students completing assignments.

They should end as emerging trustworthy engineers who can explain:

- what they built
- why it matters
- what evidence supports it
- how AI was used
- what risks remain
- what was reviewed
- what was tested
- whether the system is release-ready
- what should improve next

Syllabus Guidance

The syllabus should communicate that the course is about professional engineering accountability.

It should make clear that students will be expected to:

- work in teams
- maintain a repository
- use AI responsibly
- preserve evidence
- participate in reviews
- defend engineering decisions
- improve system maturity over time

AI use should be allowed, but governed.

Undisclosed and unverified AI dependency should be treated as an engineering risk.

Semester Sequencing

ETIS sequencing should move students from foundations to engineering practice, then into release and operational thinking.

A common arc is:

Foundation ↓ Project Launch ↓ Requirements ↓ Architecture ↓ Construction ↓ Verification ↓ Cycle 1 Release ↓ Operational Maturity ↓ Final Release Defense ↓ Stewardship Reflection

The sequence should mature students, not merely move through topics.

Assignment Sequencing

Assignments should function as engineering phase gates.

A six-checkpoint model may include:

Checkpoint	Purpose
Project Launch and Repository Foundation	Establish team, repository, workflow, AI expectations, and project intent
Requirements, Planning, Risk, and Traceability	Preserve requirements, assumptions, risks, planning, and traceability
Architecture and Review	Establish architecture, ADRs, dependencies, boundaries, and design review
Construction, Integration, Review, and Validation	Produce implementation, PR/review evidence, AI-use evidence, and test evidence
Cycle 1 Release Readiness	Defend the first controlled release with evidence
Final Release and Operational Maturity	Improve system maturity and defend final release readiness

Assignments should not be isolated submissions.

They should accumulate into an engineering evidence package.

Assessment Guidance

Assessment should evaluate both product and evidence.

Instructors should assess:

- requirements clarity
- architecture reasoning
- team accountability
- repository evidence
- AI disclosure and verification
- review quality
- testing evidence
- release readiness
- operational maturity
- final engineering defense
- professional communication

Rubrics should measure trustworthiness, not just task completion.

Evaluate the strength of the evidence before evaluating the strength of the conclusion.

Classroom Operations

The instructor should operate the classroom as an engineering organization.

That means:

- reviewing progress
- surfacing ambiguity
- challenging assumptions
- requiring evidence
- asking release-readiness questions
- guiding AI responsibility
- encouraging team accountability
- treating mistakes as learning evidence
- preserving continuity across the semester

The instructor is not merely delivering lectures.

The instructor is operating an educational engineering system.

Course Readiness

Before launching an ETIS course, instructors should confirm:

- the course purpose is clear
- the repository model is defined
- AI expectations are explicit
- assignment checkpoints are sequenced
- review moments are planned
- assessment criteria are aligned
- final defense expectations are known
- students understand that evidence matters

Continuous Improvement

Every offering should leave evidence for the next offering.

Useful instructor evidence includes:

- assignment observations
- rubric improvements
- student confusion patterns
- AI-use issues
- team coordination issues
- review-board lessons
- repository structure lessons
- grading calibration notes
- semester closeout reflections

Educational memory is educational infrastructure.

How To Use This Resource

Use this package before and during course design.

It should guide:

- syllabus development
- schedule planning
- assignment construction
- assessment design
- classroom operation
- end-of-semester improvement

Relationship to Other ETIS Products

Use this product with:

- the Educational Ecosystem Guide for overall educational architecture
- the Classroom Facilitation Guide for daily and weekly operations
- the Instructor Handbook for teaching judgment and stewardship
- the Student Professional Engineering Guide for student-facing orientation

Bottom Line

The ETIS Instructor Course Package helps instructors build courses where students do more than produce software.

They produce evidence of engineering maturity.

That is the real educational outcome.

Part I

Course Architecture

ETIS Course Design Guide

The **ETIS Course Design Guide** provides a structured approach for designing courses that teach *Engineering Trustworthy Intelligent Systems (ETIS)*.

ETIS course design is fundamentally different from traditional software engineering course design.

Traditional courses often organize instruction around technical topics.

ETIS organizes instruction around engineering responsibility, evidence generation, reviewability, operational thinking, and professional accountability.

The objective is not to help students build software.

The objective is to help students become trustworthy engineers.

Purpose

The purpose of this guide is to help instructors design ETIS-based learning experiences that are coherent, teachable, maintainable, and aligned with ETIS doctrine.

This guide helps instructors answer questions such as:

- What should students become by the end of the course?
- Which ETIS capabilities should be emphasized?
- How should assignments be sequenced?
- What evidence should students produce?
- How should AI-assisted work be governed?
- How should class time be used?
- How should engineering maturity be measured?
- How should final engineering accountability be demonstrated?

Course design should optimize for professional formation rather than content coverage.

ETIS Course Design Philosophy

ETIS courses are not collections of lectures.

ETIS courses are engineered systems.

Every component should exist for a reason.

Students should understand why they are performing an activity and what engineering responsibility it teaches.

Course components should work together as an integrated system.

This includes:

- lectures,
- readings,
- repositories,
- assignments,
- exercises,
- reviews,
- assessments,
- AI-use governance,
- release readiness activities,
- operational thinking,
- and final defense activities.

The course itself should model trustworthy engineering.

Class sessions should be engineered, not planned.

Sessions are not isolated instructional events.

Sessions are components of a larger educational system.

Each session should intentionally move students along the ETIS transformation model while creating opportunities for:

- ownership,
- evidence generation,
- reviews,
- engineering accountability,
- AI governance,
- and operational thinking.

Course design creates educational systems rather than collections of class meetings.

Foundational Course Design Principle

Course design begins with a simple principle:

> Design for professional transformation, not information transfer.

Information is abundant.

AI can generate explanations, examples, summaries, and artifacts.

The educational value of ETIS comes from helping learners practice engineering judgment.

Course Design Hierarchy

ETIS course design follows a hierarchy.

Professional Transformation Goal

↓

Engineering Capabilities

↓

Learning Experiences

↓

Assignments and Reviews

↓

Repository Evidence

↓

Assessment

↓

Professional Accountability

Design should always proceed from top to bottom.

Never start by designing assignments.

Assignments emerge from the transformation goals.

Step 1 — Define The Professional Transformation Goal

Every ETIS course should begin by defining who the learner should become.

Examples include:

Student
↓
Responsible Engineer

Student
↓
Reviewer

Student
↓
Release Defender

Student
↓
Operational Thinker

Student
↓
Future Trustworthy Engineer

A course may support multiple transformations, but one should be primary.

This transformation goal becomes the anchor for every design decision.

Step 2 — Select Primary Engineering Capabilities

ETIS contains many engineering capabilities.

No single course should emphasize all capabilities equally.

Select a focused set.

Examples include:

Foundation Capabilities

- requirements discipline,
- planning,
- estimation,
- risk identification,
- repository-centered engineering,
- architecture reasoning,
- technical communication.

Construction Capabilities

- implementation,
- testing,
- integration,
- pull requests,
- review participation,
- defect management.

Governance Capabilities

- AI-use governance,
- decision documentation,
- traceability,
- accountability,
- transparency.

Operational Capabilities

- observability,
- postmortems,
- release readiness,
- incident response,
- operational maturity.

Stewardship Capabilities

- continuous improvement,
- long-term ownership,
- engineering memory,
- trustworthy systems thinking.

Capability selection creates focus.

Step 3 — Map The ETIS Book

The ETIS book remains authoritative.

Course design should map the book intentionally.

Instructors should identify:

Required Chapters Students must understand these concepts deeply.

Supporting Chapters Students should be exposed to these concepts.

Reference Chapters Students may consult these concepts as needed.

For example:

Undergraduate Software Engineering Course Primary emphasis:

- Part I
- Part II

Selective introduction:

- Parts III and IV

Graduate Course Primary emphasis:

- Parts II, III, and IV

Professional Training Focused subset tied to organizational needs.

The course should not attempt to cover all chapters equally.

Step 4 — Design Repository Expectations

The repository should become the authoritative engineering record.

Students should understand that the repository is evidence, not storage.

Repository expectations should be defined before assignments are created.

A repository may include:

/docs /requirements /planning /architecture /decisions /reviews /testing /quality /release /operations /observability /security /governance /ai /postmortems /presentations

/src

/tests

/.github

The exact structure may vary.

The evidence expectations should not.

Step 5 — Design Engineering Phase Gates

Assignments should function as maturity gates.

Each assignment should increase engineering responsibility.

Example sequence:

Phase Gate 1 Repository foundation.

Phase Gate 2 Requirements, planning, risk, and traceability.

Phase Gate 3 Architecture and review.

Phase Gate 4 Implementation, testing, and validation.

Phase Gate 5 Cycle 1 release readiness.

Phase Gate 6 Postmortem and stabilization.

Phase Gate 7 Operational readiness.

Phase Gate 8 Final engineering defense.

Students should feel the system becoming more mature over time.

Step 6 — Design The Two-Cycle Model

ETIS courses benefit from two engineering cycles.

Cycle 1 — Can It Work? Focus:

- requirements,
- planning,
- architecture,
- implementation,
- testing,
- controlled release.

Students demonstrate responsible construction.

Cycle 2 — Can It Survive? Focus:

- feedback,
- defects,
- observability,
- risks,
- governance,
- postmortems,
- operational maturity.

Students demonstrate responsible operation.

The second cycle is often where ETIS becomes differentiated from traditional courses.

Step 7 — Design AI Governance

AI should be integrated deliberately.

Students should understand:

AI May Assist With

- requirements exploration,
- ambiguity detection,
- planning,
- architecture critique,
- implementation,
- testing,
- documentation,
- review preparation,
- operational reasoning.

Students Remain Responsible For

- correctness,
- security,
- maintainability,
- architecture fit,
- testing,
- documentation,
- disclosure,

- validation,
- professional judgment.

The governing principle remains:

> AI-assisted work is not accepted engineering work until humans review, verify, and own it.

Step 8 — Design Classroom Experiences

Class time should not become lecture-only time.

ETIS encourages active engineering experiences.

Examples include:

- requirements review,
- ambiguity workshops,
- architecture critiques,
- ADR discussions,
- AI-use reviews,
- pull request simulations,
- code review workshops,
- testing evidence reviews,
- release readiness table-tops,
- incident response simulations,
- postmortem analysis,
- final defense preparation.

Students should practice engineering, not only hear about engineering.

Relationship To Educational Operations Course design establishes classroom systems.

Those systems are later operated through the ETIS Classroom Facilitation engine.

```
Course Design
   ↓
Classroom Facilitation
   ↓
Continuous Improvement
```

Course design answers:

> What educational system should exist?

Classroom facilitation answers:

> How should instructors operate that system?

These educational engines intentionally work together.

Step 9 — Design Assessment

Assessment should evaluate both products and evidence.

Students should be assessed on:

- engineering intent,
- evidence quality,

- architecture reasoning,
- review participation,
- AI accountability,
- testing discipline,
- risk communication,
- release readiness,
- operational thinking,
- professional defense.

A functioning application alone is insufficient.

Grades measure performance.

Engineering maturity signals measure transformation.

Both are important.

Grades measure what students produced.

Maturity signals help instructors observe who students are becoming.

Course designs should intentionally create opportunities for instructors to observe engineering maturity throughout the semester.

Step 10 — Design The Final Defense

Every ETIS course should culminate in accountability.

Students should answer questions such as:

- What problem were you solving?
- Why was this architecture selected?
- How did AI assist?
- What evidence supports your claims?
- What risks remain?
- How would this system be operated?
- What would you improve?
- Why is this release defensible?

Final defense is one of the defining ETIS experiences.

Recommended Course Components

A mature ETIS course may include:

Course Introduction

Repository Foundation

Student Starter Kit

Reading Assignments

Engineering Phase Gates

Classroom Exercises

Reviews and Critiques

AI Governance

Release Readiness

Operational Thinking

Final Engineering Defense

Not every course needs every component.

The design should remain purposeful.

Common Course Design Mistakes

Mistake 1 — Designing Around Topics Topics are insufficient.

Design around professional transformation.

Mistake 2 — Overloading Students With Process Discipline is not bureaucracy.

Every artifact should teach responsibility.

Mistake 3 — Assessing Only Technical Output Assessment must include evidence and accountability.

Mistake 4 — Treating AI As Either Forbidden Or Unlimited AI should be governed.

Mistake 5 — Ignoring Operations Software does not end at deployment.

Mistake 6 — Copying COMP330 COMP330 is an implementation, not a universal template.

Mistake 7 — Forgetting The Final Defense Students should leave with experience defending engineering decisions.

Course Design Review Checklist

Before launching a course, the instructor should be able to answer:

Transformation

- Who should learners become?

Capabilities

- What engineering capabilities are primary?

Book Mapping

- Which ETIS chapters are required?

Repository

- What evidence will students create?

Phase Gates

- How will maturity increase?

AI Governance

- How will AI be governed?

Classroom Experiences

- How will students practice engineering?

Assessment

- How will accountability be measured?

Final Defense

- How will students demonstrate engineering maturity?

If these questions are unclear, the course design is incomplete.

Relationship To Educational Stewardship

Course design is the beginning of educational stewardship.

Educational systems should improve over time rather than restart every semester.

The ETIS Instructor Notes engine preserves educational wisdom gained while operating courses.

Course Design
↓
Classroom Facilitation
↓
Instructor Notes
↓
Continuous Improvement

Course design creates educational systems.

Classroom facilitation operates those systems.

Instructor notes preserve institutional memory and continuously improve those systems.

Educational systems are engineered.

Educational systems are also stewarded.

Stewardship Rules

Course design should preserve these boundaries:

- Do not duplicate shared assets.
- Do not duplicate the student starter kit.
- Do not duplicate the ETIS book.
- Do not treat COMP330 as the ecosystem.
- Do not create assignments without purpose.
- Do not create documentation theater.
- Do not teach AI without accountability.
- Do not teach software without operational responsibility.

The course should remain teachable, maintainable, and adaptable.

Guiding Standard

Every course component should answer at least one of these questions:

- What engineering responsibility is being taught?
- What evidence will students create?
- What judgment will students practice?
- What review will students survive?
- What risk will students learn to see?
- What professional habit will students carry forward?

If a component cannot answer one of those questions, it should be reconsidered.

Core Commitment

The ETIS Course Design Guide exists to help instructors build courses that form trustworthy engineers.

Students should leave the course understanding that software engineering is not merely software construction.

It is the disciplined practice of creating systems that can be understood, reviewed, governed, operated, improved, and trusted over time.

ETIS Adoption Planning Guide

The **ETIS Adoption Planning Guide** helps instructors, course designers, and educational leaders decide how to adopt *Engineering Trustworthy Intelligent Systems (ETIS)* within a specific course, program, institution, or professional training environment.

Adoption is not file copying.

Adoption is the responsible translation of ETIS doctrine into a teachable implementation while preserving educational purpose, architectural boundaries, provenance, evidence expectations, and long-term stewardship.

Purpose

The purpose of this guide is to help instructors plan an ETIS adoption before building course materials.

A successful ETIS adoption should be intentional, bounded, teachable, assessable, and maintainable.

This guide helps instructors decide:

- what kind of ETIS adoption is appropriate,
- which ETIS learning path should be emphasized,
- which parts of the ETIS book are primary,
- which shared assets should be consumed,
- which student starter kit should be used,
- which assignments should function as engineering phase gates,
- how AI use will be governed,
- how repository evidence will be evaluated,
- how release readiness will be demonstrated,
- and how the adoption will preserve alignment with ETIS over time.

Adoption Principle

ETIS adoption begins with one governing principle:

> Adopt ETIS to form trustworthy engineers, not to add more course documents.

An ETIS course should not become a paperwork exercise.

ETIS adoption also follows these principles:

> Institutions should inherit ETIS doctrine, not ETIS implementations.

> Standardize outcomes. Localize execution.

Local context is not architectural drift.

Institutions should preserve ETIS philosophy while adapting implementation to their own environments.

The purpose is to help learners practice disciplined engineering judgment through intent, context, evidence, review, governance, operation, and accountability.

Relationship to the ETIS Book

The ETIS book remains authoritative.

An adoption may emphasize different parts of the book depending on audience, course length, and learning goals, but the adoption should not contradict the book's doctrine.

ETIS Book

↓

Educational Architecture

↓

Instructor Course Package

↓

Course Implementation

↓

Student Engineering Practice

The course implementation adapts ETIS.

It does not replace ETIS.

Educational Engine Architecture

The Instructor Course Package now consists of six educational engines.

Educational Intent Engine

↓

Educational Sequencing Engine

↓

Educational Accountability Engine

↓

Educational Evaluation Engine

↓

Educational Operations Engine

↓

Educational Stewardship Engine

Institutional adoptions should consume these engines as an integrated educational system rather than as isolated directories.

Together they create a complete educational operating model.

Adoption Is Not Implementation Copying

The Loyola COMP 330 implementation is the flagship adoption example.

It is a reference model, not a universal template.

Instructors should study COMP 330 to understand how ETIS can be taught through repository-centered engineering, phase gates, AI-use governance, team accountability, review, release readiness, and final engineering defense.

However, adopters should not copy COMP 330 blindly.

Every course has its own constraints:

- audience,

- semester length,
- student maturity,
- class size,
- institutional policies,
- project expectations,
- technical prerequisites,
- assessment rules,
- calendar structure,
- available tools,
- instructor capacity,
- and local educational goals.

A good adoption preserves ETIS doctrine while adapting implementation details responsibly.

The governing principle should now be:

> Institutions should inherit ETIS doctrine, not ETIS implementations.

Standardize outcomes.

Localize execution.

Local context is not architectural drift.

The educational outcomes should remain stable while implementation mechanisms adapt to institutional realities.

Adoption Planning Questions

Before creating course materials, the instructor should answer the following questions.

1. What is the adoption context? Identify the instructional environment.

Examples include:

- undergraduate software engineering course,
- graduate software engineering course,
- senior capstone,
- AI governance module,
- software architecture module,
- DevOps or operational readiness module,
- professional training program,
- corporate engineering workshop,
- or organizational governance training.

The adoption context determines scope, depth, pacing, assessment, and artifact expectations.

2. What learner transformation is expected? Define the professional growth expected from learners.

Possible transformation goals include:

- student to responsible engineer,
- developer to reviewer,
- reviewer to architect,
- team member to release defender,
- builder to operational thinker,
- or practitioner to trustworthy engineering steward.

The adoption should be designed around transformation, not merely content coverage.

3. Which ETIS capabilities are primary? Select the primary capabilities the adoption will teach.

Examples include:

- requirements discipline,
- repository-centered engineering,
- team coordination,
- architecture reasoning,
- decision documentation,
- AI-use governance,
- implementation under review,
- testing and validation,
- release readiness,
- observability,
- operational readiness,
- incident response,
- postmortems,
- security governance,
- review boards,
- and stewardship.

A course does not need to teach everything equally.

A strong adoption is focused enough to be teachable.

4. Which parts of the ETIS book are primary? Map the adoption to the book.

A full-semester undergraduate software engineering course may emphasize Parts I and II while selectively introducing Parts III and IV.

A graduate architecture or governance course may place more emphasis on Parts III and IV.

A professional training module may focus on a small number of chapters tied to a specific engineering capability.

The adoption should make explicit which chapters are required, which are supporting, and which are optional.

5. What evidence will students produce? ETIS learning is repository-centered.

Students should produce evidence that another reviewer can inspect.

Evidence may include:

- requirements,
- use cases,
- planning records,
- risk registers,
- architecture notes,
- ADRs,
- AI-use logs,
- pull request reviews,
- test evidence,
- defect records,

- release readiness records,
- postmortems,
- operational readiness evidence,
- governance records,
- and final defense materials.

The adoption should define the evidence standard before assignments begin.

6. What phase gates will structure the course? ETIS assignments should function as engineering phase gates.

Possible phase gates include:

1. repository foundation,
2. requirements and planning review,
3. architecture review,
4. implementation and validation review,
5. Cycle 1 release readiness,
6. postmortem and stabilization review,
7. operational readiness review,
8. final release defense.

Each phase gate should increase system maturity and student accountability.

7. How will AI use be governed? An ETIS adoption should define AI-use expectations clearly.

The course should specify:

- what AI use is allowed,
- what AI use is prohibited,
- what AI use must be disclosed,
- how AI output must be reviewed,
- where AI-use evidence is recorded,
- how students verify AI-assisted work,
- and how responsibility remains with the human team.

The governing principle remains:

> AI-assisted work is not accepted engineering work until humans review, verify, and own it.

8. How will assessment work? Assessment should evaluate both product and evidence.

A working demo is not enough.

The instructor should define how students will be assessed on:

- engineering intent,
- evidence quality,
- technical execution,
- review participation,
- AI-use accountability,
- testing discipline,
- architecture reasoning,
- risk visibility,
- release readiness,
- operational thinking,

- and professional defense.

Rubrics should reward disciplined engineering behavior, not only visible functionality.

9. What starter environment will students use? The adoption should decide whether students will use an ETIS Student Starter Kit.

A starter kit should provide a structured repository environment that supports evidence creation, team coordination, AI-use governance, reviews, testing, and release readiness.

The starter kit should be introduced as an engineering environment, not as a folder full of templates.

10. How will the adoption be maintained? An ETIS adoption should be maintainable beyond one semester.

The instructor should decide:

- what materials are local to the course,
- what materials are reusable,
- what improvements should be proposed upstream,
- what evidence should be preserved,
- what lessons learned should be recorded,
- and how the next offering will improve without drifting from ETIS doctrine.

11. How will educational stewardship be preserved? Educational systems should become smarter every semester.

Institutions should define:

- how lessons learned will be captured,
- how student patterns will be documented,
- how course corrections will be preserved,
- how engineering maturity signals will be observed,
- and how future instructors will inherit institutional memory.

Educational memory is educational infrastructure.

Educational systems should improve over time rather than restart every semester.

Adoption Models

ETIS can be adopted at different scales.

Full Adoption ETIS becomes the primary framework for the course.

This model is appropriate for a full-semester software engineering course, capstone course, or graduate engineering course.

A full adoption usually includes:

- repository-centered project work,
- team-based engineering,
- staged phase gates,
- AI-use governance,
- architecture and review,
- two-cycle delivery,
- release readiness,

- operational maturity,
- and final engineering defense.

Partial Adoption ETIS provides selected modules, practices, or assessments within an existing course.

This model is appropriate when the instructor wants to introduce ETIS capabilities without redesigning the full course.

Examples include:

- AI-use governance module,
- release readiness module,
- architecture review module,
- operational readiness module,
- repository evidence module,
- or postmortem module.

Capstone Adoption ETIS provides structure for student teams building larger systems.

This model emphasizes:

- intent,
- team accountability,
- evidence,
- architecture,
- review,
- release readiness,
- stakeholder communication,
- operational maturity,
- and final defense.

Professional Adoption ETIS supports professional or organizational training.

This model may emphasize:

- AI-assisted engineering governance,
- review boards,
- release governance,
- operational trust,
- system stewardship,
- incident learning,
- and engineering accountability.

Recommended Adoption Sequence

A disciplined adoption should proceed in stages.

Step 1 — Define the Adoption Boundary State what the course will and will not attempt to teach.

Avoid trying to cover all ETIS capabilities at equal depth.

Step 2 — Select the Learning Path Choose the professional transformation path most appropriate for the learners.

Examples:

- student to responsible engineer,
- student to release defender,
- engineer to reviewer,
- engineer to operational thinker,
- or practitioner to trustworthy system steward.

Step 3 — Map the Book Identify required, supporting, and optional ETIS chapters.

This prevents the course from becoming either too broad or disconnected from the authoritative text.

Step 4 — Select Shared Assets Choose reusable assets from the ETIS educational ecosystem.

Do not copy or modify shared assets unnecessarily.

Consume them intentionally.

Step 5 — Select or Adapt the Student Starter Kit Decide what repository structure students will use.

Preserve repository-centered teaching.

Step 6 — Design Phase Gates Define the assignment sequence as maturity gates.

Each gate should produce reviewable evidence.

Step 7 — Define Assessment Create or select rubrics that evaluate product, process, evidence, review, AI governance, and release defensibility.

Step 8 — Plan Facilitation Decide how class time will support reviews, critiques, simulations, tabletops, workshops, and defense preparation.

Step 9 — Preserve Provenance Record where materials came from.

Do not erase Loyola provenance when using assets derived from COMP 330.

Step 10 — Capture Lessons Learned After the adoption, preserve what worked, what failed, what confused students, what improved learning, and what should change next time.

Step 11 — Contribute Stewardship Memory Determine what educational wisdom should flow back into the ecosystem.

Examples include:

- common student patterns,
- AI transition observations,
- maturity signals,
- institutional adaptations,
- course corrections,
- and reusable instructor guidance.

Every implementation should leave the system smarter than it found it.

Adoption Planning Template

Instructors may use the following planning structure.

Course or Program Name:

Institution or Organization:

Adoption Type: Full / Partial / Capstone / Professional / Module

Primary Audience:

Course Length:

Primary ETIS Capabilities:

Required ETIS Chapters:

Supporting ETIS Chapters:

Optional ETIS Chapters:

Student Starter Kit:

Shared Assets Consumed:

Major Phase Gates:

AI-Use Governance Model:

Assessment Model:

Repository Evidence Expectations:

Final Defense or Capstone Evidence:

Local Adaptations:

Materials Created Locally:

Materials Proposed for Upstream Contribution:

Known Risks:

Maintenance Plan:

This template should be completed before major course materials are finalized.

Common Adoption Mistakes

Mistake 1 — Treating ETIS as a Topic List ETIS is not a checklist of subjects.

It is an engineering formation model.

Mistake 2 — Copying COMP 330 Without Adaptation COMP 330 is a flagship implementation.

It is not a universal course shell.

Mistake 3 — Assigning Documents Without Engineering Purpose Every artifact should teach responsibility, evidence, judgment, review, risk awareness, or professional accountability.

Mistake 4 — Overloading Students With Process ETIS should create disciplined engineering behavior, not paperwork fatigue.

Mistake 5 — Ignoring AI Governance AI use must be taught as accountable engineering practice.

Mistake 6 — Assessing Only the Demo A demo shows that something appears to work.

It does not prove that the system is understandable, reviewable, governable, operable, improvable, or trustworthy.

Mistake 7 — Erasing Provenance If an asset originated in Loyola COMP 330, that provenance should remain visible.

Educational history is part of engineering memory.

Mistake 8 — Treating ETIS As A Loyola Product Loyola COMP330 is the flagship implementation.

It is not the educational standard.

Institutions should inherit ETIS doctrine, not copy Loyola.

Adoption Readiness Checklist

Before launching an ETIS adoption, the instructor should confirm:

- The adoption type is clear.
- The learning transformation goal is explicit.
- Required ETIS chapters are identified.
- Shared assets have been selected.
- Student repository expectations are defined.
- Assignment phase gates are sequenced.
- AI-use rules are documented.
- Assessment criteria are clear.
- Release readiness expectations are defined.
- Final defense expectations are defined.
- Institutional constraints have been considered.
- Local adaptations are documented.
- Provenance is preserved.
- Maintenance responsibilities are understood.

If these items are unclear, the adoption is not ready.

Stewardship Expectations

Every ETIS adoption should leave evidence.

After the course or training experience, instructors should preserve:

- what was taught,
- what assets were used,
- what adaptations were made,
- what students struggled with,
- what evidence quality looked like,
- what assessment criteria worked,
- what AI-use issues emerged,
- what should improve next time,
- and what might be reusable by future adopters.

Adoption is not complete when the course ends.

Adoption is complete when learning from the course has been captured for future stewardship.

Every semester should leave the system smarter than it found it.

Educational memory is educational infrastructure.

Educational stewardship should become part of normal course operations rather than an afterthought performed after the semester ends.

Institutions should preserve educational memory as part of adoption.

Educational memory may include:

- common student patterns,
- maturity signals,
- AI transition observations,
- difficult conversations,
- course corrections,
- and institutional adaptations.

Educational stewardship should become part of normal operations.

Promotion Back Into ETIS

Not every local course artifact should become a shared ETIS asset.

Promotion should occur only when an artifact demonstrates reuse value beyond its original course.

An artifact may be considered for promotion when it:

- supports more than one implementation,
- strengthens ETIS teaching capability,
- preserves clear provenance,
- has a defined consumer,
- has a stable purpose,
- avoids unnecessary local assumptions,
- and can be maintained over time.

Promotion requires review.

Premature generalization should be avoided.

Final Standard

An ETIS adoption is successful when students do more than complete assignments.

A successful adoption produces learners who can explain:

- what they intended,
- what they built,
- how they used AI,
- what they verified,
- what evidence supports their claims,
- what risks remain,
- why their release is defensible,
- how the system could be operated,
- and what they would improve next.

The goal is not adoption for its own sake.

The goal is the formation of trustworthy engineers.

Successful ETIS adoptions should simultaneously produce three outcomes:

- trustworthy engineers,
- reusable educational assets,
- and educational knowledge for future instructors.

Educational systems are engineered.

Educational systems are also stewarded.

ETIS Course Readiness Checklist

The **ETIS Course Readiness Checklist** is a pre-launch review document for instructors preparing to teach an ETIS-based course.

Its purpose is to ensure that course design decisions have been made before students begin major engineering work.

This checklist is not administrative paperwork.

It is a quality assurance review for educational systems.

A course is considered ready when an instructor can confidently answer the questions in this document.

Purpose

This checklist helps instructors verify that an ETIS course has been intentionally designed.

The checklist validates that:

- educational goals are clear,
- professional transformation goals are defined,
- ETIS book coverage is intentional,
- repositories are prepared,
- assignments are sequenced,
- AI governance is established,
- assessments are designed,
- and engineering accountability is visible.

The checklist should be completed before the first major student deliverable.

Course Information

Complete the following information.

Course Name:

Institution:

Department:

Instructor(s):

Term:

Course Format: Semester / Quarter / Module / Professional Training

Delivery Model: In-Person / Hybrid / Online

Primary Audience:

Expected Enrollment:

Project Type: Individual / Team-Based / Mixed

Section 1 — Professional Transformation

The course should be designed around who students will become.

Verify

[ ] The primary professional transformation goal has been defined.

[ ] The transformation has been communicated to students.

[ ] The course is designed around engineering maturity rather than topic coverage.

Primary Transformation Goal

Student

↓

____

Examples:

Student → Responsible Engineer

Student → Reviewer

Student → Release Defender

Student → Operational Thinker

Student → Future Trustworthy Engineer

Section 2 — ETIS Book Alignment

The ETIS book remains authoritative.

Verify

[ ] Required chapters have been identified.

[ ] Supporting chapters have been identified.

[ ] Optional chapters have been identified.

[ ] Book coverage matches course scope.

Record

Required Chapters:

Supporting Chapters:

Optional Chapters:

Section 3 — Engineering Capabilities

Select the primary capabilities students will practice.

Verify

[ ] Capabilities have been intentionally selected.

[ ] The course is not attempting to teach everything equally.

Primary Capabilities

Check all that apply.

[ ] Requirements Discipline

[ ] Planning

[ ] Estimation

[ ] Risk Management

[ ] Repository-Centered Engineering

[ ] Architecture Reasoning

[ ] Decision Documentation

[ ] AI Governance

[ ] Pull Requests and Reviews

[ ] Testing and Validation

[ ] Release Readiness

[ ] Operational Thinking

[ ] Observability

[ ] Security Governance

[ ] Incident Response

[ ] Postmortems

[ ] Stewardship

Section 4 — Repository Readiness

Students should receive a prepared engineering environment.

Verify

[ ] A Student Starter Kit has been selected.

[ ] Repository expectations are documented.

[ ] Evidence locations are defined.

[ ] Students understand that repositories are evidence systems.

Repository Areas

Verify expectations have been defined for:

[ ] Requirements

[ ] Planning

[ ] Architecture

[ ] Decisions

[ ] Reviews

[ ] Testing

[ ] Quality

[ ] Release

[ ] Operations

[ ] Governance

[ ] AI

[ ] Postmortems

Section 5 — Engineering Phase Gates

Assignments should function as maturity gates.

Verify

[ ] Phase gates have been defined.

[ ] Each phase gate increases engineering maturity.

[ ] Each phase gate produces evidence.

Planned Phase Gates

Phase Gate 1:

Phase Gate 2:

Phase Gate 3:

Phase Gate 4:

Phase Gate 5:

Phase Gate 6:

Phase Gate 7:

Phase Gate 8:

Section 6 — Two-Cycle Design

Verify

[ ] Cycle 1 activities have been defined.

[ ] Cycle 2 activities have been defined.

[ ] Students understand the difference.

Record

Cycle 1 — Can It Work? Primary Activities:

Cycle 2 — Can It Survive? Primary Activities:

Section 7 — AI Governance Readiness

AI use should be intentionally governed.

Verify

[ ] AI expectations are documented.

[ ] AI disclosure requirements are documented.

[ ] AI review expectations are documented.

[ ] AI ownership expectations are documented.

Record

Allowed AI Use:

Restricted AI Use:

Required Disclosures:

AI Evidence Location:

Section 8 — Classroom Experience Readiness

Students should actively practice engineering.

Verify

Check planned activities.

[ ] Requirements Review

[ ] Ambiguity Workshop

[ ] Architecture Critique

[ ] ADR Review

[ ] AI-Use Review

[ ] Pull Request Simulation

[ ] Code Review Workshop

[ ] Testing Evidence Review

[ ] Release Readiness Tabletop

[ ] Incident Response Tabletop

[ ] Postmortem Analysis

[ ] Final Defense Preparation

Section 9 — Assessment Readiness

Assessment should evaluate both product and evidence.

Verify

Assessment criteria exist for:

[ ] Engineering Intent

[ ] Evidence Quality

[ ] Technical Execution

[ ] Architecture Reasoning

[ ] Review Participation

[ ] AI Accountability

[ ] Testing Discipline

[ ] Risk Communication

[ ] Release Readiness

[ ] Operational Thinking

[ ] Professional Accountability

Section 10 — Final Defense Readiness

Every ETIS course should culminate in accountability.

Verify

Students will answer questions such as:

[ ] What problem were you solving?

[ ] Why was this architecture selected?

[ ] How did AI assist?

[ ] What evidence supports your claims?

[ ] What risks remain?

[ ] How would this system be operated?

[ ] What would you improve?

[ ] Why is this release defensible?

Section 11 — Local Adaptation Review

Every implementation will have local constraints.

Record

Institutional Constraints:

Calendar Constraints:

Technology Constraints:

Assessment Constraints:

Local Adaptations:

Section 12 — Stewardship Review

The course should be maintainable over time.

Verify

[ ] Reusable assets are separated from local assets.

[ ] Loyola provenance has been preserved.

[ ] Shared assets have not been duplicated.

[ ] COMP330 has not become the ecosystem.

[ ] Lessons learned will be captured after the course.

Final Readiness Decision

Complete this review before launching the course.

[ ] Ready To Launch

[ ] Mostly Ready

[ ] Significant Work Needed

Instructor Confidence Review

An instructor should be able to answer these questions confidently.

What engineering responsibilities are students learning?

What evidence will students create?

How will AI be governed?

How will engineering maturity increase?

How will students practice engineering?

How will students defend engineering decisions?

What professional habits should students carry forward?

If these answers are unclear, the course is not ready.

Guiding Principle

The purpose of ETIS is not to create more assignments.

The purpose of ETIS is to create engineers who can define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

This checklist exists to ensure the course itself is engineered responsibly before students are asked to engineer responsibly.

Part II

Syllabus Guidance

ETIS Academic Integrity Guidance

The **ETIS Academic Integrity Guidance** document provides reusable syllabus language and educational principles for teaching honesty, transparency, accountability, and professional ethics within ETIS-based courses.

This document is not intended to replace institutional academic integrity policies.

Instead, it helps instructors communicate how academic integrity applies within repository-centered, AI-assisted, team-based engineering environments.

ETIS teaches integrity as a professional engineering responsibility rather than solely a compliance requirement.

Purpose

The purpose of this document is to help instructors establish clear expectations for academic integrity within ETIS implementations.

Academic integrity should support trust.

Students should understand that integrity is not simply about avoiding misconduct.

Integrity is part of engineering itself.

Trustworthy systems cannot be built without trustworthy engineering behavior.

Students should learn that engineering trust begins with honest engineering practices.

Educational Philosophy

ETIS teaches that engineering accountability depends upon transparency.

Students should practice:

- honesty,
- ownership,
- attribution,
- transparency,
- evidence preservation,
- risk communication,
- and professional responsibility.

Integrity is not a separate activity.

Integrity is embedded throughout the engineering lifecycle.

The central principle is:

> Trustworthy systems require trustworthy engineering behavior.

Relationship To AI Governance

Academic integrity and AI governance are related but distinct concepts.

Academic integrity asks:

> Was the work represented honestly?

AI governance asks:

Was AI used responsibly?

Both are necessary.

Students should understand that responsible AI use does not automatically satisfy academic integrity expectations.

Likewise, avoiding AI does not automatically demonstrate integrity.

The two disciplines work together.

Core Integrity Principles

Every ETIS implementation should communicate the following principles.

Principle 1 — Students Own Their Work

Students remain responsible for all submitted work.

Responsibility cannot be delegated to teammates, AI systems, external services, or online resources.

Principle 2 — Engineering Work Should Be Transparent

Important contributions should be visible.

Students should avoid obscuring how work was performed.

Transparency strengthens trust.

Principle 3 — Attribution Matters

Students should appropriately acknowledge external contributions.

Sources, references, and meaningful external assistance should be disclosed.

Principle 4 — Evidence Should Be Honest

Engineering evidence should accurately reflect engineering work.

Fabricated evidence undermines trust.

Principle 5 — Professional Integrity Matters

Engineering behavior should reflect professional standards expected in real engineering environments.

Recommended Syllabus Statement

The following language may be used directly in a syllabus.

Standard Academic Integrity Statement

Academic integrity is a foundational expectation of this course.

Students are expected to complete engineering work honestly, transparently, and professionally. Engineering artifacts, repository evidence, reviews, testing activities, AI disclosures, and team contributions should accurately represent the work performed.

Students remain accountable for all submitted work regardless of whether teammates, AI tools, online resources, or external references were used.

Trustworthy systems require trustworthy engineering behavior.

Academic integrity violations undermine both learning and professional accountability.

Institutional academic integrity policies remain fully applicable throughout this course.

Short Academic Integrity Statement

The following language may be used in abbreviated syllabi.

Students are expected to conduct themselves honestly, transparently, and professionally. All engineering artifacts, evidence, reviews, AI disclosures, and contributions should accurately represent the work performed.

First-Day Student Statement

The following language may be used during course introductions.

Integrity is not simply about avoiding cheating.

Integrity is about whether another engineer can trust your work.

Throughout this course you will create requirements, architecture decisions, reviews, tests, AI disclosures, release evidence, and final defenses.

Every one of those activities depends on honesty.

The engineering profession depends on trust.

That trust begins with how you represent your own work.

Integrity Expectations For Repository-Centered Engineering

Repositories should accurately reflect engineering activity.

Students should preserve honest records of:

- engineering decisions,
- reviews,
- testing,
- AI use,
- risks,
- contributions,
- and evidence.

Repositories should never be manipulated to create a false appearance of engineering maturity.

Transparency matters more than perfection.

Team Integrity Expectations

ETIS frequently uses team-based engineering.

Teams should:

- communicate honestly,
- share responsibilities appropriately,
- review work collaboratively,
- disclose concerns early,
- and participate professionally.

Students should avoid hiding behind team structures.

Every student remains individually accountable.

Attribution Expectations

Students should acknowledge meaningful external assistance.

Examples include:

- books,
- articles,
- online references,
- AI systems,
- open-source projects,
- tutorials,
- and other educational resources.

Attribution demonstrates professionalism.

It does not diminish student ownership.

Repository Ownership Expectations

Students should understand that repositories are engineering records.

Repository history should accurately represent engineering work.

Students should avoid:

- obscuring ownership,
- disguising contributions,
- hiding changes,
- or misrepresenting activity.

Engineering history matters.

AI And Integrity

AI is permitted when used according to course policies.

However, AI use should remain transparent.

Students should disclose meaningful AI use appropriately.

Students should not:

- conceal meaningful AI involvement,
- present AI output as unaided work when disclosure is required,
- fabricate AI disclosures,
- or transfer responsibility to AI systems.

AI does not remove accountability.

Examples Of Integrity Violations

The following behaviors violate ETIS integrity expectations.

Fabricating Evidence

Examples:

- fabricated testing,
- fabricated review records,
- fabricated approvals,
- fabricated stakeholder feedback,
- fabricated meeting participation.

Misrepresenting Contributions

Examples:

- claiming work performed by others,
- hiding team participation,
- disguising authorship,
- obscuring ownership.

Concealing Meaningful AI Use

Examples:

- intentionally hiding required AI disclosures,
- misrepresenting AI involvement.

Misrepresenting Repository History

Examples:

- creating false engineering activity,
- falsifying review participation,
- manipulating evidence to create a misleading narrative.

Misrepresenting Risks Or Limitations

Examples:

- hiding known defects,
- concealing known limitations,
- overstating confidence.

Honest risk communication is part of integrity.

Professional Integrity Behaviors

Students should practice behaviors such as:

- asking for help appropriately,
- acknowledging uncertainty,
- documenting limitations,
- communicating risks,
- correcting mistakes,
- reviewing work honestly,
- and accepting feedback professionally.

Integrity is often demonstrated through ordinary engineering behaviors.

Integrity And Assessment

Assessment should reward honest engineering.

Students should not be penalized for:

- identifying limitations,
- communicating uncertainty,
- documenting risks,
- or admitting mistakes.

Trust grows when students feel safe being honest.

Concealment should never become the easiest option.

Common Student Misconceptions

Misconception 1 — Integrity Means Never Making Mistakes

Mistakes are expected.

Concealment is the problem.

Misconception 2 — Attribution Reduces Credit

Attribution demonstrates professionalism.

Misconception 3 — AI Makes Integrity Irrelevant

AI increases the need for integrity.

Misconception 4 — Team Membership Eliminates Individual Responsibility

Every student remains accountable.

Misconception 5 — A Working Demo Is Enough

Engineering claims require evidence.

Common Instructor Mistakes

Mistake 1 — Combining Academic Integrity And AI Governance Into One Policy

They are related but distinct.

Mistake 2 — Making Integrity Entirely Punitive

Integrity should also be educational.

Mistake 3 — Focusing Only On Plagiarism

Repository-centered engineering introduces additional integrity responsibilities.

Mistake 4 — Ignoring Risk Communication

Honesty includes communicating limitations.

Mistake 5 — Rewarding Appearance Over Transparency

Students should not feel pressure to hide problems.

Adaptation Rules

Institutions may adapt this guidance.

However, adaptations should preserve these principles:

- honesty matters,
- ownership matters,
- attribution matters,
- transparency matters,
- evidence matters,
- risk communication matters,
- and accountability matters.

These principles should remain visible.

Integrity Reflection Questions

Students should periodically ask themselves:

- Does this accurately represent the work performed?
- Could I explain how this was created?
- Have I acknowledged meaningful assistance?
- Am I hiding anything important?
- Have I communicated risks honestly?
- Would another engineer trust this representation?
- Am I willing to defend this work publicly?

If these answers are uncomfortable, integrity should be reevaluated.

Guiding Standard

Integrity within ETIS can be summarized with one question:

> If another engineer reviewed my work, would they trust the story my evidence tells?

That story should be honest, transparent, and defensible.

Core Commitment

The purpose of academic integrity within ETIS is not to catch students making mistakes.

The purpose is to help students become professionals who understand that trust is earned through honest engineering behavior.

Trustworthy systems cannot be separated from trustworthy engineers.

Integrity is one of the foundations upon which trustworthy intelligent systems are built.

ETIS AI Use Policy Guidance

The **ETIS AI Use Policy Guidance** document provides reusable syllabus language and governance principles for teaching AI-assisted engineering within ETIS-based courses.

This document is not intended to replace institutional AI policies.

Instead, it provides guidance that instructors may adopt, adapt, and integrate into their own syllabi while preserving ETIS educational philosophy.

The goal is to teach AI as an engineering capability under human accountability.

Purpose

The purpose of this document is to help instructors communicate clear expectations regarding AI use.

ETIS does not teach students to avoid AI.

ETIS teaches students to govern AI.

Students should learn that AI can accelerate engineering activities while simultaneously increasing the need for human engineering discipline.

AI assistance does not remove accountability.

It increases accountability.

The central ETIS principle remains:

> AI-assisted work is not accepted engineering work until humans review, verify, and own it.

Educational Philosophy

AI is now part of modern engineering practice.

Students should learn how to use AI responsibly rather than learning how to avoid it.

ETIS teaches that AI is an engineering capability, not an engineering replacement.

Students remain responsible for:

- defining intent,
- engineering context,
- bounding authority,
- verifying behavior,
- explaining decisions,
- communicating risks,
- and owning outcomes.

Human accountability remains constant regardless of how much AI assistance was used.

AI Governance Principles

Every ETIS course should communicate the following principles.

Principle 1 — AI Is Allowed

AI is an accepted engineering capability within ETIS.

Responsible use is encouraged.

Students should learn how AI can augment engineering work.

Principle 2 — AI Is Not A Substitute For Engineering

AI may accelerate work.

AI does not replace engineering judgment.

Students remain responsible for submitted work.

Principle 3 — AI Use Should Be Transparent

Meaningful AI use should be disclosed.

Hidden AI use undermines accountability.

Transparency strengthens trust.

Principle 4 — AI Output Must Be Verified

AI output is a proposal.

Verification remains a human responsibility.

Students should assume AI outputs may be incomplete, incorrect, insecure, outdated, or contextually inappropriate.

Principle 5 — Human Ownership Is Non-Transferable

Students cannot transfer responsibility to AI.

Students own all submitted work.

Recommended Syllabus Statement

The following statement may be used directly in a syllabus.

Standard AI Use Statement

Artificial intelligence tools are permitted in this course when used responsibly and transparently.

AI should be treated as an engineering capability that assists human work rather than replaces engineering responsibility.

Students remain accountable for all submitted work regardless of how much AI assistance was used.

Meaningful AI use should be disclosed, reviewed, verified, and owned by the student or team.

AI-generated output should never be accepted without human evaluation.

The governing principle for this course is:

> AI-assisted work is not accepted engineering work until humans review, verify, and own it.

Short AI Use Statement

The following language may be used in abbreviated syllabi.

AI use is permitted in this course when used responsibly, transparently, and under human accountability. Students remain responsible for all submitted work and must review, verify, and own AI-assisted outputs.

First-Day Student Statement

The following language may be used during course introductions.

You are not being asked to avoid AI.

You are being asked to learn how to govern AI.

The engineering profession is changing rapidly, and AI is becoming a normal engineering capability.

However, engineering responsibility has not changed.

Your responsibility is to decide when AI should be used, how AI should be used, what risks AI introduces, how AI output should be verified, and how AI use should be disclosed.

You remain accountable for every engineering decision you submit.

Recommended AI Usage Categories

Instructors may classify AI activities into categories.

These categories provide clarity for students.

Category A — Generally Encouraged

AI may be used to support:

- brainstorming,
- requirements exploration,
- ambiguity identification,
- planning,
- architecture critique,
- alternative generation,
- documentation drafting,
- test idea generation,
- defect exploration,
- operational reasoning,
- and learning support.

These activities still require human review.

Category B — Allowed With Disclosure

AI may assist with:

- code generation,
- code explanations,
- test generation,
- documentation generation,
- review preparation,
- refactoring suggestions,
- architectural alternatives,
- and implementation acceleration.

Students should disclose meaningful use.

Students remain responsible for verification.

Category C — Restricted Activities

Institutions or instructors may restrict AI use for:

- individual assessments,
- examinations,
- competency demonstrations,
- foundational exercises,
- and skill-building activities.

Restrictions should be communicated clearly.

Category D — Prohibited Activities

AI should never be used to:

- impersonate another student,
- fabricate evidence,
- fabricate testing,
- fabricate reviews,
- fabricate approvals,
- fabricate repository history,
- fabricate meeting participation,
- or conceal meaningful AI involvement.

Concealed AI use is a trust violation.

AI Disclosure Expectations

Meaningful AI use should be disclosed.

Disclosure does not require documenting every prompt.

Disclosure should communicate engineering relevance.

Students should explain:

- what AI was used for,
- why AI was used,
- how outputs were verified,
- what changes were made,
- and what risks were considered.

The goal is accountability rather than surveillance.

Example AI Disclosure Language

The following language may be used by students.

Short Disclosure Example

AI was used to explore architecture alternatives and generate an initial implementation draft. Team members reviewed, modified, tested, and verified all resulting artifacts before acceptance.

Expanded Disclosure Example

AI was used to generate alternative implementation approaches and testing ideas. Team members reviewed outputs for correctness, security, maintainability, and architectural fit. AI-generated material was modified prior to integration. Final responsibility remained with the team.

AI Use Logs

Some ETIS implementations may require AI-use logs.

An AI-use log should not become busywork.

Its purpose is to preserve engineering accountability.

AI-use logs may capture:

- activity performed,
- AI capability used,
- purpose,
- verification activities,
- modifications made,
- and lessons learned.

The objective is transparency.

The objective is not surveillance.

AI Risk Awareness

Students should understand that AI introduces engineering risks.

Risks include:

- incorrect information,
- hallucinations,
- outdated recommendations,
- security vulnerabilities,
- architecture mismatches,
- hidden assumptions,
- poor maintainability,
- excessive complexity,
- missing context,
- and overconfidence.

Students should learn to actively look for these risks.

AI Verification Expectations

Students should verify AI-assisted outputs before acceptance.

Verification activities may include:

- reviewing correctness,
- checking architectural alignment,
- validating requirements fit,
- testing outputs,
- evaluating security implications,
- evaluating maintainability,
- checking dependencies,
- and reviewing assumptions.

Verification is an engineering responsibility.

AI And Team Accountability

AI should not replace collaboration.

Students should continue to:

- discuss alternatives,
- review work together,
- challenge assumptions,
- explain decisions,
- communicate risks,
- and build shared understanding.

Teams remain responsible for outcomes.

AI is a contributor, not a teammate.

AI And Academic Integrity

AI governance and academic integrity are related but different.

Academic integrity addresses honesty.

AI governance addresses responsible engineering.

Both are required.

Institutions may layer local academic integrity policies on top of ETIS guidance.

AI Policy Adaptation Rules

Institutions may adapt this guidance.

However, adaptations should preserve these principles:

- AI may be used responsibly.
- Human accountability remains constant.
- Meaningful AI use should be transparent.
- AI output requires verification.
- Students own outcomes.
- Trust depends on evidence.
- AI governance is part of engineering.

These principles should remain visible.

Common Instructor Mistakes

Mistake 1 — Banning AI Completely

Students should learn how to govern AI.

Avoiding AI entirely does not prepare students for modern engineering environments.

Mistake 2 — Allowing Unlimited AI Use

AI without boundaries undermines engineering accountability.

Mistake 3 — Treating AI As Academic Misconduct By Default

Responsible AI use should be taught.

Misuse should be addressed separately.

Mistake 4 — Focusing Only On Disclosure

Disclosure alone is insufficient.

Students must verify and own outcomes.

Mistake 5 — Turning AI Governance Into Surveillance

The objective is professional accountability.

Not prompt policing.

Guiding Standard

Students should be able to answer these questions:

- Why did I use AI?
- What did AI help produce?
- How did I verify it?
- What risks did AI introduce?
- What did I change?
- Why do I trust this output?
- Can I defend this work without AI present?
- Am I willing to own the outcome?

If students cannot answer these questions, AI has not been governed responsibly.

Core Commitment

The purpose of ETIS AI governance is not to control AI.

The purpose is to teach engineers how to work responsibly in a world where AI capabilities continue to expand.

The future of software engineering will not be determined by who uses AI.

It will be determined by who can use AI while preserving trust.

Trustworthy engineers define intent, engineer context, bound authority, verify behavior, explain decisions, and own outcomes.

AI changes the tools.

It does not change the responsibility.

ETIS Course Description Library

The **ETIS Course Description Library** provides reusable course description language for instructors adopting *Engineering Trustworthy Intelligent Systems (ETIS)*.

These descriptions are intended to help instructors describe ETIS-based learning experiences in syllabi, learning management systems, course proposals, program documentation, and professional training materials.

This file does not prescribe one course.

It provides adaptable language for different instructional contexts while preserving the ETIS educational philosophy.

Purpose

The purpose of this library is to help instructors communicate ETIS clearly, professionally, and consistently.

A strong ETIS course description should explain that the course is not only about building software.

It should communicate that students will practice disciplined engineering through intent, context, evidence, review, AI governance, release readiness, operational thinking, and professional accountability.

Course descriptions may be adapted for local institutional requirements, but they should preserve the core ETIS message:

> AI can produce artifacts. Engineers create trust.

How To Use This Library

Instructors should select the course description that best matches their instructional context and adapt it as needed.

Descriptions may be used for:

- syllabus course descriptions,
- catalog proposals,
- LMS course introductions,
- program documentation,
- course redesign materials,
- adoption planning,
- professional development offerings,
- and training announcements.

When adapting a description, preserve the relationship between software engineering, AI-assisted work, evidence, review, governance, and long-term trust.

Core ETIS Description

The following description may be used as a general-purpose ETIS course description.

General ETIS Course Description This course introduces *Engineering Trustworthy Intelligent Systems (ETIS)* as a disciplined approach to software engineering in the AI era. Students learn to build systems that are not only functional, but understandable, reviewable, governable, operable, improvable, and trustworthy over time. The course emphasizes repository-centered engineering, requirements discipline, architecture reasoning, AI-assisted development under human accountability, testing and validation, release readiness, operational thinking, and engineering stewardship. Students practice producing evidence of intent, decisions, verification, risks, AI use, and professional responsibility. The central goal is to help learners move beyond building working software toward becoming engineers who can define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

Undergraduate Software Engineering Course

Standard Description This course teaches software engineering as a professional discipline for building trustworthy systems in the AI era. Students work in teams to plan, design, implement, review, test, and defend a software system using repository-centered engineering practices. The course emphasizes requirements, planning, architecture, implementation under review, AI-use governance, testing, release readiness, operational thinking, and professional accountability. Students learn that a working demo is not enough; trustworthy engineering requires evidence that a system can be understood, reviewed, governed, operated, improved, and responsibly released.

Short Description This course teaches software engineering through the ETIS framework, emphasizing team-based development, repository evidence, AI-use governance, review, testing, release readiness, and professional accountability. Students learn to build systems that are not merely functional, but understandable, reviewable, governable, operable, and trustworthy.

Catalog-Style Description An introduction to software engineering as a disciplined, evidence-centered, AI-aware professional practice. Topics include requirements, planning, architecture, implementation, testing, review, AI-assisted development governance, release readiness, operational responsibility, and engineering stewardship. Students work in teams and use a repository as the authoritative record of engineering intent, decisions, evidence, and accountability.

Graduate Software Engineering Course

Standard Description This graduate course examines software engineering as the discipline of creating systems that can be trusted, governed, operated, and improved over time. Using the ETIS framework, students study advanced topics in architecture, evidence-centered engineering, AI-assisted development, reviewability, governance, operational readiness, release accountability, and engineering stewardship. The course emphasizes professional judgment, system boundaries, decision records, verification, risk communication, and long-term ownership. Students are expected to critique, design, evaluate, and defend engineering approaches suitable for complex socio-technical systems in the AI era.

Short Description This course uses ETIS to examine advanced software engineering through architecture, governance, AI accountability, operational trust, evidence, review, and stewardship. Students learn to evaluate and defend systems beyond functional correctness.

Catalog-Style Description Advanced study of software engineering as a disciplined approach to trustworthy system design, construction, governance, operation, and stewardship. Emphasis is placed on architecture, evidence, AI-assisted engineering, review systems, operational readiness, release governance, risk, accountability, and professional judgment.

Senior Capstone Course

Standard Description This capstone course uses the ETIS framework to guide student teams through the disciplined design, construction, review, validation, and defense of a software system. Students produce repository evidence of requirements, plans, architecture decisions, implementation, testing, reviews, AI use, risks, release readiness, and operational considerations. The course emphasizes professional accountability, team coordination, stakeholder communication, and final engineering defense.

Students are expected to demonstrate not only what they built, but why it is understandable, reviewable, governable, operable, improvable, and defensible.

Short Description This capstone course uses ETIS to structure team-based software engineering through repository evidence, AI governance, reviews, release readiness, operational thinking, and final engineering defense.

Catalog-Style Description A culminating team-based software engineering experience using the ETIS framework. Students plan, design, implement, review, test, and defend a software system while producing evidence of engineering intent, decisions, validation, AI use, risks, and release readiness.

AI Governance Course Or Module

Standard Description This course introduces AI governance as an engineering responsibility. Using the ETIS framework, students examine how AI-assisted work should be bounded, reviewed, disclosed, verified, and governed within software engineering environments. The course emphasizes human accountability, context control, evidence creation, AI-use logs, review practices, risk communication, and trustworthy system stewardship. Students learn that AI output is not accepted engineering work until humans review, verify, and own it.

Short Description This course uses ETIS to teach AI governance as accountable engineering practice, emphasizing disclosure, verification, human oversight, context control, review, and evidence.

Catalog-Style Description Study of AI governance within software engineering practice. Topics include human oversight, accountability, AI-use disclosure, verification, context control, evidence generation, reviewability, risk, and trustworthy system stewardship.

Software Architecture Course Or Module

Standard Description This course uses ETIS to teach software architecture as a governance and trust responsibility. Students learn how architecture establishes boundaries, manages dependencies, controls context, supports review, enables operational readiness, and creates conditions for trustworthy system evolution. The course emphasizes architectural reasoning, decision records, tradeoff analysis, AI-assisted design critique, governance points, evidence, and long-term maintainability. Students practice explaining and defending architecture decisions in terms of trust, accountability, operability, and future change.

Short Description This course applies ETIS to software architecture, emphasizing boundaries, dependencies, decision records, governance, reviewability, operational readiness, and trustworthy system evolution.

Catalog-Style Description Study of software architecture through the ETIS framework, with emphasis on system boundaries, dependencies, architectural decisions, governance points, reviewability, operational readiness, maintainability, and long-term trust.

DevOps And Operational Readiness Course Or Module

Standard Description This course uses ETIS to teach operational readiness as a core software engineering responsibility. Students learn that engineering does not end when code runs. The course emphasizes release readiness, observability, runbooks, incident response, postmortems, reliability, operational evidence, deployment accountability, and continuous improvement. Students practice evaluating whether a system can be operated, monitored, supported, explained, and improved after release.

Short Description This course applies ETIS to DevOps and operational readiness, emphasizing release evidence, observability, runbooks, incident response, postmortems, reliability, and operational accountability.

Catalog-Style Description Study of operational readiness and DevOps through the ETIS framework. Topics include release readiness, observability, runbooks, incident response, postmortems, reliability, operational evidence, and engineering accountability after deployment.

AI-Assisted Software Engineering Course Or Module

Standard Description This course examines AI-assisted software engineering through the ETIS framework. Students learn how AI tools can support requirements exploration, planning, architecture critique, implementation, testing, documentation, review preparation, and operational reasoning. The course emphasizes that AI acceleration increases the need for human engineering discipline. Students practice governing AI-assisted work through disclosure, verification, review, evidence, security awareness, maintainability checks, and professional ownership.

Short Description This course teaches AI-assisted software engineering as disciplined, accountable practice. Students use ETIS to govern AI-supported work through disclosure, verification, review, evidence, and human ownership.

Catalog-Style Description Study of AI-assisted software engineering using the ETIS framework. Topics include AI-supported requirements, design, implementation, testing, documentation, review, verification, governance, disclosure, risk, and professional accountability.

Software Engineering Professional Practice Course

Standard Description This course teaches software engineering as professional practice rather than isolated technical production. Using ETIS, students examine how engineers define intent, coordinate teams, document decisions, use AI responsibly, create evidence, review work, communicate risks, prepare releases, and accept accountability for outcomes. The course emphasizes habits of professional engineering judgment that transfer across tools, projects, organizations, and technologies.

Short Description This course uses ETIS to teach professional software engineering practice through evidence, review, AI accountability, team coordination, release readiness, risk communication, and stewardship.

Catalog-Style Description Study of software engineering as professional practice, emphasizing engineering judgment, team coordination, evidence, review, AI governance, risk communication, release readiness, and professional accountability.

Software Engineering Methods Course

Standard Description This course introduces software engineering methods through the ETIS framework. Students learn how engineering teams move from intent to implementation through requirements, planning, architecture, decisions, reviews, testing, release evidence, and operational readiness. The course emphasizes disciplined methods that help teams produce systems that are understandable, reviewable, governable, operable, and improvable over time. AI-assisted work is treated as a governed engineering capability requiring human verification and ownership.

Short Description This course teaches software engineering methods using ETIS, emphasizing requirements, planning, architecture, reviews, testing, AI governance, release evidence, and operational readiness.

Catalog-Style Description Introduction to software engineering methods using the ETIS framework. Topics include requirements, planning, architecture, decision documentation, reviews, testing, AI-assisted engineering, release readiness, operational readiness, and evidence-centered practice.

Engineering Review And Release Readiness Course Or Module

Standard Description This course uses ETIS to teach engineering review and release readiness as professional accountability practices. Students learn how to evaluate whether a system is ready to be reviewed, released, operated, and defended. The course emphasizes repository evidence, pull request discipline, architecture review, testing evidence, AI-use review, risk review, release readiness records, operational readiness, and final engineering defense. Students practice judging whether claims about a system are supported by evidence.

Short Description This course applies ETIS to engineering review and release readiness, emphasizing evidence, risk, review, AI-use accountability, release records, operational readiness, and final defense.

Catalog-Style Description Study of engineering review and release readiness using the ETIS framework. Topics include repository evidence, pull requests, architecture review, testing evidence, AI-use review, risk, release records, operational readiness, and defense of engineering claims.

Professional Training Program

Standard Description This professional training program introduces ETIS as a practical framework for engineering trustworthy intelligent systems in organizational environments. Participants learn how to govern AI-assisted engineering, preserve evidence, review system behavior, manage risk, prepare release decisions, improve operational readiness, and strengthen accountability across the engineering lifecycle. The program is designed for teams, technical leaders, architects, reviewers, governance stakeholders, and professionals responsible for systems that must be trusted over time.

Short Description This professional training program uses ETIS to help teams govern AI-assisted engineering, improve review practices, preserve evidence, strengthen release readiness, and build trustworthy systems.

Program Description A professional development program focused on trustworthy engineering in the AI era. Participants apply ETIS practices to AI governance, repository evidence, review systems, release readiness, operational trust, risk communication, and long-term system stewardship.

Corporate Engineering Workshop

Standard Description This workshop introduces ETIS as a practical model for improving engineering accountability in AI-assisted software development. Participants examine how teams can define intent, control context, review AI-supported work, preserve evidence, evaluate release readiness, communicate risk, and improve operational trust. The workshop emphasizes immediately usable practices for engineering teams, technical leads, reviewers, architects, and governance stakeholders.

Short Description This workshop uses ETIS to help engineering teams improve AI accountability, evidence practices, review quality, release readiness, risk communication, and operational trust.

Workshop Description A practical workshop for teams adopting AI-assisted engineering practices. Topics include intent definition, context control, AI-use review, evidence generation, release readiness, risk communication, operational trust, and stewardship.

Executive Or Leadership Briefing

Standard Description This briefing introduces ETIS as a governance and engineering framework for trustworthy intelligent systems. Leaders examine why AI-assisted development increases the need for disciplined engineering, reviewable evidence, bounded authority, operational readiness, and accountable release decisions. The briefing is designed for leaders responsible for engineering strategy, AI adoption, risk governance, operational trust, and long-term system stewardship.

Short Description This briefing introduces ETIS to leaders as a framework for AI-era engineering governance, evidence, reviewability, release accountability, operational trust, and stewardship.

Briefing Description An executive-level introduction to ETIS as a framework for governing trustworthy intelligent systems. Emphasis is placed on AI accountability, engineering evidence, review systems, release governance, operational readiness, risk, and organizational stewardship.

First-Day Student-Facing Description

The following description may be used in a syllabus, LMS welcome page, or first-day course overview.

This course is about learning to think and act like a software engineer in the AI era.

You will use modern tools, including AI where appropriate, but tools do not replace engineering responsibility. You will work with a team, maintain a repository, make decisions, review work, test claims, disclose meaningful AI use, communicate risks, prepare release evidence, and defend your engineering choices.

The goal is not only to build something that works.

The goal is to build something that another person can understand, review, govern, operate, improve, and trust.

Instructor-Facing Adoption Description

The following description may be used in adoption proposals or instructor planning documents.

This ETIS-based course uses the *Engineering Trustworthy Intelligent Systems* framework to teach software engineering as a disciplined, evidence-centered, AI-aware professional practice. The course is designed around student transformation rather than topic coverage alone. Learners practice requirements discipline, architecture reasoning, AI-use governance, review participation, repository evidence creation, release readiness, operational thinking, and professional defense. The course can be adapted for undergraduate, graduate, capstone, modular, professional, or organizational training contexts while preserving the core ETIS doctrine that trustworthy systems require accountable human engineering.

Program-Level Description

The following description may be used by departments or programs describing ETIS adoption across multiple courses.

ETIS provides a program-level framework for teaching trustworthy engineering in the AI era. Across courses, students learn that modern software engineering involves more than implementation. They practice defining intent, engineering context, governing AI-assisted work, producing evidence, reviewing decisions, validating behavior, preparing releases, reasoning about operations, communicating risk, and accepting accountability for outcomes. ETIS supports a progression from student work toward professional engineering judgment and long-term system stewardship.

Description Selection Guidance

Use the following guidance when selecting a description.

Use Undergraduate Software Engineering When ETIS is the primary framework for a team-based undergraduate software engineering course.

Use Graduate Software Engineering When the course emphasizes architecture, governance, operational trust, AI delegation, reviewability, and professional judgment.

Use Capstone When students are building substantial projects and must defend engineering decisions, evidence, risks, and release readiness.

Use AI Governance When the course or module focuses primarily on human oversight, AI-use disclosure, bounded authority, accountability, and verification.

Use Software Architecture When the primary emphasis is boundaries, dependencies, decisions, governance points, and long-term maintainability.

Use DevOps And Operational Readiness When the primary emphasis is release readiness, observability, operations, postmortems, reliability, and supportability.

Use Professional Training When teaching working professionals, engineering teams, leaders, reviewers, architects, or governance stakeholders.

Adaptation Rules

Instructors may adapt these descriptions to local needs.

However, adapted descriptions should preserve the following principles:

- ETIS teaches engineering trust, not tool usage alone.
- AI assistance increases the need for engineering discipline.
- Repository evidence matters.
- Reviewability matters.
- Governance matters.
- Operational readiness matters.
- A demo is not operational proof.
- Students remain accountable for outcomes.
- The ETIS book remains authoritative.

Final Standard

A strong ETIS course description should make clear that the course forms engineers.

It should not sound like a generic software engineering course with AI added.

It should communicate that students will learn to build systems that can be understood, reviewed, governed, operated, improved, and trusted over time.

ETIS Learning Outcomes Library

The **ETIS Learning Outcomes Library** provides reusable learning outcomes that instructors may adopt, adapt, and assemble into ETIS-based courses.

These outcomes are intentionally organized around engineering responsibilities rather than isolated technical topics.

The goal of ETIS education is not simply to transfer information.

The goal is to help learners develop engineering judgment, accountability, operational awareness, and professional responsibility.

Learning outcomes should describe what learners can do, explain, evaluate, defend, and own.

Purpose

This library helps instructors build courses that develop trustworthy engineers.

The outcomes in this document may be used for:

- syllabi,
- course proposals,
- accreditation support,
- program mapping,
- LMS course pages,
- assessment planning,
- and adoption planning.

Instructors should select outcomes that align with the course scope and transformation goals.

Courses should remain focused.

No single course needs every outcome.

ETIS Educational Philosophy

ETIS organizes learning around engineering responsibilities.

Students should progressively learn how to:

- define intent,
- engineer context,
- bound authority,
- verify behavior,
- operate reality,
- explain decisions,
- and own outcomes.

These responsibilities become the foundation for all learning outcomes.

Learning Outcome Categories

The outcomes are organized into the following domains:

Engineering Intent

Repository-Centered Engineering

Requirements And Planning

Architecture And Design

Implementation Under Review

AI Governance

Testing And Validation

Release Readiness

Operational Thinking

Governance And Risk

Professional Accountability

Engineering Stewardship

Instructors should intentionally select domains rather than trying to cover all of them equally.

Engineering Intent Outcomes

Students will be able to:

EI-1 Define engineering intent before beginning implementation activities.

EI-2 Differentiate between problem statements, solution assumptions, and engineering decisions.

EI-3 Identify ambiguities that require clarification before work begins.

EI-4 Explain why engineering intent must remain visible throughout the lifecycle.

EI-5 Communicate engineering goals to technical and non-technical stakeholders.

Repository-Centered Engineering Outcomes

Students will be able to:

RCE-1 Use repositories as authoritative engineering records rather than code storage locations.

RCE-2 Organize engineering evidence so that another reviewer can understand the work performed.

RCE-3 Create repository structures that support reviewability and long-term maintainability.

RCE-4 Distinguish between engineering artifacts and engineering evidence.

RCE-5 Explain how repositories preserve engineering memory over time.

Requirements And Planning Outcomes

Students will be able to:

RP-1 Translate stakeholder needs into engineering requirements.

RP-2 Identify assumptions, dependencies, and risks during planning activities.

RP-3 Develop plans that align engineering effort with project objectives.

RP-4 Evaluate the impact of changing requirements.

RP-5 Communicate tradeoffs associated with planning decisions.

Architecture And Design Outcomes

Students will be able to:

AD-1 Evaluate architectural alternatives using engineering reasoning.

AD-2 Explain why boundaries are essential to system trustworthiness.

AD-3 Identify dependencies that create engineering risk.

AD-4 Document architecture decisions and their rationale.

AD-5 Defend architecture choices using evidence and tradeoffs.

Implementation Under Review Outcomes

Students will be able to:

IR-1 Implement solutions within established engineering boundaries.

IR-2 Participate in engineering reviews constructively and professionally.

IR-3 Incorporate feedback into engineering improvements.

IR-4 Differentiate between coding and engineering.

IR-5 Explain how review improves software quality and trustworthiness.

AI Governance Outcomes

Students will be able to:

AIG-1 Use AI as an engineering capability under human accountability.

AIG-2 Disclose meaningful AI use appropriately.

AIG-3 Verify AI-assisted outputs before accepting them.

AIG-4 Evaluate risks associated with AI-generated artifacts.

AIG-5 Explain why human ownership remains essential in AI-assisted engineering.

AIG-6 Bound AI authority within defined engineering contexts.

Testing And Validation Outcomes

Students will be able to:

TV-1 Develop evidence that supports engineering claims.

TV-2 Differentiate between testing activities and engineering validation.

TV-3 Identify limitations in testing approaches.

TV-4 Evaluate whether evidence sufficiently supports system behavior claims.

TV-5 Communicate known defects, risks, and uncertainties.

Release Readiness Outcomes

Students will be able to:

RR-1 Evaluate whether a system is ready for release.

RR-2 Identify evidence required to support release decisions.

RR-3 Communicate known risks associated with a release.

RR-4 Explain why a functioning demo is insufficient evidence of release readiness.

RR-5 Defend release decisions using evidence.

Operational Thinking Outcomes

Students will be able to:

OT-1 Explain how systems will be operated after deployment.

OT-2 Identify operational risks that emerge after implementation.

OT-3 Evaluate the role of observability in system trustworthiness.

OT-4 Explain how postmortems improve future engineering decisions.

OT-5 Differentiate between deploying software and operating systems.

Governance And Risk Outcomes

Students will be able to:

GR-1 Identify governance responsibilities throughout the engineering lifecycle.

GR-2 Evaluate engineering risks continuously.

GR-3 Communicate engineering limitations transparently.

GR-4 Recognize where oversight is required.

GR-5 Explain why governance is an architectural responsibility.

Professional Accountability Outcomes

Students will be able to:

PA-1 Accept responsibility for engineering outcomes.

PA-2 Communicate engineering decisions professionally.

PA-3 Participate effectively in engineering teams.

PA-4 Communicate uncertainties honestly.

PA-5 Provide evidence that supports engineering claims.

PA-6 Explain limitations without overstating confidence.

Engineering Stewardship Outcomes

Students will be able to:

ES-1 Evaluate long-term impacts of engineering decisions.

ES-2 Preserve engineering memory for future maintainers.

ES-3 Improve systems through iterative learning.

ES-4 Balance short-term implementation goals with long-term maintainability.

ES-5 Recognize that trustworthy systems evolve over time.

Recommended Outcome Counts

The following ranges are recommendations.

Undergraduate Software Engineering 8-12 outcomes

Graduate Software Engineering 10-15 outcomes

Senior Capstone 10-15 outcomes

AI Governance Module 5-8 outcomes

Software Architecture Module 6-10 outcomes

DevOps And Operational Readiness Module 6-10 outcomes

Professional Training Program 5-10 outcomes

Select outcomes intentionally.

Avoid overloading a course.

Example Outcome Sets

Undergraduate Software Engineering

Recommended outcomes:

EI-1

RCE-1

RP-1

AD-2

IR-2

AIG-3

TV-1

RR-4

PA-1

ES-5

Graduate Software Engineering

Recommended outcomes:

EI-4

RCE-5

AD-5

AIG-6

RR-5

OT-3

GR-5

PA-6

ES-2

ES-4

Outcome Selection Rules

When selecting outcomes:

Choose Breadth Carefully Do not attempt to teach every domain deeply.

Prioritize Transformation Select outcomes that support who learners should become.

Align Assessments Every selected outcome should have corresponding evidence.

Align Activities Students should practice the behavior being assessed.

Align Reviews Students should experience accountability.

Learning outcomes should not become isolated statements disconnected from the course experience.

Outcomes Should Map To Evidence

Every ETIS outcome should answer:

> What evidence would prove that students achieved this outcome?

Examples:

Outcome Domain	Example Evidence
Requirements	Requirements package
Architecture	ADRs and architecture reviews
AI Governance	AI-use logs
Testing	Validation evidence
Release Readiness	Release records
Operational Thinking	Runbooks and postmortems
Accountability	Final engineering defense

Learning outcomes should remain observable.

What Learning Outcomes Are Not

Learning outcomes are not:

- lecture topics,
- chapter lists,
- assignment titles,
- software features,
- tool proficiency statements,

- or documentation requirements.

They are statements of professional capability.

Final Standard

A strong ETIS learning outcome should answer one question:

> What responsibility should the learner be able to own?

If an outcome cannot be tied to engineering responsibility, it should be reconsidered.

The goal of ETIS education is not to produce students who know more terminology.

The goal is to produce engineers who can define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

ETIS Repository Expectations Guidance

The **ETIS Repository Expectations Guidance** document provides reusable syllabus language and governance principles for teaching repository-centered engineering within ETIS-based courses.

This document helps instructors establish the repository as the authoritative engineering record rather than a location for storing code and assignment submissions.

The repository is one of the defining characteristics of ETIS education.

Students should understand from the beginning of the course that repositories preserve engineering intent, decisions, evidence, accountability, and engineering memory.

Purpose

The purpose of this document is to help instructors establish consistent repository expectations across ETIS implementations.

Repository expectations should answer a simple question:

> If another engineer examined this repository six months from now, would they understand what happened?

ETIS treats repositories as professional engineering systems rather than technical storage systems.

Students should learn how repositories support:

- engineering intent,
- engineering evidence,
- reviewability,
- accountability,
- transparency,
- operational thinking,
- and long-term stewardship.

Educational Philosophy

ETIS teaches repository-centered engineering.

The repository is not where engineering ends.

The repository is where engineering becomes visible.

Students should understand that repositories preserve evidence of:

- what was intended,
- what decisions were made,
- what work was performed,
- what was reviewed,
- what was tested,
- what AI contributed,
- what risks remain,
- and why engineering decisions were accepted.

A repository should help future engineers understand both the system and the engineering that created the system.

Core Repository Principle

The following principle should remain visible throughout ETIS implementations.

> The learning management system tells students what to do. The repository proves what the team actually did.

Assignments may be submitted through institutional systems.

Engineering evidence should live in the repository.

Recommended Syllabus Statement

The following language may be used directly in a syllabus.

Standard Repository Statement

This course uses repository-centered engineering. The repository serves as the authoritative engineering record for the team and system throughout the semester.

Students are expected to preserve evidence of engineering intent, requirements, plans, architecture decisions, implementation activities, reviews, testing, AI use, release readiness, operational considerations, and final engineering accountability.

The repository is not a code submission mechanism. It is an engineering evidence system.

Students should assume that another engineer, reviewer, or future maintainer may eventually inspect the repository to understand the work that was performed.

Short Repository Statement

The following language may be used in abbreviated syllabi.

This course uses repository-centered engineering. The repository serves as the authoritative engineering record and should preserve evidence of decisions, reviews, testing, AI use, risks, and engineering accountability.

First-Day Student Statement

The following language may be used during course introductions.

You are not building a collection of assignments.

You are building an engineering system.

The repository is where that system becomes visible.

Your repository should help another engineer answer questions such as:

- What problem were you solving?
- What decisions did you make?
- Why did you make them?
- What changed over time?
- What did you review?
- What did you test?
- How did AI help?
- What risks remain?
- Why is this release defensible?

The repository should answer those questions without requiring you to stand next to it and explain everything.

Repository Responsibilities

Students should understand that repositories support multiple engineering responsibilities.

These include:

Responsibility 1 — Preserve Intent

The repository should preserve why work is being performed.

Examples:

- problem statements,
- requirements,
- assumptions,
- goals,
- scope boundaries.

Responsibility 2 — Preserve Decisions

The repository should preserve important engineering decisions.

Examples:

- architecture decisions,
- tradeoffs,
- alternatives considered,
- rationale.

Responsibility 3 — Preserve Evidence

The repository should preserve evidence that supports engineering claims.

Examples:

- test evidence,
- validation records,
- review artifacts,
- release evidence.

Responsibility 4 — Preserve Accountability

The repository should make engineering participation visible.

Examples:

- pull requests,
- reviews,
- AI disclosures,
- ownership records.

Responsibility 5 — Preserve Operational Thinking

The repository should demonstrate that students considered life after implementation.

Examples:

- risks,
- observability,
- runbooks,
- postmortems,
- release readiness.

Responsibility 6 — Preserve Engineering Memory

The repository should help future engineers understand the engineering history of the system.

Repository Evidence Categories

A strong ETIS repository often contains evidence categories such as:

/docs /requirements /planning /architecture /decisions /reviews /testing /quality /release /operations /observability /security /governance /ai /postmortems /presentations

/src

/tests

/.github

This is an example.

Exact structures may vary by implementation.

The evidence expectations should remain stable.

Repository Design Principles

Students should learn these principles.

Principle 1 — Repositories Are Evidence Systems

Repositories should explain engineering work.

Not merely store engineering work.

Principle 2 — Repositories Should Support Review

A reviewer should be able to navigate the repository efficiently.

Organization matters.

Principle 3 — Repositories Should Explain Decisions

Engineering decisions should remain visible.

Principle 4 — Repositories Should Support Accountability

Contributions and reviews should be traceable.

Principle 5 — Repositories Should Support Future Engineers

Repositories should outlive the semester.

Repository Quality Characteristics

Students should strive to make repositories:

- understandable,
- navigable,
- reviewable,
- maintainable,
- transparent,
- evidence-driven,
- accountable,
- and trustworthy.

These qualities are often more important than aesthetic perfection.

Repository Review Questions

Students should periodically ask:

- Could another engineer understand this repository?
- Could another engineer find important information quickly?
- Are engineering decisions visible?
- Is AI use transparent?
- Are risks documented?
- Are reviews preserved?
- Is evidence connected to claims?
- Could this repository survive team turnover?

If the answer is no, the repository should improve.

AI And Repositories

Repositories should preserve meaningful AI use.

Examples include:

- AI-use logs,
- architecture critiques,
- review notes,
- verification activities,
- lessons learned.

AI contributions should remain visible.

Transparency strengthens trust.

Repository Expectations For Teams

Teams should establish shared repository habits.

Teams should:

- agree on organization,
- review work regularly,
- communicate changes,
- preserve evidence continuously,

- maintain documentation,
- and improve repository quality over time.

Repository stewardship is a team responsibility.

Repository Expectations For Instructors

Instructors should evaluate repositories as engineering systems.

Evaluation should extend beyond code quality.

Instructors should look for evidence of:

- engineering intent,
- planning,
- decision making,
- reviews,
- testing,
- AI governance,
- release readiness,
- operational thinking,
- and accountability.

Common Student Misconceptions

Misconception 1 — The Repository Is A Submission Folder

It is an engineering record.

Misconception 2 — Only Code Matters

Engineering evidence matters.

Misconception 3 — Documentation Is Separate From Engineering

Documentation is part of engineering.

Misconception 4 — AI Use Does Not Need To Be Visible

Transparency matters.

Misconception 5 — Repositories End At Deployment

Repositories should support operation and future change.

Common Instructor Mistakes

Mistake 1 — Grading Only Code

Evaluate evidence.

Mistake 2 — Waiting Until The End Of The Semester To Inspect Repositories

Repository quality should be evaluated continuously.

Mistake 3 — Treating Documentation As Administrative Overhead

Documentation preserves engineering memory.

Mistake 4 — Over-Prescribing Folder Structures

Teach responsibilities first.

Structures are secondary.

Mistake 5 — Disconnecting Repositories From Assessments

Repository evidence should directly support assessment activities.

Adaptation Rules

Institutions may adapt repository structures.

However, adaptations should preserve these principles:

- repositories preserve engineering intent,
- repositories preserve evidence,
- repositories preserve accountability,
- repositories support review,
- repositories support operations,
- repositories preserve engineering memory.

These principles should remain visible.

Guiding Standard

Students should be able to answer these questions.

- What problem were we solving?
- What decisions did we make?
- What evidence supports our claims?
- What did AI help produce?
- What did humans verify?
- What risks remain?
- Why is this release defensible?
- Could another engineer understand our repository?

If those answers are difficult to find, the repository needs improvement.

Core Commitment

ETIS repositories are not assignment containers.

They are professional engineering systems.

Students should leave ETIS understanding that repositories preserve engineering memory.

Code may eventually change.

Teams may eventually disappear.

Technologies may eventually evolve.

Engineering memory should remain.

That is one of the foundations of trustworthy intelligent systems.

ETIS Syllabus Assembly Guide

The **ETIS Syllabus Assembly Guide** provides a repeatable process for building ETIS-based course syllabi.

This guide does not provide a single ETIS syllabus.

Instead, it provides a structured assembly process that allows instructors to build institution-specific syllabi while preserving ETIS educational philosophy.

The goal is not standardization.

The goal is consistency of educational purpose.

Every ETIS syllabus should communicate that software engineering is a professional discipline centered on engineering trust.

Purpose

The purpose of this guide is to help instructors assemble coherent ETIS syllabi without rebuilding the educational ecosystem from scratch.

This guide helps instructors answer questions such as:

- What type of ETIS course am I building?
- What learner transformation am I pursuing?
- Which ETIS chapters are primary?
- Which learning outcomes should be selected?
- How should AI be governed?
- What repository expectations should students understand?
- What professional expectations should be communicated?
- What evidence will students create?

The resulting syllabus should clearly communicate both educational expectations and engineering expectations.

Syllabus Philosophy

Traditional syllabi often communicate logistics.

ETIS syllabi communicate professional formation.

An ETIS syllabus should answer:

> What kind of engineer is this course attempting to create?

Students should understand from the first day that the course is designed around engineering responsibilities rather than assignment completion.

ETIS Syllabus Assembly Model

ETIS syllabi should be assembled from reusable components.

Professional Transformation Goal ↓ Course Type ↓ ETIS Book Mapping ↓ Learning Outcomes ↓ AI Governance ↓ Repository Expectations ↓ Professional Expectations ↓ Assessment Philosophy ↓ Local Institutional Requirements ↓ Final Syllabus

The process should proceed in this order.

Step 1 — Define The Course Type

Begin by identifying the instructional environment.

Examples include:

- undergraduate software engineering,
- graduate software engineering,
- senior capstone,
- AI governance,
- software architecture,
- DevOps and operational readiness,
- professional training,
- or organizational education.

The course type establishes scope and depth.

Document this decision first.

Step 2 — Define The Professional Transformation Goal

Identify who learners should become.

Examples:

Student ↓ Responsible Engineer

Student ↓ Reviewer

Student ↓ Release Defender

Student ↓ Operational Thinker

Student ↓ Future Trustworthy Engineer

One transformation should be primary.

This decision should influence every remaining section.

Step 3 — Map The ETIS Book

The ETIS book remains authoritative.

Select:

Required Chapters Students must understand these concepts deeply.

Supporting Chapters Students should interact with these concepts.

Reference Chapters Students may consult these concepts as needed.

The syllabus should make this distinction visible.

Avoid attempting to cover all 39 chapters equally.

Step 4 — Select Learning Outcomes

Use:

learning_outcomes_library.md

Select outcomes intentionally.

Recommended ranges:

Undergraduate: 8-12

Graduate: 10-15

Capstone: 10-15

Modules: 5-8

Professional Training: 5-10

Learning outcomes should align with transformation goals.

Step 5 — Assemble AI Governance Language

Use:

ai_use_policy_guidance.md

Select language appropriate for the course.

The syllabus should clearly define:

- AI philosophy,
- acceptable AI use,
- restricted AI use,
- disclosure expectations,
- verification expectations,
- accountability expectations.

Students should understand that ETIS teaches AI governance rather than AI avoidance.

Step 6 — Assemble Repository Expectations

Use:

repository_expectations_guidance.md

The syllabus should communicate:

- repository purpose,
- evidence expectations,
- accountability expectations,
- review expectations,
- engineering memory expectations.

Students should understand that repositories are engineering systems.

Step 7 — Assemble Professional Expectations

Use:

professional_expectations_guidance.md

The syllabus should communicate expectations regarding:

- teamwork,
- communication,
- accountability,
- deadlines,
- risk communication,
- reviews,
- engineering participation.

Professional expectations should be visible early.

Step 8 — Assemble Academic Integrity Language

Use:

academic_integrity_guidance.md

Students should understand:

- honesty expectations,
- ownership expectations,
- attribution expectations,
- transparency expectations,
- integrity expectations.

Academic integrity and AI governance should remain separate sections.

Step 9 — Define Assessment Philosophy

The syllabus should communicate how engineering will be evaluated.

Assessment should extend beyond technical output.

Students should understand that they will be evaluated on:

- engineering intent,
- evidence quality,
- review participation,
- AI accountability,
- testing discipline,
- architecture reasoning,
- release readiness,
- operational thinking,
- and professional accountability.

The syllabus should communicate this explicitly.

Step 10 — Define Local Institutional Information

Finally, add institution-specific information.

Examples:

- course logistics,

- meeting schedules,
- office hours,
- grading percentages,
- university policies,
- calendars,
- technology requirements,
- accessibility statements.

These items should be added last.

Local information should not drive ETIS educational philosophy.

Recommended Syllabus Structure

A mature ETIS syllabus may include sections such as:

1. Course Description
2. Educational Mission
3. Professional Transformation Goal
4. Learning Outcomes
5. ETIS Book Mapping
6. Student Repository Expectations
7. AI Governance
8. Professional Expectations
9. Academic Integrity
10. Assessment Philosophy
11. Course Logistics
12. Institutional Policies

Instructors may adapt this structure.

Syllabus Review Questions

Before finalizing a syllabus, an instructor should ask:

- Does this syllabus explain what engineers students will become?
- Does this syllabus explain engineering responsibilities?
- Does this syllabus explain repository expectations?
- Does this syllabus explain AI governance?
- Does this syllabus explain assessment expectations?
- Does this syllabus explain accountability?
- Does this syllabus communicate professional expectations?

If these answers are unclear, the syllabus needs refinement.

Common Assembly Mistakes

Mistake 1 — Starting With Grading Percentages

Educational purpose should come first.

Mistake 2 — Organizing Around Topics Alone

Organize around professional transformation.

Mistake 3 — Treating AI As A Separate Topic

AI governance should be integrated throughout the course.

Mistake 4 — Forgetting Repository Expectations

Repository-centered engineering is foundational.

Mistake 5 — Combining AI Governance And Academic Integrity

Keep these separate.

Mistake 6 — Overloading Learning Outcomes

Focus creates stronger courses.

Mistake 7 — Making The Syllabus Administrative

The syllabus should communicate engineering identity.

ETIS Syllabus Build Pipeline

The assembly process can be summarized as:

Select Course Type ↓ Select Transformation Goal ↓ Map ETIS Book ↓ Select Learning Outcomes ↓ Assemble AI Governance ↓ Assemble Repository Expectations ↓ Assemble Professional Expectations ↓ Assemble Academic Integrity ↓ Define Assessment Philosophy ↓ Add Institutional Requirements ↓ Publish Syllabus

This process should remain stable across ETIS implementations.

Stewardship Rules

Future maintainers should preserve these rules.

- Do not create a universal ETIS syllabus.
- Do not duplicate COMP330 materials.
- Do not make assets semester-specific.
- Do not prioritize logistics over educational philosophy.
- Do not disconnect syllabi from ETIS doctrine.
- Do not create administrative bureaucracy.

The syllabus should remain a professional formation document.

Guiding Standard

Every ETIS syllabus should answer one question:

> What kind of engineer will a student become after completing this course?

If the syllabus cannot answer that question clearly, it needs improvement.

Core Commitment

The purpose of ETIS syllabus assembly is not to create better course documents.

The purpose is to help instructors intentionally communicate what software engineering has become in the AI era.

Students should leave understanding that software engineering is not merely software construction.

It is the disciplined practice of creating systems that can be understood, reviewed, governed, operated, improved, and trusted over time.

Part III

Schedule Guidance

ETIS Book Mapping Guidance

The **ETIS Book Mapping Guidance** document provides recommendations for translating the 39-chapter ETIS book into educational experiences.

This document does not prescribe a single teaching sequence.

Instead, it helps instructors transform ETIS doctrine into teachable learning journeys.

The ETIS book should be viewed as a capability library rather than a linear textbook.

The objective is not to teach every chapter equally.

The objective is to intentionally develop trustworthy engineers.

Purpose

The purpose of this document is to help instructors answer questions such as:

- Which ETIS chapters belong together?
- Which chapters are foundational?
- Which chapters are advanced?
- Which chapters are appropriate for undergraduate courses?
- Which chapters are appropriate for graduate courses?
- Which chapters are appropriate for modules?
- Which chapters are appropriate for professional training?
- How should the 39 chapters be consumed without overwhelming learners?

This document helps educators intentionally map ETIS doctrine into educational experiences.

Guiding Principle

ETIS chapters should be mapped according to engineering capability development rather than chapter order.

The book is not designed to be consumed sequentially in every educational environment.

The book is designed to support multiple educational pathways.

The central question should always be:

> What engineering capability are we trying to develop?

Chapter selection should answer that question.

Educational Philosophy

Traditional textbooks are often consumed linearly.

Chapter 1
↓
Chapter 2
↓
Chapter 3

ETIS should be consumed as a capability system.

Capability Goal

↓

Relevant ETIS Chapters

↓

Educational Experience

↓

Evidence Creation

↓

Professional Accountability

The capability drives the chapter selection.

The chapter numbering should not drive the course.

The ETIS Capability Domains

The 39 chapters naturally organize into four educational domains.

Foundations

↓

Construction

↓

Operations

↓

Stewardship

These domains provide the highest-level mapping structure.

Domain 1 — Foundations

Purpose:

Establish engineering philosophy and professional identity.

Students learn:

- what trustworthy engineering is,
- why ETIS exists,
- why repositories matter,
- why evidence matters,
- why AI changes engineering.

Primary chapters:

Chapters 1-7

Typical audiences:

- undergraduate courses,

- graduate courses,
- professional training,
- leadership briefings.

Educational emphasis:

- engineering trust,
- evidence,
- professional accountability.

Domain 2 — Construction

Purpose:

Teach how trustworthy systems are engineered.

Students learn:

- requirements,
- planning,
- architecture,
- AI-assisted engineering,
- reviews,
- testing,
- release readiness.

Primary chapters:

Chapters 8-18

Typical audiences:

- software engineering,
- architecture,
- capstone,
- graduate engineering.

Educational emphasis:

- disciplined construction,
- reviewability,
- accountability.

Domain 3 — Operations

Purpose:

Teach engineering beyond implementation.

Students learn:

- operations,
- postmortems,
- reliability,
- security,
- observability,
- release governance,
- incident response.

Primary chapters:

Chapters 19-32

Typical audiences:

- upper-level undergraduate,
- graduate,
- DevOps,
- professional training.

Educational emphasis:

- operational trust,
- system survivability,
- accountability after deployment.

Domain 4 — Stewardship

Purpose:

Teach long-term engineering ownership.

Students learn:

- agentic systems,
- context engineering,
- human oversight,
- explainability,
- stewardship.

Primary chapters:

Chapters 33-39

Typical audiences:

- graduate students,
- advanced undergraduates,
- architects,
- leaders,
- professionals.

Educational emphasis:

- future-oriented engineering,
- AI-era stewardship,
- long-term accountability.

Educational Consumption Models

The same ETIS book supports different educational experiences.

Model 1 — Undergraduate Software Engineering

Primary emphasis:

Parts I and II

Selected chapters from III and IV

Recommended coverage:

Chapters 1-18

Selected:

23

25

26

31

33

34

35

36

38

Educational focus:

- foundations,
- construction,
- selected operational awareness,
- future stewardship.

Model 2 — Graduate Software Engineering

Primary emphasis:

All Parts

Greater depth in III and IV

Recommended coverage:

1-39

Educational focus:

- architecture,
- governance,
- AI accountability,
- operational trust,
- stewardship.

Depth should increase rather than breadth.

Model 3 — Senior Capstone

Primary emphasis:

Parts II and III

Selected IV

Recommended coverage:

8-32

Selected:

33

34

35

38

Educational focus:

- construction,
- release readiness,
- operations,
- accountability.

Model 4 — AI Governance Module

Recommended chapters:

14

28

33

34

35

36

38

Educational focus:

- AI governance,
- bounded authority,
- context engineering,
- oversight,
- stewardship.

Model 5 — Software Architecture Module

Recommended chapters:

11

13

15

34

35

Educational focus:

- boundaries,
- decisions,
- context,
- oversight.

Model 6 — Operational Readiness Module

Recommended chapters:

23

24

25

26

29

30

31

Educational focus:

- postmortems,
- reliability,
- incident response,
- release governance.

Model 7 — Professional Training

Recommended chapters:

1

2

4

14

26

31

33

34

35

38

Educational focus:

- trust,
- AI accountability,
- evidence,
- operations,
- stewardship.

Avoid attempting all 39 chapters.

Model 8 — Leadership Briefing

Recommended chapters:

1

4

14

26

31

33

38

Educational focus:

- engineering trust,
- governance,
- organizational accountability,
- stewardship.

ETIS Educational Maturity Mapping

The ETIS transformation model maps naturally to the book.

Student
↓
Responsible Engineer
↓
Reviewer
↓
Architect
↓
Release Defender
↓
Operational Thinker
↓
Future Trustworthy Engineer

The educational experience should progressively introduce chapters that support each stage.

Recommended Progressive Exposure

Students should not encounter all ETIS concepts immediately.

A healthy progression looks like:

Foundations

↓

Construction

↓

Operations

↓

Stewardship

This progression should remain visible.

Chapter Selection Rules

Instructors should ask:

Rule 1

What engineering capability am I teaching?

Rule 2

What professional transformation am I pursuing?

Rule 3

What chapters support that transformation?

Rule 4

What evidence should students create?

Rule 5

What accountability experience should students have?

Chapter selection should always support these questions.

Common Mapping Mistakes

Mistake 1 — Teaching Chapters Sequentially

ETIS is a capability system.

Mistake 2 — Trying To Cover All 39 Chapters

Focus produces stronger learning.

Mistake 3 — Ignoring Operations

Students should see life after implementation.

Mistake 4 — Ignoring Stewardship

Students should see future engineering responsibilities.

Mistake 5 — Treating Part IV As Optional

Part IV increasingly defines AI-era engineering.

Mistake 6 — Overloading Undergraduate Students

Introduce concepts progressively.

Mistake 7 — Making The Book The Course

The book supports the educational experience.

The educational experience remains primary.

Adaptation Rules

Educational environments may adapt chapter selections.

However, they should preserve exposure to:

- engineering trust,
- evidence,
- AI accountability,
- review,
- release readiness,
- operational thinking,
- stewardship.

These remain constitutional ETIS concepts.

Guiding Standard

A successful ETIS book mapping should allow learners to answer:

- Why should this system be trusted?
- What evidence supports our claims?
- How was AI governed?
- What did humans verify?
- What risks remain?
- How will the system operate?
- Who owns the outcome?
- How will the system evolve?

If learners cannot answer these questions, chapter selection should be reconsidered.

Core Commitment

The ETIS book is not a traditional textbook.

It is a doctrine for engineering trustworthy intelligent systems.

Educational experiences should consume ETIS intentionally rather than sequentially.

The goal is not to complete chapters.

The goal is to form engineers who can define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

ETIS Module Sequence Guidance

The **ETIS Module Sequence Guidance** document provides recommendations for building focused ETIS educational modules.

Modules are intentionally smaller educational experiences that teach one or more ETIS capabilities without requiring adoption of an entire ETIS course.

Modules should be capability-centric rather than lifecycle-centric.

The objective is not to shrink a semester into a smaller package.

The objective is to deliver meaningful ETIS experiences that can be embedded into existing educational environments.

Purpose

The purpose of this document is to help instructors build focused ETIS modules that remain coherent, teachable, and maintainable.

This document helps instructors answer questions such as:

- How can ETIS be introduced into existing courses?
- How large should a module be?
- Which ETIS capabilities work well together?
- How should modules be sequenced?
- What activities belong inside a module?
- What evidence should students produce?
- How should AI governance be integrated?
- How should students demonstrate accountability?

Modules should create meaningful professional experiences without requiring a full ETIS adoption.

Guiding Principle

ETIS modules should answer one question:

> What engineering capability should learners strengthen?

Examples include:

- AI governance,
- architecture reasoning,
- repository-centered engineering,
- release readiness,
- operational thinking,
- engineering reviews,
- evidence creation,
- or stewardship.

The capability determines the module.

The module should not attempt to teach everything.

Module Philosophy

Traditional modules often teach isolated topics.

ETIS modules teach engineering responsibilities.

Traditional modules may look like:

Week 1 Architecture

Week 2 Testing

Week 3 Deployment

ETIS modules should look more like:

Capability

↓

Practice

↓

Review

↓

Evidence

↓

Accountability

Students should experience a complete learning loop.

ETIS Module Architecture

Most ETIS modules should contain five educational stages.

Introduction

↓

Capability Development

↓

Practice

↓

Evidence Creation

↓

Reflection And Accountability

This architecture should remain relatively stable regardless of module size.

Module Categories

ETIS modules generally fall into one of six categories.

Foundational Modules

AI Governance Modules

Architecture Modules

Review Modules

Operational Modules

Stewardship Modules

Each category serves a different educational purpose.

Foundational Modules

These modules introduce ETIS concepts.

Examples:

- trustworthy engineering,
- repository-centered engineering,
- engineering evidence,
- AI accountability,
- engineering maturity.

Recommended Duration 2-6 weeks

Typical Activities

- discussions,
- repository walkthroughs,
- small exercises,
- reflection activities.

Typical Evidence

- short engineering records,
- repository artifacts,
- engineering reflections.

AI Governance Modules

These modules teach responsible AI-assisted engineering.

Topics may include:

- AI accountability,
- bounded authority,
- disclosure,
- verification,
- context engineering,
- risk awareness.

Recommended Duration 2-6 weeks

Typical Activities

- AI critique exercises,
- disclosure exercises,
- governance discussions,

- verification activities.

Typical Evidence

- AI-use logs,
- verification reports,
- risk assessments.

Architecture Modules

These modules teach architectural reasoning.

Topics may include:

- boundaries,
- dependencies,
- tradeoffs,
- ADRs,
- governance points.

Recommended Duration 3-6 weeks

Typical Activities

- architecture critiques,
- ADR exercises,
- tradeoff analysis.

Typical Evidence

- architecture packages,
- decision records,
- review evidence.

Review Modules

These modules teach engineering review systems.

Topics may include:

- pull requests,
- review participation,
- evidence evaluation,
- engineering critiques.

Recommended Duration 2-5 weeks

Typical Activities

- review simulations,
- peer reviews,
- critique sessions.

Typical Evidence

- review records,
- feedback summaries,
- improvement plans.

Operational Modules

These modules teach engineering after implementation.

Topics may include:

- release readiness,
- observability,
- postmortems,
- operational risk,
- incident response.

Recommended Duration 3-6 weeks

Typical Activities

- release reviews,
- tabletop exercises,
- incident simulations.

Typical Evidence

- release records,
- postmortems,
- operational readiness packages.

Stewardship Modules

These modules teach long-term engineering thinking.

Topics may include:

- system evolution,
- engineering memory,
- governance,
- sustainability,
- continuous improvement.

Recommended Duration 2-5 weeks

Typical Activities

- case analysis,
- stewardship discussions,
- system evolution exercises.

Typical Evidence

- stewardship plans,
- improvement recommendations,
- governance summaries.

Recommended Module Sizes

ETIS modules generally work best within these durations.

Two-Week Module

Purpose:

Focused exposure.

Structure:

Week 1

Introduction

↓

Capability Development

Week 2

Practice

↓

Evidence

↓

Reflection

Four-Week Module

Purpose:

Capability development.

Structure:

Week 1

Introduction

Week 2

Capability Development

Week 3

Practice And Review

Week 4

Evidence And Accountability

Six-Week Module

Purpose:

Deep capability development.

Structure:

Weeks 1-2

Introduction And Capability Development

Weeks 3-4

Practice And Review

Week 5

Evidence Creation

Week 6

Reflection And Accountability

Module Design Recommendations

Every module should contain:

Educational Purpose

Students should know why the capability matters.

Engineering Practice

Students should perform engineering work.

Reviews

Students should experience accountability.

Evidence

Students should preserve engineering evidence.

Reflection

Students should explain what they learned.

AI Governance Within Modules

AI governance should appear inside every module.

Even if AI is not the primary topic.

Students should understand:

- AI is allowed.
- AI is governed.
- AI should be disclosed.
- AI should be verified.
- Humans own outcomes.

This message should remain consistent throughout ETIS.

Repository Expectations Within Modules

Even small modules should use repository-centered thinking.

Students should preserve evidence rather than submit isolated assignments.

The amount of evidence may shrink.

The philosophy should not.

Classroom Experience Recommendations

ETIS modules work especially well with active learning.

Examples include:

- architecture critiques,
- ambiguity workshops,
- AI governance discussions,
- review simulations,
- tabletop exercises,
- incident simulations,
- postmortem discussions.

Avoid lecture-only modules.

Students should practice engineering.

Recommended Instructor Roles

Instructor roles should evolve during modules.

```
Introduction
Teacher
    ↓
Practice
Coach
    ↓
Review
Reviewer
    ↓
Accountability
Engineering Defense Board
```

Students should progressively assume more responsibility.

Common Module Mistakes

Mistake 1 — Shrinking A Semester Into A Module

Modules should be capability-focused.

Mistake 2 — Trying To Cover Too Many ETIS Capabilities

Focus creates stronger modules.

Mistake 3 — Eliminating Evidence

Students should still preserve evidence.

Mistake 4 — Removing Accountability

Students should still defend work.

Mistake 5 — Making Modules Lecture Heavy

Students should practice engineering.

Mistake 6 — Ignoring AI Governance

AI governance should remain visible.

Mistake 7 — Treating Modules As Standalone Topics

Modules should still reinforce ETIS doctrine.

Adaptation Rules

Modules may adapt to local constraints.

However, they should preserve:

- engineering intent,
- evidence,
- review,
- AI accountability,
- reflection,
- and professional ownership.

These elements should remain visible.

Guiding Standard

A successful ETIS module should allow students to answer:

- What engineering capability did I strengthen?
- What evidence did I create?
- How did AI assist?
- What did I verify?
- What did I review?
- What risks did I identify?
- What responsibility did I own?

If students cannot answer these questions, the module needs redesign.

Core Commitment

The purpose of ETIS modules is not to create smaller ETIS courses.

The purpose is to make ETIS capabilities accessible, adaptable, and adoptable.

Students should leave every ETIS module stronger than they entered.

Even small educational experiences should help learners define intent, engineer context, bound authority, verify behavior, explain decisions, and own outcomes.

That progression remains one of the foundations of ETIS education.

ETIS Phase Gate Sequencing Guidance

The **ETIS Phase Gate Sequencing Guidance** document provides recommendations for organizing ETIS educational experiences around progressive engineering accountability.

This document does not describe assignment schedules.

It describes engineering maturity gates.

Phase gates are one of the primary mechanisms ETIS uses to transform students into trustworthy engineers.

Each phase gate represents an increase in engineering responsibility.

The objective is not to complete assignments.

The objective is to progressively assume ownership of engineering outcomes.

Purpose

The purpose of this document is to help instructors sequence educational experiences around accountability rather than task completion.

This document helps instructors answer questions such as:

- How should engineering responsibility increase over time?
- How should assignments build upon one another?
- How should students experience accountability?
- How should evidence accumulate?
- How should reviews evolve?
- How should release readiness emerge?
- How should operational thinking appear?
- How should students defend engineering decisions?

The resulting educational experience should feel cumulative rather than fragmented.

Guiding Principle

ETIS phase gates should answer one question:

> What new engineering responsibility should students now be able to own?

Every phase gate should increase professional accountability.

Students should progressively move through this transformation.

```
Student
   ↓
Responsible Engineer
   ↓
Reviewer
   ↓
Architect
   ↓
```

Release Defender

↓

Operational Thinker

↓

Future Trustworthy Engineer

Assignments support this progression.

They are not the progression.

Educational Philosophy

Traditional courses often organize work like this:

Assignment 1

↓

Assignment 2

↓

Assignment 3

↓

Assignment 4

Each assignment often becomes isolated.

ETIS organizes work differently.

Foundation Gate

↓

Intent Gate

↓

Architecture Gate

↓

Construction Gate

↓

Release Gate

↓

Operations Gate

↓

Defense Gate

Every gate inherits from previous gates.

Nothing is truly independent.

Foundational Principle

Assignments are temporary.

Phase gates are cumulative.

Students should repeatedly reuse, refine, defend, and improve previous work.

The educational experience should resemble professional engineering.

ETIS Phase Gate Architecture

Most ETIS implementations should use seven phase gates.

Phase Gate 1
Repository Foundation
↓
Phase Gate 2
Intent And Planning
↓
Phase Gate 3
Architecture And Context
↓
Phase Gate 4
Construction Under Review
↓
Phase Gate 5
Release Readiness
↓
Phase Gate 6
Operations And Improvement
↓
Phase Gate 7
Final Engineering Defense

These phase gates should remain relatively stable across ETIS implementations.

The complexity and depth may vary.

Phase Gate 1 — Repository Foundation

Educational Goal

Establish engineering expectations.

Students should understand:

- repository-centered engineering,
- AI accountability,
- engineering evidence,
- team expectations,
- professional responsibilities.

Students should understand:

> The repository is an engineering system.

Recommended Deliverables

Examples:

- team charter,
- repository initialization,
- engineering expectations acknowledgement,
- team roles.

Accountability Question

> Can another engineer understand how this team will work?

Phase Gate 2 — Intent And Planning

Educational Goal

Teach students to define work before implementation.

Topics include:

- requirements,
- stakeholders,
- assumptions,
- scope,
- planning,
- risks.

Students should understand:

> Intent precedes implementation.

Recommended Deliverables

Examples:

- requirements package,
- use cases,
- planning artifacts,
- risk register.

Accountability Question

> Have we clearly defined what we are trying to accomplish?

Phase Gate 3 — Architecture And Context

Educational Goal

Teach architectural thinking.

Topics include:

- boundaries,
- dependencies,
- decisions,
- tradeoffs,
- context engineering.

Students should understand:

> Governance is architecture.

Recommended Deliverables

Examples:

- architecture diagrams,
- ADRs,
- architecture reviews.

Accountability Question

> Why did we build the system this way?

Phase Gate 4 — Construction Under Review

Educational Goal

Teach disciplined implementation.

Topics include:

- pull requests,
- code reviews,
- testing,
- validation,
- AI-assisted engineering.

Students should understand:

> Implementation is not engineering until it has been reviewed.

Recommended Deliverables

Examples:

- implementation artifacts,
- review evidence,
- AI disclosures,
- validation evidence.

Accountability Question

> What evidence supports our engineering claims?

Phase Gate 5 — Release Readiness

Educational Goal

Teach students to evaluate release decisions.

Topics include:

- release evidence,
- known risks,
- testing completeness,
- AI accountability,
- deployment considerations.

Students should understand:

> A demo is not operational proof.

Recommended Deliverables

Examples:

- release readiness package,
- release evidence,
- risk summaries.

Accountability Question

> Why should this release be trusted?

Phase Gate 6 — Operations And Improvement

Educational Goal

Teach engineering beyond deployment.

Topics include:

- observability,
- postmortems,
- reliability,
- governance,
- operational risks.

Students should understand:

> The model is not the system.

Recommended Deliverables

Examples:

- postmortems,
- operational readiness packages,
- improvement plans.

Accountability Question

> How would this system survive in reality?

Phase Gate 7 — Final Engineering Defense

Educational Goal

Create professional accountability.

Students should defend the entire engineering journey.

Students should answer:

- What problem were we solving?
- What decisions did we make?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How would the system be operated?
- Why is this release defensible?

Students should understand:

> Engineering accountability is part of engineering itself.

Recommended Deliverables

Examples:

- defense package,
- final repository,
- engineering presentation,
- lessons learned.

Accountability Question

> Would another engineer trust our work?

Phase Gate Inheritance Model

Every phase gate should inherit from previous work.

Repository Foundation

↓

Intent

↓

Architecture

↓

Construction

↓

Release

↓

Operations

↓

Defense

Earlier artifacts should never disappear.

Students should continuously improve them.

Evidence Growth Model

Evidence should accumulate over time.

Intent Evidence

↓

Decision Evidence

↓

Review Evidence

↓

Validation Evidence

↓

Release Evidence

↓

Operational Evidence

↓

Defense Evidence

The repository should tell a complete engineering story.

Recommended Assessment Philosophy

Do not grade each gate independently.

Evaluate cumulative maturity.

Questions instructors should ask:

- Has engineering thinking matured?
- Has evidence quality improved?
- Has accountability increased?
- Has communication improved?
- Has operational awareness emerged?

Growth matters.

Recommended Instructor Role Progression

Instructor responsibilities should evolve.

Gate 1-2

Teacher

↓

Gate 3-4

Coach

↓

Gate 5-6

Reviewer

↓

Gate 7

Engineering Defense Board

Students should progressively assume ownership.

Common Phase Gate Mistakes

Mistake 1 — Treating Gates Like Assignments

Phase gates are cumulative.

Mistake 2 — Requiring New Documentation Every Time

Students should improve existing artifacts.

Mistake 3 — Delaying AI Governance

AI governance should begin immediately.

Mistake 4 — Ignoring Operations

Operations should appear before the end.

Mistake 5 — Treating Release Readiness As A Presentation

Release readiness is evidence-based accountability.

Mistake 6 — Skipping Final Defense

Final defense is one of the strongest ETIS experiences.

Mistake 7 — Disconnecting Gates From Each Other

Every gate should inherit previous work.

Adaptation Rules

Implementations may adapt phase gates.

However, they should preserve:

- engineering intent,
- context,
- evidence,

- review,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship,
- final accountability.

These are constitutional ETIS capabilities.

Guiding Standard

A successful ETIS phase gate sequence should allow students to answer:

- What did we intend?
- What evidence did we create?
- What decisions did we make?
- What did AI contribute?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?

If students cannot answer these questions, the phase gates should be redesigned.

Core Commitment

The purpose of ETIS phase gates is not to create stronger assignments.

The purpose is to create stronger engineers.

Students should progressively learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

Each phase gate should make that progression visible.

That progression is one of the foundations of ETIS education.

ETIS Professional Training Sequence Guidance

The **ETIS Professional Training Sequence Guidance** document provides recommendations for delivering ETIS within professional, organizational, leadership, and workforce development environments.

This document focuses on transforming existing professionals into more effective trustworthy engineers.

Professional training differs from academic instruction.

Participants often arrive with years of experience, existing habits, organizational constraints, and established engineering cultures.

The objective is not to teach software engineering fundamentals.

The objective is to reorganize engineering thinking for the AI era.

Purpose

The purpose of this document is to help organizations deliver ETIS effectively within professional environments.

This document helps facilitators answer questions such as:

- How should ETIS be introduced to professionals?
- How should training be sequenced?
- How should AI governance be incorporated?
- How should organizational realities be included?
- How should accountability be taught?
- How should participants apply ETIS immediately?
- How should organizations begin adoption?

Professional training should produce immediate applicability rather than delayed understanding.

Guiding Principle

ETIS professional training should answer one question:

> How can professionals improve the trustworthiness of systems they already build?

Professional training is not about creating students.

It is about evolving practitioners.

Participants should progressively move through a transformation model.

Practitioner

↓

Responsible Engineer

↓

Reviewer

↓

Architect

↓

Release Defender

↓

Operational Thinker

↓

Steward Of Trustworthy Systems

This progression should remain visible.

Professional Training Philosophy

Traditional professional training often looks like this:

Presentation

↓

Discussion

↓

Completion Certificate

ETIS professional training should look more like this:

Awareness

↓

Application

↓

Review

↓

Evidence

↓

Operational Adoption

↓

Action Plan

Participants should leave with practical changes they can implement immediately.

ETIS Professional Training Architecture

Most ETIS professional training experiences should include six phases.

Phase 1

Awareness

↓

Phase 2

Current State Evaluation

↓

Phase 3

ETIS Capability Development

↓

Phase 4

Application

↓

Phase 5

Review And Accountability

↓

Phase 6

Organizational Action Planning

This architecture should remain relatively stable regardless of duration.

Phase 1 — Awareness

Educational Goal

Help participants understand why ETIS exists.

Topics may include:

- AI-era engineering changes
- engineering trust
- evidence-centered engineering
- repository-centered engineering
- governance
- operational accountability

Participants should understand:

> AI increases engineering responsibility.

Participants should also understand:

> The model is not the system.

Phase 2 — Current State Evaluation

Educational Goal

Help participants evaluate current engineering practices.

Questions may include:

- How are decisions documented?
- How is AI governed?
- How are releases defended?
- How are risks communicated?
- How are repositories organized?
- How is engineering evidence preserved?

Participants should identify improvement opportunities.

Phase 3 — ETIS Capability Development

Educational Goal

Introduce ETIS capabilities.

Topics may include:

- engineering intent
- context engineering
- AI governance
- repository evidence
- architecture decisions
- reviews
- release readiness
- operational thinking
- stewardship

Participants should begin connecting ETIS concepts to existing environments.

Phase 4 — Application

Educational Goal

Apply ETIS concepts to realistic scenarios.

Activities may include:

- architecture critiques
- repository reviews
- release evaluations
- AI governance exercises
- incident simulations
- operational readiness discussions

Application should dominate this phase.

Phase 5 — Review And Accountability

Educational Goal

Teach participants how to evaluate engineering claims.

Participants should practice asking questions such as:

- What evidence supports this claim?
- What did AI produce?
- What did humans verify?
- What risks remain?
- Is this release defensible?
- Could another engineer understand this system?

Participants should experience accountability.

Phase 6 — Organizational Action Planning

Educational Goal

Translate ETIS into immediate organizational improvements.

Participants should identify:

- quick wins
- process improvements
- governance opportunities
- review opportunities
- documentation improvements
- release improvements

Every professional training experience should end with actionable outcomes.

Recommended Training Models

This document supports several delivery models.

Two-Day Workshop

Best for:

- engineering teams
- leadership teams
- organizational introductions

Day 1

```
Morning
ETIS Foundations
    ↓
Engineering Trust
    ↓
AI Accountability
Afternoon
Repositories
    ↓
Architecture
    ↓
Reviews
```

Day 2

```
Morning
Release Readiness
```

↓

Operations

↓

Risk

Afternoon

Organizational Action Plan

↓

Commitments

↓

Adoption Roadmap

Four-Week Cohort

Best for:

- working professionals
- leadership development
- organizational pilots

Week 1 Trustworthy Engineering

Week 2 Architecture, AI Governance, And Evidence

Week 3 Reviews, Releases, And Operations

Week 4 Stewardship And Organizational Action Plans

Six-Week Cohort

Best for:

- deeper organizational adoption

Week 1

ETIS Foundations

Week 2

Intent, Context, And Repositories

Week 3

Architecture And Reviews

Week 4

AI Governance And Validation

Week 5

Operations And Release Readiness

Week 6

Leadership Briefing

Best for:

- executives
- directors
- managers
- governance stakeholders

Topics include:

- why AI changes engineering
- why trust matters
- why governance is architecture
- why evidence matters
- why operations matter
- why stewardship matters

Avoid deep technical implementation details.

Focus on organizational accountability.

Organizational Application Philosophy

Professional training should always connect to participant environments.

Participants should repeatedly ask:

- How would this work here?
- What would change?
- What problems would this solve?
- What risks would this reduce?
- What behaviors would improve?

ETIS should feel immediately useful.

AI Governance In Professional Training

AI governance should be introduced immediately.

Participants should understand:

- AI is a capability.
- AI is governed.
- AI should be transparent.
- AI should be verified.
- Humans own outcomes.

Many professionals will already be using AI.

ETIS helps them use it responsibly.

Repository Expectations In Professional Training

Repository-centered engineering should remain visible.

Participants should evaluate repositories as engineering systems.

Questions include:

- Are decisions visible?
- Is evidence preserved?
- Are risks documented?
- Is AI use transparent?
- Could another engineer understand this repository?

These questions often create immediate organizational improvements.

Recommended Learning Activities

Professional environments benefit from active learning.

Recommended activities include:

- repository critiques
- architecture critiques
- AI governance scenarios
- release readiness exercises
- incident simulations
- postmortem exercises
- review board simulations

Avoid lecture-only experiences.

Professionals learn by applying concepts.

Instructor Or Facilitator Role Progression

Facilitators should evolve throughout the experience.

Beginning

Educator

↓

Middle

Coach

↓

Later

Reviewer

↓

Final

Strategic Advisor

The goal is increasing participant ownership.

Common Professional Training Mistakes

Mistake 1 — Teaching ETIS Like A College Course

Professionals need application, not lectures.

Mistake 2 — Ignoring Existing Organizational Processes

ETIS should improve existing systems.

Not replace everything.

Mistake 3 — Teaching Theory Without Action

Every concept should connect to immediate improvements.

Mistake 4 — Delaying AI Governance

AI governance should appear immediately.

Mistake 5 — Overloading Participants

Focus on actionable concepts.

Mistake 6 — Ending Without Action Plans

Training should produce organizational change.

Mistake 7 — Treating ETIS As Another Framework

ETIS is an engineering operating model.

That distinction should remain visible.

Adaptation Rules

Organizations may adapt pacing.

However, they should preserve:

- engineering intent
- context
- evidence
- review
- AI accountability
- release readiness
- operational thinking
- stewardship

These remain constitutional ETIS capabilities.

Guiding Standard

A successful ETIS professional training experience should allow participants to answer:

- What engineering behaviors should change?
- What evidence should improve?
- How should AI be governed?
- What risks remain hidden today?
- How should releases be defended?

- How should operations improve?
- What will we change first?

If participants cannot answer these questions, the training experience should be redesigned.

Core Commitment

The purpose of ETIS professional training is not to create course completions.

The purpose is to create organizational transformation.

Participants should leave with improved mental models, stronger engineering accountability, and a clearer understanding of what trustworthy engineering requires in the AI era.

The future of software engineering will not be determined by who adopts AI first.

It will be determined by who can use AI while preserving trust.

ETIS Quarter Sequence Guidance

The **ETIS Quarter Sequence Guidance** document provides recommendations for sequencing ETIS-based courses within quarter systems and other compressed academic calendars.

Quarter systems present unique educational challenges because instructors have less time while maintaining many of the same educational goals.

This document helps instructors compress schedules without compressing ETIS doctrine.

The objective is not to teach less ETIS.

The objective is to teach ETIS differently.

Purpose

The purpose of this document is to help instructors adapt ETIS to quarter-based environments while preserving engineering maturity progression.

This document helps instructors answer questions such as:

- What should be compressed?
- What should never be removed?
- How should assignments be combined?
- How should engineering maturity accelerate?
- How should repository expectations remain visible?
- How should AI governance be integrated?
- How should release readiness remain meaningful?
- How should final accountability be preserved?

Quarter systems require focus rather than simplification.

Guiding Principle

ETIS quarter systems should answer one question:

> How can engineering maturity be accelerated without losing engineering accountability?

Students should still progress through the ETIS transformation model.

Student ↓ Responsible Engineer ↓ Reviewer ↓ Architect ↓ Release Defender ↓ Operational Thinker ↓ Future Trustworthy Engineer

The timeline changes.

The transformation does not.

Quarter Philosophy

Quarter systems often create pressure to eliminate activities.

ETIS recommends eliminating redundancy instead.

Do not remove engineering responsibilities.

Instead, combine activities.

Traditional compression often looks like:

Remove Architecture

Remove Reviews

Remove Postmortems

Remove Release Readiness

ETIS compression should look like:

Combine Activities

Strengthen Deliverables

Reduce Administrative Overhead

Preserve Accountability

Focus should increase as time decreases.

The ETIS Quarter Architecture

Most quarter implementations should preserve six educational phases.

Phase 1 Engineering Foundations

↓

Phase 2 Intent And Planning

↓

Phase 3 Architecture And Context

↓

Phase 4 Construction Under Review

↓

Phase 5 Operations And Improvement

↓

Phase 6 Final Engineering Defense

These phases should remain visible.

They should simply be compressed.

Recommended Quarter Models

This document primarily supports:

- 10-week quarters,
- 11-week quarters,
- 12-week quarters,
- accelerated academic calendars,
- compressed institutional schedules.

The exact calendar matters less than preserving the engineering maturity arc.

ETIS Preservation Rules

The following components should never be removed.

Preserve Engineering Intent

Students must still define the problem before implementation.

Preserve Repository-Centered Engineering

Repository expectations should begin immediately.

Preserve AI Governance

AI accountability should be introduced early.

Preserve Architecture Review

Students should still practice engineering tradeoffs and decisions.

Preserve Engineering Evidence

Claims should remain connected to evidence.

Preserve Release Readiness

Students should still evaluate release defensibility.

Preserve Operational Thinking

Students should still think beyond implementation.

Preserve Final Accountability

Students should still defend engineering decisions.

Recommended 10-Week Quarter Sequence

Week 1 — Foundations And Repository Initialization

Educational goals:

- ETIS introduction
- engineering trust
- repository-centered engineering
- AI governance
- team formation

Deliverables:

- team charter
- repository initialization

Week 2 — Requirements, Planning, And Risk

Educational goals:

- intent
- stakeholders

- scope
- assumptions
- planning

Deliverables:

- requirements package
- risk register

Week 3 — Architecture And Decision Records

Educational goals:

- boundaries
- dependencies
- ADRs
- architecture reasoning

Deliverables:

- architecture package
- decision records

Weeks 4 And 5 — Construction Under Review

Educational goals:

- implementation
- reviews
- testing
- evidence

Deliverables:

- pull requests
- review evidence
- AI disclosures
- validation evidence

Week 6 — Cycle 1 Release Readiness

Educational goals:

- release evaluation
- evidence review
- risk communication

Deliverables:

- release readiness package

Week 7 — Postmortem And Stabilization

Educational goals:

- improvement
- reflection
- defect learning

Deliverables:

- postmortem package

Week 8 — Operations And Governance

Educational goals:

- observability
- supportability
- operational risks
- governance

Deliverables:

- operational readiness package

Week 9 — Final Release Preparation

Educational goals:

- repository refinement
- defense preparation
- evidence completion

Deliverables:

- defense package draft

Week 10 — Final Engineering Defense

Educational goals:

- accountability
- communication
- stewardship

Deliverables:

- final defense
- final repository

Assignment Compression Philosophy

Quarter systems should have fewer assignments.

However, each assignment should be stronger.

Example.

Avoid:

10 Small Assignments

Prefer:

5 Strong Phase Gates

Example:

Phase Gate 1

Repository Foundation

↓

Phase Gate 2

Requirements, Planning, Risk

↓

Phase Gate 3

Architecture And Construction

↓

Phase Gate 4

Release Readiness And Operations

↓

Phase Gate 5

Final Defense

Strengthen assignments rather than multiplying them.

Recommended Classroom Philosophy

Quarter systems should reduce lectures.

Increase engineering experiences.

Prioritize:

- reviews,
- critiques,
- workshops,
- exercises,
- simulations,
- tabletop discussions,
- defense rehearsals.

Students should spend more time engineering than listening.

AI Governance In Quarter Systems

AI governance should appear during Week 1.

Do not delay it.

Students will almost certainly begin using AI immediately.

Students should understand:

- AI is allowed.
- AI is governed.
- AI should be disclosed.
- AI should be verified.
- Humans own outcomes.

AI governance should become part of the course culture immediately.

Repository Expectations In Quarter Systems

Repository work should begin immediately.

Students should not wait several weeks before organizing repositories.

Repository expectations should be visible during:

- Week 1,
- every assignment,
- every review,
- release readiness,
- and final defense.

Repository discipline becomes even more important when time is compressed.

Assessment Philosophy

Quarter systems should simplify grading structures.

Assess fewer things.

Assess them deeply.

Recommended categories:

- engineering intent,
- evidence quality,
- architecture reasoning,
- AI accountability,
- review participation,
- release readiness,
- operational thinking,
- final accountability.

Avoid excessive grading granularity.

Recommended Instructor Role Progression

The instructor role should evolve throughout the quarter.

Weeks 1-2

Instructor

↓

Weeks 3-5

Coach

↓

Weeks 6-8

Reviewer

↓

Weeks 9-10

Engineering Defense Board

The instructor should progressively become less directive and more evaluative.

Common Quarter Mistakes

Mistake 1 — Removing Architecture

Architecture becomes more important under time pressure.

Mistake 2 — Delaying AI Governance

AI governance should begin immediately.

Mistake 3 — Skipping Release Readiness

Release readiness differentiates ETIS.

Mistake 4 — Eliminating Postmortems

Postmortems teach improvement.

Mistake 5 — Converting ETIS Into A Lecture Course

Students should still practice engineering.

Mistake 6 — Removing Final Defense

Final defense is one of the strongest ETIS experiences.

Mistake 7 — Overloading Students With Deliverables

Combine activities.

Strengthen artifacts.

Reduce redundancy.

Adaptation Rules

Quarter implementations may adapt pacing.

However, they should preserve:

- intent,
- context,
- evidence,
- review,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship,
- and final accountability.

These elements are constitutional ETIS components.

Guiding Standard

A successful ETIS quarter should still produce students who can answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How would the system be operated?
- Why is the release defensible?
- What would we improve next?

If students cannot answer these questions, compression has gone too far.

Core Commitment

The purpose of ETIS quarter sequencing is not to remove educational components.

The purpose is to preserve engineering maturity within compressed environments.

Time may shrink.

Engineering responsibility should not.

Students should still learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

That progression remains one of the foundations of ETIS education.

ETIS Schedule Adaptation Guide

The **ETIS Schedule Adaptation Guide** provides recommendations for adapting ETIS educational experiences to different institutions, calendars, audiences, and instructional environments while preserving ETIS doctrine.

This document does not provide schedules.

It provides adaptation principles.

Educational environments vary significantly across institutions and organizations.

ETIS is intentionally designed to be adaptable.

However, adaptation should not compromise the engineering philosophy that ETIS teaches.

The objective is to adapt implementation while preserving educational purpose.

Purpose

The purpose of this document is to help educators adapt ETIS responsibly.

This document helps instructors answer questions such as:

- How can ETIS fit my institutional environment?
- What components can be compressed?
- What components should never be removed?
- How should ETIS adapt to different learners?
- How should ETIS adapt to different academic calendars?
- How should ETIS adapt to professional training?
- How can ETIS remain recognizable across implementations?

Adaptation should strengthen ETIS adoption rather than dilute ETIS doctrine.

Guiding Principle

ETIS adaptation should answer one question:

> What can change without changing what ETIS fundamentally teaches?

Many implementation details may change.

The underlying educational philosophy should remain stable.

Adaptation should preserve engineering trust.

ETIS Adaptation Philosophy

Traditional educational adaptation often looks like this:

Environment Changes

↓

Remove Content

↓

Simplify Activities

↓

Reduce Expectations

ETIS adaptation should look like this:

Environment Changes

↓

Adjust Pacing

↓

Adjust Scope

↓

Preserve Accountability

↓

Preserve Educational Outcomes

Time constraints should not become accountability reductions.

What ETIS Is Optimizing For

ETIS is not optimizing for chapter completion.

ETIS is optimizing for professional formation.

Students should progressively learn how to:

- define intent,
- engineer context,
- bound authority,
- verify behavior,
- operate reality,
- explain decisions,
- and own outcomes.

These responsibilities should remain visible regardless of adaptation.

Constitutional ETIS Components

The following components should always remain visible.

Engineering Intent

Students should learn to define problems before building solutions.

Repository-Centered Engineering

Students should preserve engineering evidence.

AI Governance

Students should learn responsible AI-assisted engineering.

Architecture And Context

Students should reason about boundaries and decisions.

Reviews

Students should experience accountability through review.

Engineering Evidence

Students should connect claims to evidence.

Release Readiness

Students should evaluate release defensibility.

Operational Thinking

Students should think beyond implementation.

Stewardship

Students should understand long-term ownership.

Final Accountability

Students should defend engineering outcomes.

What May Be Adapted

Many implementation details may change.

Examples include:

- semester length,
- quarter systems,
- meeting frequency,
- class size,
- institutional requirements,
- technology choices,
- project size,
- grading models,
- instructional modalities.

These are implementation decisions.

They are not ETIS doctrine.

Adaptation Categories

Most adaptations fall into one of five categories.

Calendar Adaptation

↓

Audience Adaptation

↓

Project Adaptation

↓

Assessment Adaptation

↓

Delivery Adaptation

Each should be handled intentionally.

Calendar Adaptation

Examples include:

- 16-week semesters
- 15-week semesters
- 10-week quarters
- 8-week accelerated terms
- 6-week modules
- workshops

Adaptation Strategy Adjust pacing.

Do not remove educational stages.

Preserve:

- foundations,
- construction,
- validation,
- operations,
- defense.

Audience Adaptation

Different audiences require different emphases.

Examples include:

Undergraduate Students

Emphasize:

- foundations,
- construction,
- accountability.

Graduate Students

Emphasize:

- architecture,
- governance,
- professional judgment.

Capstone Students

Emphasize:

- release readiness,

- operations,
- final defense.

Professionals

Emphasize:

- organizational application,
- governance,
- operational trust.

Leaders

Emphasize:

- stewardship,
- governance,
- organizational accountability.

Project Adaptation

Projects may vary significantly.

Projects may become:

- smaller,
- larger,
- individual,
- team-based,
- simulated,
- organizational.

Project size may change.

The engineering accountability chain should remain visible.

Students should still create evidence.

Assessment Adaptation

Assessment models may change.

Examples include:

- percentage grading,
- standards-based grading,
- competency grading,
- portfolio assessment,
- pass/fail assessment.

The assessment mechanism may vary.

Professional accountability should remain visible.

Delivery Adaptation

Delivery environments may vary.

Examples include:

- face-to-face,
- hybrid,
- online,
- asynchronous,
- professional workshops.

The environment changes.

The educational philosophy should not.

Recommended Adaptation Process

Educators should adapt ETIS in the following order.

Step 1

Identify Environment

↓

Step 2

Identify Learner Type

↓

Step 3

Identify Transformation Goal

↓

Step 4

Select ETIS Components

↓

Step 5

Adjust Pacing

↓

Step 6

Adjust Scope

↓

Step 7

Verify ETIS Preservation

The preservation step is essential.

ETIS Preservation Checklist

Before finalizing an adaptation, ask:

Is engineering intent visible?

Is AI governance visible?

Is repository-centered engineering visible?

Is evidence visible?

Is review visible?

Is release readiness visible?

Is operational thinking visible?

Is accountability visible?

Is stewardship visible? If multiple answers are no, ETIS may be disappearing.

Recommended Compression Strategy

If time is limited:

Do this:

Combine Deliverables
↓
Strengthen Deliverables
↓
Reduce Redundancy
↓
Preserve Accountability

Do not do this:

Remove Architecture

Remove Reviews

Remove Operations

Remove Release Readiness

Compression should never remove professional formation.

Recommended Expansion Strategy

If additional time exists:

Expand:

- exercises,
- simulations,
- architecture critiques,
- release defenses,

- operational scenarios,
- postmortems,
- stewardship discussions.

Do not simply add more lectures.

Additional time should increase engineering experiences.

Common Adaptation Mistakes

Mistake 1 — Removing Operations

Students should understand life after deployment.

Mistake 2 — Delaying AI Governance

AI governance should appear early.

Mistake 3 — Reducing ETIS To Chapter Coverage

ETIS is a professional formation framework.

Mistake 4 — Overloading Students

Focus is more important than breadth.

Mistake 5 — Treating Repositories Like Submission Folders

Repositories are engineering systems.

Mistake 6 — Removing Final Accountability

Students should defend engineering outcomes.

Mistake 7 — Confusing Flexibility With Dilution

Adaptation should preserve doctrine.

Instructor Adaptation Questions

Instructors should periodically ask:

- What kind of engineer am I trying to create?
- What responsibilities should students own?
- What evidence should students produce?
- What should students defend?
- What should students improve?
- What should students steward?

The answers should drive adaptation decisions.

ETIS Adaptation Formula

ETIS adaptation can be summarized as:

```
Environment
    ↓
```

Audience

↓

Transformation Goal

↓

Capabilities

↓

Pacing

↓

Evidence

↓

Accountability

↓

Trustworthy Engineer

This formula should remain stable.

Guiding Standard

A successful adaptation should allow learners to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How would the system be operated?
- Why should this work be trusted?
- What would we improve next?

If learners cannot answer these questions, ETIS has been over-adapted.

Core Commitment

The purpose of ETIS adaptation is not to make ETIS easier to fit into existing environments.

The purpose is to preserve trustworthy engineering while making ETIS accessible across many environments.

Institutions may vary.

Calendars may vary.

Projects may vary.

Technologies may vary.

Engineering trust should not vary.

That consistency is one of the foundations of ETIS education.

ETIS Schedule Models Library

The **ETIS Schedule Models Library** provides reusable sequencing models for ETIS-based courses, modules, workshops, and professional learning experiences.

This file does not provide institution-specific calendars.

It provides adaptable models for sequencing engineering maturity over time.

ETIS schedules should be designed around professional formation, not merely around chapter coverage.

Purpose

The purpose of this library is to help instructors select a schedule model that fits their educational environment while preserving ETIS doctrine.

This library helps instructors plan:

- full-semester courses,
- quarter-based courses,
- accelerated courses,
- short modules,
- capstone sequences,
- graduate seminars,
- professional workshops,
- and organizational training programs.

Each model should help students progressively assume more engineering responsibility.

Guiding Principle

ETIS scheduling begins with one principle:

> Sequence engineering maturity before sequencing content.

A schedule should help learners move through a professional progression.

Engineering Foundations

↓

Engineering Construction

↓

Engineering Validation

↓

Engineering Operations

↓

Engineering Defense

Chapters, assignments, exercises, reviews, and assessments should support this progression.

Standard ETIS Maturity Sequence

Most ETIS schedules should include the following maturity stages.

Stage 1 — Engineering Foundations Students learn why ETIS exists, what trustworthy engineering requires, how repositories preserve evidence, and why AI-assisted work requires human accountability.

Stage 2 — Engineering Intent And Planning Students define the problem, requirements, scope, risks, team responsibilities, and initial evidence expectations.

Stage 3 — Architecture And Context Students reason about boundaries, dependencies, decisions, tradeoffs, AI context, and system structure.

Stage 4 — Construction Under Review Students implement under review, use AI responsibly, preserve evidence, test claims, and participate in disciplined team workflows.

Stage 5 — Validation And Release Readiness Students prepare evidence that supports claims about system behavior, quality, risk, and release defensibility.

Stage 6 — Operations And Improvement Students reason about observability, supportability, postmortems, operational readiness, reliability, and system improvement.

Stage 7 — Final Engineering Defense Students explain what they intended, what they built, how they verified it, what AI assisted, what risks remain, and why the release is defensible.

Model 1 — 15-Week Undergraduate Software Engineering Course

This is the standard ETIS semester model.

It is appropriate for a full undergraduate software engineering course with team projects.

Intended Use Use this model when ETIS is the primary course framework.

Sequence Weeks 1-2 Engineering Foundations

Weeks 3-4 Requirements, Planning, Risk, And Repository Foundation

Weeks 5-6 Architecture, Decisions, And Review

Weeks 7-9 Construction Under Review

Weeks 10-11 Cycle 1 Release Readiness

Weeks 12-13 Postmortem, Stabilization, Operations, And Governance

Weeks 14-15 Final Release Defense And Engineering Stewardship

Primary Student Experience Students progress from learning ETIS concepts to producing repository evidence, defending engineering decisions, and demonstrating release readiness.

Recommended Emphasis

- repository-centered engineering,
- team accountability,
- requirements and planning,
- architecture decisions,
- AI-use governance,
- reviews,

- testing,
- release readiness,
- operational maturity,
- final defense.

Model 2 — 16-Week Undergraduate Software Engineering Course

This model provides more breathing room than the 15-week model.

It is appropriate when the course includes a larger project, more review cycles, or more classroom exercises.

Sequence Weeks 1-2 Engineering Foundations

Weeks 3-4 Requirements, Planning, Risk, And Repository Foundation

Weeks 5-6 Architecture, Decisions, And Review

Weeks 7-9 Construction Under Review

Week 10 Cycle 1 Release Readiness

Week 11 Postmortem And Reflection

Weeks 12-13 Stabilization, Operations, Governance, And Improvement

Weeks 14-15 Final Release Preparation And Defense

Week 16 Stewardship Reflection And Course Closure

Primary Student Experience Students experience a stronger separation between construction, postmortem learning, operational improvement, and final defense.

Recommended Emphasis Use the extra time for:

- deeper architecture review,
- stronger postmortem work,
- operational readiness,
- release defense rehearsal,
- portfolio-style reflection.

Model 3 — 14-Week Compressed Semester

This model is appropriate for shorter semester structures or courses with limited instructional meetings.

Sequence Weeks 1-2 Engineering Foundations And Repository Foundation

Weeks 3-4 Requirements, Planning, Risk, And Architecture

Weeks 5-7 Construction Under Review

Weeks 8-9 Cycle 1 Release Readiness

Weeks 10-11 Postmortem, Stabilization, And Operational Thinking

Weeks 12-13 Final Release Preparation

Week 14 Final Engineering Defense

Primary Student Experience Students still experience the ETIS maturity arc, but some activities must be combined.

Recommended Emphasis Preserve:

- repository evidence,
- AI governance,
- review,
- release readiness,
- final defense.

Reduce:

- excessive lecture coverage,
- duplicate documentation,
- unnecessary intermediate deliverables.

Model 4 — 10-Week Quarter Course

This model is appropriate for quarter systems.

The pacing is faster and requires tighter assignment design.

Sequence Week 1 Engineering Foundations And Repository Setup

Week 2 Requirements, Planning, Risk, And AI Governance

Week 3 Architecture And Decision Review

Weeks 4-5 Construction Under Review

Week 6 Cycle 1 Release Readiness

Week 7 Postmortem And Stabilization

Week 8 Operations, Governance, And Risk

Week 9 Final Release Preparation

Week 10 Final Engineering Defense

Primary Student Experience Students move quickly from foundation to evidence production.

Recommended Emphasis Use fewer deliverables, but make each deliverable stronger.

Do not remove final defense.

Model 5 — 8-Week Accelerated Course

This model is appropriate for accelerated academic courses or compressed professional education.

Sequence Week 1 ETIS Foundations, Repository Setup, And AI Governance

Week 2 Requirements, Planning, And Risk

Week 3 Architecture And Decision Records

Weeks 4-5 Construction, Review, Testing, And Evidence

Week 6 Release Readiness And Postmortem

Week 7 Operations, Governance, And Improvement

Week 8 Final Defense

Primary Student Experience Students complete a compressed but coherent ETIS maturity path.

Recommended Emphasis Use a smaller project.

Reduce optional content.

Preserve the evidence chain.

Model 6 — 6-Week Module

This model is appropriate for a focused ETIS module embedded within another course.

Sequence Week 1 ETIS Foundations And Trustworthy Engineering

Week 2 Intent, Context, Requirements, And AI Governance

Week 3 Architecture, Decisions, And Review

Week 4 Validation, Evidence, And Release Readiness

Week 5 Operations, Risk, And Stewardship

Week 6 Mini Defense Or Final Review

Primary Student Experience Students experience ETIS as a focused professional framework rather than a full project lifecycle.

Recommended Emphasis Use small artifacts, case studies, or simulations.

Do not attempt a full-scale team project unless the course context supports it.

Model 7 — 4-Week Intensive Module

This model is appropriate for a short course unit, bootcamp segment, or concentrated learning experience.

Sequence Week 1 ETIS Foundations And AI Accountability

Week 2 Intent, Architecture, And Repository Evidence

Week 3 Review, Validation, Risk, And Release Readiness

Week 4 Operational Thinking And Final Defense

Primary Student Experience Students receive a high-impact introduction to ETIS concepts through concentrated practice.

Recommended Emphasis Use exercises, tabletop reviews, and short evidence artifacts.

Avoid large implementation projects.

Model 8 — 2-Week Mini Module

This model is appropriate for introducing ETIS inside an existing course.

Sequence Week 1 Trustworthy Engineering, AI Governance, And Repository Evidence

Week 2 Review, Release Readiness, Operational Thinking, And Defense

Primary Student Experience Students learn ETIS as a lens for evaluating engineering work.

Recommended Emphasis Use one case study and one review exercise.

Do not overload students with the full ETIS lifecycle.

Model 9 — Senior Capstone Sequence

This model is appropriate for project-based capstone courses.

Sequence Early Capstone Intent, Stakeholders, Requirements, Planning, And Repository Foundation

Mid Capstone Architecture, Decisions, Implementation, Review, And Validation

Late Capstone Release Readiness, Operational Thinking, Risk, And Final Defense

Post-Capstone Reflection, Stewardship, And Portfolio Evidence

Primary Student Experience Students use ETIS to make capstone work more reviewable, defensible, and professionally credible.

Recommended Emphasis

- stakeholder communication,
- evidence quality,
- architecture defense,
- AI-use disclosure,
- release readiness,
- risk communication,
- final defense.

Model 10 — Graduate Seminar Model

This model is appropriate for graduate courses emphasizing discussion, critique, architecture, governance, and professional judgment.

Sequence Segment 1 ETIS Doctrine, Trust, And Professional Responsibility

Segment 2 Architecture, Context, Governance, And AI Delegation

Segment 3 Reviewability, Evidence, Release Readiness, And Operations

Segment 4 Stewardship, Case Analysis, And Final Professional Defense

Primary Student Experience Students analyze, critique, and defend trustworthy engineering practices.

Recommended Emphasis Use fewer implementation assignments and more:

- case studies,
- architecture reviews,
- governance reviews,

- release reviews,
- professional defenses.

Model 11 — Professional Two-Day Workshop

This model is appropriate for organizational or corporate training.

Day 1 Morning ETIS Foundations, AI Accountability, And Trustworthy Engineering

Afternoon Repository Evidence, Architecture, Review, And Governance

Day 2 Morning Release Readiness, Operational Trust, Risk, And Incident Learning

Afternoon Team Application, Review Board Simulation, And Action Planning

Primary Participant Experience Participants leave with a practical model for improving AI-assisted engineering accountability.

Recommended Emphasis Use simulations and team application exercises.

Avoid academic assignment structures.

Model 12 — Professional Four-Week Cohort

This model is appropriate for working professionals meeting weekly.

Sequence Week 1 Trustworthy Engineering And AI Accountability

Week 2 Architecture, Context, And Evidence

Week 3 Review, Release Readiness, And Operational Trust

Week 4 Stewardship, Governance, And Team Action Plan

Primary Participant Experience Participants connect ETIS concepts directly to their organizational environment.

Recommended Emphasis Use real organizational scenarios where appropriate.

Produce a practical improvement plan.

Model 13 — Executive Briefing

This model is appropriate for leaders, governance stakeholders, and decision-makers.

Sequence Part 1 Why AI Increases Engineering Accountability

Part 2 What Trustworthy Engineering Requires

Part 3 Evidence, Reviewability, Governance, And Release Risk

Part 4 Organizational Stewardship And Next Steps

Primary Participant Experience Leaders understand why AI-era engineering requires disciplined governance, evidence, review systems, and operational accountability.

Recommended Emphasis Avoid technical depth.

Focus on governance responsibility, organizational risk, and decision quality.

Model Selection Guidance

Use this guidance when selecting a model.

Use A Full Semester Model When

- ETIS is the primary framework.
- Students complete a team project.
- Repository evidence is central.
- Final defense is possible.

Use A Quarter Or Accelerated Model When

- Time is limited.
- Assignments must be compressed.
- The project must be smaller.
- Phase gates must be tighter.

Use A Module Model When

- ETIS supports another course.
- One capability is primary.
- Students need exposure rather than full lifecycle practice.

Use A Professional Model When

- Participants are working practitioners.
- Organizational application matters.
- Practical action planning is more important than graded assignments.

Model Adaptation Rules

Instructors may adapt models to local constraints.

However, adaptations should preserve the maturity arc.

Do not remove:

- intent,
- context,
- evidence,
- review,
- AI accountability,
- validation,
- release readiness,
- operational thinking,
- and final accountability.

Pacing may change.

Doctrine should not.

Common Schedule Design Mistakes

Mistake 1 — Scheduling Chapters Instead Of Maturity ETIS courses should sequence responsibility, not page numbers.

Mistake 2 — Delaying Repository Work Too Long Repository expectations should begin early.

Mistake 3 — Adding AI Governance Too Late AI governance must be introduced before students use AI meaningfully.

Mistake 4 — Treating Release Readiness As A Final Presentation Release readiness is evidence-based accountability.

Mistake 5 — Skipping Postmortems Postmortems teach improvement and stewardship.

Mistake 6 — Removing Final Defense Final defense is one of the strongest ETIS learning experiences.

Final Standard

A strong ETIS schedule should make student maturity visible.

By the end of the course or training experience, learners should be able to explain:

- what they intended,
- what evidence they created,
- what decisions they made,
- how AI was governed,
- what was reviewed,
- what was verified,
- what risks remain,
- how the system could be operated,
- and why the work is defensible.

If the schedule does not build toward that accountability, it should be redesigned.

ETIS Semester Sequence Guidance

The **ETIS Semester Sequence Guidance** document provides recommendations for sequencing ETIS-based courses within traditional semester academic environments.

This document focuses on engineering maturity progression rather than calendar construction.

Semester schedules should intentionally guide students from foundational engineering concepts toward professional engineering accountability.

The objective is not to distribute content across weeks.

The objective is to progressively build trustworthy engineers.

Purpose

The purpose of this document is to help instructors design coherent ETIS semester experiences.

This document helps instructors answer questions such as:

- How should students mature over the semester?
- How should assignments be sequenced?
- When should AI governance be introduced?
- When should repository expectations be established?
- When should architecture reviews occur?
- When should release readiness appear?
- When should operational thinking be introduced?
- When should final engineering accountability occur?

The semester should feel like a progression rather than a collection of assignments.

Guiding Principle

ETIS semesters should answer one question:

> How should engineering responsibility increase over time?

Students should progressively move from:

Student

↓

Responsible Engineer

↓

Reviewer

↓

Architect

↓

Release Defender

↓

Operational Thinker

↓

Future Trustworthy Engineer

Every sequencing decision should support this progression.

Semester Philosophy

Traditional courses often follow a lecture-first model.

Lecture

↓

Homework

↓

Exam

ETIS semesters follow an engineering maturity model.

Engineering Foundations

↓

Engineering Intent

↓

Engineering Construction

↓

Engineering Validation

↓

Engineering Operations

↓

Engineering Defense

Students should feel their responsibilities increasing throughout the semester.

The ETIS Semester Architecture

Most ETIS semesters should contain six educational phases.

Phase 1

Engineering Foundations

↓

Phase 2

Intent And Planning

↓

Phase 3

Architecture And Context

↓

Phase 4

Construction Under Review

↓

Phase 5

Operations And Improvement

↓

Phase 6

Final Engineering Defense

This architecture should remain relatively stable regardless of semester length.

Phase 1 — Engineering Foundations

Educational Goal

Establish the educational philosophy.

Students should understand that software engineering has changed.

Topics may include:

- what ETIS is,
- repository-centered engineering,
- engineering evidence,
- AI accountability,
- engineering trust,
- professional responsibilities.

Students should understand:

> A demo is not operational proof.

And:

> AI proposes; engineers verify.

Recommended Activities

- ETIS introduction
- repository walkthrough
- starter kit introduction
- team formation
- AI governance discussion
- engineering maturity overview

Recommended Deliverables

- team charter
- repository initialization
- engineering expectations acknowledgement

Phase 2 — Intent And Planning

Educational Goal

Teach students to define work before building.

Topics may include:

- requirements,
- stakeholders,
- scope,
- assumptions,
- risks,
- planning,
- estimation.

Students should understand:

> Intent precedes implementation.

Recommended Activities

- requirements workshop
- ambiguity review
- risk identification
- planning exercises

Recommended Deliverables

- requirements package
- use cases
- planning artifacts
- risk register

Phase 3 — Architecture And Context

Educational Goal

Teach students to think architecturally.

Topics may include:

- boundaries,
- dependencies,
- context,
- architecture decisions,
- tradeoffs,
- reviews.

Students should understand:

> Governance is architecture.

Recommended Activities

- architecture critiques
- ADR workshops
- architecture reviews

Recommended Deliverables

- architecture diagrams
- ADRs
- architecture review evidence

Phase 4 — Construction Under Review

Educational Goal

Teach disciplined implementation.

Topics may include:

- pull requests,
- code reviews,
- testing,
- AI-assisted implementation,
- validation,
- engineering evidence.

Students should understand:

> Implementation is not engineering until it has been reviewed.

Recommended Activities

- pull request exercises
- review workshops
- testing evidence reviews

Recommended Deliverables

- implementation artifacts
- review evidence
- validation evidence
- AI disclosures

Phase 5 — Operations And Improvement

Educational Goal

Teach students that engineering continues after construction.

Topics may include:

- release readiness,
- postmortems,
- observability,
- operational readiness,
- governance,
- risks,
- continuous improvement.

Students should understand:

> The model is not the system.

Recommended Activities

- release readiness reviews
- postmortem workshops
- incident simulations

Recommended Deliverables

- release records
- postmortems
- operational evidence
- governance evidence

Phase 6 — Final Engineering Defense

Educational Goal

Create professional accountability.

Students should demonstrate that they can explain and defend engineering work.

Students should answer questions such as:

- What problem were you solving?
- What decisions did you make?
- How did AI assist?
- What evidence supports your claims?
- What risks remain?
- How would this system be operated?
- Why is this release defensible?

Students should understand:

> Engineering accountability is part of engineering itself.

Recommended Activities

- defense rehearsals
- engineering reviews
- final presentations

Recommended Deliverables

- defense package
- release package
- final repository

Semester Cadence Recommendations

Most semesters should maintain a consistent rhythm.

A healthy cadence may include:

```
Introduce Concepts
    ↓
Practice Concepts
```

↓

Review Concepts

↓

Produce Evidence

↓

Reflect And Improve

Students should experience this cycle repeatedly.

Two-Cycle Semester Model

Most ETIS semesters benefit from two engineering cycles.

Cycle 1 — Can It Work?

Focus:

- intent,
- planning,
- architecture,
- implementation,
- validation.

Deliverable:

Initial release evidence.

Cycle 2 — Can It Survive?

Focus:

- risks,
- defects,
- observability,
- governance,
- operations,
- improvement.

Deliverable:

Final release defense.

Students should clearly understand the distinction.

Semester Assessment Distribution Philosophy

Assessment should evolve throughout the semester.

Early semester:

Evaluate participation and foundations.

Middle semester:

Evaluate engineering process and evidence.

Late semester:

Evaluate accountability and defense.

Assessment maturity should mirror engineering maturity.

Recommended Semester Rhythm

Avoid introducing everything at once.

Instead:

Weeks 1-3
High Instructor Guidance
↓
Weeks 4-8
Shared Responsibility
↓
Weeks 9-12
Student Independence
↓
Weeks 13-15
Professional Accountability

The instructor should gradually become a reviewer rather than a director.

Relationship To The ETIS Book

Semester sequencing should not mirror chapter numbering.

Instead, chapters should support maturity phases.

Example:

Foundations
Parts I and II
↓
Operations
Part III
↓
Stewardship
Part IV

The book supports the semester.

The semester should not chase the book.

Common Semester Design Mistakes

Mistake 1 — Delaying Repository Expectations

Repositories should begin immediately.

Mistake 2 — Teaching AI Too Late

AI governance should appear early.

Mistake 3 — Teaching Operations Only At The End

Operational thinking should appear throughout the semester.

Mistake 4 — Treating Release Readiness As A Presentation

Release readiness is evidence-based accountability.

Mistake 5 — Skipping Postmortems

Postmortems teach stewardship.

Mistake 6 — Overloading Early Weeks

Progressively increase complexity.

Mistake 7 — Removing Final Defense

Final defense is one of the strongest ETIS learning experiences.

Adaptation Rules

Instructors may adapt semester pacing.

However, they should preserve:

- intent,
- context,
- evidence,
- review,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship,
- and final accountability.

These are constitutional ETIS elements.

Guiding Standard

A successful ETIS semester should produce students who can answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How would the system be operated?

- Why is the release defensible?
- What would we improve next?

If students cannot answer these questions, the semester needs redesign.

Core Commitment

The purpose of ETIS semester sequencing is not to optimize calendars.

The purpose is to intentionally engineer the formation of trustworthy engineers.

Students should progressively learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

The semester itself should make that progression visible.

Part IV

Assignment Sequence Guidance

ETIS Assessment Progression Guide

The **ETIS Assessment Progression Guide** provides recommendations for how assessment should progressively evolve throughout ETIS educational experiences.

This document does not define grading systems.

It defines assessment maturity systems.

ETIS intentionally moves beyond assessing technical output alone.

Students should progressively be assessed on engineering accountability.

The objective is not to determine whether students can build software.

The objective is to determine whether students can engineer trustworthy systems.

Assessment should reflect that progression.

Purpose

The purpose of this document is to help instructors intentionally assess engineering maturity.

This document helps instructors answer questions such as:

- What should be assessed early?
- What should be assessed later?
- How should assessment evolve?
- How should engineering accountability be measured?
- How should AI accountability be evaluated?
- How should evidence quality be evaluated?
- How should students demonstrate professional growth?

Assessment should progressively mirror engineering maturity.

Guiding Principle

ETIS assessment should answer one question:

> What engineering responsibility should learners now be able to own?

Students should progressively demonstrate ownership rather than completion.

Assessment should support that progression.

Educational Philosophy

Traditional software engineering assessment often emphasizes outputs.

Examples include:

- assignments,
- quizzes,
- exams,
- projects,
- presentations.

ETIS intentionally expands assessment.

Students should also be evaluated on:

- engineering intent,
- evidence quality,
- review participation,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship,
- professional ownership.

Educational assessment should resemble professional engineering evaluation.

Foundational Principle

Assessment maturity should mirror engineering maturity.

Students should progressively move from following instructions to owning engineering outcomes.

Assessment should evolve accordingly.

ETIS Assessment Maturity Model

Most ETIS educational experiences should progress through five assessment stages.

Participation
↓
Contribution
↓
Decision Making
↓
Accountability
↓
Professional Ownership

Students should feel this progression throughout the educational experience.

Stage 1 — Participation

Educational Goal

Establish foundational expectations.

Students are learning ETIS concepts.

Assessment should focus on engagement.

Examples include:

- participation,
- repository setup,
- team formation,
- engineering discussions,
- initial exercises.

Questions include:

- Is the student engaged?
- Is the student participating?
- Is the student understanding expectations?

Assessment should be supportive rather than punitive.

Stage 2 — Contribution

Educational Goal

Assess engineering contributions.

Students begin creating engineering artifacts.

Examples include:

- requirements,
- planning artifacts,
- risk identification,
- architecture participation.

Questions include:

- Is the student contributing meaningful work?
- Is the student participating professionally?
- Is the student preserving evidence?

Assessment should encourage ownership.

Stage 3 — Decision Making

Educational Goal

Assess engineering reasoning.

Students begin making engineering decisions.

Examples include:

- architecture decisions,
- AI decisions,
- tradeoff analysis,
- review participation.

Questions include:

- Can the student explain decisions?
- Can the student justify tradeoffs?
- Can the student identify risks?

Assessment should increasingly evaluate thinking.

Stage 4 — Accountability

Educational Goal

Assess engineering accountability.

Students begin defending engineering work.

Examples include:

- release readiness,
- evidence quality,
- risk communication,
- operational thinking.

Questions include:

- What evidence supports claims?
- What risks remain?
- Is the release defensible?

Assessment should increasingly evaluate responsibility.

Stage 5 — Professional Ownership

Educational Goal

Assess full engineering ownership.

Students should defend the entire engineering journey.

Questions include:

- What problem was solved?
- What decisions were made?
- What evidence exists?
- How did AI assist?
- What did humans verify?
- What risks remain?
- Why should this work be trusted?

Assessment should resemble professional evaluation.

Recommended Assessment Categories

ETIS educational environments should assess multiple dimensions.

Examples include:

Engineering Intent

Can students define the problem?

Evidence Quality

Does evidence support engineering claims?

Architecture Reasoning

Can students explain decisions?

Review Participation

Can students participate in engineering reviews?

AI Accountability

Can students responsibly use AI?

Release Readiness

Can students defend releases?

Operational Thinking

Can students think beyond implementation?

Professional Ownership

Can students own outcomes?

Assessment Growth Model

Assessment emphasis should evolve throughout the educational experience.

Early Course
Participation
↓
Middle Course
Decision Making
↓
Late Course
Accountability
↓
Final Stage
Professional Ownership

Students should experience this progression.

Assessment Should Shift From Output To Ownership

Traditional assessment often asks:

> What did you build?

ETIS assessment should increasingly ask:

> Why should another engineer trust what you built?

This distinction should remain visible.

AI Assessment Philosophy

AI changes assessment.

Students should not simply be evaluated on outputs.

Students should also be evaluated on:

- AI disclosures,
- AI verification,
- AI limitations,
- AI governance decisions.

Students should understand:

AI proposes; engineers verify.

Assessment should reinforce this behavior.

Repository Assessment Philosophy

Repositories should become assessable engineering systems.

Questions include:

- Is engineering evidence visible?
- Are decisions visible?
- Are reviews visible?
- Are risks visible?
- Is AI use transparent?

Repositories should become part of assessment.

Assessment And Final Defense

Final defenses should become one of the strongest ETIS assessment mechanisms.

Students should explain:

- intent,
- evidence,
- decisions,
- AI involvement,
- risks,
- operational considerations,
- stewardship considerations.

Students should experience authentic accountability.

Recommended Instructor Role Progression

Instructor responsibilities should evolve.

Early
Teacher
↓
Middle
Coach
↓

Later

Reviewer

↓

Final

Engineering Defense Board

Assessment should mirror this progression.

Common Assessment Mistakes

Mistake 1 — Over-Assessing Technical Output

Technical output is only one dimension.

Mistake 2 — Ignoring Evidence Quality

Evidence is foundational.

Mistake 3 — Ignoring AI Accountability

AI accountability should be visible.

Mistake 4 — Delaying Operational Thinking

Operations should appear before the end.

Mistake 5 — Grading Individual Artifacts In Isolation

Educational experiences should be cumulative.

Mistake 6 — Overweighting Presentations

Presentations summarize engineering work.

They should not replace evidence.

Mistake 7 — Waiting Until The End To Evaluate Ownership

Ownership should progressively increase.

Adaptation Rules

Educational environments may adapt grading systems.

Examples include:

- percentage grading,
- standards-based grading,
- competency grading,
- portfolio grading,
- pass/fail grading.

However, they should preserve assessment of:

- engineering intent,
- evidence,

- review,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship,
- professional ownership.

These remain constitutional ETIS capabilities.

Instructor Assessment Questions

Instructors should periodically ask:

- What engineering behavior am I assessing?
- What evidence supports this assessment?
- What decisions should students explain?
- What risks should students communicate?
- What ownership should students demonstrate?

The answers should drive assessment design.

Guiding Standard

A successful ETIS educational experience should allow students to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?
- What would we improve next?

Assessment should reinforce these questions throughout the educational experience.

Core Commitment

The purpose of ETIS assessment progression is not to create more grading mechanisms.

The purpose is to create stronger engineers.

Assessment should progressively help students learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

That progression is one of the foundations of ETIS education.

ETIS Assignment Assembly Guide

The **ETIS Assignment Assembly Guide** provides a repeatable process for designing ETIS educational experiences.

This guide does not teach instructors how to create assignments.

It teaches instructors how to engineer professional accountability experiences.

Educational work should resemble professional engineering work.

Students should progressively assume responsibility for engineering outcomes.

The objective is not assignment completion.

The objective is professional formation.

Purpose

The purpose of this document is to help instructors build ETIS educational experiences intentionally.

This guide helps instructors answer questions such as:

- What engineering responsibility am I teaching?
- What evidence should students create?
- What decisions should students make?
- What should students review?
- What should students defend?
- What should students improve?
- How should accountability increase over time?

The resulting educational experience should feel cumulative rather than fragmented.

Guiding Principle

Every ETIS educational experience should answer one question:

> What new engineering responsibility should students now be able to own?

That question should drive every educational design decision.

Assignments should support accountability.

Assignments should not become the educational objective.

Educational Philosophy

Traditional assignment design often looks like this:

```
Topic
  ↓
Instructions
  ↓
Submission
  ↓
```

Grade

ETIS assignment design should look like this:

Engineering Responsibility

↓

Engineering Activity

↓

Evidence Creation

↓

Review

↓

Accountability

↓

Reflection

Educational work should progressively resemble professional engineering work.

ETIS Assignment Assembly Pipeline

ETIS educational experiences should be assembled in the following order.

Professional Transformation Goal

↓

Engineering Responsibility

↓

Educational Capability

↓

Engineering Activities

↓

Evidence Creation

↓

Review Mechanisms

↓

Accountability Questions

↓

Reflection

↓

Phase Gate

This order should remain stable.

Step 1 — Define The Professional Transformation Goal

Begin by identifying who learners should become.

Examples include:

Student
↓
Responsible Engineer

Student
↓
Reviewer

Student
↓
Architect

Student
↓
Release Defender

Student
↓
Operational Thinker

The transformation goal should remain visible.

Step 2 — Define The Engineering Responsibility

Identify what learners should now own.

Examples include:

- defining requirements,
- making architectural decisions,
- reviewing engineering work,
- validating claims,
- evaluating release readiness,
- operating systems,
- communicating risks.

This responsibility becomes the center of the educational experience.

Step 3 — Define The Educational Capability

Identify the ETIS capability being strengthened.

Examples include:

- engineering intent,
- context engineering,
- architecture,
- review participation,
- AI governance,
- release readiness,
- operational thinking,
- stewardship.

One primary capability should exist.

Additional capabilities may support it.

Step 4 — Design Engineering Activities

Activities should simulate real engineering environments.

Examples include:

- requirements workshops,
- architecture critiques,
- review exercises,
- pull requests,
- testing activities,
- release evaluations,
- incident simulations,
- postmortems.

Avoid creating activities that only exist for grading purposes.

Students should practice engineering.

Step 5 — Define Evidence Creation

Students should create evidence that supports engineering claims.

Examples include:

- requirements,
- architecture diagrams,
- ADRs,
- AI disclosures,
- review records,
- validation evidence,
- release evidence,
- operational analyses.

Evidence should become a first-class educational artifact.

Step 6 — Design Review Mechanisms

Students should experience accountability through reviews.

Examples include:

- peer reviews,

- architecture reviews,
- repository reviews,
- evidence reviews,
- release reviews,
- instructor reviews.

Reviews should become normal engineering behaviors.

Step 7 — Define Accountability Questions

Every educational experience should ask one or more accountability questions.

Examples include:

- What problem are we solving?
- Why did we choose this approach?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- Why should this work be trusted?

These questions should become increasingly sophisticated over time.

Step 8 — Build Reflection Activities

Students should explain what they learned.

Examples include:

- lessons learned,
- improvement plans,
- postmortems,
- risk reflections,
- operational reflections.

Reflection should strengthen engineering maturity.

Step 9 — Assemble The Phase Gate

Finally, combine all components into a cumulative educational experience.

Every phase gate should inherit previous work.

Nothing should start over.

Students should continuously improve engineering artifacts.

Recommended Assignment Blueprint

Every ETIS assignment should contain these sections.

Purpose
↓
Engineering Responsibility

↓

Learning Objectives

↓

Engineering Activities

↓

Required Evidence

↓

Review Expectations

↓

AI Expectations

↓

Accountability Questions

↓

Reflection Expectations

↓

Submission Expectations

This blueprint should remain relatively stable.

AI Expectations

Every educational experience should explicitly address AI.

Students should understand:

- AI is allowed.
- AI is governed.
- AI should be disclosed.
- AI should be verified.
- Humans own outcomes.

AI governance should never be implied.

It should be explicit.

Repository Expectations

Educational work should live inside repositories.

Repositories should preserve:

- evidence,
- decisions,
- risks,
- reviews,
- AI use,
- engineering history.

Repositories are engineering systems.

They are not submission folders.

Recommended Instructor Role Progression

Instructor roles should evolve.

Early
Teacher
↓
Middle
Coach
↓
Later
Reviewer
↓
Final
Engineering Defense Board

Students should progressively assume ownership.

Assignment Complexity Growth Model

Educational complexity should increase gradually.

Follow Instructions
↓
Make Decisions
↓
Explain Decisions
↓
Defend Decisions
↓
Own Outcomes

Students should feel this progression.

Common Assignment Design Mistakes

Mistake 1 — Starting With Deliverables

Start with engineering responsibilities.

Mistake 2 — Creating Isolated Assignments

Assignments should inherit previous work.

Mistake 3 — Creating Artificial Activities

Educational work should resemble engineering work.

Mistake 4 — Delaying AI Governance

AI expectations should be visible immediately.

Mistake 5 — Ignoring Reviews

Reviews are engineering mechanisms.

Mistake 6 — Treating Repositories As Submission Systems

Repositories are engineering systems.

Mistake 7 — Grading Completion Instead Of Accountability

Professional ownership should remain visible.

Adaptation Rules

Educational environments may adapt implementation details.

However, they should preserve:

- engineering intent,
- evidence,
- review,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship,
- final accountability.

These remain constitutional ETIS capabilities.

Instructor Design Questions

Before finalizing an ETIS educational experience, ask:

- What responsibility am I teaching?
- What evidence should students create?
- What should students review?
- What should students defend?
- What should students improve?
- What should students own?

These answers should drive design decisions.

Guiding Standard

A successful ETIS assignment should allow students to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?
- What would we improve next?

If students cannot answer these questions, the educational experience should be redesigned.

Core Commitment

The purpose of ETIS assignment assembly is not to create stronger assignments.

The purpose is to create stronger engineers.

Educational work should progressively resemble professional engineering work.

Students should learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

That progression is one of the foundations of ETIS education.

ETIS Assignment Philosophy

The **ETIS Assignment Philosophy** document defines how ETIS organizes educational work.

ETIS intentionally moves beyond traditional assignment-centered education.

Educational experiences should progressively develop engineering accountability rather than simply distribute work throughout a semester.

Students should experience engineering as a cumulative professional activity rather than a sequence of independent academic exercises.

The objective is not to help students complete assignments.

The objective is to help students become trustworthy engineers.

Purpose

The purpose of this document is to establish a shared philosophy for organizing educational work within ETIS implementations.

This document helps instructors answer questions such as:

- Why does ETIS use phase gates?
- Why do assignments build upon one another?
- Why should evidence accumulate?
- Why should students repeatedly refine artifacts?
- Why should accountability increase over time?
- Why should educational experiences resemble professional engineering?

This document establishes the educational doctrine that supports ETIS assignment systems.

Guiding Principle

ETIS educational work should answer one question:

> What new engineering responsibility should students now be able to own?

Educational experiences should progressively increase responsibility.

Every activity should support that progression.

Educational Philosophy

Traditional courses often organize work like this:

Assignment 1

↓

Assignment 2

↓

Assignment 3

↓

Assignment 4

↓

Final Project

Each assignment often begins and ends independently.

Students complete work and move on.

Professional engineering rarely works this way.

ETIS intentionally organizes work differently.

Repository Foundation

↓

Intent

↓

Architecture

↓

Construction

↓

Release Readiness

↓

Operations

↓

Engineering Defense

Every stage inherits from previous work.

Nothing truly starts over.

Educational work should resemble professional engineering work.

Foundational Principle

Assignments are temporary.

Phase gates are cumulative.

Students should repeatedly improve engineering artifacts rather than repeatedly create new artifacts.

Growth should be visible.

Learning should be visible.

Engineering maturity should be visible.

Professional Formation Model

ETIS organizes educational experiences around professional transformation.

Students should progressively evolve through stages.

Student

↓

Responsible Engineer

↓

Reviewer

↓

Architect

↓

Release Defender

↓

Operational Thinker

↓

Future Trustworthy Engineer

Educational work should intentionally support this progression.

ETIS Organizes Accountability Rather Than Work

Traditional education often organizes around tasks.

Examples include:

- homework,
- quizzes,
- projects,
- presentations.

ETIS organizes around accountability.

Examples include:

- ownership,
- evidence,
- review,
- validation,
- release decisions,
- operational thinking,
- stewardship.

Students should progressively assume these responsibilities.

The Educational Accountability Chain

Most ETIS educational experiences should progressively expose students to the following chain.

Intent

↓

Context

↓

Architecture

↓

Construction

↓

Validation

↓

Release

↓

Operations

↓

Stewardship

↓

Defense

This chain should remain visible.

Educational Work Should Accumulate

ETIS intentionally avoids educational resets.

Traditional assignments often create artificial boundaries.

Example:

Week 2 Assignment

↓

Complete

↓

Discard

↓

Week 3 Assignment

ETIS promotes continuous refinement.

Example:

Intent

↓

Architecture

↓

Construction

↓

Release

↓

Operations

↓

Defense

Earlier work should remain visible.

Students should continuously improve artifacts.

Educational Artifacts Should Mature

Students should not repeatedly recreate documents.

Instead, artifacts should evolve.

Examples include:

Requirements

↓

Refined Requirements

↓

Architecture Decisions

↓

Review Evidence

↓

Release Evidence

↓

Operational Evidence

↓

Defense Evidence

The repository should tell a coherent engineering story.

Evidence Is A First-Class Educational Asset

ETIS treats evidence as a primary educational outcome.

Students should preserve evidence throughout educational experiences.

Examples include:

- requirements,
- architecture decisions,
- reviews,
- AI disclosures,
- validation results,
- release decisions,
- postmortems,

- operational analyses.

Evidence should support claims.

AI Changes Educational Expectations

AI fundamentally changes educational assumptions.

Educational systems should adapt accordingly.

ETIS teaches:

> AI proposes; engineers verify.

Students should learn:

- responsible AI use,
- verification,
- disclosure,
- bounded authority,
- accountability.

AI should increase engineering maturity rather than replace engineering thinking.

Educational Work Should Be Reviewable

Professional engineering depends upon reviews.

Educational environments should expose students to reviews early.

Examples include:

- peer reviews,
- architecture reviews,
- release reviews,
- evidence reviews,
- AI reviews.

Students should experience accountability through review systems.

Educational Work Should Be Defensible

Students should repeatedly answer questions such as:

- What problem are we solving?
- Why did we choose this design?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- Why should this work be trusted?

Students should become comfortable defending engineering decisions.

Educational Work Should Extend Beyond Implementation

Traditional software engineering courses often stop at implementation.

ETIS intentionally extends beyond implementation.

Students should also experience:

- release readiness,
- observability,
- postmortems,
- operational thinking,
- stewardship.

Students should understand:

> The model is not the system.

And:

> A demo is not operational proof.

Educational Work Should Produce Engineering Memory

Educational artifacts should preserve engineering memory.

Repositories should answer questions such as:

- What decisions were made?
- Why were they made?
- What changed?
- What evidence exists?
- What risks remain?

Students should understand that repositories are engineering systems.

Educational Work Should Increase Student Ownership

Instructor involvement should gradually decrease.

Student ownership should gradually increase.

Early Course

Instructor Guidance

↓

Middle Course

Shared Responsibility

↓

Late Course

Student Ownership

↓

Final Stage

Professional Accountability

Students should feel this progression.

Educational Work Should Feel Real

Students should experience authentic engineering environments.

Examples include:

- ambiguity,
- tradeoffs,
- constraints,
- reviews,
- revisions,
- risks,
- imperfect information.

Educational experiences should prepare students for reality.

Common Educational Misconceptions

Misconception 1 — More Assignments Create More Learning

Depth often creates more learning than quantity.

Misconception 2 — Students Should Start Fresh Frequently

Professional engineering rarely works this way.

Misconception 3 — A Working Demo Is Success

A demo is not operational proof.

Misconception 4 — AI Simplifies Education

AI increases accountability.

Misconception 5 — Repositories Are Submission Systems

Repositories are engineering systems.

Misconception 6 — Reviews Are Optional

Reviews are engineering mechanisms.

Misconception 7 — Courses End At Implementation

Engineering continues after implementation.

Instructor Design Questions

Instructors should periodically ask:

- What engineering responsibility am I teaching?
- What evidence should students create?
- What should students review?
- What should students defend?
- What should students improve?
- What should students steward?

These questions should drive educational design.

Adaptation Rules

Educational environments may adapt implementations.

However, they should preserve:

- engineering intent,
- evidence,
- review,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship,
- final accountability.

These remain constitutional ETIS concepts.

Guiding Standard

A successful ETIS educational experience should allow students to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?
- What would we improve next?

If students cannot answer these questions, educational design should be reconsidered.

Core Commitment

The purpose of ETIS assignment philosophy is not to create stronger assignments.

The purpose is to create stronger engineers.

Educational work should resemble professional engineering work.

Students should progressively learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

That progression is one of the foundations of ETIS education.

ETIS Assignment Scaling Guide

The **ETIS Assignment Scaling Guide** provides recommendations for scaling ETIS educational experiences across different environments while preserving engineering accountability.

This document does not provide alternate assignment sets.

It provides scaling principles.

Educational environments vary considerably.

Enrollment sizes differ.

Calendar structures differ.

Projects differ.

Learner populations differ.

ETIS should adapt to these differences without sacrificing professional formation.

The objective is to scale complexity rather than scale accountability.

Purpose

The purpose of this document is to help instructors scale ETIS educational experiences responsibly.

This document helps instructors answer questions such as:

- How should ETIS scale to different class sizes?
- How should ETIS scale to different calendars?
- How should ETIS scale to different learner populations?
- How should ETIS scale to professional training?
- What should change?
- What should remain constant?
- How should accountability remain visible?

Scaling should preserve educational integrity.

Guiding Principle

ETIS assignment scaling should answer one question:

> What can become simpler without reducing engineering accountability?

Complexity may change.

Accountability should not.

Educational Philosophy

Traditional scaling often looks like this:

Environment Changes

↓

Reduce Assignments

↓

Reduce Expectations

↓

Reduce Accountability

ETIS scaling should look like this:

Environment Changes

↓

Adjust Complexity

↓

Adjust Scope

↓

Preserve Accountability

↓

Preserve Professional Formation

Educational quality should remain stable.

Foundational Principle

Scale complexity, not accountability.

Students should still learn how to:

- define intent,
- engineer context,
- bound authority,
- verify behavior,
- operate reality,
- explain decisions,
- and own outcomes.

These responsibilities should remain visible.

What Should Never Change

The following ETIS elements should remain visible regardless of scale.

Engineering Intent

Students should define problems before implementation.

Repository-Centered Engineering

Students should preserve engineering evidence.

AI Governance

Students should practice responsible AI use.

Reviews

Students should experience accountability.

Engineering Evidence

Students should support claims with evidence.

Release Readiness

Students should evaluate release decisions.

Operational Thinking

Students should think beyond implementation.

Professional Ownership

Students should defend engineering outcomes.

Scaling Dimensions

Most scaling occurs across five dimensions.

```
Enrollment
   ↓
Calendar
   ↓
Learner Maturity
   ↓
Project Complexity
   ↓
Delivery Environment
```

Each dimension should be handled intentionally.

Scaling Dimension 1 — Enrollment Size

Small Classes

Examples:

10-20 students

Recommended emphasis:

- deeper discussions,
- richer reviews,
- more individual accountability,
- stronger defenses.

Students can experience highly personalized reviews.

Medium Classes

Examples:

20-50 students

Recommended emphasis:

- team structures,
- peer reviews,
- structured phase gates,
- reusable templates.

Balance instructor involvement and student ownership.

Large Classes

Examples:

50-120 students

Recommended emphasis:

- standardized phase gates,
- review rubrics,
- structured teams,
- scalable evidence systems.

Use repeatable systems.

Do not reduce accountability.

Scaling Dimension 2 — Calendar Length

Examples include:

- 16-week semesters,
- 15-week semesters,
- 10-week quarters,
- accelerated terms,
- modules.

Scaling strategy:

Adjust pacing.

Do not remove educational stages.

Preserve:

- foundations,
- construction,
- validation,
- operations,
- defense.

Scaling Dimension 3 — Learner Maturity

Different learners require different emphases.

Undergraduate Students

Increase structure.

Provide more guidance.

Graduate Students

Increase autonomy.

Increase critique.

Increase professional judgment.

Capstone Students

Increase accountability.

Increase operational thinking.

Increase release readiness.

Professionals

Increase organizational application.

Reduce academic overhead.

Leaders

Increase governance emphasis.

Reduce technical depth.

Scaling Dimension 4 — Project Complexity

Project complexity may change.

Examples include:

Small Projects

Examples:

- simulations,
- case studies,
- contained systems.

Emphasize evidence.

Medium Projects

Examples:

- team applications,
- semester systems.

Balance complexity and accountability.

Large Projects

Examples:

- capstones,
- multi-team systems.

Increase governance.

Increase review structures.

Increase release readiness.

Do not simply increase deliverables.

Scaling Dimension 5 — Delivery Environment

Examples include:

Face-To-Face

Emphasize workshops and reviews.

Hybrid

Emphasize repositories and asynchronous evidence.

Online

Emphasize written communication and evidence systems.

Asynchronous

Emphasize structured reviews and accountability checkpoints.

Professional Training

Emphasize immediate application.

The delivery environment may change.

The philosophy should not.

Recommended Scaling Strategies

When scaling up, prefer:

More Templates

↓

More Rubrics

↓

More Peer Reviews

↓

More Reusable Systems

Avoid:

More Assignments

↓

More Documentation
↓
More Administrative Overhead

Scalability should reduce friction.

Complexity Scaling Recommendations

Scale these components.

Project Scope

May change.

Technical Complexity

May change.

Number Of Features

May change.

Team Size

May change.

Lecture Time

May change.

Number Of Exercises

May change.

These are implementation decisions.

Accountability Scaling Recommendations

Do not scale these components downward.

Evidence

Should remain visible.

Reviews

Should remain visible.

AI Governance

Should remain visible.

Risk Communication

Should remain visible.

Release Readiness

Should remain visible.

Final Accountability

Should remain visible.

These are constitutional ETIS elements.

Instructor Scaling Philosophy

Instructors should gradually transition roles.

Small Courses
Coach
↓
Medium Courses
Facilitator
↓
Large Courses
System Designer

The instructor increasingly builds systems that allow accountability to scale.

Common Scaling Mistakes

Mistake 1 — Reducing Accountability

Scale complexity instead.

Mistake 2 — Adding More Assignments

Focus on stronger phase gates.

Mistake 3 — Eliminating Reviews

Reviews are engineering mechanisms.

Mistake 4 — Removing Operational Thinking

Students should understand life after implementation.

Mistake 5 — Delaying AI Governance

AI governance should appear immediately.

Mistake 6 — Overbuilding Administrative Processes

Engineering systems should remain elegant.

Mistake 7 — Confusing Bigger Courses With Different ETIS

ETIS should remain recognizable.

Instructor Scaling Questions

Instructors should periodically ask:

- What complexity should change?
- What accountability should remain?
- What evidence should students create?
- What reviews should students experience?
- What ownership should students demonstrate?

These answers should drive scaling decisions.

ETIS Scaling Formula

ETIS scaling can be summarized as:

Environment
↓
Complexity
↓
Evidence
↓
Review
↓
Accountability
↓
Trustworthy Engineer

This formula should remain stable.

Guiding Standard

A successful ETIS scaling effort should still allow learners to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?
- What would we improve next?

If learners cannot answer these questions, ETIS has been over-scaled.

Core Commitment

The purpose of ETIS assignment scaling is not to make ETIS easier to administer.

The purpose is to preserve trustworthy engineering across many educational environments.

Institutions may vary.

Learners may vary.

Projects may vary.

Technologies may vary.

Engineering trust should not vary.

That consistency is one of the foundations of ETIS education.

ETIS COMP330 Phase Gate Reference

The **ETIS COMP330 Phase Gate Reference** document preserves the flagship ETIS educational implementation currently used within Loyola University Chicago COMP330.

This document exists as a reference implementation.

It is not ETIS doctrine.

COMP330 consumes ETIS educational assets.

COMP330 also contributes proven educational assets back into the ETIS educational ecosystem.

The purpose of this document is to preserve one successful implementation model that future educators may study, adapt, and improve.

This document intentionally preserves educational provenance.

Purpose

The purpose of this document is to help educators understand how ETIS can be implemented within a semester-long undergraduate software engineering course.

This document helps instructors answer questions such as:

- What does ETIS look like in practice?
- How can phase gates be organized?
- How does accountability increase over time?
- How does evidence accumulate?
- How do release defenses emerge?
- How can AI governance be integrated?
- How can repositories become engineering systems?

COMP330 serves as one example.

It is not the only example.

Provenance

Institution:

Loyola University Chicago

Course:

COMP330 Software Engineering

Educational Role:

Flagship ETIS implementation

Educational Purpose:

Teach students how to engineer systems that can be understood, reviewed, governed, operated, improved, and trusted over time.

COMP330 Educational Philosophy

COMP330 intentionally moves beyond traditional software engineering education.

The course is not organized around isolated assignments.

The course is organized around cumulative engineering accountability.

Students progressively mature throughout the semester.

Educational work continuously evolves.

Nothing truly starts over.

The course intentionally resembles professional engineering environments.

COMP330 Professional Transformation Model

Students progressively evolve throughout the semester.

Student

↓

Responsible Engineer

↓

Reviewer

↓

Architect

↓

Release Defender

↓

Operational Thinker

↓

Future Trustworthy Engineer

The phase gates intentionally support this progression.

COMP330 Six Phase Gate Architecture

The COMP330 implementation uses six cumulative phase gates.

Phase Gate 1

Project Launch Evidence Package

↓

Phase Gate 2

Cycle 1 Planning And Estimation Package

↓

Phase Gate 3

Cycle 1 Architecture And Review Package

↓

Phase Gate 4

Cycle 1 Construction, Integration, And AI Code Review Package

↓

Phase Gate 5

Cycle 1 Release Presentation And Peer Review Package

↓

Phase Gate 6

Final Cycle Release Presentation And Peer Review Package

Each phase gate inherits previous work.

Nothing is independent.

Phase Gate 1 — Project Launch Evidence Package

Educational Goal

Establish engineering foundations.

Students establish:

- teams,
- repositories,
- expectations,
- responsibilities,
- engineering culture.

Students begin learning:

- repository-centered engineering,
- AI governance,
- evidence-centered engineering.

Typical Artifacts

Examples may include:

- team charter,
- repository initialization,
- team expectations,
- engineering norms.

Accountability Question

> How will this team engineer together?

Phase Gate 2 — Cycle 1 Planning And Estimation Package

Educational Goal

Teach engineering intent before implementation.

Students learn:

- requirements,
- use cases,
- planning,
- estimation,
- risk awareness.

Typical Artifacts

Examples may include:

- executive summary,
- requirements package,
- use cases,
- WBS,
- RACI matrix,
- QA log.

Accountability Question

> Have we clearly defined the work?

Phase Gate 3 — Cycle 1 Architecture And Review Package

Educational Goal

Teach architectural thinking.

Students learn:

- system boundaries,
- dependencies,
- tradeoffs,
- design decisions,
- reviews.

Typical Artifacts

Examples may include:

- architecture diagrams,
- ADRs,
- architecture reviews,
- system explanations.

Accountability Question

> Why did we build the system this way?

Phase Gate 4 — Cycle 1 Construction, Integration, And AI Code Review Package

Educational Goal

Teach disciplined implementation.

Students learn:

- pull requests,

- integration,
- testing,
- AI-assisted development,
- review participation.

Typical Artifacts

Examples may include:

- implementation evidence,
- AI disclosures,
- review records,
- validation evidence.

Accountability Question

> What evidence supports our engineering claims?

Phase Gate 5 — Cycle 1 Release Presentation And Peer Review Package

Educational Goal

Teach release evaluation.

Students learn:

- release readiness,
- evidence evaluation,
- peer accountability,
- risk communication.

Typical Artifacts

Examples may include:

- release summaries,
- peer reviews,
- risk assessments,
- recommendations.

Accountability Question

> Why should this release be trusted?

Phase Gate 6 — Final Cycle Release Presentation And Peer Review Package

Educational Goal

Teach full engineering accountability.

Students defend the entire engineering journey.

Students explain:

- what they intended,
- what they built,
- what AI assisted,
- what humans verified,

- what risks remain,
- how the system would operate,
- why the release is defensible.

Typical Artifacts

Examples may include:

- final repository,
- release defense,
- engineering narrative,
- lessons learned.

Accountability Question

> Would another engineer trust our work?

Evidence Growth Model

Evidence intentionally accumulates throughout the semester.

```
Intent Evidence
     ↓
Planning Evidence
     ↓
Architecture Evidence
     ↓
Review Evidence
     ↓
Validation Evidence
     ↓
Release Evidence
     ↓
Defense Evidence
```

Students continuously refine artifacts.

Nothing disappears.

AI Governance Within COMP330

AI is intentionally allowed.

Students learn:

- AI governance,
- AI disclosure,
- AI verification,
- bounded authority.

Students learn:

> AI proposes; engineers verify.

AI is treated as an engineering capability rather than an engineering replacement.

Repository Philosophy Within COMP330

Repositories are engineering systems.

Repositories preserve:

- engineering decisions,
- evidence,
- reviews,
- AI usage,
- risks,
- engineering history.

Students learn that repositories are not submission folders.

Why COMP330 Exists Inside ETIS

COMP330 serves multiple purposes.

It is:

- a flagship implementation,
- a proving ground,
- an educational laboratory,
- a reusable asset contributor.

COMP330 informs ETIS.

ETIS informs COMP330.

This relationship should remain visible.

Adaptation Guidance

Future educators should not copy COMP330 directly.

Instead, educators should ask:

- What educational philosophy should be preserved?
- What accountability progression should be preserved?
- What evidence systems should be preserved?
- What release experiences should be preserved?

Adaptations should preserve doctrine rather than duplicate implementation.

What Should Be Preserved

The following ideas should remain visible.

- repository-centered engineering,
- evidence-centered engineering,
- AI governance,

- architectural thinking,
- review systems,
- release readiness,
- operational awareness,
- cumulative accountability.

These ideas matter more than the exact phase gates.

Common Misconceptions

Misconception 1 — COMP330 Is ETIS

COMP330 is one implementation.

Misconception 2 — Future Courses Must Use Six Gates

Different environments may require different models.

Misconception 3 — These Are Assignments

These are cumulative accountability experiences.

Misconception 4 — Release Presentations Are Demonstrations

They are engineering defenses.

Misconception 5 — The Repository Is A Submission System

The repository is an engineering system.

Guiding Standard

A successful ETIS implementation should allow students to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How would this system operate?
- Why should this work be trusted?
- What would we improve next?

COMP330 intentionally builds toward these questions.

Core Commitment

The purpose of preserving COMP330 is not to standardize ETIS.

The purpose is to preserve engineering memory.

COMP330 demonstrates one successful approach for teaching software engineering in the AI era.

Future implementations may evolve.

The underlying ETIS doctrine should remain stable.

Educational work should progressively resemble professional engineering work.

That progression is one of the foundations of ETIS education.

ETIS Evidence Progression Guide

The **ETIS Evidence Progression Guide** provides recommendations for how engineering evidence should progressively mature throughout ETIS educational experiences.

This document does not define assignment deliverables.

It defines evidence systems.

Evidence is one of the foundational concepts of ETIS.

Educational experiences should teach students how to create, preserve, explain, review, and defend engineering evidence.

The objective is not simply to build systems.

The objective is to build systems that can be understood, reviewed, governed, operated, improved, and trusted over time.

Evidence makes that possible.

Purpose

The purpose of this document is to help instructors intentionally build engineering evidence throughout educational experiences.

This document helps instructors answer questions such as:

- What evidence should students create?
- When should evidence appear?
- How should evidence mature?
- How should evidence accumulate?
- How should repositories preserve evidence?
- How should students defend evidence?
- How should evidence support engineering claims?

Educational experiences should progressively strengthen engineering evidence.

Guiding Principle

ETIS evidence progression should answer one question:

> What evidence supports this engineering claim?

Students should repeatedly ask this question.

It should become a professional habit.

Every engineering claim should connect to evidence.

Educational Philosophy

Traditional courses often organize around outputs.

Examples include:

- assignments,
- projects,
- demonstrations,

- presentations.

ETIS organizes around evidence.

Examples include:

- intent evidence,
- architecture evidence,
- review evidence,
- validation evidence,
- release evidence,
- operational evidence.

Educational work should produce engineering memory.

Foundational Principle

Evidence maturity should mirror engineering maturity.

Students should progressively learn how to produce increasingly sophisticated evidence throughout the educational experience.

Educational evidence should evolve over time.

ETIS Evidence Architecture

Most ETIS educational experiences should progressively build eight evidence categories.

```
Intent Evidence
   ↓
Planning Evidence
   ↓
Architecture Evidence
   ↓
Review Evidence
   ↓
Validation Evidence
   ↓
Release Evidence
   ↓
Operational Evidence
   ↓
Defense Evidence
```

The repository should tell this story.

Evidence Category 1 — Intent Evidence

Educational Goal

Teach students to define problems before building solutions.

Intent evidence answers:

> What are we trying to accomplish?

Examples include:

- problem statements,
- stakeholder definitions,
- requirements,
- use cases,
- assumptions,
- constraints.

Students should understand:

> Intent precedes implementation.

Evidence Category 2 — Planning Evidence

Educational Goal

Teach students to organize engineering work.

Planning evidence answers:

> How will we accomplish this?

Examples include:

- work breakdown structures,
- RACI matrices,
- schedules,
- estimations,
- risks,
- priorities.

Students should understand:

> Planning reduces uncertainty.

Evidence Category 3 — Architecture Evidence

Educational Goal

Teach students to justify engineering decisions.

Architecture evidence answers:

> Why was this built this way?

Examples include:

- architecture diagrams,
- ADRs,
- dependency diagrams,
- tradeoff analyses,

- context boundaries.

Students should understand:

> Governance is architecture.

Evidence Category 4 — Review Evidence

Educational Goal

Teach students to normalize accountability.

Review evidence answers:

> Who evaluated this work?

Examples include:

- peer reviews,
- architecture reviews,
- pull request reviews,
- evidence reviews,
- release reviews.

Students should understand:

> Engineering is a reviewable discipline.

Evidence Category 5 — Validation Evidence

Educational Goal

Teach students to verify claims.

Validation evidence answers:

> How do we know this works?

Examples include:

- test results,
- verification summaries,
- integration records,
- quality assessments,
- AI verification records.

Students should understand:

> AI proposes; engineers verify.

Evidence Category 6 — Release Evidence

Educational Goal

Teach students to defend releases.

Release evidence answers:

> Why should this release be trusted?

Examples include:

- release readiness packages,
- risk summaries,
- quality summaries,
- known limitations,
- release recommendations.

Students should understand:

> A demo is not operational proof.

Evidence Category 7 — Operational Evidence

Educational Goal

Teach students that engineering continues after implementation.

Operational evidence answers:

> How will this system survive reality?

Examples include:

- postmortems,
- observability plans,
- operational readiness plans,
- reliability analyses,
- incident analyses.

Students should understand:

> The model is not the system.

Evidence Category 8 — Defense Evidence

Educational Goal

Teach students professional accountability.

Defense evidence answers:

> Why should another engineer trust this work?

Examples include:

- defense presentations,
- engineering narratives,
- evidence summaries,
- lessons learned,
- stewardship reflections.

Students should understand:

> Engineering accountability is part of engineering itself.

Evidence Growth Model

Evidence should accumulate throughout the educational experience.

> Intent

↓

Intent + Planning

↓

Intent + Planning + Architecture

↓

Intent + Planning + Architecture + Review

↓

Intent + Planning + Architecture + Review + Validation

↓

Intent + Planning + Architecture + Review + Validation + Release

↓

Intent + Planning + Architecture + Review + Validation + Release + Operations

↓

Intent + Planning + Architecture + Review + Validation + Release + Operations + Defense

Nothing should disappear.

Artifacts should mature.

Repository Expectations

Repositories should preserve evidence throughout the educational experience.

Repositories should answer questions such as:

- What problem were we solving?
- What decisions were made?
- What evidence exists?
- What risks remain?
- What was reviewed?
- What was verified?
- How would this system operate?

Repositories should become engineering memory systems.

AI Evidence Expectations

AI introduces new evidence responsibilities.

Students should preserve evidence regarding:

- AI use,
- AI prompts when appropriate,
- AI disclosures,
- AI verification,

- AI limitations.

Students should understand:

> AI involvement should be transparent.

Evidence Review Philosophy

Evidence should itself be reviewable.

Review questions include:

- Is this evidence complete?
- Is this evidence trustworthy?
- Is this evidence understandable?
- Is this evidence sufficient?
- Is this evidence actionable?

Students should learn to critique evidence quality.

Recommended Instructor Role Progression

Instructor responsibilities should evolve.

```
Early
Teacher
   ↓
Middle
Coach
   ↓
Later
Reviewer
   ↓
Final
Engineering Defense Board
```

Students should progressively assume ownership.

Common Evidence Mistakes

Mistake 1 — Treating Evidence As Documentation

Evidence supports engineering claims.

Mistake 2 — Waiting Too Long To Create Evidence

Evidence should begin immediately.

Mistake 3 — Creating Evidence Only At The End

Evidence should accumulate continuously.

Mistake 4 — Treating Repositories Like Submission Folders

Repositories are engineering systems.

Mistake 5 — Ignoring AI Evidence

AI accountability requires evidence.

Mistake 6 — Prioritizing Quantity Over Quality

Strong evidence matters more than abundant evidence.

Mistake 7 — Treating Presentations As Evidence

Presentations summarize evidence.

They are not evidence themselves.

Adaptation Rules

Educational environments may adapt implementation details.

However, they should preserve:

- intent evidence,
- architecture evidence,
- review evidence,
- validation evidence,
- release evidence,
- operational evidence,
- accountability evidence.

These remain constitutional ETIS concepts.

Instructor Design Questions

Instructors should periodically ask:

- What evidence should students create?
- What claims should that evidence support?
- Who will review that evidence?
- How will evidence mature?
- How will evidence be defended?

The answers should drive educational design.

Guiding Standard

A successful ETIS educational experience should allow students to answer:

- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- Why should this release be trusted?
- How will this system operate?
- Why should another engineer trust our work?

If students cannot answer these questions, the evidence system should be redesigned.

Core Commitment

The purpose of ETIS evidence progression is not to create more documentation.

The purpose is to create engineering memory.

Students should progressively learn how to create evidence that allows systems to be understood, reviewed, governed, operated, improved, and trusted over time.

That progression is one of the foundations of ETIS education.

ETIS Phase Gate Models Library

The **ETIS Phase Gate Models Library** provides reusable accountability architectures for ETIS educational implementations.

This document does not provide assignment collections.

It provides scalable models for sequencing professional engineering accountability.

Phase gates are one of the primary educational mechanisms ETIS uses to transform learners into trustworthy engineers.

The number of phase gates may vary.

The educational philosophy should not.

The objective is not to complete assignments.

The objective is to progressively assume engineering ownership.

Purpose

The purpose of this document is to help educators select a phase gate architecture appropriate for their educational environment.

This document helps instructors answer questions such as:

- How many phase gates should my course have?
- How should engineering accountability increase over time?
- How should educational environments scale?
- How should evidence accumulate?
- How should phase gates adapt to compressed environments?
- How should professional training differ from semesters?
- How should educational experiences remain recognizable as ETIS?

Phase gate models provide reusable accountability architectures.

Guiding Principle

ETIS phase gates should answer one question:

> What engineering responsibility should learners now be able to own?

Every phase gate should increase accountability.

Students should progressively evolve through stages.

Student

↓

Responsible Engineer

↓

Reviewer

↓

Architect

↓

Release Defender

↓

Operational Thinker

↓

Future Trustworthy Engineer

Phase gates should support this progression.

Educational Philosophy

Traditional courses often organize educational work like this:

Assignment

↓

Assignment

↓

Assignment

↓

Project

ETIS organizes educational work differently.

Foundation

↓

Intent

↓

Architecture

↓

Construction

↓

Release

↓

Operations

↓

Defense

Educational work should resemble professional engineering work.

Selecting A Phase Gate Model

Choose the model based upon educational goals rather than calendar length.

Consider:

- learner maturity,
- available instructional time,
- project complexity,
- desired accountability depth,
- educational environment.

Phase gate count should support educational outcomes.

More phase gates do not automatically create better learning.

Model 1 — Three Phase Gate Architecture

Intended Environment

Examples:

- professional training,
- workshops,
- executive education,
- short modules.

Educational Goal

Introduce ETIS without overwhelming learners.

Architecture

Phase Gate 1

Intent And Foundation

↓

Phase Gate 2

Construction And Validation

↓

Phase Gate 3

Operational Accountability And Defense

Primary Student Experience

Learners experience ETIS as an accountability framework.

Recommended Evidence

Examples:

- engineering intent,
- architecture decisions,
- validation evidence,
- operational reflections,
- defense summaries.

Model 2 — Four Phase Gate Architecture

Intended Environment

Examples:

- accelerated courses,
- six-week modules,
- focused ETIS experiences.

Educational Goal

Teach a complete ETIS experience in compressed environments.

Architecture

Phase Gate 1

Repository Foundation

↓

Phase Gate 2

Intent And Architecture

↓

Phase Gate 3

Construction And Release

↓

Phase Gate 4

Operations And Defense

Primary Student Experience

Students experience the full engineering lifecycle at a compressed scale.

Recommended Evidence

Examples:

- repository evidence,
- planning evidence,
- architecture evidence,
- review evidence,
- release evidence,
- operational evidence.

Model 3 — Five Phase Gate Architecture

Intended Environment

Examples:

- quarter systems,
- capstones,
- advanced undergraduate environments.

Educational Goal

Balance educational depth with compressed schedules.

Architecture

Phase Gate 1

Repository Foundation

↓

Phase Gate 2

Intent And Planning

↓

Phase Gate 3

Architecture And Construction

↓

Phase Gate 4

Release And Operations

↓

Phase Gate 5

Final Defense

Primary Student Experience

Students experience increasing accountability while reducing educational overhead.

Recommended Evidence

Examples:

- repository evidence,
- planning evidence,
- architecture evidence,
- validation evidence,
- release evidence,
- operational evidence,
- defense evidence.

Model 4 — Six Phase Gate Architecture

Intended Environment

Examples:

- undergraduate software engineering,
- flagship implementations,
- semester-long ETIS courses.

Educational Goal

Deliver the full ETIS educational experience.

Architecture

Phase Gate 1

Repository Foundation

↓

Phase Gate 2

Intent And Planning

↓

Phase Gate 3

Architecture And Review

↓

Phase Gate 4

Construction And Validation

↓

Phase Gate 5

Release Readiness And Operations

↓

Phase Gate 6

Final Engineering Defense

Primary Student Experience

Students progressively mature into trustworthy engineers.

This is the flagship ETIS educational model.

Recommended Evidence Growth Model

Evidence should accumulate throughout every model.

Intent Evidence

↓

Planning Evidence

↓

Architecture Evidence

↓

Review Evidence

↓

Validation Evidence

↓

Release Evidence

↓

Operational Evidence

↓

Defense Evidence

The repository should tell a complete engineering story.

Recommended Accountability Growth Model

Students should progressively assume ownership.

Follow Instructions

↓

Make Decisions

↓

Explain Decisions

↓

Defend Decisions

↓

Own Outcomes

Educational experiences should make this progression visible.

Instructor Role Progression

Instructor responsibilities should evolve.

Early

Teacher

↓

Middle

Coach

↓

Later

Reviewer

↓

Final

Engineering Defense Board

Students should progressively become independent.

Model Selection Guidance

Choose Three Gates When

- time is extremely limited,
- introducing ETIS concepts,
- working with executives,
- running workshops.

Choose Four Gates When

- teaching compressed courses,
- building modules,
- introducing ETIS within existing programs.

Choose Five Gates When

- operating within quarter systems,
- teaching capstones,
- balancing depth and time.

Choose Six Gates When

- ETIS is the primary course framework,
- implementing flagship courses,
- maximizing professional formation.

Common Mistakes

Mistake 1 — Creating Too Many Gates

More gates do not equal more learning.

Mistake 2 — Making Gates Independent

Phase gates should inherit previous work.

Mistake 3 — Delaying AI Governance

AI accountability should begin immediately.

Mistake 4 — Removing Operations

Students should understand life after implementation.

Mistake 5 — Treating Gates Like Assignments

Phase gates are accountability systems.

Mistake 6 — Eliminating Final Defense

Students should defend engineering outcomes.

Mistake 7 — Prioritizing Calendar Efficiency Over Professional Formation

Educational transformation remains primary.

Adaptation Rules

Implementations may adapt gate structures.

However, they should preserve:

- engineering intent,
- evidence,
- review,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship,
- final accountability.

These remain constitutional ETIS capabilities.

Guiding Standard

A successful ETIS phase gate model should allow learners to answer:

- What problem are we solving?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How would this system operate?
- Why should this work be trusted?
- What would we improve next?

If learners cannot answer these questions, the model should be reconsidered.

Core Commitment

The purpose of ETIS phase gate models is not to create assignment structures.

The purpose is to create accountability architectures.

Educational work should progressively resemble professional engineering work.

Students should learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

That progression is one of the foundations of ETIS education.

Part V

Assessment Guidance

ETIS AI Assessment Guidance

The **ETIS AI Assessment Guidance** document provides recommendations for evaluating AI-assisted engineering throughout ETIS educational experiences.

This document intentionally moves beyond AI detection approaches.

ETIS does not teach students to avoid AI.

ETIS teaches students to govern AI.

The objective is not to determine whether AI was used.

The objective is to determine whether AI was used responsibly.

Students should progressively learn how to incorporate AI into engineering environments while preserving trust, accountability, and professional ownership.

AI usage is not an academic violation.

Undisclosed and unverified AI dependency is an engineering risk.

Purpose

The purpose of this document is to help instructors intentionally evaluate AI-assisted engineering.

This document helps instructors answer questions such as:

- How should AI usage be evaluated?
- What should students disclose?
- What should students verify?
- How should AI accountability be assessed?
- How should AI risks be assessed?
- How should AI be incorporated into engineering work?
- How should AI reinforce trustworthy engineering?

Educational evaluation should reinforce responsible AI use rather than AI avoidance.

Guiding Principle

ETIS AI assessment should answer one question:

> How did the engineer responsibly govern AI?

Students should repeatedly answer this question.

It should become a professional habit.

Educational Philosophy

Many educational environments currently approach AI this way:

```
AI Used
  ↓
Detect AI
  ↓
Determine Penalty
```

ETIS intentionally uses a different model.

AI Used

↓

Disclose AI

↓

Verify AI

↓

Explain AI

↓

Bound AI Authority

↓

Own Outcomes

Educational evaluation should reinforce this progression.

Foundational Principle

Do not assess AI avoidance.

Assess AI responsibility.

Students should learn how to responsibly integrate AI into engineering environments.

That skill will increasingly define future engineers.

ETIS AI Philosophy

ETIS preserves one foundational doctrine.

> AI proposes; engineers verify.

Students should understand:

- AI is a capability.
- AI is governed.
- AI has limitations.
- AI requires verification.
- Humans own outcomes.

These concepts should remain visible throughout educational experiences.

AI Assessment Architecture

Most ETIS educational environments should evaluate AI across seven dimensions.

AI Transparency

↓

AI Intent

↓

AI Verification

↓

AI Limitations

↓

AI Risk Awareness

↓

AI Integration

↓

Human Ownership

Students should progressively mature across all dimensions.

Dimension 1 — AI Transparency

Educational Goal

Evaluate whether AI usage is visible.

Questions include:

- Was AI usage disclosed?
- Was AI involvement explained?
- Is AI usage understandable?

Students should understand:

> Hidden AI usage creates engineering risk.

Examples include:

- AI use logs,
- AI disclosure sections,
- AI acknowledgements.

Dimension 2 — AI Intent

Educational Goal

Evaluate why AI was used.

Questions include:

- Why was AI used?
- Was AI appropriate for the task?
- Was AI intentionally applied?

Students should understand:

> AI usage should be purposeful.

Examples include:

- brainstorming,

- summarization,
- code generation,
- architectural exploration,
- documentation assistance.

Students should justify usage decisions.

Dimension 3 — AI Verification

Educational Goal

Evaluate whether AI outputs were verified.

Questions include:

- What was verified?
- How was verification performed?
- Were errors identified?
- Were assumptions challenged?

Students should understand:

> AI outputs are inputs to engineering, not engineering itself.

Verification should become a normal engineering behavior.

Dimension 4 — AI Limitations

Educational Goal

Evaluate awareness of AI limitations.

Questions include:

- Were limitations identified?
- Were uncertainties acknowledged?
- Were hallucination risks considered?

Students should understand:

> AI confidence is not evidence.

Students should learn healthy skepticism.

Dimension 5 — AI Risk Awareness

Educational Goal

Evaluate AI governance thinking.

Questions include:

- What risks were introduced?
- What risks were mitigated?
- What risks remain?

Examples include:

- factual errors,
- hidden assumptions,

- security risks,
- operational risks,
- overdependence risks.

Students should understand:

> AI changes engineering risks.

Dimension 6 — AI Integration

Educational Goal

Evaluate how AI fits into engineering workflows.

Questions include:

- Was AI integrated appropriately?
- Did AI support engineering work?
- Did AI replace engineering thinking?

Students should understand:

> AI should augment engineering.

It should not replace engineering.

Dimension 7 — Human Ownership

Educational Goal

Evaluate engineering accountability.

Questions include:

- Who owns the outcome?
- Who verified the work?
- Who accepts responsibility?

Students should understand:

> Humans own outcomes.

This should remain non-negotiable.

Recommended AI Evidence

Students should preserve evidence regarding AI usage.

Examples include:

- AI use logs,
- AI disclosures,
- verification notes,
- limitations identified,
- risk analyses,
- engineering reflections.

AI involvement should become transparent.

Recommended AI Disclosure Questions

Students should periodically answer:

- How did AI assist?
- Why was AI used?
- What was verified?
- What limitations were discovered?
- What risks remain?
- What decisions did humans make?

These questions should become normal engineering behaviors.

AI Assessment Maturity Model

Students should progressively evolve.

AI Awareness

↓

AI Transparency

↓

AI Verification

↓

AI Governance

↓

AI Accountability

↓

Human Ownership

Educational environments should reinforce this progression.

AI And Repository Expectations

Repositories should preserve AI usage.

Repositories should answer questions such as:

- Where was AI used?
- Why was AI used?
- What was verified?
- What risks remain?
- Who owns the outcomes?

Students should understand:

> Repositories are engineering memory systems.

AI And Engineering Defenses

Students should explain AI usage during defenses.

Questions include:

- Where did AI assist?
- What was verified?
- What risks remain?
- What limitations were discovered?
- What decisions remained human decisions?

AI should become visible rather than hidden.

Recommended Instructor Role Progression

Instructor responsibilities should evolve.

Early

Teacher

↓

Middle

Coach

↓

Later

Reviewer

↓

Final

Engineering Defense Board

Students should progressively assume ownership.

Common AI Assessment Mistakes

Mistake 1 — Focusing On AI Detection

Detection is not the educational objective.

Mistake 2 — Penalizing AI Usage

AI is an engineering capability.

Mistake 3 — Ignoring AI Governance

Governance should be explicit.

Mistake 4 — Ignoring Verification

Verification is essential.

Mistake 5 — Rewarding AI Output Alone

AI output is not engineering.

Mistake 6 — Ignoring Human Ownership

Humans own outcomes.

Mistake 7 — Treating AI As Exceptional

AI should become a governed engineering capability.

Adaptation Rules

Educational environments may adapt AI implementation details.

However, they should preserve evaluation of:

- AI transparency,
- AI verification,
- AI limitations,
- AI governance,
- AI accountability,
- human ownership.

These remain constitutional ETIS capabilities.

Instructor Evaluation Questions

Before evaluating AI-assisted work, ask:

- Why was AI used?
- Was AI disclosed?
- What was verified?
- What limitations were discovered?
- What risks remain?
- Who owns the outcome?

These questions should drive evaluation.

Guiding Standard

A successful ETIS educational experience should allow learners to answer:

- How did AI assist?
- Why was AI used?
- What did humans verify?
- What limitations were discovered?
- What risks remain?
- What decisions remained human decisions?
- Why should another engineer trust this work?

If learners cannot answer these questions, AI governance should be strengthened.

Core Commitment

The purpose of ETIS AI assessment is not to regulate AI usage.

The purpose is to create trustworthy engineers who can responsibly govern AI.

Students should progressively learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes while using AI as a professional capability.

The future will not belong to engineers who avoid AI.

It will belong to engineers who can responsibly govern AI while preserving trust.

ETIS Assessment Assembly Guide

The **ETIS Assessment Assembly Guide** provides a repeatable process for designing ETIS engineering evaluation systems.

This guide does not teach instructors how to create grading schemes.

It teaches instructors how to engineer evaluation systems that reinforce trustworthy engineering.

Assessment should intentionally shape professional behaviors.

Educational evaluation should resemble professional engineering evaluation.

The objective is not to assign grades.

The objective is to develop trustworthy engineers.

Purpose

The purpose of this document is to help instructors intentionally build ETIS assessment systems.

This guide helps instructors answer questions such as:

- What engineering behaviors should be evaluated?
- What evidence should support evaluation?
- How should engineering maturity be measured?
- How should AI accountability be evaluated?
- How should reviews be evaluated?
- How should professional ownership emerge?
- How should assessment evolve throughout the educational experience?

Assessment systems should be engineered rather than assembled ad hoc.

Guiding Principle

Every ETIS assessment system should answer one question:

> What engineering responsibility should learners now be able to own?

That question should drive every assessment decision.

Assessment should reinforce engineering accountability.

Assessment should not simply measure assignment completion.

Educational Philosophy

Traditional assessment design often looks like this:

Assignment

↓

Submission

↓

Points

↓

Grade

ETIS assessment design should look like this:

Engineering Responsibility

↓

Engineering Behaviors

↓

Evidence

↓

Reviews

↓

Accountability

↓

Professional Ownership

Educational evaluation should progressively resemble professional engineering evaluation.

ETIS Assessment Assembly Pipeline

Assessment systems should be assembled in the following order.

Professional Transformation Goal

↓

Engineering Responsibility

↓

Observable Behaviors

↓

Evidence Requirements

↓

Review Mechanisms

↓

AI Expectations

↓

Accountability Questions

↓

Evaluation Criteria

↓

Assessment System

This order should remain stable.

Step 1 — Define The Professional Transformation Goal

Begin by identifying who learners should become.

Examples include:

Student
↓
Responsible Engineer

Student
↓
Reviewer

Student
↓
Architect

Student
↓
Release Defender

Student
↓
Operational Thinker

The transformation goal should remain visible.

Step 2 — Define The Engineering Responsibility

Identify what learners should now own.

Examples include:

- defining engineering intent,
- making engineering decisions,
- reviewing engineering work,
- evaluating risks,
- defending releases,
- operating systems,
- communicating tradeoffs.

Responsibilities should drive evaluation.

Step 3 — Define Observable Behaviors

Assess behaviors rather than outputs alone.

Examples include:

- communicating clearly,
- participating in reviews,
- justifying decisions,
- identifying risks,
- validating claims,
- documenting evidence,
- collaborating professionally.

Observable behaviors should be explicit.

Step 4 — Define Evidence Requirements

Assessment should connect directly to evidence.

Examples include:

- requirements,
- architecture decisions,
- review records,
- AI disclosures,
- validation evidence,
- release evidence,
- operational analyses.

Evidence should support evaluation.

Step 5 — Design Review Mechanisms

Students should experience accountability systems.

Examples include:

- peer reviews,
- architecture reviews,
- repository reviews,
- evidence reviews,
- release reviews.

Reviews should become assessable activities.

Step 6 — Define AI Expectations

Every assessment system should explicitly address AI.

Students should understand:

- AI is allowed.
- AI is governed.
- AI should be disclosed.
- AI should be verified.
- Humans own outcomes.

AI governance should be visible.

Step 7 — Define Accountability Questions

Every assessment system should ask accountability questions.

Examples include:

- What problem are we solving?
- Why did we choose this approach?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- Why should this work be trusted?

Students should repeatedly answer these questions.

Step 8 — Build Evaluation Criteria

Evaluation criteria should be consistent throughout ETIS.

Common dimensions include:

Engineering Intent

Can students define the problem?

Evidence Quality

Can students support claims?

Engineering Reasoning

Can students explain decisions?

Review Participation

Can students participate professionally?

AI Accountability

Can students responsibly use AI?

Release Readiness

Can students defend releases?

Operational Thinking

Can students think beyond implementation?

Professional Ownership

Can students own outcomes?

Step 9 — Assemble The Assessment System

Combine all components into a coherent engineering evaluation system.

Assessment should become cumulative rather than fragmented.

Students should continuously mature throughout the educational experience.

Recommended Assessment Blueprint

Every ETIS assessment system should contain these sections.

Purpose

↓

Engineering Responsibility

↓

Observable Behaviors

↓

Evidence Expectations

↓

Review Expectations

↓

AI Expectations

↓

Accountability Questions

↓

Evaluation Criteria

↓

Professional Ownership Expectations

This blueprint should remain relatively stable.

Assessment Should Mature Over Time

Assessment emphasis should evolve.

Participation

↓

Contribution

↓

Decision Making

↓

Accountability

↓

Professional Ownership

Students should feel this progression.

Repository Expectations

Repositories should become assessable engineering systems.

Repositories should preserve:

- decisions,
- evidence,
- reviews,
- risks,
- AI usage,
- engineering history.

Repositories should support evaluation.

Students should understand:

> Repositories are engineering systems.

Instructor Role Progression

Instructor roles should evolve.

Early
Teacher
↓
Middle
Coach
↓
Later
Reviewer
↓
Final
Engineering Defense Board

Students should progressively assume ownership.

Common Assessment Design Mistakes

Mistake 1 — Starting With Points

Start with engineering responsibilities.

Mistake 2 — Evaluating Technical Output Alone

Technical output is only one dimension.

Mistake 3 — Ignoring Evidence

Evidence should support engineering claims.

Mistake 4 — Delaying AI Governance

AI expectations should be visible immediately.

Mistake 5 — Ignoring Reviews

Reviews are engineering mechanisms.

Mistake 6 — Assessing Everything Equally

Assessment should mature over time.

Mistake 7 — Waiting Until The End To Assess Ownership

Ownership should progressively increase.

Adaptation Rules

Educational environments may adapt implementation details.

However, they should preserve evaluation of:

- engineering intent,
- evidence,
- reviews,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship,
- professional ownership.

These remain constitutional ETIS capabilities.

Instructor Design Questions

Before finalizing an ETIS assessment system, ask:

- What engineering behavior am I evaluating?
- What evidence supports evaluation?
- What decisions should students explain?
- What risks should students communicate?
- What ownership should students demonstrate?

These answers should drive design decisions.

Guiding Standard

A successful ETIS assessment system should allow learners to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?

- What would we improve next?

If learners cannot answer these questions, the assessment system should be redesigned.

Core Commitment

The purpose of ETIS assessment assembly is not to create stronger grading systems.

The purpose is to create stronger engineers.

Assess the engineer students are becoming, not only the artifacts they produce.

Educational evaluation should progressively help students learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

That progression is one of the foundations of ETIS education.

ETIS Assessment Models Library

The **ETIS Assessment Models Library** provides reusable architectures for evaluating engineering maturity throughout ETIS educational experiences.

This document does not provide grading systems.

It provides engineering evaluation systems.

ETIS intentionally moves beyond evaluating technical output alone.

Students should progressively be evaluated on increasingly sophisticated engineering responsibilities.

The objective is not to determine whether students can build software.

The objective is to determine whether students can engineer trustworthy systems.

Assessment models provide reusable evaluation architectures that may be adapted to many educational environments.

Purpose

The purpose of this document is to help instructors select assessment models that align with desired educational outcomes.

This document helps instructors answer questions such as:

- What should be evaluated early?
- What should be evaluated later?
- How should evaluation mature?
- How should engineering accountability be evaluated?
- How should AI governance be evaluated?
- How should professional ownership emerge?
- How should ETIS remain recognizable across environments?

Assessment models should reinforce professional transformation.

Guiding Principle

ETIS assessment should answer one question:

> What engineering responsibility should learners now be able to own?

Assessment should progressively evaluate increasing ownership.

Students should gradually move beyond completion toward accountability.

Educational Philosophy

Traditional educational assessment often looks like this:

Homework

↓

Project

↓

Presentation

↓

Exam

ETIS intentionally expands evaluation.

Intent

↓

Evidence

↓

Reasoning

↓

Reviews

↓

AI Accountability

↓

Release Readiness

↓

Operations

↓

Professional Ownership

Educational evaluation should progressively resemble professional engineering evaluation.

Foundational Principle

Evaluate behaviors before evaluating artifacts.

Artifacts matter.

Behaviors matter more.

Strong engineering behaviors eventually create strong engineering artifacts.

The reverse is not always true.

Selecting An Assessment Model

Assessment models should be selected based upon educational goals.

Consider:

- learner maturity,
- available time,
- educational depth,
- project complexity,
- desired accountability.

Assessment complexity should align with educational goals.

More assessment categories do not automatically improve learning.

Model 1 — Foundational Assessment Model

Educational Goal

Establish engineering expectations.

Primary Focus

Participation and contribution.

Evaluation Categories

Participation

↓

Engagement

↓

Repository Habits

↓

Engineering Communication

Example Behaviors

Students:

- participate professionally,
- engage in discussions,
- establish repositories,
- understand expectations.

Recommended Environments

Examples:

- introductory ETIS environments,
- short modules,
- workshops.

Model 2 — Developmental Assessment Model

Educational Goal

Strengthen engineering reasoning.

Primary Focus

Contribution and decision making.

Evaluation Categories

Intent

↓

Planning

↓

Architecture

↓

Engineering Decisions

Example Behaviors

Students:

- define problems,
- analyze tradeoffs,
- make decisions,
- justify decisions.

Recommended Environments

Examples:

- undergraduate courses,
- accelerated environments,
- architecture modules.

Model 3 — Accountability Assessment Model

Educational Goal

Evaluate engineering accountability.

Primary Focus

Evidence and reviews.

Evaluation Categories

Evidence

↓

Reviews

↓

Validation

↓

Risk Communication

Example Behaviors

Students:

- support claims,
- participate in reviews,
- communicate risks,
- verify engineering work.

Recommended Environments

Examples:

- software engineering courses,
- capstones,
- advanced modules.

Model 4 — Operational Assessment Model

Educational Goal

Evaluate life beyond implementation.

Primary Focus

Release readiness and operational thinking.

Evaluation Categories

Release Readiness
↓
Operations
↓
Risk Awareness
↓
Continuous Improvement

Example Behaviors

Students:

- defend releases,
- identify risks,
- analyze failures,
- recommend improvements.

Recommended Environments

Examples:

- capstones,
- graduate courses,
- professional training.

Model 5 — Professional Ownership Model

Educational Goal

Evaluate full engineering ownership.

Primary Focus

End-to-end accountability.

Evaluation Categories

Intent

↓

Evidence

↓

Reviews

↓

AI Accountability

↓

Release Readiness

↓

Operations

↓

Defense

↓

Ownership

Example Behaviors

Students:

- explain decisions,
- defend claims,
- communicate risks,
- steward systems.

Recommended Environments

Examples:

- flagship ETIS courses,
- graduate environments,
- advanced professional programs.

Assessment Growth Model

Students should progressively evolve.

Participation

↓

Contribution

↓

Decision Making

↓

Accountability

↓

Professional Ownership

Assessment should reinforce this progression.

Recommended Evaluation Dimensions

ETIS educational environments should progressively evaluate these dimensions.

Engineering Intent

Can students define the problem?

Evidence Quality

Can students support engineering claims?

Engineering Reasoning

Can students explain decisions?

Review Participation

Can students participate in accountability systems?

AI Accountability

Can students responsibly govern AI use?

Release Readiness

Can students defend releases?

Operational Thinking

Can students think beyond implementation?

Professional Ownership

Can students own outcomes?

AI Evaluation Expectations

AI should become a visible evaluation category.

Students should be evaluated on:

- AI disclosures,
- AI verification,
- AI limitations,
- AI governance decisions.

Students should understand:

> AI proposes; engineers verify.

Repository Evaluation Expectations

Repositories should become assessable engineering systems.

Repositories should answer:

- What decisions were made?
- What evidence exists?
- What reviews occurred?
- What risks remain?
- Is AI use transparent?

Repositories should become part of assessment.

Instructor Role Progression

Instructor responsibilities should evolve.

Early

Teacher

↓

Middle

Coach

↓

Later

Reviewer

↓

Final

Engineering Defense Board

Students should progressively assume ownership.

Common Model Selection Mistakes

Mistake 1 — Evaluating Technical Output Alone

Technical output is only one dimension.

Mistake 2 — Assessing Everything Immediately

Assessment should mature over time.

Mistake 3 — Delaying AI Accountability

AI governance should appear early.

Mistake 4 — Ignoring Operational Thinking

Students should understand life after implementation.

Mistake 5 — Ignoring Professional Ownership

Ownership should emerge throughout the experience.

Mistake 6 — Treating Presentations As Assessment

Presentations summarize engineering work.

They do not replace evaluation systems.

Mistake 7 — Overcomplicating Assessment

Assessment systems should remain understandable.

Adaptation Rules

Educational environments may adapt assessment models.

However, they should preserve evaluation of:

- engineering intent,
- evidence,
- reviews,
- AI accountability,
- release readiness,
- operational thinking,
- professional ownership.

These remain constitutional ETIS capabilities.

Instructor Selection Questions

Before selecting an assessment model, ask:

- What engineering behavior am I evaluating?
- What ownership should students demonstrate?
- What evidence should students create?
- What decisions should students explain?
- What risks should students communicate?

These answers should drive model selection.

Guiding Standard

A successful ETIS assessment model should allow learners to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?
- What would we improve next?

If learners cannot answer these questions, the assessment model should be redesigned.

Core Commitment

The purpose of ETIS assessment models is not to create stronger grading systems.

The purpose is to create stronger engineers.

Educational evaluation should progressively help students learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

That progression is one of the foundations of ETIS education.

ETIS Assessment Philosophy

The **ETIS Assessment Philosophy** document defines how ETIS evaluates engineering growth throughout educational experiences.

ETIS intentionally moves beyond traditional grading models.

The objective is not simply to determine whether students can build software.

The objective is to determine whether students can engineer systems that can be understood, reviewed, governed, operated, improved, and trusted over time.

Students should progressively learn how to assume engineering responsibility.

Assessment should reinforce that progression.

Engineering accountability is the educational outcome, not the side effect.

Purpose

The purpose of this document is to establish a shared assessment philosophy throughout ETIS educational environments.

This document helps instructors answer questions such as:

- Why does ETIS assess differently?
- What should be assessed?
- How should assessment evolve?
- Why is engineering accountability important?
- Why is evidence important?
- Why should students defend engineering decisions?
- How should AI governance be evaluated?

This document establishes the educational doctrine that supports ETIS assessment systems.

Guiding Principle

ETIS assessment should answer one question:

> What engineering responsibility should learners now be able to own?

Students should progressively assume greater ownership throughout educational experiences.

Assessment should reinforce that progression.

Educational Philosophy

Traditional software engineering education often evaluates technical outputs.

Examples include:

Homework

↓

Project

↓

Presentation

↓

Exam

ETIS intentionally expands assessment.

Students should also be evaluated on:

Intent

↓

Evidence

↓

Architecture

↓

Reviews

↓

AI Accountability

↓

Release Readiness

↓

Operational Thinking

↓

Professional Ownership

Educational assessment should progressively resemble professional engineering evaluation.

Foundational Principle

Assessment maturity should mirror engineering maturity.

Students should gradually evolve from following instructions to owning engineering outcomes.

Assessment should evolve accordingly.

ETIS Assesses Professional Formation

ETIS educational experiences are intentionally transformative.

Students should progressively evolve.

Student

↓

Responsible Engineer

↓

Reviewer

↓

Architect

↓

Release Defender

↓

Operational Thinker

↓

Future Trustworthy Engineer

Assessment should intentionally support this progression.

ETIS Assesses More Than Technical Output

Technical output remains important.

However, technical output alone is insufficient.

Students should also be assessed on:

- engineering reasoning,
- evidence quality,
- review participation,
- AI governance,
- risk awareness,
- release decisions,
- operational thinking,
- stewardship,
- ownership.

Students should understand:

> Building software is only one part of engineering.

ETIS Assesses Engineering Intent

Students should learn to define problems before implementing solutions.

Questions include:

- What problem are we solving?
- Who are the stakeholders?
- What assumptions exist?
- What constraints exist?

Intent should be visible.

ETIS Assesses Evidence

Students should support engineering claims with evidence.

Questions include:

- What evidence exists?
- Is the evidence trustworthy?

- Is the evidence understandable?
- Is the evidence sufficient?

Students should understand:

> Claims require evidence.

ETIS Assesses Engineering Decisions

Students should explain why decisions were made.

Questions include:

- Why was this architecture selected?
- What alternatives were considered?
- What tradeoffs exist?
- What risks remain?

Engineering reasoning should be visible.

ETIS Assesses Reviews

Students should participate in accountability systems.

Examples include:

- peer reviews,
- architecture reviews,
- repository reviews,
- evidence reviews,
- release reviews.

Students should understand:

> Engineering is a reviewable discipline.

ETIS Assesses AI Accountability

AI fundamentally changes educational expectations.

Students should be assessed on:

- AI disclosures,
- AI verification,
- AI governance,
- AI limitations.

Students should understand:

> AI proposes; engineers verify.

AI usage should be transparent.

ETIS Assesses Release Readiness

Students should experience release accountability.

Questions include:

- Why should this release be trusted?
- What evidence supports the release?
- What risks remain?

Students should understand:

> A demo is not operational proof.

ETIS Assesses Operational Thinking

Students should think beyond implementation.

Questions include:

- How would this system survive reality?
- How would it be monitored?
- How would incidents be handled?
- What improvements would be made?

Students should understand:

> The model is not the system.

ETIS Assesses Professional Ownership

Students should eventually own engineering outcomes.

Questions include:

- What decisions did we make?
- What evidence exists?
- What risks remain?
- What would we improve?

Ownership should become visible.

Assessment Should Evolve Over Time

Assessment emphasis should gradually change.

```
Early Course
Participation
   ↓
Middle Course
Decision Making
   ↓
Late Course
Accountability
   ↓
Final Stage
Professional Ownership
```

Students should feel this progression.

Educational Work Should Become Defensible

Students should repeatedly explain engineering decisions.

Questions include:

- Why did we build this?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?

Students should become comfortable defending engineering work.

Repositories Should Become Assessable Systems

Repositories should become part of assessment.

Repositories should answer questions such as:

- What decisions were made?
- What evidence exists?
- What reviews occurred?
- What risks remain?
- Is AI usage visible?

Students should understand:

> Repositories are engineering systems.

Instructor Roles Should Evolve

Instructor responsibilities should progressively change.

Early

Teacher

↓

Middle

Coach

↓

Later

Reviewer

↓

Final

Engineering Defense Board

Students should progressively assume ownership.

Common Assessment Misconceptions

Misconception 1 — Working Software Is Success

Working software is one dimension of success.

Misconception 2 — Assessment Is Grading

Assessment is engineering evaluation.

Misconception 3 — AI Makes Assessment Harder

AI changes what should be assessed.

Misconception 4 — Presentations Are Sufficient

Presentations summarize engineering work.

They do not replace evidence.

Misconception 5 — Reviews Are Optional

Reviews are engineering mechanisms.

Misconception 6 — Operations Belong Outside Education

Operations should be visible.

Misconception 7 — Ownership Only Matters At The End

Ownership should progressively increase.

Adaptation Rules

Educational environments may adapt assessment implementations.

However, they should preserve evaluation of:

- engineering intent,
- evidence,
- reviews,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship,
- professional ownership.

These remain constitutional ETIS capabilities.

Instructor Assessment Questions

Instructors should periodically ask:

- What engineering behavior am I evaluating?
- What evidence supports evaluation?
- What decisions should students explain?
- What risks should students communicate?
- What ownership should students demonstrate?

These answers should drive assessment design.

Guiding Standard

A successful ETIS educational experience should allow students to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?
- What would we improve next?

Assessment should reinforce these questions throughout the educational experience.

Core Commitment

The purpose of ETIS assessment philosophy is not to create stronger grading systems.

The purpose is to create stronger engineers.

Educational assessment should progressively help students learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

That progression is one of the foundations of ETIS education.

ETIS Assessment Scaling Guide

The **ETIS Assessment Scaling Guide** provides recommendations for scaling ETIS engineering evaluation systems across different educational environments while preserving educational integrity.

This document intentionally moves beyond scaling grading systems.

ETIS assessment focuses on engineering accountability rather than assignment completion.

Educational environments vary considerably.

Enrollment sizes differ.

Calendars differ.

Learner populations differ.

Delivery methods differ.

ETIS should adapt to these differences without sacrificing trustworthy engineering.

The objective is to scale evaluation mechanisms rather than reduce engineering expectations.

Purpose

The purpose of this document is to help instructors scale ETIS assessment systems responsibly.

This document helps instructors answer questions such as:

- How should ETIS assessment scale to different class sizes?
- How should ETIS assessment scale to online environments?
- How should ETIS assessment scale to professionals?
- What should change?
- What should remain constant?
- How should engineering accountability remain visible?

Assessment should preserve educational integrity regardless of scale.

Guiding Principle

ETIS assessment scaling should answer one question:

> What evaluation mechanisms can change without reducing engineering accountability?

Educational environments may change.

Engineering accountability should not.

Educational Philosophy

Traditional scaling often looks like this:

Bigger Environment

↓

Less Feedback

↓

Less Accountability

↓

Less Ownership

ETIS intentionally uses a different model.

Bigger Environment

↓

Different Evaluation Systems

↓

Preserved Accountability

↓

Preserved Ownership

Educational quality should remain stable.

Foundational Principle

Scale evaluation mechanisms, not engineering expectations.

Students should still learn how to:

- define intent,
- engineer context,
- bound authority,
- verify behavior,
- operate reality,
- explain decisions,
- own outcomes.

These expectations should remain visible.

What Should Never Change

The following ETIS evaluation categories should remain visible regardless of scale.

Engineering Intent

Students should define problems before implementation.

Evidence Quality

Students should support engineering claims.

Review Participation

Students should experience accountability.

AI Accountability

Students should practice responsible AI use.

Release Readiness

Students should defend engineering work.

Operational Thinking

Students should think beyond implementation.

Professional Ownership

Students should own outcomes.

These categories remain constitutional ETIS capabilities.

Scaling Dimensions

Most scaling occurs across five dimensions.

Enrollment
↓
Calendar
↓
Learner Maturity
↓
Delivery Environment
↓
Instructor Resources

Each dimension should be handled intentionally.

Scaling Dimension 1 — Enrollment Size

Small Classes

Examples:

10-20 students

Recommended emphasis:

- individual coaching,
- personalized reviews,
- deeper discussions,
- richer defenses.

Evaluation may become highly personalized.

Medium Classes

Examples:

20-50 students

Recommended emphasis:

- peer reviews,
- standardized rubrics,
- structured checkpoints,
- reusable templates.

Balance instructor involvement and student ownership.

Large Classes

Examples:

50-120 students

Recommended emphasis:

- reusable systems,
- peer accountability,
- defense teams,
- standardized evidence evaluation.

Scale systems.

Do not reduce expectations.

Scaling Dimension 2 — Calendar Structures

Examples include:

- semesters,
- quarters,
- accelerated terms,
- modules,
- workshops.

Scaling strategy:

Adjust timing.

Do not remove evaluation categories.

Preserve:

- evidence,
- reviews,
- AI accountability,
- release readiness,
- ownership.

Educational progression should remain visible.

Scaling Dimension 3 — Learner Maturity

Different learners require different evaluation emphasis.

Undergraduate Students

Increase structure.

Increase guidance.

Increase scaffolding.

Graduate Students

Increase autonomy.

Increase critique.

Increase judgment.

Capstone Students

Increase accountability.

Increase operational thinking.

Increase ownership.

Professionals

Increase real-world application.

Reduce academic overhead.

Leaders

Increase governance emphasis.

Reduce technical detail.

Scaling Dimension 4 — Delivery Environment

Examples include:

Face-To-Face

Emphasize discussions and defenses.

Hybrid

Emphasize repositories and asynchronous reviews.

Online

Emphasize written evidence and structured accountability.

Asynchronous

Emphasize checkpoints and repository evidence.

Professional Training

Emphasize immediate applicability.

The environment may change.

The philosophy should not.

Scaling Dimension 5 — Instructor Resources

Different instructional resources require different systems.

Single Instructor

Emphasize automation, templates, and peer reviews.

Instructor With Teaching Assistants

Distribute review responsibilities.

Use consistent rubrics.

Multi-Instructor Environments

Standardize evaluation systems.

Normalize evidence expectations.

External Reviewers

Introduce authentic engineering defenses.

Different resources require different workflows.

The educational philosophy should remain stable.

Recommended Scaling Strategies

When scaling up, prefer:

```
More Templates
   ↓
More Rubrics
   ↓
More Peer Reviews
   ↓
More Structured Evidence Systems
```

Avoid:

```
More Administrative Overhead
   ↓
More Manual Grading
   ↓
More Isolated Assessments
```

Scalability should reduce friction.

What May Scale

The following may change.

Team Size

May change.

Number Of Exercises

May change.

Technical Complexity

May change.

Review Frequency

May change.

Delivery Method

May change.

These are implementation decisions.

What Should Not Scale Down

The following should remain visible.

Evidence

Should remain visible.

Accountability

Should remain visible.

AI Governance

Should remain visible.

Risk Communication

Should remain visible.

Operational Thinking

Should remain visible.

Professional Ownership

Should remain visible.

These remain constitutional ETIS capabilities.

Instructor Scaling Philosophy

Instructor roles should evolve.

```
Small Courses
Coach
   ↓
Medium Courses
```

Facilitator

↓

Large Courses

System Designer

Instructors increasingly build systems that allow accountability to scale.

Assessment Automation Philosophy

As environments scale, automation should support instructors.

Automation may assist with:

- rubric consistency,
- evidence tracking,
- checkpoint monitoring,
- feedback organization.

Automation should never replace engineering evaluation.

Humans should retain ownership.

Students should understand:

> AI proposes; engineers verify.

This doctrine applies to instructors as well.

Common Assessment Scaling Mistakes

Mistake 1 — Reducing Accountability

Scale mechanisms instead.

Mistake 2 — Over-Relying On Manual Grading

Build reusable systems.

Mistake 3 — Eliminating Reviews

Reviews are engineering mechanisms.

Mistake 4 — Ignoring AI Accountability

AI governance should remain visible.

Mistake 5 — Removing Operational Thinking

Students should understand life after implementation.

Mistake 6 — Treating Large Courses As Different ETIS Systems

ETIS should remain recognizable.

Mistake 7 — Prioritizing Efficiency Over Professional Formation

Professional transformation remains primary.

Instructor Scaling Questions

Before scaling assessment systems, ask:

- What accountability should remain?
- What evaluation mechanisms can change?
- What evidence should students create?
- What reviews should students experience?
- What ownership should students demonstrate?

These answers should drive scaling decisions.

ETIS Assessment Scaling Formula

ETIS assessment scaling can be summarized as:

Environment
↓
Evaluation Mechanisms
↓
Evidence
↓
Reviews
↓
Accountability
↓
Trustworthy Engineer

This formula should remain stable.

Guiding Standard

A successful ETIS scaling effort should still allow learners to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?
- What would we improve next?

If learners cannot answer these questions, ETIS has been over-scaled.

Core Commitment

The purpose of ETIS assessment scaling is not to make courses easier to administer.

The purpose is to preserve trustworthy engineering across many educational environments.

Institutions may vary.

Learners may vary.

Projects may vary.

Technologies may vary.

Engineering trust should not vary.

That consistency is one of the foundations of ETIS education.

ETIS COMP330 Assessment Reference

The **ETIS COMP330 Assessment Reference** document preserves the flagship ETIS assessment implementation currently used within Loyola University Chicago COMP330.

This document exists as a reference implementation.

It is not ETIS doctrine.

COMP330 consumes ETIS educational assets.

COMP330 also contributes proven educational assets back into the ETIS educational ecosystem.

The purpose of this document is to preserve one successful implementation model that future educators may study, adapt, and improve.

Educational provenance is intentionally preserved.

Purpose

The purpose of this document is to help educators understand how ETIS assessment can be implemented within a semester-long undergraduate software engineering course.

This document helps instructors answer questions such as:

- What does ETIS assessment look like in practice?
- How does engineering accountability increase?
- How does assessment evolve?
- How is AI governance evaluated?
- How is evidence evaluated?
- How do repositories become assessable systems?
- How do engineering defenses emerge?

COMP330 serves as one example.

It is not the only example.

Provenance

Institution:

Loyola University Chicago

Course:

COMP330 Software Engineering

Educational Role:

Flagship ETIS implementation

Educational Purpose:

Teach students how to engineer systems that can be understood, reviewed, governed, operated, improved, and trusted over time.

COMP330 Assessment Philosophy

COMP330 intentionally moves beyond traditional software engineering assessment.

The course is not organized around grading isolated assignments.

The course is organized around evaluating cumulative engineering accountability.

Students progressively mature throughout the semester.

Assessment continuously evolves.

Nothing truly starts over.

The course intentionally resembles professional engineering environments.

COMP330 evaluates engineering accountability before engineering sophistication.

COMP330 Professional Transformation Model

Students progressively evolve throughout the semester.

Student

↓

Responsible Engineer

↓

Reviewer

↓

Architect

↓

Release Defender

↓

Operational Thinker

↓

Future Trustworthy Engineer

Assessment intentionally supports this progression.

COMP330 Assessment Growth Model

Assessment emphasis intentionally changes throughout the semester.

Participation

↓

Contribution

↓

Decision Making

↓

Accountability

↓

Professional Ownership

Students should feel this progression.

What COMP330 Evaluates

COMP330 intentionally evaluates multiple engineering dimensions.

Engineering Intent

↓

Evidence Quality

↓

Engineering Reasoning

↓

Review Participation

↓

AI Accountability

↓

Release Readiness

↓

Operational Thinking

↓

Professional Ownership

Students are evaluated across all dimensions.

Not every dimension appears equally throughout the entire semester.

Their visibility progressively increases.

Assessment Stage 1 — Engineering Foundations

Educational Goal

Establish professional expectations.

Students begin learning:

- repository-centered engineering,
- evidence-centered engineering,
- team accountability,
- AI governance.

Examples of evaluation include:

- participation,
- repository setup,
- communication,
- team engagement.

Primary evaluation question:

> Are students beginning to behave like engineers?

Assessment Stage 2 — Intent And Planning

Educational Goal

Evaluate engineering intent.

Students begin learning:

- requirements,
- use cases,
- planning,
- estimation,
- prioritization.

Examples of evaluation include:

- requirements quality,
- planning quality,
- decomposition quality,
- risk identification.

Primary evaluation question:

> Have students clearly defined the work?

Assessment Stage 3 — Architecture And Decision Making

Educational Goal

Evaluate engineering reasoning.

Students begin learning:

- architecture decisions,
- tradeoffs,
- system boundaries,
- reviews.

Examples of evaluation include:

- architectural clarity,
- tradeoff explanations,
- design justification,
- review participation.

Primary evaluation question:

> Can students explain why the system was built this way?

Assessment Stage 4 — Construction And Verification

Educational Goal

Evaluate engineering discipline.

Students begin learning:

- integration,
- testing,
- pull requests,
- AI-assisted engineering,
- verification.

Examples of evaluation include:

- validation evidence,
- review quality,
- AI disclosures,
- engineering discipline.

Primary evaluation question:

> What evidence supports engineering claims?

Assessment Stage 5 — Release Accountability

Educational Goal

Evaluate engineering defensibility.

Students begin learning:

- release readiness,
- risk communication,
- peer accountability.

Examples of evaluation include:

- release recommendations,
- risk assessments,
- evidence quality.

Primary evaluation question:

> Why should this release be trusted?

Assessment Stage 6 — Professional Ownership

Educational Goal

Evaluate full engineering accountability.

Students explain:

- what they intended,
- what they built,
- how AI assisted,
- what humans verified,
- what risks remain,
- how the system would operate,
- what they would improve.

Examples of evaluation include:

- engineering defenses,
- evidence narratives,
- stewardship reflections.

Primary evaluation question:

> Would another engineer trust this work?

AI Assessment Within COMP330

AI is intentionally allowed.

Students are evaluated on:

- AI disclosures,
- AI verification,
- AI limitations,
- AI governance decisions.

Students learn:

> AI proposes; engineers verify.

AI is treated as an engineering capability rather than an engineering replacement.

Evidence Assessment Within COMP330

Evidence is a first-class assessment category.

Students are evaluated on:

- evidence quality,
- evidence completeness,
- evidence traceability,
- evidence trustworthiness.

Students learn:

> Claims require evidence.

Repository Assessment Within COMP330

Repositories are engineering systems.

Repositories preserve:

- engineering decisions,
- evidence,
- reviews,
- AI usage,
- risks,
- engineering history.

Repositories themselves become assessable systems.

Students learn:

> Repositories are not submission folders.

Engineering Defenses Within COMP330

Engineering defenses become one of the strongest assessment mechanisms.

Students explain:

- intent,
- architecture,
- evidence,
- AI involvement,
- risks,
- operational considerations,
- future improvements.

Students experience authentic engineering accountability.

Why COMP330 Exists Inside ETIS

COMP330 serves multiple purposes.

It is:

- a flagship implementation,
- a proving ground,
- an educational laboratory,
- a reusable asset contributor.

COMP330 informs ETIS.

ETIS informs COMP330.

This relationship should remain visible.

Adaptation Guidance

Future educators should not copy COMP330 directly.

Instead, educators should ask:

- What educational philosophy should be preserved?
- What accountability progression should be preserved?
- What evidence systems should be preserved?
- What evaluation systems should be preserved?

Adaptations should preserve doctrine rather than duplicate implementation.

What Should Be Preserved

The following ideas should remain visible.

- repository-centered engineering,
- evidence-centered engineering,
- AI governance,
- architectural thinking,
- review systems,
- release readiness,
- operational awareness,

- cumulative accountability.

These ideas matter more than exact grading systems.

Common Misconceptions

Misconception 1 — COMP330 Is ETIS

COMP330 is one implementation.

Misconception 2 — COMP330 Is Assignment-Centered

COMP330 is accountability-centered.

Misconception 3 — Working Software Is Success

Working software is only one dimension.

Misconception 4 — AI Should Be Avoided

AI should be governed.

Misconception 5 — Repositories Are Submission Systems

Repositories are engineering systems.

Guiding Standard

A successful ETIS implementation should allow students to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How would this system operate?
- Why should this work be trusted?
- What would we improve next?

COMP330 intentionally builds toward these questions.

Core Commitment

The purpose of preserving COMP330 is not to standardize ETIS.

The purpose is to preserve engineering memory.

COMP330 demonstrates one successful approach for teaching software engineering in the AI era.

Future implementations may evolve.

The underlying ETIS doctrine should remain stable.

Educational assessment should progressively resemble professional engineering evaluation.

That progression is one of the foundations of ETIS education.

ETIS Engineering Defense Guidance

The **ETIS Engineering Defense Guidance** document provides recommendations for designing, conducting, and evaluating engineering defenses throughout ETIS educational experiences.

This document intentionally moves beyond traditional presentations.

ETIS treats engineering defenses as professional accountability experiences.

Students should progressively learn how to explain, justify, review, and defend engineering decisions.

The objective is not to present completed work.

The objective is to demonstrate trustworthy engineering.

Engineering work is not complete until it can be defended.

Purpose

The purpose of this document is to help instructors build authentic engineering defense experiences.

This document helps instructors answer questions such as:

- What is an engineering defense?
- Why are engineering defenses important?
- How should defenses be conducted?
- What should students defend?
- How should AI usage be discussed?
- How should risks be discussed?
- How should engineering accountability be evaluated?

Engineering defenses should reinforce professional ownership.

Guiding Principle

Every engineering defense should answer one question:

> Why should another engineer trust this work?

Students should repeatedly answer this question.

It should become a professional habit.

Educational Philosophy

Traditional educational experiences often end with presentations.

Examples include:

```
Demonstration
   ↓
Slides
   ↓
Questions
   ↓
```

Grade

ETIS intentionally expands this model.

Evidence

↓

Engineering Narrative

↓

Review

↓

Defense

↓

Accountability

↓

Trust

Students should experience authentic engineering accountability.

Foundational Principle

Engineering work is not complete until it can be defended.

Students should progressively learn how to explain:

- what they built,
- why they built it,
- what evidence supports it,
- how AI assisted,
- what risks remain,
- how it would operate,
- why it should be trusted.

These should become normal engineering behaviors.

What Is An Engineering Defense?

An engineering defense is a structured explanation of engineering decisions supported by evidence.

Students should explain:

- intent,
- decisions,
- evidence,
- reviews,
- AI involvement,
- risks,
- operational considerations,
- future improvements.

Defenses should resemble professional engineering discussions.

They should not resemble marketing presentations.

ETIS Defense Architecture

Most ETIS engineering defenses should follow this progression.

Intent
↓
Architecture
↓
Evidence
↓
Reviews
↓
AI Governance
↓
Release Readiness
↓
Operations
↓
Stewardship

Students should become comfortable discussing every stage.

Defense Section 1 — Intent

Educational Goal

Explain the problem before explaining the solution.

Questions include:

- What problem were we solving?
- Who were the stakeholders?
- What constraints existed?
- What assumptions existed?

Students should understand:

> Intent precedes implementation.

Defense Section 2 — Architecture

Educational Goal

Explain engineering decisions.

Questions include:

- Why was this architecture selected?
- What alternatives were considered?
- What tradeoffs were accepted?
- What boundaries were established?

Students should understand:

> Governance is architecture.

Defense Section 3 — Evidence

Educational Goal

Support engineering claims.

Questions include:

- What evidence exists?
- What evidence supports claims?
- What evidence is strongest?
- What evidence is weakest?

Students should understand:

> Claims require evidence.

Defense Section 4 — Reviews

Educational Goal

Normalize engineering accountability.

Questions include:

- What was reviewed?
- Who reviewed it?
- What feedback was received?
- What improvements occurred?

Students should understand:

> Engineering is a reviewable discipline.

Defense Section 5 — AI Governance

Educational Goal

Normalize AI transparency.

Questions include:

- How did AI assist?
- Why was AI used?
- What was verified?
- What limitations were discovered?

Students should understand:

> AI proposes; engineers verify.

Defense Section 6 — Release Readiness

Educational Goal

Teach engineering defensibility.

Questions include:

- Why should this release be trusted?
- What risks remain?
- What limitations remain?
- What evidence supports release?

Students should understand:

> A demo is not operational proof.

Defense Section 7 — Operational Thinking

Educational Goal

Teach life beyond implementation.

Questions include:

- How would this system survive reality?
- How would incidents be handled?
- How would the system be monitored?
- What improvements would be prioritized?

Students should understand:

> The model is not the system.

Defense Section 8 — Stewardship

Educational Goal

Teach continuous ownership.

Questions include:

- What would we improve next?
- What technical debt exists?
- What future risks exist?
- What should future engineers know?

Students should understand:

> Engineering ownership continues after delivery.

Recommended Defense Sequence

Students should organize defenses using a consistent flow.

Problem

↓

Intent

↓

Architecture

↓

Evidence

↓

Reviews

↓

AI Usage

↓

Release Readiness

↓

Operations

↓

Future Improvements

This sequence should remain stable.

Recommended Defense Boards

ETIS encourages structured review boards.

Examples include:

Peer Defense Board

Students review one another.

Instructor Defense Board

Instructor serves as reviewer.

Multi-Instructor Defense Board

Multiple evaluators participate.

External Professional Board

Industry participants evaluate work.

Mixed Defense Board

Students, instructors, and professionals participate together.

Different environments may choose different models.

Recommended Defense Questions

Every engineering defense should include questions such as:

Intent Questions

- What problem did you solve?
- Why does this problem matter?

Architecture Questions

- Why was this design selected?
- What alternatives were considered?

Evidence Questions

- What evidence supports your claims?
- What evidence is still missing?

AI Questions

- How did AI assist?
- What did humans verify?

Risk Questions

- What risks remain?
- Which risks concern you most?

Operational Questions

- How would this system survive production?

Stewardship Questions

- What would you improve next?

Recommended Defense Maturity Model

Students should progressively evolve.

Explain Work

↓

Explain Decisions

↓

Defend Decisions

↓

Defend Evidence

↓

Own Outcomes

Students should experience this progression.

Repository Expectations

Repositories should support defenses.

Repositories should preserve:

- evidence,
- decisions,
- reviews,
- risks,
- AI usage,
- engineering history.

Students should understand:

> Repositories are engineering systems.

Instructor Role Progression

Instructor responsibilities should evolve.

Early
Teacher
↓
Middle
Coach
↓
Later
Reviewer
↓
Final
Engineering Defense Board

Students should progressively assume ownership.

Common Engineering Defense Mistakes

Mistake 1 — Treating Defenses As Presentations

Defenses are accountability experiences.

Mistake 2 — Focusing On Features

Focus on engineering decisions.

Mistake 3 — Ignoring Evidence

Evidence should drive discussions.

Mistake 4 — Hiding AI Usage

AI should be transparent.

Mistake 5 — Ignoring Risks

Risks should be visible.

Mistake 6 — Ignoring Operations

Students should think beyond implementation.

Mistake 7 — Ending With Demonstrations

Demonstrations should support defenses.

They should not replace them.

Adaptation Rules

Educational environments may adapt engineering defenses.

However, defenses should preserve discussions around:

- engineering intent,
- architecture,
- evidence,
- reviews,
- AI accountability,
- release readiness,
- operational thinking,
- stewardship.

These remain constitutional ETIS capabilities.

Instructor Design Questions

Before building a defense experience, ask:

- What decisions should students explain?
- What evidence should students defend?
- What risks should students communicate?
- What AI usage should be disclosed?
- What ownership should students demonstrate?

These answers should drive defense design.

Guiding Standard

A successful ETIS engineering defense should allow learners to answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?
- What would we improve next?

If learners cannot answer these questions, the defense experience should be redesigned.

Core Commitment

The purpose of ETIS engineering defenses is not to create stronger presentations.

The purpose is to create stronger engineers.

Students should progressively learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

Engineering accountability should become a normal engineering behavior.

That progression is one of the foundations of ETIS education.

ETIS Evidence-Based Assessment Guide

The **ETIS Evidence-Based Assessment Guide** provides recommendations for evaluating engineering evidence throughout ETIS educational experiences.

This document intentionally moves beyond traditional artifact evaluation.

ETIS teaches students that engineering claims require supporting evidence.

Students should progressively learn how to create, organize, review, explain, and defend engineering evidence.

The objective is not simply to evaluate outputs.

The objective is to evaluate whether engineering conclusions are supported by trustworthy evidence.

Evidence should become a first-class educational asset.

Purpose

The purpose of this document is to help instructors intentionally evaluate engineering evidence.

This document helps instructors answer questions such as:

- What evidence should students create?
- How should evidence be evaluated?
- What makes evidence trustworthy?
- How should evidence mature?
- How should repositories preserve evidence?
- How should evidence support engineering decisions?
- How should evidence support release decisions?

Educational evaluation should reinforce evidence-centered engineering.

Guiding Principle

Evidence-based assessment should answer one question:

> What evidence supports this engineering claim?

Students should repeatedly ask this question throughout ETIS educational experiences.

It should become a professional habit.

Educational Philosophy

Traditional educational evaluation often looks like this:

```
Assignment
    ↓
Submission
    ↓
Grade
```

ETIS intentionally expands evaluation.

Engineering Claim

↓

Supporting Evidence

↓

Review

↓

Verification

↓

Accountability

↓

Trust

Students should learn that evidence creates trust.

Foundational Principle

Evaluate the strength of the evidence before evaluating the strength of the conclusion.

Strong conclusions built on weak evidence should not be trusted.

Students should learn this distinction early.

Why Evidence Matters

Evidence allows engineering work to be:

- understood,
- reviewed,
- governed,
- operated,
- improved,
- and trusted over time.

Without evidence, engineering becomes opinion.

Students should understand this relationship.

ETIS Evidence Categories

Most ETIS educational experiences should evaluate evidence across eight categories.

Intent Evidence

↓

Planning Evidence

↓

Architecture Evidence

↓

Review Evidence

↓

Validation Evidence

↓

Release Evidence

↓

Operational Evidence

↓

Defense Evidence

Evidence should progressively mature.

Evidence Category 1 — Intent Evidence

Purpose

Evaluate problem definition quality.

Questions include:

- Is the problem clearly defined?
- Are stakeholders identified?
- Are assumptions visible?
- Are constraints documented?

Examples include:

- requirements,
- use cases,
- problem statements.

Students should understand:

> Intent precedes implementation.

Evidence Category 2 — Planning Evidence

Purpose

Evaluate engineering organization.

Questions include:

- Is work decomposed?
- Are responsibilities assigned?
- Are risks identified?
- Are priorities visible?

Examples include:

- WBS,
- RACI matrices,
- schedules,

- risk registers.

Students should understand:

> Planning reduces uncertainty.

Evidence Category 3 — Architecture Evidence

Purpose

Evaluate engineering decisions.

Questions include:

- Are decisions justified?
- Are boundaries visible?
- Are tradeoffs explained?
- Are alternatives considered?

Examples include:

- architecture diagrams,
- ADRs,
- dependency maps.

Students should understand:

> Governance is architecture.

Evidence Category 4 — Review Evidence

Purpose

Evaluate accountability mechanisms.

Questions include:

- Was work reviewed?
- Was feedback meaningful?
- Was feedback incorporated?
- Did quality improve?

Examples include:

- peer reviews,
- architecture reviews,
- pull request reviews.

Students should understand:

> Engineering is a reviewable discipline.

Evidence Category 5 — Validation Evidence

Purpose

Evaluate verification activities.

Questions include:

- Was functionality verified?
- Were assumptions tested?
- Were integrations validated?
- Were AI outputs verified?

Examples include:

- test results,
- verification summaries,
- integration evidence.

Students should understand:

> AI proposes; engineers verify.

Evidence Category 6 — Release Evidence

Purpose

Evaluate engineering defensibility.

Questions include:

- Why should this release be trusted?
- What risks remain?
- What limitations exist?
- What evidence supports deployment?

Examples include:

- release readiness packages,
- risk summaries,
- recommendations.

Students should understand:

> A demo is not operational proof.

Evidence Category 7 — Operational Evidence

Purpose

Evaluate life after implementation.

Questions include:

- How will the system be monitored?
- How will failures be handled?
- What improvements are needed?
- What risks remain?

Examples include:

- postmortems,
- observability plans,
- reliability analyses.

Students should understand:

> The model is not the system.

Evidence Category 8 — Defense Evidence

Purpose

Evaluate professional accountability.

Questions include:

- Can decisions be explained?
- Can risks be communicated?
- Can evidence be defended?
- Can ownership be demonstrated?

Examples include:

- engineering defenses,
- engineering narratives,
- stewardship reflections.

Students should understand:

> Engineering accountability is part of engineering itself.

Evidence Quality Dimensions

Every evidence category should be evaluated across multiple dimensions.

Completeness

Is required information present?

Accuracy

Is the evidence correct?

Clarity

Is the evidence understandable?

Traceability

Can evidence be connected to engineering decisions?

Sufficiency

Is the evidence adequate?

Actionability

Can another engineer use this evidence?

Trustworthiness

Can another engineer trust this evidence?

Evidence Maturity Model

Evidence should progressively evolve.

    Create Evidence

↓

Organize Evidence

↓

Explain Evidence

↓

Review Evidence

↓

Defend Evidence

↓

Steward Evidence

Students should experience this progression.

Repository Expectations

Repositories should become evidence systems.

Repositories should preserve:

- decisions,
- evidence,
- reviews,
- risks,
- AI usage,
- engineering history.

Students should understand:

> Repositories are engineering systems.

AI Evidence Expectations

AI introduces additional evidence requirements.

Students should preserve evidence regarding:

- AI disclosures,
- AI verification,
- AI limitations,
- AI decisions.

AI involvement should be transparent.

Recommended Instructor Role Progression

Instructor responsibilities should evolve.

Early

Teacher

↓

Middle
Coach
↓
Later
Reviewer
↓
Final
Engineering Defense Board

Students should progressively assume ownership.

Common Evidence Assessment Mistakes

Mistake 1 — Assessing Conclusions Instead Of Evidence

Evidence should be evaluated first.

Mistake 2 — Treating Evidence As Documentation

Evidence supports engineering claims.

Mistake 3 — Waiting Until The End To Create Evidence

Evidence should accumulate continuously.

Mistake 4 — Ignoring AI Evidence

AI accountability requires evidence.

Mistake 5 — Overemphasizing Quantity

Strong evidence matters more than abundant evidence.

Mistake 6 — Treating Presentations As Evidence

Presentations summarize evidence.

They are not evidence themselves.

Mistake 7 — Ignoring Traceability

Evidence should connect decisions to outcomes.

Adaptation Rules

Educational environments may adapt implementation details.

However, evidence-based assessment should preserve evaluation of:

- evidence quality,
- evidence traceability,
- evidence sufficiency,
- evidence trustworthiness,

- evidence accountability.

These remain constitutional ETIS capabilities.

Instructor Assessment Questions

Before evaluating evidence, ask:

- What claim is being made?
- What evidence supports the claim?
- Is the evidence trustworthy?
- Is the evidence sufficient?
- Can another engineer understand the evidence?
- Can another engineer act upon the evidence?

These questions should drive evaluation.

Guiding Standard

A successful ETIS educational experience should allow learners to answer:

- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- Why should this release be trusted?
- How will this system operate?
- Why should another engineer trust our work?

If learners cannot answer these questions, the evidence system should be redesigned.

Core Commitment

The purpose of ETIS evidence-based assessment is not to create more documentation.

The purpose is to create stronger engineering memory.

Students should progressively learn how to create evidence that allows systems to be understood, reviewed, governed, operated, improved, and trusted over time.

That progression is one of the foundations of ETIS education.

ETIS Rubric Design Guide

The **ETIS Rubric Design Guide** provides recommendations for designing rubrics that evaluate engineering maturity throughout ETIS educational experiences.

This document intentionally moves beyond traditional grading rubrics.

ETIS rubrics should evaluate engineering behaviors, engineering thinking, and engineering accountability.

The objective is not to determine whether students completed a task.

The objective is to determine whether students are becoming trustworthy engineers.

Rubrics should reinforce that progression.

Purpose

The purpose of this document is to help instructors build ETIS-aligned rubrics.

This document helps instructors answer questions such as:

- What should ETIS rubrics evaluate?
- How should rubric categories differ from traditional courses?
- How should AI accountability be evaluated?
- How should evidence be evaluated?
- How should professional ownership be evaluated?
- How should rubric complexity evolve?
- How should rubrics reinforce trustworthy engineering?

Rubrics should become professional evaluation systems rather than grading checklists.

Guiding Principle

Every ETIS rubric should answer one question:

> What engineering behaviors should learners demonstrate?

Rubrics should reinforce behaviors rather than reward task completion alone.

Educational Philosophy

Traditional rubrics often emphasize artifacts.

Examples include:

Correctness

↓

Completeness

↓

Formatting

↓

Documentation

ETIS intentionally expands evaluation.

Intent

↓

Evidence

↓

Engineering Reasoning

↓

Reviews

↓

AI Accountability

↓

Release Readiness

↓

Operational Thinking

↓

Professional Ownership

Educational evaluation should progressively resemble professional engineering evaluation.

Foundational Principle

Rubrics should measure trustworthiness, not just task completion.

Students should progressively learn how to become engineers others can trust.

Rubrics should reinforce those behaviors.

ETIS Rubric Architecture

Most ETIS rubrics should be built from eight reusable dimensions.

Engineering Intent

↓

Evidence Quality

↓

Engineering Reasoning

↓

Review Participation

↓

AI Accountability

↓

Release Readiness

↓

Operational Thinking

↓

Professional Ownership

Not every educational experience requires all eight dimensions.

However, these dimensions should remain visible throughout ETIS.

Dimension 1 — Engineering Intent

Educational Goal

Evaluate whether students clearly define problems before building solutions.

Questions include:

- Is the problem well defined?
- Are stakeholders identified?
- Are assumptions visible?
- Are constraints visible?

Students should understand:

> Intent precedes implementation.

Dimension 2 — Evidence Quality

Educational Goal

Evaluate whether engineering claims are supported.

Questions include:

- Is evidence complete?
- Is evidence understandable?
- Is evidence sufficient?
- Is evidence trustworthy?

Students should understand:

> Claims require evidence.

Dimension 3 — Engineering Reasoning

Educational Goal

Evaluate engineering decisions.

Questions include:

- Can decisions be explained?
- Can tradeoffs be justified?
- Are alternatives considered?
- Are risks identified?

Students should understand:

> Engineering is a decision-making discipline.

Dimension 4 — Review Participation

Educational Goal

Evaluate accountability behaviors.

Questions include:

- Did students participate in reviews?
- Did students provide useful feedback?
- Did students respond to feedback?
- Did quality improve?

Students should understand:

> Engineering is a reviewable discipline.

Dimension 5 — AI Accountability

Educational Goal

Evaluate responsible AI use.

Questions include:

- Was AI disclosed?
- Was AI verified?
- Were limitations identified?
- Did humans retain ownership?

Students should understand:

> AI proposes; engineers verify.

Dimension 6 — Release Readiness

Educational Goal

Evaluate engineering defensibility.

Questions include:

- Is the work trustworthy?
- Are risks communicated?
- Are limitations understood?
- Is release readiness justified?

Students should understand:

> A demo is not operational proof.

Dimension 7 — Operational Thinking

Educational Goal

Evaluate thinking beyond implementation.

Questions include:

- How will the system be monitored?
- How will failures be handled?
- What improvements are needed?
- What risks remain?

Students should understand:

> The model is not the system.

Dimension 8 — Professional Ownership

Educational Goal

Evaluate end-to-end accountability.

Questions include:

- Can students defend decisions?
- Can students explain evidence?
- Can students communicate risks?
- Can students own outcomes?

Students should understand:

> Engineering accountability is part of engineering itself.

Recommended Rubric Performance Levels

ETIS rubrics should use maturity language rather than completion language.

Recommended progression:

Emerging
↓
Developing
↓
Proficient
↓
Advanced
↓
Professional

Avoid:

Poor

Average

Good

Excellent

Maturity language reinforces growth.

Recommended Rubric Construction Process

Build rubrics in this order.

Engineering Responsibility
↓
Observable Behaviors
↓
Evidence Expectations
↓
Accountability Questions
↓
Performance Levels
↓
Rubric

This order should remain stable.

Early Course Rubrics

Early rubrics should emphasize:

- participation,
- communication,
- engineering habits,
- repository behaviors.

Do not overemphasize technical sophistication.

Mid-Course Rubrics

Mid-course rubrics should emphasize:

- architecture,
- decisions,
- evidence,
- reviews,
- AI accountability.

Professional behaviors should become more visible.

Late Course Rubrics

Late rubrics should emphasize:

- release readiness,
- operational thinking,
- accountability,
- ownership.

Students should increasingly defend work.

Repository Expectations

Repositories should become assessable systems.

Repositories should preserve:

- evidence,
- decisions,
- reviews,
- risks,
- AI usage,
- engineering history.

Repositories should support rubric evaluation.

Students should understand:

> Repositories are engineering systems.

AI Rubric Expectations

AI should appear explicitly.

Do not hide AI expectations.

Rubrics should evaluate:

- AI transparency,
- AI verification,
- AI limitations,
- AI accountability.

Avoid evaluating AI avoidance.

Evaluate AI responsibility instead.

Common Rubric Mistakes

Mistake 1 — Overemphasizing Correctness

Correctness is only one dimension.

Mistake 2 — Ignoring Evidence

Evidence is foundational.

Mistake 3 — Ignoring Reviews

Reviews are engineering mechanisms.

Mistake 4 — Hiding AI Expectations

AI expectations should be explicit.

Mistake 5 — Overweighting Presentations

Presentations summarize engineering work.

They do not replace evidence.

Mistake 6 — Assessing Technical Complexity Alone

Complexity is not maturity.

Mistake 7 — Ignoring Professional Ownership

Ownership should progressively increase.

Adaptation Rules

Educational environments may adapt rubric implementations.

However, rubrics should preserve evaluation of:

- engineering intent,
- evidence,
- engineering reasoning,
- reviews,
- AI accountability,
- release readiness,
- operational thinking,
- professional ownership.

These remain constitutional ETIS capabilities.

Instructor Rubric Questions

Before finalizing a rubric, ask:

- What engineering behavior am I evaluating?
- What evidence supports evaluation?
- What decisions should students explain?
- What risks should students communicate?
- What ownership should students demonstrate?

These answers should drive rubric design.

Guiding Standard

A successful ETIS rubric should allow instructors to determine whether learners can answer:

- What problem did we solve?
- What evidence supports our claims?
- How did AI assist?
- What did humans verify?
- What risks remain?
- How will this system operate?
- Why should this work be trusted?
- What would we improve next?

If rubrics cannot evaluate these questions, they should be redesigned.

Core Commitment

The purpose of ETIS rubric design is not to create stronger grading sheets.

The purpose is to create stronger engineers.

Rubrics should progressively help students learn how to define intent, engineer context, bound authority, verify behavior, operate reality, explain decisions, and own outcomes.

That progression is one of the foundations of ETIS education.

Appendix

Course Design Reference

ETIS 39-Chapter Learning Map

Purpose

The **ETIS 39-Chapter Learning Map** translates the complete structure of *Engineering Trustworthy Intelligent Systems* into educational use.

The ETIS book is intentionally comprehensive. It is not expected that every course, workshop, professional training program, or adoption model will use all 39 chapters with equal emphasis.

This map helps instructors, students, engineers, leaders, and organizations decide how to use the book based on learning goals, professional roles, available instructional time, and desired engineering outcomes.

This document is not a table of contents.

It is an educational consumption map.

Relationship to the ETIS Book

The ETIS book remains complete, frozen, and authoritative.

This learning map does not reduce, replace, or rewrite the book.

Instead, it helps educational adopters decide how to emphasize the book responsibly.

A course may emphasize selected chapters.

A workshop may use a chapter cluster.

A professional training program may follow a role-specific path.

A student project may use only the chapters needed for a specific engineering phase.

That selective use is acceptable when it remains connected to the larger ETIS framework.

Relationship to Audience Learning Paths

This document should be read together with:

audience_learning_paths.md

The audience learning paths explain **who** is consuming ETIS.

This document explains **how the 39 chapters support educational use**.

Together, the two documents help instructors and adopters make responsible decisions about emphasis, sequencing, and depth.

Core Educational Principle

ETIS is a cumulative professional transformation framework.

The full book moves the reader through a progression:

Foundation
↓
Engineering
↓
Construction

↓

Validation

↓

Operation

↓

Governance

↓

Stewardship

Not every learner will travel through the entire progression at the same depth.

However, every ETIS learning path should preserve the central discipline:

Trustworthy intelligent systems require evidence, review, governance, operational awareness, human accountability, and stewardship over time.

The goal is not to read chapters for coverage.

The goal is to develop trustworthy engineering capability.

Learning Emphasis Levels

This map uses four levels of educational emphasis.

Core A chapter should be taught directly and connected to assignments, exercises, reviews, or assessments.

Supporting A chapter should be introduced and referenced, but may not require full instructional depth.

Selective A chapter should be used when relevant to the audience, project, or implementation.

Capstone A chapter should be used to frame professional identity, future practice, synthesis, or reflection.

These levels help instructors avoid the false choice between teaching everything equally or omitting important material entirely.

The Four-Part Learning Architecture

ETIS is organized into four major educational movements.

Book Part	Chapters	Educational Role	Core Learner Transformation
Part I — Foundations of Trustworthy Engineering	1-7	Establishes the engineering worldview	From programmer to responsible engineer
Part II — Engineering Construction	8-22	Teaches disciplined construction, evidence, review, and release defense	From builder to reviewer and release defender

Book Part	Chapters	Educational Role	Core Learner Transformation
Part III — Operational Trust	23-32	Extends engineering into operation, runtime evidence, governance, and organizational confidence	From release defender to operational thinker
Part IV — Trustworthy Intelligent Systems	33-39	Addresses intelligent systems, oversight, context, complexity, and stewardship	From operational thinker to future trustworthy engineer

The four parts should be understood as a professional development arc.

Part I establishes why trustworthy engineering matters.

Part II teaches how disciplined engineering work is created and defended.

Part III teaches what happens when systems enter operational reality.

Part IV teaches how engineers steward intelligent systems in an AI-shaped future.

Part I Learning Role: Foundations of Trustworthy Engineering

Part I establishes the worldview required for ETIS.

Learners encounter software engineering as a sociotechnical, evidence-centered, AI-aware, accountability-driven discipline.

Educational Function Part I teaches learners why software engineering must be more than coding.

It introduces failure, complexity, uncertainty, AI disruption, judgment, oversight, communication, review, and accountability.

Primary Learner Shift

Programmer

↓

Responsible Engineer

Typical Educational Use Part I should usually be introduced early in a course, workshop, or training path.

It provides the vocabulary needed for the rest of ETIS.

Chapters

Ch.	Chapter Title	Educational Role	Learner Capability
1	Engineering Trustworthy Intelligent Systems	Establishes the ETIS worldview	Understand trustworthiness as an engineering responsibility
2	Why Software Projects Fail	Explains why discipline is necessary	Recognize failure patterns in complexity, coordination, quality, and governance

Ch.	Chapter Title	Educational Role	Learner Capability
3	Complexity, Coordination, and Sociotechnical Systems	Frames systems as people-process-technology environments	Analyze sociotechnical risk and coordination complexity
4	Lifecycle Models and Engineering Under Uncertainty	Connects lifecycle choice to uncertainty and risk	Select lifecycle approaches based on consequence and uncertainty
5	AI Changes the Software Lifecycle	Introduces AI-era lifecycle disruption	Understand how AI affects requirements, design, coding, testing, and operations
6	Engineering Judgment and Human Oversight	Establishes accountability in AI-assisted work	Apply human judgment to generated or automated outputs
7	Teams, Communication, Review, and Accountability	Builds team engineering discipline	Coordinate work through roles, communication, review, and accountability

Evidence Learners Should Produce Learners should be able to produce evidence such as:

- Engineering responsibility reflections
- Failure-pattern analysis
- Team working agreements
- Communication expectations
- AI-use expectations
- Review participation evidence
- Initial project standards

Part II Learning Role: Engineering Construction

Part II teaches learners how to move from project launch through requirements, planning, architecture, implementation, review, testing, and release defense.

This part is central to undergraduate software engineering education and the COMP330 implementation.

Educational Function Part II teaches disciplined construction.

It connects intent, artifacts, repositories, decisions, AI-assisted work, testing, traceability, and release evidence.

Primary Learner Shift

Builder
↓
Reviewer
↓
Release Defender

Typical Educational Use Part II should usually carry the heaviest instructional load in a software engineering course.

It directly supports project checkpoints, repository work, review practices, testing, and final release defense.

Chapters

Ch.	Chapter Title	Educational Role	Learner Capability
8	Project Launch and Engineering Standards	Begins controlled engineering work	Establish team standards, workflow, and engineering expectations
9	Repository-Centered Engineering	Defines the repository as engineering memory	Use repositories as evidence systems, not just code storage
10	Requirements, Stakeholders, and Engineering the Right Problem	Teaches requirements discipline	Translate stakeholder needs into reviewable requirements
11	AI-Assisted Requirements Engineering	Shows responsible AI use in requirements	Use AI to critique and improve requirements without surrendering judgment
12	Planning, Estimation, Risk, and Tradeoffs	Connects planning to reality	Make work, estimates, risks, and tradeoffs visible
13	Architecture Fundamentals	Establishes structural design thinking	Define components, interfaces, responsibilities, and boundaries
14	Intelligent Systems Architecture	Extends architecture into AI-enabled systems	Design context, tools, permissions, orchestration, and governance boundaries
15	Architecture Reviews and ADRs	Makes decisions reviewable	Create and evaluate ADRs, review findings, and design tradeoffs
16	AI-Assisted Design and Coding	Governs AI-assisted construction	Use AI for implementation while preserving ownership and verification
17	Pull Requests, Reviews, and CI/CD	Teaches controlled change	Use branches, PRs, reviews, tests, and CI/CD as engineering controls
18	Reviewing AI-Generated Systems	Makes generated work inspectable	Review AI-generated code, artifacts, workflows, and risks
19	Testing and Verification Fundamentals	Establishes test evidence	Connect requirements to tests, defects, and verification evidence
20	Testing Intelligent and AI-Assisted Systems	Extends testing to AI-era uncertainty	Validate nondeterministic, AI-assisted, or context-sensitive behavior
21	Release Readiness and Engineering Evidence	Turns completion into defensible release judgment	Build release evidence, known limitations, and risk disposition
22	Engineering Presentations and Release Defense	Teaches evidence-backed communication	Defend engineering claims with repository evidence

Evidence Learners Should Produce Learners should be able to produce evidence such as:

- Repository structure
- Requirements
- Assumptions and open questions
- Risk register
- Architecture overview

- ADRs
- Pull requests
- Review evidence
- AI-use log
- Test plan
- Test evidence
- Defect log
- Release notes
- Release readiness review
- Engineering presentation or release defense

Part III Learning Role: Operational Trust

Part III extends engineering beyond construction and release.

It teaches learners that engineering responsibility continues after a system appears to work.

Educational Function Part III introduces operational evidence, stabilization, observability, runbooks, security governance, reliability, incident response, release authority, and organizational trust.

Primary Learner Shift

Release Defender

↓

Operational Thinker

Typical Educational Use Part III may be taught deeply in graduate courses, professional training, operations-focused courses, capstones, and advanced software engineering sequences.

In undergraduate project courses, Part III is often used selectively during Cycle 2, release readiness, postmortem work, and final reflection.

Chapters

Ch.	Chapter Title	Educational Role	Learner Capability
23	Postmortems and Engineering Learning	Converts problems into learning	Conduct postmortems and preserve lessons
24	Defect Reduction and Stabilization	Focuses on maturity after defects	Reduce defect patterns and stabilize systems
25	Observability and Runtime Evidence	Connects systems to runtime reality	Use logs, metrics, traces, and runtime evidence
26	Operational Readiness and Runbooks	Prepares systems for operation	Create runbooks, support paths, and readiness evidence
27	Security Engineering and Governance	Treats security as lifecycle responsibility	Identify permissions, controls, risks, and governance obligations
28	AI Governance and Controlled Delegation	Defines AI authority boundaries	Bound, review, audit, and control AI delegation
29	Reliability Engineering and Failure Analysis	Teaches failure-oriented engineering	Analyze reliability, failure modes, and recovery assumptions
30	Operational Incident Response	Teaches response under stress	Manage incidents, evidence, roles, communication, and recovery

Ch.	Chapter Title	Educational Role	Learner Capability
31	Release Governance and Approval Authority	Connects release to authority	Define who can approve, defer, reject, or escalate release decisions
32	Trust, Transparency, and Organizational Confidence	Connects evidence to institutional trust	Communicate trustworthiness through transparency and accountability

Evidence Learners Should Produce Learners should be able to produce evidence such as:

- Postmortem
- Defect trend analysis
- Observability notes
- Runtime evidence
- Runbook
- Security governance checklist
- Delegation boundary analysis
- Reliability assumptions
- Incident response notes
- Release approval rationale
- Transparency or trust summary

Part IV Learning Role: Trustworthy Intelligent Systems

Part IV addresses the future-facing responsibilities of engineers working with intelligent systems.

It focuses on agentic systems, context engineering, human oversight, understandability, complexity, stewardship, and professional identity.

Educational Function Part IV helps learners understand how trustworthy engineering evolves as systems become more intelligent, distributed, automated, and organizationally consequential.

Primary Learner Shift

Operational Thinker

↓

Future Trustworthy Engineer

Typical Educational Use Part IV may be used deeply in AI governance, architecture, graduate, enterprise, or professional training contexts.

In undergraduate courses, Part IV often works best as capstone framing, professional reflection, or future-practice orientation.

Chapters

Ch.	Chapter Title	Educational Role	Learner Capability
33	Agentic Systems and Workflow Orchestration	Introduces systems that act	Understand agents, workflows, orchestration, and control
34	Enterprise AI Architecture and Context Engineering	Defines enterprise AI context architecture	Engineer authoritative context, systems of record, and control planes

Ch.	Chapter Title	Educational Role	Learner Capability
35	Human Oversight in Intelligent Systems	Deepens oversight as system design	Design human review, approval, intervention, and accountability
36	Understandability and Operational Transparency	Makes systems explainable enough to govern	Improve transparency, interpretability, and operational understanding
37	Complexity, Cognitive Load, and Understandability	Connects complexity to human limits	Reduce cognitive load and design for maintainable understanding
38	Engineering Stewardship in the AI Era	Defines long-term responsibility	Preserve trust through maintenance, governance, and organizational memory
39	The Future Trustworthy Engineer	Culminates in professional identity	Integrate ETIS responsibilities into future engineering practice

Evidence Learners Should Produce Learners should be able to produce evidence such as:

- Agent boundary analysis
- Workflow orchestration review
- Context engineering map
- Human oversight plan
- Understandability review
- Cognitive load analysis
- Stewardship reflection
- Future engineer professional identity statement

COMP330 Selective Use Model

COMP330 should not attempt to teach all 39 chapters equally.

Instead, COMP330 should use ETIS as the governing framework for a semester-long software engineering experience.

COMP330 primarily emphasizes Parts I and II while selectively introducing Parts III and IV.

This reflects the practical reality of a one-semester undergraduate software engineering course that must combine reading, project work, teamwork, repository practice, reviews, testing, demonstrations, and release defense.

Primary COMP330 Coverage

COMP330 should strongly emphasize:

1. Engineering Trustworthy Intelligent Systems
2. Why Software Projects Fail
3. Lifecycle Models and Engineering Under Uncertainty
4. AI Changes the Software Lifecycle
5. Engineering Judgment and Human Oversight
6. Teams, Communication, Review, and Accountability
7. Project Launch and Engineering Standards
8. Repository-Centered Engineering
9. Requirements, Stakeholders, and Engineering the Right Problem
10. AI-Assisted Requirements Engineering
11. Planning, Estimation, Risk, and Tradeoffs

12. Architecture Fundamentals
13. Intelligent Systems Architecture
14. Architecture Reviews and ADRs
15. AI-Assisted Design and Coding
16. Pull Requests, Reviews, and CI/CD
17. Reviewing AI-Generated Systems
18. Testing and Verification Fundamentals
19. Testing Intelligent and AI-Assisted Systems
20. Release Readiness and Engineering Evidence
21. Engineering Presentations and Release Defense

These chapters align most directly with a team-based undergraduate software engineering course.

Chapter 3 may also be introduced selectively when discussing complexity, team coordination, sociotechnical systems, or project failure.

Secondary COMP330 Coverage

COMP330 should selectively introduce:

23. Postmortems and Engineering Learning
24. Defect Reduction and Stabilization
25. Observability and Runtime Evidence
26. Operational Readiness and Runbooks
27. Security Engineering and Governance
28. AI Governance and Controlled Delegation
29. Reliability Engineering and Failure Analysis
30. Operational Incident Response
31. Release Governance and Approval Authority
32. Trust, Transparency, and Organizational Confidence

These chapters are especially useful for Cycle 2, final presentations, operational maturity, release readiness, and professional reflection.

Capstone COMP330 Coverage

COMP330 should use the final part as a capstone framing device rather than full-depth coverage.

33. Agentic Systems and Workflow Orchestration
34. Enterprise AI Architecture and Context Engineering
35. Human Oversight in Intelligent Systems
36. Understandability and Operational Transparency
37. Complexity, Cognitive Load, and Understandability
38. Engineering Stewardship in the AI Era
39. The Future Trustworthy Engineer

These chapters help students understand where the profession is going and why their project evidence, AI-use discipline, engineering reviews, and release defense matter.

Chapter-by-Chapter Learning Map

Ch.	Chapter Title	Learning Role	Core Competency	Common Educa-tional Use
1	Engineering Trustworthy Intelligent Systems	Establishes the ETIS worldview	Understand trustworthiness as an engineering responsibility	Core
2	Why Software Projects Fail	Explains why discipline is necessary	Recognize failure patterns in complexity, coordination, quality, and governance	Core
3	Complexity, Coordination, and Sociotechnical Systems	Frames systems as people-process-technology environments	Analyze sociotechnical risk and coordination complexity	Supporting
4	Lifecycle Models and Engineering Under Uncertainty	Connects lifecycle choice to uncertainty and risk	Select lifecycle approaches based on consequence and uncertainty	Core
5	AI Changes the Software Lifecycle	Introduces AI-era lifecycle disruption	Understand how AI affects requirements, design, coding, testing, and operations	Core
6	Engineering Judgment and Human Oversight	Establishes accountability in AI-assisted work	Apply human judgment to generated or automated outputs	Core
7	Teams, Communication, Review, and Accountability	Builds team engineering discipline	Coordinate work through roles, communication, review, and accountability	Core
8	Project Launch and Engineering Standards	Begins controlled engineering work	Establish team standards, workflow, and engineering expectations	Core
9	Repository-Centered Engineering	Defines the repository as engineering memory	Use repositories as evidence systems, not just code storage	Core
10	Requirements, Stakeholders, and Engineering the Right Problem	Teaches requirements discipline	Translate stakeholder needs into reviewable requirements	Core
11	AI-Assisted Requirements Engineering	Shows responsible AI use in requirements	Use AI to critique and improve requirements without surrendering judgment	Core
12	Planning, Estimation, Risk, and Tradeoffs	Connects planning to reality	Make work, estimates, risks, and tradeoffs visible	Core
13	Architecture Fundamentals	Establishes structural design thinking	Define components, interfaces, responsibilities, and boundaries	Core
14	Intelligent Systems Architecture	Extends architecture into AI-enabled systems	Design context, tools, permissions, orchestration, and governance boundaries	Core
15	Architecture Reviews and ADRs	Makes decisions reviewable	Create and evaluate ADRs, review findings, and design tradeoffs	Core
16	AI-Assisted Design and Coding	Governs AI-assisted construction	Use AI for implementation while preserving ownership and verification	Core

Ch.	Chapter Title	Learning Role	Core Competency	Common Educa-tional Use
17	Pull Requests, Reviews, and CI/CD	Teaches controlled change	Use branches, PRs, reviews, tests, and CI/CD as engineering controls	Core
18	Reviewing AI-Generated Systems	Makes generated work inspectable	Review AI-generated code, artifacts, workflows, and risks	Core
19	Testing and Verification Fundamentals	Establishes test evidence	Connect requirements to tests, defects, and verification evidence	Core
20	Testing Intelligent and AI-Assisted Systems	Extends testing to AI-era uncertainty	Validate nondeterministic, AI-assisted, or context-sensitive behavior	Core
21	Release Readiness and Engineering Evidence	Turns completion into defensible release judgment	Build release evidence, known limitations, and risk disposition	Core
22	Engineering Presentations and Release Defense	Teaches evidence-backed communication	Defend engineering claims with repository evidence	Core
23	Postmortems and Engineering Learning	Converts problems into learning	Conduct postmortems and preserve lessons	Supporting
24	Defect Reduction and Stabilization	Focuses on maturity after defects	Reduce defect patterns and stabilize systems	Selective
25	Observability and Runtime Evidence	Connects systems to runtime reality	Use logs, metrics, traces, and runtime evidence	Supporting
26	Operational Readiness and Runbooks	Prepares systems for operation	Create runbooks, support paths, and readiness evidence	Supporting
27	Security Engineering and Governance	Treats security as lifecycle responsibility	Identify permissions, controls, risks, and governance obligations	Supporting
28	AI Governance and Controlled Delegation	Defines AI authority boundaries	Bound, review, audit, and control AI delegation	Supporting
29	Reliability Engineering and Failure Analysis	Teaches failure-oriented engineering	Analyze reliability, failure modes, and recovery assumptions	Selective
30	Operational Incident Response	Teaches response under stress	Manage incidents, evidence, roles, communication, and recovery	Selective
31	Release Governance and Approval Authority	Connects release to authority	Define who can approve, defer, reject, or escalate release decisions	Selective
32	Trust, Transparency, and Organizational Confidence	Connects evidence to institutional trust	Communicate trustworthiness through transparency and accountability	Supporting
33	Agentic Systems and Workflow Orchestration	Introduces systems that act	Understand agents, workflows, orchestration, and control	Capstone
34	Enterprise AI Architecture and Context Engineering	Defines enterprise AI context architecture	Engineer authoritative context, systems of record, and control planes	Capstone

Ch.	Chapter Title	Learning Role	Core Competency	Common Educational Use
35	Human Oversight in Intelligent Systems	Deepens oversight as system design	Design human review, approval, intervention, and accountability	Capstone
36	Understandability and Operational Transparency	Makes systems explainable enough to govern	Improve transparency, interpretability, and operational understanding	Capstone
37	Complexity, Cognitive Load, and Understandability	Connects complexity to human limits	Reduce cognitive load and design for maintainable understanding	Capstone
38	Engineering Stewardship in the AI Era	Defines long-term responsibility	Preserve trust through maintenance, governance, and organizational memory	Capstone
39	The Future Trustworthy Engineer	Culminates in professional identity	Integrate ETIS responsibilities into future engineering practice	Capstone

Competency Mapping

ETIS chapters can also be grouped by competency.

These groupings help instructors create modules, assignments, classroom exercises, workshops, and professional learning paths.

Engineering Mindset 1, 2, 3, 4, 5, 6, 7

Learners develop the worldview needed to treat software as a sociotechnical, evidence-based, governed engineering discipline.

Repository and Evidence Practice 8, 9, 15, 17, 21, 22, 23, 26, 38

Learners learn how to preserve engineering memory, support review, defend release claims, and steward knowledge over time.

Requirements, Planning, and Tradeoffs 10, 11, 12

Learners learn how to turn vague needs into bounded, reviewable, and manageable engineering commitments.

Architecture and Design 13, 14, 15, 34, 36, 37

Learners learn how to design boundaries, decisions, context, understandability, and governance structures.

AI-Assisted Engineering 5, 6, 11, 14, 16, 18, 20, 28, 33, 34, 35, 36

Learners learn how to use AI responsibly while preserving human accountability, verification, and control.

Testing, Verification, and Release 18, 19, 20, 21, 22, 31

Learners learn how to defend quality claims with evidence rather than confidence.

Operations and Runtime Trust 23, 24, 25, 26, 29, 30, 32

Learners learn how systems behave after release and how organizations observe, recover, learn, and improve.

Governance, Security, and Authority 6, 14, 18, 27, 28, 31, 33, 35, 36

Learners learn how authority, approval, security, auditability, oversight, and escalation must be engineered.

Stewardship and Professional Identity 32, 35, 36, 37, 38, 39

Learners learn how trustworthy engineering becomes a durable professional identity.

Recommended Chapter Bundles

Chapter bundles support selective adoption.

They should be treated as starting points, not rigid prescriptions.

Essential Undergraduate Bundle 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 16, 17, 19, 21, 22, 23, 25, 26, 27, 32, 39

Use when the course has limited time but needs broad ETIS exposure.

This bundle prioritizes professional formation, project discipline, repository-centered engineering, testing, release defense, operational awareness, and future professional identity.

COMP330 Project Bundle 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 28, 34, 39

Use for a team-based software engineering course with AI-era emphasis.

This bundle supports requirements, architecture, AI-assisted engineering, review, testing, release defense, and selective operational trust.

AI Governance Bundle 5, 6, 11, 14, 16, 18, 20, 27, 28, 31, 33, 34, 35, 36, 39

Use for AI governance, risk, oversight, or review-board training.

This bundle emphasizes human accountability, AI authority boundaries, verification, oversight, context, transparency, and future professional responsibility.

Operations and Reliability Bundle 21, 23, 24, 25, 26, 27, 29, 30, 31, 32, 38

Use for DevOps, SRE, operational readiness, incident response, and reliability training.

This bundle emphasizes release evidence, runtime behavior, failure analysis, incident response, authority, trust, and stewardship.

Architecture and Stewardship Bundle 3, 6, 9, 12, 13, 14, 15, 21, 26, 28, 34, 35, 36, 37, 38, 39

Use for architects, technical leads, graduate students, and organizational adoption.

This bundle emphasizes complexity, judgment, repositories, planning, architecture, AI delegation, context, oversight, understandability, cognitive load, stewardship, and professional identity.

Using This Map

This map should be used when designing:

- Course syllabi
- Weekly schedules
- Reading assignments
- Project checkpoints
- Classroom activities
- Review-board exercises
- Rubrics
- Professional workshops
- Student starter kits
- Instructor packages
- Adoption examples

The map should not be treated as a rigid requirement.

Its purpose is alignment.

A course may use fewer chapters.

A workshop may use only one chapter cluster.

A professional team may use a single bundle.

That is acceptable.

What matters is that each educational use of ETIS remains connected to the larger framework of evidence, governance, operations, accountability, and stewardship.

Guiding Rule

Every selective adoption of ETIS should be able to answer:

What part of the trustworthy engineering journey are we teaching, and what evidence will learners produce to prove they practiced it?

If the answer is clear, the adoption is likely aligned with ETIS.

If the answer is unclear, the educational design should be revised.

Core Thesis

The ETIS book is one complete framework.

There are many legitimate ways to teach it.

A responsible learning map does not weaken the framework by allowing selective use.

It strengthens the framework by helping each audience enter the trustworthy engineering journey at the right depth, with the right emphasis, and with clear expectations for evidence, judgment, and accountability.

Engineering Maturity Model

Purpose

This document defines the ETIS (Engineering Trustworthy Intelligent Systems) Engineering Maturity Model.

The purpose of this model is to describe how engineering behavior improves over time.

While the Professional Transformation Model explains who the learner is becoming, the Engineering Maturity Model explains how that growth becomes visible in practice.

This model helps instructors, learners, reviewers, and adopters recognize the difference between immature task completion and mature engineering responsibility.

Core Philosophy

Engineering maturity is not measured by confidence, activity, volume, or tool usage.

Engineering maturity is demonstrated through behavior.

A mature engineer defines intent clearly, preserves context, creates evidence, reviews work, verifies claims, communicates risks, uses AI responsibly, prepares for operation, and owns outcomes.

The purpose of this model is to make those behaviors observable.

Relationship to Professional Transformation

The Professional Transformation Model answers:

Who is the learner becoming?

The Engineering Maturity Model answers:

How does mature engineering behavior appear?

The models are complementary.

A learner may begin as a student, but their maturity grows as they demonstrate more disciplined engineering behavior across specific dimensions.

Maturity Levels

The ETIS Engineering Maturity Model uses five maturity levels.

Level 1: Task Completion

Level 2: Structured Participation

Level 3: Evidence-Based Engineering

Level 4: Reviewable Engineering Judgment

Level 5: Stewardship-Oriented Engineering

These levels are developmental.

They are not grades.

They are not labels to attach permanently to a learner.

They describe observable patterns of engineering behavior.

Level 1: Task Completion

Description At Level 1, the learner focuses primarily on completing assigned work.

The learner may produce code, documents, or project artifacts, but the work is often disconnected from broader engineering context.

Common Behaviors The learner may:

- Complete assigned tasks
- Produce required files
- Submit work on time
- Follow explicit instructions
- Use tools when directed

Limitations The learner may not yet:

- Understand why artifacts matter
- Connect work to requirements or risks
- Preserve context for others
- Explain decisions clearly
- Verify AI-assisted work deeply
- Think beyond immediate completion

Typical Mindset I did what was assigned.

Development Need The learner needs to move from task completion toward structured participation in an engineering system.

Level 2: Structured Participation

Description At Level 2, the learner begins participating in engineering practices with more consistency.

The learner understands that engineering work requires structure, coordination, communication, and artifacts.

Common Behaviors The learner may:

- Use the repository more consistently
- Participate in team communication
- Follow agreed workflows
- Create basic engineering artifacts
- Attend to requirements, risks, and roles
- Begin documenting AI usage
- Participate in reviews

Limitations The learner may still:

- Treat artifacts as forms rather than evidence
- Rely heavily on templates without understanding
- Produce shallow explanations
- Verify work inconsistently
- Participate in review passively

- Struggle to connect artifacts across the lifecycle

Typical Mindset I am following the engineering process.

Development Need The learner needs to move from following process toward creating evidence that supports engineering claims.

Level 3: Evidence-Based Engineering

Description At Level 3, the learner begins using artifacts, repositories, tests, reviews, and documentation as engineering evidence.

The learner understands that claims require support.

Common Behaviors The learner may:

- Connect requirements to design and testing
- Use the repository as an engineering memory system
- Preserve assumptions and decisions
- Create useful test evidence
- Document risks and limitations
- Explain AI-assisted contributions
- Prepare evidence for reviews or presentations

Limitations The learner may still:

- Need help evaluating evidence quality
- Miss deeper tradeoffs
- Treat review as confirmation rather than improvement
- Underestimate operational consequences
- Struggle to defend decisions under questioning

Typical Mindset We need evidence to support our claims.

Development Need The learner needs to move from evidence creation toward reviewable engineering judgment.

Level 4: Reviewable Engineering Judgment

Description At Level 4, the learner demonstrates the ability to make, explain, and defend engineering decisions.

The learner can evaluate alternatives, recognize risks, participate meaningfully in review, and revise work based on evidence.

Common Behaviors The learner may:

- Explain tradeoffs clearly
- Defend architectural and implementation choices
- Conduct meaningful reviews
- Challenge assumptions constructively
- Evaluate AI-generated outputs critically
- Connect release readiness to evidence
- Identify operational risks

- Communicate limitations honestly

Limitations The learner may still:

- Need more experience with real operational consequences
- Require support in complex governance situations
- Have limited exposure to long-term maintenance and stewardship
- Struggle with ambiguity under organizational pressure

Typical Mindset Can this decision be reviewed, defended, and improved?

Development Need The learner needs to move from reviewable judgment toward long-term stewardship.

Level 5: Stewardship-Oriented Engineering

Description At Level 5, the learner thinks beyond immediate delivery.

The learner understands that engineering work must remain trustworthy over time.

This level emphasizes operational reality, governance, transparency, human oversight, organizational memory, and sustained accountability.

Common Behaviors The learner may:

- Think in terms of lifecycle consequences
- Design for maintainability and understandability
- Preserve organizational learning
- Anticipate failure and recovery
- Bound AI and automation authority
- Support governance and oversight
- Communicate trustworthiness to stakeholders
- Own outcomes beyond initial delivery

Limitations Level 5 is aspirational in most undergraduate settings.

Students may be introduced to this level without fully mastering it.

Typical Mindset How do we preserve trust over time?

Development Need The learner continues developing professional depth through experience, reflection, and responsibility.

Maturity Dimensions

Engineering maturity should be observed across multiple dimensions.

The following dimensions help instructors and reviewers evaluate growth.

Dimension 1: Intent Definition

Immature Pattern The learner accepts tasks without clarifying purpose.

Mature Pattern The learner clarifies goals, stakeholders, constraints, and success criteria.

Evidence Possible evidence includes:

- Requirements
- Acceptance criteria
- Assumptions and open questions
- Stakeholder summaries
- Scope statements

Dimension 2: Context Engineering

Immature Pattern The learner assumes others already understand the work.

Mature Pattern The learner preserves context so teammates, reviewers, AI systems, instructors, and future maintainers can understand the work.

Evidence Possible evidence includes:

- README files
- Architecture notes
- Decision records
- Traceability documents
- Planning artifacts
- Review summaries

Dimension 3: Repository Discipline

Immature Pattern The repository is treated as a file dump or code container.

Mature Pattern The repository becomes an engineering memory system.

Evidence Possible evidence includes:

- Organized repository structure
- Meaningful commits
- Pull requests
- Documentation
- Test evidence
- Release evidence
- Operational notes

Dimension 4: Evidence Creation

Immature Pattern The learner makes claims without support.

Mature Pattern The learner supports claims with artifacts, tests, reviews, decisions, and traceability.

Evidence Possible evidence includes:

- Test plans
- Test results
- Traceability summaries
- Defect logs
- Release readiness reviews
- Known limitations

- AI verification notes

Dimension 5: Review Participation

Immature Pattern The learner treats review as approval or criticism.

Mature Pattern The learner treats review as engineering improvement.

Evidence Possible evidence includes:

- Pull request reviews
- Architecture review notes
- Requirements review findings
- Code review comments
- Release readiness feedback
- Updated artifacts after review

Dimension 6: AI Responsibility

Immature Pattern The learner uses AI output without sufficient understanding or verification.

Mature Pattern The learner uses AI as a bounded collaborator and remains accountable for results.

Evidence Possible evidence includes:

- AI-use logs
- AI verification notes
- Disclosures
- Human review comments
- Test evidence for AI-assisted work
- Explanation of accepted and rejected AI suggestions

Dimension 7: Testing and Verification

Immature Pattern The learner treats testing as a final task or demonstration support.

Mature Pattern The learner designs verification evidence throughout the lifecycle.

Evidence Possible evidence includes:

- Test strategy
- Test cases
- CI evidence
- Defect logs
- Regression notes
- Requirement-to-test traceability

Dimension 8: Release Readiness

Immature Pattern The learner equates working software with readiness.

Mature Pattern The learner distinguishes demo readiness from release readiness.

Evidence Possible evidence includes:

- Release notes
- Known limitations
- Risk disposition
- Traceability summary
- Release readiness review
- Engineering defense presentation

Dimension 9: Operational Thinking

Immature Pattern The learner assumes engineering ends at submission or demo.

Mature Pattern The learner considers runtime behavior, failure, support, monitoring, recovery, and improvement.

Evidence Possible evidence includes:

- Runbooks
- Observability notes
- Incident response notes
- Postmortems
- Operational readiness reviews
- Support assumptions

Dimension 10: Accountability

Immature Pattern The learner views responsibility as assigned work completion.

Mature Pattern The learner owns decisions, evidence, limitations, risks, and outcomes.

Evidence Possible evidence includes:

- Decision explanations
- Risk communication
- Review responses
- Reflection artifacts
- Release defense
- Team responsibility records

Relationship to Assessment

The Engineering Maturity Model should inform assessment.

Assessment should evaluate whether learners are becoming more mature engineers, not merely whether they completed assigned artifacts.

A learner who produces many files but cannot explain their decisions has not demonstrated strong maturity.

A learner who uses AI extensively but cannot verify the result has not demonstrated strong maturity.

A learner who completes a demo but cannot explain release risks has not demonstrated strong maturity.

Assessment should reward visible growth in engineering responsibility.

Relationship to Instructor Analysis

Instructor analysis tools may help evaluate aspects of engineering maturity.

For example, repository analysis scripts may inspect:

- Repository structure
- Commit activity
- Pull request usage
- Documentation completeness
- Test evidence
- Traceability
- AI-use records
- Release readiness artifacts

However, automated analysis should not replace human judgment.

Analysis tools can surface evidence.

Instructors and reviewers must interpret that evidence.

The maturity model defines what the evidence means.

Relationship to Student Reflection

Students should use this model to understand their own growth.

Reflection prompts may include:

- Where did I move beyond task completion?
- What evidence did I create that supports engineering claims?
- How did I use review to improve the work?
- How did I verify AI-assisted contributions?
- What operational risks did I identify?
- What would I improve if this system continued beyond the course?

Reflection should connect student experience to maturity growth.

Relationship to COMP330

In COMP330, most students will begin between Level 1 and Level 2.

The course should help students move toward Level 3 and introduce Level 4 behaviors.

Cycle 1 primarily moves students from task completion toward structured participation.

Cycle 2 primarily moves students from structured participation toward evidence-based engineering and release defense.

Some students may demonstrate elements of Level 4.

Level 5 is introduced as a professional horizon rather than expected full mastery.

Use Guidance

This model should be used to guide:

- Assignment design
- Rubric development
- Repository review
- Instructor feedback

- Student reflection
- Project checkpoints
- Release readiness reviews
- Professional portfolio development

The model should not be used mechanically.

It should support judgment.

Core Thesis

Engineering maturity is visible in behavior.

Students mature when they move from task completion toward evidence, review, judgment, operational thinking, and stewardship.

The ETIS Engineering Maturity Model exists to make that growth visible, teachable, assessable, and improvable.

COMP 330 Two-Cycle Engineering Project Model

Software Engineering in the AI Era

Loyola University Chicago

An ETIS Educational Ecosystem Flagship Implementation

Provenance

This document originated from Loyola University Chicago COMP 330 — Software Engineering.

It serves as the flagship implementation of the ETIS Two-Cycle Engineering Model.

While many concepts may eventually become reusable educational assets, this document intentionally preserves COMP330 provenance because it reflects a specific educational implementation.

Future ETIS educational implementations may adapt the model while preserving the underlying principles.

Core Idea

The COMP 330 project is not two coding deadlines.

Cycle 1 proves that the team can build a controlled vertical slice with evidence.

Cycle 2 proves that the team can learn from Cycle 1, improve the system, and make a defensible final release judgment.

1. Purpose of the Two-Cycle Project Model

The COMP 330 project is organized as two development cycles because professional software engineering is not a one-shot coding exercise.

Teams must learn to define the problem, make commitments, build a controlled first release, review evidence, learn from defects and feedback, and then mature the system.

The two-cycle model intentionally separates first-release capability from engineering maturity.

A working Cycle 1 demo matters, but it is not the end of the project. The stronger professional question is what the team learned from Cycle 1 and how that evidence changed Cycle 2 decisions.

Cycle 1: Controlled Vertical Slice Cycle 1 asks:

Can it work?

The team demonstrates that it can organize, plan, architect, construct, review, test, and release a small working system with evidence.

Cycle 2: Evidence-Based Maturity Cycle 2 asks:

Can it survive?

The team uses postmortem evidence, defects, risks, estimates, testing gaps, review feedback, and presentation feedback to improve quality, security, governance, observability, and release readiness.

2. Team Size, Team Formation, and Team Engineering Expectations

Teams will be formed by Week 2.

The expected team size is usually four to five students.

A three-person team can work, but it is fragile.

A six-person team is possible, but it should be treated as the upper limit because coordination overhead and hidden work increase quickly.

Software engineering is a team discipline.

Everyone on the team is a developer and engineering contributor first. Every student is expected to contribute to implementation, review, testing, documentation, and repository evidence.

Additional operational roles exist to make ownership visible, not to excuse anyone from technical contribution.

Every team member must own visible repository work.

Repository-visible workflow evidence should include issues, branches, pull requests, reviews, commits, and traceability links as the project matures.

Every team member must participate in review and evidence creation.

Every team member must understand the system well enough to explain the team's main engineering claims.

Every role must have a primary owner and a backup so the team does not fail when one person is unavailable.

> **Professional warning**
> Uneven contribution is common in student teams and in industry teams. COMP 330 does not assume every team works perfectly. It requires visible ownership, weekly status checks, GitHub evidence, pull requests, reviews, peer feedback, and role accountability so contribution problems become visible early instead of appearing at the deadline.

3. Professional Responsibility Roles

The following five operational roles should be assigned to students.

Smaller teams may combine roles.

Larger teams may split responsibilities if approved by the instructor.

The role model is intentionally lightweight. It creates accountability without turning the project into bureaucracy.

Role	Primary Responsibility	Typical GitHub Evidence Owned
Team Lead	Meeting cadence, team coordination, decision visibility, milestone readiness, team accountability	`/docs/team/team-charter.md`, `/docs/team/working-agreements.md`, meeting notes, milestone checklists
Planning & Process Lead	Scope, task plan, estimates, risks, schedule, issue hygiene, progress tracking	`/docs/planning/`, GitHub Issues, `/docs/planning/risk-register.md`, `/docs/planning/traceability.md`
Architecture & Development Lead	Architecture coherence, repository structure, implementation coordination, technical decision records	`/src`, `/docs/architecture/`, `/docs/decisions/`, `README.md`

Role	Primary Responsibility	Typical GitHub Evidence Owned
Quality & Review Lead	Testing, pull request discipline, review evidence, CI/CD checks, defect tracking, validation evidence	`/tests`, `/docs/testing/`, `/docs/reviews/`, `/.github/workflows/`, `/docs/quality/`
Operations & Evidence Lead	Release evidence, observability, known limitations, AI-use evidence, final evidence index, presentation evidence readiness	`/docs/release/`, `/docs/observability/`, `/docs/ai/`, `/docs/security/`, security-governance checklist

Role ownership means responsibility for correctness, completeness, and readiness of the evidence at the proper point in the cycle.

It does not mean the owner is the only person allowed to update the artifact.

Any team member may update any evidence artifact when doing legitimate project work.

The owner is responsible for making sure the artifact is accurate, current, reviewable, and linked to the rest of the project evidence when the milestone arrives.

Ownership Rules

Ownership Level	Responsibility
Primary owner	Accountable for keeping the artifact correct and ready
Backup owner	Able to step in if the primary owner is unavailable
All team members	Allowed and expected to update evidence when their work affects it
Team	Responsible for reviewing important evidence before submission or presentation

4. Minimum Weekly Cadence Meetings

Each team must run a minimum weekly cadence meeting.

This can be on Zoom or in person, on any day and time the team chooses.

The purpose is not to hold a long engineering design session.

The purpose is to keep status, ownership, blockers, commitments, and evidence visible.

When done correctly, the weekly cadence meeting should usually take 15 to 30 minutes, with a target closer to 15 minutes.

Longer design discussions, debugging sessions, architecture debates, or pair-programming work should be scheduled separately.

Recommended Cadence Agenda Each weekly cadence meeting should answer:

1. What did each person complete since the last checkpoint?
2. What is each person doing next?

3. What is blocked, unclear, risky, or behind schedule?
4. What GitHub evidence changed?
5. What agreement or decision did the team make?
6. What repository evidence, issue updates, pull requests, or documentation changes are required before the next checkpoint?
7. What must be ready before the next class, milestone, or submission?

GitHub evidence may include:

- Issues
- Branches
- Pull requests
- Tests
- Documentation
- AI-use log entries
- Architecture decisions
- Release evidence

Cadence rule
A cadence meeting is a status and accountability checkpoint. It is not the place to design the entire architecture, debug a hard failure, or debate every implementation option. Those are real engineering meetings, but they should be scheduled separately.

5. Cycle 1 Expectations: Controlled Vertical Slice

Cycle 1 is intentionally longer than the raw implementation size of the default project would require in industry.

A professional team assigned a small version of this project might implement much of the basic functionality quickly.

COMP 330 slows the work down on purpose because students are learning the full engineering discipline around the work, not just the code.

Cycle 1 is the crawl phase.

The team is learning how to form, define scope, create evidence, make estimates, document architecture, use GitHub professionally, review AI-assisted work, test the system, and present a release.

Weekly progress is still required.

Teams that wait until the deadline will struggle because late work leaves no time for review, testing, integration, defect correction, or evidence cleanup.

Cycle 1 Minimum Expectations By the end of Cycle 1, the team should have:

- Formed the team and assigned roles, with backups identified
- Created and used the GitHub repository as the authoritative engineering record
- Established the repository `README.md` as the professional front door to the project and major engineering evidence
- Defined a Cycle 1 vertical slice and kept it intentionally small
- Documented requirements, acceptance criteria, assumptions, risks, and out-of-scope decisions
- Maintained task plan, estimates, schedule, traceability, and team commitments
- Made architecture, interfaces, data/context ownership, and governance boundaries reviewable
- Completed implementation through issues, branches, pull requests, reviews, and tests
- Disclosed, reviewed, verified, and owned AI use
- Presented the Cycle 1 release with evidence, known limitations, risks, and lessons learned

Cycle 1 Professional Standard Cycle 1 does not prove that the system is mature.

Cycle 1 proves that the team can create a controlled, reviewable, evidence-backed first release.

6. Cycle 2 Expectations: Evidence-Based Maturity

Cycle 2 is shorter and should not become a feature binge.

It is not a restart.

It is also not a promise to complete every improvement idea listed in the project overview.

Cycle 2 work must be selected and scoped based on evidence from Cycle 1.

The Cycle 1 postmortem is the decision point.

Teams should use defects, missed estimates, weak tests, review feedback, architecture pain, AI-use issues, presentation questions, and known limitations to choose the highest-value maturity work.

The goal is to improve the credibility of the system, not simply add sparkle.

Cycle 2 Minimum Expectations During Cycle 2, the team should:

- Review Cycle 1 evidence before choosing Cycle 2 work
- Identify root causes, not just symptoms
- Re-estimate remaining and new work based on actual Cycle 1 performance
- Select a small number of maturity targets the team can complete and prove
- Improve tests, defects, architecture, observability, runtime visibility, security/governance, AI-use controls, or release documentation as appropriate
- Document what was intentionally deferred and why
- Prepare a final release argument supported by repository evidence
- Identify the repository state associated with major releases using release tags, versions, or equivalent release markers when appropriate

Cycle 2 Professional Standard Cycle 2 proves whether the team can learn.

A mature Cycle 2 release should show improved engineering judgment, stronger evidence, clearer release claims, and more realistic awareness of operational risk.

7. Evidence Ownership by Role and Cycle

The table below connects operational roles to the main evidence they are expected to keep healthy.

These are ownership lanes, not walls.

Teammates may and should help each other, but the owner is accountable for readiness at each milestone.

Engineering evidence should be reviewable without requiring verbal clarification from the original team.

Role	Evidence Owned	Why It Matters
Team Lead	Team charter, working agreements, meeting notes, milestone readiness	Keeps team decisions, communication rhythm, and submission readiness visible
Planning & Process Lead	Scope, task plan, estimates, risk register, schedule, traceability	Keeps work bounded, sequenced, owned, and connected to requirements

Role	Evidence Owned	Why It Matters
Architecture & Development Lead	Architecture package, decision records, repository structure, implementation coordination	Keeps the design understandable, changeable, and aligned with implementation
Quality & Review Lead	Tests, defect log, pull request review evidence, CI/CD evidence, validation evidence	Keeps quality claims backed by inspection, tests, reviews, and defect discipline
Operations & Evidence Lead	AI-use log, release notes, known limitations, observability evidence, security/governance evidence, final evidence index	Keeps release claims, AI accountability, operational maturity, and final presentation evidence defensible

8. How This Compares to Industry Practice

Industry teams often operate with short status rituals, explicit ownership, issue tracking, pull requests, build checks, release notes, and postmortems.

COMP 330 uses a classroom version of that pattern.

The goal is not to imitate every industry tool or ceremony.

The goal is to practice the engineering habits that make team software controllable.

Professional engineering maturity includes operational stewardship, accountable review behavior, honest risk communication, and evidence that another engineer can independently inspect.

In professional teams, a 15-minute status check is not where all engineering work happens.

It is where the team synchronizes.

The real work happens in issues, design notes, pull requests, tests, reviews, documentation updates, and focused working sessions.

COMP 330 expects the same distinction.

Professional teams also experience uneven contribution, missed estimates, unclear ownership, and communication failures.

The professional response is not denial.

The professional response is visibility: identify the problem, adjust responsibilities, document risks, and make sure the evidence reflects reality.

9. What Is Not Acceptable

The following behaviors are not acceptable in COMP 330 project work:

- Waiting until the deadline to create GitHub evidence
- Treating the project as separate individual parts that are never integrated
- Using AI-generated work that the team cannot explain or verify
- Skipping reviews because the code appears to work
- Holding vague meetings with no decisions, owners, or next actions
- Submitting a polished document that does not point to repository evidence
- Presenting a demo while hiding known limitations, defects, or risks
- Treating the final presentation as proof instead of repository-supported engineering evidence
- Allowing one or two students to carry the project while others remain invisible

10. What Teams Should Do Immediately

Teams should do the following as soon as project work begins:

1. Confirm team membership by Week 2.
2. Select a team name and primary communication channel.
3. Assign the five professional responsibility roles and identify backups.
4. Choose the default project or propose an instructor-approved alternative.
5. Schedule the weekly cadence meeting.
6. Create the GitHub repository and minimum folder structure.
7. Draft the team charter, role matrix, working agreements, and AI-use policy.
8. Define the Cycle 1 vertical slice and the first small set of issues.
9. Identify which evidence artifacts each role owner must keep current before the first submission.

Relationship to the ETIS Learning Models

This document operationalizes the Two-Cycle Engineering Model inside the Loyola COMP330 flagship implementation.

It connects directly to the broader ETIS learning model architecture.

Professional Transformation Model The two-cycle model helps students move from:

Student

↓

Responsible Engineer

↓

Reviewer

↓

Release Defender

Cycle 1 introduces responsible engineering behavior.

Cycle 2 strengthens reviewability, release defense, and operational awareness.

Engineering Maturity Model Cycle 1 primarily moves students from task completion toward structured participation.

Cycle 2 moves students toward evidence-based engineering and reviewable engineering judgment.

Software Engineering Learning Progression The two-cycle model gives the learning progression a project rhythm.

Cycle 1 emphasizes orientation, intent formation, engineering structure, and initial construction.

Cycle 2 emphasizes improved construction, verification, release defense, operational awareness, and professional reflection.

Final Takeaway

The project succeeds when the team can show both a useful system and the engineering evidence behind it.

A professional team does not ask others to trust a demo.

It shows the requirements, design, review, tests, risks, AI-use decisions, release notes, and maturity evidence that make the demo credible.