COMP 3550

10.4 — PERSONAL REFLECTION: GROWTH AS AN ENGINEER

Week 10: Measuring Team and Project Successes

YOUR JOURNEY THROUGH COMP 3350

Core Themes We Experienced

- Agile in Action — from theory to running sprints and adapting to change
- Testing for Confidence — unit, integration, characterization tests
- Teamwork & Collaboration — version control, code reviews, collective ownership
- Architecture & Design — patterns, refactoring, and working with legacy systems

YOUR JOURNEY THROUGH COMP 3350

Core Themes We Experienced

- Agile in Action — from theory to running sprints and adapting to change
- Testing for Confidence — unit, integration, characterization tests
- Teamwork & Collaboration — version control, code reviews, collective ownership
- Architecture & Design — patterns, refactoring, and working with legacy systems

Course Vision from Week 1 (revisited)

By the end of this course, you should be able to:

- Build software that solves a real problem as part of a team
- Apply iterative, test-driven, and design-focused practices
- Work effectively in a shared codebase with evolving requirements
- Communicate and collaborate like a professional engineer

THE POWER OF FEEDBACK LOOPS

Feedback Matters

- Surfaces issues early, when they’re easier and cheaper to fix
- Builds shared understanding of code, goals, and priorities
- Encourages continuous improvement — small tweaks over big reworks

THE POWER OF FEEDBACK LOOPS

Feedback Matters

- Surfaces issues early, when they’re easier and cheaper to fix
- Builds shared understanding of code, goals, and priorities
- Encourages continuous improvement — small tweaks over big reworks

Where We See Feedback Loops

- Peer Reviews / Pull Requests
  - Catch bugs, improve design, share knowledge
  - Discuss why a change was made, not just what
- Retrospectives
  - Reflect on what went well, what to improve, and action items
  - Builds team trust and resilience
- PR / MR Comments
  - Inline suggestions, clarifications, and examples
  - Great for mentoring and onboarding newer teammates

IMPOSTER SYNDROME IS NORMAL

- Thinking “I don’t belong here” or “Everyone else knows more than me”
- Comparing your inside (doubts, struggles) to others’ outside (finished work)
- Feeling like your success is luck, not skill

IMPOSTER SYNDROME IS NORMAL

- Thinking “I don’t belong here” or “Everyone else knows more than me”
- Comparing your inside (doubts, struggles) to others’ outside (finished work)
- Feeling like your success is luck, not skill

- Everyone feels behind at some point — even experienced engineers
- Struggling doesn’t mean you’re failing — it means you’re learning
- Growth happens outside your comfort zone

IMPOSTER SYNDROME IS NORMAL

- Thinking “I don’t belong here” or “Everyone else knows more than me”
- Comparing your inside (doubts, struggles) to others’ outside (finished work)
- Feeling like your success is luck, not skill

- Everyone feels behind at some point — even experienced engineers
- Struggling doesn’t mean you’re failing — it means you’re learning
- Growth happens outside your comfort zone

- I can’t promise it goes away but...
- Acknowledge it — naming the feeling helps reduce its power
- Focus on learning — measure **progress by what you’ve learned, not perfection**
- Share with peers — you’ll often discover they feel the same
- Keep a “wins” list — small successes add up over time

CONTINUING THE JOURNEY

- **Keep Growing After COMP 3350**
  - Join Open-Source Projects
  - Contribute to codebases you care about
  - Learn real-world collaboration skills from diverse teams
- **Revisit Old Projects**
  - Refactor with new skills
  - Add tests, improve design, document better
- **Build Small Tools**
  - Automate a daily task
  - Experiment with new frameworks or APIs without big commitments
- **Read Other People's Code**
  - See different problem-solving styles
  - Learn patterns, idioms, and shortcuts you might not discover alone
- **Join me (or whoever) in COMP 4350 in the future!**

FINAL REFLECTION

- **What are you proudest of this term?**
  - A feature you built?
  - A problem you solved?
  - A way you supported your team?
- **What's one way you now think differently about software?**
  - A new process or tool you value?
  - A change in how you approach design, testing, or teamwork?