Introduction

17-313: Foundations of Software Engineering

https://cmu-313.github.io

Michael Hilton and Josh Sunshine

Spring 2026

Introductions

Michael Hilton

Full Teaching Professor at CMU

A.S. Grossmont Community College 1999

B.S. San Diego State University - 2002

Software Engineer at DoD - 2002 to 2011

M.S. Cal Poly San Luis Obispo - 2013

PhD at Oregon State - 2017

Internship at Microsoft Research - Summer 2017

Assistant/Associate/Full Teaching Professor at CMU - Fall 2017 to current

Josh Sunshine

Associate Professor

B.A. Brandeis University, 2004.

Ph.D., Carnegie Mellon University, 2013

Faculty @ CMU since 2014

Teaching Assistants

Autumn Qiu

Juan Ageitos

Bing Bhakdibhumi

Alejandro Estrada

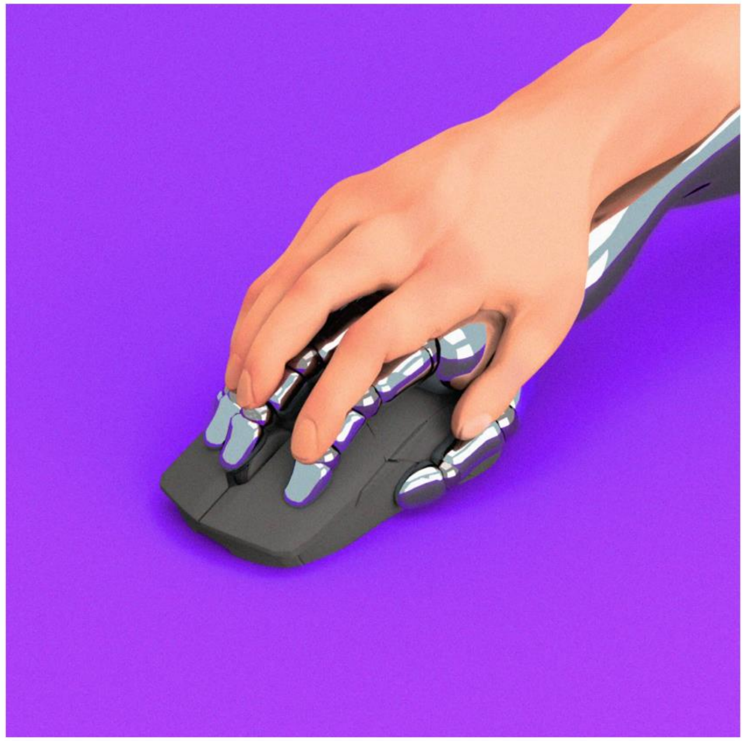
Kaia Newman

Hwei-Shin Harriman

Lisa Huang

You're a Computer Science Major. Don't Panic.

Nov. 12, 2025


Erik Carter

By Mary Shaw and Michael Hilton
Dr. Shaw and Dr. Hilton teach software engineering at Carnegie Mellon University.

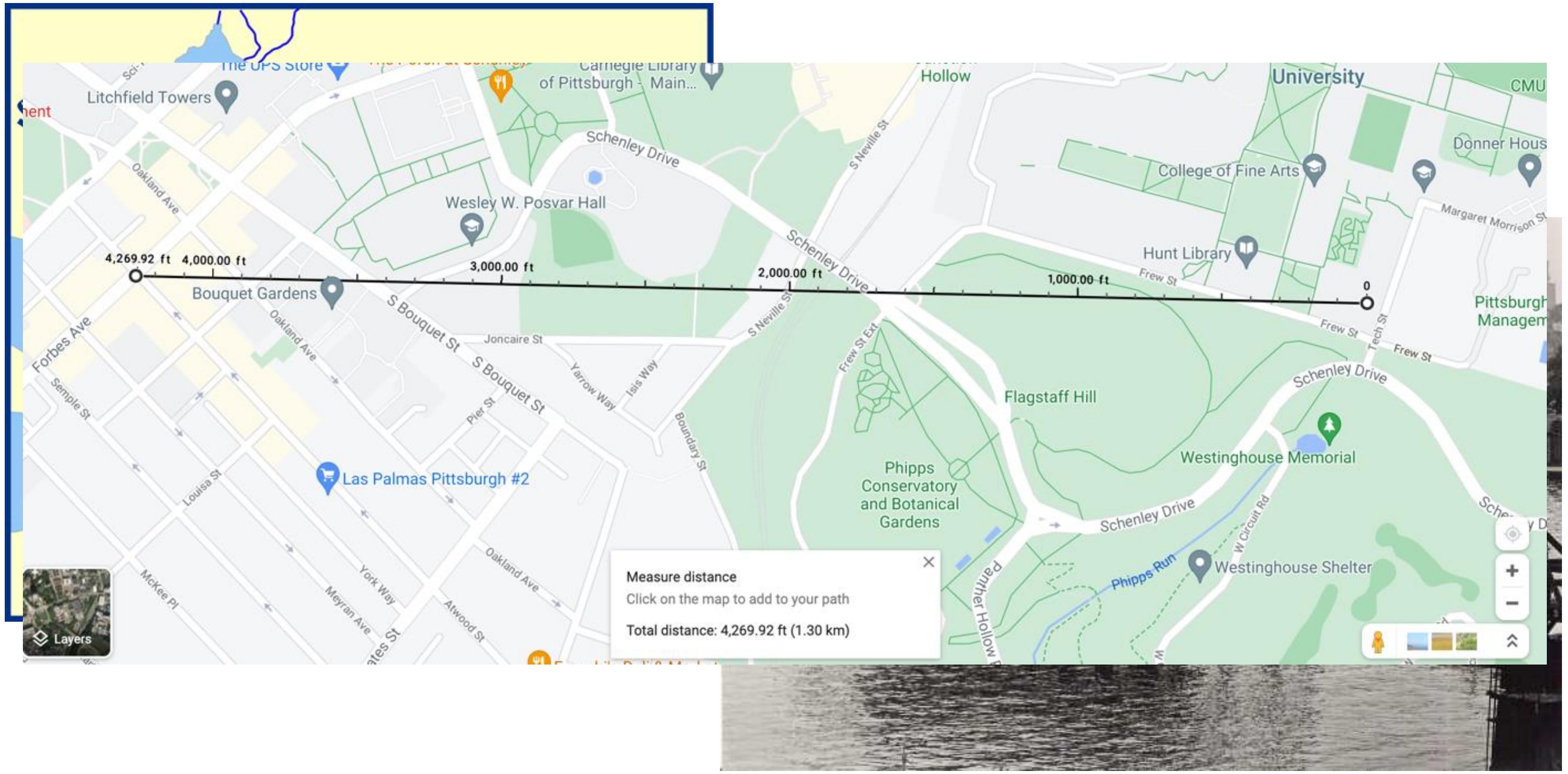
For decades, computer science students have been taught a central skill: using computers to solve problems. In practice, that has meant programming or writing code to tell a machine how to perform tasks, like sorting a list of numbers or finding the most effective way to deploy snowplows during a winter storm.

Now, generative artificial intelligence tools like ChatGPT and Claude can write those programs themselves, producing code in much the same way they write essays and legal briefs: by analyzing a vast number of similar texts and assembling new ones that look similar. A student can ask ChatGPT to write a program that sorts a list of numbers and get a working answer in seconds.

This new capability is transforming how computer scientists get work done. Microsoft's chief executive, Satya Nadella, has said that up to 30 percent of the company's code is being written by A.I. at this point. Even nonprogrammers suddenly have the ability to create their own software tools.

Vasa

Vasa



Measure distance

Click on the map to add to your path

Total distance: 4,269.92 ft (1.30 km)

What happened is now called “Vasa syndrome”

- Changing shipbuilding orders
- No specifications for modified keel
- Shifting armaments requirements

Requirements

- Shipwright’s death

Teams

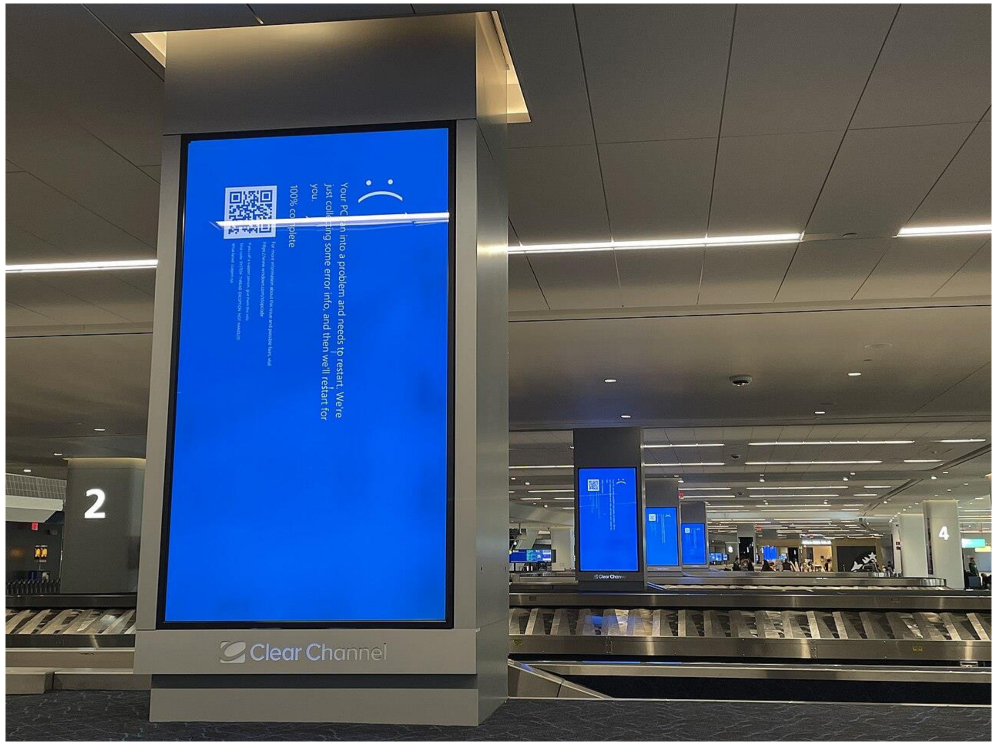
- No way to calculate stability, stiffness, or sailing characteristics

Metrics

- Failed pre-launch stability tests

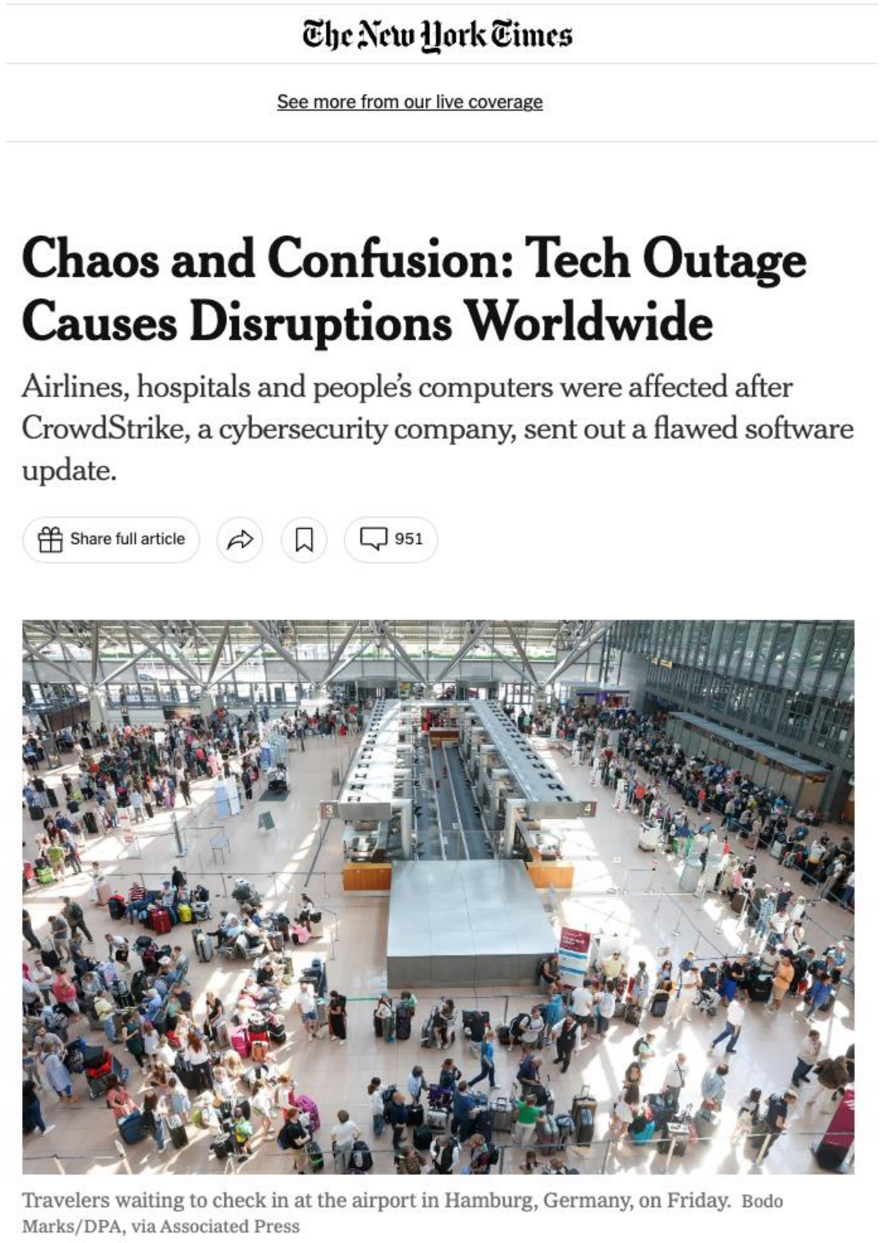
QA

The CrowdStrike Incident (2024)



LaGuardia Airport, New York City

Photo by Smishra1 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=150535443



The New York Times

See more from our live coverage

Chaos and Confusion: Tech Outage Causes Disruptions Worldwide

Airlines, hospitals and people's computers were affected after CrowdStrike, a cybersecurity company, sent out a flawed software update.

Travelers waiting to check in at the airport in Hamburg, Germany, on Friday. Bodo Marks/DPA, via Associated Press

Complex software engineering issues involved

• Software running with kernel-level privilege can crash the system (BSOD) and prevent booting. • Content updates more continuous than software updates	**Requirements (Architecture & Design)**
• Insufficient testing of content *and* parser	**QA**
• No staged roll-outs. • No way to fix issue remotely after impact.	**DevOps**
• Skewed incentives	**Metrics**
• Limited liability	**Licenses**

Software Engineering?

- What is engineering?
- And how is it different from hacking/programming?

1968 NATO Conference on Software Engineering

- Provocative Title
- Call for Action
- “Software crisis”

Margaret Hamilton

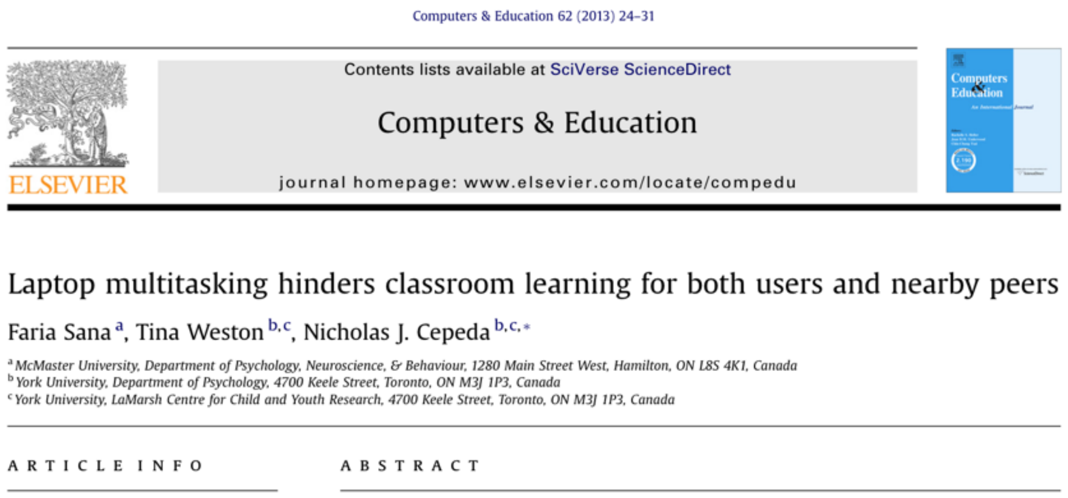
Course infrastructure and logistics

Recent FCE:

I might have found this course exceedingly annoying. But I can't deny the outcomes. I do feel that I've learned a lot in this class. About working in teams, with software that are hard to work with, about software development practices, technical details like Github workflows, open source projects, AI integration. When it's all said and done, I am grateful for the class and what I've taken away from it.

Smoking Section

- Last full row


Computers & Education 62 (2013) 24–31

Contents lists available at SciVerse ScienceDirect

Computers & Education

journal homepage: www.elsevier.com/locate/compedu

Laptop multitasking hinders classroom learning for both users and nearby peers

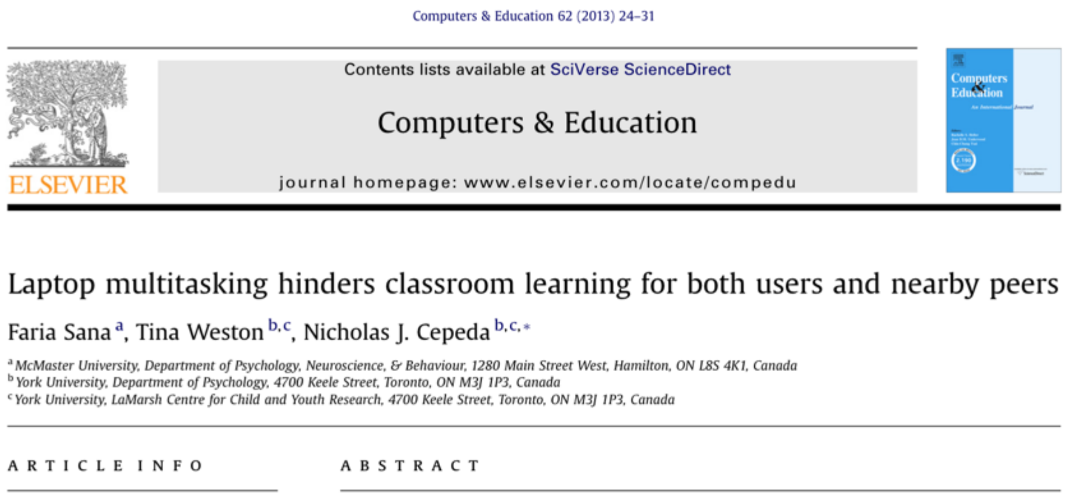
Faria Sana[a], Tina Weston[b,c], Nicholas J. Cepeda[b,c,*]

[a] McMaster University, Department of Psychology, Neuroscience, & Behaviour, 1280 Main Street West, Hamilton, ON L8S 4K1, Canada
[b] York University, Department of Psychology, 4700 Keele Street, Toronto, ON M3J 1P3, Canada
[c] York University, LaMarsh Centre for Child and Youth Research, 4700 Keele Street, Toronto, ON M3J 1P3, Canada

ARTICLE INFO

ABSTRACT

"…participants who multitasked on a laptop during a lecture scored lower on a test compared to those who did not multitask, and participants who were in direct view of a multitasking peer scored lower on a test compared to those who were not. The results demonstrate that multitasking on a laptop poses a significant distraction to both users and fellow students and can be detrimental to comprehension of lecture content."


Computers & Education 62 (2013) 24–31

Contents lists available at SciVerse ScienceDirect

Computers & Education

journal homepage: www.elsevier.com/locate/compedu

Laptop multitasking hinders classroom learning for both users and nearby peers

Faria Sana[a], Tina Weston[b,c], Nicholas J. Cepeda[b,c,*]

[a] McMaster University, Department of Psychology, Neuroscience, & Behaviour, 1280 Main Street West, Hamilton, ON L8S 4K1, Canada
[b] York University, Department of Psychology, 4700 Keele Street, Toronto, ON M3J 1P3, Canada
[c] York University, LaMarsh Centre for Child and Youth Research, 4700 Keele Street, Toronto, ON M3J 1P3, Canada

ARTICLE INFO

ABSTRACT

"…participants who multitasked on a laptop during a lecture scored lower on a test compared to those who did not multitask, and participants who were in direct view of a multitasking peer scored lower on a test compared to those who were not. The results demonstrate that multitasking on a laptop poses a significant distraction to both users and fellow students and can be detrimental to comprehension of lecture content."

Course infrastructure and logistics

- Infrastructure/source of truth
  - Course website: schedule, slides, syllabus, office hours
  - Canvas (and Gradescope) homework, grades, other material
    Slack for communication and collaboration.
    Git/GitHub for coding and collaboration
- Logistics
  Lecture in-person only
  All recitations are in-person (Except yesterday)
- Office Hours are flexible.
  - Office Hours are on website. Please come by!!
- If you want to talk to us, DM/email BOTH INSTRUCTORS at once.
  - TRUST us, it's faster. If you DM us individually, we might both assume the other instructor will reply to you first.

Connect with us for the class

- All links on our course website: https://cmu-313.github.io
- Canvas: https://canvas.cmu.edu/courses/51340
- Gradescope: https://www.gradescope.com/courses/1213157
- Slack: Check your emails (will send again if you just enroll)
    - Slack vs Piazza

Course Themes

- Software engineering as a human process
- Requirements and Specifications
- Metrics and Measurement
- Software Quality: Testing + CI + Security
- Continuous Deployment and DevOps
- Software Project Teams
- Managing Time, Teams, and Risks
- Software Architecture and Design Docs
- Scaling and Performance, Trade-offs
- AI/ML in Software Engineering
- Open-Source Software
- Strategic Thinking about Software

Prerequisites

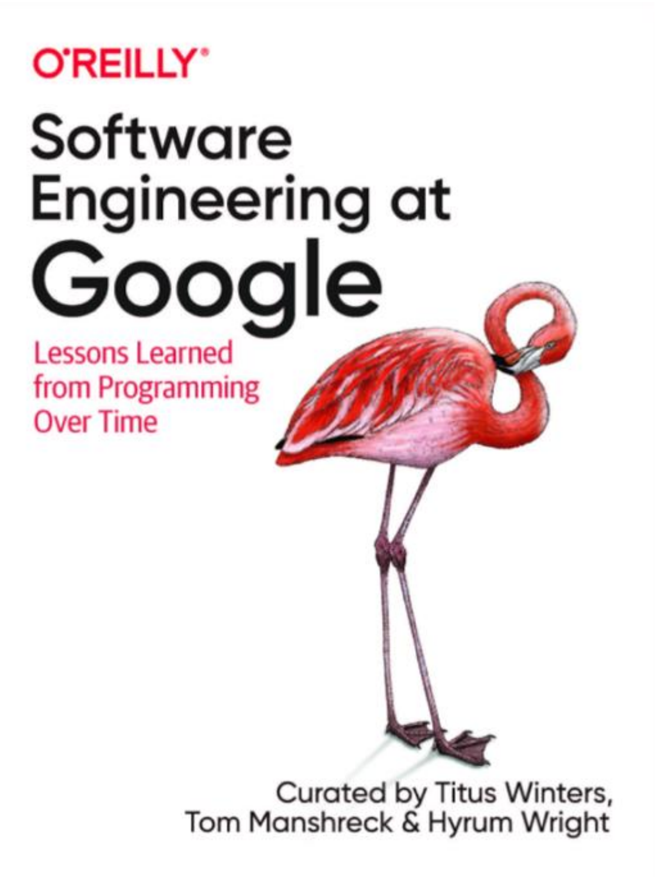
- Assumes working knowledge of popular programming languages (e.g., 15-121, 15-122)
- You will have the best experience if you have already had an internship or been involved in a large-ish software development project (ask us if you have any questions)
- How is it different from 17-214?
  - 17-313 largely focused on human issues and quality beyond functional correctness
  - 17-313 focused on larger scale

Readings, Quizzes, and Participation Activities

- Reading assignments for some lectures
  - Preparing in-class discussions: background material, case descriptions, possibly also podcast, video, Wikipedia
- In-person activities
  - **Lecture**: Active learning exercises in almost every lecture
    - Most will be done on physical paper and turned in at the end of class.
  - **Recitation**: Working sessions, submission on Canvas/Gradescope
- All of the above count as graded "participation activities"
  - You may miss up to 4 participation activities with no grade penalty (No need to send emails asking permission)

Textbook

- No single textbook
- Assigned readings from different sources
  - Book chapters (library)
  - News articles
  - Lecture notes
- Recommended supplementary reading: Software Engineering at Google
  - Available for free online (legally!): https://abseil.io/resources/swe-book


O'REILLY
Software Engineering at Google
Lessons Learned from Programming Over Time
Curated by Titus Winters, Tom Manshreck & Hyrum Wright

Gaining Experience: Central to 17-313!

- Case study analyses
- Team assignments
  - Teamwork exercises
- Open-source engagement

- **Hands-on experience is key!!!**
  - “Learn by Doing”

Evaluation

- Assignments (50 %)
  - Regular homework projects, mostly in teams with individual component
  - Open-source engagement
- Midterms (30 %)
- Participation activities (20 %)
  - In-class exercises
  - Pre-class reading assignments
  - Recitation exercises

“Homework” Assignments / Projects

- P1: Setup and test a large existing software product
  - Get up-to-speed with new technologies quickly and on your own
- P2: Collaborative development on a large software project
  - Add features and follow SE process
- P3: Continuous Integration + Deployment
- P4: Develop a design doc and integrate with an AI service.
- P5: Open-source Excursion
  - Open-ended project: contribute to an OSS project using everything you have learned; get kudos for having PRs merged

Recitations

- Practical tasks, preparation for homework, extra material, discussions
- Have your GitHub account at the ready.
  - Bring your laptop!
- This week only: Async/offline recitation for NodeBB (to prepare for P1 and P2). **Don't forget to finish by Friday.**
- From P2 onward: Project teams will all be in the same recitations… Good (forced) opportunity to meet in person

Warning! Course & HW structure may be different than what you are used to…

- Lecture topics are on high-level ideas about software engineering; case studies and experiences
- Projects require applying these ideas to technical artifacts
- Projects simulate "real-world" professional SE experience
- Technical aspects of project <u>will not be taught in class</u>
  - Explicit learning goal: learn new tools, languages, etc. on your own
  - Ask for help when needed; recitations provide demos and resources
- Project requirements are often <u>vague or under-specified</u> (intentionally)
  - Feel free to ask for clarifications, but expect subjective responses
  - Focus for assessment is engagement, not absolute correctness

Team Assignments

- Mirror realistic setting
- Assigned teams throughout the semester
  - Fill in team building survey before next lecture
- Teamwork surveys every week
- Conflict resolution process as needed
- Most team assignments have individual components

Professionalism

- Being a professional means, you must work well with others
- The best professionals are those who make those around them better
- If you feel someone is not treating you or someone else in a professional manner, you have two options:
  - If you feel you have the standing to do so, speak up!
  - Reach out to the course staff, and we will meet with you privately to discuss it, as well as preserve your anonymity

Final Projects

- Open-source excursion is the most fun part of the course!
- Very open-ended project. 24% of overall grade.
- Brings together everything you will have learned from lecture and prior assignments
- Teamwork and communication is very important
- In-person presentation in finals week (no exam)
- ***Do NOT book flight tickets for end-of-semester holidays until finals are scheduled.*** You will lose points for missing final presentations if you fly out early.

Late day policy

- **Assignments**: No late days
  - Simply doesn't work with team assignments
  - Plan for unexpected delays ahead of time (not just before deadline).
- **Participation activities (lecture + recitation)**: Accommodations in case of health issues, travel for interviews, university sports, etc.
  - Everyone gets **4 free absences**. No need to inform us beforehand.
  - Beyond 4 absences, participation grade can be affected.
  - Inform us of extended absences (e.g., hospitalization). We can help you make up some of the lost points in such cases, in conjunction with your advisor.
- If you have an assignment due after a trip, turn it in *before* you leave.
  - You may not have Internet where you're going.
  - Your return travel may be delayed beyond the assignment deadline!

Generative AI Policy

You may use generative AI technologies for assisting in code development. **We very strongly discourage** using AI to **generate** writing responses, but we encourage its use for editing your writing. In addition, **you are liable for factually inaccurate answers or unspecific rambling produced by AI tools**, and we will assess significant penalties to low quality (AI Slop) responses. It is your responsibility to validate and verify AI-produced content before submitting it for class purposes.

If you automate your thinking, you automate away the learning.

Academic Honesty

- In group work, be honest about contribution of group members; do not cover for others
- Let us know if you have concerns about teammate’s work.
- Attribute content to any Generative AI service you use.
- DO NOT submit participation sheets for people who are not in class

For next class: survey, scheduling

First-week Survey due Thursday

- Form groups based on schedule availability
  - **This is ridiculously important.**
  - Identify experience and working styles.
  - Participation point
- Help us shape the course based on
  - your background knowledge
  - your interests
- Posting to Canvas. We will also post on Slack after inviting you all
- https://forms.gle/sowXfM9PoDJxYELM8

Project P1 posted online

- P1A **due this Friday** (August 29th, 11:59pm)
  - Only 5% of total P1 points – meant to ensure you start on time
  - Only need to be able to install and run NodeBB locally

- P1B **due next Thursday** (September 4th, 11:59pm)
  - Refactor code in a single file to address code quality warnings.
  - Validate change via test coverage and manual testing
  - Expect to be **technically challenging** for non-experts; purpose is to learn new things and engage with a large code base without much hand-holding.