

CSE 403

Software Engineering

Course introduction

Today's outline

- The CSE 403 team
- What is software engineering
- Course overview and expectations
- Assignment 1: pitch a project

TAs



Medha Gupta



Tijana Minic



Lucas Varotto



Yuki Yang

Your TA has multiple roles:

- Venture capitalist who expects results
- Manager who helps when you have difficulty
- Grader

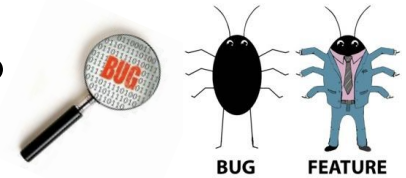
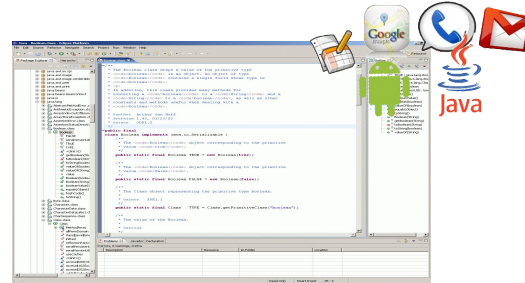
What is Software Engineering?



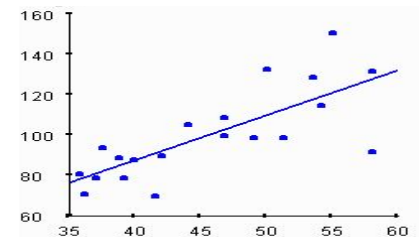
Consider what a software engineer must do to
deliver a product/service
that **delights the customer** on an ongoing basis

What is Software Engineering?

- Developing in an IDE and software ecosystem?
- Debugging and maintaining a software system?
- Deploying and running a software system?
- Empirically evaluating a software system?
- Writing (design) docs?

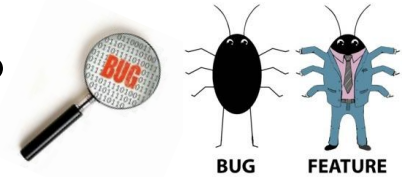
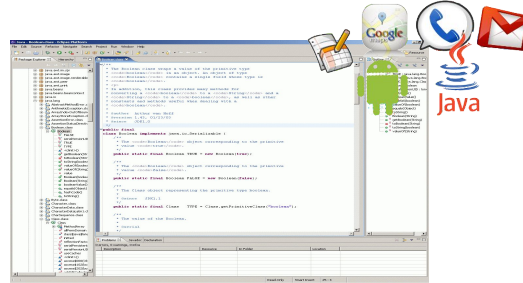


```
Closure-0 --- just@qatar:/tmp/Closure-0 --- bash -- 117x47
just@qatar:/tmp/Closure-0$ tail -30 ~/projects/defects4j/exit.sh
44 wget -nv $EVOSUITE_URL/$EVOSUITE_RT_JAR
# Set symlink for the supported version of Prostate
ln -sf $DIR_LIB_GEN/$EVOSUITE_RT_JAR $DIR_LIB_GEN/evosuite-current.jar
ln -sf $DIR_RT/$EVOSUITE_RT_JAR $DIR_LIB_RT/evosuite-rt.jar
#
# Download Randoom
echo "Setting up Randoom ..."
RANDOOM_VERSION="2.1.0"
RANDOOM_URL="https://github.com/randoom/randoom/releases/download/v${RANDOOM_VERSION}"
RANDOOM_JAR="randoom-${RANDOOM_VERSION}.jar"
od $DIR_LIB_GEN 44 { 1 -f $RANDOOM_JAR }
44 wget -nv $RANDOOM_URL/$RANDOOM_JAR
# Set symlink for the supported version of Randoom
ln -sf $DIR_LIB_GEN/$RANDOOM_JAR $DIR_LIB_GEN/randoom-current.jar
echo "Defects4J successfully initialized."
just@qatar:/tmp/Closure-0$ defects4j test -v
Running ant (compile.tests)..... OK
Running ant (run.dev.tests)..... OK
Failing tests: 0
just@qatar:/tmp/Closure-0$
```

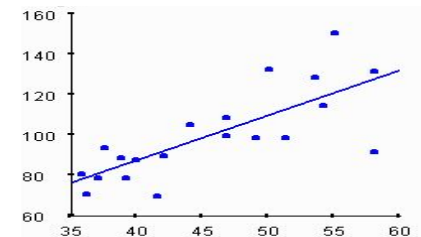
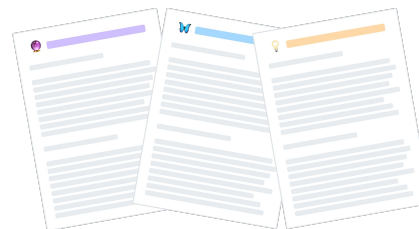


What is Software Engineering?

- Developing in an IDE and software ecosystem?
- Debugging and maintaining a software system?
- Deploying and running a software system?
- Empirically evaluating a software system?
- Writing (design) docs?



```
Closure-0 --- rust@gator:/tmp/Closure-0 --- bash -- 117x47
rust@gator:/tmp/Closure-0$ tail -30 ~/projects/defects4j/exit.sh
44 wget -nv $EVOSUITE_URL/$EVOSUITE_RT_JAR
# Set symlink for the supported version of Probita
ln -sf $DIR_LIB_GEN/$EVOSUITE_RT_JAR $DIR_LIB_GEN/evosuite-current.jar
ln -sf $DIR_RT/$EVOSUITE_RT_JAR $DIR_LIB_RT/evosuite-rt.jar
#
# Download Randoop
#
echo "Setting up Randoop ..."
RANDOOP_VERSION="2.1.0"
RANDOOP_URL="https://github.com/randop/randop/releases/download/v${RANDOOP_VERSION}"
RANDOOP_JAR="randoop-${RANDOOP_VERSION}.jar"
od $DIR_LIB_GEN 44 { | | $RANDOOP_JAR }
44 wget -nv $RANDOOP_URL/$RANDOOP_JAR
# Set symlink for the supported version of Randoop
ln -sf $DIR_LIB_GEN/$RANDOOP_JAR $DIR_LIB_GEN/randop-current.jar
#
echo "Defects4J successfully initialized."
rust@gator:/tmp/Closure-0$ defects4j test -v
Running ant (compile.tests)..... OK
Running ant (run.dev.tests)..... OK
Failing tests: 0
rust@gator:/tmp/Closure-0$
```



All of the above and much more!

Software Engineering is more than writing code

Software Engineering is the complete process of specifying,

requirements engineering, specifications, documentation

designing, software architecture and design, UI

developing, programming (just one of many important tasks)

analyzing,

testing, debugging, linting, verification, performance engineering

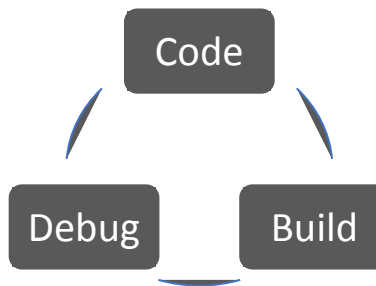
deploying,

DevOps, CI, packaging, operation, remote diagnostics,
documentation, websites

& maintaining refactoring, extensions, adaptation, issue tracking
a software system. nearly every system contains software

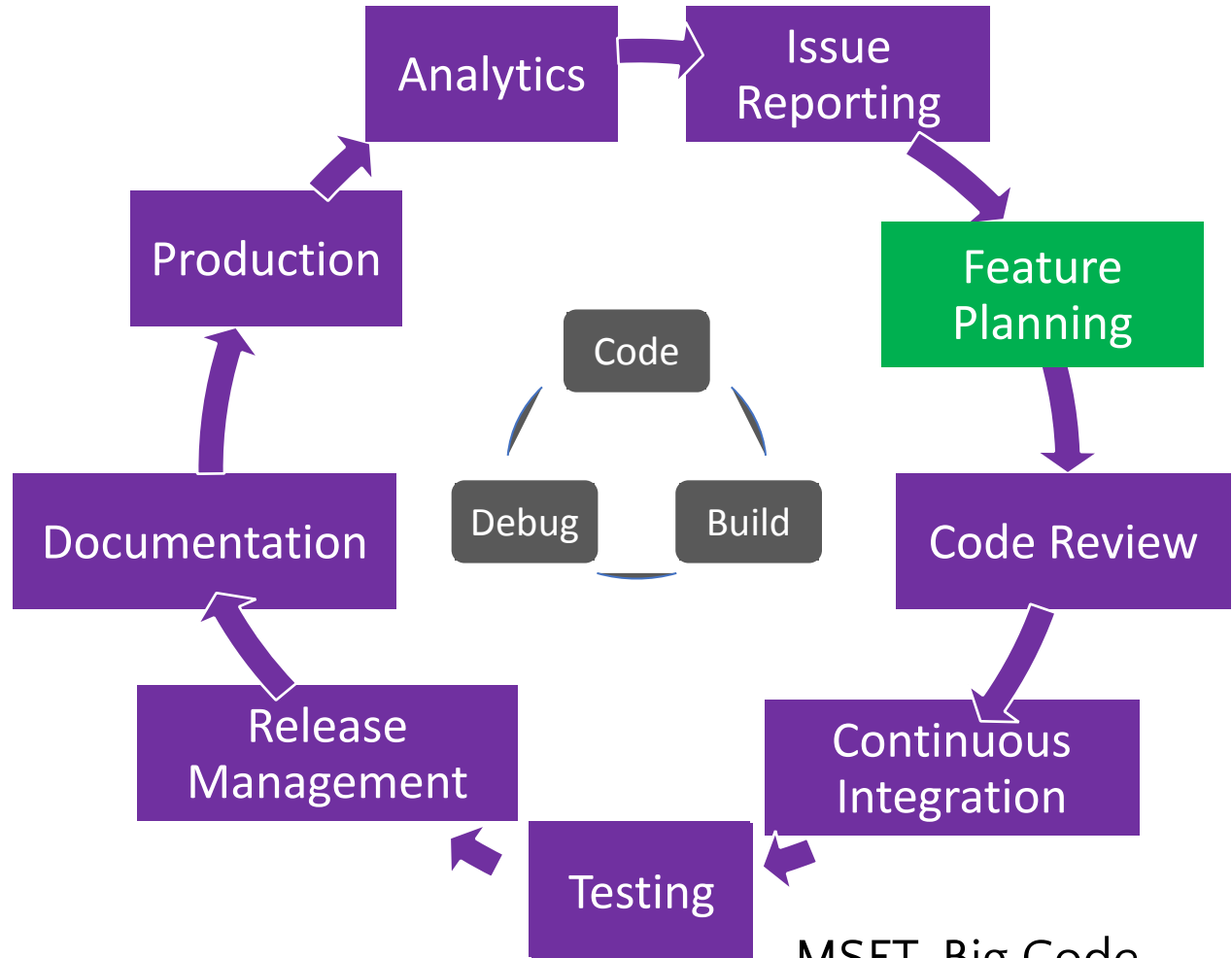
Software Engineering in practice

CSE intro-level
courses focus on the
inner loop of software
development



Software Engineering in practice

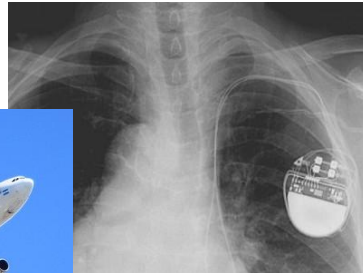
CSE 403
focuses on
the outer loop



MSFT, Big Code Summit, 2019

Why is Software Engineering important?

Software is everywhere!



Apple finally fixes 'gotofail' OS X

The 'Heartbleed' security flaw that affects most of the Internet

released a system software update known as the "gotofail"



Facebook Patches Access Token Leak

Users should change their passwords to mitigate threats posed by the accidental leak of perhaps millions of account identity details.



'The largest IT outage in history'

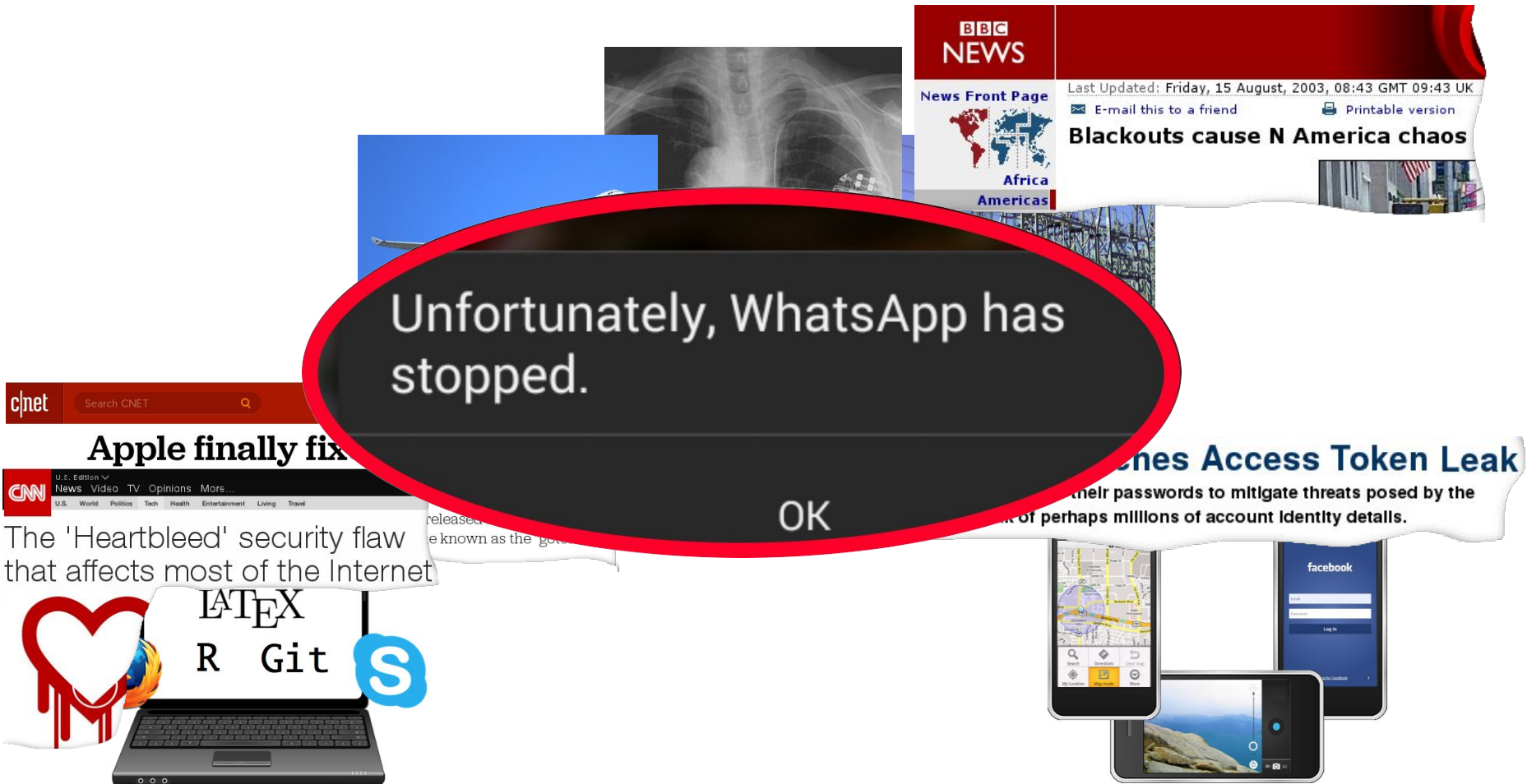
CROWDSTRIKE

Why is Software Engineering important?

Software is everywhere!

Unfortunately, WhatsApp has stopped.

OK



Summary: Software Engineering

What is Software Engineering?

- The **complete process** of creating a software system.

Why is it important?

It is a path to a successful product!

- Decomposes a complex engineering problem.
- Organizes processes and effort.
- Improves software reliability.
- Improves developer productivity.
- Delivers a solution that satisfies the customer.

Both **technical** and **management** contributions are essential.

What can you learn in CSE 403?

- Learn software development best practices
- Understand how software is produced – from conception to release and maintenance
- Develop skills to effectively collaborate with others
- Experience the responsibilities, issues and tradeoffs involved in making decisions as software engineers

Grounded by working as a team to incrementally deliver a real software product/service

CSE 403 vs. an internship

Why not just get an internship instead of taking CSE 403?

CSE 403 vs. an internship

Internship:

- Get paid
- See the real world, warts and all

CSE 403:

- Choose your own project
- Significant input into the development process
- All aspects of the project: conception to deployment
- Detailed feedback and support
- Internships may require project experience

After an internship, there is still more to learn and practice. Both are valuable experiences. With a good attitude, you will benefit, such as honing your mentorship skills.

Does CSE 403 still matter?



This is a question worth asking! Or: what parts still matter?

Does CSE 403 still matter?



If you cannot solve a problem, you cannot delegate it or evaluate a solution.

- You are held responsible for the code you “produce”

AI is bad at setting direction (vs. executing a plan)

- CS was never about coding — it is about problem solving
- Prompting requires the same skills that design does:
precision, completeness, decomposition, abstraction, ...

AI is bad at innovation

- What does the user need (as opposed to want)?
- The need for engineering judgment (vs. reward hacking)

LLMs are weak outside their training data

Should a student take a software engineering class, given that AI agents are extremely effective at producing software?

Claude says:



Should a student take a software engineering class, given that AI agents are extremely effective at producing software?

Claude says:

This is a genuinely interesting question



Should a student take a software engineering class, given that AI agents are extremely effective at producing software?

Claude says:

This is a **genuinely interesting question**, and the answer is pretty clearly **yes**.



- **AI agents are tools, not autonomous engineers.** Someone has to define requirements, evaluate outputs, make architectural decisions, and know when the AI is confidently wrong.
- **You can't supervise what you don't understand.** ...
- **The floor rose, not the ceiling.** AI makes mediocre coding faster and cheaper. What's now scarce and valuable is the ability to build *complex, reliable, well-architected systems*.
- **Debugging and maintenance dominate real work.** ...
- **Security and correctness matter more now.** AI agents introduce new failure modes — hallucinated APIs, subtle logic errors, insecure patterns. Someone has to know enough to catch those.

Should a student take a software engineering class, given that AI agents are extremely effective at producing software?

Claude says:

This is a **genuinely interesting question**, and the answer is pretty clearly **yes**.



#1, #2, and #5 seem redundant with one another.

Fair point. All three are essentially saying "you need to understand the thing you're overseeing." That's one argument, not three.

faster and cheaper. What's now scarce and valuable is the

For #4, can't an LLM do the debugging and maintenance?

To a significant extent, yes — and that does weaken the argument. LLMs are genuinely good at debugging, reading error traces, and making localized fixes.

Course overview

5 meetings per week

- **Lectures:** MWF; some in-class activities, especially on Fridays
- **Team meeting:** Tuesday
- **Meet with your TA:** Thursday

The first week and a half have a different schedule.

Logistics: resources

- **Course website:** cs.uw.edu/403

All relevant information is on the website, or linked from it.

- Submit assignments via **Canvas**: <https://canvas.uw.edu>
- Discussions via **Ed**

Course overview: grading

Grading

- 45%: Course project
 - 70% project milestones
 - 30% final project review
- 20%: In-class exercises and individual assignments
- 20%: Exams
- 15%: Participation
 - Engagement in project meetings
 - In-class polls, discussions, Q&A, etc.
- No final exam
 - Final presentations in final exam slot (maybe)

Course overview: expectations

- Ability to **program** (in any programming language)
- Actively **participate** in discussions (in class, in teams)
- Take **initiative** and be accountable (to yourself, to your team)
- Commit to responsible **teamwork** and communication
- **Reflect** on experiences and continuously **improve**
- **Go beyond adequate**

Class and team expectations

Participate
Engage
Take initiative
Be respectful
Be responsible
****Communicate****
Reflect
Improve
Deliver

Course overview: use of AI

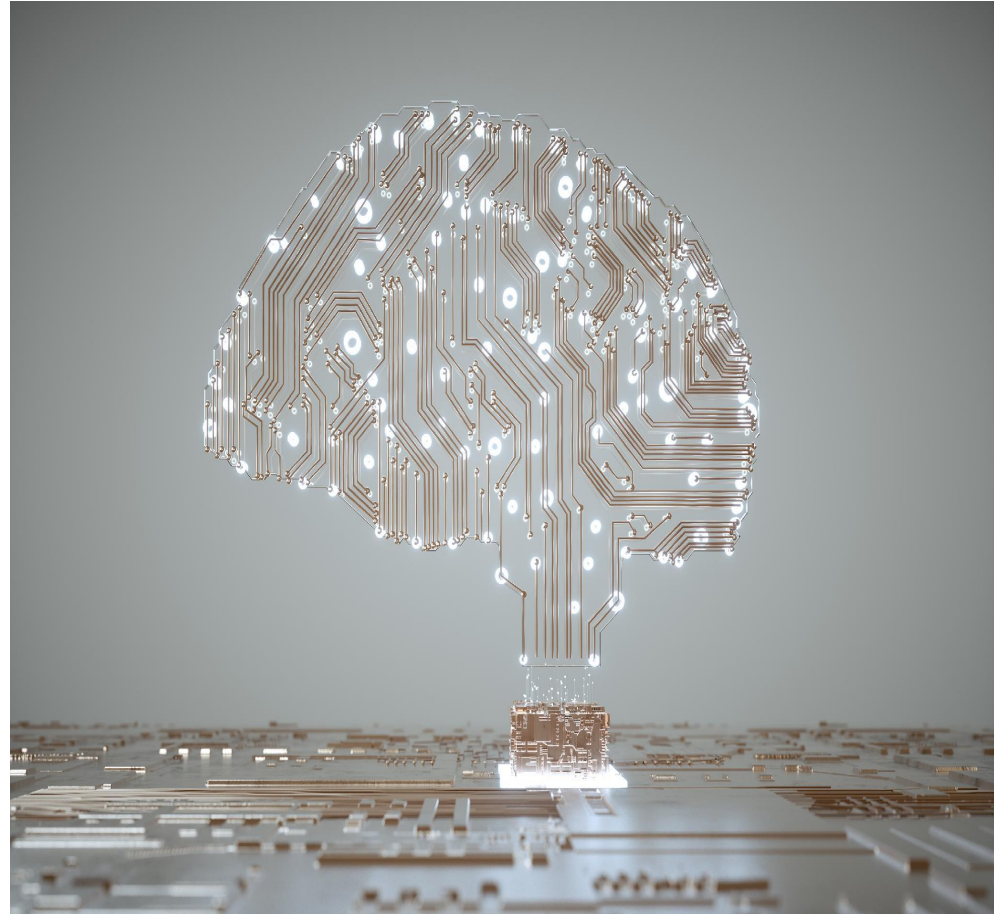
Use of AI is encouraged

- AI is never required

Do not use AI for text in assignment submissions

- AI tends to be verbose and includes irrelevant details

See policy section of syllabus



Course overview: workload and topics

Workload

- One project assignment each week
- Readings
- In-class exercises

Course overview: topics and project

- **Software processes, requirements, and specification**
 - Range of software development processes.
 - Precise capture of requirements and specifications.
- **Software development**
 - Decompose a complex problem and build abstractions.
 - Improve your coding skills.
 - Effectively use version control, build systems, and code review.
 - Continuously develop and integrate code.
- **Software testing and debugging**
 - Effective, complete, and automated tests
 - Modern testing and debugging techniques
- **Deliver a significant product as part of a technical team**

You will experience

- Be exposed to some of the best software development practices in use today
- Understand how software is produced – from conception to continuous development and release
- Develop skills to effectively collaborate with others towards a common delivery goal
- Experience the responsibilities, issues and tradeoffs involved in making decisions as software engineers

Course project overview

What's the
difference
between a
PROJECT
and a
PRODUCT?

Teamwork

For **1 week**, you will work on a pitch with a **partner**.

- A partner is required. Find one in tomorrow's section.

For **9 weeks**, you will work in a **5- or 6-person team**,
with teammates you choose,
on a project you choose.

The CSE 403 project

- This week: develop a **product pitch**
- A subset will move forward (Shark Tank)
- You will join a project team
- The rest of the quarter, your team develops the product, with incremental deliveries including demos
- Weekly cadence:
 - **Tues** section for team meeting
 - **Tues** milestone deliverables
 - **Wed** project update report
 - **Thurs** section for project meetings with TA

Course project: example categories

Create a *new* program/app/plugin/feature that scratches your itch.

Example categories

- Productivity and convenience apps
- Optimization problems and data science
- Gaming and making
- Extensions or plugins to open-source software
- Software engineering research (prototypes & experiments)

Some example products

- Browser plug-in to consolidate cookie management
- Plugin using AI to automatically add code comments
- CallHome reminder; topic ideas from calendar, news
- HowTheyVote tool to identify Congress voting history
- Music editor with search and splice capabilities

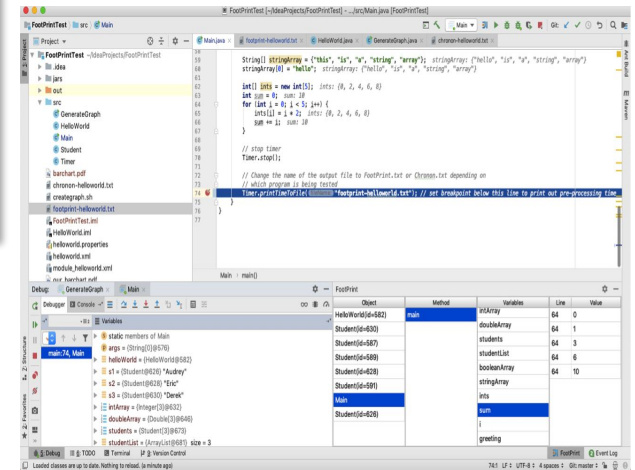
The SuiteRates application has a blue header with a house icon and the text 'SuiteRates'. Below the header is a navigation bar with links: 'Home | My Groups | Join a Group | Create a Group | Logout'. The main content area is titled '7th Ave S. St. Patty's Party'. It features a section 'Add an Expense' with a form containing fields for 'Name', 'Amount' (\$100), 'Due date' (1/2/07), and checkboxes for 'recurring?' and 'Upload receipt'. Below this is a table of 'Members' with columns 'Name', 'Weight (%)', and 'Amount'. The table lists: You (10, 10), Tom (5, 5), Joe (85, 85), Nancy, Maria, Tammy, George, Chad, and Mary. At the bottom right is a 'post expense' button.

Name	Weight (%)	Amount
You	10	10
Tom	5	5
Joe	85	85
Nancy		
Maria		
Tammy		
George		
Chad		
Mary		

- App to split roommate costs/payments

The DuoCode login page has a blue header with the 'DuoCode' logo. Below the logo is a paragraph: 'DuoCode makes learning code more fun and accessible than ever. It caters to a diverse range of skill levels and needs. Start learning today!'. There is a 'How-to' section with links 'Sign up or Sign in to get started! For a detailed user guide, please visit the Help page.' and a 'Log In' form with 'Username' and 'Password' fields and a 'Login' button. At the bottom, there are links for 'Don't have an account yet?' and 'Create an account'.

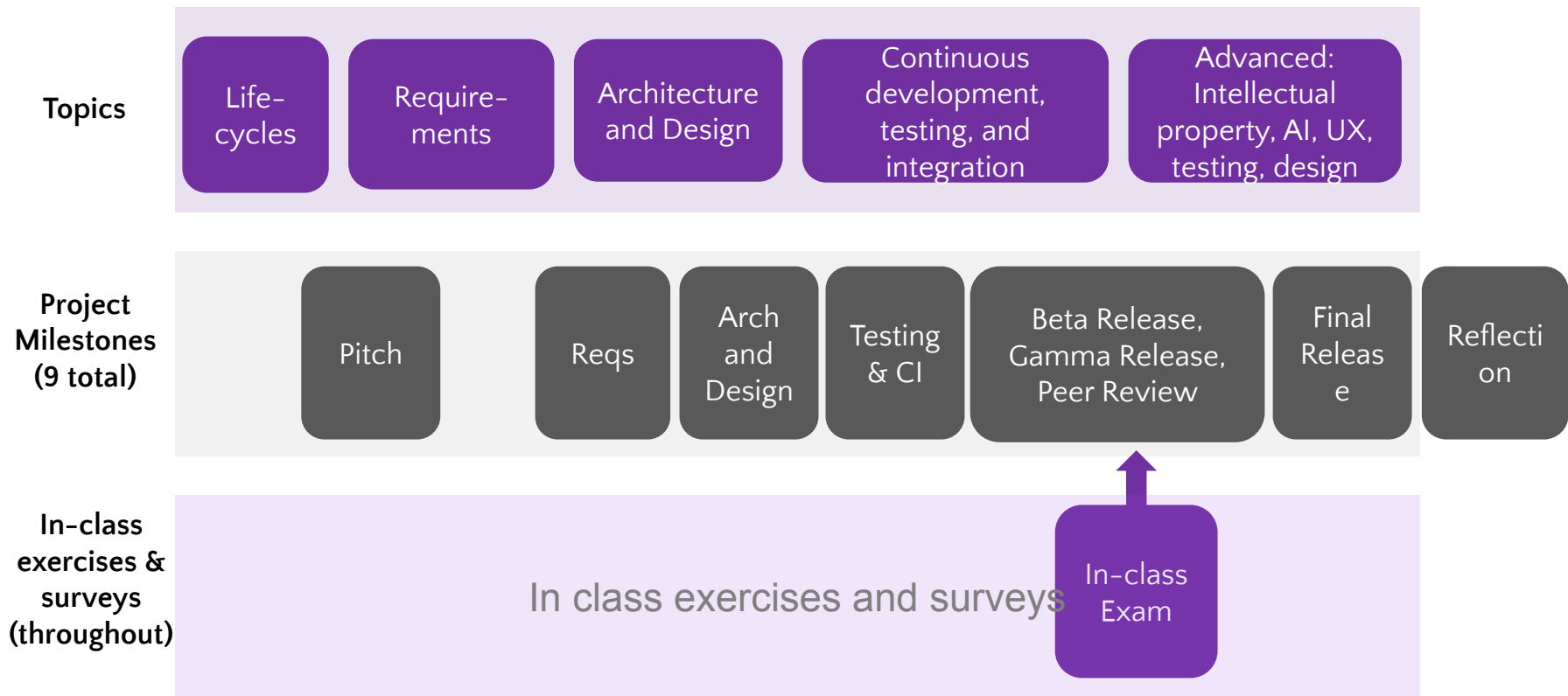
- DuoCode (inspired by Duolingo) to learn coding



- Plugin to view history of variables

Course overview: *approximate* schedule

Important: See Calendar and Canvas for current details of topics and assignments



CSE 403: challenges for students

Creativity in product pitches

- Go beyond social-networking product ideas

Teamwork

- Effective communication and coordination
- Different backgrounds, skills, and incentives

Complexity

- Tooling and technology stacks
- Scale of code base

Uncertainty

- No simple check-box grading
- Trade-offs, decisions, and justifications

CSE 403: challenges for students and staff

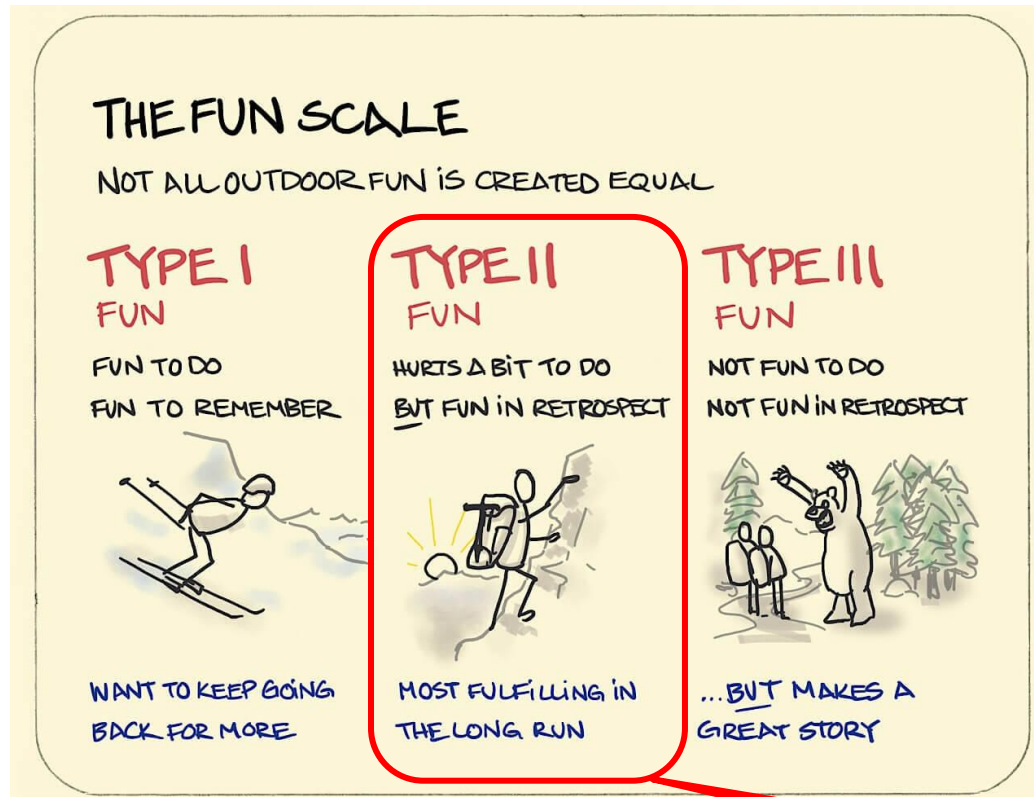
The Week-1 rush



Lecture time (12:30)



CSE 403: mostly type II fun



Sweet spot for teaching

Assignment 1 – Project Proposals

- Prepare a product pitch in teams of 2
 - Identify your group and submit the group to Canvas (see Calendar)
 - Think about a problem (a pain point) for some target customer set
 - Competitive analysis: what already exists?
 - Describe your solution, its value proposition, and its high-level technical approach
- Record and submit your pitch
- View others' pitches
- Form a team

See Canvas and the class calendar for more specifics

What's next?

- The Joel Test: basic software engineering processes
- Work on project pitch tomorrow in section

One last but important piece of information

- Classroom emergency preparedness
 - Know your exits
 - Fire situation
 - Shelter in place situation