Software Design and Development

Administrative Matters First

- Syllabus
- Reverse roll call.

This course’s target audience

- This course is intended as a final semester undergraduate course.
  - It builds on the undergraduate software engineering course
  - I assume you know
    - The basic ideas behind automated tests
    - Design patterns
    - Agile and Waterfall design
    - The basic idea of continuous integration
    - How to use version control software like git
  - I am aware that some of you have done this at a more theoretical than practical level.

How did we get here?

- Our modern program is ABET accredited
- Required “continuous improvement” where we are constantly looking at our program and finding ways to make it better
- This course originally was designed to patch several holes in the curriculum
  - But wasn’t the intended purpose.
- As we have improved the program we can use it for its intended purpose to reinforce and complete the rest of the undergraduate program
  - And use it for graduates to complete a transition from undergraduate to graduate.

Constituent groups

- So where did this continuous improvement come from?
  - Outside industry advisory board
  - Graduating seniors
    - Asking sophomores vs asking seniors who are applying for jobs
    - Is any company asking to see transcripts any more?
  - Alumni
  - Graduate advisory board
    - Which consists of graduates from our program along with a few other Master’s degree holders.
  - Departmental self-reflection

Group Work

- At one time we didn't do enough group work
- Theoretically today we do it in comp152, comp350, comp390 and here.
  - And for the grads in several classes.
- Group work is tricky enough at the university level
  - Without the added complexity of remote
  - But real software dev work is done in groups – exclusively
  - Not always fun at first for all students
- But how well did it actually go – especially earlier?

• Not enough work on large projects

- Large Projects in academia are/were quite scarce
- I first heard this critique in our industry advisory board
- Then from many (many) talks by industry professionals as the biggest problem with academic computer science programs
  - At one point Nowhere in our required courses were students given a large code base, some documentation, and a pointer to a problem and told “fix it”
  - This is a glaring hole
  - Have we done it already elsewhere in the program yet?

Living with your own work

- Another thing that we need to see more of is students who have to live and work with the results of their own earlier decisions
  - Something that industry practitioners argue for a lot
  - Give the first (of many?) rants about industry code vs academic code
- I believe we are now doing this in comp390 software engineering?
- How about at the graduate level?
-
- You will have the opportunity to do this in the first project

Project oriented goals for the course

- You sit down in front of a ‘big’ (40 files or so) project and
  - Realize its no big deal, nothing that 20-40 hours won’t be enough to get your head around.
  - You just start using documentation and testing to learn the new system.
- And on our way there
  - Practice agile development by building a larger project in smaller incremental steps.
- Have your <gasp> moment here – not at your first job
  - And “impostor syndrome?”

Recent Alumni Survey

- One of the questions we asked on our alumni survey (hindsight is 20-20):
- What did you wish you learned at Bridgewater
  - Top answers:
    - Current Software Engineering practices
    - Web Development
    - Databases
    - Soft skills/dealing with people/companies
    - Building large projects
    - Using frameworks/APIs
  - Web development is offered regularly now.
  - How many of the others did we cover before this class?

The last major course goal

- Lifetime learning
  - One of the major goals of a college education
    - Teach you how to learn on your own.
  - When you begin (first two semesters) want lots of “hand holding”/help
  - By this course we want to take off the training wheels
    - I’ve heard from some colleagues that some reluctance
    - But as some of you who were in the recent CS club presentations
    - you’ll get “go look at this” in industry – and you have to go learn it and see if it will work.
    - Get some practice here

Soft Skills and Hard skills

- Soft Skills
  - I wanted a book on developing software in real like
  - I've tried books, but in recent semesters I've used some podcasts
  - Cowboy coding bad!
  - Take care of yourself
  - People with both coding and people skills will go far.
    - People with only one will need to work on the other

Getting ready to start

- Quick survey
  - How many of you have done json API work before?

Assignment

- Read the first chapter of the Pragmatic Programmer: “A Pragmatic Philosophy”
  - we’ll discuss this in a week
- Join the slack workspace for this class
  - Install slack on devices you will use and check it regularly
  - The join link is in the one and only blackboard message for this class.
- If you don’t have a github account for class use, make one
  - And send me your github ID. !!!!!!
    - This is the one that people often fail to remember. It is worth a quiz grade
    - Send it even if you’ve had me before and sent it to me once before.
- Make a free serpapi account
  - https://serpapi.com/