0ECE366 – Software Engineering & Large Systems Design
Spring 2023

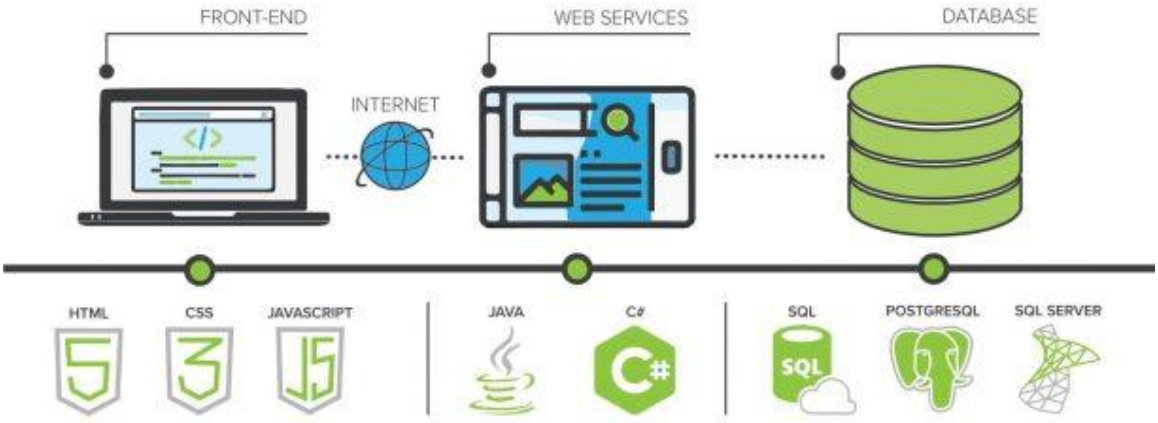
Department of Electrical Engineering, Albert Nerken School of Engineering
The Cooper Union for the Advancement of Science and Art

Semester Project

Project Overview

This project will span the entire semester and will be done in groups of 3-4. Each group will work on a project of their choosing and must be approved by me. If your group has trouble coming up with an idea, please chat with me in office hours.

The project must be a full-stack project, meaning there is a database, a service layer, and some front end or user interface (UI). In addition, this project should be deployed to the cloud for the rest of the class to demo in the final presentations. This project should be as feature complete as possible and hopefully something that you'd be proud to put on your résumé.



Project Requirements

- Your source code must be hosted on Github in an organization that your group will own.
- Your service layer must be written in Java.
- You must use Docker for development.
- Your final project must have a non-command-line user interface (web, mobile, GUI, etc.).
- Your service must use a persistent database.
- Your project must be hosted on the cloud so that everyone can access it on their computers via the internet.
- Your project must allow for basic user registration, login, basic profile, and account management.
- Your project should be achievable in one semester.

Project Suggestions

- A multiplayer game (e.g. Scrabble, Poker, Uno, Set, Monopoly)
- A social app (e.g. Twitter, Facebook, Instagram)
- A marketplace app (e.g. Uber, Grubhub, Zillow, Amazon)

Throughout the semester, you will emulate what it is like to work for as a software engineer. The requirements will be signed off by me and if there are any changes in your requirements, you are expected to properly document them and inform me in a timely manner.

You will be learning a lot of new things in this course and will need to learn more things on your own with online references. Make sure to reference any resources or code that you use. Also, take advantage of office hours and use me a resource of general knowledge as well.

Project Schedule

Jan 30 – Project Proposals & Pitch Presentations

Deliverables:

- Proposal document (no more than 3 pages)
- Short presentation (no more than 5 minutes)

You group must decide what you want to build. The proposal should be a detailed description of your project, the features you intend to implement, a rough breakdown and estimation of each of these features, and some high-level diagrams on the various components, objects, modules, and technology decisions that you decide to use. You may use the general stack that we will go over throughout the class (Postgres -> Spring Boot -> React) or decide to do something slightly differently.

The presentation should be no more than 5 minutes. In these 5 minutes, you will do a pitch of your project to the class. The presentation is 5/25 points of this grade. You should clearly review your project plans and discuss any risks you foresee in the project execution.

Here is a framework for your proposal:

- Proposal: What is the title of your project? Who is on your team? What are you building?
- Scope: What features will you deliver? Break these down into task and estimate the effort. What risks do you think you will encounter?
- Team: How are you dividing the work up? How will you keep each other accountable?

Mar 6 – Project Database and Backend Demos

Deliverables:

- Github organization with your repository(ies) containing your project code
- Prototype of Database
- Prototype of Service Implementation
- Demo in class (5-10 minutes)

Github Workflow

- Create issues to document your project and organize the parts of your project with appropriate tags
- Ensure that you are doing code reviews via Pull Requests in Github

Database

- Design your database schemas for the features you expect for your project
- Ensure that your database can start up locally with Docker

Service

- Start with the core features of your project
- This will involve the basic functionalities of the service interaction with the database along with the various service operations
    - Create, Read, Update, and Delete (CRUD)
    - HTTP GET/POST/PUT/DELETE operations
- You can hard code information as necessary
- You should have about 50% of your features implemented in this service

Other Requirements

- Tag a "release" in Github from the main branch for this submission
- All team members must write code and comment on PR's in Github
- Start thinking about organizing your code into modules/parts
- The service layer must be written in Java
- The entire project should be able to be built using Docker
- Make sure that you are regularly updating your issues in Github as that is the documentation of your requirements and completion status for the project. Be sure to document any changes as well.
- All team members should speak in the demos.
- Be ready for questions and to ask questions.

Other Notes

- No user interface is required yet; command line can be used in the demo
- Feel free to reach out to me for any PR's you'd like me to look at
- No unit tests are required yet, but if you have the time, you can start reading up on them and adding them in for extra credit. You will get an extra point for every 20% of the code you cover with the tests.

Apr 17 – Local Working Project Demos

Deliverables:

- A local end-to-end working version of your project
- This includes the entire stack: database, service, and UI
- Demo in class (5-10 minutes)

Service and Database

- The database schema design should be finalized
- You should have at least 75% of your features complete by this time
- The service code should have at least 25% of the code covered by unit tests

User Interface

- The user interface or front end should be in JavaScript or TypeScript. React is also strongly encouraged.
- This should interact with your local service and database that will run your local Docker

May 8 – Final Presentation

Deliverables:

- A fully hosted working project
- Demo in class (5-10 minutes)
- The class should be able to access this project and play the game (5-10 minutes)
- The service code should have at least 50% of the code covered by unit tests