CMPS 3390 Project 3

Fall 2024

BEFORE YOU BEGIN: FORM A TEAM & WRITE PROPOSAL

For Project 3 you will be responsible for all aspects of the project: who you will work with, what you will build, and how you will build it. There will be a loose set of requirements that must be met, but for the most part you should focus on applying all of the principles of application development you have learned throughout the semester.

Form Your Team

- Discuss your experiences/knowledge with other students in class
- Find others with similar goals, brainstorm app ideas, and be willing to compromise
- Decide on key features, tools, and timeline

Submit Proposal

You have two options:

1. Have one member of your team write a short proposal with the following information:
    - A list of team member's names
    - A short paragraph explaining your application (elevator pitch)
    - A list of the frameworks/tools/techniques you plan to use
2. Have your whole team come to my office to propose your idea in person.

Accept Feedback

I will be reviewing your proposals and providing you feedback and suggestions (either through email or in person). While I may not expect you to use ALL of my feedback, I would hope that you see the value of clearly defining all aspects of your app idea BEFORE you begin the development process. Solidifying your ideas and communicating with me and your team on a regular basis will help ensure that you meet all of the requirements by the due-date.

PROJECT REQUIREMENTS

1. **PREPARATION PHASE (Complete at least 4 of the following)**
    - User Profiles/Stories
    - Features/Requirements Document
    - Relational DB Schema/Diagram
    - App Architecture Document
    - Style Guide / UI Design Document
    - Research log

2. **MAIN APP FEATURES (Required)**
    - Version control to manage code changes
    - At least one pure server-side controller
    - At least 2 http API calls from the client **or** a websocket
    - Client-Side Data Model Classes
    - Well designed Graphical UI/UX

3. **ADDITIONAL FEATURES (Demonstrate understanding and use at least 4 of the following)**
    - Architecture patterns (e.g., MVC, MVVM, MVP, MVI, etc)
    - Design patterns (e.g., Factory Method, Builder Class, Singleton, Iterator, etc)
    - Persistent Data Storage (either locally or behind an API)
    - Concurrency/multithreading
    - Client/Server data validation/sanitization
    - Server-side rendering
    - 3rd-party APIs/integrations
    - API testing (e.g., curl, insomnia, postman, JaSON)
    - State-handing with asynchronous functions (callbacks **vs** promises **vs** async/await)
    - Virtual Machines/Containers