COMP 9783 Front-end Development

Final-Project - Custom App Development (32% of the course mark)

Name:

Student Number:

For the final project, students will **create their own unique application** and **implement a specific feature**. This project will allow students to demonstrate their understanding of front-end technologies, as well as their ability to apply these skills creatively to develop a fully functional application.

Outcomes

By the end of this final project, you should have implemented a feature of an application, using any of the diverse stack of front-end technologies covered in class such as:

- JavaScript
    - Vanilla JS
    - Node JS
        - Express
        - EJS
    - React
- CSS
    - Bulma
    - Bootstrap
    - Tailwind
    - Sakura

Final-Project Guidelines:

Here are guidelines for the Final-Project.

1. Think of **any application** that **you wish to implement** and choose to **build a specific feature**. For instance, It can either be a **complicated app** like an **E-Commerce app** with **many features** or a **simple app** like an **Expense tracker** with **fewer features**. For instance if you decide to go with an **E-Commerce app**, then the **features** would be **online store**, **payment** and **inventory management**, just **choose** to **implement a feature** that **you are capable of coding**. **Regardless** of **app complexity**, you have to **decide** the **scope** based on several factors such as **time constraints** and **knowledge of the different technology stacks**.
2. The feature chosen to be implemented should demonstrate **CRUD (create, read, update, and delete) tasks**.
3. Choose **any framework** or **technology stack** to support your app.
4. For backend data, your instructor will help on how to build a simple mock backend server for your app.

Mark breakdown

- HTML, CSS and JavaScript coding = 75 %
    - **Keep it Simple:** Write code that is easy to understand. Avoid unnecessary complexity.
    - **Descriptive Naming:** Use meaningful variable, function, and class names that clearly describe their purpose.
    - **Commenting:** Write comments to explain why certain decisions were made, not just what the code does.
    - **Single Responsibility Principle:** Each function or module should have a single responsibility.
    - **Reusability:** Write reusable functions and components to avoid duplication.
    - **Graceful Failures:** Handle errors gracefully to avoid crashing the application.
    - **Logging:** Log errors and important events to aid in debugging and monitoring.

- **ES6+:** Utilize modern JavaScript features (e.g., arrow functions, destructuring, template literals).
- **Async/Await:** Use async/await for handling asynchronous operations instead of callbacks or then/catch.
- **Scope Management:** Limit the use of global variables to avoid conflicts and unintended side effects.
- **Modules:** Use modules to encapsulate code and manage dependencies.
- **Array Methods:** Use array methods (e.g., map, filter, reduce) instead of for loops when appropriate.
- **Minimize Side Effects:** Keep functions pure and minimize side effects during iterations.
- **Functional Components:** Prefer functional components with hooks over class components.
- **Component Reusability:** Create small, reusable components to build a scalable UI.
- **Separation of Concerns:** Separate UI logic, state management, and business logic.
- **Local State:** Use local state for component-specific data.
- **Context API:** Use React Context for global state management when necessary.
- **useEffect:** Use useEffect for side effects, ensuring dependencies are correctly managed.
- **Cleanup:** Always clean up effects to avoid memory leaks and unintended behavior.
- **Default Props:** Define default props for optional props to ensure components behave consistently.

- Functionality of the feature: 25 %
  - Is the functionality chosen able to demonstrate **CRUD tasks**?

Submission:

1. Create a repo on GitHub with the name: **COMP9783-55868-Final-Project**.
    a. If the repo is **private** add: **roderickkit-bernardo** as a **collaborator**.
2. Upload all the files (**except for node_modules**) needed to run your app to this repo: **Final-Project-COMP9783-55868**.
3. Send an email to roderick.bernardo@georgebrown.ca with details about your repo and if necessary include build instructions.