User Documentation for Student Projects

Introduction

This document provides detailed descriptions, scopes, and technologies to be used in executing your final project as listed below. These projects are designed to reinforce your knowledge of Git, HTML, CSS, and JavaScript while introducing fundamental concepts in UI interaction, CRUD operations, local storage, and API integration.

Note:

1. The group leader is to create a git repository with the `title of your project`, and `add all group members` to the repository as collaborators. Each group member must make atleast one commit to the repository showing that he/she participated in the final project. This project is aimed at not only reinforcing you knowledge, but also to enshrine the act of collaboration between developers
2. Write a comprehensive markdown file named `Readme.md` that includes the details of the project and its dependencies/technologies as well as the importance or relevance of the project. Also include directives on how to get the web app up and running on another person's local pc
3. When you are done with this project, each member of the group is to write a comprehensive article concerning the project journey from start to completion, detailing his/her contribution to the project. This article post should be made on LinkedIn and Facebook. Make sure to include the link to the github repository in the post, and tag all your group members who contributed to this project. Finally, give the necessary shoutouts to the sponsors of the program and also to your tutor.

---

1. Quiz App (Interactive UI)

Project Overview

The Quiz App is an interactive web-based application that presents multiple-choice questions to users, records their responses, and provides immediate feedback. The goal is to create an engaging user experience while reinforcing JavaScript event handling and DOM manipulation skills.

Scope

- Display a series of multiple-choice questions.
- Track user answers and calculate the score.
- Provide immediate feedback on whether an answer is correct or incorrect.
- Display the final score after all questions are answered.
- Implement a restart option to retake the quiz.

Technology Stack

- **HTML**: Structure the quiz interface.
- **CSS**: Style the UI for better user experience.
- **JavaScript**: Handle user interactions, track scores, and update the UI dynamically.
- **Git**: Version control and collaboration.

Expected Outcome

Students will learn how to manipulate the DOM dynamically, manage events, and provide user feedback in real-time.

2. To-Do List App (CRUD Operations, Local Storage)

Project Overview

The To-Do List App is a simple task management application that allows users to add, edit, delete, and mark tasks as completed. The application stores data in the browser's local storage to retain tasks even after the page is refreshed.

Scope

- Users can add new tasks.
- Users can edit and delete tasks.
- Users can mark tasks as completed.
- Data should persist using local storage.
- UI should reflect task states dynamically.

Technology Stack

- **HTML**: Structure the to-do list.
- **CSS**: Style the application for an appealing layout.
- **JavaScript**: Implement CRUD operations (Create, Read, Update, Delete) and manage local storage.
- **Git**: Track changes and maintain project history.

Expected Outcome

Students will understand CRUD operations, how to work with local storage, and how to maintain state in JavaScript applications.

3. Weather App (Fetch API)

Project Overview

The Weather App fetches real-time weather data from an API based on user input (city name) and displays temperature, weather conditions, and other relevant information.

Scope

- Users can enter a city name to get real-time weather information.
- Fetch weather data using an API.
- Display key weather details (temperature, conditions, etc.).
- Handle errors for invalid city names or network issues.
- Provide a responsive UI.
- Utilize the OpenWeather API OpenWeatherMap API

Technology Stack

- **HTML**: Basic input and display structure.
- **CSS**: Style the app for a clean UI.
- **JavaScript**: Fetch data from a weather API and dynamically update the UI.
- **Git**: Manage version control and track progress.

Expected Outcome

Students will learn how to fetch external data using the Fetch API, handle API responses, and manage asynchronous JavaScript operations.

Conclusion

These projects will reinforce students' understanding of fundamental web development concepts while providing hands-on experience in real-world applications. They will also practice using Git for version control, ensuring they follow industry-standard development workflows.