Project Name: Tasks Manager App

Overview: Task Manager App is a web application designed to help users manage their tasks efficiently. It provides an interactive platform for users to create, organize, track, start, and complete tasks effectively. The app aims to streamline the task management process and enhance productivity.

Planning

- **Brainstorming**:
  - Reviewed previous lab exercises and examples for inspiration.
  - Decided on the project idea: Task Manager App - A task management application.
- **MVP Features**:
  - Task Creation: Users can create new tasks specifying details such as title, description, due date, priority.
  - Task Complete: Users can mark tasks as completed and move to another sections.
  - Task List: Users can view a list of all tasks and filter them based on completion status or due date.
  - Task Deletion: Users can delete tasks from the task list or completed tasks, removing them permanently from the database.
  - User Search: Users can search for tasks by their titles.
  - Profile Update: Users can update their profile image and username.
- **Wireframes**:
  - Wireframe for user login View
  - Wireframe for user sign up View.
  - Wireframe for Task Creation View
  - Wireframe for Task List View
  - Wireframe for Task Complete View
- **To-Do List for Coding**:
  1. Set up HTML, CSS, and JS files and link them together for the front end.
  2. Create a basic server using Node.js and Express.js for the back end.
  3. Build HTML structure for login page, sign up page, Task Creation, Task List, and Task complete views.
  4. Write JavaScript functions for handling task creation, deletion, tracking, and completion (front and back end).
  5. Style each view using CSS, including flexbox layout and animations.
  6. Ensure at least one view is responsive for different screen sizes.
  7. Implement GET endpoints on the server for fetching tasks.
  8. Implement POST endpoints for creating new tasks.
  9. Implement PUT endpoints for updating task status, including start and completion.
  10. Utilize Sequelize for database operations.

11. Create a seed file or function to populate initial task data.
12. Connect the front end to the server using Axios for making requests.
13. Implement DELETE endpoints on the server for deleting tasks.

- **Pass off Plan**:
  - Review the plan with a staff member for feedback and adjustments.
- **Set up GitHub Repo**:
  - Create a folder on the computer.
  - Initialize a remote repository on GitHub.
  - Connect the local folder to the remote repo.
  - Upload the completed planning documentation to the GitHub repo.

Data Model

- **Tasks**:
  - ID (Primary Key)
  - Title
  - Description
  - Due Date
  - Priority (Low, Medium, High)
  - Status (Pending, In Progress, Completed)
- **Completed_tasks**:
  - ID (Primary Key)
  - Title
  - Description
  - Due date
  - Priority
  - Status
  - Com_id reference tasks(id)

Users:

- ID (Primary Key)
- Username
- email address.
- password

Database

- Project includes a seed file or function to populate initial task data.
- App uses 3 tables for tasks, completed_tasks and Users.
- App uses a foreign key to link tasks with completed_tasks
- App uses a foreign key to link tasks with users.

Coding

- Follow the outlined workflow for coding, starting with setting up HTML, CSS, and JS files.
- Build and test each feature incrementally, ensuring functionality and styling.
- Continuously update and refine the codebase based on testing and feedback.

Presentation

- Discuss project purpose and demonstrate MVP features.
- Do not discuss broken/unimplemented features.
- Recording is between 2-3 minutes.

This plan provides a structured approach for developing the Airlines Mechanics Tasks application, including database design and data model, ensuring adherence to the project requirements, and facilitating effective execution throughout the development process.