Group Number: 4

Group Members: Nick DeVeau, Jannatul Zaman, Zihao Li, Joshua Jones, Vrutik Patel

PlanPal Development Design

Intended Use of the System:

PlanPal is designed to cater to a wide range of users, regardless of their occupation or background. The app will allow users to label their tasks with unique and memorable names, assign deadlines, and receive timely reminders for upcoming tasks. Whether a professional managing multiple projects, a student organizing assignments, or anyone looking to keep track of daily chores, PlanPal will help users stay on top of their responsibilities. The app is optimized for both desktop and mobile platforms, providing flexibility for users to manage their tasks anytime, anywhere.

Overall Functionality:

PlanPal will help users manage their tasks and schedules effectively by providing an intuitive interface to:

- Create, edit, and delete tasks.
- Assign deadlines and set task priorities.
- Receive customized or pre-set notifications for upcoming deadlines.
- Filter and search tasks based on various criteria like status, due date, or priority.

Main Components of the System:

1. **Front-End (User Interface):**
    - The front end of the app will be built using **React**, which is well-suited for web development due to its simplicity and component-based architecture. React will enable the creation of a dynamic and interactive user interface that is easy to navigate.
    - **Tools and Environment:** We will use **Node.js** and **npm** to manage and run the development environment. This setup provides the flexibility to later convert the app to React Native if we decide to expand to mobile platforms, as React and React Native share a similar structure and syntax.

2. **Back-End (Database and Data Management):**

- For the back end, we will use **Firebase**, a platform specifically designed for web and mobile applications that require robust user authentication and secure data storage. Firebase offers:
    - **Database Services:** Firebase provides a scalable database solution (Firestore or Realtime Database) for storing user data like task lists, reminders, and account details.
    - **User Authentication:** Firebase Authentication simplifies the management of user accounts by handling sign-ups, logins, and password resets, and includes built-in security features.
- **API:** We will use a **REST API** to facilitate communication between the front end and back end. REST is a simple and widely-adopted choice that provides the necessary tools for our app's straightforward data transfer requirements.

3. **Hosting:**
    - **Firebase Hosting** will be used to deploy and serve the web app. Firebase provides secure and scalable hosting solutions that are well-integrated with other Firebase services, ensuring smooth deployment and operation.
    - For a custom domain name, we will use a domain registrar such as **GoDaddy** to register our domain and configure it to point to Firebase Hosting.

4. **User Authentication:**
    - **Login and Registration:** Implement user accounts with secure registration and login functionality, utilizing Firebase Authentication or OAuth for managing user credentials.
    - **Security Measures:** Employ encryption for secure password storage and transmission, ensuring user data remains protected at all times.