Project Report for Recipe & Meal Planner PWA

Project Overview

The Recipe & Meal Planner PWA is designed to help users view recipes, plan meals, and create grocery lists, offering a seamless and organized cooking experience. The app serves families and individuals looking to streamline meal preparation and planning. By providing offline functionality and easy synchronization, it ensures users can access their data anytime, anywhere.

Technical Implementation

The app uses the following technology stack to provide robust functionality:

- **Frontend**: HTML, CSS (Materialize Framework), and JavaScript for responsive and user-friendly interfaces.
- **Backend**: Firebase Firestore for online data storage, managing user accounts, and secure data transactions.
- **Offline Storage**: IndexedDB ensures the app is functional offline, storing user data locally. The integration of Firebase and IndexedDB ensures smooth data synchronization and secure user authentication.
- **Service Worker**: Enables offline access by caching assets and scripts.
- **Manifest.json**: Provides metadata for PWA installation, enhancing user accessibility.

Challenges and Solutions

1. **Data Syncing**:

   I encountered a problem with syncing meal plan data between Firebase and IndexedDB. While the data was saved offline in IndexedDB, it was not automatically synchronized with Firebase when the app reconnected to the internet. To resolve this, I implemented an event listener for the [Online] event, which triggered a function to retrieve data from IndexedDB and upload it to Firebase. Additionally, I added error handling to ensure the synchronization process would not fail silently if there were network or authentication issues.

2. **User Authentication Issues**:

   Initially, there were inconsistencies in retrieving the [currentUser] object when the app loaded. This led to errors in saving and fetching user-specific data. I addressed this by adding a [userLoggedIn] custom event, which ensured that user data was only accessed after authentication was fully established. This change significantly improved the app's reliability.

3. **Saving a User's Dataset in Firebase**:

   At first, saving meal plans for individual users in Firebase was a challenge because I needed to structure the data to allow efficient retrieval and updates. I solved this by organizing the

database into a [users] collection, where each user document included a [mealPlans] field. I used the authenticated user's UID to uniquely identify their data and ensured proper error handling when performing CRUD operations. This approach streamlined the storage of user-specific datasets.

Lessons Learned

This project enhanced my understanding of building Progressive Web Applications with reliable offline functionality. I learned how to integrate IndexedDB and Firebase for effective data management, troubleshoot complex synchronization issues, and design user-friendly interfaces. Additionally, I have gained practical experience in debugging, testing cross-device compatibility, and optimizing app performance for real-world use cases. For example, I have wasted a good amount of time (maybe about 4-6 hours) searching for the error from my console log, finally realizing that just clearing site data from [application >> storage] solves it. All in all, this course has been a productive one and I have learned a lot from it.