**AI Powered Recipe APP**

**Muhammad Zafeer(2521113)**

**1. Introduction**

This document outlines the structure and functionality of an AI-powered Expo app designed to manage and display recipes. The app leverages the Gemini API to generate recipe ideas based on user-provided titles. The app also utilizes AsyncStorage for local data storage, ensuring data persistence even when the app is offline.

**2. App Structure**

The app comprises four key components:

**2.1 Model Component**

* A reusable component for adding new recipes and editing existing ones.
* Contains a text input field for capturing recipe titles.
* Triggers the Gemini API to generate recipe details based on the input title.

**2.2 Index Component (Home Screen)**

* Renders a flat list to display a collection of recipes.
* Includes a search bar to filter recipes by keywords.
* Allows users to navigate to individual recipe details or edit existing recipes.

**2.3 Dynamic ID Component (Recipe Detail Page)**

* Displays detailed information about a specific recipe.
* Provides options to edit or delete the selected recipe.
* Leverages the Model component for recipe editing.

**2.4 Layout Component**

* Serves as the app's entry point.
* Coordinates interactions between other components.
* Handles navigation and data flow within the app.

**3. Component Interactions**

**3.1 Adding a New Recipe**

1. User navigates to the "Add Recipe" screen.
2. Model component prompts for a recipe title.
3. Gemini API generates recipe details.
4. Generated recipe is added to AsyncStorage and displayed in the recipe list.

**3.2 Editing a Recipe**

1. User selects a recipe from the list or detail page.
2. Model component renders with existing title and details.
3. User modifies title or details.
4. Updated recipe is saved to AsyncStorage and reflected in the recipe list.

**3.3 Deleting a Recipe**

1. User selects a recipe from the detail page.
2. App prompts for confirmation.
3. If confirmed, the recipe is removed from AsyncStorage and the recipe list is updated.

**3.4 Searching Recipes**

1. User enters keywords in the Index screen's search bar.
2. Recipe list is filtered to display matching recipes.

**4. Technology Stack**

* **Expo:** Cross-platform mobile app framework
* **React Native:** JavaScript framework for native mobile apps
* **Gemini API:** AI-powered API for creative text generation
* **AsyncStorage:** Local data storage for offline persistence

**5. Benefits of Using AsyncStorage**

* **Offline Functionality:** Ensures data availability even without an internet connection.
* **Improved User Experience:** Provides a seamless experience by avoiding reliance on network connectivity.
* **Faster Performance:** Reduces loading times by accessing data locally.
* **Enhanced Security:** Stores sensitive data locally, minimizing exposure to potential breaches.

**6. Considerations for Using AsyncStorage**

* **Storage Limitations:** Has limited storage capacity, suitable for smaller datasets.
* **Data Security:** Sensitive data should be encrypted before storage.
* **Data Synchronization:** Requires manual or automated synchronization with a backend server for larger datasets.

By effectively utilizing AsyncStorage, the app can provide a robust and reliable user experience, even in scenarios with intermittent internet connectivity.