TIERS LIMITED SUMMER INTERNSHIP 2024

MOBILE APP DEVELOPMENT

Final Project

Final Project

Project Overview

Create a comprehensive Custom App using Flutter and Firebase. This app should incorporate all the key concepts learned throughout the course. The app must be user-friendly, visually appealing, and demonstrate a solid understanding of Flutter development principles and Firebase integration.

Functionalities:

1. **User Profile Management:**
    - **Local Storage of User Data:** Store user name, email and other information locally using SQLite or Shared Preferences.
    - **Firebase Data Segmentation:** Use the locally stored email as a key to segment user data in Firebase Realtime Database, ensuring that each user has their own separate data. Fetch and store data specific to each user's email.
2. **Data Management:**
    - **Add Data Entries:** Allow users to add new entries.
    - **Update Data Entries:** Enable users to update existing data.
    - **Delete Data Entries:** Provide functionality to delete records.
    - **View Data Entries:** Display a list of entries where each entry is clickable to show detailed information.
3. **Database Integration:**
    - Use Firebase Realtime Database to store and retrieve records.
    - Ensure that data is synchronized in real-time across different devices.
4. **Firebase Integration:**
    - **Real-time Updates:** Ensure that the app displays real-time updates when data changes in Firebase.
    - **Data Conversion:** Handle data conversion between JSON and Dart objects for Firebase operations.
5. **Data Presentation:**
    - **ListView and GridView:** Use ListView to show a list of entries. Optionally, use GridView to display items in a grid format.
    - **Search Functionality:** Implement a search feature to filter entries.
6. **State Management:**
    - Manage app state using stateful widgets and setState.
    - Demonstrate proper state management by updating the UI based on data changes in the Fierabase.
7. **Navigation and Routing:**
    - Implement a **NavigationDrawer** or **BottomNavigationBar** for main navigation according to your app design.

8. **Dialogs and Alerts:**
    - **Add Entry Dialog:** Use dialogs to collect information for adding a new entry.
    - **Edit Entry Dialog:** Use dialogs to update existing entry information.
    - **Confirmation Dialogs:** Show confirmation dialogs for delete actions.
9. **UI Design:**
    - Ensure that the app's user interface is attractive and not overly simplistic. The design should be intuitive, modern, and provide a pleasant user experience.
    - Use consistent theming, appropriate color schemes, and thoughtful layout arrangements to enhance the overall appeal of the app.
10. **Error Handling and Loading Indicators:**
    - Display loading indicators while fetching data from Firebase.
    - Handle errors gracefully and provide user feedback to ensure a smooth experience.
11. **Final Touches:**
    - Test the app thoroughly to ensure all functionalities work as expected.

Deliverables:

1. **Source Code:** Push your complete project code to GitHub and submit the link to the repository.
2. **Documentation:** Provide a brief document outlining what your app does, how to set it up, and any special features.
3. **Demo:** Include a short video or live demo of the app showcasing all implemented features.

Guidelines:

- **Originality:** You can create any app of your choice, but it should not be similar to any mini-projects built during the course.
- **Functionality:** All required features are implemented and work as expected.
- **Complexity:** Do not create a mini-app with just one or two screens. The app should be a full-fledged project that demonstrates your skills and understanding of the topics covered in the course.
- **Code Quality:** Write clean code with appropriate comments to make your code easy to understand and maintain.
- **API Integeration** (OPTIONAL): You may use any API to fetch and store data in Firebase. For example, if you're building a music app, you can use a music data API.
- **UI/UX:** The user interface is intuitive, responsive, visually appealing, and not overly simplistic.
- **Firebase Integration:** Correct and efficient use of Firebase services for data storage and real-time updates.
- **Error Handling:** Proper error handling and user feedback for a smooth experience.