Report: To-Do App with Firebase Integration

Github Repo Link: https://github.com/AvidAli1/note-taking-app

Name: Ali Ahmed

1. App Functionality Overview:

The app is a **To-Do List** application that allows users to manage notes with a simple sign-up and login mechanism. It leverages Firebase for user authentication and Firestore for storing and managing notes. The app has three main screens:

1. **Welcome Screen**: This is the first screen users see when launching the app. It provides options to either log in or sign up.
2. **Signup Screen**: Users can create an account by providing an email and password. Once the account is successfully created, they are navigated back to the Login screen.
3. **Note Management**: After logging in, users can create, edit, delete, and view their notes. The notes are fetched from Firestore, and any changes made to them are synced with Firebase.

Key Features:

- **Sign Up & Login**: Firebase Authentication is used to handle user registration and login.
- **Note Management**: Users can add, update, and delete notes.
- **Firestore Integration**: Notes are stored and fetched from Firebase Firestore.

2. Firebase Setup

The app is integrated with Firebase in the following ways:

- **Firebase Authentication**: Firebase's createUserWithEmailAndPassword and signInWithEmailAndPassword methods are used to authenticate users via email and password.
- **Firestore**: The app uses Firestore to store and manage the notes. Notes are fetched and updated in real-time using Firestore's snapshot listener.
- **Dependencies**: The app depends on the firebase_auth and cloud_firestore Flutter packages to handle authentication and database operations.

3. Firestore Rules

The current Firestore rules allow **full access to all documents** in the database without authentication. This means that any user (whether authenticated or not) can read and write data to any Firestore document. Here are the current Firestore rules:

plaintext

CopyEdit

```
rules_version = '2';


service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if true;
    }
  }
}
```

- **Explanation**:
    - allow read, write: if true;: This rule means anyone can read and write to any document in the Firestore database.
    - **Security Implications**: Since this rule is open, it is important to restrict access to the Firestore database for production by implementing user authentication and validation in the Firestore rules. Typically, you would use Firebase Authentication to limit access to users who are authenticated.

Recommended Firestore Rules for Production: To improve security, the Firestore rules should be updated to require authentication for accessing user data. Here's an example of more secure rules:

plaintext

CopyEdit

```
rules_version = '2';


service cloud.firestore {
  match /databases/{database}/documents {
    match /notes/{noteId} {
      // Only authenticated users can read and write their own notes
      allow read, write: if request.auth != null && request.auth.uid == resource.data.userId;
    }
  }
```

}

This rule ensures that only authenticated users can read and write notes, and that users can only access their own notes (identified by the userId field in the Firestore document).

4. Conclusion

This To-Do app allows users to manage notes securely, with Firebase Authentication handling user sign-up and login, and Firestore serving as the backend database for storing notes. The current Firestore security rules need to be updated to ensure data security by restricting access to authenticated users only.

Ending Remarks:

Extra features were not implemented but the code is fully functional with the core requirements fulfilled. You have user authentication and storage of email and password on Firestore database. Then you are taken to the homescreen where you are allowed to make add, edit or delete notes and all the changes are reflected in the Firestore Database.