Flutter — Login & Signup with Firebase (Quiz Submission)

Author: Generated by ChatGPT (example content for your quiz)

This PDF contains:
1) Step-by-step setup for Firebase + Flutter (email/password auth).
2) Complete example Flutter code (main.dart, login & signup pages, auth service).
3) Placeholder output screenshots (login & signup screens).
4) Instructions to create a Git repo and push the project, plus a sample GitHub link.

Use this as a template for your quiz. Replace placeholders (projectId, google-services files) with your Firebase project values and run on an emulator or device to capture real screenshots.

Firebase Setup & Project Configuration

Firebase setup notes: 1. In Firebase console create a new project. 2. Add Android app: provide package name (e.g. com.example.flutter_auth_demo).   - Download google-services.json and place android/app/. 3. Add iOS app if needed and download GoogleService-Info.plist into ios/Runner. 4. In project-level build.gradle and app-level gradle files, add Firebase plugins per docs. 5. Enable Email/Password Sign-in provider in Firebase Console > Authentication > Sign-in method.

pubspec.yaml (dependencies)

```
# pubspec.yaml (add these dependencies)
dependencies:
  flutter:
    sdk: flutter
  firebase_core: ^2.0.0
  firebase_auth: ^4.0.0
```

main.dart

```
 // main.dart import 'package:firebase_core/firebase_core.dart'; import
'package:flutter/material.dart'; import 'login_page.dart'; import 'signup_page.dart';
void main() async {   WidgetsFlutterBinding.ensureInitialized();   await
Firebase.initializeApp();   runApp(MyApp()); }  class MyApp extends StatelessWidget {
@override   Widget build(BuildContext context) {     return MaterialApp(       title:
'Flutter Firebase Auth Demo',       theme: ThemeData(primarySwatch: Colors.blue),
home: LoginPage(),       routes: {'/signup': (_) => SignupPage()},     );   } }
```

auth_service.dart

```
 // auth_service.dart import 'package:firebase_auth/firebase_auth.dart';  class
AuthService {   final FirebaseAuth _auth = FirebaseAuth.instance;
Future<UserCredential> signUp(String email, String password) async {     return await
_auth.createUserWithEmailAndPassword(email: email, password: password);   }
Future<UserCredential> signIn(String email, String password) async {     return await
_auth.signInWithEmailAndPassword(email: email, password: password);   }    Future<void>
signOut() async {     await _auth.signOut();   } }
```

login_page.dart

```
 // login_page.dart import 'package:flutter/material.dart'; import 'auth_service.dart';
class LoginPage extends StatefulWidget {   @override   _LoginPageState createState() =>
_LoginPageState(); }  class _LoginPageState extends State<LoginPage> {   final _emailCtl =
TextEditingController();   final _passCtl = TextEditingController();   final AuthService
_auth = AuthService();   bool loading = false;    void _login() async {     setState(() =>
loading = true);     try {       await _auth.signIn(_emailCtl.text.trim(),
_passCtl.text.trim());       ScaffoldMessenger.of(context).showSnackBar(SnackBar(content:
Text('Logged in')));     } on Exception catch (e) {
ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text('Error: $e')));     }
finally {       setState(() => loading = false);     }   }    @override   Widget
build(BuildContext context) {     return Scaffold(       appBar: AppBar(title: const
Text('Login')),       body: Padding(         padding: EdgeInsets.all(16),         child:
Column(children: [           TextField(controller: _emailCtl, decoration:
InputDecoration(labelText: 'Email')),           TextField(controller: _passCtl,
decoration: InputDecoration(labelText: 'Password'), obscureText: true),
SizedBox(height: 16),           ElevatedButton(onPressed: loading ? null : _login, child:
loading ? CircularProgressIndicator() : Text('Login')),           TextButton(onPressed: ()
=> Navigator.pushNamed(context, '/signup'), child: Text('Create account')),         ]),
),     );   } }
```

signup_page.dart

```
 // signup_page.dart import 'package:flutter/material.dart'; import 'auth_service.dart';
class SignupPage extends StatefulWidget {   @override   _SignupPageState createState() =>
_SignupPageState(); }  class _SignupPageState extends State<SignupPage> {   final
_emailCtl = TextEditingController();   final _passCtl = TextEditingController();   final
AuthService _auth = AuthService();   bool loading = false;    void _signup() async {
setState(() => loading = true);     try {       await _auth.signUp(_emailCtl.text.trim(),
_passCtl.text.trim());       ScaffoldMessenger.of(context).showSnackBar(SnackBar(content:
Text('Account created')));       Navigator.pop(context);     } on Exception catch (e) {
ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text('Error: $e')));     }
finally {       setState(() => loading = false);     }   }    @override   Widget
build(BuildContext context) {     return Scaffold(       appBar: AppBar(title: const
Text('Sign up')),       body: Padding(         padding: EdgeInsets.all(16),         child:
Column(children: [           TextField(controller: _emailCtl, decoration:
InputDecoration(labelText: 'Email')),           TextField(controller: _passCtl,
decoration: InputDecoration(labelText: 'Password'), obscureText: true),
SizedBox(height: 16),           ElevatedButton(onPressed: loading ? null : _signup, child:
loading ? CircularProgressIndicator() : Text('Sign up')),         ]),       ),     );   }
}
```

```
Create a git repository and push (example):
cd your_flutter_project
git init
git add .
git commit -m "Initial commit - Flutter Firebase Auth demo"
# Create a GitHub repo via GitHub UI, then:
git remote add origin https://github.com/YOUR_USERNAME/flutter-firebase-auth-demo.g
git push -u origin main
```

Git Instructions & Notes

```
Sample GitHub link (replace with your repo after you push):
https://github.com/YOUR_USERNAME/flutter-firebase-auth-demo
```

Figure 1: Login screen (placeholder).

Figure 2: Signup screen (placeholder).

ts (place