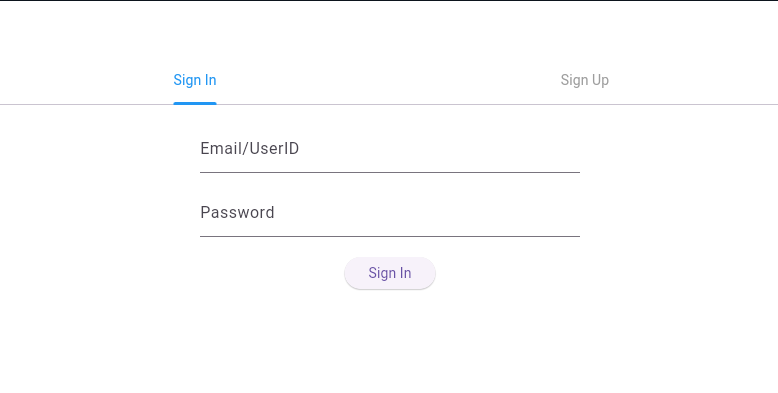
**COURSE NAME: FLUTTER & FIREBASE**

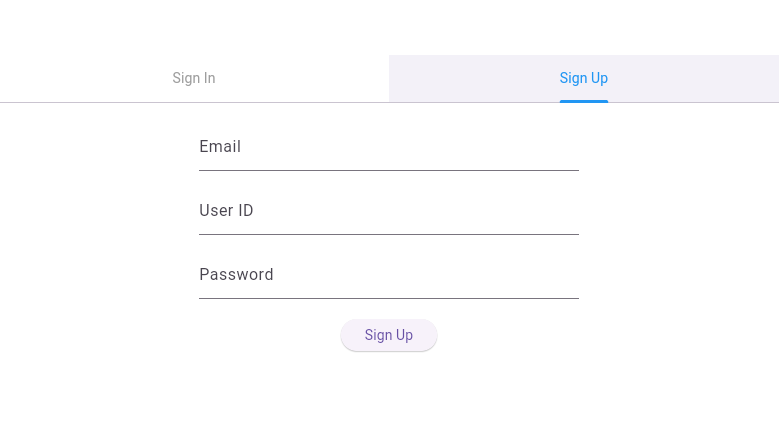
**TASK 1: CREATE A SIGN IN/SIGN-UP PAGE IN FLUTTER WHICH INCLUDES, BUILDING THE USER INTERFACE (UI) AND IMPLEMENTING THE AUTHENTICATION LOGIC.**

**SIGN IN/SIGN UP FORM CODE:**

import 'package:flutter/material.dart';  
void main() {  
  runApp(const AuthApp());  
}  
class AuthApp extends StatelessWidget {  
  const AuthApp({super.key});  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      title: 'Sign In / Sign Up',  
      theme: ThemeData(  
        primarySwatch: Colors.blue,  
        scaffoldBackgroundColor: Colors.white, // White background  
      ),  
      home: const AuthPage(),  
      debugShowCheckedModeBanner: false,  
    );  
  }  
}  
class AuthPage extends StatefulWidget {  
  const AuthPage({super.key});  
  @override  
  State<AuthPage> createState() => \_AuthPageState();  
}  
class \_AuthPageState extends State<AuthPage> with SingleTickerProviderStateMixin {  
  late TabController \_tabController;  
  final List<Map<String, String>> \_users = [];  
  final \_signInIdentifierController = TextEditingController(); // Email/UserID  
  final \_signInPasswordController = TextEditingController();  
  final \_signUpEmailController = TextEditingController();  
  final \_signUpUserIdController = TextEditingController();  
  final \_signUpPasswordController = TextEditingController();  
  @override  
  void initState() {  
    super.initState();  
    \_tabController = TabController(length: 2, vsync: this);  
  }  
  void \_signUp() {  
    final email = \_signUpEmailController.text.trim();  
    final userId = \_signUpUserIdController.text.trim();  
    final password = \_signUpPasswordController.text.trim();  
    if (email.isEmpty || userId.isEmpty || password.isEmpty) {  
      \_showMessage("All fields are required!");  
      return;  
    }  
    if (\_users.any((u) => u['email'] == email || u['userId'] == userId)) {  
      \_showMessage("User already exists! Try sign-in.");  
      return;  
    }  
    \_users.add({  
      'email': email,  
      'userId': userId,  
      'password': password,  
    });  
    \_showMessage("Sign-up successful! Now sign in.");  
    \_tabController.animateTo(0); // Switch to Sign-in tab  
  }  
  void \_signIn() {  
    final identifier = \_signInIdentifierController.text.trim(); // Email/UserId  
    final password = \_signInPasswordController.text.trim();  
    final user = \_users.firstWhere(  
      (u) => u['email'] == identifier || u['userId'] == identifier,  
      orElse: () => {},  
    );  
    if (user.isEmpty) {  
      \_showMessage("User not found! Please sign up.");  
      return;  
    }  
    if (user['password'] != password) {  
      \_showMessage("Invalid password!");  
      return;  
    }  
    \_showMessage("Welcome back, ${user['userId']}!");  
  }  
  void \_showMessage(String message) {  
    ScaffoldMessenger.of(context).showSnackBar(  
      SnackBar(content: Text(message), duration: const Duration(seconds: 2)),  
    );  
  }  
  @override  
  Widget build(BuildContext context) {  
    double width = MediaQuery.of(context).size.width;  
    double padding = width > 600 ? 200 : 20;  
    return Scaffold(  
      appBar: AppBar(  
        backgroundColor: Colors.white,  
        elevation: 0,  
        bottom: TabBar(  
          controller: \_tabController,  
          labelColor: Colors.blue,  
          unselectedLabelColor: Colors.grey,  
          indicatorColor: Colors.blue,  
          tabs: const [  
            Tab(text: "Sign In"),  
            Tab(text: "Sign Up"),  
          ],  
        ),  
      ),  
      body: TabBarView(  
        controller: \_tabController,  
        children: [  
           
          // ------------------ SIGN IN ------------------  
          Padding(  
            padding: EdgeInsets.symmetric(horizontal: padding, vertical: 20),  
            child: Column(  
              children: [  
                TextField(  
                  controller: \_signInIdentifierController,  
                  decoration: const InputDecoration(labelText: "Email/UserID"),  
                ),  
                const SizedBox(height: 16),  
                TextField(  
                  controller: \_signInPasswordController,  
                  decoration: const InputDecoration(labelText: "Password"),  
                  obscureText: true,  
                ),  
                const SizedBox(height: 20),  
                ElevatedButton(  
                  onPressed: \_signIn,  
                  child: const Text("Sign In"),  
                ),  
              ],  
            ),  
          ),  
  
          // ------------------ SIGN UP ------------------  
          Padding(  
            padding: EdgeInsets.symmetric(horizontal: padding, vertical: 20),  
            child: Column(  
              children: [  
                TextField(  
                  controller: \_signUpEmailController,  
                  decoration: const InputDecoration(labelText: "Email"),  
                ),  
                const SizedBox(height: 16),  
                TextField(  
                  controller: \_signUpUserIdController,  
                  decoration: const InputDecoration(labelText: "User ID"),  
                ),  
                const SizedBox(height: 16),  
                TextField(  
                  controller: \_signUpPasswordController,  
                  decoration: const InputDecoration(labelText: "Password"),  
                  obscureText: true,  
                ),  
                const SizedBox(height: 20),  
                ElevatedButton(  
                  onPressed: \_signUp,  
                  child: const Text("Sign Up"),  
                ),  
              ],  
            ),  
          ),  
        ],  
      ),  
    );  
  }  
}

**OUTPUT:**

****

****

**GITHUB LINK:** [**https://github.com/vaishnavi8754/Flutter\_tasks/**](https://github.com/vaishnavi8754/Flutter_tasks/)