**MPL Lab 2  
Mayur Jaiswal**

**D15B**

**24  
Aim:** To design flutter UI using common widgets

**Theory:** Flutter is a UI toolkit by Google used to build natively compiled applications using a single codebase. Flutter provides a rich set of **predefined widgets** that follow a declarative approach to UI design. Widgets like Text, Container, Row, Column, Image, and ListView are the building blocks of any UI in Flutter. These widgets can be combined, styled, and customized to build responsive and attractive user interfaces.

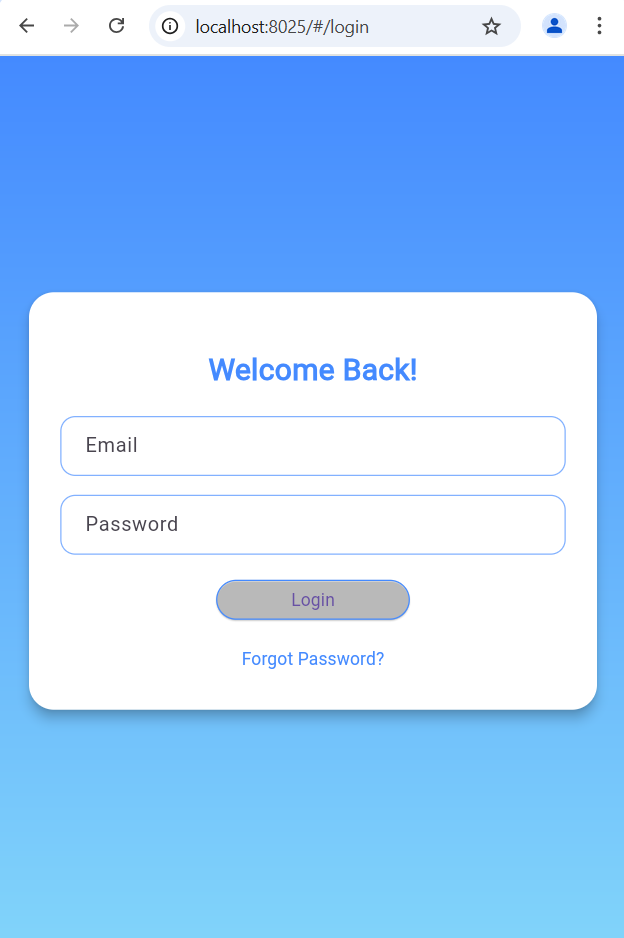
Flutter apps are structured as a **widget tree**, where each UI component is a widget (even layouts and styling). Using these widgets, developers can design both simple and complex UIs efficiently.

**Steps to Design Flutter UI Using Common Widgets**

1. **Set Up Flutter Project:**
2. **Use MaterialApp as Root Widget:**
3. **Design UI with Common Widgets in HomePage:**
4. **Use Layout Widgets for Positioning:**
   * Column and Row for vertical/horizontal layout.
   * Padding, SizedBox, and Align for spacing and alignment.
5. **Style UI Elements:**
   * Use TextStyle, BoxDecoration, and ButtonStyle.
   * Customize using ThemeData in MaterialApp.

**Code :**

import 'package:flutter/material.dart';  
import 'package:provider/provider.dart';  
  
class LoginScreen extends StatefulWidget {  
 @override  
 \_LoginScreenState createState() => \_LoginScreenState();  
}  
  
class \_LoginScreenState extends State<LoginScreen> {  
 final TextEditingController emailController = TextEditingController();  
 final TextEditingController passwordController = TextEditingController();  
 bool isLoading = *false*;  
  
 void loginUser(BuildContext context) async {  
 setState(() => isLoading = *true*);  
 try {  
 await Provider.of<AuthProvider>(context, listen: *false*)  
 .login(emailController.text, passwordController.text);  
 Navigator.pushReplacementNamed(context, '/home');  
 } catch (e) {  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text("Login Failed: ${e.toString()}"),  
 backgroundColor: Colors.redAccent,  
 ),  
 );  
 }  
 setState(() => isLoading = *false*);  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 body: Container(  
 width: double.infinity,  
 decoration: BoxDecoration(  
 gradient: LinearGradient(  
 begin: Alignment.topCenter,  
 end: Alignment.bottomCenter,  
 colors: [Colors.blueAccent, Colors.lightBlue.shade200],  
 ),  
 ),  
 child: Center(  
 child: Padding(  
 padding: EdgeInsets.all(*20.0*),  
 child: Card(  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(*20*),  
 ),  
 elevation: *8*,  
 color: Colors.white,  
 child: Padding(  
 padding: EdgeInsets.all(*25.0*),  
 child: Column(  
 mainAxisSize: MainAxisSize.min,  
 children: [  
 SizedBox(height: *20*),  
 Text(  
 "Welcome Back!",  
 style: TextStyle(  
 fontSize: *24*,  
 fontWeight: FontWeight.bold,  
 color: Colors.blueAccent,  
 ),  
 ),  
 SizedBox(height: *20*),  
 \_buildTextField(  
 controller: emailController,  
 label: "Email"),  
 SizedBox(height: *15*),  
 \_buildTextField(  
 controller: passwordController,  
 label: "Password",  
 isPassword: *true*),  
 SizedBox(height: *20*),  
 isLoading  
 ? CircularProgressIndicator()  
 : ElevatedButton(  
 onPressed: () => loginUser(context),  
 child: Text("Login"),  
 style: ElevatedButton.styleFrom(  
 padding: EdgeInsets.symmetric(vertical: *12*, horizontal: *60*),  
 backgroundColor: Colors.transparent, // Make background transparent  
 side: BorderSide(color: Colors.blueAccent), // Optional: Adds a border to the button  
 ),  
 ),  
 SizedBox(height: *15*),  
 TextButton(  
 onPressed: () {}, // Add forgot password logic  
 child: Text(  
 "Forgot Password?",  
 style: TextStyle(color: Colors.blueAccent),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
  
 Widget \_buildTextField(  
 {required TextEditingController controller, required String label, bool isPassword = *false*}) {  
 return TextField(  
 controller: controller,  
 obscureText: isPassword,  
 decoration: InputDecoration(  
 labelText: label,  
 filled: *true*,  
 fillColor: Colors.white,  
 contentPadding: EdgeInsets.symmetric(horizontal: *20*, vertical: *14*),  
 border: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(*12*),  
 borderSide: BorderSide(color: Colors.blueAccent),  
 ),  
 enabledBorder: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(*12*),  
 borderSide: BorderSide(color: Colors.blueAccent.shade100),  
 ),  
 focusedBorder: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(*12*),  
 borderSide: BorderSide(color: Colors.blueAccent, width: *2*),  
 ),  
 ),  
 );  
 }  
}

**Output:  
**