**Experiment No: 4**

**Aim:** Design an Interactive Form using form Widget and Navigation.

**LO2:** Design and Develop interactive Flutter App by using widgets, layouts, gestures and

animation

**Theory:**

Forms are an essential part of any application, allowing users to enter and submit data. In Flutter, the **Form** widget provides a structured way to manage multiple input fields with validation and state management. Additionally, navigation helps users move between different screens after form submission.

**Importance of Interactive Forms**

Interactive forms improve the user experience by providing real-time validation, guiding users with hints, and ensuring smooth transitions between screens.

**Components of an Interactive Form**

1. **Form Widget**
   * A container that holds multiple input fields.
   * It works with GlobalKey<FormState>() to manage form validation and submission.
2. **TextFormField**
   * Used for input fields like name, email, and password.
   * Supports built-in validation for checking user input.
3. **Validation**
   * Ensures correct data entry (e.g., checking if an email is valid).
   * Prevents submission if the input does not meet requirements.
4. **Navigation**
   * Moves users from one screen to another (e.g., Login to Homepage).
   * Uses Navigator.push() to transition between pages.
5. **UI Enhancements**
   * Adding colors, icons, and animations improves user interaction.
   * Using SizedBox and Padding ensures proper spacing.

**Steps to Create an Interactive Form in Flutter**

1. **Create a Form with Input Fields**
   * Use Form and TextFormField widgets for user input.
   * Assign a GlobalKey<FormState>() to manage form validation.
2. **Add Validation Logic**
   * Use validator in TextFormField to check inputs.
   * Display error messages for incorrect entries.
3. **Handle Form Submission**
   * Use onPressed in the submit button to validate the form.
   * If valid, navigate to the next page using Navigator.push().
4. **Navigate to Another Screen**
   * After successful login, move to the homepage using MaterialPageRoute().
   * Ensure proper state management for user data.
5. **Enhance the UI**
   * Use gradient backgrounds, shadows, and animations.
   * Add icons for a modern look.

**Example Use Case**

* A **Login Page** with username and password fields.
* A **Submit Button** that checks validation and navigates to the homepage.
* A **Signup Link** for new users to register.

Interactive forms are a crucial part of mobile applications, allowing users to input and submit data efficiently. In Flutter, the **Form** widget provides a structured way to handle user inputs, ensuring validation and smooth interaction. Forms typically consist of input fields such as **TextFormField**, where users can enter details like usernames, passwords, or email addresses. These fields are wrapped inside a **Form** widget, which helps in managing their state and validating the input before submission.

Validation is an essential aspect of interactive forms. It ensures that users enter correct and complete information before proceeding further. Flutter allows developers to add custom validation logic within the **validator** property of the **TextFormField** widget. For example, a password field can check if the input meets the required length or complexity. If the input is invalid, the form displays an error message, guiding the user to correct their entry.

Another important feature of interactive forms is **navigation**. Once the user submits a form successfully, they are usually redirected to another page, such as a homepage or a confirmation screen. This is achieved using Flutter’s **Navigator** class, which enables seamless screen transitions. The **Navigator.push()** method is used to move to a new page, while **Navigator.pop()** can take the user back to the previous screen. This navigation system helps in building an intuitive and user-friendly experience.

In addition to validation and navigation, Flutter provides customization options to enhance the visual appearance of forms. Features like animations, gradient backgrounds, and **Glassmorphism effects** can be added to make the form more engaging. Proper spacing, clear labels, and well-designed input fields improve usability and accessibility.

Overall, designing an interactive form using the **Form** widget and **Navigation** in Flutter helps in creating a smooth and structured data collection process. With proper validation, user feedback, and seamless navigation, forms become more effective in providing a great user experience.

**Code:**

**Main.dart**

import 'package:flutter/material.dart';

import 'screens/login\_page.dart';

import 'screens/signup\_page.dart';

void main() {

runApp(const MyApp());

}

class MyApp extends StatelessWidget {

const MyApp({super.key});

@override

Widget build(BuildContext context) {

return MaterialApp(

debugShowCheckedModeBanner: false,

title: 'Login App',

theme: ThemeData(primarySwatch: Colors.blue),

home: const LoginPage(), // Start with LoginPage

routes: {

'/login': (context) => const LoginPage(),

'/signup': (context) => const SignupPage(),

},

);

}

}

**Homepage.dart**

import 'package:flutter/material.dart';

class HomePage extends StatelessWidget {

const HomePage({super.key});

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: const Text('Home Page')),

body: const Center(

child: Text(

"Welcome to Home Page",

style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),

),

),

);

}

}

**Loginpage.dart**

import 'package:flutter/material.dart';

import 'home\_page.dart';

import '../widgets/custom\_textfield.dart';

class LoginPage extends StatefulWidget {

const LoginPage({Key? key}) : super(key: key);

@override

State<LoginPage> createState() => \_LoginPageState();

}

class \_LoginPageState extends State<LoginPage> {

final \_formKey = GlobalKey<FormState>();

final FocusNode \_usernameFocusNode = FocusNode();

final FocusNode \_passwordFocusNode = FocusNode();

@override

void dispose() {

\_usernameFocusNode.dispose();

\_passwordFocusNode.dispose();

super.dispose();

}

@override

Widget build(BuildContext context) {

return Scaffold(

body: Stack(

fit: StackFit.expand,

children: [

// Background Gradient

Container(

decoration: const BoxDecoration(

gradient: LinearGradient(

colors: [Colors.blueAccent, Color(0xFF3B5998)],

begin: Alignment.topLeft,

end: Alignment.bottomRight,

),

),

),

// Content

Center(

child: SingleChildScrollView(

child: Padding(

padding: const EdgeInsets.symmetric(horizontal: 30, vertical: 50),

child: Column(

mainAxisSize: MainAxisSize.min,

children: [

const Text(

"Welcome Back",

style: TextStyle(fontSize: 32, fontWeight: FontWeight.bold, color: Colors.white),

),

const SizedBox(height: 20),

// Form Fields

Form(

key: \_formKey,

child: Column(

children: [

CustomTextField(

labelText: "Username",

icon: Icons.person,

focusNode: \_usernameFocusNode,

validator: (value) {

if (value!.isEmpty) return "Username cannot be empty";

if (value.length < 4) return "Enter a valid name";

return null;

},

),

const SizedBox(height: 20),

CustomTextField(

labelText: "Password",

icon: Icons.lock,

obscureText: true,

focusNode: \_passwordFocusNode,

validator: (value) {

if (value!.isEmpty) return "Password cannot be empty";

if (value.length < 6) return "Password must be at least 6 characters";

return null;

},

),

const SizedBox(height: 30),

// Login Button

ElevatedButton(

onPressed: () {

if (\_formKey.currentState!.validate()) {

Navigator.pushReplacement(

context,

MaterialPageRoute(builder: (context) => const HomePage()),

);

}

},

style: ElevatedButton.styleFrom(

backgroundColor: Colors.blueAccent,

foregroundColor: Colors.white,

shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(12)),

padding: const EdgeInsets.symmetric(vertical: 15, horizontal: 30),

elevation: 10,

),

child: const Text("Login", style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold)),

),

],

),

),

const SizedBox(height: 20),

// SignUp Redirect

Row(

mainAxisAlignment: MainAxisAlignment.center,

children: [

const Text("Don't have an account?", style: TextStyle(color: Colors.white)),

TextButton(

onPressed: () {

// Navigate to SignUp Page

},

child: const Text("Sign Up", style: TextStyle(fontSize: 16, color: Colors.white70)),

),

],

),

],

),

),

),

),

],

), );}}

**Signuppage.dart**

import 'package:flutter/material.dart';

import '../screens/home\_page.dart';

import '../widgets/custom\_text\_field.dart';

import '../widgets/gradient\_background.dart';

class SignupPage extends StatefulWidget {

const SignupPage({Key? key}) : super(key: key);

@override

State<SignupPage> createState() => \_SignupPageState();

}

class \_SignupPageState extends State<SignupPage> {

final \_formKey = GlobalKey<FormState>();

final TextEditingController \_passwordController = TextEditingController();

@override

void dispose() {

\_passwordController.dispose();

super.dispose();

}

@override

Widget build(BuildContext context) {

return Scaffold(

body: Stack(

fit: StackFit.expand,

children: [

const GradientBackground(),

Center(

child: SingleChildScrollView(

padding: const EdgeInsets.symmetric(horizontal: 30),

child: Column(

mainAxisSize: MainAxisSize.min,

children: [

const Text("Create an Account",

style: TextStyle(fontSize: 32, fontWeight: FontWeight.bold, color: Colors.white)),

const SizedBox(height: 20),

Form(

key: \_formKey,

child: Column(

children: [

CustomTextField(label: "Username", icon: Icons.person, obscureText: false),

const SizedBox(height: 20),

CustomTextField(

label: "Password", icon: Icons.lock, obscureText: true, controller: \_passwordController),

const SizedBox(height: 20),

CustomTextField(

label: "Confirm Password",

icon: Icons.lock,

obscureText: true,

validator: (value) {

if (value != \_passwordController.text) return "Passwords do not match";

return null;

},

),

const SizedBox(height: 30),

ElevatedButton(

onPressed: () {

if (\_formKey.currentState!.validate()) {

Navigator.pushReplacement(

context,

MaterialPageRoute(builder: (context) => const HomePage()),

);

}

},

style: ElevatedButton.styleFrom(backgroundColor: Colors.blueAccent),

child: const Text("Sign Up", style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold)),

),

const SizedBox(height: 20),

Row(

mainAxisAlignment: MainAxisAlignment.center,

children: [

const Text("Already have an account?", style: TextStyle(color: Colors.white)),

TextButton(

onPressed: () => Navigator.pop(context),

child: const Text("Login", style: TextStyle(fontSize: 16, color: Colors.white70)),

),

],

),

],

),

),

],

),

),

),

],

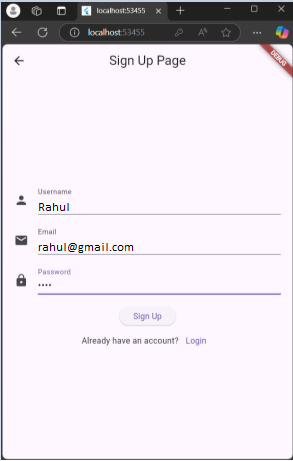
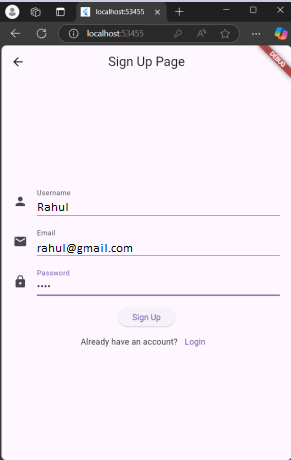
),

);

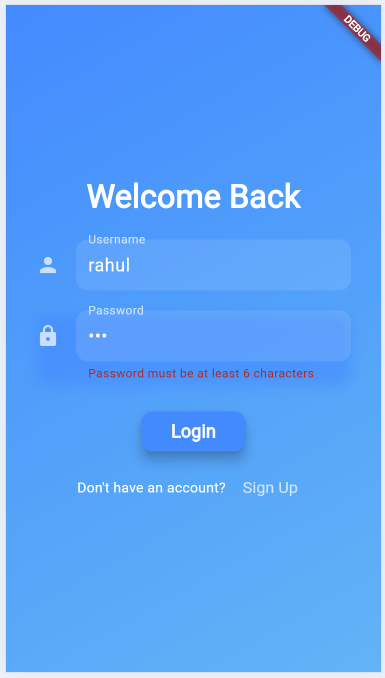
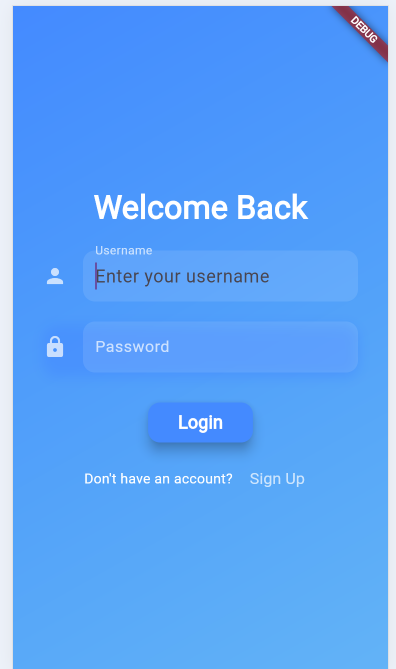
}

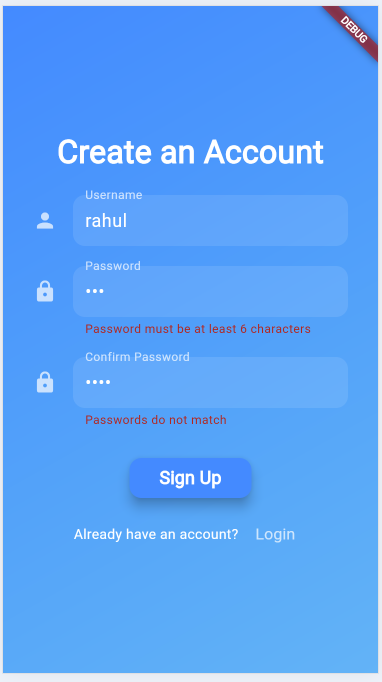
}

Output:



Task Output:





**Conclusion**: We learned that using form widgets and navigation improves user interaction by making forms more dynamic and easy to navigate, enhancing the overall user experience.

**LO’s:** LO2.

**PO’s:** PO1, PO5, PO6, PO8, PO9, PO10, PO12.