**Experiment No:04**

**Aim: To create an interactive Form using form widget**

**Theory:**

In Flutter, the Form widget is a crucial component for building interactive user input forms. It facilitates input validation, data submission, and error handling. Here's a brief overview of creating an interactive form using the Form widget in Flutter:

1. **What is a Form?**
   1. A Form widget is a container that holds multiple form fields, allowing users to input data.
   2. It manages the state of the form and provides methods for validation and submission.
2. **Creating a Form:**
   1. To create a form, wrap your form fields within a Form widget.
   2. Use the GlobalKey<FormState> to uniquely identify the form and access its state.
3. **Form Fields:**
   1. Form fields such as TextFormField, DropdownButtonFormField, etc., are used to collect user input.
   2. Each form field should be provided with a controller (for controlled input) and a validator function to validate user input.
4. **Validation:**
   1. Validation ensures that user input meets specific criteria before submission.
   2. Use the validator property of form fields to specify validation logic.
   3. Validators are functions that return an error message if validation fails, or null if the input is valid.
5. **Submission:**
   1. Submission occurs when the user interacts with a submit button or similar action.
   2. Use the onPressed callback of a button to trigger form submission.
   3. Inside the submission handler, validate the form using the validate method of the FormState.
   4. If the form is valid, proceed with the submission logic (e.g., saving data to a database).
6. **Error Handling:**
   1. If form validation fails, display error messages to the user to guide them in correcting their input.
   2. Error messages can be displayed below each form field or as a general error message at the top of the form.
7. **Cleaning Up:**
   1. Dispose of form controllers and other resources in the dispose method of the State object to prevent memory leaks.
8. **Additional Features:**
   1. Flutter provides various widgets and utilities for enhancing forms, such as InputDecoration for customizing form field appearance, FocusNode for managing focus between fields, and SnackBar for displaying feedback messages.

**Code:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Interactive Form App',

theme: ThemeData(

primarySwatch: Colors.blue,

),

home: MyHomePage(),

);

}

}

class MyHomePage extends StatefulWidget {

@override

\_MyHomePageState createState() => \_MyHomePageState();

}

class \_MyHomePageState extends State<MyHomePage> {

final \_formKey = GlobalKey<FormState>();

TextEditingController \_nameController = TextEditingController();

TextEditingController \_emailController = TextEditingController();

bool \_subscribeToNewsletter = false;

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Interactive Form App'),

),

body: Padding(

padding: const EdgeInsets.all(16.0),

child: Form(

key: \_formKey,

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

TextFormField(

controller: \_nameController,

decoration: InputDecoration(labelText: 'Name'),

validator: (value) {

if (value == null || value.isEmpty) {

return 'Please enter your name';

}

return null;

},

),

SizedBox(height: 16.0),

TextFormField(

controller: \_emailController,

decoration: InputDecoration(labelText: 'Email'),

validator: (value) {

if (value == null || value.isEmpty) {

return 'Please enter your email';

} else if (!RegExp(r'^[\w-]+(\.[\w-]+)\*@[\w-]+(\.[\w-]+)+$')

.hasMatch(value)) {

return 'Please enter a valid email address';

}

return null;

},

),

SizedBox(height: 16.0),

Row(

children: [

Checkbox(

value: \_subscribeToNewsletter,

onChanged: (value) {

setState(() {

\_subscribeToNewsletter = value!;

});

},

),

Text('Subscribe to Newsletter'),

],

),

SizedBox(height: 16.0),

ElevatedButton(

onPressed: () {

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

// Form validation passed, submit the data

\_submitForm();

}

},

child: Text('Submit'),

),

],

),

),

),

);

}

void \_submitForm() {

// Handle form submission logic here

String name = \_nameController.text;

String email = \_emailController.text;

// For demonstration purposes, printing the form data

print('Name: $name');

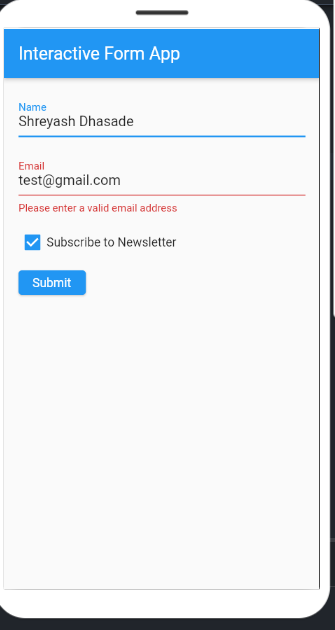
print('Email: $email');

print('Subscribe to Newsletter: $\_subscribeToNewsletter');

}

}

**Output:**

****

**2.Login Form:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Login Form',

theme: ThemeData(

primaryColor: Colors.blue, // Change primary color of the app

appBarTheme: AppBarTheme(

titleTextStyle: TextStyle(

fontSize: 20.0,

fontWeight: FontWeight.bold,

color: Colors.white, // Change text color of the AppBar title

),

),

),

home: LoginForm(),

);

}

}

class LoginForm extends StatefulWidget {

@override

\_LoginFormState createState() => \_LoginFormState();

}

class \_LoginFormState extends State<LoginForm> {

final GlobalKey<FormState> \_formKey = GlobalKey<FormState>();

TextEditingController \_emailController = TextEditingController();

TextEditingController \_passwordController = TextEditingController();

@override

void dispose() {

\_emailController.dispose();

\_passwordController.dispose();

super.dispose();

}

void \_submitForm() {

if (\_formKey.currentState != null && \_formKey.currentState!.validate()) {

// If the form is valid, perform actions like login authentication

ScaffoldMessenger.of(context).showSnackBar(

SnackBar(content: Text('Login Successful!')),

);

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Login Form'),

),

body: Padding(

padding: EdgeInsets.all(16.0),

child: Form(

key: \_formKey,

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

crossAxisAlignment: CrossAxisAlignment.stretch,

children: <Widget>[

ElevatedButton(

onPressed: () {

// Action for Login using Google

},

child: Text('Login using Google'),

),

SizedBox(height: 20),

Text(

'Login Options',

style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),

textAlign: TextAlign.center,

),

SizedBox(height: 20),

TextFormField(

controller: \_emailController,

decoration: InputDecoration(

labelText: 'Email',

border: OutlineInputBorder(),

),

validator: (value) {

if (value == null || value.isEmpty) {

return 'Please enter your email';

}

if (!value.contains('@')) {

return 'Please enter a valid email';

}

return null;

},

),

SizedBox(height: 10),

TextFormField(

controller: \_passwordController,

obscureText: true,

decoration: InputDecoration(

labelText: 'Password',

border: OutlineInputBorder(),

),

validator: (value) {

if (value == null || value.isEmpty) {

return 'Please enter your password';

}

if (value.length < 6) {

return 'Password must be at least 6 characters long';

}

return null;

},

),

SizedBox(height: 20),

ElevatedButton(

onPressed: \_submitForm,

child: Text('Submit'),

),

SizedBox(height: 10),

Row(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Text('Don\'t have an account?'),

TextButton(

onPressed: () {

// Action for Register Button

},

child: Text('Register'),

),

],

),

],

),

),

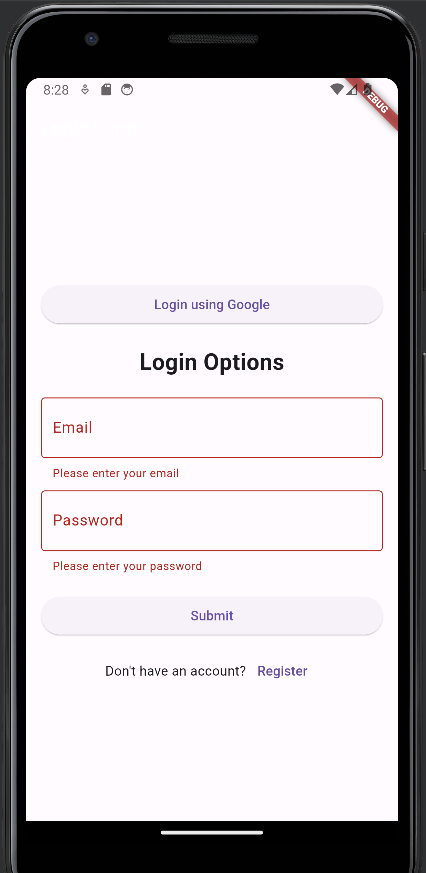
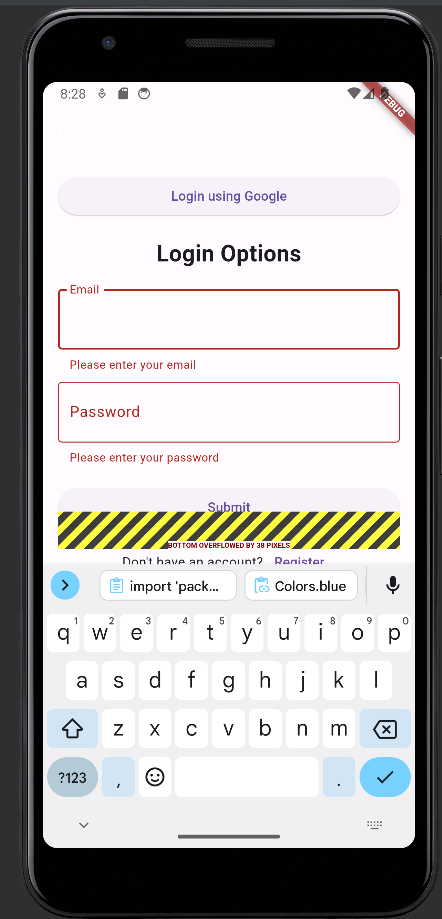
),

);

}

}

**Output:**

****

**Conclusion:**

I have successfully created an interactive Form using form widget in Flutter.