

Swapnil Mulani

D15B-73

EXPERIMENT-4

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?

- a. A Form widget is a container that holds multiple form fields, allowing users to input data.
- b. It manages the state of the form and provides methods for validation and Submission.

2. Creating a Form:

- a. To create a form, wrap your form fields within a Form widget.
- b. Use the GlobalKey<FormState> to uniquely identify the form and access its state.

3. Form Fields:

- a. Form fields such as TextFormField, DropdownButtonFormField, etc., are used to collect user input.
- b. Each form field should be provided with a controller (for controlled input) and a validator function to validate user input.

4. Validation:

- a. Validation ensures that user input meets specific criteria before submission.
- b. Use the validator property of form fields to specify validation logic.
- c. Validators are functions that return an error message if validation fails, or null if the input is valid.

5. Submission:

- a. Submission occurs when the user interacts with a submit button or similar action.
- b. Use the onPressed callback of a button to trigger form submission.
- c. Inside the submission handler, validate the form using the validate method of the FormState.
- d. If the form is valid, proceed with the submission logic (e.g., saving data to a database).

6. Error Handling:

- a. If form validation fails, display error messages to the user to guide them in correcting their input.
- b. Error messages can be displayed below each form field or as a general error message at the top of the form.

7. Cleaning Up:

- a. Dispose of form controllers and other resources in the dispose method of the State object to prevent memory leaks.

8. Additional Features:

- a. 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(  
      home: MySignUpForm(),  
      theme: ThemeData(  
        primaryColor: Color.fromARGB(255, 53, 18, 150),  
        colorScheme: ColorScheme.fromSwatch(primarySwatch: Colors.teal),  
        fontFamily: 'Arial',  
      ),  
    );  
  }  
}
```

```
class MySignUpForm extends StatefulWidget {  
  @override
```

```
_MySignUpFormState createState() => _MySignUpFormState();  
}
```

```
class _MySignUpFormState extends State<MySignUpForm> {  
  final GlobalKey<FormState> _formKey = GlobalKey<FormState>();  
  final TextEditingController _nameController = TextEditingController();  
  String _email = "";  
  String _password = "";
```

```
String? _validateName(String? value) {  
  if (value == null || value.isEmpty) {  
    return 'Please enter your name';  
  }  
  return null;  
}
```

```
String? _validateEmail(String? value) {  
  if (value == null || value.isEmpty) {  
    return 'Please enter your email';  
  } else if (!RegExp(r'^[\w-]+(\.[\w-]+)*@([\w-]+\.)+[a-zA-Z]{2,7}$').  
    .hasMatch(value)) {  
    return 'Please enter a valid email address';  
  }  
  return null;  
}
```

```
String? _validatePassword(String? value) {  
  if (value == null || value.isEmpty) {  
    return 'Please enter your password';  
  } else if (value.length < 8) {  
    return 'Password must be at least 8 characters';  
  } else if (!RegExp(r'^(?=.*?[a-z])(?=.*?[A-Z])(?=.*?[0-9])(?=.*?[!@#\$&*~]).{8,}$').  
    .hasMatch(value)) {  
    return 'Password must contain at least one uppercase letter, lowercase letter, number,  
and special character';  
  }  
  return null;
```

```
}
```

```
void _submitForm() {  
  if (_formKey.currentState?.validate() ?? false) {  
    _formKey.currentState?.save();  
    _showSignUpCompleteDialog(_nameController.text);  
  }  
}
```

```
void _showSignUpCompleteDialog(String name) {  
  showDialog(  
    context: context,  
    builder: (BuildContext context) {  
      return AlertDialog(  
        title: Text('Account creation Complete'),  
        content: Text(  
          'Congratulations, $name! You have successfully created an account.'),  
        actions: <Widget>[  
          TextButton(  
            onPressed: () {  
              Navigator.of(context).pop();  
            },  
            child: Text('OK'),  
          ),  
        ],  
      );  
    },  
  );  
}
```

```
@override
```

```
Widget build(BuildContext context) {  
  return Scaffold(  
    appBar: AppBar(  
      title: Text('Account creation Form'),  
      backgroundColor: Color.fromARGB(255, 177, 70, 226),  
    ),  
  ),  
}
```

```
body: Container(  
  decoration: BoxDecoration(  
    gradient: LinearGradient(  
      colors: [  
        Color.fromRGBO(190, 52, 184, 1),  
        Colors.blue,  
      ],  
      begin: Alignment.topCenter,  
      end: Alignment.bottomCenter,  
    ),  
  ),  
  child: Padding(  
    padding: const EdgeInsets.all(16.0),  
    child: Form(  
      key: _formKey,  
      child: Column(  
        crossAxisAlignment: CrossAxisAlignment.stretch,  
        children: [  
          Image.asset(  
            'assets/images/sneaker_image.jpg',  
            height: 150,  
            fit: BoxFit.cover,  
          ),  
          SizedBox(height: 16),  
          TextFormField(  
            controller: _nameController,  
            decoration: InputDecoration(  
              labelText: 'Name',  
              hintText: 'Enter your name',  
              border: OutlineInputBorder(),  
              prefixIcon: Icon(Icons.person),  
            ),  
            style: TextStyle(  
              fontSize: 16,  
              color: Colors.black87,  
            ),  
            validator: _validateName,
```

```
onSaved: (value) {
  _nameController.text = value ?? "";
},
),
 SizedBox(height: 16),
 TextFormField(
  decoration: InputDecoration(
    labelText: 'Email',
    hintText: 'Enter your email',
    border: OutlineInputBorder(),
    prefixIcon: Icon(Icons.email),
  ),
  style: TextStyle(
    fontSize: 16,
    color: Colors.black87,
  ),
  validator: _validateEmail,
  onSaved: (value) {
    _email = value ?? "";
  },
),
 SizedBox(height: 16),
 TextFormField(
  obscureText: true,
  decoration: InputDecoration(
    labelText: 'Password',
    hintText: 'Enter your password',
    border: OutlineInputBorder(),
    prefixIcon: Icon(Icons.lock),
  ),
  style: TextStyle(
    fontSize: 16,
    color: Colors.black87,
  ),
  validator: _validatePassword,
  onChanged: (value) {
    setState(() {
```

```

        _password = value;
      });
    },
  ),
  SizedBox(height: 8),
  Text(
    _passwordStrength(_password),
    style: TextStyle(
      color: _passwordStrengthColor(_password),
    ),
  ),
  SizedBox(height: 16),
  ElevatedButton(
    onPressed: _submitForm,
    child: Text(
      'Sign Up',
      style: TextStyle(
        fontSize: 18,
        color: Colors.white,
      ),
    ),
  ),
  style: ButtonStyle(
    backgroundColor:
      MaterialStateProperty.all(const Color.fromARGB(255, 107, 0, 150)),
    padding: MaterialStateProperty.all(
      EdgeInsets.symmetric(vertical: 12),
    ),
  ),
],
),
),
),
),
);
}
String _passwordStrength(String password) {

```

```

if (password.isEmpty) {
    return "";
} else if (password.length < 8) {
    return 'Weak password';
} else if (RegExp(r'^[\w-]+(\.[\w-]+)*@([\w-]+\.)+[a-zA-Z]{2,7}$')
    .hasMatch(password)) {
    return 'Weak password';
} else if (!RegExp(r'^(?=.*?[a-z])(?=.*?[A-Z])(?=.*?[0-9])(?=.*?[!@#\$&*~]).{8,}$')
    .hasMatch(password)) {
    return 'Medium password';
} else {
    return 'Strong password';
}
}

```

```

Color _passwordStrengthColor(String password) {
    if (password.isEmpty || password.length < 8) {
        return Colors.red;
    } else if (!RegExp(r'^(?=.*?[a-z])(?=.*?[A-Z])(?=.*?[0-9])(?=.*?[!@#\$&*~]).{8,}$')
        .hasMatch(password)) {
        return Colors.orange;
    } else {
        return Color.fromARGB(255, 166, 68, 196);
    }
}
}

```


OUTPUT:

The image displays two side-by-side screenshots of a Flutter account creation form, along with a confirmation message.

Left Screenshot: The form is titled "Account creation Form" and features the Reddit logo. It contains three input fields: "Name" (placeholder: "Enter your name"), "Email" (placeholder: "Email"), and "Password" (placeholder: "Password"). A "Sign Up" button is at the bottom.

Right Screenshot: The form is filled out with the following data: Name: "SwapMulani", Email: "Swapnil@gmail.com", and Password: "*****". A "Sign Up" button is at the bottom. Below the password field, there is a red text label: "Strong password".

Confirmation Message: A green banner at the bottom right displays the text "Account creation Complete".

CONCLUSION: Hence, we have successfully implemented the Flutter interactive Form using form widget.