Shreyash G Dhekane D15A Roll.no:16

EXPERIMENT NO :- 04

AIM: - To create an interactive Form using a form widget.

Theory: -

A form in Flutter is a structured container that collects user input through various fields like text fields, dropdowns, checkboxes, and buttons. It plays a crucial role in applications that require user data entry, such as login pages, registration forms, and feedback submissions. Flutter provides the Form widget, which works alongside TextFormField and other input elements to manage validation, state handling, and error messages efficiently. By using form validation techniques, developers can ensure data accuracy and enhance user experience.
When you create a form, it is necessary to provide the GlobalKey. This key uniquely identifies the form and allows you to do any validation in the form fields. The form widget uses child widget TextFormField to provide the users to enter the text field. This widget renders a material design text field and also allows us to display validation errors when they occur.

Creation of a Form

1. While creating a form in Flutter, the Form widget is essential as it acts as a container for grouping multiple form fields and managing validation.
2. A GlobalKey<FormState> is required to uniquely identify the form and enable validation or data retrieval from the form fields.
3. The TextFormField widget is used to provide input fields where users can enter data such as names, phone numbers, or email addresses.
4. To enhance the appearance and usability of input fields, InputDecoration is used, allowing customization of labels, icons, borders, and hint text.
5. Validation plays a crucial role in forms, and the validator property within TextFormField ensures user input meets specific criteria before submission.
6. Different types of input require appropriate keyboard types, such as TextInputType.number for numeric fields or TextInputType.emailAddress for email fields.
7. Proper state management is needed to store and retrieve user input, ensuring the form data is processed correctly.
8. A submit button is necessary to trigger form validation and submit the collected data for further processing.

Some Properties of Form Widget

- key: A GlobalKey that uniquely identifies the Form. You can use this key to interact with the form, such as validating, resetting, or saving its state.
- child: The child widget that contains the form fields. Typically, this is a Column, ListView, or another widget that allows you to arrange the form fields vertically.
- autovalidateMode: An enum that specifies when the form should automatically validate its fields.
- Some Methods of Form Widget
- validate(): This method is used to trigger the validation of all the form fields within the Form. It returns true if all fields are valid, otherwise false. You can use it to check the overall validity of the form before submitting it.
- save(): This method is used to save the current values of all form fields. It invokes the onSaved callback for each field. Typically, this method is called after validation succeeds.
- reset(): Resets the form to its initial state, clearing any user-entered data.
- currentState: A getter that returns the current FormState associated with the Form.

Code:

login_screen.dart

```
import 'package:flutter/material.dart';
import
'package:firebase_auth/firebase_auth.dart';
import 'main_screen.dart'; // Import the
MainScreen (not HomeScreen)

class LoginScreen extends StatefulWidget {
  const LoginScreen({super.key});

  @override
  State<LoginScreen> createState() =>
_LoginScreenState();
}

class _LoginScreenState extends
State<LoginScreen> {
  // Controllers for input fields
  final TextEditingController emailController =
TextEditingController();
  final TextEditingController
passwordController = TextEditingController();

  // Firebase Auth instance
  final FirebaseAuth _auth =
FirebaseAuth.instance;

  bool _isLoading = false;

  @override
  void dispose() {
    emailController.dispose();
    passwordController.dispose();
    super.dispose();
  }

  /// Sign in using Firebase Authentication
  Future<void> _login() async {
    setState(() {
      _isLoading = true;
    });

    try {
      // Attempt sign in with email and password
      await
_auth.signInWithEmailAndPassword(
        email: emailController.text.trim(),
        password: passwordController.text,
      );

      // On successful login, navigate to the
MainScreen with BottomNavigationBar.
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) =>
const MainScreen()), // Navigate to
MainScreen
      );
    } on FirebaseAuthException catch (e) {
      String message = 'Login failed';
      if (e.code == 'user-not-found') {
        message = 'No user found for that
email.';
      } else if (e.code == 'wrong-password') {
        message = 'Incorrect password
provided.';
      }

ScaffoldMessenger.of(context).showSnackBar
(
        SnackBar(content: Text(message)),
      );
    } catch (e) {

ScaffoldMessenger.of(context).showSnackBar
(
        const SnackBar(content: Text('An error
occurred.')),
      );
    }

    setState(() {
      _isLoading = false;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.black,
      appBar: AppBar(
        backgroundColor: Colors.black,
```

```
      title: const Text("Log In", style:
TextStyle(color: Colors.white)),
      centerTitle: true,
      actions: [
       // Skip button for direct access (login
later)
        TextButton(
         onPressed: () {
          Navigator.pushReplacement(
           context,
           MaterialPageRoute(builder:
(context) => const MainScreen()), // Navigate
to MainScreen
          );
         },
         child: const Text("Skip", style:
TextStyle(color: Colors.white, fontSize: 16)),
        ),
      ],
      leading: IconButton(
       icon: const Icon(Icons.arrow_back,
color: Colors.white),
       onPressed: () {
        Navigator.pop(context);
       },
      ),
     ),
     body: Padding(
      padding: const
EdgeInsets.symmetric(horizontal: 20),
      child: Column(
       children: [
        const SizedBox(height: 30),
        // Email TextField
        TextField(
         controller: emailController,
         style: const TextStyle(color:
Colors.white),
         decoration: InputDecoration(
          labelText: "Email Address",
          labelStyle: const TextStyle(color:
Colors.white70),
          enabledBorder: OutlineInputBorder(
           borderSide: const
BorderSide(color: Colors.white38),
           borderRadius:
BorderRadius.circular(8),
          ),
          focusedBorder: OutlineInputBorder(
           borderSide: const
BorderSide(color: Colors.blue),
           borderRadius:
BorderRadius.circular(8),
          ),
         ),
        ),
        const SizedBox(height: 20),
        // Password TextField
        TextField(
         controller: passwordController,
         obscureText: true,
         style: const TextStyle(color:
Colors.white),
         decoration: InputDecoration(
          labelText: "Password",
          labelStyle: const TextStyle(color:
Colors.white70),
          enabledBorder: OutlineInputBorder(
           borderSide: const
BorderSide(color: Colors.white38),
           borderRadius:
BorderRadius.circular(8),
          ),
          focusedBorder: OutlineInputBorder(
           borderSide: const
BorderSide(color: Colors.blue),
           borderRadius:
BorderRadius.circular(8),
          ),
         ),
        ),
        const SizedBox(height: 20),
        // Login button or a loading indicator
        _isLoading
          ? const CircularProgressIndicator()
          : ElevatedButton(
         style: ElevatedButton.styleFrom(
          backgroundColor: Colors.blue,
          minimumSize: const
Size(double.infinity, 50),
         ),
         onPressed: _login,
         child: const Text("Log In", style:
TextStyle(color: Colors.white)),
        ),
        const SizedBox(height: 10),
        TextButton(
         onPressed: () {
```

```
            // Implement forgot password logic if
desired.
          },
          child: const Text("Forgot password?",
style: TextStyle(color: Colors.blue)),
        ),
        const SizedBox(height: 20),
        const Row(
          children: [
            Expanded(child: Divider(color:
Colors.white)),
            Padding(
              padding:
EdgeInsets.symmetric(horizontal: 10),
              child: Text("OR", style:
TextStyle(color: Colors.white)),
            ),
            Expanded(child: Divider(color:
Colors.white)),
          ],
        ),
        // Additional social login buttons can
be added here
        const Spacer(),
        // Navigation to registration screen
        TextButton(
          onPressed: () {
            Navigator.pushNamed(context,
'/register');
          },
          child: const Text("Don't have an
account? Register", style: TextStyle(color:
Colors.blue, fontSize: 16)),
        ),
        const SizedBox(height: 20),
      ],
    ),
  ),
 );
 }
}
```

registration_screen.dart

```
import 'package:flutter/material.dart';
import
'package:firebase_auth/firebase_auth.dart';

class RegistrationScreen extends
StatefulWidget {
 const RegistrationScreen({super.key});

 @override
 State<RegistrationScreen> createState() =>
_RegistrationScreenState();
}

class _RegistrationScreenState extends
State<RegistrationScreen> {
 // Controllers for input fields
 final TextEditingController nameController =
TextEditingController();
 final TextEditingController emailController =
TextEditingController();
 final TextEditingController
passwordController = TextEditingController();
 final TextEditingController
confirmPasswordController =
TextEditingController();

 // Firebase Auth instance
 final FirebaseAuth _auth =
FirebaseAuth.instance;

 bool _isLoading = false;

 @override
 void dispose() {
  nameController.dispose();
  emailController.dispose();
  passwordController.dispose();
  confirmPasswordController.dispose();
  super.dispose();
 }

 /// Register a new user with Firebase
Authentication
 Future<void> _register() async {
  final email = emailController.text.trim();
  final password = passwordController.text;
```

```
    final confirmPassword =
confirmPasswordController.text;

    if (password != confirmPassword) {

ScaffoldMessenger.of(context).showSnackBar
(
        const SnackBar(content:
Text("Passwords do not match")),
      );
      return;
    }

    setState(() {
      _isLoading = true;
    });

    try {
      // Create the user
      UserCredential userCredential = await
_auth.createUserWithEmailAndPassword(
        email: email,
        password: password,
      );

      // Optionally update the display name
      if (userCredential.user != null &&
nameController.text.trim().isNotEmpty) {
        await
userCredential.user!.updateDisplayName(na
meController.text.trim());
      }

      // On successful registration, navigate to
main screen.

Navigator.pushReplacementNamed(context,
'/main');
    } on FirebaseAuthException catch (e) {
      String message = 'Registration failed';
      if (e.code == 'weak-password') {
        message = 'The password provided is
too weak.';
      } else if (e.code == 'email-already-in-use')
{
        message = 'An account already exists for
that email.';
      }

ScaffoldMessenger.of(context).showSnackBar
(
        SnackBar(content: Text(message)),
      );
    } catch (e) {

ScaffoldMessenger.of(context).showSnackBar
(
        const SnackBar(content: Text("An error
occurred")),
      );
    }

    setState(() {
      _isLoading = false;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.black,
      appBar: AppBar(
        backgroundColor: Colors.black,
        title: const Text("Register", style:
TextStyle(color: Colors.white)),
        centerTitle: true,
        leading: IconButton(
          icon: const Icon(Icons.arrow_back,
color: Colors.white),
          onPressed: () {
            Navigator.pop(context);
          },
        ),
      ),
      body: SingleChildScrollView(
        padding: const
EdgeInsets.symmetric(horizontal: 20, vertical:
20),
        child: Column(
          children: [
            const SizedBox(height: 20),
            // Full Name field (optional)
            TextField(
              controller: nameController,
              style: const TextStyle(color:
Colors.white),
              decoration: InputDecoration(
                labelText: "Full Name",
```

```
            labelStyle: const TextStyle(color:
Colors.white70),
            enabledBorder: OutlineInputBorder(
              borderSide: const
BorderSide(color: Colors.white38),
              borderRadius:
BorderRadius.circular(8),
            ),
            focusedBorder: OutlineInputBorder(
              borderSide: const
BorderSide(color: Colors.blue),
              borderRadius:
BorderRadius.circular(8),
            ),
          ),
        ),
        const SizedBox(height: 20),
        // Email Address field
        TextField(
          controller: emailController,
          style: const TextStyle(color:
Colors.white),
          decoration: InputDecoration(
            labelText: "Email Address",
            labelStyle: const TextStyle(color:
Colors.white70),
            enabledBorder: OutlineInputBorder(
              borderSide: const
BorderSide(color: Colors.white38),
              borderRadius:
BorderRadius.circular(8),
            ),
            focusedBorder: OutlineInputBorder(
              borderSide: const
BorderSide(color: Colors.blue),
              borderRadius:
BorderRadius.circular(8),
            ),
          ),
        ),
        const SizedBox(height: 20),
        // Password field
        TextField(
          controller: passwordController,
          obscureText: true,
          style: const TextStyle(color:
Colors.white),
          decoration: InputDecoration(
            labelText: "Password",
            labelStyle: const TextStyle(color:
Colors.white70),
            enabledBorder: OutlineInputBorder(
              borderSide: const
BorderSide(color: Colors.white38),
              borderRadius:
BorderRadius.circular(8),
            ),
            focusedBorder: OutlineInputBorder(
              borderSide: const
BorderSide(color: Colors.blue),
              borderRadius:
BorderRadius.circular(8),
            ),
          ),
        ),
        const SizedBox(height: 20),
        // Confirm Password field
        TextField(
          controller:
confirmPasswordController,
          obscureText: true,
          style: const TextStyle(color:
Colors.white),
          decoration: InputDecoration(
            labelText: "Confirm Password",
            labelStyle: const TextStyle(color:
Colors.white70),
            enabledBorder: OutlineInputBorder(
              borderSide: const
BorderSide(color: Colors.white38),
              borderRadius:
BorderRadius.circular(8),
            ),
            focusedBorder: OutlineInputBorder(
              borderSide: const
BorderSide(color: Colors.blue),
              borderRadius:
BorderRadius.circular(8),
            ),
          ),
        ),
        const SizedBox(height: 20),
        _isLoading
            ? const CircularProgressIndicator()
            : ElevatedButton(
          style: ElevatedButton.styleFrom(
            backgroundColor: Colors.blue,
            minimumSize: const
Size(double.infinity, 50),
```

```
            ),
            onPressed: _register,
            child: const Text("Register", style:
TextStyle(color: Colors.white)),
          ),
          const SizedBox(height: 10),
                // Optionally add a link to the login
page if already have an account.
              ],
            ),
          ),
        );
      }
    }
```

Screenshots: