Experiment 4

Aim: To create an interactive Form using a form widget.

Theory: Form widget is used to group multiple form fields together, like TextFormFields, and manage their validation and submission. Here’s how you can use a Form with a TextFormField that includes validation logic, and a button to validate and submit the form.

Key Concepts in Interactive Forms

1. **Form Widgets** – These are components like text fields, checkboxes, radio buttons, dropdowns, and buttons that allow users to input data.
2. **Validation** – Ensuring that user input meets specific criteria (e.g., email format, required fields).
3. **State Management** – Forms can store data dynamically and update based on user interactions.
4. **Event Handling** – Capturing user actions like clicks, typing, or selections to trigger specific responses.
5. **UI/UX Considerations** – Forms should be visually appealing, easy to navigate, and provide clear feedback.

Implementation Approach

- Create a StatefulWidget to manage form state.
- Define a GlobalKey<FormState> to uniquely identify and manage the form.
- Wrap form fields inside a Form widget using the GlobalKey.
- Add TextFormField widgets inside the Form.
- Provide a validator function for each TextFormField to define validation rules.
- Add a submit button (e.g., ElevatedButton) below the form fields.
- On button press, call _formKey.currentState!.validate() to check if all fields are valid.
- If validation passes, proceed with form submission logic (e.g., show message, save data, navigate).
- (Optional) Use TextEditingControllers to retrieve or manipulate input data.

Code in event.dart:

textfield.dart

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

class CustomTextField extends StatelessWidget {
  final String text;
  final TextEditingController? controller;
  final bool obscureText;

  const CustomTextField({
    Key? key,
    required this.text,
    this.controller,
    this.obscureText = false,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container(
      decoration: BoxDecoration(
        borderRadius: BorderRadius.circular(15.0),
        color: Colors.white,
      ),
      child: TextField(
        controller: controller,
        obscureText: obscureText,
        decoration: InputDecoration(
          hintText: text,
          hintStyle: GoogleFonts.poppins(
            color: Colors.black,
            fontSize: 12,
          ),
          border: OutlineInputBorder(
            borderRadius: BorderRadius.circular(15.0),
          ),
          contentPadding: const EdgeInsets.symmetric(
            vertical: 12.0,
            horizontal: 16.0,
          ),
          enabledBorder: OutlineInputBorder(
            borderSide: BorderSide(color: Colors.black12),
            borderRadius: BorderRadius.circular(15),
          ),
          focusedBorder: OutlineInputBorder(
            borderRadius: const BorderRadius.all(
              Radius.circular(15),
            ),
            borderSide: BorderSide(
              color: Colors.black12,
            ),
          ),
        ),
      ),
    );
  }
}
```

signup.dart

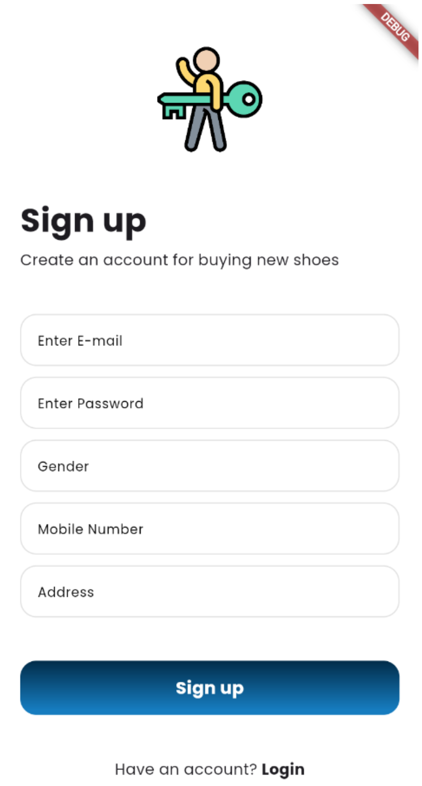
```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:skillxchange/auth/login.dart';
import 'package:skillxchange/common/button.dart';
import 'package:skillxchange/common/textfield.dart';
import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';

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

  @override
  State<Signup> createState() => _SignupState();
}

class _SignupState extends State<Signup> {
```

```
final TextEditingController emailController = TextEditingController();
final TextEditingController passwordController = TextEditingController();
final TextEditingController genderController = TextEditingController();
final TextEditingController mobileController = TextEditingController();
final TextEditingController addressController = TextEditingController();
bool isLoading = false;

void signUpUser() async {
  setState(() {
    isLoading = true;
  });

  try {
    // Firebase authentication
    UserCredential userCredential =
        await FirebaseAuth.instance.createUserWithEmailAndPassword(
      email: emailController.text.trim(),
      password: passwordController.text.trim(),
    );

    // Store additional user details in Firestore
    await FirebaseFirestore.instance
        .collection('users')
        .doc(userCredential.user!.uid)
        .set({
      "email": emailController.text.trim(),
      "gender": genderController.text.trim(),
      "mobile": mobileController.text.trim(),
      "address": addressController.text.trim(),
      "createdAt": DateTime.now(),
    });

    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(content: Text("Signup Successful!")),
    );

    // Navigate to login screen
    Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => const Login()),
      );
    } on FirebaseAuthException catch (e) {
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(content: Text(e.message ?? "Signup failed")),
      );
    }

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      body: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.all(20.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              const SizedBox(height: 100),
              Row(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  SizedBox(
                    height: 100,
                    width: 100,
                    child: Center(child: Image.asset("assets/key.png")),
                  ),
                ],
              ),
              const SizedBox(height: 40),
              Text(
                "Sign up",
                style: GoogleFonts.poppins(
                  fontSize: 30,
                  fontWeight: FontWeight.bold,
                ),
              ),
              const SizedBox(height: 5),
              Text(
                "Create an account for buying new shoes",
                style: GoogleFonts.poppins(),
              ),
              const SizedBox(height: 40),

              CustomTextField(
                text: "Enter E-mail",
                controller: emailController,
              ),
              const SizedBox(height: 10),

              CustomTextField(
                text: "Enter Password",
                controller: passwordController,
                obscureText: true,
              ),
              const SizedBox(height: 10),

              CustomTextField(
                text: "Gender",
                controller: genderController,
              ),
              const SizedBox(height: 10),

              CustomTextField(
                text: "Mobile Number",
                controller: mobileController,
              ),
              const SizedBox(height: 10),

              CustomTextField(
                      text: "Address",
                      controller: addressController,
                     ),
                     const SizedBox(height: 40),

                     isLoading
                       ? const Center(child: CircularProgressIndicator())
                       : CustomButton(
                         onTap: signUpUser,
                         text: "Sign up",
                        ),
                     const SizedBox(height: 40),

                     InkWell(
                      onTap: () {
                       Navigator.pushReplacement(
                        context,
                        MaterialPageRoute(builder: (context) => const Login()),
                       );
                      },
                      child: Row(
                       mainAxisAlignment: MainAxisAlignment.center,
                       children: [
                        Text(
                         "Have an account? ",
                         style: GoogleFonts.poppins(),
                        ),
                        Text(
                         "Login",
                         style: GoogleFonts.poppins(fontWeight: FontWeight.bold),
                        ),
                       ],
                      ),
                     ),
                    ],
                   ),
                  ),
                 ),
                  );
                }
              }
```

On clicking on Submit:

```
void signUpUser() async {
    setState(() {
      isLoading = true;
    });

    try {
      // Firebase authentication
      UserCredential userCredential =
          await FirebaseAuth.instance.createUserWithEmailAndPassword(
        email: emailController.text.trim(),
        password: passwordController.text.trim(),
      );

      // Store additional user details in Firestore
      await FirebaseFirestore.instance
          .collection('users')
          .doc(userCredential.user!.uid)
          .set({
        "email": emailController.text.trim(),
        "gender": genderController.text.trim(),
        "mobile": mobileController.text.trim(),
        "address": addressController.text.trim(),
        "createdAt": DateTime.now(),
      });

      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text("Signup Successful!")),
      );

      // Navigate to login screen
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => const Login()),
      );
    } on FirebaseAuthException catch (e) {
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(content: Text(e.message ?? "Signup failed")),
      );
    }

    setState(() {
      isLoading = false;
    });
  }
```

Output:



Sign up

Create an account for buying new shoes

Have an account? **Login**

Github Link: https://github.com/brijeshforcollege/flutter

Conclusion

Implementing form validation in Flutter using Form and TextFormField provides a structured and efficient way to collect and validate user input. By leveraging the GlobalKey<FormState> and validator functions, you can ensure that the data entered meets your requirements before processing it. This approach helps enhance app reliability, user experience, and data accuracy in a clean and maintainable way.