Name : Naikwadi Yash Shivdas

MPL Practical 04

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

Theory:

Forms are an essential part of mobile applications as they allow users to input and submit data. In Flutter, the Form widget provides a structured way to handle user input, validation, and submission. It works along with TextFormField to create fields for user interaction, such as entering text, passwords, or email addresses. Forms also use a GlobalKey<FormState> to manage validation and state.

In this experiment, we implemented a login and signup form using the Form widget. The login form includes fields for email and password, while the signup form additionally has name and confirm password fields. We have applied validation to ensure that:

- The email field accepts only valid email formats.
- The password field requires at least six characters.
- The confirm password field ensures both passwords match.

Navigation is handled properly so that once a user logs in or signs up successfully, they cannot go back to the login screen using the back button. This is achieved using Navigator.pushAndRemoveUntil().

Code:

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

void main() {
 runApp(const GreenFundConnect());
}

class GreenFundConnect extends StatelessWidget {
 const GreenFundConnect({Key? key}) : super(key: key);

 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   debugShowCheckedModeBanner: false,
   title: 'GreenFund Connect',
   theme: ThemeData(
    primarySwatch: Colors.green,
    primaryColor: const Color(0xFF2E7D32), // Dark green
    secondaryHeaderColor: const Color(0xFF81C784), // Light green
    fontFamily: GoogleFonts.poppins().fontFamily,
    textTheme: TextTheme(
     headlineLarge: GoogleFonts.montserrat(
```

```
          fontWeight: FontWeight.bold,
          color: const Color(0xFF2E7D32),
        ),
        titleLarge: GoogleFonts.poppins(
          fontWeight: FontWeight.w600,
          color: Colors.black87,
        ),
        bodyLarge: GoogleFonts.poppins(
          color: Colors.black87,
        ),
        bodyMedium: GoogleFonts.poppins(
          color: Colors.black87,
        ),
      ),
      elevatedButtonTheme: ElevatedButtonThemeData(
        style: ElevatedButton.styleFrom(
          backgroundColor: const Color(0xFF2E7D32),
          foregroundColor: Colors.white,
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(10),
          ),
          padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 12),
        ),
      ),
    ),
    home: const LoginScreen(),
  );
 }
}

class LoginScreen extends StatelessWidget {
 const LoginScreen({Key? key}) : super(key: key);

 @override
 Widget build(BuildContext context) {
  final _formKey = GlobalKey<FormState>();
  final TextEditingController emailController = TextEditingController();
  final TextEditingController passwordController = TextEditingController();

  return Scaffold(
   appBar: AppBar(title: const Text('Login')),
   body: Padding(
```

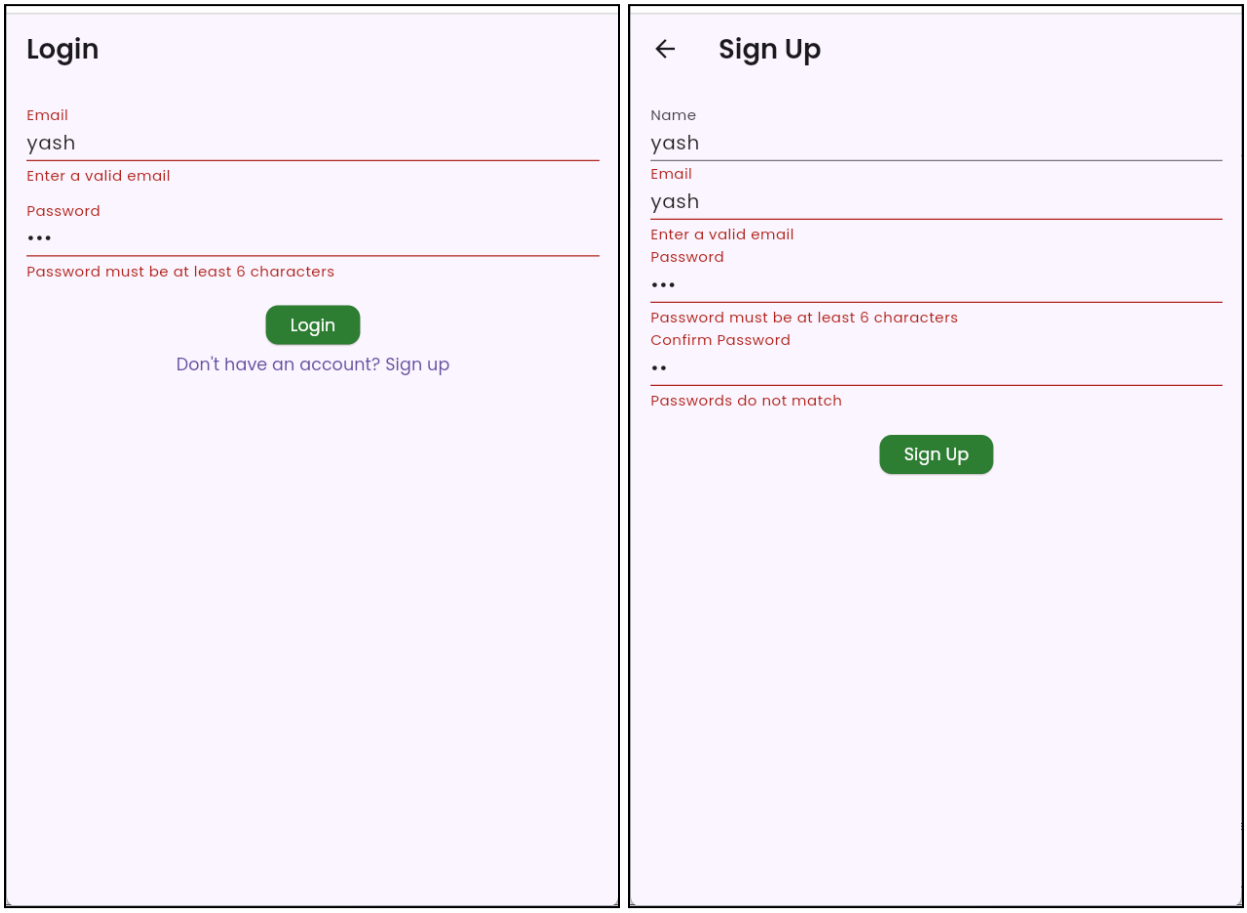
```
padding: const EdgeInsets.all(16.0),
child: Form(
 key: _formKey,
 child: Column(
  children: [
   TextFormField(
    controller: emailController,
    decoration: const InputDecoration(labelText: 'Email'),
    keyboardType: TextInputType.emailAddress,
    validator: (value) {
     if (value == null || value.isEmpty) {
      return 'Please enter an email';
     } else if (!RegExp(r'^[^@]+@[^@]+\.[^@]+').hasMatch(value)) {
      return 'Enter a valid email';
     }
     return null;
    },
   ),
   const SizedBox(height: 10),
   TextFormField(
    controller: passwordController,
    decoration: const InputDecoration(labelText: 'Password'),
    obscureText: true,
    validator: (value) {
     if (value == null || value.length < 6) {
      return 'Password must be at least 6 characters';
     }
     return null;
    },
   ),
   const SizedBox(height: 20),
   ElevatedButton(
    onPressed: () {
     if (_formKey.currentState!.validate()) {
      Navigator.push(
       context,
       MaterialPageRoute(builder: (context) => const HomeScreen()),
      );
     }
    },
    child: const Text('Login'),
   ),
```

```
            TextButton(
              onPressed: () {
                Navigator.push(
                  context,
                  MaterialPageRoute(builder: (context) => const SignupScreen()),
                );
              },
              child: const Text('Don\'t have an account? Sign up'),
            ),
          ],
        ),
      ),
    ),
  );
 }
}

class SignupScreen extends StatelessWidget {
 const SignupScreen({Key? key}) : super(key: key);

 @override
 Widget build(BuildContext context) {
  final _formKey = GlobalKey<FormState>();
  final TextEditingController nameController = TextEditingController();
  final TextEditingController emailController = TextEditingController();
  final TextEditingController passwordController = TextEditingController();
  final TextEditingController confirmPasswordController = TextEditingController();

  return Scaffold(
   appBar: AppBar(title: const Text('Sign Up')),
   body: Padding(
    padding: const EdgeInsets.all(16.0),
    child: Form(
     key: _formKey,
     child: Column(
      children: [
       TextFormField(
        controller: nameController,
        decoration: const InputDecoration(labelText: 'Name'),
        validator: (value) => value!.isEmpty ? 'Enter your name' : null,
       ),
       TextFormField(
```

```
  controller: emailController,
  decoration: const InputDecoration(labelText: 'Email'),
  keyboardType: TextInputType.emailAddress,
  validator: (value) {
   if (value == null || value.isEmpty) {
    return 'Please enter an email';
   } else if (!RegExp(r'^[^@]+@[^@]+\.[^@]+').hasMatch(value)) {
    return 'Enter a valid email';
   }
   return null;
  },
 ),
 TextFormField(
  controller: passwordController,
  decoration: const InputDecoration(labelText: 'Password'),
  obscureText: true,
  validator: (value) {
   if (value == null || value.length < 6) {
    return 'Password must be at least 6 characters';
   }
   return null;
  },
 ),
 TextFormField(
  controller: confirmPasswordController,
  decoration: const InputDecoration(labelText: 'Confirm Password'),
  obscureText: true,
  validator: (value) {
   if (value != passwordController.text) {
    return 'Passwords do not match';
   }
   return null;
  },
 ),
 const SizedBox(height: 20),
 ElevatedButton(
  onPressed: () {
   if (_formKey.currentState!.validate()) {
    Navigator.push(
     context,
     MaterialPageRoute(builder: (context) => const HomeScreen()),
    );
```

```
              }
            },
            child: const Text('Sign Up'),
          ),
        ],
      ),
    ),
  ),
 );
 }
}
```

Screenshot:



Conclusion

In this experiment, we successfully implemented interactive login and signup forms with proper validation, ensuring correct user input. Initially, we faced an issue where users could navigate back to the login screen after logging in, but we resolved it by using Navigator.pushAndRemoveUntil() to prevent back navigation.