Darash Mishra D15B 39

Experiment 04

Aim: create an interactive form using form widgets in flutter.

Theory:
Flutter provides a comprehensive set of widgets for building forms, allowing developers to create interactive user interfaces for collecting user input. These form widgets streamline the process of handling user input validation, submission, and data processing. Form Widget: The Form widget is the foundation for creating forms in Flutter. It acts as a container for form fields and provides methods for form validation and submission.
The Form widget manages the form state internally and provides access to the FormState object, which can be used to interact with the form fields.

- TextFormField Widget: The TextFormField widget is used to create text input fields within a form. It provides various properties for customizing the appearance and behavior of the input field, such as decoration, validation, and input formatting. Developers can specify validators to enforce input constraints and error messages to provide feedback to users when input validation fails.
- Form Validation: Flutter provides built-in support for form validation using the validator property of form fields. Developers can define validator functions that evaluate the input value and return an error message if the input does not meet the specified criteria. The Form widget automatically triggers validation when the form is submitted, and displays error messages for invalid fields.
- Form Submission: Form submission in Flutter involves handling user input after the form is validated. Developers typically use the onPressed callback of a submit button to trigger form submission. Within the submit callback, developers can access the current state of the form using the FormState object and retrieve the values of individual form fields for further processing, such as data submission to a server or local storage.
- Feedback and Error Handling: Providing feedback to users during form interaction is crucial for a positive user experience. Flutter allows developers to display error messages and visual indicators to guide users when input validation fails. Error messages can be displayed inline with form fields or in a separate section of the form, depending on the design requirements. Additionally, developers can use dialogs or snack bars to provide feedback upon successful form submission or error handling. By leveraging these form widgets and techniques, developers can create intuitive and responsive forms in Flutter applications, enabling seamless interaction with users and efficient data collection and processing.

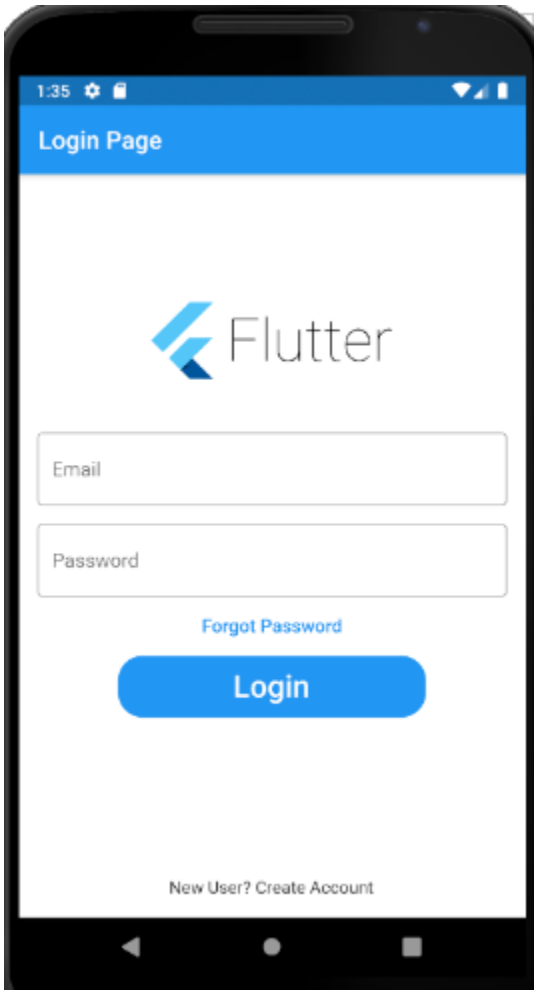
Code:

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

void main() {
  runApp(MyApp());
}
```

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Form Example',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: MyForm(),
    );
  }
}

class MyForm extends StatefulWidget {
  @override
  _MyFormState createState() => _MyFormState();
}

class _MyFormState extends State<MyForm> {
  final _formKey = GlobalKey<FormState>();
  String _name = '';
  String _email = '';

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Flutter Form Example'),
      ),
      body: Padding(
        padding: EdgeInsets.all(16.0),
        child: Form(
          key: _formKey,
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: <Widget>[
              TextFormField(
                decoration: InputDecoration(
                  labelText: 'Name',
                ),
                validator: (value) {
                  if (value == null || value.isEmpty) {
                    return 'Please enter your name';
```

```
            }
            return null;
          },
          onSaved: (value) {
            _name = value!;
          },
        ),
        TextFormField(
          decoration: InputDecoration(
            labelText: 'Email',
          ),
          validator: (value) {
            if (value == null || value.isEmpty) {
              return 'Please enter your email';
            }
            return null;
          },
          onSaved: (value) {
            _email = value!;
          },
        ),
        Padding(
          padding: EdgeInsets.symmetric(vertical: 16.0),
          child: ElevatedButton(
            onPressed: () {
              if (_formKey.currentState!.validate()) {
                _formKey.currentState!.save();
                ScaffoldMessenger.of(context).showSnackBar(
                  SnackBar(
                    content: Text('Form submitted!'),
                  ),
                );
                // Here you can do something with the form data like sending it to a server
                print('Name: $_name');
                print('Email: $_email');
              }
            },
            child: Text('Submit'),
          ),
        ),
      ],
    ),
  ),
),
```

```
    );
  }
}
```


1:35
Login Page
Flutter
Email
Password
Forgot Password
Login
New User? Create Account