Nishtha Joshi
D15B 25
MPL Experiment 4

Aim: To create an interactive form in Flutter using form widget

Theory :
Introduction to Forms in Flutter:

In Flutter, forms are used to collect user input in a structured manner. Forms typically consist of various input fields such as text fields, checkboxes, radio buttons, dropdowns, etc., allowing users to input data and submit it. The `Form` widget is the fundamental component for building forms in Flutter.

Key Components of Forms:

1. Form Widget: The `Form` widget is the container for form elements. It maintains the form state and provides methods to manage form submission, validation, and data retrieval. It is typically used as the root widget of a form.

2. TextFormField Widget: The `TextFormField` widget is used to collect text input from users. It provides features for input validation, input formatting, and handling user interactions such as focus changes and text changes.

3. FormField Widget: The `FormField` widget is the base class for form fields in Flutter. It provides a generic way to manage the state and validation of form fields. Widgets like `TextFormField`, `CheckboxFormField`, `RadioFormField`, etc., are subclasses of `FormField`.

4. FormState: The `FormState` class represents the mutable state of a `Form` widget. It provides methods to interact with form fields, validate form data, and manage form submission.

Building an Interactive Form:

To create an interactive form in Flutter, follow these general steps:

1. Define Form Fields: Determine the types of input fields required for your form (e.g., text fields, checkboxes, radio buttons) and add corresponding form field widgets inside the `Form` widget.

2. Handle Form Submission: Implement a callback function that handles form submission. This function typically validates the form data, processes it, and may perform actions like saving data to a database or sending it to a server.

3. Validate Input: Implement validation logic for each form field to ensure that the entered data meets the required criteria. Use the `validator` property of form field widgets to specify validation functions.

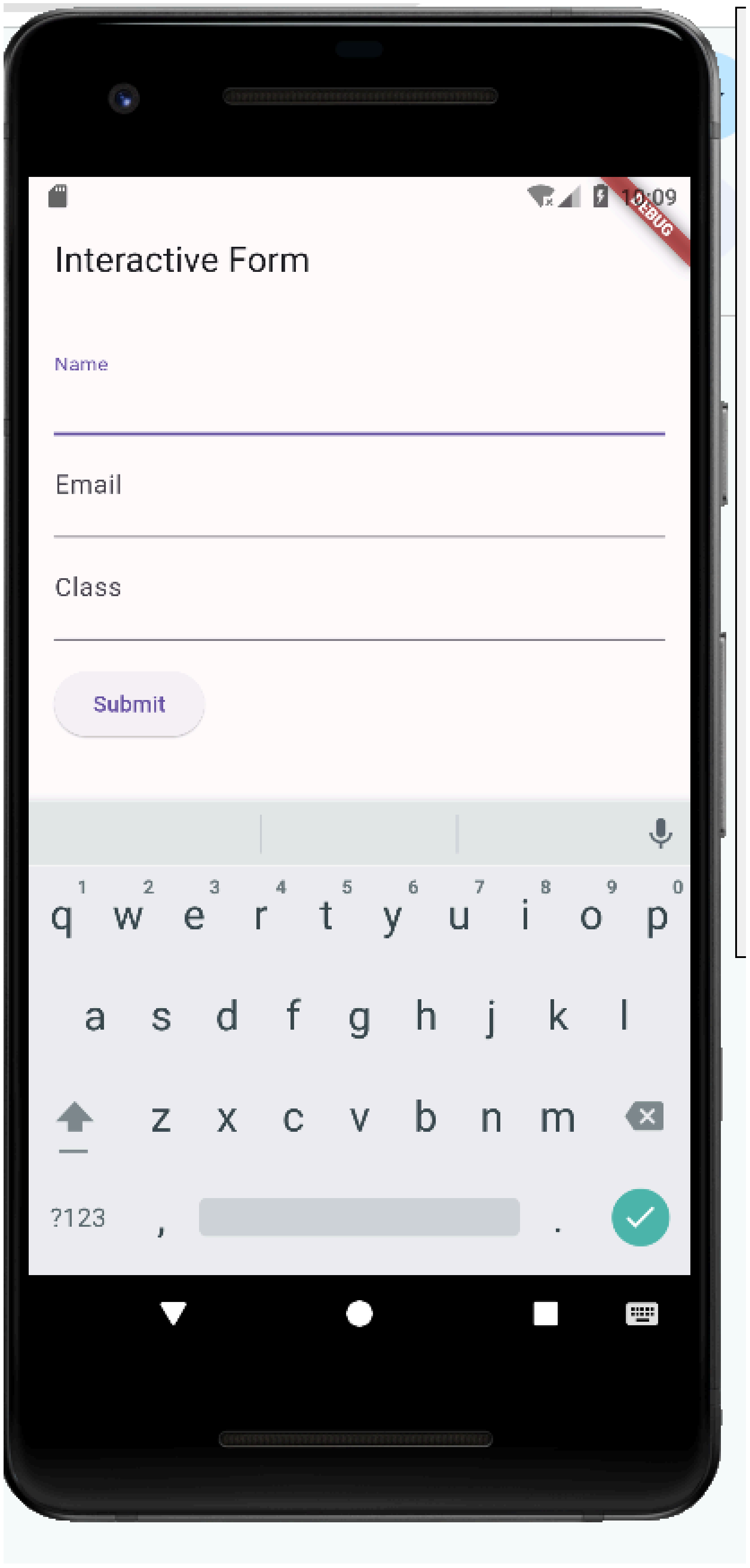
4. Handle User Interaction: Use various event handlers provided by form field widgets to handle user interactions such as text changes, focus changes, checkbox selections, etc.

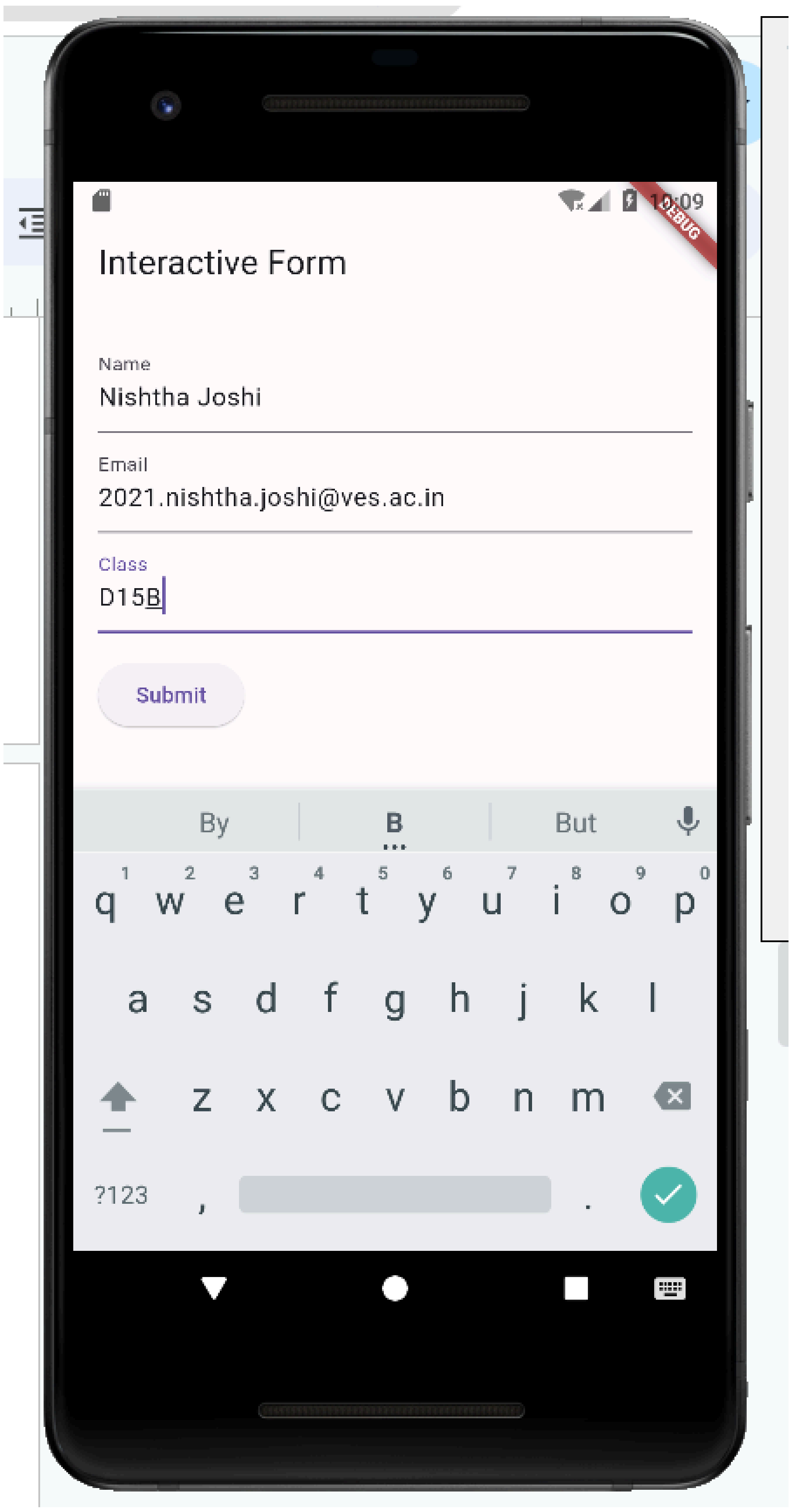
5. Update Form State: Manage the state of the form and form fields using the `FormState` object. Use methods like `validate()`, `save()`, and `reset()` to interact with form fields and trigger form validation, data saving, and resetting.

Best Practices:

- Error Handling: Provide meaningful error messages to users when input validation fails.
- Accessibility: Ensure that your form is accessible to all users, including those using assistive technologies.
- Responsive Design: Design your form to adapt to different screen sizes and orientations.
- Testing: Thoroughly test your form to ensure it works as expected in different scenarios, including edge cases and error conditions.

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Interactive Form',
      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 = '';
  String _rollNumber = '';

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Interactive Form'),
      ),
      body: Padding(
        padding: const 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!;
                },
              ),
              TextFormField(
                decoration: InputDecoration(
                  labelText: 'Class ',
                ),
                validator: (value) {
                  if (value == null || value.isEmpty) {
                    return 'Please enter your class ';
                  }
                  return null;
                },
                onSaved: (value) {
                  _rollNumber = value!;
                },
              ),
              SizedBox(height: 16),
              ElevatedButton(
                onPressed: () {
                  if (_formKey.currentState!.validate()) {
                    _formKey.currentState!.save();

                    print('Name: $_name, Email: $_email, Roll Number:
$_rollNumber');

                  }
                },
                child: Text('Submit'),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```


10:09
DEBUG
Interactive Form
Name
Email
Class
Submit
?123



Interactive Form

Name
Nishtha Joshi

Email
2021.nishtha.joshi@ves.ac.in

Class
D15B

Submit