EXPERIMENT 3

Aim:

To create a simple interactive form in Flutter, including form validation
and data processing on submission. The form includes text input fields, a dropdown menu, and a checkbox.

Theory:

In Flutter, forms are essential components for collecting and validating user input within applications. The `Form` widget serves as the foundation for building interactive and responsive forms. It encapsulates various form fields, such as text input, checkboxes, and dropdowns, enabling developers to create a cohesive user experience. The `Form` widget also facilitates efficient form validation, ensuring that user inputs meet specified criteria before processing. Flutter's reactive paradigm allows developers to seamlessly manage and update the UI based on user interactions within the form, promoting a robust and flexible approach to form design.

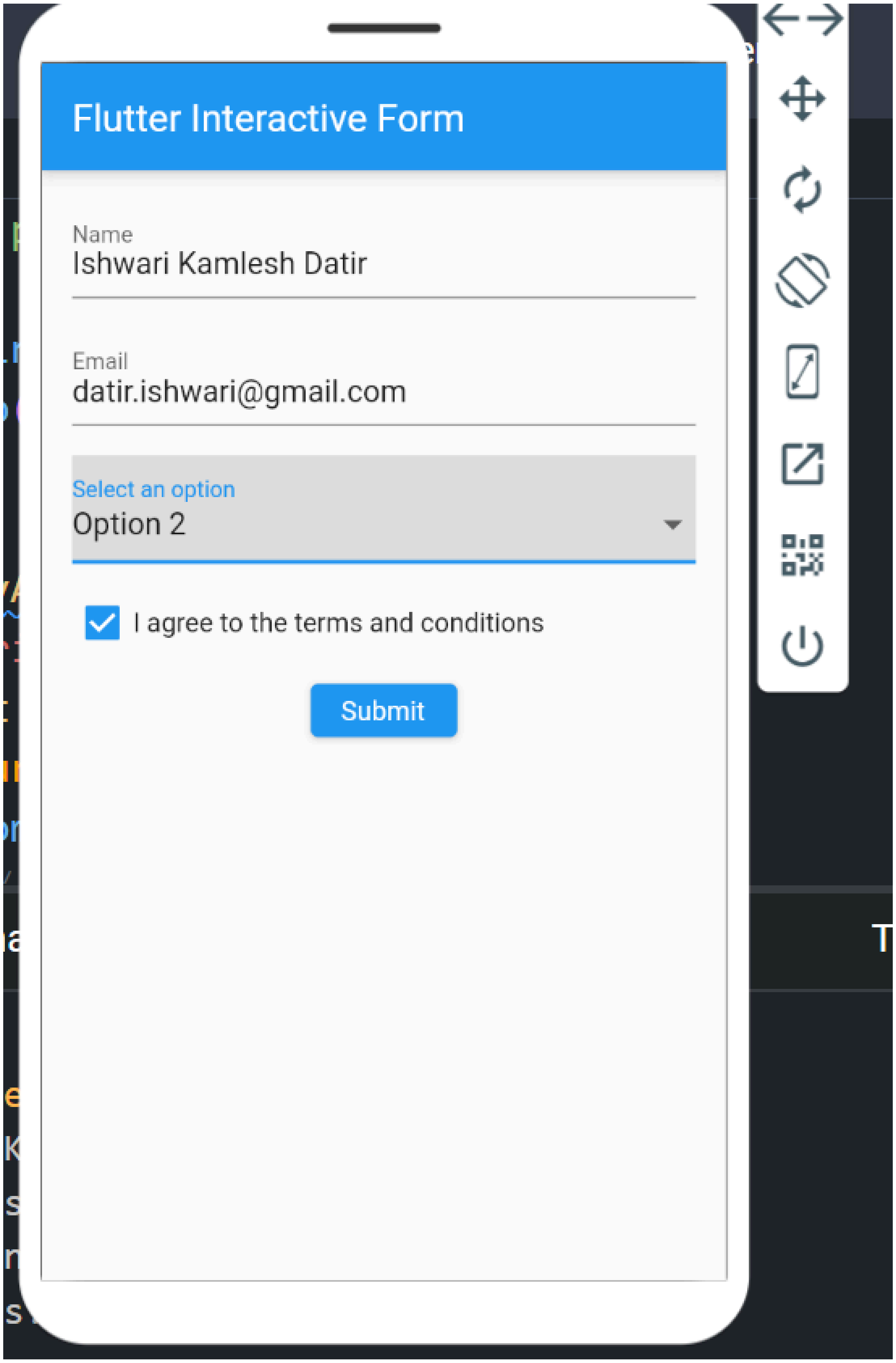
The stateful nature of Flutter widgets, exemplified by the combination of a stateful `MyForm` widget and its associated state class, underscores the versatility of Flutter in handling dynamic user interfaces. The state class, `_MyFormState`, not only manages the state of the form but also encapsulates the logic for form validation, submission, and any additional interactivity. This modular design promotes maintainability and scalability, making it easier to extend the functionality of the form or integrate it with other components. In conclusion, Flutter's form-building approach empowers developers to create interactive and responsive user interfaces with streamlined form management and enhanced user experience.

Moreover, Flutter provides a declarative UI paradigm, allowing developers to express the desired appearance and behavior of the form without delving into the intricacies of imperative code. The expressive syntax, combined with Flutter's hot reload feature, facilitates rapid iteration and debugging during the form development process. This results in an efficient and intuitive development experience, fostering the creation of visually appealing and functional forms. Additionally, the Flutter framework's cross-platform capabilities ensure that the developed forms seamlessly adapt to various devices and screen sizes, enhancing the versatility of applications across different platforms. Overall, Flutter's approach to form development promotes both simplicity and sophistication, making it a powerful framework for crafting compelling user interfaces with dynamic and interactive forms.

Code:

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: MyForm(),
    );
  }
}

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

class _MyFormState extends State<MyForm> {
  final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
  String _name = "";
  String _email = "";
  String _selectedOption = "";
  bool _isChecked = false;

  List<String> _dropdownOptions = ["Option 1", "Option 2", "Option 3"];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Flutter Interactive Form'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
child: Form(
  key: _formKey,
  child: Column(
    children: [
      TextFormField(
        decoration: InputDecoration(labelText: 'Name'),
        validator: (value) {
          if (value == null || value.isEmpty) {
            return 'Please enter your name';
          }
          return null;
        },
        onSaved: (value) {
          _name = value!;
        },
      ),
      SizedBox(height: 16),
      TextFormField(
        decoration: InputDecoration(labelText: 'Email'),
        validator: (value) {
          if (value == null || value.isEmpty) {
            return 'Please enter your email';
          }
                    return null;
        },
        onSaved: (value) {
          _email = value!;
        },
      ),
      SizedBox(height: 16),
      DropdownButtonFormField<String>(
        value: _selectedOption,
        items: _dropdownOptions
            .map((option) => DropdownMenuItem<String>(
                  child: Text(option),
                  value: option,
                ))
            .toList(),
        onChanged: (String? value) {
          setState(() {
                _selectedOption = value!;
              });
            },
            validator: (value) {
              if (value == null || value.isEmpty) {
                return 'Please select an option';
              }
              return null;
            },
            decoration: InputDecoration(labelText: 'Select an option'),
          ),
          SizedBox(height: 16),
          Row(
            children: [
              Checkbox(
                value: _isChecked,
                onChanged: (value) {
                  setState(() {
                    _isChecked = value!;
                  });
                },
              ),
              Text('I agree to the terms and conditions'),
            ],
          ),
          SizedBox(height: 16),
          ElevatedButton(
            onPressed: _submitForm,
            child: Text('Submit'),
          ),
        ],
      ),
    ),
  ),
 );
}

void _submitForm() {
  if (_formKey.currentState!.validate()) {
    _formKey.currentState!.save();
```

```
          print('Name: $_name');
     print('Email: $_email');
     print('Selected Option: $_selectedOption');
     print('Agreed to terms: $_isChecked');
   }
  }
}
```



Flutter Interactive Form

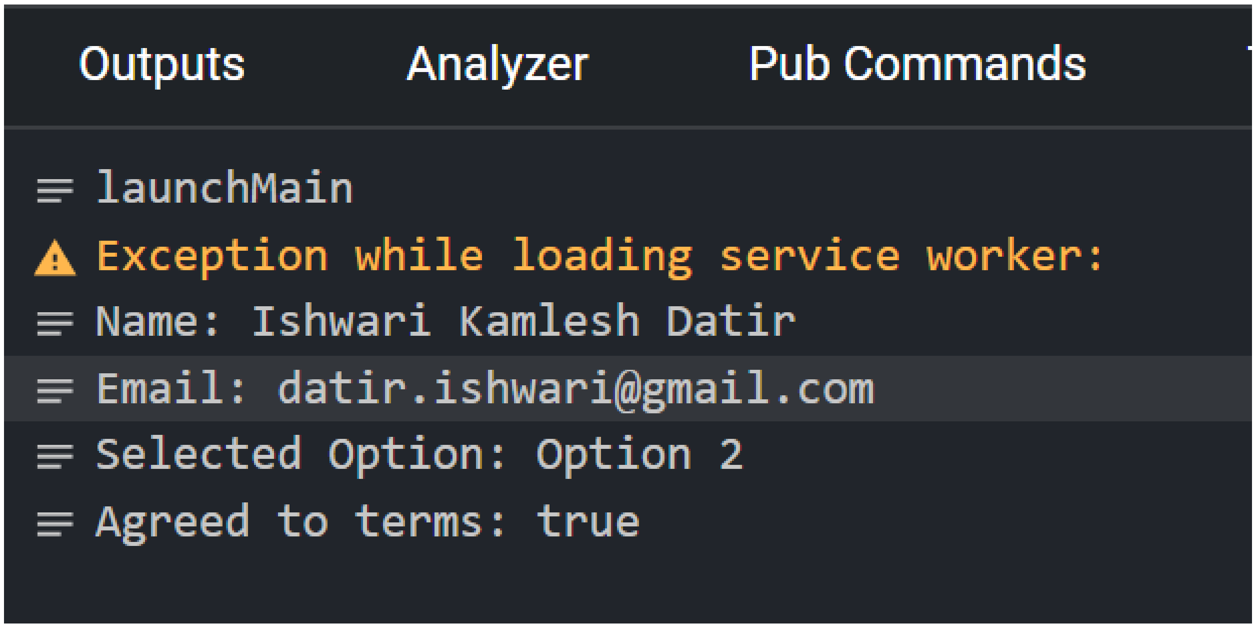
Name
Ishwari Kamlesh Datir

Email
datir.ishwari@gmail.com

Select an option
Option 2

☑ I agree to the terms and conditions

Submit



```
launchMain
Exception while loading service worker:
Name: Ishwari Kamlesh Datir
Email: datir.ishwari@gmail.com
Selected Option: Option 2
Agreed to terms: true
```

Explaining step by step:

1. **Import Flutter Material package**:

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

This line imports the Flutter Material package, which provides widgets and tools for building material design applications.

2. **Define the main app class**:

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

The `main` function is the entry point of the Flutter app. It calls `runApp` with an instance of the `MyApp` class to start the application.

3. **Create the main app widget class**:

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: MyForm(),
    );
  }
}
```

`MyApp` is a stateless widget that returns a `MaterialApp`. It sets the home property to `MyForm()`, which is the main form widget.

4. **Define the form widget class**:

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

`MyForm` is a stateful widget that returns an instance of `_MyFormState`. This is because we need to maintain the state of the form.

5. **Create the form state class**:

```
class _MyFormState extends State<MyForm> {

  final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
  String _name = "";
  String _email = "";
  String _selectedOption = "";
  bool _isChecked = false;
  List<String> _dropdownOptions = ["Option 1", "Option 2", "Option 3"];
```

`_MyFormState` is the state class for `MyForm`. It holds the state variables like `_name`, `_email`, `_selectedOption`, `_isChecked`, and `_dropdownOptions`.

6. **Build the form UI**:

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Flutter Interactive Form'),
    ),
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: Form(
        key: _formKey,
        child: Column(
          children: [

            TextFormField(
              // ... (Name input field)
            ),
            SizedBox(height: 16),

            // ... (Email input field)

            SizedBox(height: 16),


            DropdownButtonFormField<String>(
              // ... (Dropdown button)
            ),
            SizedBox(height: 16),


            Row(
              // ... (Checkbox)
            ),
            SizedBox(height: 16),


            ElevatedButton(
              // ... (Submit button)
            ),
          ],
        ),
        ),
      ),
    );
  }
```

This `build` method defines the UI for the form using various Flutter widgets like `Scaffold`, `AppBar`, `Form`, `TextFormField`, `DropdownButtonFormField`, `Checkbox`, and `ElevatedButton`.

7. **Implement form validation and data processing**:

```
  void _submitForm() {
    if (_formKey.currentState!.validate()) {
      _formKey.currentState!.save();


      print('Name: $_name');
      print('Email: $_email');
      print('Selected Option: $_selectedOption');
      print('Agreed to terms: $_isChecked');
    }
  }
```

The `_submitForm` function is called when the form is submitted. It checks if the form is valid, saves the form data, and then processes the data. In this example, it prints the data to the console.

8. **Run the app**:
Finally, the app is run by calling `runApp(MyApp())` in the `main` function, and the user interface defined in the `MyApp` class is displayed on the screen.

Conclusion:Thus, we conclude that the process of creating an interactive form in Flutter entails defining a `MyForm` widget, utilizing the `Form` widget for UI design, and implementing logic within its associated state class to facilitate user input, validation, and submission handling. This approach ensures a streamlined and effective form-building experience in Flutter.