

Aim: To create an interactive Form using a form widget

Theory:

Steps:

- 1.Create a Form with a GlobalKey.
- 2.Add a TextFormField with validation logic.
- 3.Create a button to validate and submit the form.

Form Widget: The Form widget in Flutter provides a way to manage and validate a group of form fields. It is a container for various form-related widgets. A form usually consists of multiple form fields, such as text fields, checkboxes, radio buttons, and buttons.

Form Fields:

1. **TextFormField:** The TextFormField widget is used for single-line text input. It automatically handles validation, error messages, and updating the form state.

Example:

```
TextFormField(  
  decoration: InputDecoration(labelText:  
    'Username'), validator: (value)  
  { if  
    (value == null || value.isEmpty)  
  {  
    return 'Please enter your username';  
  }  
  return null;  
}, )
```

2. **DropDownButtonFormField:**

The DropDownButtonFormField widget creates a dropdown menu and automatically manages its state.

Example:

```
DropDownButtonFormField(  
  value: selectedCountry,  
  items: ['USA', 'Canada', 'UK'].map((String country)  
  {  
    return DropdownMenuItem( value: country, child: Text(country),  
  );  
}).toList(), onChanged: (value)  
  { setState()  
    { selectedCountry = value; }  
  );  
},
```

```
decoration: InputDecoration(labelText: 'Country'),  
)
```

Form Validation: Form validation ensures that the user input meets specific criteria before allowing form submission. It helps maintain data integrity. Validation is typically done using the validator property of form fields. Validators are functions that return an error message if the input is invalid or null if it's valid

Code :

```
import 'package:flutter/material.dart';  
  
void main() {  
  runApp(const MyApp());  
}  
  
class MyApp extends StatelessWidget {  
  const MyApp({Key? key}) : super(key: key);  
  
  @override  
  Widget build(BuildContext context) {  
    return const MaterialApp(  
      home: MyForm(),  
    );  
  }  
}  
  
class MyForm extends StatefulWidget {  
  const MyForm({Key? key}) : super(key: key);  
  
  @override  
  _MyFormState createState() => _MyFormState();  
}  
  
class _MyFormState extends State<MyForm> {  
  final GlobalKey<FormState> _formKey = GlobalKey<FormState>();  
  String? _gender;  
  String? _selectedCity;  
  final List<String> _cities = ['Mumbai', 'Thane', 'Chembur', 'Kurla', 'Dadar'];  
  
  @override  
  Widget build(BuildContext context) {  
    return Scaffold(  
      appBar: AppBar(  

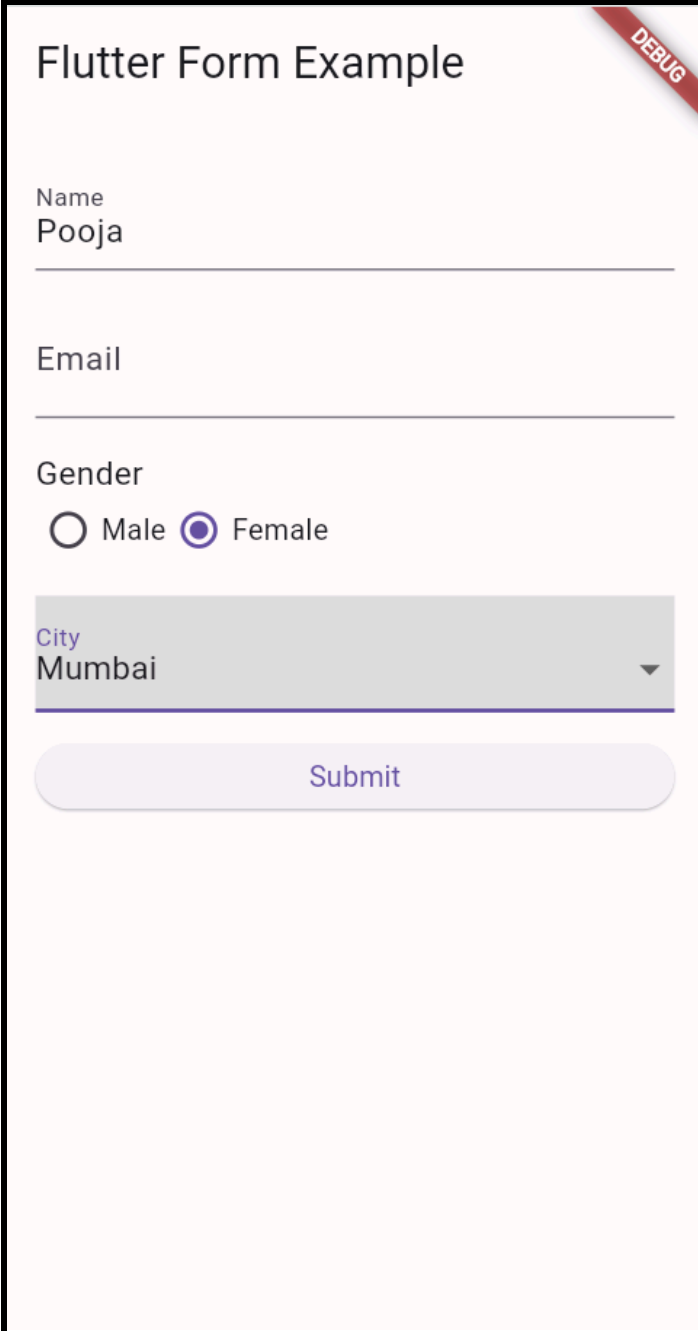
```

```
title: const Text('Flutter Form Example'),
),
body: Padding(
padding: const EdgeInsets.all(16.0),
child: Form(
key: _formKey,
child: Column(
crossAxisAlignment: CrossAxisAlignment.stretch,
children: [
TextFormField(
decoration: const InputDecoration(labelText: 'Name'),
validator: (value) {
if (value == null || value.isEmpty) {
return 'Please enter your name';
}
return null;
},
onSaved: (value) {
// Not implemented in this example
},
),
const SizedBox(height: 16),
TextFormField(
decoration: const InputDecoration(labelText: 'Email'),
validator: (value) {
if (value == null || value.isEmpty || !value.contains('@')) {
return 'Please enter a valid email address';
}
return null;
},
onSaved: (value) {
// Not implemented in this example
},
),
const SizedBox(height: 16),
const Text('Gender', style: TextStyle(fontSize: 16)),
Row(
children: [
Radio(
value: 'Male',
groupValue: _gender,
onChanged: (value) {
setState(() {
_gender = value.toString();

```

```
    });  
  },  
),  
const Text('Male'),  
Radio(  
  value: 'Female',  
  groupValue: _gender,  
  onChanged: (value) {  
    setState(() {  
      _gender = value.toString();  
    });  
  },  
),  
const Text('Female'),  
],  
,  
const SizedBox(height: 16),  
DropdownButtonFormField<String>(  
  value: _selectedCity,  
  items: _cities.map((city) {  
    return DropdownMenuItem<String>(  
      value: city,  
      child: Text(city),  
    );  
  }).toList(),  
  onChanged: (value) {  
    setState(() {  
      _selectedCity = value;  
    });  
  },  
  decoration: const InputDecoration(labelText: 'City'),  
  validator: (value) {  
    if (value == null || value.isEmpty) {  
      return 'Please select a city';  
    }  
    return null;  
  },  
,  
,  
const SizedBox(height: 16),  
ElevatedButton(  
  onPressed: () {  
    _submitForm();  
  },  
  child: const Text('Submit'),
```

```
        ),  
        ],  
    ),  
    ),  
    );  
}  
  
void _submitForm() {  
    if (_formKey.currentState!.validate()) {  
        _formKey.currentState!.save();  
        // Form is valid, do something with the data  
    }  
}  
}
```

A screenshot of a Flutter form titled "Flutter Form Example" with a red "DEBUG" banner in the top right corner. The form contains several input fields: a "Name" field with the text "Pooja", an "Email" field, a "Gender" section with radio buttons for "Male" and "Female" (where "Female" is selected), and a "City" dropdown menu showing "Mumbai". At the bottom is a light purple rounded button labeled "Submit".

Flutter Form Example

DEBUG

Name
Pooja

Email

Gender
☐ Male ☒ Female

City
Mumbai

Submit

Conclusion: This experiment showed us how to use form widgets in Flutter to make interactive and easy-to-use forms in our apps. We learned to add fields like name and email, validate user input, and create a submit button. This knowledge can be applied to build login and sign-up pages with user authentication in our projects.