**St. Francis Institute of Technology, Mumbai-400 103**

**Department of Information Technology**

**A.Y. 2024-2025**

**Class: TE-IT A/B, Semester: VI**

**Subject: MAD & PWA LAB**

**Experiment – 3: Creation of Forms for mobile Apps.**

1. Aim: To create an interactive form using form widget.

2. Objectives: After study of this experiment, the student will be able to

● Develop the App UI by incorporating form widget.

3. Outcomes: After study of this experiment, the student will be able to

● Design and Develop interactive Flutter App by using widgets (L604.2).

4. Prerequisite: Dart Programming Language.

5. Requirements: Android Studio, Flutter framework, Internet Connection.

6. Pre-Experiment Exercise:

Brief Theory:

● The Form widget is an optional container for grouping together multiple form field

widgets.

● The benefit of using a Form widget is to validate each text field as a group.

● You can group TextFormField widgets to manually or automatically validate them.

● The TextFormField widget wraps a TextField widget to provide validation when

enclosed in a Form widget.

● If all text fields pass the FormState validate method, then it returns true. If any text

fields contain errors, it displays the appropriate error message for each text field, and

the FormState validate method returns false. This process gives you the ability to use

FormState to check for any validation errors instead of checking each text field for

errors and not allowing the posting of invalid data.

● The Form widget needs a unique key to identify it and is created by using GlobalKey.

This GlobalKey value is unique across the entire app.

● We can create and validate a form using the following 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.

7. Laboratory Exercise

A. Program

i. Create a form for the mobile app having two text fields: person’s name and

contact number. The form should be able to accept the input when the user

submits the form. Also validate the form for null values and display appropriate

messages.

B. Result/Observation

i. Print out of program code and output.

8. Post-Experiment Exercise

A. Questions:

1. Add one more text field along with validation, and show the output. Also validate

the ‘phone number’ field to ensure that only numeric value is accepted in the

‘phone number’ field.

B. Conclusion:

1. Write what you have learnt in the experiment.

C. References:

1. https://api.flutter.dev/flutter/widgets/Form-class.html

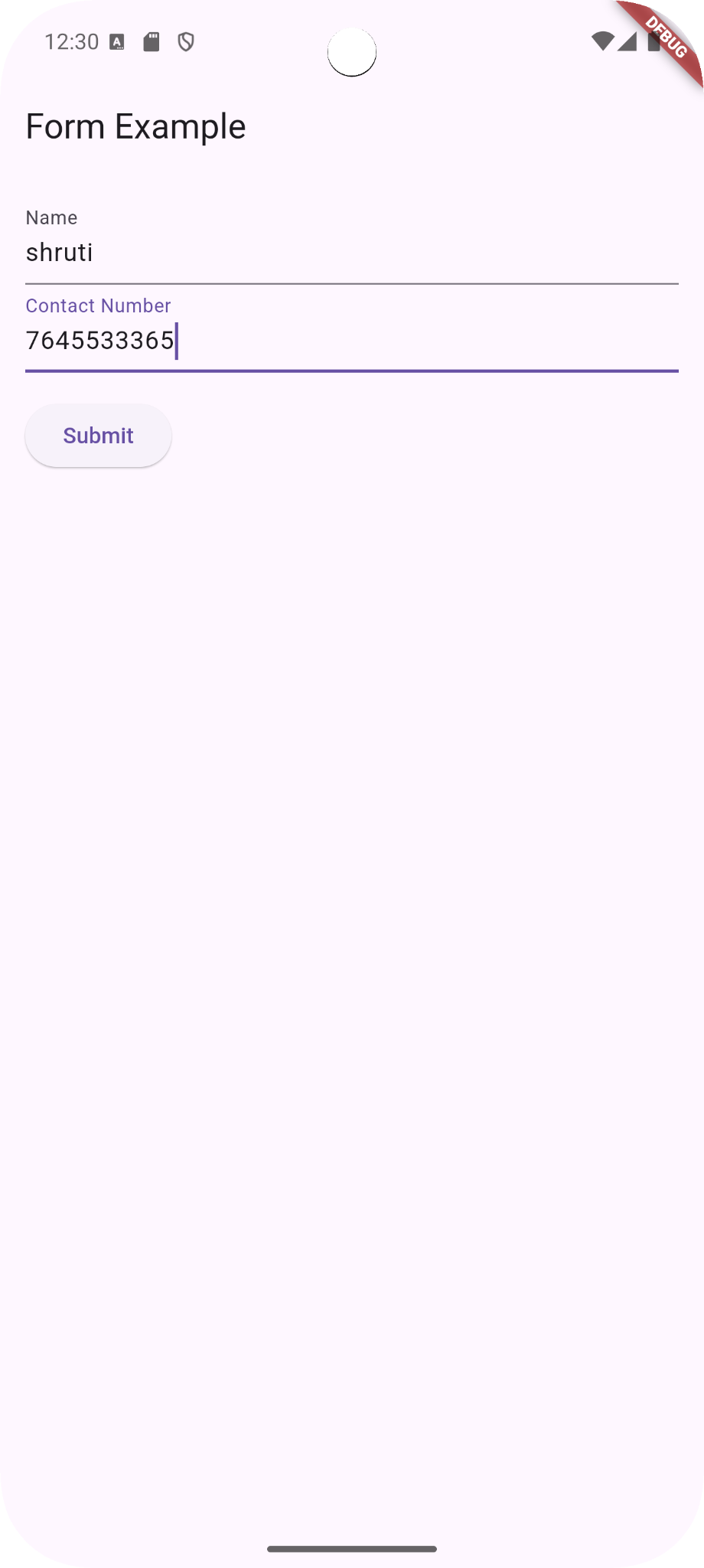
2. <https://docs.flutter.dev/cookbook/forms/validation>

7. Laboratory Exercise

A. Program

i. Create a form for the mobile app having two text fields: person’s name and contact number. The form should be able to accept the input when the user submits the form. Also validate the form for null values and display appropriate messages.

| import 'package:flutter/material.dart';  void main() {  runApp(MyApp());  }  class MyApp extends StatelessWidget {  @override  Widget build(BuildContext context) {  return MaterialApp(  home: Scaffold(  appBar: AppBar(title: Text('Form Example')),  body: Padding(  padding: const EdgeInsets.all(16.0),  child: MyForm(),  ),  ),  );  }  }  class MyForm extends StatefulWidget {  @override  \_MyFormState createState() => \_MyFormState();  }  class \_MyFormState extends State<MyForm> {  final \_formKey = GlobalKey<FormState>();  final \_nameController = TextEditingController();  final \_contactController = TextEditingController();  // Function to handle form submission  void \_submitForm() {  if (\_formKey.currentState?.validate() ?? false) {  ScaffoldMessenger.*of*(context).showSnackBar(  SnackBar(content: Text('Form Submitted Successfully')),  );  }  }  @override  Widget build(BuildContext context) {  return Form(  key: \_formKey,  child: Column( | crossAxisAlignment: CrossAxisAlignment.start,  children: <Widget>[  TextFormField(  controller: \_nameController,  decoration: InputDecoration(labelText: 'Name'),  validator: (value) {  if (value == null || value.isEmpty) {  return 'Please enter a name';  }  return null;  },  ),  TextFormField(  controller: \_contactController,  decoration: InputDecoration(labelText: 'Contact Number'),  keyboardType: TextInputType.*phone*,  validator: (value) {  if (value == null || value.isEmpty) {  return 'Please enter a contact number';  } else if (!RegExp(r'^[0-9]+$').hasMatch(value)) {  return 'Please enter a valid contact number';  }  return null;  },  ),  Padding(  padding: const EdgeInsets.symmetric(vertical: 16.0),  child: ElevatedButton(  onPressed: \_submitForm,  child: Text('Submit'),  ),  ),  ],  ),  );  }  } |
| --- | --- |

**OUTPUT** : 

**Post-Experiment Exercise**

A. Questions:

1. Add one more text field along with validation, and show the output. Also validate

the ‘phone number’ field to ensure that only numeric value is accepted in the

‘phone number’ field.

| **import 'package:flutter/material.dart';**  void main() {  runApp(MyApp());  }  class MyApp extends StatelessWidget {  @override  Widget build(BuildContext context) {  return MaterialApp(  home: Scaffold(  backgroundColor: Color(0xFFFBB0B0), // Baby pink background color  appBar: AppBar(title: Text('Form Example')),  body: Padding(  padding: const EdgeInsets.all(16.0),  child: MyForm(),  ),  ),  );  }  }  class MyForm extends StatefulWidget {  @override  \_MyFormState createState() => \_MyFormState();  }  class \_MyFormState extends State<MyForm> {  final \_formKey = GlobalKey<FormState>();  final \_nameController = TextEditingController();  final \_contactController = TextEditingController();  final \_emailController = TextEditingController();  String? \_gender; // To store selected gender | // Function to handle form submission  void \_submitForm() {  if (\_formKey.currentState?.validate() ?? false) {  if (\_gender == null) {  ScaffoldMessenger.*of*(context).showSnackBar(  SnackBar(content: Text('Please select a gender')),  );  } else {  ),  // Email field  TextFormField(  controller: \_emailController,  decoration: InputDecoration(labelText: 'Email'),  keyboardType: TextInputType.*emailAddress*,  validator: (value) {  if (value == null || value.isEmpty) {  return 'Please enter an email';  } else if (!RegExp(r'^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$').hasMatch(value)) {  return 'Please enter a valid email address';  }  return null;  },  ),  // Gender field - Radio buttons  Row(  children: <Widget>[  Text('Gender: '),  Row(  children: <Widget>[  Radio<String>( |
| --- | --- |

| value: 'Male',  groupValue: \_gender,  onChanged: (String? value) {  setState(() {  \_gender = value;  });  },  ),  Text('Male'),  Radio<String>(  value: 'Female',  groupValue: \_gender,  onChanged: (String? value) {  setState(() {  \_gender = value;  });  },  ),  Text('Female'),  Radio<String>(  value: 'Other',  groupValue: \_gender,  onChanged: (String? value) {  setState(() {  \_gender = value;  });  },  ),  Text('Other'),  ],  ),  ],  ),  Padding(  padding: const EdgeInsets.symmetric(vertical: 16.0),  child: ElevatedButton(  onPressed: \_submitForm,  child: Text('Submit'),  ),  ),  ],  ),  ); |
| --- |

**OUTPUT**:

