

### **EXPERIMENT NO.4**

<b>Experiment No 4:</b> <b>To create an interactive form using form widget</b>	
<b>ROLL NO</b>	<b>28</b>
<b>NAME</b>	<b>Anushka Karhadkar</b>
<b>CLASS</b>	<b>D15-B</b>
<b>SUBJECT</b>	<b>MAD &amp; PWA Lab</b>
<b>LO-MAPPE D</b>	

**Aim :** To create an interactive form using form widget

## **Theory :**

### Flutter Forms

Forms are an integral part of all modern mobile and web applications. It is mainly used to interact with the app as well as gather information from the users. They can perform many tasks, which depend on the nature of your business requirements and logic, such as authentication of the user, adding user, searching, filtering, ordering, booking, etc. A form can contain text fields, buttons, checkboxes, radio buttons, etc.

### Creating Form

Flutter provides a Form widget to create a form. The form widget acts as a container, which allows us to group and validate the multiple form fields. When you create a form, it is necessary to provide the GlobalKey. This key uniquely identifies the form and allows you to do any validation in the form fields.

The form widget uses child widget TextFormField to provide the users to enter the text field. This widget renders a material design text field and also allows us to display validation errors when they occur.

### Form validation

Validation is a method, which allows us to correct or confirms a certain standard. It ensures the authentication of the entered data.

Validating forms is a common practice in all digital interactions. To validate a form in a flutter, we need to implement mainly three steps.

Step 1: Use the Form widget with a global key.

Step 2: Use TextFormField to give the input field with validator property.

Step 3: Create a button to validate form fields and display validation errors.

The validator() function in the TextFormField to validates the input properties. If the user gives the wrong input, the validator function returns a string that contains an error message; otherwise, the validator function returns null. In the validator function, make sure that the TextFormField is not empty. Otherwise, it returns an error message.

The validator() function can be written as below code snippets:

```
validator: (value) {  
    if (value.isEmpty) {  
        return 'Please enter some text';  
    }  
    return null;  
},
```

**Code:**

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Blood Donation Form',
      home: BloodDonationForm(),
    );
  }
}

class BloodDonationForm extends StatefulWidget {
  @override
  _BloodDonationFormState createState() => _BloodDonationFormState();
}

class _BloodDonationFormState extends State<BloodDonationForm> {
  final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
  TextEditingController _nameController = TextEditingController();
  TextEditingController _bloodTypeController = TextEditingController();
  TextEditingController _contactInfoController = TextEditingController();

  @override
```

```
Widget build(BuildContext context) {  
  return Scaffold(  
    appBar: AppBar(  
      title: Text('Blood Donation Form'),  
    ),  
    body: Padding(  
      padding: const EdgeInsets.all(16.0),  
      child: Form(  
        key: _formKey,  
        child: Column(  
          crossAxisAlignment: CrossAxisAlignment.start,  
          children: [  
            TextFormField(  
              controller: _nameController,  
              decoration: InputDecoration(  
                labelText: 'Donor Name',  
                hintText: 'Enter donor name',  
              ),  
              validator: (value) {  
                if (value == null || value.isEmpty) {  
                  return 'Please enter donor name';  
                }  
                return null;  
              },  
            ),  
            SizedBox(height: 16.0),  
            TextFormField(  
              controller: _bloodTypeController,  
              decoration: InputDecoration(  
                labelText: 'Blood Type',
```

```

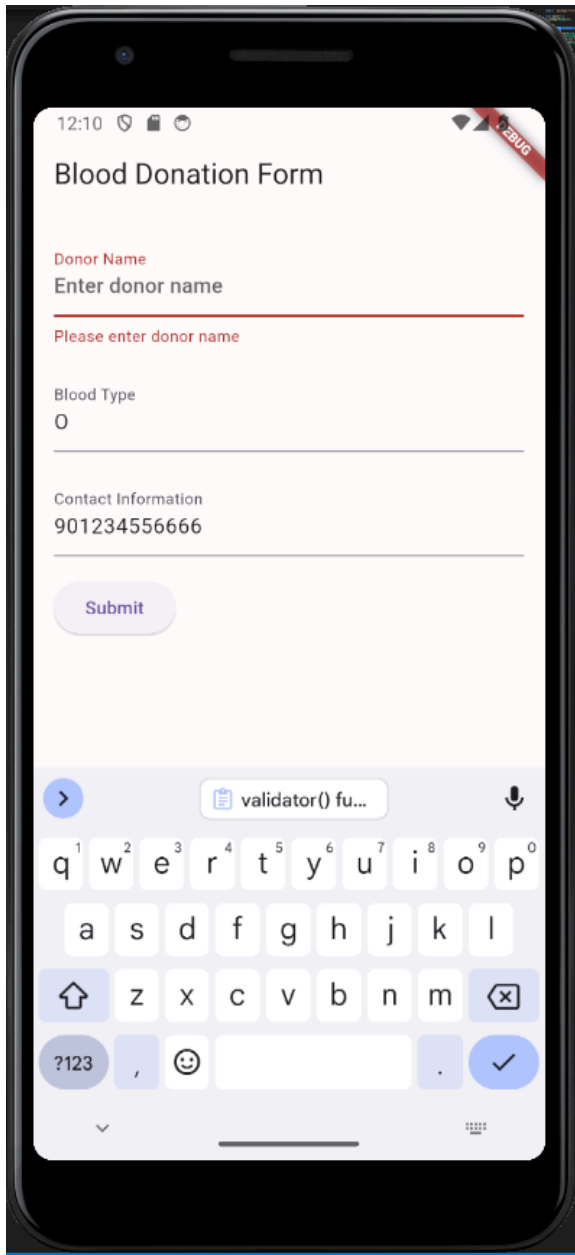
        hintText: 'Enter blood type (e.g., A+, B-, O)',
    ),
    validator: (value) {
        if (value == null || value.isEmpty) {
            return 'Please enter blood type';
        }
        // Add more complex blood type validation if needed
        return null;
    },
),
 SizedBox(height: 16.0),
 TextFormField(
    controller: _contactInfoController,
    decoration: InputDecoration(
        labelText: 'Contact Information',
        hintText: 'Enter contact information',
    ),
    validator: (value) {
        if (value == null || value.isEmpty) {
            return 'Please enter contact information';
        }
        return null;
    },
),
 SizedBox(height: 16.0),
 ElevatedButton(
    onPressed: () {
        if (_formKey.currentState!.validate()) {
            // Form is valid, process the data
            String name = _nameController.text;

```

```
String bloodType = _bloodTypeController.text;
String contactInfo = _contactInfoController.text;

// Process the blood donation form data
print(
  'Donor Name: $name, Blood Type: $bloodType, Contact Info: $contactInfo');
}
},
child: Text('Submit'),
),
],
),
),
),
);
}
}
```

## Output:



12:10

### Blood Donation Form

Donor Name  
Enter donor name

Please enter donor name

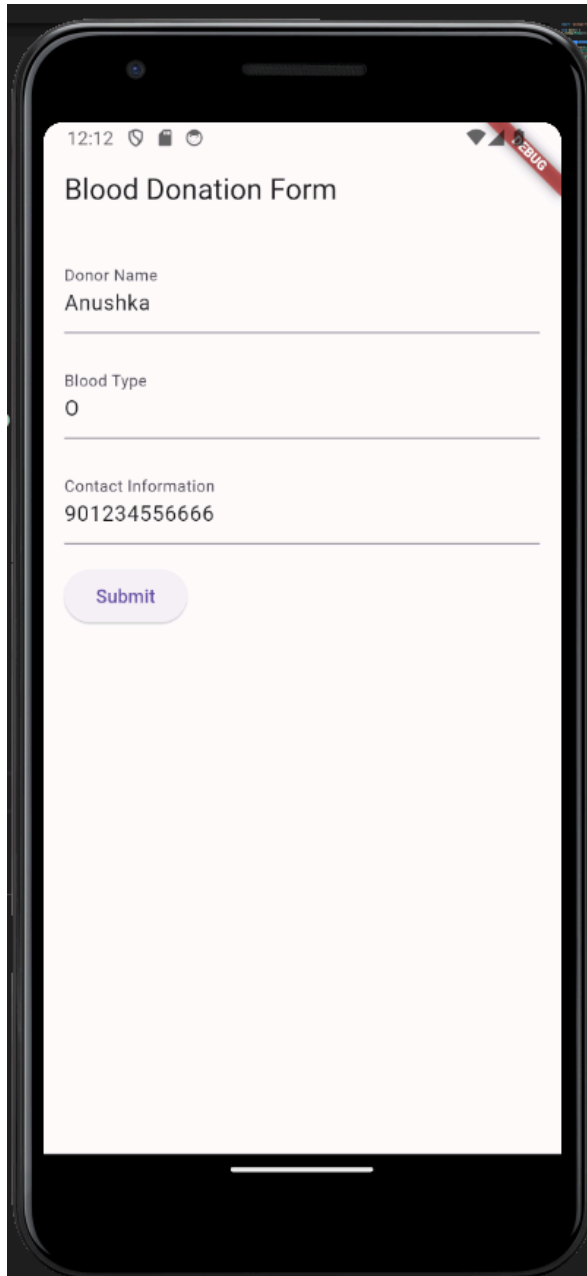
Blood Type  
O

Contact Information  
901234556666

Submit

validator() fu...

q w e r t y u i o p  
a s d f g h j k l  
z x c v b n m  
?123 , .



12:12

### Blood Donation Form

Donor Name  
Anushka

Blood Type  
O

Contact Information  
901234556666

Submit

**Conclusion:**

We understood and implemented a blood donation form and validated it by using form widgets and created a form to solicit critical details such as the donor's name, blood type, and contact information, for users to express their willingness to contribute to blood donation initiatives.