

## EXPERIMENT NO. 4

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

### Theory:

Creating an interactive form using the Form widget in Flutter involves designing a user interface to collect input data from users, validate it, and process it accordingly. Here's a simplified breakdown:

#### 1. User Interface Design:

- Design the form UI using widgets like TextFormField, DropdownButtonFormField, CheckboxListTile, etc., to collect user input.
- Each form field widget requires configuration such as decoration for styling, validation logic, and a way to save user input.

#### 2. Validation:

- Ensure the data entered by users meets certain criteria.
- Validators are functions attached to form fields that return an error message if the input is invalid, or null if it's valid.

#### 3. Form Submission:

- Collect input data, validate it, and process it when the user submits the form.
- Associate a function with a button's onPressed event to handle form submission.
- Before submission, validate the input data to ensure it meets required criteria.

#### 4. State Management:

- Use StatefulWidget to manage form state as it can hold mutable state that changes over time.
- Form state includes current values of form fields, whether the form is valid, and any error messages for invalid fields.

#### 5. Key Concepts:

- `Form`: Widget used to group form fields and provide access to form state for actions like validation and saving.
- `FormField`: Base class for form fields handling interaction with the form, focus, validation, and error display.
- `GlobalKey<FormState>`: Unique key for identifying a Form widget and accessing its state for actions like validation and saving.

By understanding these concepts, developers can effectively design and implement interactive forms in Flutter using the Form widget.

## Explanation:

### 1. Import Required Packages:

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

- In this snippet, we import the 'material.dart' package from Flutter. This package provides a set of pre-designed widgets following the material design guidelines.

### 2. Define a StatefulWidget for the Form:

```
dart
class FeedbackForm extends StatefulWidget {
  @override
  _FeedbackFormState createState() => _FeedbackFormState();
}
```

- Here, we define a StatefulWidget named 'FeedbackForm'. Stateful widgets are mutable and can change over time, which makes them suitable for forms that need to manage user input.

### 3. Create a GlobalKey for the Form:

```
dart
final _formKey = GlobalKey<FormState>();
```

- We create a 'GlobalKey' object named '\_formKey'. This key is used to uniquely identify the form and access its state, such as validation and saving user input.

### 4. Implement the Form State:

```
dart
class _FeedbackFormState extends State<FeedbackForm> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Feedback Form'),
      ),
      body: Form(
        key: _formKey,
        child: Column(
          // Form fields will be added here
        ),
      ),
    );
  }
}
```

```
);  
}  
}
```

- In this snippet, we define the state class '\_FeedbackFormState' for the 'FeedbackForm' widget. Inside the 'build' method, we return a 'Scaffold' widget containing a 'Form' widget. The 'Form' widget holds all the form fields.

#### 5. Add Form Fields:

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

- Here, we add a 'TextFormField' widget inside the 'Form' widget. This widget provides a text input field where users can enter their name. We also provide a 'validator' function to validate the user's input.

#### 6. Implement Validation Logic:

- The 'validator' function inside the 'TextFormField' widget validates the user input. In this case, it checks if the input is empty and returns an error message if it is.

#### 7. Manage Form Submission:

```
dart  
void _submitForm() {  
  if (_formKey.currentState!.validate()) {  
    // Form is valid, proceed with submission  
    _formKey.currentState!.save();  
    // Handle form data submission  
  }  
}
```

- Here, we define a function '\_submitForm' to handle form submission. We validate the form using the 'validate' method of the form's current state. If the form is valid, we call the 'save' method to save the form data.

#### 8. Bind Submission Function to Button:

```
dart
ElevatedButton(
  onPressed: _submitForm,
  child: Text('Submit'),
),
```

- We create an 'ElevatedButton' widget with an 'onPressed' callback that triggers the '\_submitForm' function when the button is pressed. The button displays the text "Submit".

#### 9. Implement Data Submission Logic:

```
dart
void _submitForm() {
  if (_formKey.currentState!.validate()) {
    // Form is valid, proceed with submission
    _formKey.currentState!.save();
    // Handle form data submission
    // For example, print the submitted data
    print('Form data submitted!');
  }
}
```

- Inside the '\_submitForm' function, we perform additional actions after validating the form, such as submitting the data. In this example, we print a message indicating that the form data has been submitted.

#### 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 MaterialApp(
    title: 'Feedback App',
    theme: ThemeData(
      primarySwatch: Colors.blue,
      inputDecorationTheme: const InputDecorationTheme(
        border: OutlineInputBorder(),
        contentPadding: EdgeInsets.symmetric(vertical: 12.0,
horizontal: 16.0),
      ),
    ),
    home: const FeedbackForm(),
  );
}

class FeedbackForm extends StatefulWidget {
  const FeedbackForm({Key? key}) : super(key: key);

  @override
  _FeedbackFormState createState() => _FeedbackFormState();
}

class _FeedbackFormState extends State<FeedbackForm> {
  final _formKey = GlobalKey<FormState>();
  String _name = '';
  String _email = '';
  String _category = 'General'; // Initialize to a default value
  String _feedback = '';

  void _submitForm() {
    if (_formKey.currentState!.validate()) {
      // Form is valid, proceed with submission
      _formKey.currentState!.save();
      // Print the data in the form
      print('Name: $_name');
      print('Email: $_email');
      print('Category: $_category');
      print('Feedback: $_feedback');
    }
  }
}

```

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Feedback Form'),
    ),
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: Form(
        key: _formKey,
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            TextFormField(
              decoration: const InputDecoration(
                labelText: 'Name',
                prefixIcon: Icon(Icons.person),
              ),
              validator: (value) {
                if (value == null || value.isEmpty) {
                  return 'Please enter your name';
                }
                return null;
              },
              onSave: (value) {
                _name = value!;
              },
            ),
            const SizedBox(height: 16.0),
            TextFormField(
              decoration: const InputDecoration(
                labelText: 'Email',
                prefixIcon: Icon(Icons.email),
              ),
              validator: (value) {
                if (value == null || value.isEmpty) {
                  return 'Please enter your email';
                }
                if (!value.contains('@')) {
                  return 'Please enter a valid email';
                }
                return null;
              },
            ),
          ],
        ),
      ),
    ),
  );
}
```

```

        onSave: (value) {
          _email = value!;
        },
      ),
    const SizedBox(height: 16.0),
    DropdownButtonFormField<String>(
      decoration: const InputDecoration(
        labelText: 'Category',
        prefixIcon: Icon(Icons.category),
      ),
      value: _category,
      items: ['General', 'Bug Report', 'Feature Request']
        .map((category) {
          return DropdownMenuItem<String>(
            value: category,
            child: Text(category),
          );
        }).toList(),
      onChanged: (value) {
        setState(() {
          _category = value!;
        });
      },
      validator: (value) {
        if (value == null || value.isEmpty) {
          return 'Please select a category';
        }
        return null;
      },
    ),
    const SizedBox(height: 16.0),
    TextFormField(
      decoration: const InputDecoration(
        labelText: 'Feedback',
        prefixIcon: Icon(Icons.feedback),
      ),
      maxLines: 5,
      validator: (value) {
        if (value == null || value.isEmpty) {
          return 'Please enter your feedback';
        }
        return null;
      },
    ),

```

```
        onSave: (value) {
          _feedback = value!;
        },
      ),
      const SizedBox(height: 20),
      SizedBox(
        width: double.infinity,
        child: ElevatedButton(
          onPressed: _submitForm,
          child: const Text('Submit'),
        ),
      ),
    ],
  ),
),
),
),
);
}
```

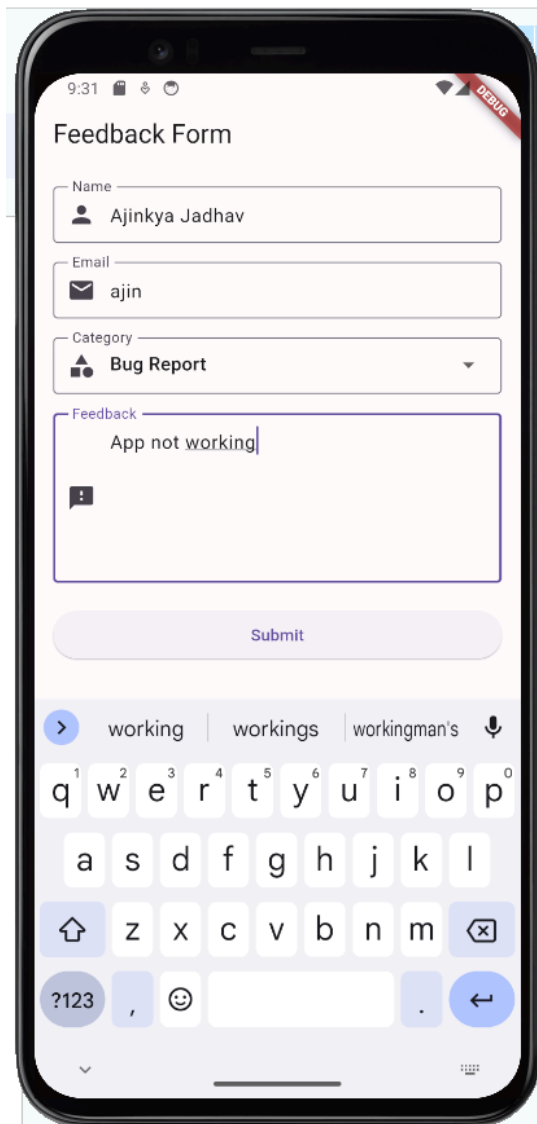


## Output:

Empty Form:

The image shows a mobile application interface for a 'Feedback Form'. The form is displayed on a smartphone screen with a black bezel. At the top, the status bar shows the time as 9:30 and various icons. The app's title bar is white with the text 'Feedback Form'. Below the title bar, there are four input fields: a 'Name' field with a person icon, an 'Email' field with an envelope icon, a 'Category' dropdown menu with a triangle icon and the text 'General', and a large 'Feedback' text area with a speech bubble icon. Below these fields is a purple 'Submit' button. At the bottom of the screen, a keyboard is visible with a search bar at the top containing the text 'import 'packa...' and 'Colors.blue'. The keyboard has a standard QWERTY layout with a blue checkmark button on the right.

After filling the form:



9:31

### Feedback Form

Name  
Ajinkya Jadhav

Email  
ajin

Category  
Bug Report

Feedback  
App not working

Submit

working | workings | workingman's

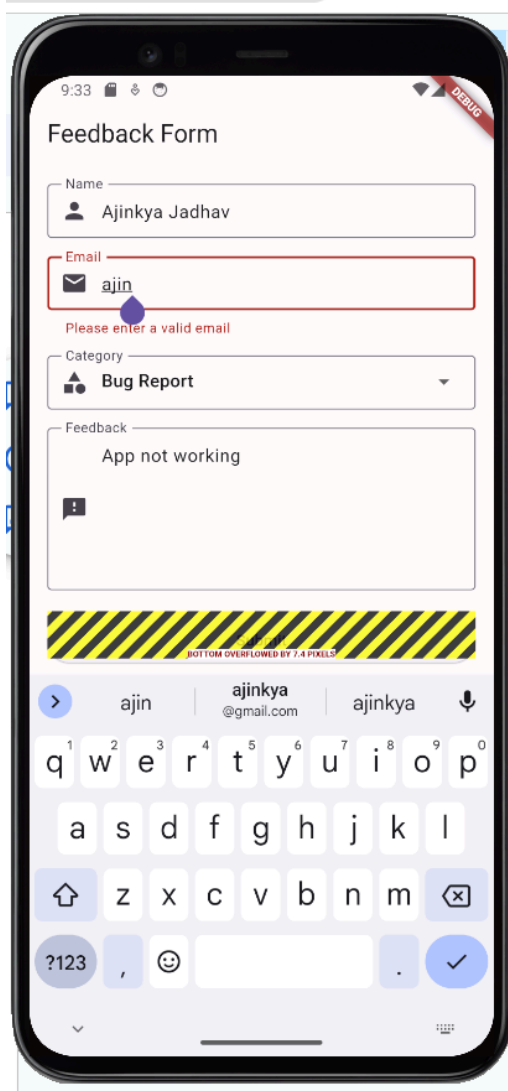
q w e r t y u i o p

a s d f g h j k l

z x c v b n m

?123 , .

## Form Validation:



After putting the correct values the form is submitted and the data is printed

```
D/EGL_emulation(19467): app_time_stats: avg=562.99ms min=15.98ms max=2698.86ms count=5
I/flutter (19467): Name: Ajinkya Jadhav
I/flutter (19467): Email: ajinkya@gmail.com
I/flutter (19467): Category: Bug Report
I/flutter (19467): Feedback: App not working
```

## Conclusion:

We have successfully learned how to apply form widget and perform data validation in the inputs given by user. I have tried to implement an organized UI to make it look appealing.