Name:Priyank Kulkarni **Class:D15B** **Roll No:32**

MAD Experiment 4

Aim: To create an interactive form using form widget

Theory:

The Form widget in Flutter is a fundamental widget for building forms. It provides a way to group multiple form fields together, perform validation on those fields, and manage their state. In this article, we are going to implement the Form widget and explore some properties and Methods of it. A sample video is given below to get an idea about what we are going to do in this article.

Some Properties of Form Widget

- **key**: A GlobalKey that uniquely identifies the Form. You can use this key to interact with the form, such as validating, resetting, or saving its state.
- **child**: The child widget that contains the form fields. Typically, this is a Column, ListView, or another widget that allows you to arrange the form fields vertically.
- **autovalidateMode:** An enum that specifies when the form should automatically validate its fields.

Some Methods of Form Widget

- **validate():** This method is used to trigger the validation of all the form fields within the Form. It returns true if all fields are valid, otherwise false. You can use it to check the overall validity of the form before submitting it.
- **save():** This method is used to save the current values of all form fields. It invokes the onSaved callback for each field. Typically, this method is called after validation succeeds.
- **reset():** Resets the form to its initial state, clearing any user-entered data.
- **currentState:** A getter that returns the current FormState associated with the Form.

Code:

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

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

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

class _FormPageState extends State<FormPage> {
  final _formKey = GlobalKey<FormState>();
```

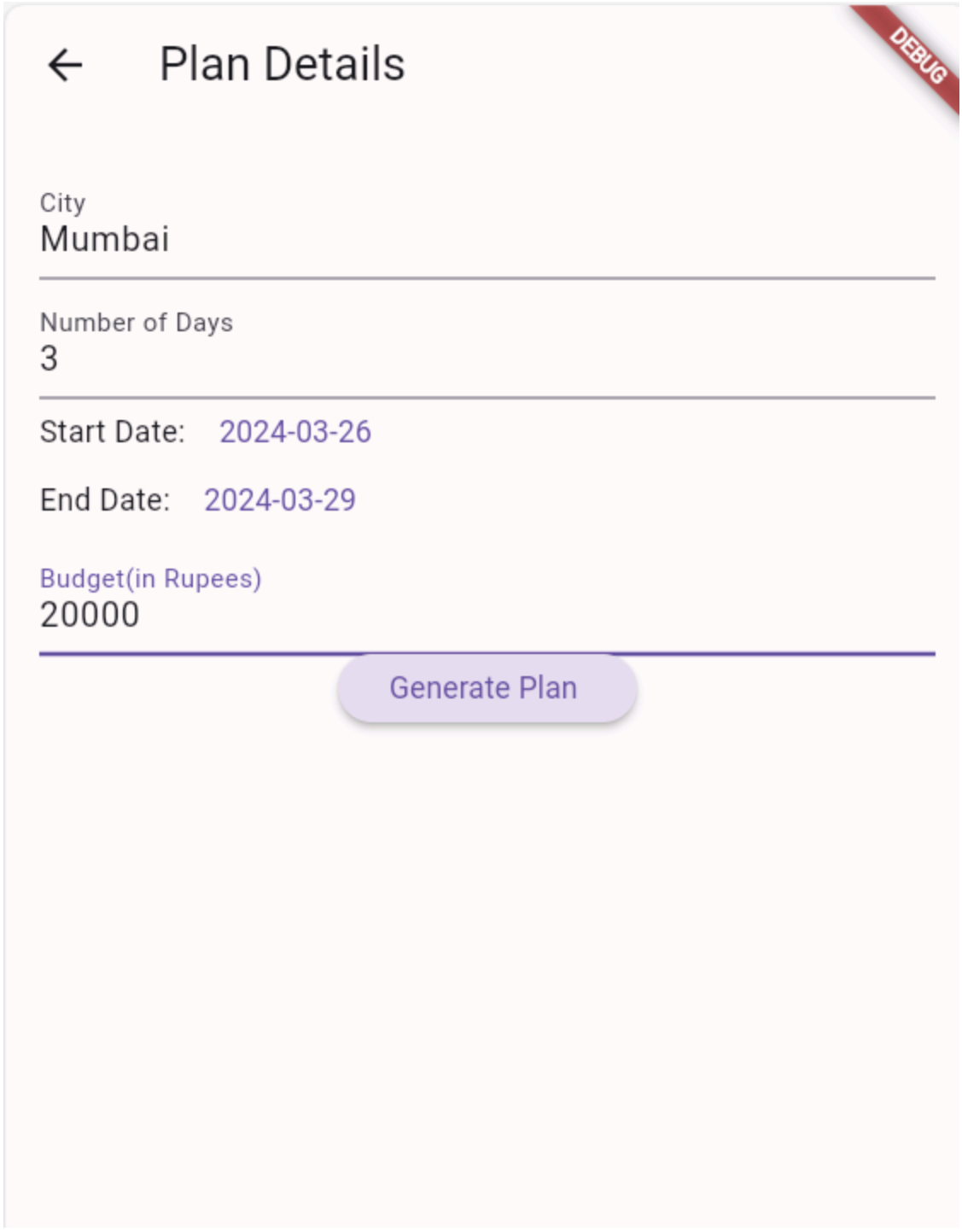
```
String _city = '';
int _days = 0;
DateTime _startDate = DateTime.now();
DateTime _endDate = DateTime.now();
double _budget = 0.0;
String _openAIResponse = '';
String _openAIError = '';

Future<void> _callOpenAI() async {
  try {
    // Replace 'YOUR_OPENAI_API_KEY' with your actual key
    OpenAI.apiKey = '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';
    String prompt =
        'Create a day wise trip plan for a $_days day trip to $_city in Rs $_budget';

    // Use OpenAI.instance.completions.create(...)
    OpenAICompletionModel response = await OpenAI.instance.completion.create(
      model: 'davinci-002', // Choose an appropriate OpenAI engine
      prompt: prompt,
      maxTokens: 1000, // Adjust maxTokens for response length
      n: 1,
      stop: null,
      temperature: 0.7,
    );

    setState(() {
      _openAIResponse = response.choices!.first.text.trim();

      _openAIError = '';
    });
  } catch (error) {
    setState(() {
      _openAIError = 'Error: ${error.toString()}';
      _openAIResponse = '';
    });
    print(error); // Print the complete error message for debugging
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Plan Details'),
```

```
),
body: Padding(
  padding: const EdgeInsets.all(16.0),
  child: Form(
    key: _formKey,
    child: Column(
      children: [
        TextFormField(
          decoration: const InputDecoration(labelText: 'City'),
          validator: (value) {
            if (value!.isEmpty) {
              return 'Please enter a city';
            }
            return null;
          },
          onSaved: (value) => setState(() => _city = value!),
        ),
        TextFormField(
          decoration: const InputDecoration(labelText: 'Number of Days'),
          keyboardType: TextInputType.number,
          validator: (value) {
            if (value!.isEmpty) {
              return 'Please enter number of days';
            }
            return null;
          },
          onSaved: (value) => setState(() => _days = int.parse(value!)),
        ),
        Row(
          children: [
            const Text('Start Date: '),
            TextButton(
              onPressed: () async {
                final pickedDate = await showDatePicker(
                  context: context,
                  initialDate: _startDate,
                  firstDate: DateTime.now(),
                  lastDate: DateTime(2100),
                );
                if (pickedDate != null) {
                  setState(() => _startDate = pickedDate);
                }
              },
              child: Text(_startDate.toString().split(' ').first),
```

```
    ),
  ],
),
Row(
  children: [
    const Text('End Date: '),
    TextButton(
      onPressed: () async {
        final pickedDate = await showDatePicker(
          context: context,
          initialDate: _endDate,
          firstDate: DateTime.now(),
          lastDate: DateTime(2100),
        );
        if (pickedDate != null) {
          setState(() => _endDate = pickedDate);
        }
      },
      child: Text(_endDate.toString().split(' ').first),
    ),
  ],
),
TextFormField(
  decoration:
      const InputDecoration(labelText: 'Budget(in Rupees)'),
  keyboardType: TextInputType.number,
  validator: (value) {
    if (value!.isEmpty) {
      return 'Please enter a budget';
    }
    return null;
  },
  onSaved: (value) =>
      setState(() => _budget = double.parse(value!)),
),
ElevatedButton(
  onPressed: () {
    if (_formKey.currentState!.validate()) {
      _formKey.currentState!.save();
      _callOpenAI();
    }
  },
  child: const Text('Generate Plan '),
),
```

```
            if (_openAIResponse.isNotEmpty)
              Container(
                height: 300.0, // Adjust height as needed
                child: SingleChildScrollView(
                  // Allow scrolling within response
                  child: Text(
                    _openAIResponse,
                    style: const TextStyle(fontSize: 16.0),
                  ),
                ),
              ),
          ],
        ),
      ),
    ),
  );
 }
}
```

Output:


Plan Details
DEBUG
City
Mumbai
Number of Days
3
Start Date: 2024-03-26
End Date: 2024-03-29
Budget(in Rupees)
20000
Generate Plan