ðŸ“Œ Project Plan

Creating a calculator app using Flutter is a great way to learn the framework! Below is a step-by-step guide to help you build a simple calculator app.

Step 1: Set Up Your Flutter Environment
1. Install Flutter by following the official [Flutter installation guide](https://flutter.dev/docs/get-started/install).
2. Create a new Flutter project:
```bash
flutter create calculator_app
cd calculator_app
```

Step 2: Design the User Interface
The calculator app will have:
- A display area to show the input and result.
- Buttons for numbers (0-9) and operations (+, -, *, /, =, C).

Hereâ€™s the code for the UI:

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

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

class CalculatorApp extends StatelessWidget {
@override
Widget build(BuildContext context) {
return MaterialApp(
title: 'Calculator',
theme: ThemeData(
primarySwatch: Colors.blue,
),
home: CalculatorScreen(),
);
}
}

class CalculatorScreen extends StatefulWidget {
@override
_CalculatorScreenState createState() => _CalculatorScreenState();
}

class _CalculatorScreenState extends State {
String _output = "0";
String _currentInput = "";
double _num1 = 0;
double _num2 = 0;
String _operator = "";

void _buttonPressed(String buttonText) {
setState(() {
if (buttonText == "C") {
_output = "0";
_currentInput = "";
_num1 = 0;
_num2 = 0;
_operator = "";
```

```
} else if (buttonText == "+" || buttonText == "-" || buttonText == "*" || buttonText == "/") {
_num1 = double.parse(_currentInput);
_operator = buttonText;
_currentInput = "";
} else if (buttonText == "=") {
_num2 = double.parse(_currentInput);
if (_operator == "+") {
_output = (_num1 + _num2).toString();
} else if (_operator == "-") {
_output = (_num1 - _num2).toString();
} else if (_operator == "*") {
_output = (_num1 * _num2).toString();
} else if (_operator == "/") {
_output = (_num1 / _num2).toString();
}
_currentInput = _output;
_num1 = 0;
_num2 = 0;
_operator = "";
} else {
_currentInput += buttonText;
_output = _currentInput;
}
});
}

Widget _buildButton(String buttonText) {
return Expanded(
child: ElevatedButton(
onPressed: () => _buttonPressed(buttonText),
child: Text(
buttonText,
style: TextStyle(fontSize: 24),
),
),
);
}

@override
Widget build(BuildContext context) {
return Scaffold(
appBar: AppBar(
title: Text('Calculator'),
),
body: Column(
children: [
Expanded(
child: Container(
padding: EdgeInsets.all(16),
alignment: Alignment.bottomRight,
child: Text(
_output,
style: TextStyle(fontSize: 48, fontWeight: FontWeight.bold),
),
),
),
Row(
children: [
_buildButton("7"),
_buildButton("8"),
_buildButton("9"),
_buildButton("/"),
],
),
Row(
```

```
children: [
_buildButton("4"),
_buildButton("5"),
_buildButton("6"),
_buildButton("*"),
],
),
Row(
children: [
_buildButton("1"),
_buildButton("2"),
_buildButton("3"),
_buildButton("-"),
],
),
Row(
children: [
_buildButton("0"),
_buildButton("C"),
_buildButton("="),
_buildButton("+"),
],
),
],
),
);
}
}
```

Step 3: Run the App
1. Save the code in `lib/main.dart`.
2. Run the app using:
```bash
flutter run
```

Step 4: Customize and Enhance
- Add more features like decimal support, backspace, or scientific operations.
- Improve the UI with themes, gradients, or animations.
- Test the app on different devices and screen sizes.

**Step 5: Publish (