import 'package:flutter/material.dart';

void main() {

runApp(CalculatorApp());

}

class CalculatorApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

debugShowCheckedModeBanner: false,

home: CalculatorScreen(),

);

}

}

class CalculatorScreen extends StatefulWidget {

@override

\_CalculatorScreenState createState() => \_CalculatorScreenState();

}

class \_CalculatorScreenState extends State<CalculatorScreen> {

String displayText = '0';

String \_result = '';

String \_operand1 = '';

String \_operand2 = '';

String \_operator = '';

bool \_decimalUsed = false; // Track if decimal point has been used

// Function to handle button presses

void \_buttonPressed(String text) {

setState(() {

if (text == 'C') {

displayText = '0';

\_operand1 = '';

\_operand2 = '';

\_operator = '';

\_result = '';

\_decimalUsed = false;

} else if (text == '=') {

if (\_operand1.isNotEmpty &&

\_operand2.isNotEmpty &&

\_operator.isNotEmpty) {

double num1 = double.parse(\_operand1);

double num2 = double.parse(\_operand2);

switch (\_operator) {

case '+':

\_result = (num1 + num2).toString();

break;

case '-':

\_result = (num1 - num2).toString();

break;

case 'x':

\_result = (num1 \* num2).toString();

break;

case '/':

if (num2 != 0) {

\_result = (num1 / num2).toString();

} else {

\_result = 'Error';

}

break;

}

displayText = \_result;

\_operand1 = \_result;

\_operand2 = '';

\_operator = '';

\_decimalUsed = false; // Reset decimal usage after calculation

}

} else if (text == '+' || text == '-' || text == 'x' || text == '/') {

// Only update operator if we have already entered an operand

if (\_operand1.isNotEmpty && \_operator.isEmpty) {

\_operator = text;

displayText =

\_operand1; // Show first operand until the second operand is entered

} else if (\_operand2.isNotEmpty && \_operator.isNotEmpty) {

// If second operand exists, calculate and update the display

\_buttonPressed('=');

\_operator = text; // Set the new operator

displayText = \_result;

}

} else if (text == '.') {

// Allow decimal point only if it hasn't been used

if (!\_decimalUsed) {

if (\_operator.isEmpty) {

\_operand1 += text;

displayText = \_operand1;

} else {

\_operand2 += text;

displayText = \_operand2;

}

\_decimalUsed = true; // Mark decimal point as used

}

} else {

// Add digits to operand

if (\_operator.isEmpty) {

\_operand1 += text;

displayText = \_operand1;

} else {

\_operand2 += text;

displayText = \_operand2;

}

}

});

}

// Calculator buttons

final List<List<String>> buttons = [

['C', '(', ')', '/'],

['7', '8', '9', 'x'],

['4', '5', '6', '-'],

['1', '2', '3', '+'],

['0', '.', '=', ''], // Last row with 3 buttons

];

@override

Widget build(BuildContext context) {

return Scaffold(

backgroundColor: Colors.black87, // Dark background for the calculator

appBar: AppBar(

title: Text('Kalkulyator'),

centerTitle: true,

backgroundColor: Colors.blue[400],

),

body: Column(

children: [

// Display area

Expanded(

flex: 1,

child: Container(

color: Colors.white, // White background for the display

alignment: Alignment.centerRight,

padding: EdgeInsets.all(20),

child: Text(

displayText, // Display the current input

style: TextStyle(fontSize: 60, color: Colors.blue),

),

),

),

// Buttons area

Expanded(

flex: 3,

child: Column(

children: [

Expanded(

flex: 4,

child: GridView.builder(

physics: NeverScrollableScrollPhysics(),

itemCount:

20, // Total buttons including the last row with 3 buttons

gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(

crossAxisCount: 4,

childAspectRatio: 1, // Square-shaped buttons

),

itemBuilder: (BuildContext context, int index) {

int row = index ~/ 4;

int col = index % 4;

// Ignore empty spaces for the last row

if (buttons[row][col] == '') return Container();

return GestureDetector(

onTap: () {

\_buttonPressed(buttons[row][col]);

},

child: Container(

margin: EdgeInsets.all(2),

color:

Colors.grey[900], // Dark background for buttons

alignment: Alignment.center,

child: Text(

buttons[row][col],

style: TextStyle(

color: Colors.white, // White text

fontSize: 40,

fontWeight: FontWeight.bold,

),

),

),

);

},

),

),

// Last row with custom layout

Expanded(

flex: 1,

child: Row(

children: [

// '=' button spanning two columns

Expanded(

flex: 2,

child: GestureDetector(

onTap: () {

\_buttonPressed('=');

},

child: Container(

margin: EdgeInsets.all(2),

color: Colors.grey[900], // Dark background for '='

alignment: Alignment.center,

child: Text(

'=',

style: TextStyle(

color: Colors.white,

fontSize: 50,

fontWeight: FontWeight.bold,

),

),

),

),

),

// '0' button

Expanded(

flex: 1,

child: GestureDetector(

onTap: () {

\_buttonPressed('0');

},

child: Container(

margin: EdgeInsets.all(2),

color: Colors.grey[900], // Dark background for '0'

alignment: Alignment.center,

child: Text(

'0',

style: TextStyle(

color: Colors.white,

fontSize: 38,

fontWeight: FontWeight.bold,

),

),

),

),

),

// '.' button

Expanded(

flex: 1,

child: GestureDetector(

onTap: () {

\_buttonPressed('.');

},

child: Container(

margin: EdgeInsets.all(2),

color: Colors.grey[900], // Dark background for '.'

alignment: Alignment.center,

child: Text(

'.',

style: TextStyle(

color: Colors.white,

fontSize: 40,

fontWeight: FontWeight.bold,

),

),

),

),

),

],

),

),

],

),

),

],

),

);

}

}