import 'package:flutter/material.dart';

import 'package:flutter/services.dart';

import 'package:google\_fonts/google\_fonts.dart';

void main() => runApp(MaterialApp(

home: Lab1V3(),

));

class Lab1V3 extends StatefulWidget {

const Lab1V3({super.key});

@override

\_Lab1PanelState createState()=> \_Lab1PanelState();

}

class \_Lab1PanelState extends State<Lab1V3> {

Color BackgroundColor = Color(0xFFD1D9E5);

Color Header = Color(0x7377868F);

Color ColorBackM = Color(0xFF007BFF);

Color ColorBackF = Color(0xFFFFFFFF);

Color ColorTextM = Colors.white;

Color ColorTextF = Color(0xFF007BFF);

int weight = 70;

int age = 22;

bool SelectM = true;

double IMC = 0;

String formattedImc="0";

String raspuns="";

double height = 0; // Folosim String pentru a stoca textul introdus

void Calculate(double height,int weight ) {

if (height > 0) {

height /= 100;

IMC = weight / (height \* height);

formattedImc = IMC.toStringAsFixed(2);

}

}

String getIMCInterpretation(double bmi, int age, bool esteBarbat) {

// Interpretarea IMC-ului pentru adulți

if (age >= 18) {

if (bmi < 18.5) {

return "Subponderal";

} else if (bmi >= 18.5 && bmi < 25) {

return "Greutate normală";

} else if (bmi >= 25 && bmi < 30) {

return "Supraponderal";

} else if (bmi >= 30 && bmi < 35) {

return "Obezitate clasa I";

} else if (bmi >= 35 && bmi < 40) {

return "Obezitate clasa II";

} else {

return "Obezitate clasa III";

}

}

// Interpretarea IMC-ului pentru copii/adolescenți

else {

// Percentile de IMC pentru copii/adolescenți

if (esteBarbat) {

// Logica pentru băieți

if (bmi < 5) {

return "Subponderal sever";

} else if (bmi >= 5 && bmi < 15) {

return "Subponderal";

} else if (bmi >= 15 && bmi < 85) {

return "Greutate normală";

} else if (bmi >= 85 && bmi < 95) {

return "Supraponderal";

} else {

return "Obezitate";

}

} else {

// Logica pentru fete

if (bmi < 5) {

return "Subponderal sever";

} else if (bmi >= 5 && bmi < 15) {

return "Subponderal";

} else if (bmi >= 15 && bmi < 85) {

return "Greutate normală";

} else if (bmi >= 85 && bmi < 95) {

return "Supraponderal";

} else {

return "Obezitate";

}

}

}

}

void UpdateColors() {

if (SelectM) {

ColorBackM = Color(0xFF007BFF);

ColorTextM = Colors.white;

ColorTextF = Color(0xFF007BFF);

ColorBackF = Colors.white;

} else {

ColorBackF = Color(0xFF007BFF);

ColorTextF = Colors.white;

ColorTextM = Color(0xFF007BFF);

ColorBackM = Colors.white;

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

backgroundColor: BackgroundColor,

// appBar: AppBar(

// title: Text(

// "Lab1 V3 Jitaru Evelin",

// style: TextStyle(

// fontStyle: FontStyle.italic,

// color: Colors.amber,

// ),

// ),

// centerTitle: true,

// backgroundColor: Header,

// ),

body: SafeArea(

child: Column(

children: [

Row(

children: [

Container(

margin: EdgeInsets.symmetric(vertical: 15, horizontal: 50),

child: Column(

children: [

Container(

margin: EdgeInsets.only(right: 100),

child: Text(

"Welcome😊",

style: GoogleFonts.notoColorEmoji(color: Colors.black),

),

),

Container(

child: Text(

"BMI Calculator",

style: GoogleFonts.poppins(

color: Colors.black,

fontWeight: FontWeight.w600,

fontSize: 24,

),

),

),

],

),

),

],

),

Row(

mainAxisAlignment: MainAxisAlignment.spaceAround,

children: [

Container(

margin: EdgeInsets.symmetric(vertical: 10, horizontal: 20),

child: ElevatedButton.icon(

onPressed: () {

setState(() {

SelectM = true;

UpdateColors();

});

},

style: ElevatedButton.styleFrom(

padding: EdgeInsets.symmetric(vertical: 18, horizontal: 23),

backgroundColor: ColorBackM,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(10),

),

),

icon: Icon(

Icons.male,

color: ColorTextM,

size: 18,

),

label: Padding(

padding: EdgeInsets.only(right: 10),

child: Text(

"Male ",

style: TextStyle(

color: ColorTextM,

fontSize: 16,

fontWeight: FontWeight.w300,

),

),

),

),

),

Container(

margin: EdgeInsets.symmetric(vertical: 10, horizontal: 20),

child: ElevatedButton.icon(

onPressed: () {

setState(() {

SelectM = false;

UpdateColors();

});

},

style: ElevatedButton.styleFrom(

padding: EdgeInsets.symmetric(vertical: 18, horizontal: 20),

backgroundColor: ColorBackF,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(10),

),

),

icon: Icon(

Icons.female,

color: ColorTextF,

size: 18,

),

label: Padding(

padding: EdgeInsets.only(right: 10),

child: Text(

"Female",

style: TextStyle(

color: ColorTextF,

fontSize: 16,

fontWeight: FontWeight.w300,

),

),

),

),

),

],

),

Row(

mainAxisAlignment: MainAxisAlignment.spaceAround,

children: [

Container(

decoration: BoxDecoration(

color: Colors.white,

borderRadius: BorderRadius.circular(10),

),

padding: EdgeInsets.all(10),

child: Column(

mainAxisAlignment: MainAxisAlignment.spaceAround,

children: [

Text("Weight"),

Padding(

padding: EdgeInsets.only(top: 30, bottom: 30),

child: Text("$weight",

style: TextStyle(fontSize: 50, fontWeight: FontWeight.bold),

),

),

Row(

mainAxisAlignment: MainAxisAlignment.spaceBetween,

children: [

TextButton(

style: ElevatedButton.styleFrom(

backgroundColor: Color(0xFF007BFF),

padding: EdgeInsets.all(0),

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(10),

),

fixedSize: Size(10, 10),

),

onPressed: () {

setState(() {

if(weight>0)

weight--;

else

weight=0;

});

},

child: Icon(Icons.remove, color: Colors.white, size: 20),

),

Padding(padding: EdgeInsets.all(20)),

TextButton(

style: ElevatedButton.styleFrom(

backgroundColor: Color(0xFF007BFF),

padding: EdgeInsets.all(0),

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(10),

),

fixedSize: Size(10, 10),

),

onPressed: () {

setState(() {

weight++;

});

},

child: Icon(Icons.add, color: Colors.white, size: 20),

),

],

),

],

),

),

Container(

decoration: BoxDecoration(

color: Colors.white,

borderRadius: BorderRadius.circular(10),

),

padding: EdgeInsets.all(10),

child: Column(

mainAxisAlignment: MainAxisAlignment.spaceAround,

children: [

Text("Age"),

Padding(

padding: EdgeInsets.only(top: 30, bottom: 30),

child: Text(

"$age",

style: TextStyle(fontSize: 50, fontWeight: FontWeight.bold),

),

),

Row(

mainAxisAlignment: MainAxisAlignment.spaceBetween,

children: [

TextButton(

style: ElevatedButton.styleFrom(

backgroundColor: Color(0xFF007BFF),

padding: EdgeInsets.all(0),

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(10),

),

fixedSize: Size(10, 10),

),

onPressed: () {

setState(() {

if(age>0)

age--;

else

age=0;

});

},

child: Icon(Icons.remove, color: Colors.white, size: 20),

),

Padding(padding: EdgeInsets.all(20)),

TextButton(

style: ElevatedButton.styleFrom(

backgroundColor: Color(0xFF007BFF),

padding: EdgeInsets.all(0),

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(10),

),

fixedSize: Size(10, 10),

),

onPressed: () {

setState(() {

if(age>=119)

age=119;

else

age++;

});

},

child: Icon(Icons.add, color: Colors.white, size: 20),

),

],

),

],

),

),

],

),

Row(

children: [

Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Container(

margin: EdgeInsets.only(left: 40, top: 20),

child: Text(

"Height",

style: GoogleFonts.aDLaMDisplay(fontSize: 12),

),

),

// Adaugă spațiu între text și câmpul de input

Container(

width: 200, // Ajustează lățimea câmpului de input

height: 58,

margin: EdgeInsets.only(left: 30),

padding: EdgeInsets.all(8.0),

child: TextField(

onChanged: (value) {

setState(() {

height = double.tryParse(value)??0.0;

if(height>=290.0){// Actualizează starea cu textul introdus

height=290;

}

print(height);

});

},

keyboardType: TextInputType.number, // Specifică că doar cifrele sunt permise

inputFormatters: <TextInputFormatter>[

FilteringTextInputFormatter.allow(RegExp(r'[0-9.]')), // Permite cifre și punctul zecimal

],

decoration:const InputDecoration(

labelText: 'Height',

labelStyle:TextStyle(fontSize: 13,color: Colors.grey) ,

border:OutlineInputBorder(

borderSide:BorderSide(color: Colors.white),

),

enabledBorder: OutlineInputBorder(

borderSide:BorderSide(color: Colors.white),

),

focusedBorder:OutlineInputBorder(

borderSide:BorderSide(color: Color(0xFF007BFF) )) ,

filled: true,

fillColor: Colors.white

),

),

),

],

),

],

),

Row(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Column(

children: [

Container(

padding: EdgeInsets.only(top: 20),

child: Text("$formattedImc",style: TextStyle(fontSize: MediaQuery.of(context).size.width < 600 ? MediaQuery.of(context).size.width\*0.15 : MediaQuery.of(context).size.width\* 0.05 ,color:Color(0xFF007BFF)),),

),

Container(

child: Text("$raspuns",style: TextStyle(fontSize: MediaQuery.of(context).size.width < 600 ? MediaQuery.of(context).size.width\*0.10 : MediaQuery.of(context).size.width\* 0.03,color:Color(0xFF007BFF) ),)

),

Container(

padding: EdgeInsets.only( top:MediaQuery.of(context).size.width < 600 ?50 : 0),

width: MediaQuery.of(context).size.width \* 0.85,

child: ElevatedButton(

onPressed: () {

setState(() {

Calculate(height,weight);

String newRaspuns = getIMCInterpretation(IMC, age, SelectM);

// Compară și actualizează raspunsul dacă este necesar

if (raspuns != newRaspuns) {

raspuns = newRaspuns;

}

});

},

style: ElevatedButton.styleFrom(

padding: EdgeInsets.symmetric(vertical: 18, horizontal: 40),

backgroundColor:Color(0xFF007BFF),

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(10),

),

),

child: const Padding(

padding: EdgeInsets.only(right: 10),

child: Text(

"Go to BackEnd",

style: TextStyle(

color: Colors.white,

fontSize: 16,

fontWeight: FontWeight.w300,

),

),

),

),

),

],

)

],

)

],

),

),

);

}

}