**MINISTERUL EDUCAȚIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Informatică şi Ingineria Sistemelor**

**RAPORT**

Lucrare practică nr.1

la cursul „***Programarea aplicațiilor mobile***”

**Tema: Introducerea în programarea cross platform utilizând Flutter**

**A efectuat student (ul/a): \_\_\_\_\_\_\_\_\_\_\_\_**(V.Alexandra, gr. IA-222 )

**A controlat:\_\_\_\_\_\_\_\_\_\_\_\_\_\_**(lect.univ.Buza D. )

**Chişinău 2024**

**CUPRINS**

[INTRODUCERE 3](#_Toc4462)

[REALIZAREA PRACTICĂ A SARCINII DE LUCRU 4](#_Toc22855)

[CONCLUZII 14](#_Toc20914)

[BIBLIOGRAFII 15](#_Toc27099)

**INTRODUCERE**

Flutter este un framework open-source dezvoltat de Google, destinat creării de aplicații mobile, web și desktop dintr-o singură bază de cod. Folosind limbajul de programare Dart, Flutter permite dezvoltatorilor să creeze interfețe utilizator prietenoase și atractive, cu animații fluide și performanță ridicată. Datorită arhitecturii sale reactive, Flutter facilitează construirea de aplicații care se adaptează rapid la modificările stării, oferind o experiență de utilizare fluidă.

Unul dintre avantajele majore ale Flutter este biblioteca sa vastă de widget-uri, care permite personalizarea ușoară a interfeței utilizator. De asemenea, Flutter oferă suport pentru dezvoltarea multiplatformă, ceea ce înseamnă că aplicațiile create pot rula pe iOS, Android, web și desktop fără a necesita modificări semnificative ale codului. Ecosistemul Flutter include instrumente puternice, cum ar fi Flutter DevTools, care ajută la depanarea și optimizarea aplicațiilor. În plus, comunitatea activă de dezvoltatori contribuie constant cu pluginuri și pachete, extinzând astfel funcționalitățile framework-ului.

Prin combinația de performanță, flexibilitate și ușurință în utilizare, Flutter a devenit o alegere populară pentru dezvoltatorii care doresc să construiască aplicații moderne și de înaltă calitate.

**REALIZAREA PRACTICĂ A SARCINII DE LUCRU**

Actualitatea temei Flutter este evidentă în contextul dezvoltării rapide a tehnologiei mobile și a cerințelor tot mai mari ale utilizatorilor pentru aplicații performante și atractive. Într-o eră în care utilizatorii caută experiențe fluide și intuitive pe multiple platforme, Flutter se dovedește a fi un instrument esențial pentru dezvoltatori. Adoptarea crescută a Flutter în industrie este susținută de capacitatea sa de a permite dezvoltarea multiplatformă dintr-o singură bază de cod, reducând astfel timpul și costurile asociate cu crearea și întreținerea aplicațiilor. Aceasta este o soluție eficientă, mai ales pentru startup-uri și companii care doresc să lanseze rapid produse pe piață, fără a compromite calitatea.

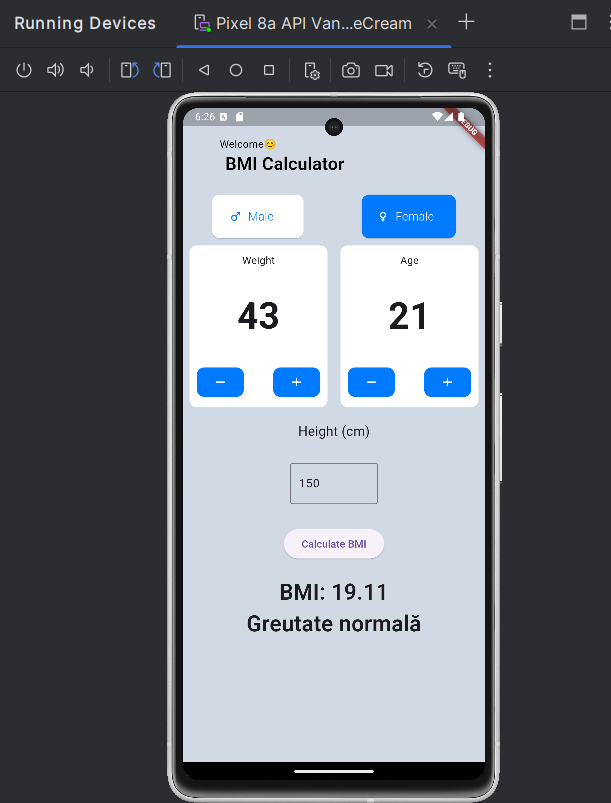
**Varianta 3**

**Realizați o aplicație mobilă ce va calcula indicele mesei corporale(IMC).**

Programul:

import 'package:flutter/material.dart';  
import 'package:flutter/services.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 headerColor = 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;  
  
 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) {  
 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";  
 }  
 } else {  
 if (esteBarbat) {  
 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 {  
 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,  
 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: TextStyle(color: Colors.black),  
 ),  
 ),  
 Container(  
 child: Text(  
 "BMI Calculator",  
 style: TextStyle(  
 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(() {  
 age++;  
 });  
 },  
 child: Icon(Icons.add, color: Colors.white, size: 20),  
 ),  
 ],  
 ),  
 ],  
 ),  
 ),  
 ],  
 ),  
 Container(  
 margin: EdgeInsets.symmetric(vertical: 20, horizontal: 50),  
 child: Column(  
 children: [  
 Text(  
 "Height (cm)",  
 style: TextStyle(fontSize: 18),  
 ),  
 Padding(  
 padding: EdgeInsets.only(top: 30, bottom: 30),  
 child: SizedBox(  
 width: 120,  
 child: TextField(  
 onChanged: (value) {  
 height = double.tryParse(value) ?? 0;  
 },  
 keyboardType: TextInputType.number,  
 decoration: InputDecoration(  
 border: OutlineInputBorder(),  
 ),  
 ),  
 ),  
 ),  
 ElevatedButton(  
 onPressed: () {  
 setState(() {  
 calculate(height, weight);  
 raspuns = getIMCInterpretation(imc, age, selectM);  
 });  
 },  
 child: Text("Calculate BMI"),  
 ),  
 ],  
 ),  
 ),  
 Container(  
 child: Text(  
 "BMI: $formattedImc\n$raspuns",  
 style: TextStyle(fontSize: 30, fontWeight: FontWeight.w600),  
 textAlign: TextAlign.center,  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

Rezultat:



Flutter beneficiază de suport constant din partea Google, ceea ce asigură actualizări regulate și îmbunătățiri ale performanței. Comunitatea activă de dezvoltatori contribuie la extinderea funcționalităților framework-ului prin crearea de pluginuri și biblioteci, astfel încât să răspundă nevoilor diverse ale dezvoltatorilor.

Aplicația creată este un calculator de indice de masă corporală (IMC), concepută pentru a oferi utilizatorilor o modalitate simplă și interactivă de a-și evalua starea de sănătate pe baza greutății, înălțimii, vârstei și sexului. Aceasta permite utilizatorilor să introducă informațiile necesare și să obțină rapid calculul IMC-ului, împreună cu o interpretare a rezultatelor.

**CONCLUZII**

În concluzie, aplicația de calculare a indicelui de masă corporală (IMC) realizată cu Flutter ilustrează eficiența și versatilitatea acestui framework în dezvoltarea de soluții mobile. Prin capacitatea sa de a integra funcționalități complexe într-o interfață intuitivă, Flutter oferă dezvoltatorilor instrumentele necesare pentru a crea aplicații care nu doar îndeplinesc cerințele utilizatorilor, ci le și depășesc așteptările.

Implementarea unei aplicații de sănătate precum aceasta este relevantă în contextul actual, în care tot mai multe persoane își doresc să își monitorizeze starea de sănătate și să ia decizii informate. Prin intermediul unei interfețe prietenoase, utilizatorii pot obține rapid informații esențiale, ceea ce le poate influența stilul de viață și alegerile alimentare.

De asemenea, utilizarea Flutter permite o dezvoltare rapidă și eficientă, cu o bază de cod unificată pentru multiple platforme, ceea ce reduce costurile și timpul de dezvoltare. În plus, comunitatea activă din jurul Flutter contribuie la extinderea constantă a funcționalităților, asigurând că dezvoltatorii au acces la cele mai noi tehnologii și resurse.

În final, aplicația nu doar că demonstrează abilitățile tehnice ale dezvoltatorilor, ci și relevanța și importanța soluțiilor software în domeniul sănătății. Aceasta subliniază rolul esențial al tehnologiei în îmbunătățirea calității vieții utilizatorilor, oferind instrumente utile pentru gestionarea sănătății personale.

**BIBLIOGRAFII**

1. <https://www.figma.com/design/NgvT5b29DcWtycAtGzNE3K/Laboratoare-2024?node-id=443-677&node-type=frame&t=yHVxxDKJRi29Ywfm-0>