**LUCRARE DE ATESTAT**

Teoria Grafurilor + Test Grilă

Absolvent: Profesor coordonator:

Filipoiu Patrick – Lari Mureșan Vasile Ciprian

Bistrița 2021

**Cuprins**

[**INTRODUCERE 3**](#_Toc72357760)

[**MODUL DE IMPLEMENTARE 4**](#_Toc72357761)

[**MANUALUL DE UTILIZARE 5**](#_Toc72357762)

[**COD SURSĂ 10**](#_Toc72357763)

[**BIBLIOGRAFIE 19**](#_Toc72357764)

# **INTRODUCERE**

Tema proiectului o constituie dezvoltarea unui program de **prezentare a grafurilor**, cu interfață vizuala realizată în programul **Visual Studio**, **Windows Froms App**. Aplicația este destinată elevilor de clasa a XI-a care studiază capitolul de Teoria Grafurilor și profesorilor care predau la clasa a XI-a.

În contextul societății actuale, caracterizată printr-o explozie informațională fără precedent în istoria omenirii, sistemele informatice reprezintă unul din elementele fundamentale care generează și controlează fluxurile informaționale la nivel micro și macroeconomic.

Limbajul C# a fost dezvoltat de o echipă restrânsă de ingineri de la Microsoft. Acesta permite programarea structurată, modulară şi orientată obiectual, conform perceptelor moderne ale programării profesioniste. Principiile de bază ale programării pe obiecte (INCAPSULARE, MOSTENIRE, POLIMORFISM) sunt elemente fundamentale ale programării C#.

Aplicația este organizată pe 4 secțiuni:

1. **Acasă**: În această secțiune se află o scurtă descriere a aplicației.
2. **Teorie**: Această secțiune este dedicată teoriei grafurilor, începând cu noțiunile de grafuri și terminându-se cu explicarea și codul sursă al unor algoritmi de parcurgere.
3. **Test**: Zona de desfășurare a acestei secțiuni se face într-un browser. Această secțiune a fost realizată cu scopul testării cunoștințelor acumulate la secțiunea teorie.
4. **Despre**: Secțiunea despre reprezintă sfârșitul proiectului, oferind informații despre cine a creat proiect și de către cine a fost coordonat.

# **MODUL DE IMPLEMENTARE**

* **Interfața vizuală**

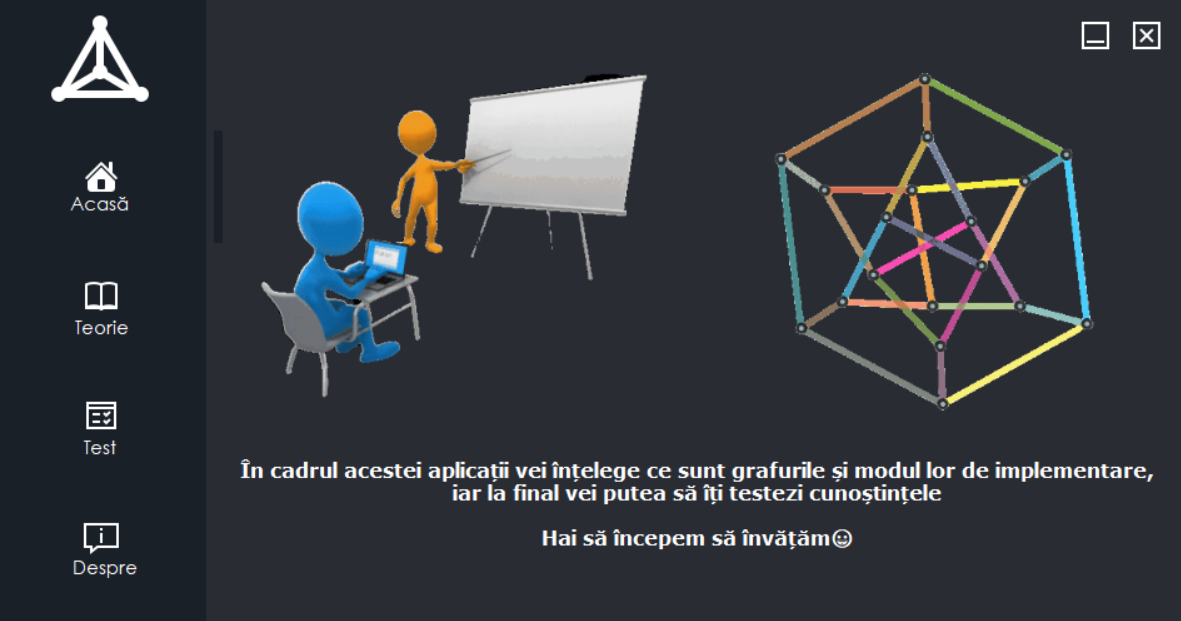
Inițial, am creat interfața vizuală a programului, creând toate elementele necesare ale acestei aplicații. Am început prin a crea un design modern și simplist al feresteri **Main\_Form.cs[Design]**. Am continuat prin a crea câte un UserControl pentru fiecare secțiune în parte, de la secțiunea **Acasă** și până la secțiunea **Despre**, și de a le face cât mai sugestive pentru a putea fi înțelese de cât mai multe persoane.

* **Funcționalitatea aplicației**

După terminarea design-ului aplicației am început să lucrez la partea de funcționalitate a acesteia. Asocierea evenimentului **Click** butoanelor de pe **Main\_Form.cs[Design]** a fost primul pas. În cadrul evenimentului **Click**, se face comutarea între UserControl-ul fiecărei secțiuni. După am creat câte o pagină web pentru fiecare capitol din cadrul Teoriei Grafurilor, utilizând **HTML** și **CSS**, pe care le-am inserat într-un **webBrowser** din cadrul secțiunii Teorie. Pentru a creat un design modern și testului grilă, am ales să folosesc **HTML**, **CSS** și **Javascript**. Întrebările din testul grilă se generează aleatoriu, precum și ordinea răspunsurilor acestor, iar la final am creat un tabel care oferă informații colectate pe parcursul testului precum numarul de întrebări corecte sau greșite.

# **MANUALUL DE UTILIZARE**

1. **Acasă**În această secțiune, utilizatorul poate să vadă o scurtă introducere a aplicației astfel încât să își fac o mică ideea despre aplicație.



1. **Teorie**

Utilizatorul este întâmpinat în această secțiune de un meniu cu butoane sugestive pentru navigarea între paginile cu teorie. Butoanele **Grafuri Neorientate, Grafuri Orientate, Grafuri Ponderate, Arbori cu Rădăcină** conțin submeniuri care pot fi accesate. De asemenea, s-ar putea ca uneori bordura de la paginile web să nu aibă dimensiunea prestabilită, astfel, dacă se întâmplă acest lucru, se mai face o dată click pe același buton care duce la pagina respectivă.

O imagine care conține text, captură de ecran, monitor

Descriere generată automat

O imagine care conține text, captură de ecran, monitor

Descriere generată automat

O imagine care conține text

Descriere generată automat

1. **Test**

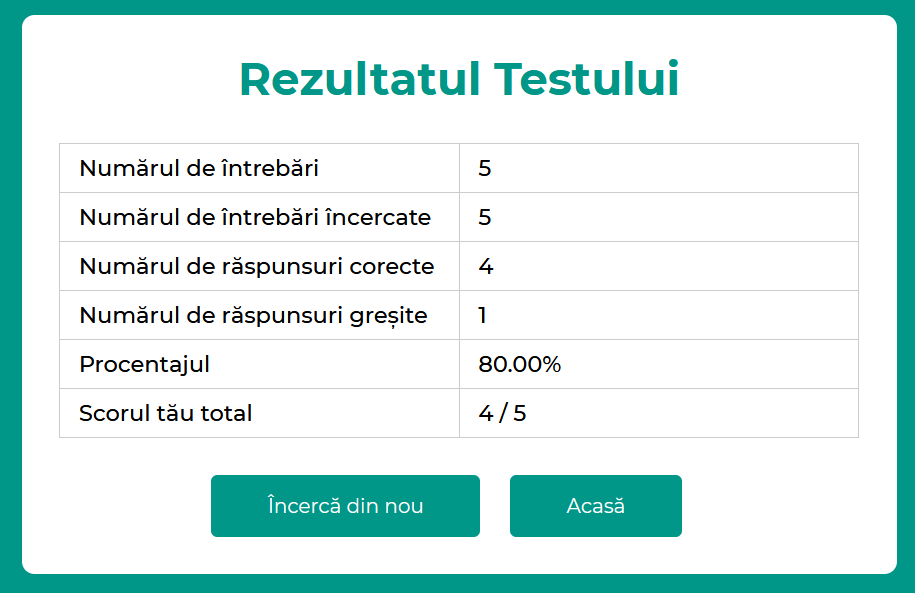
În urma evenimentului de click pe butonul **Test**, utilizatorul va fi condus pe o pagină web care se va deschide în browser-ul prestabilit pe sistemul de operare al acestuia. Pe respectiva pagină, utilizatorul va fi întâmpinat de o interfață care ii va oferi detalii despre testul respectiv. Pentru a începe testul, utilizatorul trebuie să apese click pe butonul **Începe testul**. În urma acestui eveniment, utilizatorului i se vor adresa o serie de întrebări la care trebuie să raspundă. În cazul în care utilizatorul a răspuns corect, varianta acestuia se va color cu verde și va fi înregistrată mai jos printr-o **bifă** pe fundal verde. În caz contrar, utilizatorului i se va color varianta aleasă cu roșu și i se va arăta varianta corectă, urmând ca pe urmă răspunsul lui să fie înregistrat mai jos printr-un **X** pe fundal roșu. La final, utilizatorul va putea vedea scorul lui într-un tabel.

O imagine care conține text

Descriere generată automat

O imagine care conține text

Descriere generată automat



1. **Despre**

În această secțiune, utilizatorului i se vor oferi informații despre autroul proiectului și despre coordonatorul acestuia.

O imagine care conține text

Descriere generată automat

# **COD SURSĂ**

* Prezentarea codului **C#**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Atestat\_Informatica\_Grafuri

{

public partial class Main\_Form : Form

{

public Main\_Form()

{

InitializeComponent();

panelSelector.Height = buttonHome.Height;

panelSelector.Top = buttonHome.Top;

homeUserControl.BringToFront();

}

private void buttonHome\_Click(object sender, EventArgs e)

{

panelSelector.Height = buttonHome.Height;

panelSelector.Top = buttonHome.Top;

homeUserControl.BringToFront();

}

private void buttonTheory\_Click(object sender, EventArgs e)

{

panelSelector.Height = buttonTheory.Height;

panelSelector.Top = buttonTheory.Top;

theoryUserControl.BringToFront();

}

private void buttonTest\_Click(object sender, EventArgs e)

{

panelSelector.Height = buttonTheory.Height;

panelSelector.Top = buttonTest.Top;

System.Diagnostics.Process.Start(Application.StartupPath + @"\Test\" + "Multiple\_Choice\_Test.html");

}

private void buttonAbout\_Click(object sender, EventArgs e)

{

panelSelector.Height = buttonAbout.Height;

panelSelector.Top = buttonAbout.Top;

aboutUserControl.BringToFront();

}

private void buttonMinimize\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void buttonExit\_Click(object sender, EventArgs e)

{

Application.Exit();

}

}

}

* Prezentarea codului **Javascript**

const questionNumber = document.querySelector(".question-number");

const questionText = document.querySelector(".question-text");

const optionContainer = document.querySelector(".option-container");

const answersIndicatorContainer = document.querySelector(".answers-indicator");

const homeBox = document.querySelector(".home-box");

const quizBox = document.querySelector(".quiz-box");

const resultBox = document.querySelector(".result-box");

let questionCounter = 0;

let currentQuestion;

let availableQuestions = [];

let availableOptions = [];

let correctAnswers = 0;

let attempt = 0;

//push the question into availableQuestions Array

function setAvailableQuestions() {

const totalQuestion = quiz.length;

for (let i = 0; i < totalQuestion; ++i){

availableQuestions.push(quiz[i]);

}

}

//set question number and question and options

function getNewQuestion(){

//set question number

questionNumber.innerHTML = "Întrebarea " + (questionCounter + 1) + " din " + quiz.length;

//set question text

//get random question

const questionIndex = availableQuestions[Math.floor(Math.random() \* availableQuestions.length)]

currentQuestion = questionIndex;

questionText.innerHTML = currentQuestion.q;

//get the position of 'questionIndex' from the availableQuestions Array

const index1 = availableQuestions.indexOf(questionIndex);

//remove the 'questionIndex' from the availableQuestions Array

availableQuestions.splice(index1, 1);

//set options

//get the length of options

const optionLen = currentQuestion.options.length;

//push options into availableOptions Array

for (let i = 0; i < optionLen; ++i){

availableOptions.push(i);

}

optionContainer.innerHTML = '';

let animationDelay = 0.15;

//create options in html

for (let i = 0; i < optionLen; ++i){

//random option

const optionIndex = availableOptions[Math.floor(Math.random() \* availableOptions.length)]

//get the position of 'optionIndex' from the availableOptions Array

const index2 = availableOptions.indexOf(optionIndex);

//remove the 'optionIndex' from the availableOptions Array

availableOptions.splice(index2, 1);

const option = document.createElement("div");

option.innerHTML = currentQuestion.options[optionIndex];

option.id = optionIndex;

option.style.animationDelay = animationDelay + 's';

animationDelay = animationDelay + 0.15;

option.className = "option";

optionContainer.appendChild(option);

option.setAttribute("onclick", "getResult(this)");

}

questionCounter++;

}

//get the result of current attempt question

function getResult(element){

const id = parseInt(element.id);

//get the answer by comparing the id of clicked option

if (id === currentQuestion.answer){

//set the green color to the correct option

element.classList.add("correct");

//add indicator to correct mark

updateAnswerIndicator("correct");

correctAnswers++;

}

else {

//set the red color to the incorrect option

element.classList.add("wrong");

//add indicator to wrong mark

updateAnswerIndicator("wrong");

//if the answer is incorrect, show the correct option by adding green color into the correct option

const optionLen = optionContainer.children.length;

for (let i = 0; i < optionLen; ++i){

if (parseInt(optionContainer.children[i].id) === currentQuestion.answer)

optionContainer.children[i].classList.add("correct");

}

}

attempt++;

unclickableOptions();}

# **BIBLIOGRAFIE**

* <https://www.pbinfo.ro/>
* <https://stackoverflow.com/>