Proiect

*la cursul*

Programare Orientata spre Obiecte(POO)

Sesiunea : **Februarie 2023**

Soft de generare si gestionare a meditatiilor(eMeditatii)

* **Coordonator** : Conf.univ.dr. Puchianu Crenguța

Conf.univ.dr. Șerban Cristina

* **Realizator** : Student Tufan Ionut-Petrisor Info2-gr3

Student Tudor Vladut-Alexandru Info2-gr3

Cuprins

[**I.** **Enuntul Problemei** 3](#_Toc126539270)

[**II Diagrama UML** 4](#_Toc126539271)

[**III Prezentarea soft-ului** 4](#_Toc126539272)

[1.Login.java 4](#_Toc126539273)

[3.StudentFrame.java 14](#_Toc126539274)

[4.TeacherFrame.java 33](#_Toc126539275)

[5.Professor.java 56](#_Toc126539276)

[6.Student.java 56](#_Toc126539277)

[7.Review.java 57](#_Toc126539278)

[8.Request.java 58](#_Toc126539279)

[9.User.java 58](#_Toc126539280)

[10.UserRegistration.java 60](#_Toc126539281)

[11.Validate.java 61](#_Toc126539282)

[12.RegisterRequests.java 62](#_Toc126539283)

[13. RegisterReviews.java 62](#_Toc126539284)

[14.Users.txt 63](#_Toc126539285)

[15.Requests.txt 63](#_Toc126539286)

[16.Reviews.txt 63](#_Toc126539287)

[17.Sessions.txt 63](#_Toc126539288)

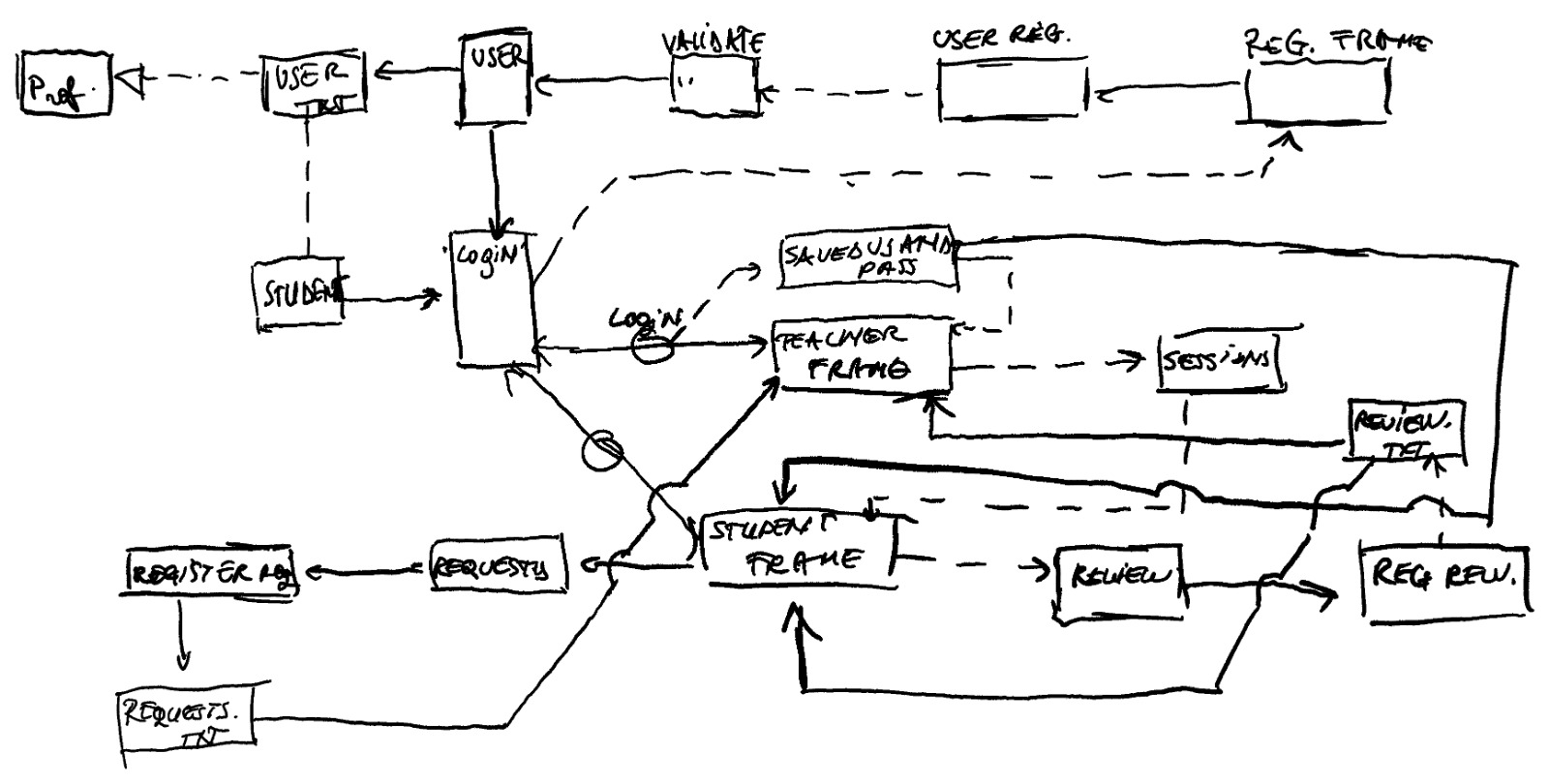
[18.savedUsAndPass 63](#_Toc126539289)

[**IV Bibliografie** 64](#_Toc126539290)

# **Enuntul Problemei**

Sistemul EMeditatii realizeaza conexiunea intre elevi/cursanti si profesori/tutori pentru a stabili ora, data, locul si pretul lectiilor particulare pe care vor sa le urmeze cursantii. Profesorii au posibilitatea de a-şi crea conturi in aplicatie, de a modifica datele personale, de a adauga certificate care atesta experinta profesionala şi de a gestiona materiile predate. In plus orice profesor logat in aplicatie, poate adauga poze, accepta/respinge cererile cursantilor si anula sedintele. Dupa ce profesorul accepta/respinge cererea sau anuleaza sedinta - sistemul o afiseaza cursantului cand se logheaza in ziua respectiva. Similar, cursantii au posibilitatea de a-si crea conturi in aplicatie, de a modifica datele personale si de a gestiona materiile preferate. In plus orice cursant logat in aplicatie, poate adauga poza de profil, trimite cereri catre profesori si anula sedintele stabilite anterior. Dupa ce cursantul trimite o cerere sau anuleaza o sedinta – sistemul o afiseaza profesorului cand se logheaza in ziua respectiva. Dupa terminarea pregatirii (meditatiilor) cu un profesor, orice cursant poate sa completeze o recenzie cu date despre modul in care s-au desfasurat sedintele, daca recomanda profesorul si altor cursanti si ce nota ii acorda. Profesorul poate cere aplicatiei sa listeze la imprimanta recenziile primite.

# **II Diagrama UML**

****

# **III Prezentarea soft-ului**

## 1.Login.java

package emeditatii;

import java.awt.\*;

import java.awt.event.\*;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import javax.swing.\*;

public class Login extends JFrame implements ActionListener {

public String usernameString;

JPanel panel;

JLabel user\_label, password\_label, message;

JTextField userName\_text;

JPasswordField password\_text;

JButton login, cancel;

Login() {

Dimension dim = Toolkit.getDefaultToolkit().getScreenSize();

this.setLocation(dim.width/3-this.getSize().width, dim.height/4-this.getSize().height/3);

// Username Label

user\_label = new JLabel();

user\_label.setText("Username :");

user\_label.setFont(new Font("Courier", Font.BOLD,30));

userName\_text = new JTextField();

userName\_text.setFont(new Font("Courier", Font.BOLD,20));

user\_label.setHorizontalAlignment(SwingConstants.CENTER);

// Password Label

password\_label = new JLabel();

password\_label.setText("Password :");

password\_text = new JPasswordField();

password\_text.setFont(new Font("Courier", Font.BOLD,20));

password\_label.setFont(new Font("Courier", Font.BOLD,30));

password\_label.setHorizontalAlignment(SwingConstants.CENTER);

// Submit

login = new JButton("Login");

panel = new JPanel(new GridLayout (3, 2));

panel.add(user\_label);

panel.add(userName\_text);

panel.add(password\_label);

panel.add(password\_text);

message = new JLabel();

panel.add(login);

JButton register = new JButton("Register");

register.setBounds(50,100,95,30);

panel.add(register);

usernameString = userName\_text.getText();

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// Adding the listeners to components..

login.addActionListener(this);

add(panel, BorderLayout.CENTER);

setTitle("EMeditatii");

setSize(450,350);

setVisible(true);

ImageIcon img = new ImageIcon("E:\\education.png");

setIconImage(img.getImage());

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

register.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent ae) {

dispose();

new RegisterFrame();

}

});

login.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent ae) {

String savedUseString=userName\_text.getText();

String savedpaString=password\_text.getText();

try{

if(isLoginValid(userName\_text.getText(), password\_text.getText())){

BufferedWriter writer = new BufferedWriter(new FileWriter("savedUsAndPass.txt", false));

writer.write(savedUseString+","+savedpaString);

writer.newLine();

writer.close();

JOptionPane.showMessageDialog(null, "Login successfully...");

if(isProfessor(userName\_text.getText())){

dispose();

new TeacherFrame();

}

else{

dispose();

new StudentFrame();

}

}else{

JOptionPane.showMessageDialog(null, "Username or password is incorrect!");

}

}catch(IOException ex){

JOptionPane.showMessageDialog(null, ex.getMessage());

}

}

});

}

public boolean isLoginValid(String username, String password) throws IOException {

BufferedReader reader = new BufferedReader(new FileReader("Users.txt"));

String line;

while ((line = reader.readLine()) != null) {

String[] fields = line.split(",");

if (fields[0].equals(username) && fields[2].equals(password)) {

reader.close();

return true;

}

}

reader.close();

return false;

}

public boolean isProfessor(String username) throws IOException {

BufferedReader reader = new BufferedReader(new FileReader("Users.txt"));

String line;

while ((line = reader.readLine()) != null) {

String[] fields = line.split(",");

if (fields[0].equals(username)) {

reader.close();

return Boolean.parseBoolean(fields[5]);

}

}

reader.close();

return false;

}

public static void main(String[] args) {

new Login();

}

public void actionPerformed(ActionEvent ae) {

}

}

**2.RegisterFrame.Java**

package emeditatii;

import java.awt.\*;

import java.awt.event.\*;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

import javax.swing.\*;

public class RegisterFrame extends JFrame implements ActionListener {

private final Container c;

private final JLabel title;

private final JLabel name;

private final JTextField tname;

private final JLabel mno;

private final JTextField tmno;

private final JLabel gender;

private final JRadioButton male;

private final JRadioButton female;

private final ButtonGroup gengp;

private final JLabel add;

private final JTextArea tadd;

private final JCheckBox term;

private final JButton sub;

private final JButton reset;

private final JTextArea tout;

private final JLabel res;

private final JTextArea resadd;

private final JLabel password;

private final JPasswordField tpassword;

private final JCheckBox profesor;

private final JButton profesorButon;

public RegisterFrame() {

ImageIcon img = new ImageIcon("E:\\education.png");

setIconImage(img.getImage());

setTitle("Registration Page");

setBounds(300, 90, 900, 600);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setResizable(false);

c = getContentPane();

c.setLayout(null);

title = new JLabel("Registration Form");

title.setFont(new Font("Serif", Font.BOLD, 30));

title.setSize(300, 40);

title.setLocation(300, 30);

c.add(title);

name = new JLabel("Username");

name.setFont(new Font("Serif", Font.PLAIN, 20));

name.setSize(100, 20);

name.setLocation(100, 100);

c.add(name);

password = new JLabel("Password");

password.setFont(new Font("Serif", Font.PLAIN, 20));

password.setSize(200, 30);

password.setLocation(100, 345);

c.add(password);

tname = new JTextField();

tname.setFont(new Font("Serif", Font.PLAIN, 15));

tname.setSize(190, 20);

tname.setLocation(200, 100);

c.add(tname);

tpassword = new JPasswordField();

tpassword.setFont(new Font("Serif", Font.PLAIN, 15));

tpassword.setSize(190, 20);

tpassword.setLocation(200, 350);

c.add(tpassword);

mno = new JLabel("Mobile");

mno.setFont(new Font("Serif", Font.PLAIN, 20));

mno.setSize(100, 20);

mno.setLocation(100, 150);

c.add(mno);

tmno = new JTextField();

tmno.setFont(new Font("Serif", Font.PLAIN, 15));

tmno.setSize(190, 20);

tmno.setLocation(200, 150);

c.add(tmno);

gender = new JLabel("Gender");

gender.setFont(new Font("Serif", Font.PLAIN, 20));

gender.setSize(100, 20);

gender.setLocation(100, 200);

c.add(gender);

male = new JRadioButton("Male");

male.setFont(new Font("Serif", Font.PLAIN, 15));

male.setSelected(true);

male.setSize(75, 20);

male.setLocation(200, 200);

c.add(male);

female = new JRadioButton("Female");

female.setFont(new Font("Serif", Font.PLAIN, 15));

female.setSelected(false);

female.setSize(80, 20);

female.setLocation(275, 200);

c.add(female);

gengp = new ButtonGroup();

gengp.add(male);

gengp.add(female);

add = new JLabel("Address");

add.setFont(new Font("Serif", Font.PLAIN, 20));

add.setSize(100, 20);

add.setLocation(100, 300);

c.add(add);

tadd = new JTextArea();

tadd.setFont(new Font("Serif", Font.PLAIN, 15));

tadd.setSize(200, 75);

tadd.setLocation(200, 300);

tadd.setLineWrap(true);

c.add(tadd);

term = new JCheckBox("Accept Terms And Conditions.");

term.setFont(new Font("Serif", Font.PLAIN, 15));

term.setSize(250, 20);

term.setLocation(150, 400);

c.add(term);

profesor = new JCheckBox("Professor");

profesor.setFont(new Font("Serif", Font.PLAIN, 15));

profesor.setSize(250, 20);

profesor.setLocation(150, 380);

c.add(profesor);

sub = new JButton("Register");

sub.setFont(new Font("DIALOG", Font.BOLD, 15));

sub.setSize(100, 20);

sub.setLocation(150, 450);

sub.addActionListener((ActionListener) this);

c.add(sub);

reset = new JButton("Reset");

reset.setFont(new Font("DIALOG", Font.BOLD, 15));

reset.setSize(100, 20);

reset.setLocation(300, 450);

reset.addActionListener((ActionListener) this);

c.add(reset);

tout = new JTextArea();

tout.setFont(new Font("Serif", Font.PLAIN, 15));

tout.setSize(300, 400);

tout.setLocation(500, 100);

tout.setLineWrap(true);

tout.setEditable(false);

c.add(tout);

res = new JLabel("");

res.setFont(new Font("Serif", Font.PLAIN, 20));

res.setSize(500, 25);

res.setLocation(100, 500);

c.add(res);

resadd = new JTextArea();

resadd.setFont(new Font("Serif", Font.PLAIN, 15));

resadd.setSize(200, 75);

resadd.setLocation(580, 175);

resadd.setLineWrap(true);

c.add(resadd);

profesorButon = new JButton("Click here to go back!");

profesorButon.setFont(new Font("DIALOG", Font.BOLD, 15));

profesorButon.setSize(250, 20);

profesorButon.setLocation(150, 425);

profesorButon.addActionListener((ActionListener) this);

c.add(profesorButon);

setVisible(true);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

profesorButon.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent ae) {

dispose();

new Login();

}});

}

public boolean isUsernameAndEmailExists(String username, String email) throws IOException {

BufferedReader reader = new BufferedReader(new FileReader("Users.txt"));

String line;

while ((line = reader.readLine()) != null) {

String[] fields = line.split(",");

if (fields[0].equals(username) || fields[1].equals(email)) {

reader.close();

return true;

}

}

reader.close();

return false;

}

//--------------------method for selecting the disciplines for professors-----------------

public String showDisciplineSelection() {

String[] options = {"Computer Science", "Mathematics", "Physics"};

JOptionPane pane = new JOptionPane(

"Select a discipline:",

JOptionPane.QUESTION\_MESSAGE,

JOptionPane.DEFAULT\_OPTION,

null,

options,

options[0]

);

JDialog dialog = pane.createDialog(null, "Discipline Selection");

dialog.setAlwaysOnTop(true);

dialog.setVisible(true);

Object selectedValue = pane.getValue();

return (String) selectedValue;

}

//---------------------------method to select the faculty the student is in-----------------

public String showFacultySelection() {

String[] options = {"Informatics", "Mathematics", "Informatics and Mathematics"};

JOptionPane pane = new JOptionPane(

"Select a Faculty:",

JOptionPane.QUESTION\_MESSAGE,

JOptionPane.DEFAULT\_OPTION,

null,

options,

options[0]

);

JDialog dialog = pane.createDialog(null, "Discipline Selection");

dialog.setAlwaysOnTop(true);

dialog.setVisible(true);

Object selectedValue = pane.getValue();

return (String) selectedValue;

}

//-----------------------------------------------------------------------------

public void actionPerformed(ActionEvent e) {

Boolean isCorrect=true;

Validate valid= new Validate();

String username = tname.getText();

String emailAdress= tadd.getText();

String password = tpassword.getText();

String gender ="";

String mobileNumber=tmno.getText();

Boolean isProfessor=false;

String discipline;

String faculty;

UserRegistration uRegistration = new UserRegistration();

Professor newProfessor;

Student newStudent;

if (e.getSource() == sub) {

if (term.isSelected()) {

try{

try{

valid.validateUsername(username);

}catch(Exception exc){

JOptionPane.showMessageDialog(null, exc.getMessage());

tname.setText("");

res.setText("Registration failed, please try again!");

isCorrect=false;

}

try{

valid.validatePassword(password);

}catch(Exception exc){

JOptionPane.showMessageDialog(null, exc.getMessage());

tpassword.setText("");

res.setText("Registration failed, please try again!");

isCorrect=false;

}

try{

valid.validateEmail(emailAdress);

}catch(Exception exc){

JOptionPane.showMessageDialog(null, exc.getMessage());

tadd.setText("");

res.setText("Registration failed, please try again!");

isCorrect=false;

}

try{

valid.validateMobileNumber(mobileNumber);

}catch(Exception exc){

JOptionPane.showMessageDialog(null, exc.getMessage());

tmno.setText("");

res.setText("Registration failed, please try again!");

isCorrect=false;

}

if(isUsernameAndEmailExists(username, emailAdress) ){

JOptionPane.showMessageDialog(null, "Username or Email already exists");

res.setText("Registration failed, please try again!");

}else if(isCorrect){

String data1;

String data = "Name : " + tname.getText() + "\n"

+ "Mobile : " + tmno.getText() + "\n";

if (male.isSelected()){

data1 = "Gender : Male" + "\n";

gender="Male";

}else{

data1 = "Gender : Female" + "\n";

gender="Female";

}

String data3 = "Address : " + tadd.getText() + "\n";

String data4 = "Password : " + tpassword.getText();

tout.setText(data + data1 + data3 + data4);

tout.setEditable(false);

try{

if(profesor.isSelected()){

isProfessor=true;

discipline=showDisciplineSelection();

newProfessor = new Professor(username, emailAdress, password, gender, mobileNumber, isProfessor, discipline);

uRegistration.registerProfessor("Users.txt", newProfessor);

}else{

faculty=showFacultySelection();

newStudent = new Student(username, emailAdress, password, gender, mobileNumber, isProfessor, faculty);

uRegistration.registerStudent("Users.txt", newStudent);

}

res.setText("Registration Successfully...");

}catch(IOException ex){

JOptionPane.showMessageDialog(null, ex);

JOptionPane.showMessageDialog(null , "Users.txt cannot be oppend");

}

}

}catch(IOException Exception){

JOptionPane.showMessageDialog(null, Exception);

JOptionPane.showMessageDialog(null , "Users.txt cannot be oppend");

}

}

else {

tout.setText("");

resadd.setText("");

res.setText("Please accept the terms & conditions...");

}

}

else if (e.getSource() == reset) {

String def = "";

tname.setText(def);

tadd.setText(def);

tmno.setText(def);

res.setText(def);

tout.setText(def);

term.setSelected(false);

resadd.setText(def);

}

}

}

class Registration {

public static void main(String[] args) throws Exception

{

RegisterFrame f = new RegisterFrame();

}

}

## 3.StudentFrame.java

package emeditatii;

import java.awt.\*;

import java.awt.event.\*;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

import javax.swing.\*;

public class StudentFrame extends JFrame{

private final JFrame frame;

private final JPanel panel;

private final JLabel nume;

private final JLabel materie;

private final JTextField tnume;

private final JLabel titluCreeazaCerere;

private final JLabel bunVenit;

private final String materii[] =

{

"Geometry", "Automata Theory" , "Operating Sistems"

};

private final String ora[] =

{

"14","15","16","17","18" , "19"

};

private final String minute[] =

{

"00","30"

};

private final String loc[] =

{

"Sala 10" , "Sala 15","Sala 20" , "Sala 30"

};

private final String pret[] =

{

"100","120", "150"

};

private final JComboBox locBox;

private final JComboBox pretBox;

private final JComboBox oraBox;

private final JComboBox minuteBox;

private final JComboBox materiiBox;

private final String dates[]

= { "1", "2", "3", "4", "5",

"6", "7", "8", "9", "10",

"11", "12", "13", "14", "15",

"16", "17", "18", "19", "20",

"21", "22", "23", "24", "25",

"26", "27", "28", "29", "30",

"31" };

private final String months[]

= { "Jan", "Feb", "Mar", "Apr",

"May", "Jun", "July", "Aug",

"Sup", "Oct", "Nov", "Dec" };

private final String years[]

= { "2023" , "2024" , "2025" };

private final JComboBox date;

private final JComboBox month;

private final JComboBox year;

private final JLabel dataText;

private final JLabel informatii;

private final String listaSedinte[] = { " ", " ", " " };

private final JList listaSed;

private final JList listaRec;

private final JLabel titluDatePersonale;

private final JLabel numeHome;

private final JLabel passwordHome;

private final JLabel dataNasteriiHome;

private final JLabel adresaHome;

private final JLabel facultateHome;

private final JLabel creeazaRecenzie;

private final JLabel nota;

private final JComboBox notaBox;

private final String note[]

= { "1" , "2" , "3", "4" , "5", "6", "7", "8", "9", "10" };

private final JLabel materieRecenzie;

private final JComboBox materieBox;

private final JLabel recenzie;

private final JTextArea trecenzie;

private final JLabel trecenziileTale;

private final JTextField homeNume;

private final JTextField homePassword;

private final JTextField homeDataNasterii;

private final JTextField homeAdresa;

private final JTextField homeFacultate;

private final JLabel oraCreeazaCerere;

private final JLabel locCreeazaCerere;

private final JLabel pretCreeazaCerere;

private final JButton trimiteRecenzie;

private final String listaRecenzii[] = { "Recenzie 1" };

StudentFrame()

{

Login lg = new Login();

lg.hide();

frame = new JFrame("Student Frame");

panel = new JPanel();

frame.getContentPane();

bunVenit = new JLabel("Welcome, Student");

bunVenit.setFont(new Font("Serif", Font.BOLD, 30));

bunVenit.setSize(300, 40);

bunVenit.setLocation(500, 200);

panel.add(bunVenit);

informatii = new JLabel("You are on the Student Page!");

informatii.setFont(new Font("Serif", Font.PLAIN, 30));

informatii.setSize(500, 40);

informatii.setLocation(380, 230);

panel.add(informatii);

JButton logOut = new JButton("Log Out");

Dimension size = logOut.getPreferredSize();

logOut.setBounds(1, 560, 150, 100);

panel.add(logOut);

JButton recenzii = new JButton("Your Reviews");

recenzii.setBounds(1, 460, 150, 100);

panel.add(recenzii);

JButton creeazaRecenzii = new JButton("Leave a Review");

creeazaRecenzii.setBounds(1, 360, 150, 100);

panel.add(creeazaRecenzii);

JButton sedinte = new JButton("Sessions");

sedinte.setBounds(1, 260, 150, 100);

panel.add(sedinte);

JButton creeazaCerere = new JButton("Create Request");

creeazaCerere.setBounds(1, 160, 150, 100);

panel.add(creeazaCerere);

JButton home = new JButton("Home");

home.setBounds(1, 0, 150, 160);

panel.add(home);

Dimension dimension = Toolkit.getDefaultToolkit().getScreenSize();

int x = (int) ((dimension.getWidth() - frame.getWidth()) / 5);

int y = (int) ((dimension.getHeight() - frame.getHeight()) / 6);

frame.setLocation(x, y);

panel.setLayout(null);

panel.setBorder(BorderFactory.createEmptyBorder(100, 10, 10, 10));

frame.setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

frame.add(panel);

frame.setSize(1000, 700);

frame.setVisible(true);

ImageIcon img = new ImageIcon("E:\\education.png");

frame.setIconImage(img.getImage());

frame.setTitle("Student Page");

setVisible(true);

setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE);

logOut.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent ae) {

frame.setVisible(false);

frame.dispose();

new Login();

}});

JButton stergeSedinta = new JButton("Delete");

stergeSedinta.setBounds(810, 590, 100, 40);

stergeSedinta.setFont(new Font("Sans", Font.BOLD, 20));

panel.add(stergeSedinta);

titluCreeazaCerere = new JLabel("Create Request");

titluCreeazaCerere.setFont(new Font("Serif", Font.BOLD, 30));

titluCreeazaCerere.setSize(500, 40);

titluCreeazaCerere.setLocation(470, 100);

panel.add(titluCreeazaCerere);

nume = new JLabel("Name:");

nume.setFont(new Font("Serif", Font.BOLD, 30));

nume.setSize(300, 40);

nume.setLocation(300, 190);

panel.add(nume);

materie = new JLabel("Discipline:");

materie.setFont(new Font("Serif", Font.BOLD, 30));

materie.setSize(300, 40);

materie.setLocation(300, 250);

panel.add(materie);

JButton send = new JButton("Send");

send.setBounds(610, 540, 100, 40);

send.setFont(new Font("Sans",Font.BOLD,20));

panel.add(send);

tnume = new JTextField();

tnume.setFont(new Font("Serif", Font.BOLD, 20));

tnume.setSize(300, 40);

tnume.setLocation(405, 195);

panel.add(tnume);

materiiBox = new JComboBox(materii);

materiiBox.setFont(new Font("Arial", Font.BOLD, 20));

materiiBox.setSize(275, 30);

materiiBox.setLocation(430, 260);

panel.add(materiiBox);

dataText = new JLabel("Date:");

dataText.setFont(new Font("Serif", Font.BOLD, 30));

dataText.setSize(300, 40);

dataText.setLocation(300, 310);

panel.add(dataText);

date = new JComboBox(dates);

date.setFont(new Font("Serif", Font.PLAIN, 15));

date.setSize(100, 30);

date.setLocation(400, 318);

panel.add(date);

month = new JComboBox(months);

month.setFont(new Font("Serif", Font.PLAIN, 15));

month.setSize(100, 30);

month.setLocation(505, 318);

panel.add(month);

year = new JComboBox(years);

year.setFont(new Font("Serif", Font.PLAIN, 15));

year.setSize(100, 30);

year.setLocation(610, 318);

panel.add(year);

listaSed = new JList<String>(listaSedinte);

listaSed.setFont(new Font("Serif", Font.PLAIN, 20));

listaSed.setSize(680, 500);

listaSed.setLocation(230, 50);

panel.add(listaSed);

titluDatePersonale = new JLabel("Personal data");

titluDatePersonale.setFont(new Font("Serif", Font.BOLD, 30));

titluDatePersonale.setSize(500, 40);

titluDatePersonale.setLocation(470, 100);

panel.add(titluDatePersonale);

numeHome = new JLabel("Username:");

numeHome.setFont(new Font("Serif", Font.BOLD, 30));

numeHome.setSize(300, 40);

numeHome.setLocation(300, 200);

panel.add(numeHome);

passwordHome = new JLabel("Password:");

passwordHome.setFont(new Font("Serif", Font.BOLD, 30));

passwordHome.setSize(300, 40);

passwordHome.setLocation(300, 250);

panel.add(passwordHome);

adresaHome = new JLabel("Mail Adress:");

adresaHome.setFont(new Font("Serif", Font.BOLD, 30));

adresaHome.setSize(300, 40);

adresaHome.setLocation(300, 350);

panel.add(adresaHome);

dataNasteriiHome = new JLabel("Gender:");

dataNasteriiHome.setFont(new Font("Serif", Font.BOLD, 30));

dataNasteriiHome.setSize(300, 40);

dataNasteriiHome.setLocation(300, 300);

panel.add(dataNasteriiHome);

facultateHome = new JLabel("Faculty:");

facultateHome.setFont(new Font("Serif", Font.BOLD, 30));

facultateHome.setSize(300, 40);

facultateHome.setLocation(300, 400);

panel.add(facultateHome);

creeazaRecenzie = new JLabel("Create Review");

creeazaRecenzie.setFont(new Font("Serif", Font.BOLD, 30));

creeazaRecenzie.setSize(500, 40);

creeazaRecenzie.setLocation(470, 100);

panel.add(creeazaRecenzie);

nota = new JLabel("Rating");

nota.setFont(new Font("Serif", Font.BOLD, 30));

nota.setSize(500, 40);

nota.setLocation(300, 200);

panel.add(nota);

notaBox = new JComboBox(note);

notaBox.setFont(new Font("Arial", Font.BOLD, 20));

notaBox.setSize(275, 30);

notaBox.setLocation(435, 208);

panel.add(notaBox);

materieRecenzie = new JLabel("Discipline");

materieRecenzie.setFont(new Font("Serif", Font.BOLD, 30));

materieRecenzie.setSize(500, 40);

materieRecenzie.setLocation(300, 250);

panel.add(materieRecenzie);

materieBox = new JComboBox(materii);

materieBox.setFont(new Font("Arial", Font.BOLD, 20));

materieBox.setSize(275, 30);

materieBox.setLocation(435, 258);

panel.add(materieBox);

recenzie = new JLabel("Your comment");

recenzie.setFont(new Font("Serif", Font.BOLD, 30));

recenzie.setSize(500, 40);

recenzie.setLocation(300, 300);

panel.add(recenzie);

trecenzie = new JTextArea();

trecenzie.setFont(new Font("Serif", Font.BOLD, 20));

trecenzie.setSize(550, 220);

trecenzie.setLocation(300, 350);

panel.add(trecenzie);

trecenziileTale = new JLabel("Your Reviews");

trecenziileTale.setFont(new Font("Serif", Font.BOLD, 30));

trecenziileTale.setSize(500, 40);

trecenziileTale.setLocation(470, 100);

panel.add(trecenziileTale);

homeNume = new JTextField();

homeNume.setFont(new Font("Serif", Font.BOLD, 20));

homeNume.setSize(250, 40);

homeNume.setLocation(490, 205);

panel.add(homeNume);

homeNume.setEnabled(false);

homePassword = new JTextField();

homePassword.setFont(new Font("Serif", Font.BOLD, 20));

homePassword.setSize(250, 40);

homePassword.setLocation(490, 255);

panel.add(homePassword);

homePassword.setEnabled(false);

homeDataNasterii = new JTextField();

homeDataNasterii.setFont(new Font("Serif", Font.BOLD, 20));

homeDataNasterii.setSize(250, 40);

homeDataNasterii.setLocation(490, 305);

panel.add(homeDataNasterii);

homeDataNasterii.setEnabled(false);

homeAdresa = new JTextField();

homeAdresa.setFont(new Font("Serif", Font.BOLD, 20));

homeAdresa.setSize(250, 40);

homeAdresa.setLocation(490, 355);

panel.add(homeAdresa);

homeAdresa.setEnabled(false);

homeFacultate = new JTextField();

homeFacultate.setFont(new Font("Serif", Font.BOLD, 20));

homeFacultate.setSize(250, 40);

homeFacultate.setLocation(490, 405);

panel.add(homeFacultate);

homeFacultate.setEnabled(false);

JButton modificaHome = new JButton("Modify Data");

modificaHome.setBounds(620, 470, 120, 30);

panel.add(modificaHome);

JButton salveazaButtonHome = new JButton("Save Data");

salveazaButtonHome.setBounds(500, 470, 120, 30);

panel.add(salveazaButtonHome);

oraCreeazaCerere = new JLabel("Time:");

oraCreeazaCerere.setFont(new Font("Serif", Font.BOLD, 30));

oraCreeazaCerere.setSize(300, 40);

oraCreeazaCerere.setLocation(300, 360);

panel.add(oraCreeazaCerere);

locCreeazaCerere = new JLabel("Place:");

locCreeazaCerere.setFont(new Font("Serif", Font.BOLD, 30));

locCreeazaCerere.setSize(300, 40);

locCreeazaCerere.setLocation(300, 410);

panel.add(locCreeazaCerere);

pretCreeazaCerere = new JLabel("Price:");

pretCreeazaCerere.setFont(new Font("Serif", Font.BOLD, 30));

pretCreeazaCerere.setSize(300, 40);

pretCreeazaCerere.setLocation(300, 460);

panel.add(pretCreeazaCerere);

oraBox = new JComboBox(ora);

oraBox.setFont(new Font("Arial", Font.BOLD, 20));

oraBox.setSize(70, 30);

oraBox.setLocation(430, 370);

panel.add(oraBox);

minuteBox = new JComboBox(minute);

minuteBox.setFont(new Font("Arial", Font.BOLD, 20));

minuteBox.setSize(70, 30);

minuteBox.setLocation(510, 370);

panel.add(minuteBox);

locBox = new JComboBox(loc);

locBox.setFont(new Font("Arial", Font.BOLD, 20));

locBox.setSize(180, 30);

locBox.setLocation(400, 420);

panel.add(locBox);

pretBox = new JComboBox(pret);

pretBox.setFont(new Font("Arial", Font.BOLD, 20));

pretBox.setSize(180, 30);

pretBox.setLocation(400, 470);

panel.add(pretBox);

trimiteRecenzie = new JButton("Send");

trimiteRecenzie.setBounds(730, 585, 120, 30);

panel.add(trimiteRecenzie);

trimiteRecenzie.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String numeReviewer;

String rating;

String comment;

String discipline;

String username = new String();

String line;

BufferedReader reader;

try {

reader = new BufferedReader(new FileReader("savedUsAndPass.txt"));

while ((line = reader.readLine()) != null) {

String fields[]=line.split(",");

username=fields[0];

}

}catch (IOException e1) {

e1.printStackTrace();

}

numeReviewer=username;

rating=(String)notaBox.getSelectedItem();

comment=trecenzie.getText();

discipline=(String)materieBox.getSelectedItem();

Review review = new Review(numeReviewer, rating, comment, discipline);

RegisterReviews rr = new RegisterReviews();

try{

rr.RegisterRev("Reviews.txt", review);

JOptionPane.showMessageDialog(null, "Your review has been sent");

trecenzie.setText("");

}catch(IOException exc){

exc.getMessage();

}

}

});

listaRec = new JList<String>(listaRecenzii);

listaRec.setFont(new Font("Serif", Font.PLAIN, 15));

listaRec.setSize(680, 350);

listaRec.setLocation(230, 140);

panel.add(listaRec);

recenzii.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String username = new String();

String line;

BufferedReader reader;

try {

reader = new BufferedReader(new FileReader("savedUsAndPass.txt"));

while ((line = reader.readLine()) != null) {

String fields[]=line.split(",");

username=fields[0];

}

}catch (IOException e1) {

e1.printStackTrace();

}

int i=0;

try{

reader = new BufferedReader(new FileReader("Reviews.txt"));

String[] split;

while ((line = reader.readLine()) != null) {

split = line.split(",");

if(split[0].equals(username)){

listaRecenzii[i]=line;

i++;

}

}

}catch(IOException exc){

exc.getMessage();

}

}

});

home.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

bunVenit.hide();

informatii.hide();

stergeSedinta.hide();

materiiBox.hide();

tnume.hide();

send.hide();

materie.hide();

nume.hide();

dataText.hide();

date.hide();

month.hide();

year.hide();

titluCreeazaCerere.hide();

listaSed.hide();

titluDatePersonale.show();

numeHome.show();

passwordHome.show();

adresaHome.show();

dataNasteriiHome.show();

facultateHome.show();

nota.hide();

notaBox.hide();

materieRecenzie.hide();

materieBox.hide();

recenzie.hide();

creeazaRecenzie.hide();

trecenzie.hide();

trecenziileTale.hide();

homeNume.show();

homePassword.show();

homeDataNasterii.show();

homeAdresa.show();

homeFacultate.show();

modificaHome.show();

salveazaButtonHome.show();

locCreeazaCerere.hide();

oraCreeazaCerere.hide();

pretCreeazaCerere.hide();

oraBox.hide();

minuteBox.hide();

locBox.hide();

pretBox.hide();

trimiteRecenzie.hide();

listaRec.hide();

String username = new String();

String password = new String();

String line;

BufferedReader reader;

try {

reader = new BufferedReader(new FileReader("savedUsAndPass.txt"));

while ((line = reader.readLine()) != null) {

String fields[]=line.split(",");

username=fields[0];

password=fields[1];

}

}catch (IOException e1) {

e1.printStackTrace();

}

try {

reader = new BufferedReader(new FileReader("Users.txt"));

while ((line = reader.readLine()) != null) {

String[] fields2 = line.split(",");

if (fields2[0].equals(username) && fields2[2].equals(password)) {

homeNume.setText(username);

homePassword.setText(password);

homeAdresa.setText(fields2[1]);

homeDataNasterii.setText(fields2[3]);

homeFacultate.setText(fields2[6]);

}

}

}catch (IOException e1) {

e1.printStackTrace();

}

}

});

modificaHome.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

homeNume.setEnabled(true);

homePassword.setEnabled(true);

homeAdresa.setEnabled(true);

homeDataNasterii.setEnabled(true);

homeFacultate.setEnabled(true);

}

});

creeazaCerere.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

bunVenit.hide();

informatii.hide();

stergeSedinta.hide();

materiiBox.show();

tnume.show();

send.show();

materie.show();

nume.show();

dataText.show();

date.show();

month.show();

year.show();

titluCreeazaCerere.show();

listaSed.hide();

numeHome.hide();

titluDatePersonale.hide();

passwordHome.hide();

adresaHome.hide();

dataNasteriiHome.hide();

facultateHome.hide();

nota.hide();

notaBox.hide();

materieRecenzie.hide();

materieBox.hide();

recenzie.hide();

creeazaRecenzie.hide();

trecenzie.hide();

trecenziileTale.hide();

homeNume.hide();

homePassword.hide();

homeDataNasterii.hide();

homeAdresa.hide();

homeFacultate.hide();

modificaHome.hide();

salveazaButtonHome.hide();

locCreeazaCerere.show();

oraCreeazaCerere.show();

pretCreeazaCerere.show();

oraBox.show();

minuteBox.show();

locBox.show();

pretBox.show();

trimiteRecenzie.hide();

listaRec.hide();

String username = new String();

String password = new String();

String line;

BufferedReader reader;

try {

reader = new BufferedReader(new FileReader("savedUsAndPass.txt"));

while ((line = reader.readLine()) != null) {

String fields[]=line.split(",");

username=fields[0];

password=fields[1];

}

}catch (IOException e1) {

e1.printStackTrace();

}

tnume.setText(username);

tnume.setEnabled(false);

}

});

send.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String name;

String discipline;

String date1;

String time;

String place;

String price;

Boolean isAccepted=false;

name=tnume.getText();

discipline=(String)materiiBox.getSelectedItem();

date1= (String)date.getSelectedItem()+"/"+(String)month.getSelectedItem()+"/"+(String)year.getSelectedItem();

time=(String)oraBox.getSelectedItem()+":"+(String)minuteBox.getSelectedItem();

place=(String)locBox.getSelectedItem();

price=(String)pretBox.getSelectedItem();

Request req = new Request(name,discipline,date1,time,place,price,isAccepted);

RegisterRequests rg = new RegisterRequests();

try{

rg.registerRequest("Requests.txt",req);

JOptionPane.showMessageDialog(null, "Your request has been sent succesfully");

tnume.setText("");

}catch(IOException ee){

JOptionPane.showMessageDialog(null, ee);

}

}

});

sedinte.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

bunVenit.hide();

informatii.hide();

stergeSedinta.show();

materiiBox.hide();

tnume.hide();

send.hide();

materie.hide();

nume.hide();

dataText.hide();

date.hide();

month.hide();

year.hide();

titluCreeazaCerere.hide();

numeHome.hide();

listaSed.show();

passwordHome.hide();

adresaHome.hide();

dataNasteriiHome.hide();

facultateHome.hide();

creeazaRecenzie.hide();

nota.hide();

notaBox.hide();

materieRecenzie.hide();

materieBox.hide();

recenzie.hide();

trecenzie.hide();

trecenziileTale.hide();

homeNume.hide();

homePassword.hide();

homeDataNasterii.hide();

homeAdresa.hide();

homeFacultate.hide();

modificaHome.hide();

salveazaButtonHome.hide();

locCreeazaCerere.hide();

oraCreeazaCerere.hide();

pretCreeazaCerere.hide();

oraBox.hide();

minuteBox.hide();

locBox.hide();

pretBox.hide();

trimiteRecenzie.hide();

listaRec.hide();

int i=0;

try{

BufferedReader reader = new BufferedReader(new FileReader("Sessions.txt"));

String line;

while ((line = reader.readLine()) != null) {

listaSedinte[i]=line;

i++;

}

}catch(IOException exc){

exc.getMessage();

}

}

});

creeazaRecenzii.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

bunVenit.hide();

informatii.hide();

stergeSedinta.hide();

materiiBox.hide();

tnume.hide();

send.hide();

materie.hide();

nume.hide();

dataText.hide();

date.hide();

month.hide();

year.hide();

titluCreeazaCerere.hide();

numeHome.hide();

listaSed.hide();

passwordHome.hide();

adresaHome.hide();

dataNasteriiHome.hide();

facultateHome.hide();

creeazaRecenzie.show();

nota.show();

notaBox.show();

materieRecenzie.show();

materieBox.show();

recenzie.show();

trecenzie.show();

trecenziileTale.hide();

homeNume.hide();

homePassword.hide();

homeDataNasterii.hide();

homeAdresa.hide();

homeFacultate.hide();

modificaHome.hide();

salveazaButtonHome.hide();

locCreeazaCerere.hide();

oraCreeazaCerere.hide();

pretCreeazaCerere.hide();

oraBox.hide();

minuteBox.hide();

locBox.hide();

pretBox.hide();

trimiteRecenzie.show();

listaRec.hide();

}

});

recenzii.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

bunVenit.hide();

informatii.hide();

stergeSedinta.hide();

materiiBox.hide();

tnume.hide();

send.hide();

materie.hide();

nume.hide();

dataText.hide();

date.hide();

month.hide();

year.hide();

titluCreeazaCerere.hide();

numeHome.hide();

listaSed.hide();

passwordHome.hide();

adresaHome.hide();

dataNasteriiHome.hide();

facultateHome.hide();

creeazaRecenzie.hide();

nota.hide();

notaBox.hide();

materieRecenzie.hide();

materieBox.hide();

recenzie.hide();

trecenzie.hide();

trecenziileTale.show();

homeNume.hide();

homePassword.hide();

homeDataNasterii.hide();

homeAdresa.hide();

homeFacultate.hide();

modificaHome.hide();

salveazaButtonHome.hide();

locCreeazaCerere.hide();

oraCreeazaCerere.hide();

pretCreeazaCerere.hide();

oraBox.hide();

minuteBox.hide();

locBox.hide();

pretBox.hide();

trimiteRecenzie.hide();

listaRec.show();

}

});

stergeSedinta.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

}

});

JLabel label = new JLabel();

label.setIcon(new ImageIcon("D:\\Ovidius\\Java Proiecte\\EMeditatii\\a.png"));

Dimension size1 = label.getPreferredSize();

label.setBounds(-800, 0, 1800, 800);

panel.add(label);

stergeSedinta.hide();

materiiBox.hide();

tnume.hide();

send.hide();

materie.hide();

nume.hide();

dataText.hide();

date.hide();

month.hide();

year.hide();

titluCreeazaCerere.hide();

listaSed.hide();

titluDatePersonale.hide();

numeHome.hide();

passwordHome.hide();

adresaHome.hide();

dataNasteriiHome.hide();

facultateHome.hide();

creeazaRecenzie.hide();

nota.hide();

notaBox.hide();

materieRecenzie.hide();

materieBox.hide();

recenzie.hide();

trecenzie.hide();

trecenziileTale.hide();

homeNume.hide();

homePassword.hide();

homeDataNasterii.hide();

homeAdresa.hide();

homeFacultate.hide();

modificaHome.hide();

salveazaButtonHome.hide();

locCreeazaCerere.hide();

oraCreeazaCerere.hide();

pretCreeazaCerere.hide();

oraBox.hide();

minuteBox.hide();

locBox.hide();

pretBox.hide();

trimiteRecenzie.hide();

listaRec.hide();

}

}

## 4.TeacherFrame.java

package emeditatii;

import java.awt.\*;

import java.awt.event.\*;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.ArrayList;

import javax.swing.\*;

public class TeacherFrame {

private final JLabel imageLabel;

private final JFrame frame;

private final JPanel panel;

private final JLabel nume;

private final JLabel materie;

private final JTextField tnume;

private final JLabel titluCreeazaCerere;

private final JLabel bunVenit;

private final String materii[] =

{

};

private final String ora[] =

{

"14","15","16","17","18"

};

private final String minute[] =

{

"00","30"

};

private final String loc[] =

{

"Sala 15","Sala 20"

};

private final String pret[] =

{

"100","120"

};

private final JComboBox locBox;

private final JComboBox pretBox;

private final JComboBox oraBox;

private final JComboBox minuteBox;

private final JComboBox materiiBox;

private final String dates[]

= { "1", "2", "3", "4", "5",

"6", "7", "8", "9", "10",

"11", "12", "13", "14", "15",

"16", "17", "18", "19", "20",

"21", "22", "23", "24", "25",

"26", "27", "28", "29", "30",

"31" };

private final String months[]

= { "Jan", "Feb", "Mar", "Apr",

"May", "Jun", "July", "Aug",

"Sup", "Oct", "Nov", "Dec" };

private final String years[]

= { "2023" , "2024" , "2025" };

private final String cereri[] ={" "," "," "};

private final JList<String> cereriList;

private final JComboBox date;

private final JComboBox month;

private final JComboBox year;

private final JLabel dataText;

private final JLabel informatii;

private final String listaSedinte[] = { " "," "," " };

private final JList<String> listaSed;

private final JList listaRec;

private final JLabel titluDatePersonale;

private final JLabel numeHome;

private final JLabel prenumeHome;

private final JLabel dataNasteriiHome;

private final JLabel adresaHome;

private final JLabel facultateHome;

private final JLabel creeazaRecenzie;

private final JLabel nota;

private final JComboBox notaBox;

private final String note[]

= { "1" , "2" , "3", "4" , "5", "6", "7", "8", "9", "10" };

private final JLabel materieRecenzie;

private final JComboBox materieBox;

private final JLabel recenzie;

private final JTextArea trecenzie;

private final JLabel trecenziileTale;

private final JTextField homeNume;

private final JTextField homePrenume;

private final JTextField homeDataNasterii;

private final JTextField homeAdresa;

private final JTextField homeFacultate;

private final JLabel oraCreeazaCerere;

private final JLabel locCreeazaCerere;

private final JLabel pretCreeazaCerere;

private final JButton trimiteRecenzie;

private final String listaRecenzii[] = { "Recenzie 1" };

private final JButton acceptaCereri;

private final JButton nuAcceptaCereri;

private final JButton modificaCereri;

TeacherFrame()

{

frame = new JFrame("Teacher Frame");

panel = new JPanel();

frame.getContentPane();

String defaultpath;

imageLabel = new JLabel();

ImageIcon image = new ImageIcon("defaultImage.png");

imageLabel.setIcon(image);

imageLabel.setSize(100, 100);

imageLabel.setLocation(600,200);

imageLabel.setBackground(Color.red);

imageLabel.setCursor(Cursor.getPredefinedCursor(Cursor.HAND\_CURSOR));

imageLabel.setVisible(true);

imageLabel.addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent e) {

JFileChooser fileChooser = new JFileChooser();

fileChooser.setCurrentDirectory(new File(System.getProperty("user.home")));

int result = fileChooser.showOpenDialog(frame);

if (result == JFileChooser.APPROVE\_OPTION) {

File selectedFile = fileChooser.getSelectedFile();

imageLabel.setIcon(new ImageIcon(selectedFile.getAbsolutePath()));

}

}

});

bunVenit = new JLabel("Welcome, Teacher! ");

bunVenit.setFont(new Font("Serif", Font.BOLD, 30));

bunVenit.setSize(300, 40);

bunVenit.setLocation(500, 200);

panel.add(bunVenit);

informatii = new JLabel("You are on the Teachers page!");

informatii.setFont(new Font("Serif", Font.PLAIN, 30));

informatii.setSize(500, 40);

informatii.setLocation(380, 230);

panel.add(informatii);

JButton logOut = new JButton("Log Out");

Dimension size = logOut.getPreferredSize();

logOut.setBounds(1, 460, 150, 200);

panel.add(logOut);

JButton recenzii = new JButton("Reviews");

recenzii.setBounds(1, 360, 150, 100);

panel.add(recenzii);

recenzii.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String savedDiscipline= new String();

String username = new String();

String line;

BufferedReader reader;

try {

reader = new BufferedReader(new FileReader("savedUsAndPass.txt"));

while ((line = reader.readLine()) != null) {

String fields[]=line.split(",");

username=fields[0];

}

}catch (IOException e1) {

e1.printStackTrace();

}

try{

reader = new BufferedReader(new FileReader("Users.txt"));

String[] split;

while ((line = reader.readLine()) != null) {

split = line.split(",");

if(split[0].equals(username)){

savedDiscipline=split[6];

}

}

}catch(IOException exc){

exc.getMessage();

}

int i=0;

try{

reader = new BufferedReader(new FileReader("Reviews.txt"));

String[] split;

while ((line = reader.readLine()) != null) {

split = line.split(",");

if(split[3].equals(savedDiscipline)){

listaRecenzii[i]=line;

i++;

}

}

}catch(IOException exc){

exc.getMessage();

}

}

});

JButton sedinte = new JButton("Manage Sessions");

sedinte.setBounds(1, 260, 150, 100);

panel.add(sedinte);

JButton creeazaCerere = new JButton("Manage Requests");

creeazaCerere.setBounds(1, 160, 150, 100);

panel.add(creeazaCerere);

JButton home = new JButton("Home");

home.setBounds(1, 0, 150, 160);

panel.add(home);

Dimension dimension = Toolkit.getDefaultToolkit().getScreenSize();

int x = (int) ((dimension.getWidth() - frame.getWidth()) / 5);

int y = (int) ((dimension.getHeight() - frame.getHeight()) / 6);

frame.setLocation(x, y);

panel.setLayout(null);

panel.setBorder(BorderFactory.createEmptyBorder(100, 10, 10, 10));

frame.setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

frame.add(panel);

frame.setSize(1000, 700);

frame.setVisible(true);

ImageIcon img = new ImageIcon("E:\\education.png");

frame.setIconImage(img.getImage());

frame.setTitle("Teacher Page");

JButton stergeSedinta = new JButton("Delete");

stergeSedinta.setBounds(810, 590, 100, 40);

stergeSedinta.setFont(new Font("Sans", Font.BOLD, 20));

panel.add(stergeSedinta);

stergeSedinta.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String selectedRequest = listaSed.getSelectedValue();

ArrayList<String> requests = new ArrayList<>();

try (BufferedReader br = new BufferedReader(new FileReader("Sessions.txt"))) {

String line;

while ((line = br.readLine()) != null) {

requests.add(line);

}

} catch (IOException exc) {

exc.printStackTrace();

}

int indexToRemove = -1;

for (int i = 0; i < requests.size(); i++) {

if (requests.get(i).equals(selectedRequest)) {

indexToRemove = i;

break;

}

}

if (indexToRemove != -1) {

requests.remove(indexToRemove);

}

try (PrintWriter writer = new PrintWriter(new FileWriter("Sessions.txt"))) {

for (String request : requests) {

writer.println(request);

}

} catch (IOException exc) {

exc.printStackTrace();

}

JOptionPane.showMessageDialog(null,"The Session has been deleted");

DefaultListModel model = (DefaultListModel) cereriList.getModel();

model.removeAllElements();

for (String request : requests) {

model.addElement(request);

}

}

});

titluCreeazaCerere = new JLabel("Creeaza Cerere");

titluCreeazaCerere.setFont(new Font("Serif", Font.BOLD, 30));

titluCreeazaCerere.setSize(500, 40);

titluCreeazaCerere.setLocation(470, 100);

panel.add(titluCreeazaCerere);

nume = new JLabel("Name:");

nume.setFont(new Font("Serif", Font.BOLD, 30));

nume.setSize(300, 40);

nume.setLocation(300, 190);

panel.add(nume);

materie = new JLabel("Materie:");

materie.setFont(new Font("Serif", Font.BOLD, 30));

materie.setSize(300, 40);

materie.setLocation(300, 250);

panel.add(materie);

JButton send = new JButton("Send");

send.setBounds(610, 540, 100, 40);

send.setFont(new Font("Sans",Font.BOLD,20));

panel.add(send);

tnume = new JTextField();

tnume.setFont(new Font("Serif", Font.BOLD, 20));

tnume.setSize(300, 40);

tnume.setLocation(405, 195);

panel.add(tnume);

materiiBox = new JComboBox(materii);

materiiBox.setFont(new Font("Arial", Font.BOLD, 20));

materiiBox.setSize(275, 30);

materiiBox.setLocation(430, 260);

panel.add(materiiBox);

dataText = new JLabel("Data:");

dataText.setFont(new Font("Serif", Font.BOLD, 30));

dataText.setSize(300, 40);

dataText.setLocation(300, 310);

panel.add(dataText);

date = new JComboBox(dates);

date.setFont(new Font("Serif", Font.PLAIN, 15));

date.setSize(100, 30);

date.setLocation(400, 318);

panel.add(date);

month = new JComboBox(months);

month.setFont(new Font("Serif", Font.PLAIN, 15));

month.setSize(100, 30);

month.setLocation(505, 318);

panel.add(month);

year = new JComboBox(years);

year.setFont(new Font("Serif", Font.PLAIN, 15));

year.setSize(100, 30);

year.setLocation(610, 318);

panel.add(year);

listaSed = new JList<String>(listaSedinte);

listaSed.setFont(new Font("Serif", Font.PLAIN, 20));

listaSed.setSize(680, 500);

listaSed.setLocation(230, 50);

panel.add(listaSed);

titluDatePersonale = new JLabel("Personal data");

titluDatePersonale.setFont(new Font("Serif", Font.BOLD, 30));

titluDatePersonale.setSize(500, 40);

titluDatePersonale.setLocation(470, 100);

panel.add(titluDatePersonale);

numeHome = new JLabel("Username:");

numeHome.setFont(new Font("Serif", Font.BOLD, 30));

numeHome.setSize(300, 40);

numeHome.setLocation(300, 200);

panel.add(numeHome);

prenumeHome = new JLabel("Password:");

prenumeHome.setFont(new Font("Serif", Font.BOLD, 30));

prenumeHome.setSize(300, 40);

prenumeHome.setLocation(300, 250);

panel.add(prenumeHome);

adresaHome = new JLabel("Mail Adress:");

adresaHome.setFont(new Font("Serif", Font.BOLD, 30));

adresaHome.setSize(300, 40);

adresaHome.setLocation(300, 350);

panel.add(adresaHome);

dataNasteriiHome = new JLabel("Gender:");

dataNasteriiHome.setFont(new Font("Serif", Font.BOLD, 30));

dataNasteriiHome.setSize(300, 40);

dataNasteriiHome.setLocation(300, 300);

panel.add(dataNasteriiHome);

facultateHome = new JLabel("Discipline:");

facultateHome.setFont(new Font("Serif", Font.BOLD, 30));

facultateHome.setSize(300, 40);

facultateHome.setLocation(300, 400);

panel.add(facultateHome);

creeazaRecenzie = new JLabel("Creeaza recenzie");

creeazaRecenzie.setFont(new Font("Serif", Font.BOLD, 30));

creeazaRecenzie.setSize(500, 40);

creeazaRecenzie.setLocation(470, 100);

panel.add(creeazaRecenzie);

nota = new JLabel("Nota");

nota.setFont(new Font("Serif", Font.BOLD, 30));

nota.setSize(500, 40);

nota.setLocation(300, 200);

panel.add(nota);

notaBox = new JComboBox(note);

notaBox.setFont(new Font("Arial", Font.BOLD, 20));

notaBox.setSize(275, 30);

notaBox.setLocation(390, 208);

panel.add(notaBox);

materieRecenzie = new JLabel("Materie");

materieRecenzie.setFont(new Font("Serif", Font.BOLD, 30));

materieRecenzie.setSize(500, 40);

materieRecenzie.setLocation(300, 250);

panel.add(materieRecenzie);

materieBox = new JComboBox(materii);

materieBox.setFont(new Font("Arial", Font.BOLD, 20));

materieBox.setSize(275, 30);

materieBox.setLocation(425, 258);

panel.add(materieBox);

recenzie = new JLabel("Recenzia dumneavostra");

recenzie.setFont(new Font("Serif", Font.BOLD, 30));

recenzie.setSize(500, 40);

recenzie.setLocation(300, 300);

panel.add(recenzie);

trecenzie = new JTextArea();

trecenzie.setFont(new Font("Serif", Font.BOLD, 20));

trecenzie.setSize(550, 220);

trecenzie.setLocation(300, 350);

panel.add(trecenzie);

trecenziileTale = new JLabel("Your Reviews");

trecenziileTale.setFont(new Font("Serif", Font.BOLD, 30));

trecenziileTale.setSize(500, 40);

trecenziileTale.setLocation(470, 100);

panel.add(trecenziileTale);

homeNume = new JTextField();

homeNume.setFont(new Font("Serif", Font.BOLD, 15));

homeNume.setSize(250, 40);

homeNume.setLocation(490, 205);

panel.add(homeNume);

homeNume.setEnabled(false);

homePrenume = new JTextField();

homePrenume.setFont(new Font("Serif", Font.BOLD, 15));

homePrenume.setSize(250, 40);

homePrenume.setLocation(490, 255);

panel.add(homePrenume);

homePrenume.setEnabled(false);

homeDataNasterii = new JTextField();

homeDataNasterii.setFont(new Font("Serif", Font.BOLD, 15));

homeDataNasterii.setSize(250, 40);

homeDataNasterii.setLocation(490, 305);

panel.add(homeDataNasterii);

homeDataNasterii.setEnabled(false);

homeAdresa = new JTextField();

homeAdresa.setFont(new Font("Serif", Font.BOLD, 15));

homeAdresa.setSize(250, 40);

homeAdresa.setLocation(490, 355);

panel.add(homeAdresa);

homeAdresa.setEnabled(false);

homeFacultate = new JTextField();

homeFacultate.setFont(new Font("Serif", Font.BOLD, 15));

homeFacultate.setSize(250, 40);

homeFacultate.setLocation(490, 405);

panel.add(homeFacultate);

homeFacultate.setEnabled(false);

JButton modificaHome = new JButton("Modify Data");

modificaHome.setBounds(620, 470, 120, 30);

panel.add(modificaHome);

JButton salveazaButtonHome = new JButton("Save Data");

salveazaButtonHome.setBounds(500, 470, 120, 30);

panel.add(salveazaButtonHome);

oraCreeazaCerere = new JLabel("Time:");

oraCreeazaCerere.setFont(new Font("Serif", Font.BOLD, 30));

oraCreeazaCerere.setSize(300, 40);

oraCreeazaCerere.setLocation(300, 360);

panel.add(oraCreeazaCerere);

locCreeazaCerere = new JLabel("Place:");

locCreeazaCerere.setFont(new Font("Serif", Font.BOLD, 30));

locCreeazaCerere.setSize(300, 40);

locCreeazaCerere.setLocation(300, 410);

panel.add(locCreeazaCerere);

pretCreeazaCerere = new JLabel("Price:");

pretCreeazaCerere.setFont(new Font("Serif", Font.BOLD, 30));

pretCreeazaCerere.setSize(300, 40);

pretCreeazaCerere.setLocation(300, 460);

panel.add(pretCreeazaCerere);

oraBox = new JComboBox(ora);

oraBox.setFont(new Font("Arial", Font.BOLD, 20));

oraBox.setSize(70, 30);

oraBox.setLocation(430, 370);

panel.add(oraBox);

minuteBox = new JComboBox(minute);

minuteBox.setFont(new Font("Arial", Font.BOLD, 20));

minuteBox.setSize(70, 30);

minuteBox.setLocation(510, 370);

panel.add(minuteBox);

locBox = new JComboBox(loc);

locBox.setFont(new Font("Arial", Font.BOLD, 20));

locBox.setSize(180, 30);

locBox.setLocation(400, 420);

panel.add(locBox);

pretBox = new JComboBox(pret);

pretBox.setFont(new Font("Arial", Font.BOLD, 20));

pretBox.setSize(180, 30);

pretBox.setLocation(400, 470);

panel.add(pretBox);

trimiteRecenzie = new JButton("Send");

trimiteRecenzie.setBounds(730, 585, 120, 30);

panel.add(trimiteRecenzie);

listaRec = new JList<String>(listaRecenzii);

listaRec.setFont(new Font("Serif", Font.PLAIN, 15));

listaRec.setSize(680, 350);

listaRec.setLocation(230, 140);

panel.add(listaRec);

cereriList = new JList<String>(cereri);

cereriList.setFont(new Font("Serif", Font.PLAIN, 20));

cereriList.setSize(680, 350);

cereriList.setLocation(230, 140);

panel.add(cereriList);

acceptaCereri = new JButton("Accept");

acceptaCereri.setBounds(450, 500, 120, 30);

panel.add(acceptaCereri);

nuAcceptaCereri = new JButton("Delete");

nuAcceptaCereri.setBounds(600, 500, 120, 30);

panel.add(nuAcceptaCereri);

modificaCereri = new JButton("Modify");

modificaCereri.setBounds(750, 500, 120, 30);

panel.add(modificaCereri);

home.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

acceptaCereri.hide();

nuAcceptaCereri.hide();

modificaCereri.hide();

bunVenit.hide();

informatii.hide();

stergeSedinta.hide();

materiiBox.hide();

tnume.hide();

send.hide();

materie.hide();

nume.hide();

dataText.hide();

date.hide();

month.hide();

year.hide();

titluCreeazaCerere.hide();

listaSed.hide();

titluDatePersonale.show();

numeHome.show();

prenumeHome.show();

adresaHome.show();

dataNasteriiHome.show();

facultateHome.show();

nota.hide();

notaBox.hide();

materieRecenzie.hide();

materieBox.hide();

recenzie.hide();

creeazaRecenzie.hide();

trecenzie.hide();

trecenziileTale.hide();

homeNume.show();

homePrenume.show();

homeDataNasterii.show();

homeAdresa.show();

homeFacultate.show();

modificaHome.show();

salveazaButtonHome.show();

locCreeazaCerere.hide();

oraCreeazaCerere.hide();

pretCreeazaCerere.hide();

oraBox.hide();

minuteBox.hide();

locBox.hide();

pretBox.hide();

trimiteRecenzie.hide();

listaRec.hide();

cereriList.hide();

imageLabel.show();

String username = new String();

String password = new String();

String line;

BufferedReader reader;

try {

reader = new BufferedReader(new FileReader("savedUsAndPass.txt"));

while ((line = reader.readLine()) != null) {

String fields[]=line.split(",");

username=fields[0];

password=fields[1];

}

}catch (IOException e1) {

e1.printStackTrace();

}

try {

reader = new BufferedReader(new FileReader("Users.txt"));

while ((line = reader.readLine()) != null) {

String[] fields2 = line.split(",");

if (fields2[0].equals(username) && fields2[2].equals(password)) {

homeNume.setText(username);

homePrenume.setText(password);

homeAdresa.setText(fields2[1]);

homeDataNasterii.setText(fields2[3]);

homeFacultate.setText(fields2[6]);

}

}

}catch (IOException e1) {

e1.printStackTrace();

}

}

});

modificaHome.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

homeNume.setEnabled(true);

homePrenume.setEnabled(true);

homeAdresa.setEnabled(true);

homeDataNasterii.setEnabled(true);

homeFacultate.setEnabled(true);

}

});

creeazaCerere.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

bunVenit.hide();

informatii.hide();

stergeSedinta.hide();

materiiBox.hide();

tnume.hide();

send.hide();

materie.hide();

nume.hide();

dataText.hide();

date.hide();

month.hide();

year.hide();

titluCreeazaCerere.hide();

listaSed.hide();

numeHome.hide();

titluDatePersonale.hide();

prenumeHome.hide();

adresaHome.hide();

dataNasteriiHome.hide();

facultateHome.hide();

nota.hide();

notaBox.hide();

materieRecenzie.hide();

materieBox.hide();

recenzie.hide();

creeazaRecenzie.hide();

trecenzie.hide();

trecenziileTale.hide();

homeNume.hide();

homePrenume.hide();

homeDataNasterii.hide();

homeAdresa.hide();

homeFacultate.hide();

modificaHome.hide();

salveazaButtonHome.hide();

locCreeazaCerere.hide();

oraCreeazaCerere.hide();

pretCreeazaCerere.hide();

oraBox.hide();

minuteBox.hide();

locBox.hide();

pretBox.hide();

trimiteRecenzie.hide();

listaRec.hide();

cereriList.show();

acceptaCereri.show();

nuAcceptaCereri.show();

modificaCereri.show();

imageLabel.hide();

String username = new String();

String savedDiscipline = new String();

String line;

BufferedReader reader;

try {

reader = new BufferedReader(new FileReader("savedUsAndPass.txt"));

while ((line = reader.readLine()) != null) {

String fields[]=line.split(",");

username=fields[0];

}

}catch (IOException e1) {

e1.printStackTrace();

}

try {

reader = new BufferedReader(new FileReader("Users.txt"));

while ((line = reader.readLine()) != null) {

String[] fields2 = line.split(",");

if (fields2[0].equals(username)) {

savedDiscipline=fields2[6];

}

}

}catch (IOException e1) {

e1.printStackTrace();

}

int i=0;

try{

reader = new BufferedReader(new FileReader("Requests.txt"));

String[] split;

while ((line = reader.readLine()) != null) {

split = line.split(",");

if(split[1].equals(savedDiscipline)){

cereri[i]=line;

i++;

}

}

}catch(IOException exc){

exc.getMessage();

}

}

});

modificaCereri.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int selectedIndex = cereriList.getSelectedIndex();

String selectedString = cereriList.getSelectedValue();

if (selectedIndex != -1) {

String Split[]= selectedString.split(",");

Request selectedRequest = new Request(Split[0],Split[1],Split[2],Split[3],Split[4],Split[5],Boolean.parseBoolean(Split[6]));

String newPrice = JOptionPane.showInputDialog("Enter new price: ");

selectedRequest.setPrice(newPrice);

String newPlace = JOptionPane.showInputDialog("Enter new place: ");

selectedRequest.setPlace(newPlace);

String newDate = JOptionPane.showInputDialog("Enter new date: ");

selectedRequest.setRequestDate(newDate);

String newHour = JOptionPane.showInputDialog("Enter new hour: ");

selectedRequest.setRequestHour(newHour);

} else {

JOptionPane.showMessageDialog(null, "No request selected!");

}

}

});

acceptaCereri.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e){

int selectedIndex = cereriList.getSelectedIndex();

String selectedString = cereriList.getSelectedValue();

if (selectedIndex == -1) {

JOptionPane.showMessageDialog(null, "Please select a request.");

return;

}

String Split[]= selectedString.split(",");

Request selectedRequest = new Request(Split[0],Split[1],Split[2],Split[3],Split[4],Split[5],Boolean.parseBoolean(Split[6]));

selectedRequest.setIsAccepted(true);

JOptionPane.showMessageDialog(null, "The Request has been added to the Sessions");

try {

RegisterRequests register = new RegisterRequests();

register.registerRequest("Sessions.txt", selectedRequest);

cereriList.remove(selectedIndex);

} catch (IOException exc) {

exc.printStackTrace();

}

ArrayList<String> requests = new ArrayList<>();

try (BufferedReader br = new BufferedReader(new FileReader("Requests.txt"))) {

String line;

while ((line = br.readLine()) != null) {

requests.add(line);

}

} catch (IOException exc) {

exc.printStackTrace();

}

int indexToRemove = -1;

for (int i = 0; i < requests.size(); i++) {

if (requests.get(i).equals(selectedString)) {

indexToRemove = i;

break;

}

}

if (indexToRemove != -1) {

requests.remove(indexToRemove);

}

try (PrintWriter writer = new PrintWriter(new FileWriter("Requests.txt"))) {

for (String request : requests) {

writer.println(request);

}

} catch (IOException exc) {

exc.printStackTrace();

}

DefaultListModel model = (DefaultListModel) cereriList.getModel();

model.removeAllElements();

for (String request : requests) {

model.addElement(request);

}

}

});

nuAcceptaCereri.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String selectedRequest = cereriList.getSelectedValue();

ArrayList<String> requests = new ArrayList<>();

try (BufferedReader br = new BufferedReader(new FileReader("Requests.txt"))) {

String line;

while ((line = br.readLine()) != null) {

requests.add(line);

}

} catch (IOException exc) {

exc.printStackTrace();

}

int indexToRemove = -1;

for (int i = 0; i < requests.size(); i++) {

if (requests.get(i).equals(selectedRequest)) {

indexToRemove = i;

break;

}

}

if (indexToRemove != -1) {

requests.remove(indexToRemove);

}

try (PrintWriter writer = new PrintWriter(new FileWriter("Requests.txt"))) {

for (String request : requests) {

writer.println(request);

JOptionPane.showMessageDialog(null,"The request has been deleted");

}

} catch (IOException exc) {

exc.printStackTrace();

}

DefaultListModel model = (DefaultListModel) cereriList.getModel();

model.removeAllElements();

for (String request : requests) {

model.addElement(request);

}

}

});

sedinte.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

bunVenit.hide();

informatii.hide();

stergeSedinta.show();

materiiBox.hide();

tnume.hide();

send.hide();

materie.hide();

nume.hide();

dataText.hide();

date.hide();

month.hide();

year.hide();

titluCreeazaCerere.hide();

numeHome.hide();

listaSed.show();

prenumeHome.hide();

adresaHome.hide();

dataNasteriiHome.hide();

facultateHome.hide();

creeazaRecenzie.hide();

nota.hide();

notaBox.hide();

materieRecenzie.hide();

materieBox.hide();

recenzie.hide();

trecenzie.hide();

trecenziileTale.hide();

homeNume.hide();

homePrenume.hide();

homeDataNasterii.hide();

homeAdresa.hide();

homeFacultate.hide();

modificaHome.hide();

salveazaButtonHome.hide();

locCreeazaCerere.hide();

oraCreeazaCerere.hide();

pretCreeazaCerere.hide();

oraBox.hide();

minuteBox.hide();

locBox.hide();

pretBox.hide();

trimiteRecenzie.hide();

listaRec.hide();

cereriList.hide();

acceptaCereri.hide();

nuAcceptaCereri.hide();

modificaCereri.hide();

imageLabel.hide();

int i=0;

try{

BufferedReader reader = new BufferedReader(new FileReader("Sessions.txt"));

String line;

while ((line = reader.readLine()) != null) {

listaSedinte[i]=line;

i++;

}

}catch(IOException exc){

exc.getMessage();

}

}

});

stergeSedinta.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String selectedRequest = listaSed.getSelectedValue();

ArrayList<String> requests = new ArrayList<>();

try (BufferedReader br = new BufferedReader(new FileReader("Sessions.txt"))) {

String line;

while ((line = br.readLine()) != null) {

requests.add(line);

}

} catch (IOException exc) {

exc.printStackTrace();

}

int indexToRemove = -1;

for (int i = 0; i < requests.size(); i++) {

if (requests.get(i).equals(selectedRequest)) {

indexToRemove = i;

break;

}

}

if (indexToRemove != -1) {

requests.remove(indexToRemove);

}

try (PrintWriter writer = new PrintWriter(new FileWriter("Sessions.txt"))) {

for (String request : requests) {

writer.println(request);

JOptionPane.showMessageDialog(null,"The Session has been deleted");

}

} catch (IOException exc) {

exc.printStackTrace();

}

DefaultListModel model = (DefaultListModel) cereriList.getModel();

model.removeAllElements();

for (String request : requests) {

model.addElement(request);

}

}

});

recenzii.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

bunVenit.hide();

informatii.hide();

stergeSedinta.hide();

materiiBox.hide();

tnume.hide();

send.hide();

materie.hide();

nume.hide();

dataText.hide();

date.hide();

month.hide();

year.hide();

titluCreeazaCerere.hide();

numeHome.hide();

listaSed.hide();

prenumeHome.hide();

adresaHome.hide();

dataNasteriiHome.hide();

facultateHome.hide();

creeazaRecenzie.hide();

nota.hide();

notaBox.hide();

materieRecenzie.hide();

materieBox.hide();

recenzie.hide();

trecenzie.hide();

trecenziileTale.show();

homeNume.hide();

homePrenume.hide();

homeDataNasterii.hide();

homeAdresa.hide();

homeFacultate.hide();

modificaHome.hide();

salveazaButtonHome.hide();

locCreeazaCerere.hide();

oraCreeazaCerere.hide();

pretCreeazaCerere.hide();

oraBox.hide();

minuteBox.hide();

locBox.hide();

pretBox.hide();

trimiteRecenzie.hide();

listaRec.show();

cereriList.hide();

acceptaCereri.hide();

nuAcceptaCereri.hide();

modificaCereri.hide();

imageLabel.hide();

}

});

frame.setVisible(true);

frame.setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE);

logOut.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent ae) {

frame.setVisible(false);

frame.dispose();

new Login();

}});

JLabel label = new JLabel();

label.setIcon(new ImageIcon("D:\\Ovidius\\Java Proiecte\\EMeditatii\\a.png"));

Dimension size1 = label.getPreferredSize();

label.setBounds(-800, 0, 1800, 800);

panel.add(label);

stergeSedinta.hide();

materiiBox.hide();

tnume.hide();

send.hide();

materie.hide();

nume.hide();

dataText.hide();

date.hide();

month.hide();

year.hide();

titluCreeazaCerere.hide();

listaSed.hide();

titluDatePersonale.hide();

numeHome.hide();

prenumeHome.hide();

adresaHome.hide();

dataNasteriiHome.hide();

facultateHome.hide();

creeazaRecenzie.hide();

nota.hide();

notaBox.hide();

materieRecenzie.hide();

materieBox.hide();

recenzie.hide();

trecenzie.hide();

trecenziileTale.hide();

homeNume.hide();

homePrenume.hide();

homeDataNasterii.hide();

homeAdresa.hide();

homeFacultate.hide();

modificaHome.hide();

salveazaButtonHome.hide();

locCreeazaCerere.hide();

oraCreeazaCerere.hide();

pretCreeazaCerere.hide();

oraBox.hide();

minuteBox.hide();

locBox.hide();

pretBox.hide();

trimiteRecenzie.hide();

listaRec.hide();

cereriList.hide();

acceptaCereri.hide();

nuAcceptaCereri.hide();

modificaCereri.hide();

}

public static void main(String[] args){

TeacherFrame a = new TeacherFrame();

}

}

## 5.Professor.java

package emeditatii;

public class Professor extends User {

private String discipline;

public Professor(String name, String email, String password, String gender, String mobileNumber, boolean isProfessor, String discipline) {

super(name, email, password, gender, mobileNumber, isProfessor);

this.discipline = discipline;

}

public String getDiscipline() {

return discipline;

}

public void setDiscipline(String discipline) {

this.discipline = discipline;

}

}

## 6.Student.java

package emeditatii;

public class Student extends User {

private String faculty;

public Student(String name, String email, String password, String gender, String mobileNumber, boolean isProfessor, String faculty) {

super(name, email, password, gender, mobileNumber, isProfessor);

this.faculty = faculty;

}

public String getFaculty() {

return faculty;

}

public void setFaculty(String faculty) {

this.faculty = faculty;

}

}

## 7.Review.java

package emeditatii;

public class Review {

private String reviewerName;

private String rating;

private String comment;

private String discipline;

public Review(String reviewerName,String rating, String comment, String discipline) {

this.reviewerName = reviewerName;

this.rating = rating;

this.comment = comment;

this.discipline = discipline;

}

public String getReviewerName() {

return reviewerName;

}

public void setReviewerName(String reviewerName) {

this.reviewerName = reviewerName;

}

public String getRating() {

return rating;

}

public void setRating(String rating) {

this.rating = rating;

}

public String getComment() {

return comment;

}

public void setComment(String comment) {

this.comment = comment;

}

public String getDiscipline() {

return discipline;

}

public void setDiscipline(String discipline) {

this.discipline = discipline;

}};

## 8.Request.java

package emeditatii;

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

public class RegisterRequests {

public void registerRequest(String fileName, Request request) throws IOException {

BufferedWriter writer = new BufferedWriter(new FileWriter(fileName, true));

writer.write(request.getRequesterName()+","+request.getDiscipline()+","+request.getRequestDate()+","+request.getRequestHour()+","+request.getPlace()+","+request.getPrice()+","+request.getAccepted());

writer.newLine();

writer.close();

}

}

## 9.User.java

package emeditatii;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

import java.text.ParseException;

public class User {

private String name;

private String email;

private String password;

private String gender;

private String mobileNumber;

private boolean isProfessor;

public User(String name, String email, String password, String gender, String mobileNumber, boolean isProfessor) {

this.name = name;

this.email = email;

this.password = password;

this.gender = gender;

this.mobileNumber = mobileNumber;

this.isProfessor = isProfessor;

}

public User(String fileName) throws IOException, ParseException {

BufferedReader reader = new BufferedReader(new FileReader(fileName));

this.name = reader.readLine();

this.email = reader.readLine();

this.password = reader.readLine();

this.gender = reader.readLine();

this.mobileNumber = reader.readLine();

this.isProfessor = Boolean.parseBoolean(reader.readLine());

reader.close();

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public String getMobileNumber() {

return mobileNumber;

}

public void setMobileNumber(String mobileNumber) {

this.mobileNumber = mobileNumber;

}

public boolean isProfessor() {

return isProfessor;

}

public void setProfessor(boolean professor) {

isProfessor = professor;

}

}

## 10.UserRegistration.java

package emeditatii;

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

public class UserRegistration {

public void registerProfessor(String fileName, Professor user) throws IOException {

BufferedWriter writer = new BufferedWriter(new FileWriter(fileName, true));

writer.write(user.getName()+","+user.getEmail()+","+user.getPassword()+","+user.getGender()+","+user.getMobileNumber()+","+String.valueOf(user.isProfessor())+","+user.getDiscipline());

writer.newLine();

writer.close();

}

public void registerStudent(String fileName, Student user) throws IOException {

BufferedWriter writer = new BufferedWriter(new FileWriter(fileName, true));

writer.write(user.getName()+","+user.getEmail()+","+user.getPassword()+","+user.getGender()+","+user.getMobileNumber()+","+String.valueOf(user.isProfessor())+","+user.getFaculty());

writer.newLine();

writer.close();

}

}

## 11.Validate.java

package emeditatii;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class Validate {

public void validateUsername(String username) throws Exception {

Pattern pattern = Pattern.compile("^[a-zA-Z]+\\.[a-zA-Z]+$");

Matcher matcher = pattern.matcher(username);

if (!matcher.matches()) {

throw new Exception("Username must be in the format firstname.lastname");

}

}

public void validateEmail(String email) throws Exception {

Pattern pattern = Pattern.compile("^[a-zA-Z0-9\_!#$%&'\*+/=?`{|}~^.-]+@(gmail\\.com|yahoo\\.com|365\\.univ-ovidius\\.ro)$");

Matcher matcher = pattern.matcher(email);

if (!matcher.matches()) {

throw new Exception("Invalid email format. Must be gmail.com, yahoo.com, or 365.univ-ovidius.ro");

}

}

public void validateMobileNumber(String mobileNumber) throws Exception {

Pattern pattern = Pattern.compile("^07\\d{8}$");

Matcher matcher = pattern.matcher(mobileNumber);

if (!matcher.matches()) {

throw new Exception("Invalid mobile number format. Must be 07xxxxxxxxx");

}

}

public void validatePassword(String password) throws Exception {

// check if the password has at least 8 characters

if (password.length() < 8) {

throw new Exception("Password must have at least 8 characters");

}

// check if the password has at least one capital letter

boolean hasCapitalLetter = false;

for (int i = 0; i < password.length(); i++) {

if (Character.isUpperCase(password.charAt(i))) {

hasCapitalLetter = true;

break;

}

}

if (!hasCapitalLetter) {

throw new Exception("Password must have at least one capital letter");

}

// check if the password has at least one digit

boolean hasDigit = false;

for (int i = 0; i < password.length(); i++) {

if (Character.isDigit(password.charAt(i))) {

hasDigit = true;

break;

}

}

if (!hasDigit) {

throw new Exception("Password must have at least one digit");

}

}

}

## 12.RegisterRequests.java

package emeditatii;

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

public class RegisterRequests {

public void registerRequest(String fileName, Request request) throws IOException {

BufferedWriter writer = new BufferedWriter(new FileWriter(fileName, true));

writer.write(request.getRequesterName()+","+request.getDiscipline()+","+request.getRequestDate()+","+request.getRequestHour()+","+request.getPlace()+","+request.getPrice()+","+request.getAccepted());

writer.newLine();

writer.close();

}

}

## 13. RegisterReviews.java

package emeditatii;

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

public class RegisterReviews {

public void RegisterRev(String fileName, Review review) throws IOException {

BufferedWriter writer = new BufferedWriter(new FileWriter(fileName, true));

writer.write(review.getReviewerName()+","+review.getRating()+","+review.getComment()+","+review.getDiscipline());

writer.newLine();

writer.close();

}

}

## 14.Users.txt

Ionut.Tufan,ionuttufan04@gmail.com,Paro1,Male,0723759798,false,Informatics

Alex.Tudor,alex.tudor11@gmail.com,dddaaawee,Male,07445422244,false

andrei.alex,andreialex@gmail.com,32312312,Male,3123412331,false

alex.alex,ionut.tufan@gmail.com,pasdadsa,Male,0723759798,false

ionel.alin,ionel.alin@gmail.com,asdadadas,Male,0799999999,false

ionel.alexandru,ionelionel.alex@gmail.com,sadsadas,Male,0733545955,false

ggg.tttt,dasas@yahoo.com,dasdasd,Male,0733244421,false

alexandru.bobe,alex.bobe@365.univ-ovidius.ro,parola,Male,0722333444,true,Geometry

Ionut.Ionel,adresa@gmail.com,[C@6f53e81e,Male,0744333222,true

username.name,fdsfsfsdf@gmail.com,Parolamea1,Female,0788666555,true

Ionut.tufan,tufan@gmail.com,Parola12!,Male,0799888777,true,Informatics and Mathematics

alexa.andreea,alexa@gmail.com,Pewqeqw312!,Female,0722333444,false,Informatics

andrei.popescu,andrei@gmail.com,Parolamea1,Male,0766555444,true,Physics

alex.tudor,alexandru@gmail.com,Alex2003,Male,0739970798,false,Informatics

## 15.Requests.txt

Tufan Ionut,Matematica,4/May/2023,16:30,Sala 20,120,false

Tufan,Fizica,2/Apr/2023,14:30,Sala 15,100,false

Ionut.Tufan,Geometry,1/Jan/2023,14:00,Sala 10,100,false

## 16.Reviews.txt

Ionut.Tufan,10,i liked it a lot,Geometry

## 17.Sessions.txt

Ionut.Tufan,Geometry,1/Jan/2023,14:00,Sala 10,100,true

## 18.savedUsAndPass

Ionut.Tufan,Paro1

# **IV Bibliografie**

BIBLIOGRAFIE - PROGRAMARE ORIENTATA SPRE OBIECTE

1. A. V. Aho, J. D. Ullman, Foundations of Computer Science. C Edition, Computer Science Press, 1995

2. Leen AmmeraalandKang Zhang, Computer Graphics for Java Programmers, John Wiley & Sons, 2007

3. Crenguţa M. Bogdan, L. D. Şerbănaţi, Dezvoltarea orientată spre obiecte a programelor in Java, volumul II, Editura Politehnica Press, ed. 2, 2011

4. H. M. Deitel, P. J. Deitel, Java: How to Program, 4/e, Prentice Hall, 2003.

5. B. Eckel, Thinking in Java, 3/e, Prentice Hall, 2002.

6. D. Flanagan, B. McLaughlin, Java 1.5 Tiger: A Developer's Notebook, O'Reilly, 2004

7. D. Geary, Graphic Java 2, v.1 - Mastering the JFC, 3/e, v.2 - Swing, Sun Microsystems Press, 1999-2000

8. J. Gosling, B. Joy, G. Steele, G. Bracha, The JavaTM Language Specification, 3/e, Addison-Wesley, 2005

9. Mark Grand, Patterns in Java, Volume 1: A Catalog of Reusable Design Patterns Illustrated with UML, Second Edition, John Wiley & Sons, 2002

10. C. S. Horstmann and G. Cornell, Core Java 2: Vol.1 - Fundamentals, 6/e, Prentice Hall, 2002

11. C. S. Horstmann, Computing concepts with Java 2 Essentials, 3/e, John Wiley, 2003

12. Jonathan Knudsen, Java 2D Graphics, O’Reilly, 2001

13. James Martin, James J. Odel, Object-Oriented Methods: a Foundation. UML Edition, Prentice Hall, 1998

14. Rumbaugh J., Jacobson I., Booch G., The Unified Modeling Language. Reference Manual, Addison-Wesley, 1999

15. Specificaţia XML: http://www.w3.org/TR/2006/REC-xml-20060816/

16. L. D. Şerbănaţi, Limbaje de programare şi compilatoare, Editura Academiei, 1987

17. Crenguţa M. Bogdan, L. D. Şerbănaţi, Dezvoltarea orientată spre obiecte a programelor in Java, volumul I, Editura Politehnica Press, ed. 2, volumele 1 si 2, 2011

18. John Zukowski, Java™ 6 Platform Revealed, Apress, 2006 19. http://java.sun.com/products/jfc/tsc/articles/persistence3/

Alte cărţi utile de Java

1. A. Athanasiu, B. Costinescu, O. A. Drăgoi, F. I. Popovici, Limbajul Java. O perspectivă pragmatică, Editura Agora, 1998.

2. S.Tanasa, C.Olaru, S.Andrei, Java de la 0 la expert, Editura Polirom, 2003.

3. K. Sierra, B. Bates, Sun Certified Programmer & Developer for Java 2 Study Guide (Exam 310-035 & 310-027), McGraw-Hill Osborne Media, 2002