

UNIVERSITATEA TEHNICA DIN CLUJ-NAPOCA

1. Introducere

1.1 Descrierea Generală a Proiectului

Aplicația este un sistem informatic destinat gestiunii unei platforme de studiu. Funcționalitățile pe care le oferă aplicația vizează operații ce țin de gestiunea studenților, profesorilor și administrarea operațiilor curente din cadrul unor programe de studiu.

1.2 Scopul și Obiectivele

Crearea unui cadru online prin care se pot organiza activități didactice și interacțiuni între studenți și profesori.

2. Tema Proiectului

2.1 Prezentarea Aplicației

Aplicația propusă reprezintă un sistem informatic destinat gestionării unei platforme de studiu, care facilitează managementul studenților, profesorilor și activităților academice din cadrul unui program de învățământ. Realizată pe baza unei baze de date MySQL și având o interfață grafică intuitivă, aplicația permite realizarea eficientă a operațiunilor specifice rolurilor definite.

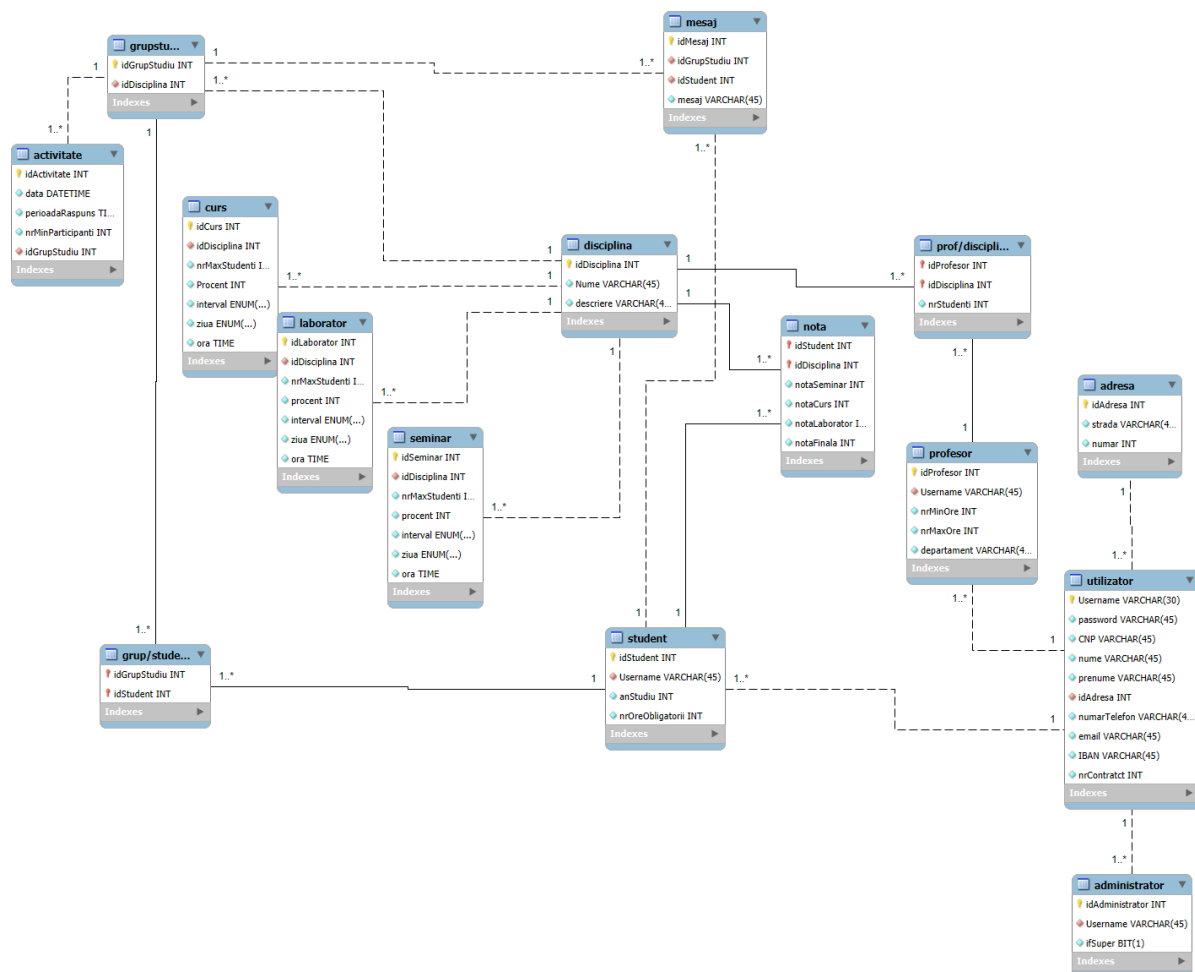
2.2 Funcționalități și Utilizatori

Aplicația oferă suport pentru trei tipuri principale de utilizatori: **student**, **profesor** și **administrator**. Printr-un sistem de autentificare și autorizare, fiecare utilizator își poate accesa datele personale precum CNP, nume, adresă, număr de telefon, email sau informații suplimentare precum cont IBAN sau contractul, fără a le putea modifica. Administratorii pot gestiona aceste date și pot atribui cursuri profesorilor și studenților, oferind vizibilitate asupra relațiilor și interacțiunilor din cadrul sistemului.

Pentru profesorii înregistrați, aplicația permite gestionarea cursurilor și activităților asociate (cursuri, seminare, laboratoare), inclusiv configurarea programării acestora, numărul maxim de studenți acceptați și tipurile de activități derulate. Studenții pot vizualiza și accesa catalogul, unde sunt evidențiate rezultatele lor și le pot descărca.

3. Descrierea Bazei de Date

3.1 Structura generală



Modelul bazei de date prezentat este unul relațional, fiind organizat în tabele interconectate care descriu clasele și relațiile din aplicație. Structura generală este bine definită, acoperind clase majore precum utilizatorii (studenți, profesori, administratori), cursurile, activitățile academice (seminare, laboratoare, cursuri) și informațiile administrative (note, mesaje, adrese). Relațiile dintre tabele sunt gestionate prin chei primare și chei străine pentru a menține structura datelor.

3.2 Detalii specifice

Utilizatori:

- Tabelul “utilizator” stochează informații personale (CNP, nume, adresă, email, IBAN) și identifică rolul fiecărei persoane.
- Profesorii și studenții sunt specificați în tabele separate (profesor, student) care conțin specificații extra, cum ar fi departamentul pentru profesori sau anul de studiu pentru studenți.

Cursuri și activități:

- “Curs”, “seminar” și “laborator” sunt tabele distincte care detaliază tipurile de activități academice. Fiecare tabel conține informații despre disciplina asociată, numărul maxim de studenți, zilele și orele în care se desfășoară.
- Tabelul “activitate” conține informațiile despre toate activitățile programate, incluzând data și perioada de răspuns.

Evaluare și note:

- Tabelul “nota” stochează rezultatele studenților pentru fiecare tip de activitate, inclusiv nota finală calculată pe baza ponderilor definite printr-o procedura MySQL.

Relații și comunicare:

- Relația dintre studenți și grupurile lor este gestionată în tabelul “grup/student”.
- Tabelul “mesaj” permite comunicarea între utilizatori, legată de cursuri sau alte activități.

3.3 Nivelul de normalizare

Modelul pare să fie cel puțin la **forma normală a treia (3NF)**:

- Fiecare atribut este atomic, ceea ce indică prima formă normală (1NF).
- Toate atributele depind complet de cheia primară din fiecare tabel, evitând dependențele parțiale (2NF).
- Nu există dependențe tranzitive între atributele non-cheie, ceea ce respectă cerințele 3NF.

Prin această normalizare, modelul minimizează redundanța datelor și asigură o integritate mai bună a bazei de date, făcând structura flexibilă și ușor de întreținut.

4. Elemente de programare a functionalitatilor

4.1 Proceduri și Triggere. Lista completa si exemple

Triggere:

În tabela “grupstudiu” avem trigger-ul deleteGrupStudiu:

```
create definer = lms@`%` trigger deleteGrupStudiu
  before delete
  on grupstudiu
  for each row
begin
  DELETE from mesajasteptare where mesajasteptare.idGrupStudiu=OLD.idGrupStudiu;
  DELETE from mesaj where mesaj.idGrupStudiu=OLD.idGrupStudiu;
end;
```

În tabela “mesaj” avem trigger-ul “deleteDataMesajAsteptare”:

```
create definer = lms@`%` trigger deleteDataMesajAsteptare
  before delete
  on mesaj
  for each row
begin
  DELETE from mesajasteptare where mesajasteptare.idGrupStudiu=OLD.idGrupStudiu;
end;
```

În tabela “Utilizator” avem “deleteDataNeacceptat”:

```
create definer = lms@`%` trigger deleteDataNeacceptat
  before delete
  on utilizator
  for each row
begin
  DELETE from profesor where profesor.Username=OLD.Username;
  DELETE from student where student.Username=OLD.Username;
  DELETE from administrator where administrator.Username=OLD.Username;
end;
```

Proceduri:

Procedura “inscriereStudent” care înscrie studentul la o anumita disciplina:

```
create
  definer = lms@%' procedure InscriereStudent(IN numeStudent varchar(45), IN disciplina varchar(45))
BEGIN
  DECLARE idD,idP,idS,idC INT;
  SELECT idStudent into idS from student where numeStudent=Username;
  SELECT idDisciplina into idD from disciplina where disciplina=Numa;
  SELECT curs.idCurs,idProfesor into idC,idP from 'curs' where curs.numarStudenti=(SELECT min(numarStudenti) from 'curs') and idDisciplina=idD;
  if(idD is not null and idP is not null and idS is not null) then
    INSERT INTO nota(idStudent, idDisciplina,idProfesor, notaSeminar, notaCurs, notaLaborator, notaFinala) VALUES
      ( idStudent idS, idDisciplina idD, idProfesor idP, notaSeminar null, notaCurs null, notaLaborator null, notaFinala null);
    UPDATE curs set numarStudenti=numarStudenti+1 WHERE idCurs=idC;
  else
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT ='Error';
  end if;
End;
```

Procedura de calculare a notei finale a fiecărui student pe baza notelor de la curs, seminar si laborator (aceasta mai e bazata si pe profesorul de care apartine studentul):

```
create
  definer = lms@%' procedure nota_finala(IN notaCurs float, IN notaLab float, IN notaSeminar float,
    IN idDisciplina float, IN idProfesor float, OUT notaFinala float)
BEGIN
  DECLARE procentCurs, procentLab, procentSeminar INT;
  SELECT curs.Procent into procentCurs from curs where curs.idProfesor=idProfesor AND curs.idDisciplina=idDisciplina;
  SELECT laborator.procent into procentLab from laborator where laborator.idProfesor=idProfesor AND laborator.idDisciplina=idDisciplina;
  SELECT seminar.procent into procentSeminar from seminar where seminar.idProfesor=idProfesor AND seminar.idDisciplina=idDisciplina;

  SET notaFinala = (notaCurs * procentCurs / 100) +
    (notaLab * procentLab / 100) +
    (notaSeminar * procentSeminar / 100);

end;
```

Procedura care selectează tipul de utilizator.

```
create
  definer = lms@%' procedure selectTypeOfUtilizator(IN username varchar(45), IN password varchar(45),
    OUT tip varchar(45))
BEGIN
  declare idA,idS,idP INT;
  set idA=0;
  set idS=0;
  set idP=0;

  SELECT administrator.idAdministrator into idA from administrator join utilizator 1.n<>1. using (Username) where username=utilizator.Username and password=utilizator.password;
  SELECT student.idStudent into idS from student join utilizator 1.n<>1. using (Username) where username=utilizator.Username and password=utilizator.password;
  SELECT profesor.idProfesor into idP from profesor join utilizator 1.n<>1. using (Username) where username=utilizator.Username and password=utilizator.password;
  if (idA!=0) then
    set tip='Administrator';
    SELECT * from utilizator join administrator 1.<>1.n. using (Username) where username=utilizator.Username and password=utilizator.password;
  else if (idS!=0) then
    set tip='Student';
    SELECT * from utilizator join student 1.<>1.n. using (Username) where username=utilizator.Username and password=utilizator.password;
  else if(idP!=0) then
    set tip='Profesor';
    SELECT * from utilizator join profesor 1.<>1.n. using (Username) where username=utilizator.Username and password=utilizator.password;
  else set tip='Invalid';
  end if;
end if;
end if;
end if;

end;
```

4.2 Alte exemple de Cod SQL

Codul SQL pentru selectarea datelor unui utilizator:

```
String url =
"jdbc:mysql://139.144.67.202:3306/lms?user=lms&password=WHlQjrrRDs5t";
Connection conn = DriverManager.getConnection(url);

PreparedStatement stmt = conn.prepareStatement(
    "SELECT u.*, s.anStudiu, p.departament, a.strada, a.numar FROM
    utilizator u LEFT JOIN student s ON u.Username = s.Username LEFT JOIN
    profesor p ON u.Username = p.Username LEFT JOIN adresa a ON u.idAdresa =
    a.idAdresa WHERE u.Username = ?");
stmt.setString(1, usernameText);
ResultSet rs = stmt.executeQuery();
```

Codul SQL pentru datele necesare din catalogul școlar:

```
// Interogarea pentru datele catalogului
String query = "SELECT disciplina.Nume AS Materie, student.Username AS NumeStudent, " +
    "nota.notaCurs, nota.notaSeminar, nota.notaLaborator, nota.notaFinala " +
    "FROM nota " +
    "JOIN student USING (idStudent) " +
    "JOIN disciplina USING (idDisciplina) " +
    "JOIN profesor USING (idProfesor) " +
    "WHERE profesor.Username = ?";
PreparedStatement stmt = conn.prepareStatement(query);
stmt.setString(1, profesor.getUsername());
```

Updateare de note:

```
// Actualizare note
String query = "UPDATE nota SET notaCurs = ?, notaSeminar = ?, notaLaborator = ?, notaFinala = ? " +
    "WHERE idStudent = ? AND idDisciplina = ? AND idProfesor = ?";
```

Codul SQL pentru selectarea grupurilor din care utilizatorul face parte:

```
String
url="jdbc:mysql://139.144.67.202:3306/lms?user=lms&password=WHlQjrrRDs5t";
Connection conn= DriverManager.getConnection(url);

PreparedStatement stmt=conn.prepareStatement("SELECT distinct
numeGrup,idGrupStudiu from grupstudiu join mesaj using (idGrupStudiu) where
mesaj.idStudent=(select idStudent from student where Username=?) and
mesaj='';");
stmt.setString(1,student.getUsername());
ResultSet rs=stmt.executeQuery();
```

Inserare de valori in tabele:

```
/// Inserarea in tabela curs
PreparedStatement insertCursStmt = conn.prepareStatement(
    sql: "INSERT INTO curs (idProfesor, idDisciplina, nrMaxStudenti, Procent, `interval`, ziua, ora, numarStudenti) " +
        "VALUES (?, ?, 0, 0, 'saptamanal', 'luni', '12:30:20', 0)";
//insertCursStmt.setInt(1,);
insertCursStmt.setInt( parameterIndex: 1, idProfesor);
insertCursStmt.setInt( parameterIndex: 2, idMaterie);
insertCursStmt.executeUpdate();

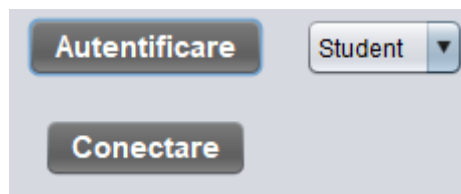
/// Inserarea si in tabela seminar
PreparedStatement insertSeminarStmt = conn.prepareStatement(
    sql: "INSERT INTO seminar (idProfesor, idDisciplina, nrMaxStudenti, procent, `interval`, ziua, ora) " +
        "VALUES (?, ?, 0, 0, 'saptamanal', 'luni', '12:30:20')";
insertSeminarStmt.setInt( parameterIndex: 1, idProfesor);
insertSeminarStmt.setInt( parameterIndex: 2, idMaterie);
insertSeminarStmt.executeUpdate();

/// Inserarea si in tabela seminar
PreparedStatement insertLabStmt = conn.prepareStatement(
    sql: "INSERT INTO laborator (idProfesor, idDisciplina, nrMaxStudenti, procent, `interval`, ziua, ora) " +
        "VALUES (?, ?, 0, 0, 'saptamanal', 'luni', '12:30:20')";
```

5. Interfața Grafică a Utilizatorului (GUI)

5.1 Descrierea functionalitatilor per tip de utilizator

Pagina Inițială:



Conține butonul pentru autentificare și butonul pentru conectare. Lângă butonul de logare se alege tipul de cont cu care dorești sa te autentifici.

Pagina de Autentificare:

Autentificare ca student:

CNP:

Nume:

Prenume:

Adresa:

Nr. Telefon:

Email:

IBAN:

Nr. Contract:

Username:

Password:

An Studiu:

Autentificare ca profesor:

CNP:

Nume:

Prenume:

Adresa:

Nr. Telefon:

Email:

IBAN:

Nr. Contract:

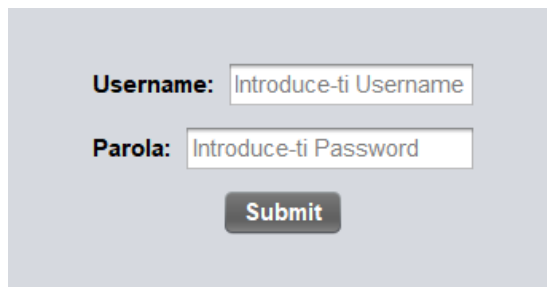
Username:

Password:

Departament:

Câmpurile trebuie completate cu date valide specifice pentru fiecare tip de utilizator.

Pagina de conectare:

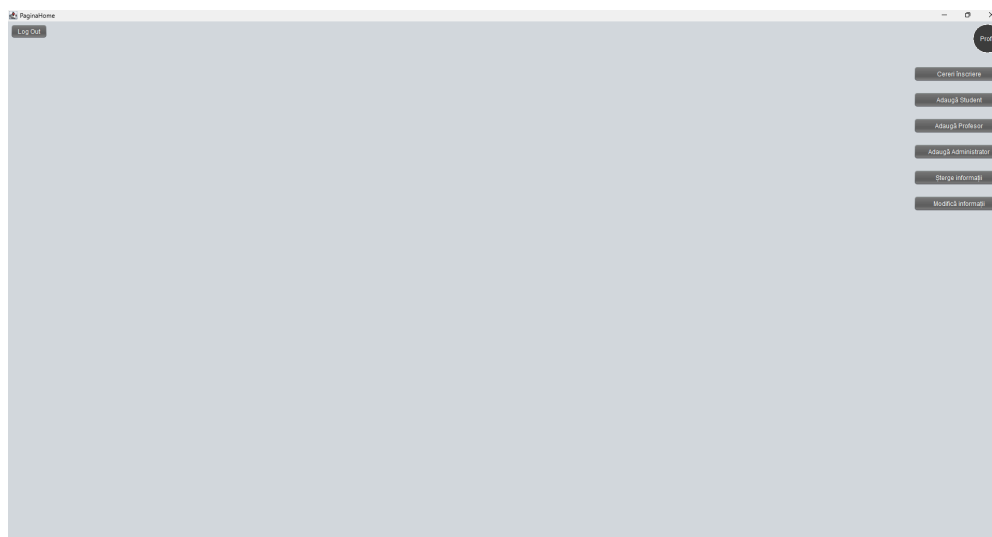


Username:

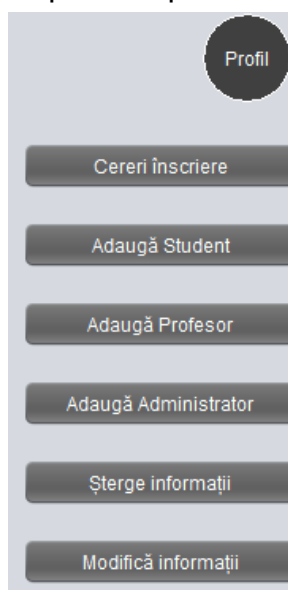
Parola:

Utilizatorul introduce Username-ul și parola iar dacă acestea sunt valide se va deschide Pagina principală pentru tipul de utilizator corespunzător contului respectiv.

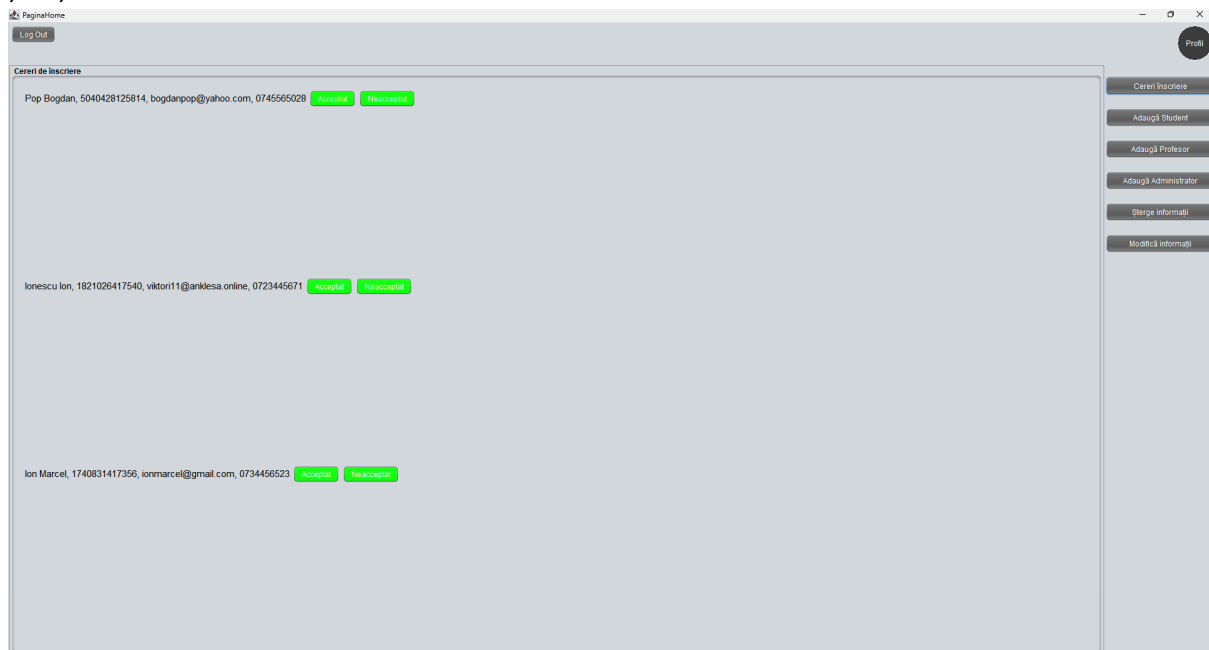
Pagina Administratorului:



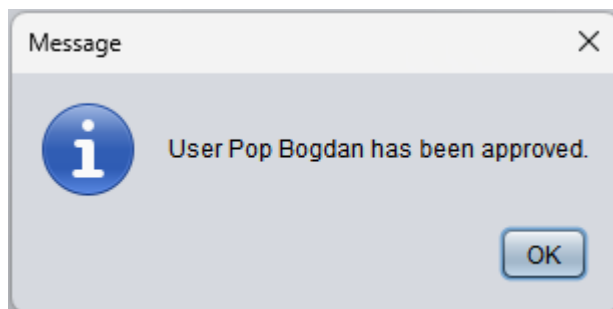
În partea dreapta sus se află butonul de profil unde se pot vizualiza datele utilizatorului conectat. Iar sub acel buton sunt 5 butoane pentru administratori, respectiv 6 pentru superAdministratori.



Butonul “Cerere înscriere” afișează lista utilizatorilor care s-au autentificat și așteaptă aprobare din partea administratorilor.

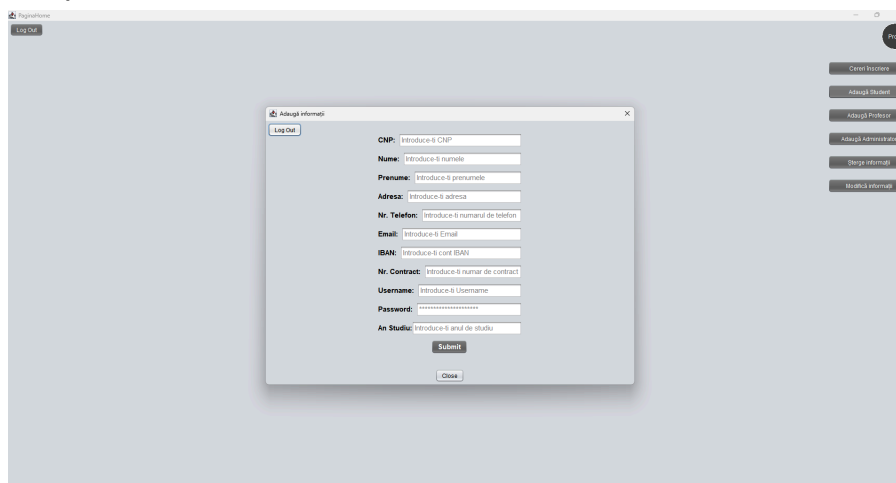


Dacă utilizatorul este acceptat se marchează și rămâne în baza de date



Altfel utilizatorul este șters din baza de date.

Butonul “Adaugă Student” deschide o fereastră ca cea de autentificare în care administratorul poate introduce date pentru a crea un cont pentru un utilizator de tip student.



La fel și pentru **Butonul “Adaugă Profesor”**, se va deschide pagina de autentificare pentru tipul profesor.

Adaugă informații

Log Out

CNP:

Nume:

Prenume:

Adresa:

Nr. Telefon:

Email:

IBAN:

Nr. Contract:

Username:

Password:

Departament:

Submit

Close

Butonul “Adaugă Administrator” va fi vizibil doar pentru Super Administratori. Acestea doar au acces la datele altor administratori și pot sa adauge sau sa modifice date.

Adaugă informații

Log Out

CNP:

Nume:

Prenume:

Adresa:

Nr. Telefon:

Email:

IBAN:

Nr. Contract:

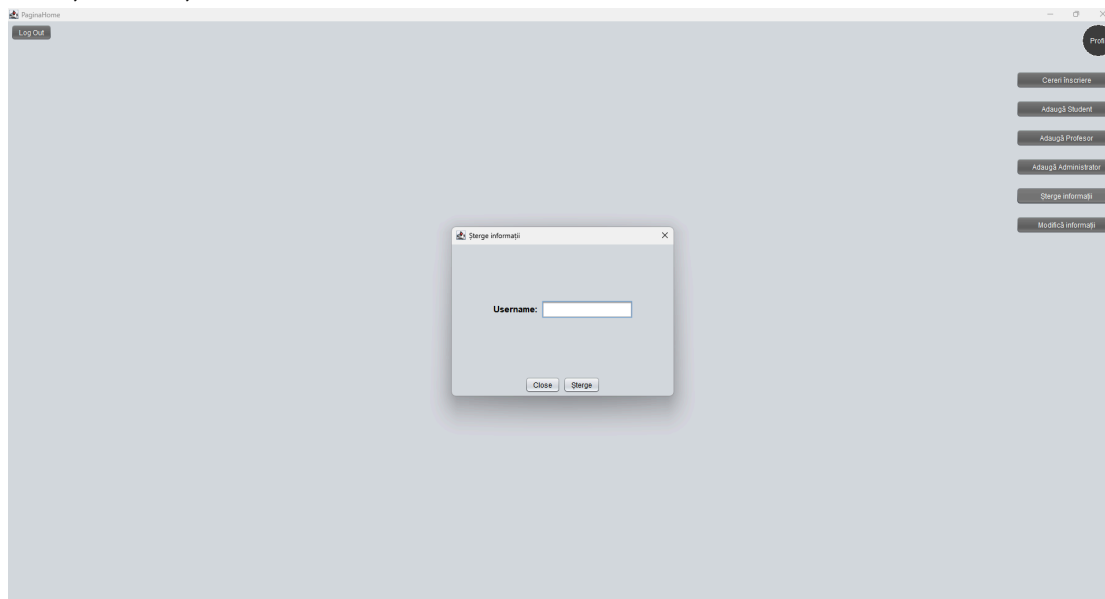
Username:

Password:

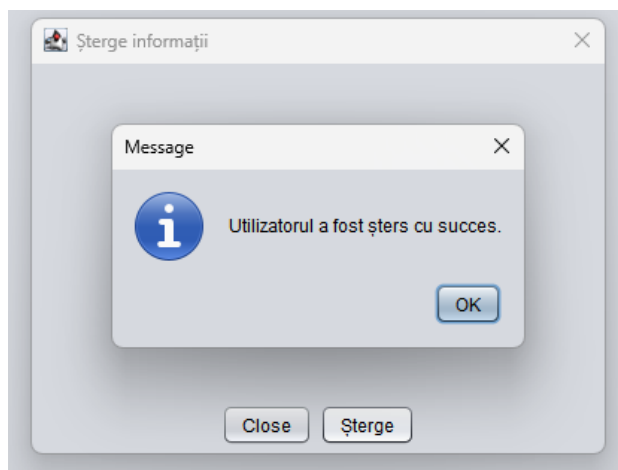
Submit

Close

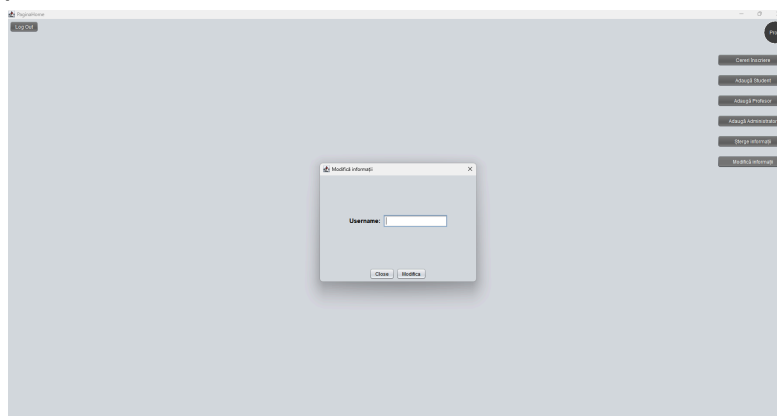
Butonul “Șterge Informații” va deschide o fereastră unde administratorul poate introduce username-ul unui utilizator la datele căruia are acces pe care dorește să-l ștergă din baza de date.



După apăsarea butonului “Șterge” se va primi o confirmare dacă ștergerea a fost făcută cu succes.



Butonul “Modifică Informații” va deschide o fereastră unde administratorul poate introduce username-ul unui utilizator la datele căruia are acces.



Dacă acesta este valid se va deschide o nouă fereastră în care va vedea datele utilizatorului, care vor putea fi modificate și apoi salvate.

The image shows two overlapping windows from a web application. The background window is titled "Modifică informații" and contains a text input field labeled "Username:" with the value "marcuandrei1". Below the field are two buttons: "Close" and "Modifica". The foreground window is titled "Modifică Utilizator" and displays a form for editing user details. The form fields and their values are as follows:

Field	Value
CNP:	5040427125814
Nume:	Marcu
Prenume:	Andrei
IdAdresa:	3
Nr. Telefon:	0745565026
Email:	marcuandrei2018@yahoo.com
IBAN:	RO82RZBR8781713469179946
Nr. Contract:	30223
Username:	marcuandrei1
Password:	****
Adresa:	Diane 77

At the bottom right of the "Modifică Utilizator" window are two buttons: "OK" and "Cancel".

Pagina Profesorului:

The image shows a web application interface for a teacher's page. The page title is "Pagina Profesorului". On the left side, there is a sidebar with the following navigation links: "Log Out", "Inscriere Curs", "Notare procentile", "Notare note", "Visualizare Orar", "Catalog Studenti", and "Descarcă Catalog". The main content area is currently empty. In the top right corner, there is a circular profile icon labeled "Profil".

Pentru **butonul de înscriere curs**:

Fiecare profesor se poate înscrie la una din materiile din baza de date.



The image shows a web interface with a button labeled "Inscriere Curs" at the top. Below it is a form titled "Inscriere Curs". Inside the form, there is a text input field with the placeholder text "Introduce-ti disciplina!". Below the input field is a button labeled "Submit".

Butonul de notare procente:

Fiecare profesor poate să își selecteze pe rând una din materiile la care predă din ComboBox-ul de mai jos și își poate introduce, in functie de curs, seminar si laborator, procentul dorit pentru a calcula notele studenților.



The image shows a web interface with a button labeled "Notare procente" at the top. Below it is a form titled "Notare procente". Inside the form, there is a dropdown menu with the text "Baze de date". Below the dropdown menu, there are three sections, each with a label, an input field, and a submit button:

- Procente curs:** Input field, Submit1 button
- Procente seminar:** Input field, Submit2 button
- Procente laborator:** Input field, Submit3 button

Butonul de notare note:

Profesorul trebuie sa selecteze prima data una din materiile predate, dupaia poate selecta unul din studenții pe care îi are la materia respectivă și poate dupaia sa introduca notele pentru curs, seminar si laborator. În urma butonului de submit vor apărea în catalog notele la fiecare disciplina pentru fiecare student.

Notare note

Notare note

Baze de date

marcelo

Nota curs:

Nota seminar:

Nota laborator:

Submit

Vizualizare orar:

JANUARY 2025						
lu.	ma.	mie.	joi.	vi.	sâ.	du.
		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		

Pentru catalog:

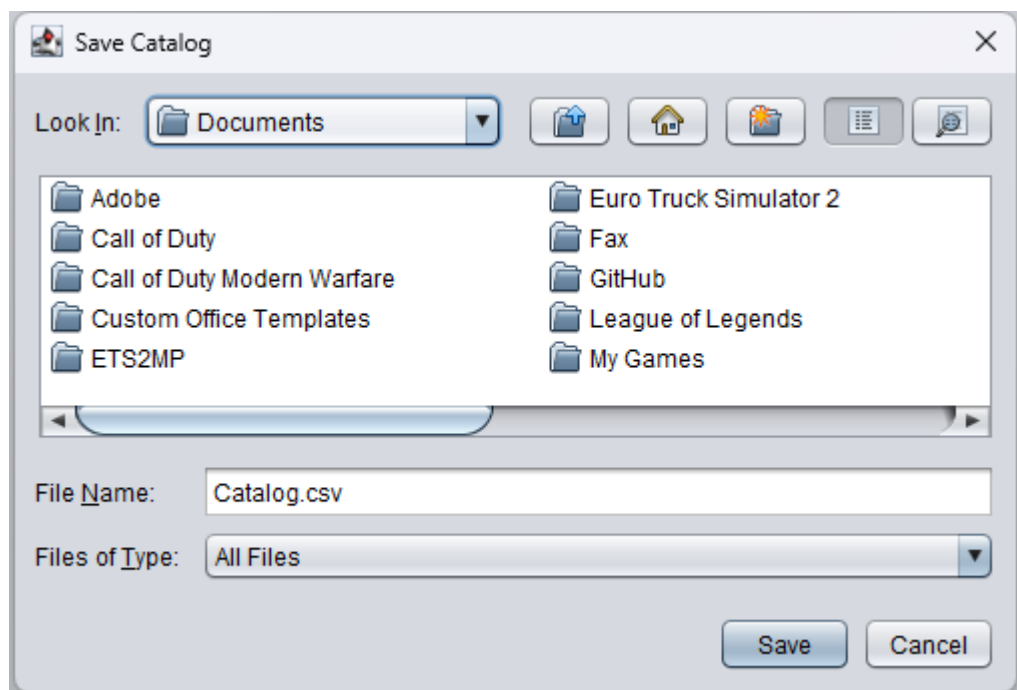
Fiecare student a profesorului în funcție de materie apare în catalog. Acești studenți au în dreptul lor notele de la curs, seminar și laborator și nota finală la materia respectivă.

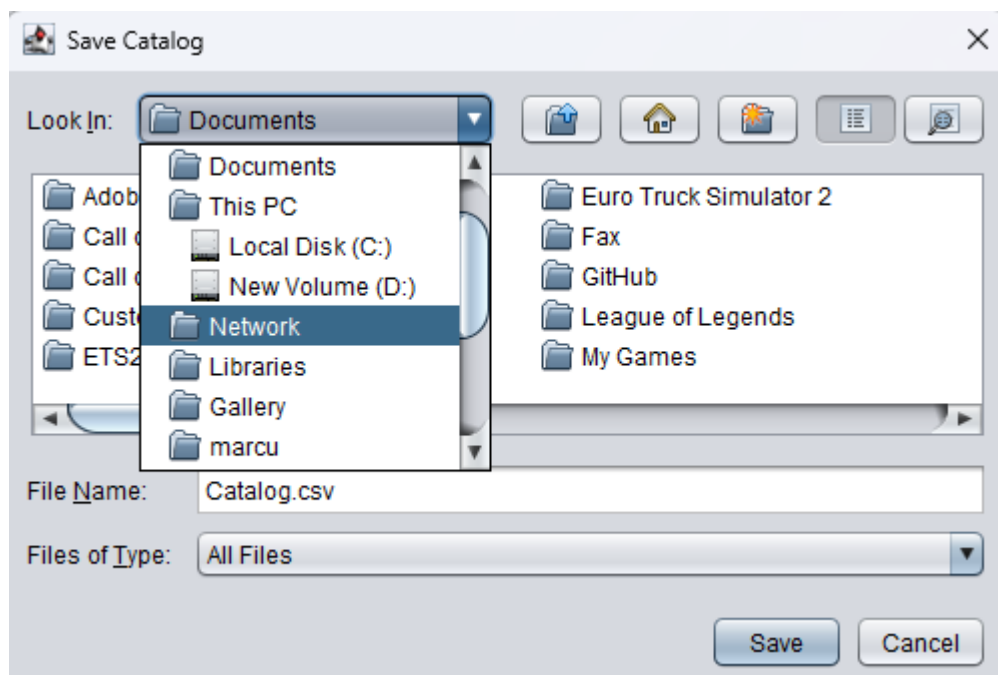
<div>Catalog Studenți</div> <div>Descarcă Catalog</div>		
Materie	Nume Student	Nota Curs
Analiza matematica I	marcelo	2
Baze de date	marcelo	1
Proiectare logica	Ionita	0

Nota Seminar	Nota Laborator	Nota Finala
6	4	3
2	3	2
0	0	0

La apăsarea butonului “Descarca catalog” se va descarca un document de tip .csv și va putea fi downloadat și deschis în excel sau alt document.

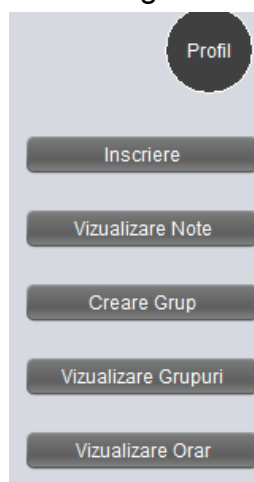
Salvarea documentului:





Pagina Studentului:

Pagina studentilor contine urmatoarele campuri:



Inscriere:

Analiza matematica I

Se poate selecta oricare din materiile din baza de date disponibile.

Vizualizare note:

Apar notele studentului intr-un catalog in functie de cursurile la care este inscris.

Materie	Nota Curs	Nota Seminar	Nota Laborator	Nota Finala
Analiza matematica I	2	6	4	3
Baze de date	1	2	3	2

Crearea de grupuri:

Nume Grup:

Introduce-ti numele grupului

Analiza matematica I

Create

Vizualizare grupuri:

Aici apar grupurile la care studentul este inscris. De exemplu:

abc

mnm

Inscrie la un nou grup

Studentul se poate si inscrie la un nou grup pe baza disciplinelor la care este inscris:

Grupuri la care te poti inscrie

Nume: bcd, Disciplina: Analiza matematica I

Inscrie-te

Nume: bd, Disciplina: Baze de date

Inscrie-te

Cautare grup:

Introduce-ti Numele

Search

Vizualizare orar:

Aici apare un orar cu fiecare activitate programata a studentilor:

JANUARY 2025						
lu.	ma.	mie.	joi	vi.	să	du
		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		

5.2 Detalii de implementare funcționalități specifice

Pentru implementarea interfeței grafice am folosit biblioteca java swing care ne-a pus la dispoziție toate componentele și clasele necesare pentru crearea unei interfețe simple și ușor de înțeles pentru utilizatori.

6. Manual de Utilizare

6.1 Înregistrare și Autentificare

Pe pagina inițială a aplicației se afișează 2 opțiuni: “Autentificare” și “Conectare”. În dreptul butonului de autentificare se poate selecta opțiunea tipului de cont care se poate crea.

Odata apasat pe butonul de “autentificare” va apărea o fereastră cu campurile necesare de completat:



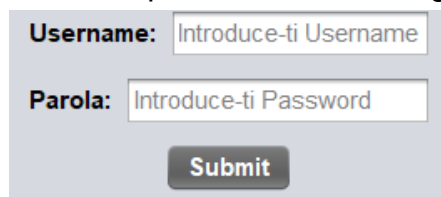
The registration form contains the following fields and labels:

- CNP: Introduce-ti CNP
- Nume: Introduce-ti numele
- Prename: Introduce-ti prenumele
- Adresa: Introduce-ti adresa
- Nr. Telefon: Introduce-ti numarul de telefon
- Email: Introduce-ti Email
- IBAN: Introduce-ti cont IBAN
- Nr. Contract: Introduce-ti numar de contract
- Username: Introduce-ti Username
- Password: *****
- An Studiu: Introduce-ti anul de studiu

A "Submit" button is located at the bottom of the form.

Datele introduse trebuie sa fie valide în funcție de câmp pentru a putea crea contul.

Iar pentru butonul de logare vor apărea câmpurile:



The login form contains the following fields and labels:

- Username: Introduce-ti Username
- Parola: Introduce-ti Password

A "Submit" button is located at the bottom of the form.

Trebuie introduse datele unui cont deja creat: un “username” valid și parola respectivă contului.

6.2 Gestionarea Informațiilor per utilizatori

Pentru fiecare fel de cont (student, profesor, admin) exista butonul de “Profil” care afișează informații specifice fiecărui utilizator.



Informațiile sunt stocate în baza de date în mai multe tabele. Toate datele comune tuturor utilizatorilor sunt puse într-un tabel numit “Utilizator”. Adresele sunt puse într-un alt tabel care este legat de tabelul utilizator prin indexul adresei. Iar datele specifice pentru un rol al utilizatorului sunt puse în 3 tabele separate: profesor, student, administrator, care sunt legate de tabela utilizator prin username.

7. Concluzii și dezvoltării ulterioare

7.1 Analiza funcționalităților curente

Catalog:

Pentru catalogul studentilor se pot modifica ordinea coloanelor cu mouse-ul:

Materie	Nume Student	Nota Curs
Analiza matematica I	marcelo	2
Baze de date	marcelo	1
Analiza matematica I	Ionita	0
Proiectare logica	Ionita	0

Nume Student	Materie	Nota Curs
marcelo	Analiza matematica I	2
marcelo	Baze de date	1
Ionita	Analiza matematica I	0
Ionita	Proiectare logica	0

Prin “drag and drop” se pot muta coloanele între ele.

Autentificare:

În pagina de autentificarea, câmpurile trebuie completate cu date valide, altfel contul nu va putea fi creat iar utilizatorul va primi un feedback:

CNP: 1234

Exemplu (funcția de validare IBAN):

```
public static void verificareIban(String s, JTextField text) throws
IOException {
    if (s.equals("Introduce-ti cont IBAN") || s.length() < 15 ||
s.length() > 34) {
        Color lightRed = new Color(255, 102, 102);
        text.setBackground(lightRed);
    }

    s = s.replaceAll("\\s+", "").toUpperCase();

    String countryCode = s.substring(0, 2);
    Integer expectedLength = 24;//pt RO
    if (expectedLength == null || s.length() != expectedLength) {
        Color lightRed = new Color(255, 102, 102);
        text.setBackground(lightRed);
        throw new IOException("IBAN invalid!");
    }
    String rearranged = s.substring(4) + s.substring(0, 4);
    StringBuilder numericIBAN = new StringBuilder();
    for (char c : rearranged.toCharArray()) {
        if (Character.isLetter(c)) {
            numericIBAN.append(c - 'A' + 10);
        } else {
            numericIBAN.append(c);
        }
    }
    BigInteger ibanNumber = new BigInteger(numericIBAN.toString());
    if (ibanNumber.mod(BigInteger.valueOf(97)).intValue() != 1){
        Color lightRed = new Color(255, 102, 102);
        text.setBackground(lightRed);
        throw new IOException("IBAN invalid!");
    }
    text.setBackground(Color.WHITE);
}
```

Modificare Date:

Când un administrator dorește sa modifice datele unui utilizator se deschide fereastra “Modifica Utilizator” care are un formular ca cel de la autentificare, dar care are datele utilizatorului completate. Datele pot fi modificate iar dacă sunt valide sunt introduse în baza de date în locul celor vechi.

Modifică Utilizator	
CNP:	1740831417356
Nume:	Ion
Prenume:	Marcel
idAdresa:	4
Nr. Telefon:	0734456523
Email:	ionmarcel@gmail.com
IBAN:	RO48PORL5184963831182661
Nr. Contract:	125
Username:	marcelo
Password:	***
Adresa:	Rozelor 3
An Studiu:	2
<input type="button" value="OK"/> <input type="button" value="Cancel"/>	

Algoritmul de programare a orarului:

Se folosește algoritmul “Graph coloring” pentru a putea organiza materiile pentru studenți astfel incat sa nu se suprapună atât cursurile și activitățile, cat și orele de la care sunt programate în funcție de studenți și profesori.

```
private static void initGraph(Graph<GraphNode> graph){
```

```
    try{
        String
url="jdbc:mysql://139.144.67.202:3306/lms?user=lms&password=WHlQjrrRDs5t";
        Connection conn= DriverManager.getConnection(url);

        PreparedStatement stmt=conn.prepareStatement("SELECT idCurs as
ID,idProfesor,Nume,'Curs'as type from curs join disciplina
using(idDisciplina)UNION ALL\n" +
            "SELECT idSeminar,idProfesor,Nume,'Seminar'as type from
seminar join disciplina using(idDisciplina) UNION ALL\n" +
            "SELECT idLaborator,idProfesor,Nume,'Laborator'as type from
laborator join disciplina using(idDisciplina);");
        ResultSet rs=stmt.executeQuery();
        List<GraphNode> nodes=new ArrayList<>();
        List<Integer> idProfesor=new ArrayList<>();
        while(rs.next()){
            GraphNode(0,rs.getInt(1),index,rs.getString(4),rs.getString(3));
```

```

        idProfesor.add(rs.getInt(2));
        index++;
    }
    for (int i = 0; i < idProfesor.size(); i++) {
        for (int j = i + 1; j < idProfesor.size(); j++) {
            // If both nodes have the same idProfesor, add an edge
            if (idProfesor.get(i).equals(idProfesor.get(j))) {
                graph.addEdge(nodes.get(i), nodes.get(j));
            }
        }
    }
    tmt=conn.prepareStatement("SELECT GROUP_CONCAT(nota.idStudent ORDER
BY nota.idStudent ASC SEPARATOR ',') AS student_ids FROM curs JOIN disciplina
USING(idDisciplina) JOIN nota ON nota.idDisciplina = curs.idDisciplina AND
nota.idProfesor = curs.idProfesor GROUP BY curs.idCurs UNION ALL\n" +
        "SELECT GROUP_CONCAT(nota.idStudent ORDER BY nota.idStudent
ASC SEPARATOR ',') AS student_ids FROM seminar JOIN disciplina
USING(idDisciplina) JOIN nota ON nota.idDisciplina = seminar.idDisciplina AND
nota.idProfesor = seminar.idProfesor GROUP BY seminar.idSeminar UNION ALL\n"
+
        "SELECT GROUP_CONCAT(nota.idStudent ORDER BY nota.idStudent
ASC SEPARATOR ',') AS student_ids FROM laborator JOIN disciplina
USING(idDisciplina) JOIN nota ON nota.idDisciplina = laborator.idDisciplina
AND nota.idProfesor = laborator.idProfesor GROUP BY laborator.idLaborator;");

```

```

private static void scheduleOrar(){ no usages
    Graph<GraphNode> graph = new Graph<>();
    initGraph(graph);
}

```

```

case "Curs":
    stmt = conn.prepareStatement( sql: "UPDATE curs set ziua=?,ora=? where idCurs=?");
    stmt.setString( parameterIndex: 1, zileleSaptamanii[zivaCurenta%5]);
    stmt.setTime( parameterIndex: 2, Time.valueOf(activityHours[oraCurenta%6]));
    stmt.setInt( parameterIndex: 3,node.getID());
    stmt.executeUpdate();
    break;
case "Seminar":
    stmt = conn.prepareStatement( sql: "UPDATE seminar set ziua=?,ora=? where idSeminar=?");
    stmt.setString( parameterIndex: 1, zileleSaptamanii[zivaCurenta%5]);
    stmt.setTime( parameterIndex: 2, Time.valueOf(activityHours[oraCurenta%6]));
    stmt.setInt( parameterIndex: 3,node.getID());
    stmt.executeUpdate();
    break;
case "Laborator":
    stmt = conn.prepareStatement( sql: "UPDATE laborator set ziua=?,ora=? where idLaborator=?");
    stmt.setString( parameterIndex: 1, zileleSaptamanii[zivaCurenta%5]);
    stmt.setTime( parameterIndex: 2, Time.valueOf(activityHours[oraCurenta%6]));
    stmt.setInt( parameterIndex: 3,node.getID());
    stmt.executeUpdate();
    break;
default:
    break;

```

7.2 Dezvoltări ulterioare

- Adaugare imagini pe grupurile de chat
- Adaugare poze (poze de profil, de aplicatie, de grupuri)
- Creare video-chat-uri
- Adaugarea de rapoarte pentru studenți
- Îmbunătățirea securității și a validării datelor
- Crearea pentru aplicații mobile