MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Documentatie Proiect

la disciplina

Baze de date

Platforma de studiu

Dregan Vlad-Vasile

Marian Andrei-Alexandru

Grupa: 30221

An academic: 2023-2027

Cuprins

1. Introducere

1.1 Descrierea Generală a Proiectului

Proiectul reprezintă un sistem informatic destinat gestiunii unei platforme de studiu, având ca scop principal facilitarea procesului educațional și gestionarea eficientă a interacțiunilor dintre studenți, profesori și administratori. Sistemul se bazează pe o bază de date MySQL, iar interacțiunea utilizatorilor se realizează exclusiv prin intermediul unei interfețe grafice intuitive.

Aplicatia permite gestionarea informațiilor despre utilizatori (studenți, profesori și administratori), organizarea și desfășurarea activităților educaționale (cursuri, seminarii, laboratoare) și oferă funcționalități avansate precum: alocarea automata a studenților și profesorilor la activități, crearea grupurilor de studiu și gestionarea notelor. Sistemul suportă autentificarea multiplu rol, incluzând utilizatori cu drepturi extinse precum super-administratorii, care pot gestiona toate tipurile de date și utilizatori.

De asemenea, aplicația pune accent pe personalizarea și automatizarea activităților, oferind funcții precum planificarea activităților în calendar, generarea rapoartelor și descărcarea cataloagelor sau listelor de activități.

1.2 Scopul si Obiectivele

Scopul principal al proiectului este dezvoltarea unui sistem informatic modern și eficient pentru gestiunea activităților educaționale, care să sprijine toate părțile implicate în procesul de învățământ. Prin intermediul acestui sistem, se dorește reducerea efortului administrativ și optimizarea procesului educațional.

Obiective specifice:

1. **Managementul utilizatorilor:**
   - Crearea, modificarea și ștergerea informațiilor despre studenți, profesori și administratori.
   - Posibilitatea autentificării și deautentificării utilizatorilor, cu vizualizarea datelor personale imediat după logare.
2. **Organizarea cursurilor și activităților:**
   - Definirea cursurilor, asignarea profesorilor și înrolarea studenților.
   - Planificarea activităților în calendar și prevenirea suprapunerilor.
3. **Gestiunea notelor:**
   - Alocarea de note pe tipuri de activități și calcularea mediei finale pe baza ponderilor stabilite de profesori.
4. **Crearea grupurilor de studiu:**

    - Permisiunea studenților de a crea, adăuga și participa la grupuri de studiu.
    - Automatizarea procesului de validare și notificarea anulării activităților din grupuri.
5. **Interfață intuitivă:**
    - Oferirea unei interfețe ușor de utilizat pentru toate categoriile de utilizatori.
    - Generarea de rapoarte și fișiere descărcabile (cataloage, liste de activități).

Prin implementarea acestor obiective, sistemul informatic contribuie la o gestiune modernă a procesului educațional, reducând timpul necesar administrării și sporind eficiența în interacțiunile dintre utilizatori.

2. Tema Proiectului

2.1 Prezentarea Aplicatiei

Aplicația dezvoltata în cadrul acestui proiect reprezintă o platformă educațională integrată, care permite gestionarea utilizatorilor, activităților și datelor aferente procesului de învățământ. Aceasta este structurată în jurul unei baze de date robuste și include o interfață grafică pentru interacțiunea utilizatorilor cu sistemul.

Platforma este dedicată utilizatorilor din cadrul unei instituții de învățământ și suportă următoarele tipuri de roluri:

- **Studenți**: Pot vizualiza activitățile și notele, se pot înscrie sau retrage de la cursuri și pot participa la grupuri de studiu.
- **Profesori**: Pot gestiona activitățile asociate cursurilor, pot nota studenții și genera rapoarte.
- **Administratori**: Pot gestiona utilizatorii și structura cursurilor, având acces la funcții avansate de organizare.

Interfața aplicației este concepută pentru a fi ușor de utilizat, facilitând accesul rapid la funcționalități și asigurând o experiență fluentă pentru utilizatori.

2.2 Funcționalități și Utilizatori

1. Studenți:

- Înscriere la cursuri și activități.
- Vizualizarea și descărcarea programului zilnic sau complet.
- Vizualizarea notelor pentru activități și media finală.
- Participarea la grupuri de studiu și gestionarea activităților din grup.

2. Profesori:

- Asignarea și planificarea activităților asociate cursurilor.
- Gestionarea cataloagelor și notarea studenților.
- Generarea și descărcarea rapoartelor privind activitățile și rezultatele studenților.

3. Administratori:

- Crearea, modificarea și ștergerea utilizatorilor și cursurilor.
- Căutarea utilizatorilor după criterii specifice (nume, tip utilizator).
- Asignarea profesorilor la cursuri și configurarea structurii activităților.

Funcționalitățile sunt gândite să optimizeze timpul alocat activităților administrative și să creeze un mediu colaborativ pentru studenți și profesori.

3. Descrierea Bazei de Date

3.1 Structura generala

Baza de date a fost proiectată pentru a gestiona informațiile dintr-o platformă educațională, incluzând utilizatorii (studenți, profesori, administratori), activitățile academice, cursurile și alte operații legate de gestiunea acestora. Structura generală este compusă din multiple tabele interconectate, fiecare având un rol specific. Principalele categorii de informații stocate sunt:

- **Utilizatorii** și informațiile asociate lor (roluri, date personale, statut);
- **Activitățile și cursurile**: inclusiv detalii despre orar, profesori și participanți;
- **Evaluarea și catalogul**: note pentru activitățile studenților și structura procentuală;
- **Grupurile de studiu**: mesajele și activitățile organizate de studenți.

3.2 Detalii specifice

Pentru fiecare tabel important, iată descrierea:

1. **Utilizator**
    - Colțurile de bază ale aplicației, stocând datele generale despre utilizatori, precum ID, nume și parola.
    - Legat de **Rol** pentru a diferenția între studenți, profesori și administratori.
2. **Utilizator_Info**
    - Detalii suplimentare pentru utilizatori, incluzând CNP, adresă, e-mail, cont IBAN și contract.
3. **Rol**
    - Definește tipul de utilizator (student, profesor, administrator).

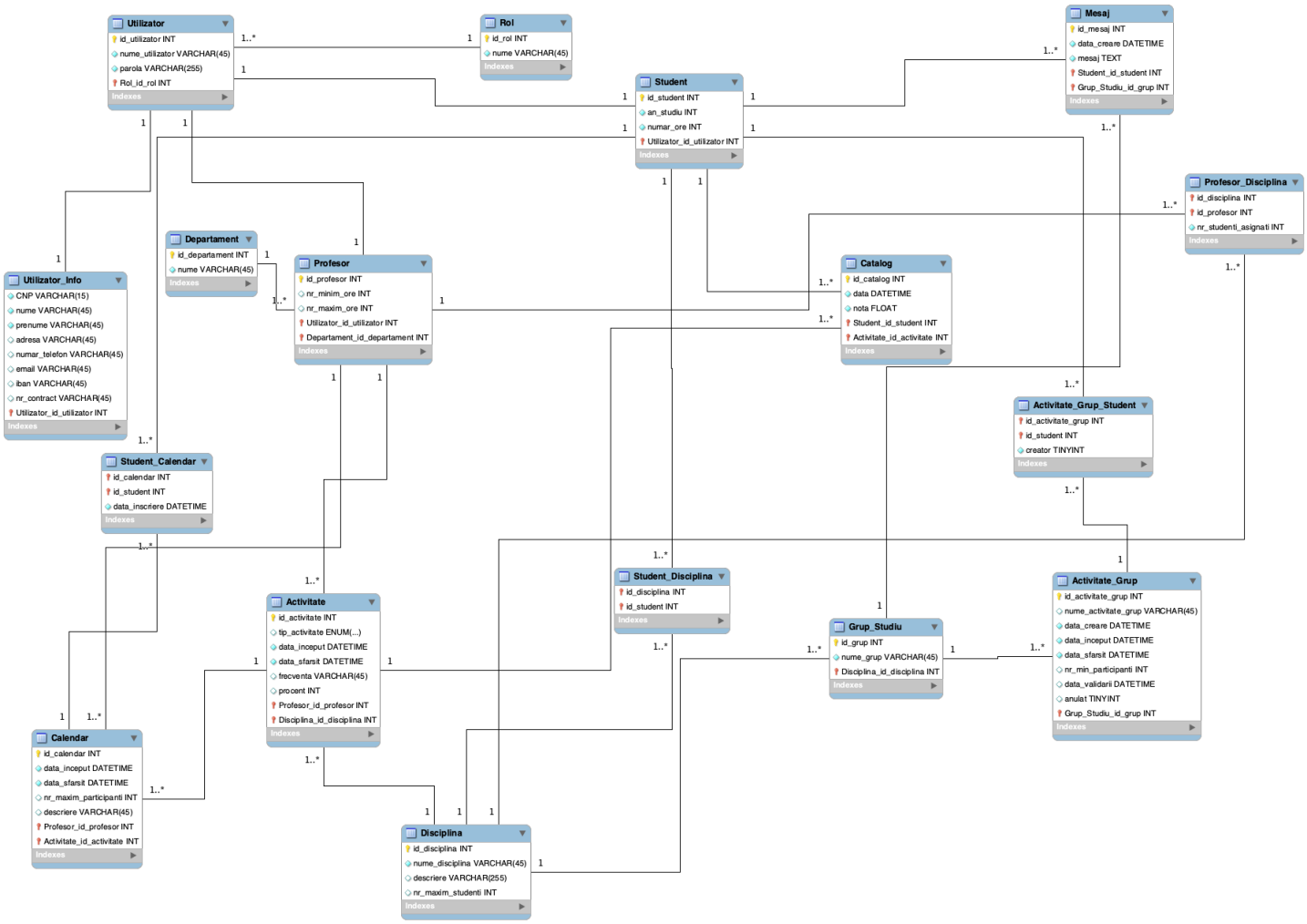
4. **Student**
    - Extinde informațiile despre utilizatorii de tip student, incluzând anul de studiu și numărul de ore necesare.
5. **Profesor**
    - Conține detalii despre profesori, cum ar fi departamentul din care fac parte și limitele orare.
6. **Disciplina**
    - Informații despre cursuri: nume, descriere, număr maxim de studenți.
7. **Activitate**
    - Stochează activitățile desfășurate pentru fiecare curs (laborator, seminar, colocviu).
8. **Calendar**
    - Orarul activităților, incluzând data de început și sfârșit, profesorii asociați și numărul maxim de participanți.
9. **Procent_Activitate**
    - Procentajele pentru fiecare tip de activitate (laborator, seminar, examen) stabilite de profesori.
10. **Grup_Studiu**
    - Grupuri create de studenți pentru colaborare suplimentară, incluzând mesajele asociate.
11. **Catalog**
    - Notele studenților pentru fiecare activitate.

3.3 Nivelul de normalizare

Baza de date este normalizată până la **Forma Normală 3 (3NF)**, pentru a evita redundanța și inconsistența datelor:

1. **Forma Normală 1 (1NF):** Fiecare tabel are coloane atomice, iar toate valorile sunt unice și neambigue.
2. **Forma Normală 2 (2NF):** Toate tabelele sunt dependente complet de cheia primară.
3. **Forma Normală 3 (3NF):** Nu există dependențe tranzitive între coloane care nu sunt chei primare. Informațiile suplimentare, cum ar fi detaliile utilizatorilor și departamentele, sunt stocate separat.

➢ *Diagrama UML a bazei de date*



4. Elemente de programare a functionalitatilor

- **Introducerea unui utilizator în baza de date**

Pentru a crea un nou utilizator în aplicație, se utilizează următoarea interogare SQL:

```
String query = "INSERT INTO Utilizator (id_rol, nume_utilizator, parola) VALUES (?, ?, ?)";
```

Această interogare permite introducerea unui utilizator, specificând:

- **id_rol** – identificatorul rolului utilizatorului (student, profesor, administrator etc.).
- **nume_utilizator** – numele de utilizator.
- **parola** – parola asociată utilizatorului.

- **Introducerea informațiilor personale ale utilizatorilor**

Pentru stocarea informațiilor personale asociate unui utilizator (CNP, nume, prenume etc.), se folosește următoarea interogare:

```
String query = "INSERT INTO Utilizator_Info (id_utilizator, CNP, nume, prenume, adresa, numar_telefon, email, iban, nr_contract) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)";
```

Această interogare permite asocierea informațiilor personale unui utilizator deja creat în tabelul **Utilizator**.

- **Creare unui profesor**

Pentru a introduce un nou profesor, se folosește următoarea interogare:

```
String query = "INSERT INTO Profesor (id_departament, id_utilizator, nr_minim_ore, nr_maxim_ore) VALUES (?, ?, ?, ?)";
```

Aceasta permite:

- Asocierea unui profesor cu un departament.
- Stabilirea unui număr minim și maxim de ore pentru profesor.

- **Crearea unui student**

Pentru a adăuga un student, se folosește următoarea interogare:

```
String query = "INSERT INTO Student (id_utilizator, an_studiu, numar_ore) VALUES (?, ?, ?)";
```

Aceasta permite introducerea unui student și specificarea:

- Anului de studiu.
- Numărului de ore atribuit.

➢ Obținerea notelor finale pentru un student

Această funcționalitate permite calcularea și afișarea notelor finale pentru un student, utilizând mai multe interogări SQL. Funcția traversează baza de date pentru a extrage informații legate de disciplinele studiate, activitățile asociate și procentajul fiecărei activități.

```
String queryIdUtilizator = "SELECT id_utilizator FROM Utilizator WHERE nume_utilizator = ?";
String queryIdStudent = "SELECT id_student FROM Student WHERE id_utilizator = ?";
String queryIdDisciplina = "SELECT id_disciplina FROM Student_Disciplina WHERE id_student = ?";
String queryNumeDisciplina = "SELECT nume_disciplina FROM Disciplina WHERE id_disciplina = ?";
String queryIdActivitate = "SELECT id_activitate FROM Activitate WHERE id_disciplina = ?";
String queryProcent = "SELECT procent FROM Activitate WHERE id_activitate = ?";
String queryNota = "SELECT nota FROM Catalog WHERE id_activitate = ? AND id_student = ?";
```

➢ Salvarea unui mesaj într-un grup

Funcționalitatea setMesaj permite unui utilizator să adauge un mesaj într-un grup specific, stocând datele în baza de date.

```
String queridUtilizator = "SELECT id_utilizator FROM Utilizator WHERE nume_utilizator = ?";
String queryidStudent = "SELECT id_student FROM Student WHERE id_utilizator = ?";
String queryInsert = "INSERT INTO Mesaj (id_grup, id_student, data_creare, mesaj) VALUES (?, ?, ?, ?)";
```

➢ Modificarea informațiilor personale ale unui utilizator

Funcționalitatea modificareInformatii permite actualizarea informațiilor personale ale unui utilizator în tabelul Utilizator_Info. Aceasta este accesibilă exclusiv administratorilor sau super-administratorilor, conform cerințelor aplicației.

```
String query = "UPDATE Utilizator_Info SET CNP = ?, nume = ?, prenume = ?, adresa = ?, numar_telefon = ?, email = ?, iban = ?, nr_contract = ? WHERE id_utilizator = ?";
```

➢ Atribuirea unui student la o disciplină

Funcționalitatea assignStudentToDisciplina permite alocarea unui student la o disciplină specifică, actualizând în același timp numărul de studenți asociați unui profesor care predă disciplina respectivă.

```
String queryIdDisciplina = "SELECT id_disciplina FROM Disciplina WHERE nume_disciplina = ?";
String querySelect = "SELECT id_profesor, nr_studenti_asignati FROM Profesor_Disciplina WHERE id_disciplina = ?";
String queryUpdateProfesor = "UPDATE Profesor_Disciplina SET nr_studenti_asignati = nr_studenti_asignati + 1 WHERE id_profesor = ?";
String queryInsertStudentDisciplina = "INSERT INTO Student_Disciplina (id_disciplina, id_student) VALUES (?, ?)";
```

- **Crearea unei activități**

Funcționalitatea CreareActivitate permite adăugarea unei activități noi în cadrul sistemului, asociind-o cu o disciplină și un profesor. Aceasta presupune mai multe etape pentru identificarea relațiilor și introducerea datelor în tabelul Activitate.

```
String queryIdDisciplina = "SELECT id_disciplina FROM Disciplina WHERE nume_disciplina = ?";
String queryIdUtilizator = "SELECT id_utilizator FROM Utilizator WHERE nume_utilizator = ?";
String queryIdProfesor = "SELECT id_profesor FROM Profesor WHERE id_utilizator = ?";
String queryInsert = "INSERT INTO Activitate (id_disciplina, id_profesor, tip_activitate, data_inceput, data_sfarsit, frecventa, procent) VALUES (?, ?, ?, ?, ?, ?, ?)";
```

- **Adăugarea unei note**

Funcția insertNota permite adăugarea unei note în tabelul Catalog asociind o activitate, un student și data la care a fost înregistrată nota.

```
String query = "INSERT INTO Catalog (id_activitate, id_student, data, nota) VALUES (?, ?, ?, ?)";
```

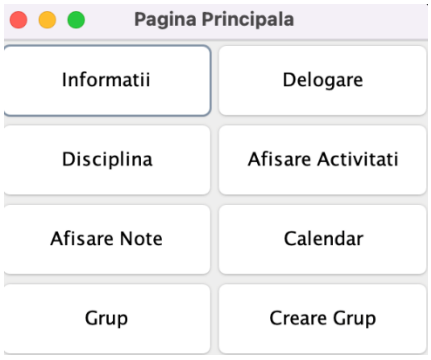
5. Interfața Grafică a Utilizatorului (GUI)

În cadrul aplicației de gestiune a platformei de studiu, Interfața Grafică a Utilizatorului (GUI) joacă un rol esențial în facilitarea interacțiunii între utilizatori și aplicație. Aceasta va fi adaptată în funcție de tipul de utilizator, asigurând accesul la funcționalități specifice.

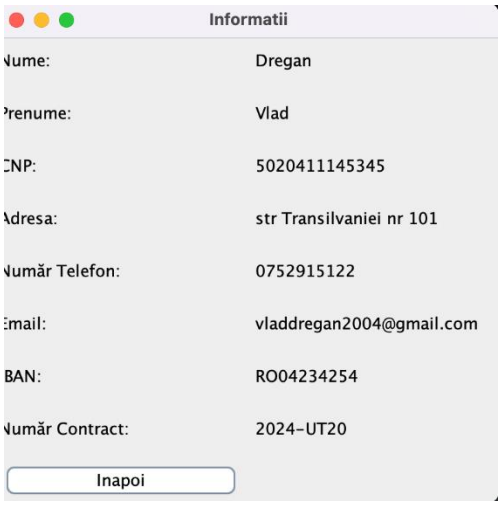
5.1 Descrierea funcționalităților per tip de utilizator

- **Student**

- **Autentificare**: Studenții se loghează cu un cont unic, unde vor introduce datele de acces (CNP, parolă).
- **Vizualizare date personale**: După autentificare, studenții își pot vizualiza datele personale, dar nu le pot modifica.
- **Înscriere la cursuri**: Studenții pot vizualiza lista de cursuri disponibile și se pot înscrie la activitățile didactice (cursuri, seminarii, laboratoare), ținând cont de locurile disponibile și de conflictele orare.
- **Vizualizare activități**: Studenții pot consulta activitățile planificate pentru ziua respectivă și pot vedea toate activitățile la care sunt înscriși.
- **Vizualizare note**: Studenții pot vizualiza notele pentru activitățile la care au participat și pot consulta istoricul acestora.
- **Grupuri de studiu**: Studenții se pot înscrie în grupuri de studiu pentru cursuri și pot interacționa cu colegii, adăugând activități și mesaje.



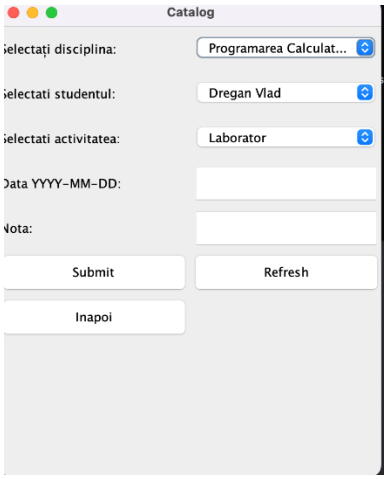
Pagina principala aplicatie ***Pagina principala student***



Afisare informatii

- **Profesor**

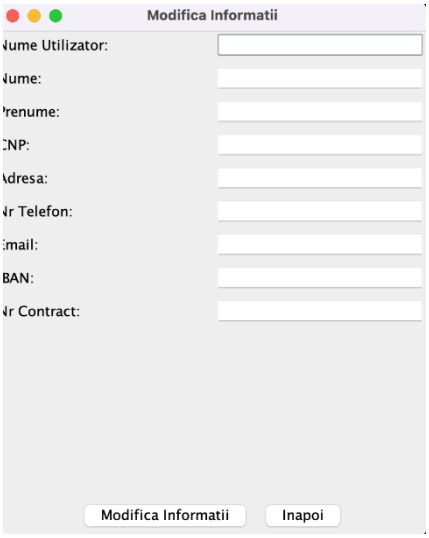
  - **Autentificare**: Profesorii se loghează folosind un cont dedicat.
  - **Vizualizare date personale**: Profesorii pot vizualiza, dar nu pot modifica, datele personale.
  - **Gestionare cursuri**: Profesorii pot vizualiza cursurile la care sunt asignați și pot modifica programul acestora, stabilind tipurile de activități (curs, seminar, laborator).
  - **Calendar activități**: Profesorii pot adăuga activități în calendar, specificând data, ora, numărul de participanți și locația.
  - **Catalog**: Profesorii pot accesa un catalog digital cu studenții și le pot adăuga note pentru fiecare activitate.
  - **Vizualizare activități**: Profesorii pot vizualiza activitățile din ziua respectivă, precum și toate activitățile la care sunt asignați.



Fereastra Catalog

- **Administrator**

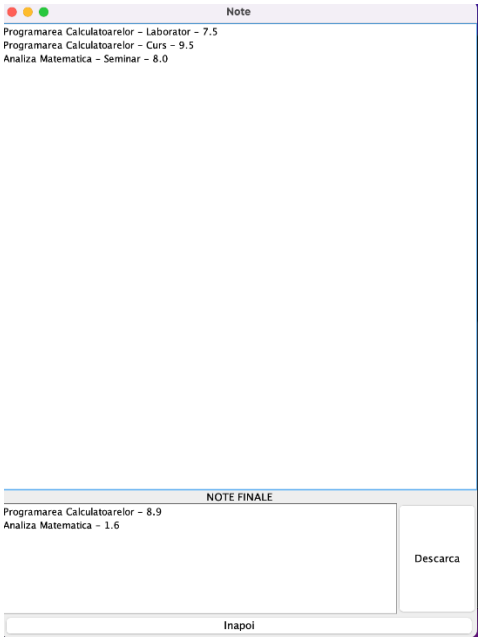
  - **Autentificare**: Administratorii se loghează cu datele lor specifice.
  - **Gestionare utilizatori**: Administratorii pot adăuga, modifica și șterge utilizatori (studenți, profesori, alți administratori) din sistem.
  - **Căutare utilizatori**: Administratorii pot căuta utilizatori după nume și filtra acestora după tip.
  - **Asignare profesori la cursuri**: Administratorii pot asigna profesori la cursuri și pot vizualiza studenții înscriși la cursurile respective.
  - **Vizualizare activități și cursuri**: Administratorii pot vizualiza activitățile din platformă, inclusiv cursurile și notele acestora.



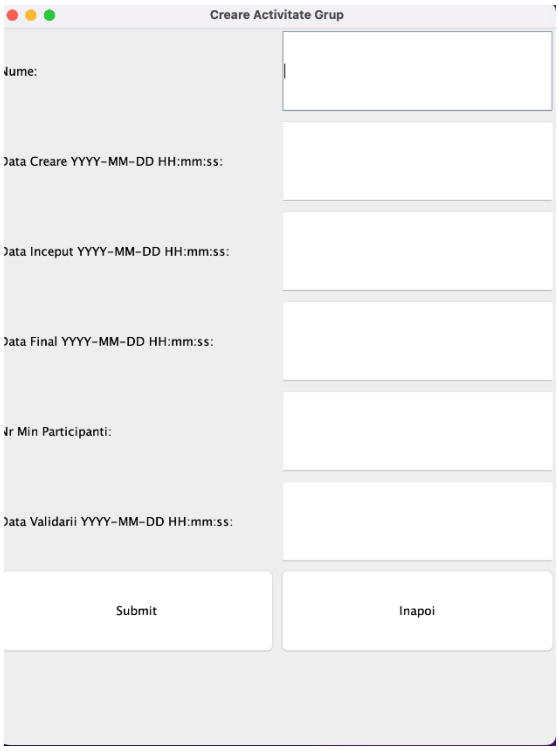
Fereastra modificare informatii - administrator

- **Super-Administrator**

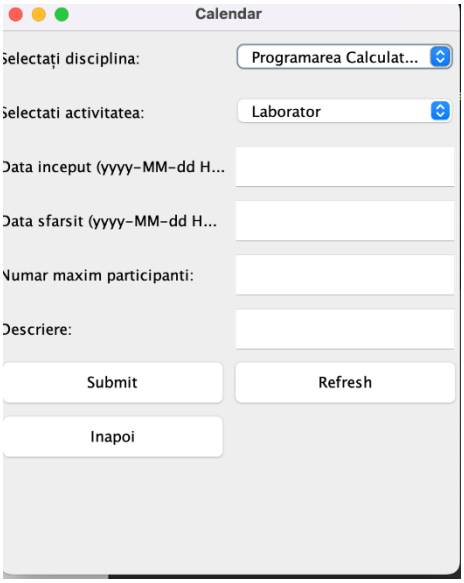
  - **Autentificare**: Super-administratorii se loghează cu un cont dedicat.
  - **Gestionare administratori**: Super-administratorii au acces la funcționalități suplimentare, inclusiv modificarea conturilor de administrator.
  - **Acces total asupra tuturor funcționalităților**: Super-administratorii au acces la toate funcțiile aplicației, inclusiv vizualizarea și modificarea oricăror informații despre utilizatori și activități.



Fereastra note student



Fereastra Creare Activitate Grup



Fereastra Calendar

5.2 Detalii de implementare funcționalități specifice

➢ *Log-in*

Funcționalitatea de **Log-in** permite utilizatorilor să se autentifice în sistem folosind numele de utilizator și parola. Aceasta validează credențialele introduse prin compararea lor cu datele stocate în baza de date.

```
public boolean Logare()  1 usage
{
    String query = "SELECT parola, id_rol FROM Utilizator WHERE nume_utilizator = ?";
    try (Connection con = getConnection();
         PreparedStatement preparedStatement = con.prepareStatement(query))
    {
        preparedStatement.setString(1, numeUtilizator);
        ResultSet resultSet = preparedStatement.executeQuery();
        if (resultSet.next()) {
            String parolaBD = resultSet.getString("parola");
            id_rol = resultSet.getInt("id_rol");
            if (parola.equals(parolaBD)) {
                return true;
            }
            else {
                return false;
            }
        }

    } catch (Exception e) {
        e.printStackTrace();
        return false;
    }
    return false;
}
```

➢ *Creare utilizator*

Funcționalitatea de **Creare utilizator** permite adăugarea unui nou utilizator în baza de date, specificând rolul acestuia, numele de utilizator și parola. După inserare, metoda returnează un indicator de succes și stochează ID-ul utilizatorului generat automat.

```
public boolean creareUtilizator()  1 usage
{
    String query = "INSERT INTO Utilizator (id_rol, nume_utilizator, parola) VALUES (?, ?, ?)";
    try (Connection con = getConnection();
         PreparedStatement preparedStatement = con.prepareStatement(query, Statement.RETURN_GENERATED_KEYS))
    {
        preparedStatement.setInt(1, id_rol);
        preparedStatement.setString(2, nume_utilizator);
        preparedStatement.setString(3, parola);
        int rows = preparedStatement.executeUpdate();
        ResultSet generatedKeys = preparedStatement.getGeneratedKeys();
        if (generatedKeys.next()) {
            id_utilizator = generatedKeys.getInt(1);
        }
        if (rows > 0)
            return true;
        else
            return false;
    }catch (SQLException e)
    {
        e.printStackTrace();
        return false;
    }
}
```

➢ *Creare grup de studiu*

Funcționalitatea de **Creare grup de studiu** permite adăugarea unui nou grup asociat unei discipline existente, identificând disciplina pe baza numelui său și înregistrând grupul în baza de date. Metoda returnează un indicator de succes, confirmând dacă grupul a fost creat cu succes.

```
public boolean creare(String numeDisciplina, String numeGrup)
{
    String queryIdDisciplina = "SELECT id_disciplina FROM Disciplina WHERE nume_disciplina = ?";
    String queryInsert = "INSERT INTO Grup_Studiu (id_disciplina, nume_grup) VALUES (?, ?)";

    try(Connection con = getConnection())
    {
        PreparedStatement preparedStatement = con.prepareStatement(queryIdDisciplina);
        preparedStatement.setString(1, numeDisciplina);
        ResultSet resultSet = preparedStatement.executeQuery();
        int idDisciplina = -1;
        if(resultSet.next())
        {
            idDisciplina = resultSet.getInt("id_disciplina");
            //System.out.println(idDisciplina);
        }
        PreparedStatement preparedStatement2 = con.prepareStatement(queryInsert);
        preparedStatement2.setInt(1, idDisciplina);
        preparedStatement2.setString(2, numeGrup);
        //System.out.println(idDisciplina + " " + numeGrup);
        int rows = preparedStatement2.executeUpdate();
        if(rows > 0)
            return true;
        else return false;
    }catch(Exception e)
    {
        e.printStackTrace();
        return false;
    }
}
```

➢ *Inserare nota*

Funcționalitatea de **Inserare notă în catalog** permite înregistrarea unei note pentru un student la o activitate specifică, stocând detalii precum identificatorii activității și studentului, data înregistrării și nota acordată. Metoda returnează un indicator de succes, confirmând dacă înregistrarea a fost realizată.

```
public boolean insertNota()  1 usage
{
    String query = "INSERT INTO Catalog (id_activitate, id_student, data, nota) VALUES (?, ?, ?, ?)";

    try(Connection con = getConnection())
    {
        PreparedStatement preparedStatement = con.prepareStatement(query);
        preparedStatement.setInt(1, idActivitate);
        preparedStatement.setInt(2, idStudent);
        preparedStatement.setString(3, data);
        preparedStatement.setFloat(4, nota);
        int rows = preparedStatement.executeUpdate();
        if (rows > 0) {
            return true;
        }
        else return false;

    }catch(SQLException e)
    {
        e.printStackTrace();
        return false;
    }
}
```

➢ *Modificare informatii (admin)*

Funcționalitatea de **Modificare informații utilizator** permite administratorului să actualizeze datele personale ale unui utilizator, cum ar fi CNP, nume, prenume, adresă, număr de telefon, email, IBAN și numărul de contract. Modificările sunt aplicate în baza de date pentru utilizatorul identificat prin id_utilizator. Metoda returnează un indicator de succes pentru a confirma dacă actualizarea a fost realizată.

```
public boolean modificareInformatii(){  1 usage
    String query = "UPDATE Utilizator_Info SET CNP = ?, nume = ?, prenume = ?, adresa = ?, numar_telefon = ?, email = ?, iban = ?, nr_contract = ? WHERE id_utilizator = ?";
    try(Connection con = getConnection())
    {
        PreparedStatement preparedStatement = con.prepareStatement(query);
        preparedStatement.setString(1, CNP);
        preparedStatement.setString(2, nume);
        preparedStatement.setString(3, prenume);
        preparedStatement.setString(4, adresa);
        preparedStatement.setString(5, numarTelefon);
        preparedStatement.setString(6, email);
        preparedStatement.setString(7, iban);
        preparedStatement.setString(8, nrContract);
        preparedStatement.setInt(9, idUtilizator);
        preparedStatement.executeUpdate();
        int rows = preparedStatement.executeUpdate();
        if (rows > 0)
            return true;

        else
            return false;
    }catch(Exception e)
    {
        e.printStackTrace();
        return false;
    }
}
```

6. Manual de Utilizare

6.1 Înregistrare și Autentificare

- **Autentificare (Logare):**

1. La deschiderea aplicației, utilizatorul este întâmpinat de fereastra principală „Platforma de Studii”. Aici sunt disponibile două opțiuni: **Înregistrare** și **Logare**.
2. Pentru autentificare, utilizatorul apasă pe butonul **Logare**. Acesta va introduce numele de utilizator și parola asociată contului său.
3. În cazul în care datele introduse sunt valide, utilizatorul este direcționat către **Pagina Principală**, unde are acces la funcționalitățile aferente rolului său (student, profesor sau administrator).

- **Creare Cont:**

1. Dacă utilizatorul nu are un cont, poate accesa opțiunea **Înregistrare**. Aceasta deschide o fereastră dedicată pentru introducerea datelor necesare creării contului.

2. În fereastra de **Creare Cont**, utilizatorul completează câmpurile obligatorii, precum: numele de utilizator, parola, numele complet, CNP, adresă, număr de telefon, email și alte informații necesare.
3. După completarea câmpurilor, utilizatorul apasă butonul **Creează Cont**, iar sistemul salvează informațiile în baza de date. În caz de succes, utilizatorul va fi notificat și poate reveni la pagina principală pentru logare.

6.2 Gestionarea Informațiilor per utilizatori

- **Student:**

1. Vizualizează informațiile personale din secțiunea **Informații Cont**.
2. Accesează secțiunea **Discipline** pentru a vedea lista disciplinelor la care este înscris.
3. Poate verifica activitățile asociate disciplinelor din secțiunea **Afisare Activități**.
4. Accesează secțiunea **Afisare Note** pentru a vizualiza notele primite la diferitele activități.
5. Poate adera la un **Grup** de studiu existent sau crea un grup nou pentru o anumită disciplină.

- **Profesor:**

1. Poate consulta și edita informațiile personale în **Informații Cont**.
2. Adaugă activități academice (cursuri, seminarii, laboratoare) în secțiunea **Afisare Activități**.
3. Inserează note pentru studenți în catalog, accesând secțiunea **Catalog**.
4. Poate crea grupuri de studiu pentru disciplinele predate folosind funcția **Creare Grup**.
5. Accesează lista de studenți și situația acestora pentru fiecare disciplină din secțiunea **Discipline**.

- **Administrator:**

1. Modifică informațiile utilizatorilor din secțiunea **Informații Cont**.
2. Adaugă sau șterge utilizatori din platformă, utilizând funcția **Creare Cont**.
3. Accesează catalogul general pentru toate disciplinele și utilizatorii în secțiunea **Catalog**.
4. Gestionează grupurile de studiu și activitățile pentru a asigura o organizare optimă.

Fiecare funcționalitate din **Pagina Principală** este accesibilă prin butoane dedicate, iar interfața este intuitivă, având scopul de a simplifica utilizarea platformei.

7. Concluzii și dezvoltari ulterioare

7.1 Analiza funcționalităților curente- cele mai interesante solutii adoptate

Platforma dezvoltată oferă o serie de funcționalități cheie care acoperă nevoile diferitelor categorii de utilizatori. Printre cele mai interesante soluții adoptate se numără:

- **Sistemul de gestionare a utilizatorilor pe roluri**: Implementarea separată a funcționalităților pentru studenți, profesori, administratori și super administratori a permis o organizare clară și eficientă. Acest lucru asigură accesul diferențiat și adaptat nevoilor fiecărui tip de utilizator.
- **Gestionarea grupurilor de studiu**: Funcționalitatea de creare și administrare a grupurilor de studiu pentru studenți facilitează colaborarea și schimbul de informații.
- **Inserarea și afișarea notelor**: Sistemul permite o gestionare simplă și transparentă a notelor studenților printr-un catalog centralizat.
- **Interfața intuitivă**: Designul ușor de utilizat și structura logică a paginilor, cum ar fi secțiunile de logare, creare cont și gestionare a informațiilor, contribuie la o experiență plăcută pentru utilizatori.

Aceste soluții asigură o bună funcționalitate a platformei, adresând cerințele principale ale utilizatorilor și oferind un punct de pornire solid pentru extinderi viitoare.

7.2 Dezvoltări Ulterioare

Pentru îmbunătățirea și extinderea platformei, sunt propuse următoarele dezvoltări ulterioare:

- **Funcționalități avansate de analiză**: Integrarea unor rapoarte avansate pentru profesori și administratori, precum performanța studenților la nivel de disciplină sau activitate.
- **Integrare cu calendare externe**: Permisiunea de a sincroniza activitățile academice cu platforme precum Google Calendar pentru o mai bună organizare.
- **Sistem de notificări**: Implementarea unui sistem de notificări automate pentru studenți privind activitățile viitoare, notele publicate sau termenele limită.
- **Acces mobil**: Dezvoltarea unei aplicații mobile dedicate care să faciliteze accesul rapid la informații, indiferent de dispozitiv.
- **Autentificare în doi pași (2FA)**: Creșterea securității prin integrarea unui sistem de autentificare cu doi factori.
- **Integrare cu alte platforme educaționale**: Extinderea funcționalităților prin conectarea la alte platforme, precum sisteme de evaluare sau biblioteci digitale.
- **Chat și colaborare în timp real**: Adăugarea unui modul de comunicare directă între membri grupurilor de studiu.
- **Feedback și sondaje**: Introducerea unor mecanisme prin care utilizatorii pot oferi feedback privind activitățile, disciplinele și funcționalitățile platformei.

Aceste dezvoltări ar contribui la creșterea funcționalității și a utilității platformei, menținând-o actuală și atractivă pentru utilizatori.