

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

Departamentul Calculatoare

**SISTEM INFORMATIC DESTINAT GESTIUNII**

**UNEI PLATFORME DE STUDIU**

Proiect la disciplina

Introducere în baze de date

*Profesor coordonator: Echipa:*

*Ivan Cosmina Dănciulescu Bianca-Andreea*

*Iepure Denisa-Alexandra*

*Grupa: 30223 Pavel Mădălina-Denisa*

***Cuprins:***

1. Introducere, argumente, scop si obiective specifice.......................................................2
2. Suport teoretic................................................................................................................3
3. Analiza cerințelor………………………………………………………………...........5
4. Modelul de date si descrierea acestuia……………………………..………………….6
5. Detalii de implementare...............................................................................................23
6. Concluzii.Dezvoltări ulterioare....................................................................................25
7. **Introducere**

Platformele de studiu au rolul de a face mai accesibilă interacțiunea dintre cadrele didactice și studenți, facilitând de asemenea procesul de învățare. Din acest considerent, universitățile de prestigiu au îmbrățișat cu interes această perspectivă de comunicare între entitățile unui mediu academic.

Proiectul nostru vizează gestionarea unei astfel de platforme, prin intermediul căreia se realizează operațiile de bază necesare funcționalității corecte și eficiente a unei instituții de învățământ superior.

Prin intermediul unei interfețe grafice va fi posibilă manipularea datelor și realizarea unor proceduri asociate celor 4 tipuri de utilizatori: super-administrator, administrator, profesor și student.

Super-administratorul și administratorul pot opera asupra celorlalți utilizatori, fiind îndreptățiți să își vizualizeze datele personale, să adauge, elimine, actualizeze informații cu privire la studenți și profesori, ș. a. Profesorii au dreptul să programeze activități, să adauge procente și note într-un catalog virtual, astfel ca studenții să aibă acces la vizualizarea situației lor școlare. O funcționalitate suplimentară a acestei platforme este aderarea studenților la grupuri de studiu, în vederea îmbunătățirii calității învățării.

1. **Suport teoretic**

Pentru proiectarea bazei de date relațională am folosit ***MySQL Workbench 8.0 CE***.

**SQL (Structured Query Language)** este limbajul ce lucrează cu bazele de date relaționale. Instrucțiunile SQL pot fi împărțite în 3 categorii de operațiuni: de manipulare a datelor, de schimbare a structurii datelor și de administrare a privilegiilor.

Apărut în 1995, MySQL este un sistem de management al bazelor de date relaționale care stochează și prezintă datele sub formă de tabel, organizate în rânduri și coloane. Este compatibil cu majoritatea sistemelor de operare, inclusiv Windows, Linux, NetWare, Novell, Solaris și alte variante ale UNIX. Acesta are o arhitectură unică a motorului de stocare care îl face mai rapid, mai ieftin și mai fiabil, iar faptul că acceptă un număr mare de aplicații încorporate îl face foarte flexibil. Alt avantaj al acestui sistem este capabilitatea de gestionare a mai mult de 50 de milioane de rânduri. Acest lucru este suficient pentru a gestiona aproape orice cantitate de date. Deși limita implicită de dimensiune a fișierului este de 4 GB, dar poate fi mărită la 8 TB.

Totodată, pentru realizarea interfeței grafice am folosit ***Eclipse***, implementând codul în limbajul de programare ***Java.***

Java este un limbaj de programare orientat pe obiecte utilizat pe scară largă și o platformă software care rulează pe miliarde de dispozitive, inclusiv notebook-uri, dispozitive mobile, console de jocuri, dispozitive medicale și multe altele. Regulile și sintaxa Java se bazează pe limbajele C și C++. Un avantaj major al dezvoltării de software cu Java este portabilitatea acestuia. Când limbajul a fost inventat în 1991 de James Gosling de la Sun Microsystems (dobândit mai târziu de Oracle), scopul principal a fost acela de a putea „scrie o dată, rula oriunde”.

Platforma software Java constă din JVM, Java API și un mediu de dezvoltare complet. JVM-ul analizează și rulează (interpretează) bytecode Java. API-ul Java constă dintr-un set extins de biblioteci, inclusiv obiecte de bază, funcții de rețea și securitate; generare Extensible Markup Language (XML); și servicii web. Luate împreună, limbajul Java și platforma software Java creează o tehnologie puternică și dovedită pentru dezvoltarea de software pentru întreprinderi.

Pentru a realiza legătura între baza de date și interfața grafică am folosit ***Java Database Connectivity(JDBC)***.

JDBC este specificația JavaSoft a unei interfețe standard de programare a aplicațiilor (API) care permite programelor Java să acceseze sistemele de gestionare a bazelor de date. API-ul JDBC constă dintr-un set de interfețe și clase scrise în limbajul de programare Java.

Folosind aceste interfețe și clase standard, programatorii pot scrie aplicații care se conectează la baze de date, pot trimite interogări scrise în limbaj de interogare structurat (SQL) și pot procesa rezultatele.

Deoarece JDBC este o specificație standard, un program Java care utilizează API-ul JDBC se poate conecta la orice sistem de gestionare a bazelor de date (DBMS), atâta timp cât există un driver pentru acel SGBD anume.

1. Analiza cerințelor

Aplicația gestionează acțiunile celor 4 tipuri de utilizatori, baza de date stocând următoarele informații:

* Datele personale ale fiecărui utilizator(CNP, nume, prenume, adresă, număr de telefon, email, IBAN ș.a.);
* ID-uri unice pentru fiecare tabelă;
* Materiile la care profesorii pot fi asignați și studenții înscriși;
* Activitățiile din cadrul unei materii(curs, laborator sau seminar);
* Datele examinărilor fiecărei activități;
* Orarul studenților și al profesorilor;
* Grupurile de studiu, participanții și intervalele orare pentru fiecare activitate;

Totodată, operațiile care trebuie să se regăsească în baza de date sunt:

1. Cele destinate administratorului și super-administratorului

* Adăugare/Update/Ștergere administrator
* Adăugare/Update/Ștergere profesor
* Adăugare/Update/Ștergere student
* Căutarea tipului de utilizator conform numelui
* Căutarea unui curs
* Asignarea profesorilor la cursuri

1. Cele destinate profesorilor:

* Adăugarea procentelor
* Programarea activităților
* Vizualizarea activităților din ziua curentă
* Adăugarea notelor

1. Cele destinate studenților:

* Înscrierea/Renunțarea la cursuri
* Înscrierea la un grup de studiu
* Adăugare/Înscriere meet
* Vizualizare membrii

1. Modelul de date și descrierea acestuia

Platforma noastră de studiu este formată din 19 tabele, care au drept cheie primară id-ul fiecărei tabele în parte, și au legătură între ele prin intermediul cheilor străine.

1. Tabela ***SuperAdministrator*** stochează datele personale ale acestora:

CREATE TABLE `superadministrator` (

`id\_super\_administrator` int NOT NULL AUTO\_INCREMENT,

`CNP` varchar(13) NOT NULL,

`nume` varchar(45) DEFAULT NULL,

`prenume` varchar(45) DEFAULT NULL,

`adresa` varchar(45) DEFAULT NULL,

`nr\_telefon` varchar(10) DEFAULT NULL,

`email` varchar(45) DEFAULT NULL,

`IBAN` varchar(16) DEFAULT NULL,

`nr\_contract` int DEFAULT NULL,

`data\_inceput` date DEFAULT NULL,

`data\_final` date DEFAULT NULL,

PRIMARY KEY (`id\_super\_administrator`)

1. Tabela Administrator este asemănătoare celei anteioare:

CREATE TABLE `administrator` (

`id\_administrator` int NOT NULL AUTO\_INCREMENT,

`CNP` varchar(13) NOT NULL,

`nume` varchar(45) DEFAULT NULL,

`prenume` varchar(45) DEFAULT NULL,

`adresa` varchar(45) DEFAULT NULL,

`nr\_telefon` varchar(10) DEFAULT NULL,

`email` varchar(45) DEFAULT NULL,

`IBAN` varchar(16) DEFAULT NULL,

`nr\_contract` int DEFAULT NULL,

PRIMARY KEY (`id\_administrator`));

1. Tabela ***profesor:***

CREATE TABLE `profesor` (

`id\_profesor` int NOT NULL AUTO\_INCREMENT,

`CNP` varchar(13) NOT NULL,

`nume` varchar(45) DEFAULT NULL,

`prenume` varchar(45) DEFAULT NULL,

`adresa` varchar(45) DEFAULT NULL,

`nr\_telefon` varchar(10) DEFAULT NULL,

`email` varchar(45) DEFAULT NULL,

`IBAN` varchar(16) DEFAULT NULL,

`nr\_contract` int DEFAULT NULL,

PRIMARY KEY (`id\_profesor`));

1. În plus, pentru a reține alte detalii precum materia predată, numărul

minim și maxim de ore, dar și capacitatea maximă de studenți am implementat tabela ***informații profesor:***

CREATE TABLE `informatii\_profesor` (

`id\_info` int NOT NULL AUTO\_INCREMENT,

`activitate\_predata` varchar(45) DEFAULT NULL,

`min\_ore\_predare` int DEFAULT NULL,

`max\_ore\_predare` int DEFAULT NULL,

`departament` varchar(45) DEFAULT NULL,

`tip` varchar(40) DEFAULT NULL,

`nr\_max\_stud` int DEFAULT NULL,

`id\_profesor` int DEFAULT NULL,

PRIMARY KEY (`id\_info`),

KEY `id\_profesor` (`id\_profesor`),

CONSTRAINT `informatii\_profesor\_ibfk\_1` FOREIGN KEY (`id\_profesor`) REFERENCES `profesor` (`id\_profesor`) ON DELETE CASCADE);

1. Tabela ***student*** reține caracteristicile indiviziilor, dar și anul de studiu și

id-ul grupului de studiu la care s-ar putea înscrie:

CREATE TABLE `student` (

`id\_student` int NOT NULL AUTO\_INCREMENT,

`CNP` varchar(13) NOT NULL,

`nume` varchar(45) DEFAULT NULL,

`prenume` varchar(45) DEFAULT NULL,

`adresa` varchar(45) DEFAULT NULL,

`nr\_telefon` varchar(10) DEFAULT NULL,

`email` varchar(45) DEFAULT NULL,

`IBAN` varchar(16) DEFAULT NULL,

`nr\_contract` varchar(10) DEFAULT NULL,

`an\_studiu` varchar(10) DEFAULT NULL,

`nr\_ore` varchar(10) DEFAULT NULL,

`id\_grup` int DEFAULT NULL,

PRIMARY KEY (`id\_student`),

KEY `id\_grup` (`id\_grup`),

CONSTRAINT `student\_ibfk\_1` FOREIGN KEY (`id\_grup`) REFERENCES `grup\_studiu` (`id\_grup`) ON DELETE CASCADE);

1. ***Contractul de studiu*** are rolul de a indica data înscrierii și a posibilei

renunțări la facultate a fiecărui student:

CREATE TABLE `contract\_studiu` (

`id\_contract` int NOT NULL AUTO\_INCREMENT,

`data\_inscrierii` date DEFAULT NULL,

`data\_retragerii` date DEFAULT NULL,

`id\_student` int DEFAULT NULL,

PRIMARY KEY (`id\_contract`),

KEY `id\_student` (`id\_student`),

CONSTRAINT `contract\_studiu\_ibfk\_1` FOREIGN KEY (`id\_student`) REFERENCES `student` (`id\_student`) ON DELETE CASCADE);

1. Tabela ***înscriere\_student*** conține id-ul studentului, materia la care este

înscris, tipul activității și perioada de desfășurare a acesteia:

CREATE TABLE `inscriere\_student` (

`id\_inscriere\_s` int NOT NULL AUTO\_INCREMENT,

`denumire\_materie` varchar(45) DEFAULT NULL,

`tip` varchar(45) DEFAULT NULL,

`zi` varchar(45) DEFAULT NULL,

`ora` varchar(45) DEFAULT NULL,

`id\_student` int DEFAULT NULL,

PRIMARY KEY (`id\_inscriere\_s`),

KEY `id\_student` (`id\_student`),

CONSTRAINT `inscriere\_student\_ibfk\_1` FOREIGN KEY (`id\_student`) REFERENCES `student` (`id\_student`) ON DELETE CASCADE);

1. Pentru a realiza legătura între profesor, materia predată și studentul

înscris la aceasta am creat tabela ***înscriere***:

CREATE TABLE `inscriere` (

`id\_inscriere` int NOT NULL AUTO\_INCREMENT,

`id\_student` int DEFAULT NULL,

`id\_materie` int DEFAULT NULL,

`id\_profesor` int DEFAULT NULL,

`data\_inscrierii` date DEFAULT NULL,

PRIMARY KEY (`id\_inscriere`),

KEY `id\_student` (`id\_student`),

KEY `id\_profesor` (`id\_profesor`),

KEY `id\_materie` (`id\_materie`),

CONSTRAINT `inscriere\_ibfk\_1` FOREIGN KEY (`id\_student`) REFERENCES `student` (`id\_student`) ON DELETE CASCADE,

CONSTRAINT `inscriere\_ibfk\_2` FOREIGN KEY (`id\_profesor`) REFERENCES `profesor` (`id\_profesor`) ON DELETE CASCADE,

CONSTRAINT `inscriere\_ibfk\_3` FOREIGN KEY (`id\_materie`) REFERENCES `materie` (`id\_materie`) ON DELETE CASCADE));

1. Tabela ***materie*** reține doar id-ul și denumirea acesteia:

CREATE TABLE `materie` (

`id\_materie` int NOT NULL AUTO\_INCREMENT,

`descriere` varchar(45) DEFAULT NULL,

PRIMARY KEY (`id\_materie`));

1. , 11), 12) Tipurile activităților sunt: ***curs, laborator și seminar,*** iar

acestea conțin notele obținute la fiecare probă:

CREATE TABLE `curs` (

`id\_curs` int NOT NULL AUTO\_INCREMENT,

`nota\_partial` double DEFAULT NULL,

`nota\_examen` double DEFAULT NULL,

`nota\_restanta` double DEFAULT NULL,

`id\_materie` int DEFAULT NULL,

`id\_info` int DEFAULT NULL,

`id\_student` int DEFAULT NULL,

PRIMARY KEY (`id\_curs`),

KEY `id\_student` (`id\_student`),

KEY `id\_materie` (`id\_materie`),

KEY `id\_info` (`id\_info`),

CONSTRAINT `curs\_ibfk\_1` FOREIGN KEY (`id\_student`) REFERENCES `student` (`id\_student`) ON DELETE CASCADE,

CONSTRAINT `curs\_ibfk\_2` FOREIGN KEY (`id\_materie`) REFERENCES `materie` (`id\_materie`) ON DELETE CASCADE,

CONSTRAINT `curs\_ibfk\_3` FOREIGN KEY (`id\_info`) REFERENCES `informatii\_profesor` (`id\_info`) ON DELETE CASCADE));

CREATE TABLE `laborator` (

`id\_lab` int NOT NULL AUTO\_INCREMENT,

`nota\_lab` double DEFAULT NULL,

`nota\_restanta` double DEFAULT NULL,

`id\_materie` int DEFAULT NULL,

`id\_info` int DEFAULT NULL,

`id\_student` int DEFAULT NULL,

PRIMARY KEY (`id\_lab`),

KEY `id\_student` (`id\_student`),

KEY `id\_materie` (`id\_materie`),

KEY `id\_info` (`id\_info`),

CONSTRAINT `laborator\_ibfk\_1` FOREIGN KEY (`id\_student`) REFERENCES `student` (`id\_student`) ON DELETE CASCADE,

CONSTRAINT `laborator\_ibfk\_2` FOREIGN KEY (`id\_materie`) REFERENCES `materie` (`id\_materie`) ON DELETE CASCADE,

CONSTRAINT `laborator\_ibfk\_3` FOREIGN KEY (`id\_info`) REFERENCES `informatii\_profesor` (`id\_info`) ON DELETE CASCADE);

CREATE TABLE `seminar` (

`id\_seminar` int NOT NULL AUTO\_INCREMENT,

`nota\_seminar` double DEFAULT NULL,

`id\_materie` int DEFAULT NULL,

`id\_info` int DEFAULT NULL,

`id\_student` int DEFAULT NULL,

PRIMARY KEY (`id\_seminar`),

KEY `id\_student` (`id\_student`),

KEY `id\_materie` (`id\_materie`),

KEY `id\_info` (`id\_info`),

CONSTRAINT `seminar\_ibfk\_1` FOREIGN KEY (`id\_student`) REFERENCES `student` (`id\_student`) ON DELETE CASCADE,

CONSTRAINT `seminar\_ibfk\_2` FOREIGN KEY (`id\_materie`) REFERENCES `materie` (`id\_materie`) ON DELETE CASCADE,

CONSTRAINT `seminar\_ibfk\_3` FOREIGN KEY (`id\_info`) REFERENCES `informatii\_profesor` (`id\_info`) ON DELETE CASCADE);

13) Tabela ***catalog*** înglobează nota și situația finală a elevilor:

CREATE TABLE `catalog` (

`id\_catalog` int NOT NULL AUTO\_INCREMENT,

`nota\_finala` int DEFAULT NULL,

`situatie\_finala` varchar(30) DEFAULT NULL,

`id\_student` int DEFAULT NULL,

`id\_materie` int DEFAULT NULL,

`id\_info` int DEFAULT NULL,

PRIMARY KEY (`id\_catalog`),

KEY `id\_student` (`id\_student`),

KEY `id\_materie` (`id\_materie`),

KEY `id\_info` (`id\_info`),

CONSTRAINT `catalog\_ibfk\_1` FOREIGN KEY (`id\_student`) REFERENCES `student` (`id\_student`) ON DELETE CASCADE,

CONSTRAINT `catalog\_ibfk\_2` FOREIGN KEY (`id\_materie`) REFERENCES `materie` (`id\_materie`) ON DELETE CASCADE,

CONSTRAINT `catalog\_ibfk\_3` FOREIGN KEY (`id\_info`) REFERENCES `informatii\_profesor` (`id\_info`) ON DELETE CASCADE);

14) ***Calendarul*** cuprinde datele examinărilor pentru fiecare tip de activitate:

CREATE TABLE `calendar` (

`id\_calendar` int NOT NULL AUTO\_INCREMENT,

`data\_partial` date DEFAULT NULL,

`data\_colocviu` date DEFAULT NULL,

`data\_examen` date DEFAULT NULL,

`data\_restanta\_lab` date DEFAULT NULL,

`data\_restanta\_curs` date DEFAULT NULL,

`id\_materie` int DEFAULT NULL,

`id\_info` int DEFAULT NULL,

PRIMARY KEY (`id\_calendar`),

KEY `id\_materie` (`id\_materie`),

KEY `id\_info` (`id\_info`),

CONSTRAINT `calendar\_ibfk\_1` FOREIGN KEY (`id\_materie`) REFERENCES `materie` (`id\_materie`) ON DELETE CASCADE,

CONSTRAINT `calendar\_ibfk\_2` FOREIGN KEY (`id\_info`) REFERENCES `informatii\_profesor` (`id\_info`) ON DELETE CASCADE);

15) O altă tabelă de legătură este tabela ***Activități***, care conține procentele cursului, laboratorului și seminarului, ziua și ora și numărul activităților:

CREATE TABLE `activitati` (

`id\_activitati` int NOT NULL AUTO\_INCREMENT,

`activitate\_predata` varchar(45) DEFAULT NULL,

`zi\_lab` varchar(45) DEFAULT NULL,

`zi\_curs` varchar(45) DEFAULT NULL,

`zi\_seminar` varchar(45) DEFAULT NULL,

`ora\_lab` varchar(45) DEFAULT NULL,

`ora\_curs` varchar(45) DEFAULT NULL,

`ora\_seminar` varchar(45) DEFAULT NULL,

`procent\_lab` varchar(45) DEFAULT NULL,

`procent\_curs` varchar(45) DEFAULT NULL,

`procent\_seminar` varchar(45) DEFAULT NULL,

`nr\_ore\_lab` int DEFAULT NULL,

`nr\_ore\_curs` int DEFAULT NULL,

`nr\_ore\_seminar` int DEFAULT NULL,

`id\_info` int DEFAULT NULL,

PRIMARY KEY (`id\_activitati`),

KEY `id\_info` (`id\_info`),

CONSTRAINT `activitati\_ibfk\_1` FOREIGN KEY (`id\_info`) REFERENCES `informatii\_profesor` (`id\_info`) ON DELETE CASCADE);

16) Studenții au posibilitatea de a se înscrie la ***grupuri de studiu*** pentru a-și îmbunătăți situația școlară:

CREATE TABLE `grup\_studiu` (

`id\_grup` int NOT NULL AUTO\_INCREMENT,

`id\_materie` int DEFAULT NULL,

`nr\_participanti` int DEFAULT NULL,

`nr\_max\_participanti` int DEFAULT NULL,

PRIMARY KEY (`id\_grup`),

KEY `id\_materie` (`id\_materie`),

CONSTRAINT `grup\_studiu\_ibfk\_1` FOREIGN KEY (`id\_materie`) REFERENCES `materie` (`id\_materie`) ON DELETE CASCADE);

17) Dar, pentru a fi incluși în grupuri trebuie să se înscrie, de aceea am creat ***tabela inscriere\_meet\_grup:***

CREATE TABLE `inscriere\_meet\_grup` (

`id\_inscriere\_meet` int NOT NULL AUTO\_INCREMENT,

`data\_meet` date DEFAULT NULL,

`ora\_meet` varchar(45) DEFAULT NULL,

`timp\_expirare` int DEFAULT NULL,

`scriere` varchar(45) DEFAULT NULL,

`ora\_creare` time DEFAULT NULL,

`id\_meet` int DEFAULT NULL,

`id\_student` int DEFAULT NULL,

PRIMARY KEY (`id\_inscriere\_meet`),

KEY `id\_meet` (`id\_meet`),

KEY `id\_student` (`id\_student`),

CONSTRAINT `inscriere\_meet\_grup\_ibfk\_1` FOREIGN KEY (`id\_meet`) REFERENCES `meet\_grup` (`id\_meet`) ON DELETE CASCADE,

CONSTRAINT `inscriere\_meet\_grup\_ibfk\_2` FOREIGN KEY (`id\_student`) REFERENCES `student` (`id\_student`) ON DELETE CASCADE);

18) Tabela ***meet\_grup*** este implementată cu scopul de a permite studenților să programeze întâlniri:

CREATE TABLE `inscriere\_meet\_grup` (

`id\_inscriere\_meet` int NOT NULL AUTO\_INCREMENT,

`data\_meet` date DEFAULT NULL,

`ora\_meet` varchar(45) DEFAULT NULL,

`timp\_expirare` int DEFAULT NULL,

`scriere` varchar(45) DEFAULT NULL,

`ora\_creare` time DEFAULT NULL,

`id\_meet` int DEFAULT NULL,

`id\_student` int DEFAULT NULL,

PRIMARY KEY (`id\_inscriere\_meet`),

KEY `id\_meet` (`id\_meet`),

KEY `id\_student` (`id\_student`),

CONSTRAINT `inscriere\_meet\_grup\_ibfk\_1` FOREIGN KEY (`id\_meet`) REFERENCES `meet\_grup` (`id\_meet`) ON DELETE CASCADE,

CONSTRAINT `inscriere\_meet\_grup\_ibfk\_2` FOREIGN KEY (`id\_student`) REFERENCES `student` (`id\_student`) ON DELETE CASCADE);

19) Pentru a permite și profesorilor să participe la activitățile din cadrul grupurilor de studiu, am creat tabela ***activitate\_suplimentară***:

CREATE TABLE `activitate\_suplimentara` (

`id\_activ\_sup` int NOT NULL AUTO\_INCREMENT,

`id\_student` int DEFAULT NULL,

`id\_grup` int DEFAULT NULL,

`id\_profesor` int DEFAULT NULL,

PRIMARY KEY (`id\_activ\_sup`),

KEY `id\_student` (`id\_student`),

KEY `id\_grup` (`id\_grup`),

KEY `id\_profesor` (`id\_profesor`),

CONSTRAINT `activitate\_suplimentara\_ibfk\_1` FOREIGN KEY (`id\_student`) REFERENCES `student` (`id\_student`) ON DELETE CASCADE,

CONSTRAINT `activitate\_suplimentara\_ibfk\_2` FOREIGN KEY (`id\_grup`) REFERENCES `grup\_studiu` (`id\_grup`) ON DELETE CASCADE,

CONSTRAINT `activitate\_suplimentara\_ibfk\_3` FOREIGN KEY (`id\_profesor`) REFERENCES `profesor` (`id\_profesor`) ON DELETE CASCADE);

Funcțiile fiecărui utilizator din cadrul proiectului sunt implementate folosind MySQL, sub forma unor proceduri care poartă nume sugestive. Acestea sunt ulterior apelate în interfața grafică.

* O parte din procedurile implementate:

**Identificarea id-ul-ui unui profesor care predă o anumită materie:**

CREATE DEFINER=`root`@`localhost` PROCEDURE `activitati\_profesor`(in nume\_prof varchar(45), in prenume\_prof varchar(45))

begin

set @IDprof = (Select profesor.id\_profesor from profesor

where profesor.nume=nume\_prof and profesor.prenume=prenume\_prof);

select informatii\_profesor.tip, informatii\_profesor.activitate\_predata

from informatii\_profesor where informatii\_profesor.id\_profesor=@IDprof;

end ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `activitati\_student10`( in nume\_stud varchar(45), in prenume\_stud varchar(45), in CNP varchar(45),in nume\_curs varchar(45),in tip varchar(45),in oraa varchar(45),in zii varchar(45))

begin

set @Idmaterie=(SELECT materie.id\_materie from materie where materie.descriere=nume\_curs);

SET @Idprof=(Select informatii\_profesor.id\_profesor FROM platforma\_studiu.informatii\_profesor where informatii\_profesor.activitate\_predata=nume\_curs and informatii\_profesor.tip=tip group by id\_profesor order by COUNT(\*) limit 1);

set @Idstudent=(SELECT student.id\_student from student where student.nume=nume\_stud and student.prenume=prenume\_stud and student.CNP=CNP);

set @Idinfolab=(select informatii\_profesor.id\_info from informatii\_profesor where informatii\_profesor.id\_profesor=@Idprof and informatii\_profesor.activitate\_predata=nume\_curs and informatii\_profesor.tip='laborator');

set @nr\_max\_stud\_lab=(SELECT informatii\_profesor.nr\_max\_stud from informatii\_profesor where informatii\_profesor.id\_info=@Idinfolab);

set @Idinfocurs=(select informatii\_profesor.id\_info from informatii\_profesor where informatii\_profesor.id\_profesor=@Idprof and informatii\_profesor.activitate\_predata=nume\_curs and informatii\_profesor.tip='curs');

set @nr\_max\_stud\_curs=(SELECT informatii\_profesor.nr\_max\_stud from informatii\_profesor where informatii\_profesor.id\_info=@Idinfocurs);

set @Idinfosem=(select informatii\_profesor.id\_info from informatii\_profesor where informatii\_profesor.id\_profesor=@Idprof and informatii\_profesor.activitate\_predata=nume\_curs and informatii\_profesor.tip='seminar');

set @nr\_max\_stud\_sem=(SELECT informatii\_profesor.nr\_max\_stud from informatii\_profesor where informatii\_profesor.id\_info=@Idinfosem);

set @idstudocupat=(select inscriere\_student.id\_inscriere\_s from inscriere\_student where inscriere\_student.ora=oraa and inscriere\_student.zi=zii and inscriere\_student.id\_student=@Idstudent);

-- select @idstudocupat;

set @IDlab=( SELECT MAX(laborator.id\_lab) FROM laborator) + 1;

if @IDlab IS NULL then

set @IDlab=1;

end if;

set @IDcurs=( SELECT MAX(curs.id\_curs) FROM curs) + 1;

if @IDcurs IS NULL then

set @IDcurs=1;

end if;

set @IDsem=( SELECT MAX(seminar.id\_seminar) FROM seminar) + 1;

if @IDsem IS NULL then

set @IDsem=1;

end if;

set @IDin=( SELECT MAX(inscriere.id\_inscriere) FROM inscriere) + 1;

if @IDin IS NULL then

set @IDin=1;

end if;

if tip='laborator' then

if (SELECT COUNT(\*) from laborator where laborator.id\_info=@Idinfolab )< @nr\_max\_stud\_lab and (@idstudocupat is NULL) then

insert into laborator values(@IDlab,NULL,NULL,@Idmaterie,@Idinfolab,@Idstudent);

end if;

end if;

if tip='curs' then

IF (SELECT COUNT(\*) from curs where curs.id\_info=@Idinfocurs)< @nr\_max\_stud\_curs and (@idstudocupat is NULL) then

select @idstudocupat;

insert into curs values(@IDcurs,NULL,NULL,NULL,@Idmaterie,@Idinfocurs,@Idstudent);

insert into inscriere values(@IDin,@Idstudent,@Idmaterie,@Idprof, current\_date());

insert into catalog values(@IDcatalog,NULL,NULL,@Idstudent,@Idmaterie,@Idinfocurs);

end if;

end if;

if tip='seminar' then

IF (SELECT COUNT(\*) from seminar where seminar.id\_info=@Idinfosem)< @nr\_max\_stud\_sem and (@idstudocupat is NULL) then

insert into seminar values(@IDsem,NULL,@Idmaterie,@Idinfosem,@Idstudent);

end if;

end if;

end ;;

**Setarea procentelor de către profesorul de curs pentru materiile pe care le predă:**

CREATE DEFINER=`root`@`localhost` PROCEDURE `adaugaprocente13`(in id\_prof INT,in denumire\_materie varchar(45),in procentcurs INT, in procentseminar INT,in procentlab INT)

begin

SET @IDmaterie=(SELECT materie.id\_materie

from materie,profesor

where materie.descriere=denumire\_materie

and profesor.id\_profesor=id\_prof

);

update activitati set activitati.procent\_lab=procentlab where activitati.activitate\_predata=denumire\_materie;

update activitati set activitati.procent\_seminar=procentseminar where activitati.activitate\_predata=denumire\_materie;

update activitati set activitati.procent\_curs=procentcurs where activitati.activitate\_predata=denumire\_materie;

end ;;

**Procedura de introducere a notelor în catalog de către profesori este următoarea:**

CREATE DEFINER=`root`@`localhost` PROCEDURE `adauga\_note2`( in nume\_curs varchar(45), in nume\_prof varchar(45), in prenume\_prof varchar(45),in nume\_stud varchar(45),in prenume\_stud varchar(45),in CNP varchar(45),

in nota\_lab double,in nota\_res\_lab double, nota\_seminar double,nota\_partial double,nota\_examen double, nota\_res\_ex double)

begin

set @Idprof=(select profesor.id\_profesor from profesor where profesor.nume=nume\_prof and profesor.prenume=prenume\_prof);

set @Idmaterie=(select materie.id\_materie from materie where materie.descriere=nume\_curs);

set @Idstud=(select student.id\_student from student where student.nume=nume\_stud and student.prenume=prenume\_stud and student.CNP=CNP);

set @Idinfolab=(select informatii\_profesor.id\_info from informatii\_profesor where informatii\_profesor.id\_profesor=@Idprof and informatii\_profesor.activitate\_predata=nume\_curs and informatii\_profesor.tip='laborator');

set @Idinfocurs=(select informatii\_profesor.id\_info from informatii\_profesor where informatii\_profesor.id\_profesor=@Idprof and informatii\_profesor.activitate\_predata=nume\_curs and informatii\_profesor.tip='curs');

set @Idinfosem=(select informatii\_profesor.id\_info from informatii\_profesor where informatii\_profesor.id\_profesor=@Idprof and informatii\_profesor.activitate\_predata=nume\_curs and informatii\_profesor.tip='seminar');

if nota\_lab IS NOT NULL then update laborator set laborator.nota\_lab=nota\_lab where laborator.id\_info=@Idinfolab;

if nota\_res\_lab IS NOT NULL then update laborator set laborator.nota\_restanta=nota\_res\_lab where laborator.id\_info=@Idinfolab;

end if;

end if;

if nota\_seminar IS NOT NULL then update seminar set seminar.nota\_seminar=nota\_seminar where seminar.id\_info=@Idinfosem;

end if;

if nota\_partial IS NOT NULL then update curs set curs.nota\_partial=nota\_partial where curs.id\_info=@Idinfocurs;

if nota\_examen IS NOT NULL then update curs set curs.nota\_examen=nota\_examen where curs.id\_info=@Idinfocurs;

if nota\_res\_ex IS NOT NULL then update curs set curs.nota\_restanta=nota\_res\_ex where curs.id\_info=@Idinfocurs;

end if;

end if;

end if;

end ;;

**Procedura apelată pentru afișarea studenților:**

CREATE DEFINER=`root`@`localhost` PROCEDURE `afiseaza\_studenti1`( in descriere varchar(45), in nume\_prof varchar(45), in prenume\_prof varchar(45))

begin

set @Idprof=(select profesor.id\_profesor from profesor where profesor.nume=nume\_prof and profesor.prenume=prenume\_prof);

set @Idmaterie=(select materie.id\_materie from materie where materie.descriere=descriere);

select student.nume, student.prenume, student.CNP

from student, inscriere

where student.id\_student=inscriere.id\_student

and

inscriere.id\_materie=@Idmaterie and inscriere.id\_profesor=@Idprof;

end ;;

**Procedura de căutare a cursului pe care o pot realiza doar administratorii și superadministratorul este:**

CREATE DEFINER=`root`@`localhost` PROCEDURE `cauta\_curs15`( in descriere varchar(45))

begin

select profesor.nume, profesor.prenume from profesor, inscriere, materie where profesor.id\_profesor=inscriere.id\_profesor and

materie.id\_materie=inscriere.id\_materie and materie.descriere=descriere;

end ;;

**Pentru studenții care se înscriu la grupurile de studiu, următoarea procedură le acordă dreptul de a organiza o întâlnire:**

CREATE DEFINER=`root`@`localhost` PROCEDURE `creare\_meet`( in data\_meet DATE, in ora\_meet varchar(45), in nr\_min\_participanti int,in timp\_expirare int, in descriere varchar(45),in nume\_materie varchar(45),in nume\_prof varchar(45))

begin

set @ID\_meet=( SELECT MAX(meet\_grup.id\_meet) FROM platforma\_studiu.meet\_grup) + 1;

if @ID\_meet IS NULL then

set @ID\_meet=1;

end if;

set@IDmaterie=(SELECT materie.id\_materie from materie where materie.descriere=nume\_materie);

set@Idgrup=(SELECT grup\_studiu.id\_grup from grup\_studiu where grup\_studiu.id\_materie=@IDmaterie);

set @IDprof=(select profesor.id\_profesor from profesor where profesor.nume=nume\_prof);

insert into meet\_grup values(@ID\_meet,data\_meet,ora\_meet,nr\_min\_participanti,NULL,timp\_expirare,current\_time(),descriere,@Idgrup,@IDprof);

end ;;

**Pentru afișarea orarului profesorului asignat la un curs, în interfața grafică se apelează procedura:**

CREATE DEFINER=`root`@`localhost` PROCEDURE `orar\_prof\_curs`(in nume\_prof varchar(45), in prenume\_prof varchar(45), in nume\_zi varchar(30))

begin

set @IDprof = (Select profesor.id\_profesor from profesor

where profesor.nume=nume\_prof and profesor.prenume=prenume\_prof);

select 'curs',activitati.activitate\_predata, activitati.ora\_curs

from activitati,informatii\_profesor where

activitati.id\_info=informatii\_profesor.id\_info and activitati.zi\_curs=nume\_zi and informatii\_profesor.id\_profesor=@IDprof;

end ;;

BONUS: **Posibilitatea adăugării unui cadru didactic la o activitate din grupul de studiu**

CREATE DEFINER=`root`@`localhost` PROCEDURE `inscriere\_meet`( in nume\_stud varchar(45), in prenume\_stud varchar(45), in CNP varchar(45),in den\_materie varchar(45), descriere\_meet varchar(45), nume\_prof varchar(45))

begin

set @ID\_inscriereM=( SELECT MAX(inscriere\_meet\_grup.id\_inscriere\_meet) FROM platforma\_studiu.inscriere\_meet\_grup) + 1;

if @ID\_inscriereM IS NULL then

set @ID\_inscriereM=1;

end if;

-- select concat\_WS(' ',profesor.nume , profesor.prenume) as Nume

set@Idmaterie=(SELECT materie.id\_materie from materie where materie.descriere=den\_materie);

set @IDgrup=(select grup\_studiu.id\_grup from grup\_studiu where grup\_studiu.id\_materie=@Idmaterie);

set @IDprof=(select profesor.id\_profesor from profesor where profesor.nume=nume\_prof);

update meet\_grup set meet\_grup.nr\_curent\_participanti=meet\_grup.nr\_curent\_participanti+1 where meet\_grup.id\_grup=@IDgrup;

set@IDmeet=(SELECT meet\_grup.id\_meet from meet\_grup where meet\_grup.id\_grup=@IDgrup and meet\_grup.scriere=descriere\_meet and meet\_grup.id\_profesor=@IDprof);

set@data\_meet=(SELECT meet\_grup.data\_meet from meet\_grup where meet\_grup.id\_grup=@IDgrup and meet\_grup.scriere=descriere\_meet and meet\_grup.id\_profesor=@IDprof);

set@ora\_meet=(SELECT meet\_grup.ora\_meet from meet\_grup where meet\_grup.id\_grup=@IDgrup and meet\_grup.scriere=descriere\_meet and meet\_grup.id\_profesor=@IDprof);

set@timp\_expirare=(SELECT meet\_grup.timp\_expirare from meet\_grup where meet\_grup.id\_grup=@IDgrup and meet\_grup.scriere=descriere\_meet and meet\_grup.id\_profesor=@IDprof);

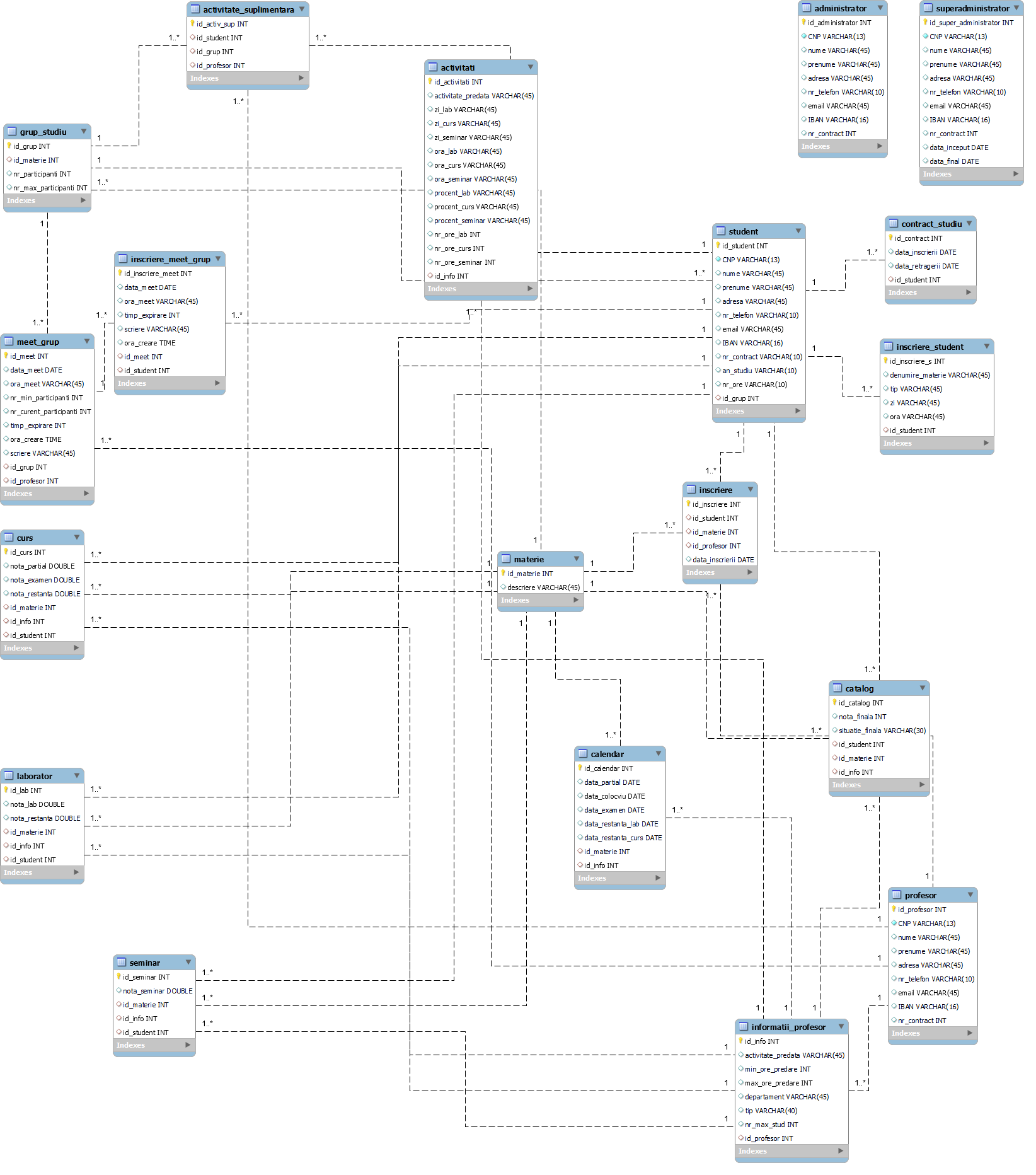
set @IDstud=(select student.id\_student from student where student.nume=nume\_stud and student.prenume=prenume\_stud and student.CNP=CNP);

insert into inscriere\_meet\_grup values(@ID\_inscriereM,@data\_meet,@ora\_meet,@timp\_expirare,descriere\_meet,current\_time(),@IDmeet,@IDstud);

end ;;

DELIMITER ;

Diagrama ERR:



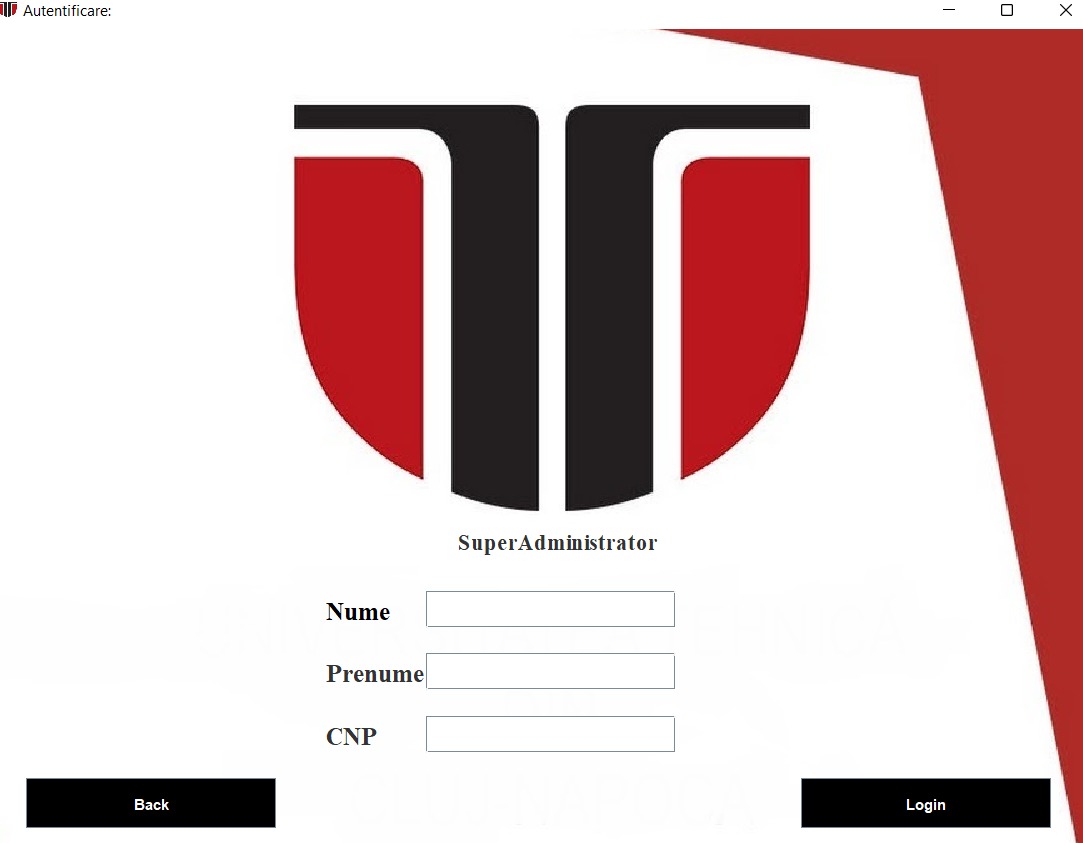
1. Detalii de implementare

În urma implementării tabelelor și a procedurilor necesare, am realizat conexiunea bazei de date MySQL cu Java, pe baza user-ului de tip „root” și a parolei personale din WorkBench.

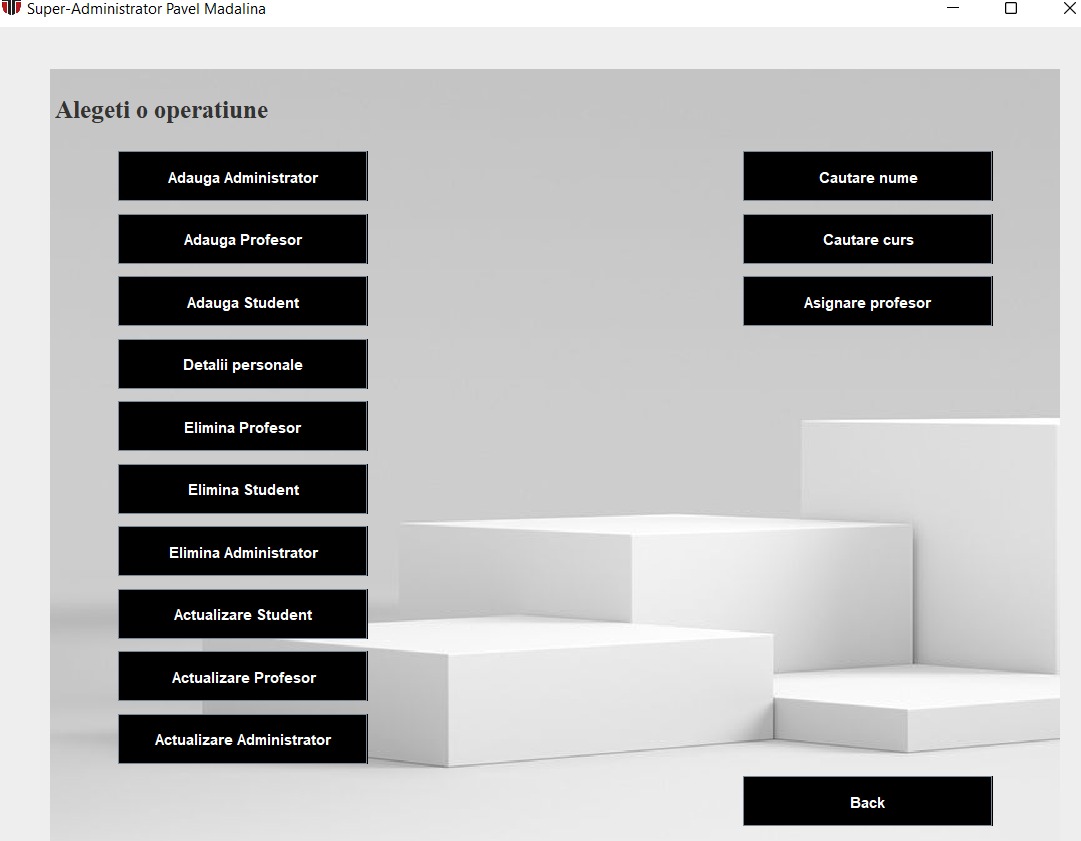
La rularea clasei Login din Eclipse se va deschide pagina de start, care va permite utilizatorului să selecteze în ce calitate se află.

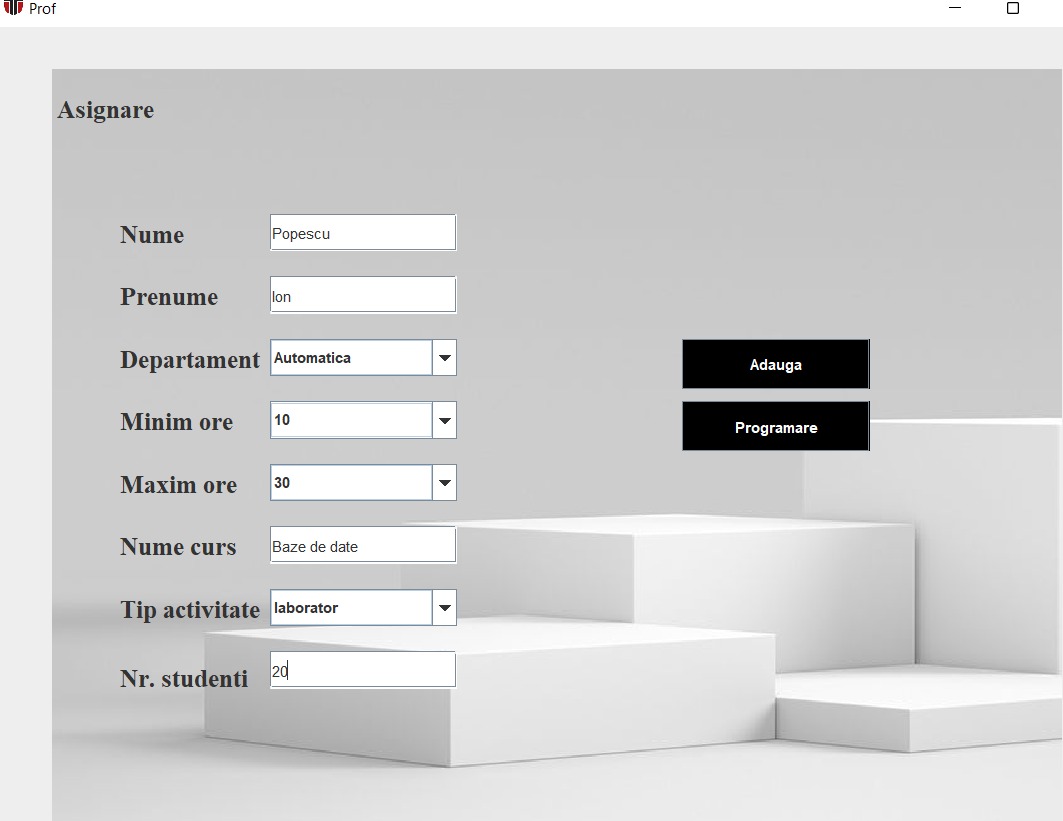


Autentificarea se realizează pe baza numelui, a prenumelui și a CNP-ului fiecărui utilizator. Dacă detaliile introduse apar în baza de date și sunt corecte, atunci utilizatorului i se va permite sa efectueze operațiile de care dispune individul respectiv.



Spre exemplu, la logarea în funcția de superadministrator vor fi disponibile butoanele de: vizualizare a detaliilor personale, de adăugare, ștergere și actualizare a profesorilor, studenților și administratorilor, căutare a cursurilor, căutare în funcție de nume și asignarea profesorilor la curs.





1. Concluzii. Dezvoltări ulterioare

În consecință, sistemul de gestiune al platformei de studiu implementat de noi are o funcționalitate fluidă, corectă și eficientă. Datorită îmbinării bazelor de date cu limbajul de programare Java, am făcut posibilă o accesare ușoară a platformei, pe baza numelui și a CNP-ului utilizatorului. De asemenea, designul îmbunătățește calitatea lucrării, datorită impactului vizual plăcut la accesare.

O serie de îmbunătățiri care ar putea fi aduse platformei în viitor sunt:

* Posibilitatea lăsării unor recenzii din partea studenților vis-a-vis de modalitatea de predare a profesorilor
* Permiterea accesului la adăugarea notelor și a profesorilor de laborator sau seminar
* Permiterea plății online a taxelor de reexaminare în cazul probelor nepromovate