# FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE

# DEPARTAMENTUL CALCULATOARE

## PROIECT

# la disciplina

# Introducere in Baze de Date

# Nume proiect: Platforma de studiu

# Profesor coordonator: Echipa de proiect:

Cosmina Ivan Silviu Grumazescu

Tamas Dari

Andrei-Denis Alexandru

Grupa 30221, Seria A, CTI

# An academic: 2021-2022

**Cuprins**

1. [**Introducere**](#_Introducere)
   1. Introducere, argumente, scop si obiective specifice
2. [**Suportul teoretic**](#_Suportul_teoretic)
3. [**Analiza cerințelor utilizatorilor (Specificațiile de proiect)**](#_Analiza_cerințelor_utilizatorilor)
   1. [Ipotezele specifice domeniului ales pentru proiect (cerințe, constrângeri)](#_Ipotezele_specifice_domeniului)
   2. [Organizare structurata(tabelar) a cerințelor utilizator](#_Organizare_structurată_a)
   3. [Determinarea si caracterizarea de profiluri de utilizatori](#_Determinarea_si_caracterizarea)
4. [**Modelul de date si descrierea acestuia**](#_Modelul_de_date)
   1. [Entități si atributele lor](#_Entități_si_atributele) (descriere detaliata **– implementarea fizica** )
   2. [Diagrama EER/UML pentru modelul de date complet](#_Diagrama_EER/UML_pentru)
   3. [Normalizarea datelor](#_Normalizarea_datelor)
5. [**Detalii de implementare MySQL**](#_Detalii_de_implementare)
   1. [DDL](#_DDL_(Data_Definition)
   2. [DML](#_DML_(Data_Manipulation)
   3. [SQL programatic](#_SQL_Programatic)
6. [**Detalii de implementare Java**](#_Detalii_de_implementare_1)
7. [**Utilizarea aplicației**](#_Utilizarea_aplicației)
8. [**Concluzii. Limitări si dezvoltări ulterioare**](#_Concluzii._Limitări_si)
9. [**Bibliografie**](#_Bibliografie)

# Introducere

Secolul XXI, supranumit și secolul vitezei, a adus cu el dezvoltări exponențiale și, în unele cazuri, revoluționare, în domeniul tehnologiei informației și în domeniul comunicațiilor. Pe fondul schimbărilor rapide și progresului tehnologic înregistrat, precum și pe fondul tendinței de globalizare a educației universitare, s-au deschis noi perspective pentru sistemul educațional universitar, această fiind completat cu metode moderne de abordare a educației precum și legăturii cu studenții.

Globalizarea a adus și dezavantajul de a facilita răspândirea rapidă a bolilor și virusurilor, iar pandemia generată de virusul SARS-COV 2 a evidențiat nevoia de digitalizare a sistemelor universitare. Proiectul prezentat reprezintă o abordare minimalistă, cu foarte mult loc de dezvoltare ulterioară, a satisfacerii nevoii menționate anterior, și presupune dezvoltarea unei aplicații ce lucrează cu baze de date pentru managementul unei platforme de studiu. Aplicația oferă sprijin atât pentru interogarea bazei de date, cât și pentru manipularea informațiilor stocate în ea, sprijin oferit de interfața grafică creată cu scopul unei interacțiuni facile între utilizator și sistemul universitar.

Pentru dezvoltarea proiectului s-au folosit următoarele:

* Apache NetBeans - mediu de dezvoltare integrat pentru Java
* MySQL Workbench - tool folosit pentru crearea bazei de date, popularea inițială, crearea de proceduri si triggere, crearea diagramei UML a tabelelor
* JDBC - pentru stabilirea conexiunii dintre aplicatie si baza de date
* Apache POI - oferă biblioteci Java pure pentru citirea și scrierea fișierelor în formate Microsoft Office, cum ar fi Word, PowerPoint și **Excel**

# Suportul teoretic

**MySQL**

MySQL este un sistem de gestionare a bazelor de date relaționale open-source care este utilizat în principal pentru aplicațiile online. MySQL poate crea și gestiona baze de date foarte utile (cum ar fi informații despre angajați, inventar și multe altele). Un SGBD relațional înseamnă că datele găzduite în structură sunt capabile să recunoască relațiile dintre informațiile stocate. Fiecare bază de date conține tabele. Fiecare tabel (denumit și o relație) conține una sau mai multe categorii de date stocate în coloane (denumite și atribute). Fiecare rând (denumit, de asemenea, o înregistrare sau „tuple”) conține o informație unică (altfel menționată ca și cheie) pentru categoriile definite în coloane.

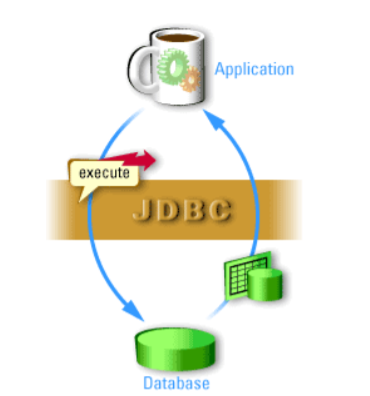
Există un acronim popular în industria tehnologiei: LAMP si WAMP. Acest acronim înseamnă Linux/Windows Apache MySQL PHP și este unul dintre cele mai populare „web stack-uri” de pe planetă (un „stack” este o stivă de software care interacționează împreună).

**Java**

Java este un limbaj de programare OOP sau orientat-obiect, dezvoltat de James Gosling la Sun Microsystems (acum filială Oracle), la începutul anilor ʼ90 și lansat în 1995. Limbajul împrumută o mare parte din sintaxa C și C++, dar are un model al obiectelor mai simplu. Java este un program în care se pot produce aplicații. După ce un programator dezvoltă o aplicație Java, aceasta poate rula pe majoritatea sistemelor de operare (OS), incluzând Windows, Linux și Mac OS, lucru ce face Java un limbaj versatil.

Este important de menționat că Java poate fi folosit pentru a dezvolta aplicații *complete*, care pot rula pe un singur computer sau care pot fi distribuite între servere și clienți într-o rețea. De asemenea, poate fi folosit pentru a programa miniaplicații sau *applets* care nu sunt independente, ci sunt parte a unei pagini web și facilitează interacțiunea utilizatorului cu interfața grafică.

**JDBC (Java Database Connectivity)**

JDBC (Java Database Connectivity) este tehnologia Java de acces la baze de date relaționale. Este independentă de tipul bazei de date, fiind orientată pe utilizarea de adaptoare (drivere) între client și SGBD.

Procesul de conectare la o bază de date implică înregistrarea unui driver corespunzător și realizarea unei conexiuni cu baza de date. O conexiune la o bază de date reprezintă un context prin care sunt trimise secvențe SQL către baza de date și sunt primite rezultate la nivel de aplicație client. API-ul JDBC furnizează acces orientat obiect la bazele de date, prin definirea de clase și interfețe ce modelează concepte abstracte specifice tehnologiei, semnificația celor mai importante concepte JDBC fiind următoarea:

* Driver - clasa Driver și Driver Manager.
* Conexiunea la baza de date – clasa Connection
* Interogări SQL - clasele Statement, PreparedStatement, CallableStatement
* Mulțimi rezultat al execuției - clasa ResultSet

Specificarea accesului la o bază de date creată și găzduită de un SGBD , dintr-o aplicație client Java presupune mai multe etape, ce trebuie parcurse într-o secvențiere strictă și anume: înregistrarea unui driver ce va fi utilizat pentru translatarea către baza de date a frazelor SQL, respectiv stabilirea conexiunii cu baza de date, corect specificată prin intermediul unui string de conexiune specific fiecărei baze de date.

# Analiza cerințelor utilizatorilor (Specificații de proiect)

# Ipotezele specifice domeniului ales pentru proiect

Se dorește implementarea unui sistem informatic destinat gestiunii unei platforme de studiu. Aplicația va folosi un sistem de gestiune pentru baze de date MySQL, iar interacțiunea cu aceasta va fi realizata doar prin interfața grafica. Funcționalitățile pe care le va oferi programul vizează operații ce țin de gestiunea studenților, profesorilor si administrarea operațiilor curente din cadrul unor programe de studiu.

Aplicația va putea fi accesată, pe baza unui proces de autentificare, de către mai multe tipuri de utilizatori: studenți, profesori, administratori. Pentru fiecare tip de utilizator se vor retine informații precum CNP, nume, prenume, adresa, număr de telefon, email, cont IBAN, numărul de contract. Fiecare utilizator își va putea vizualiza datele personale imediat după ce va accesa sistemul informatic, fără a avea însă posibilitatea de a le modifica. Totodată, programul trebuie să ofere și o funcționalitate pentru deautentificare, prin care se revine la fereastra care solicită datele de acces, astfel încât și un alt utilizator să îl poată folosi ulterior, fără a fi necesară repornirea sa.

Utilizatorul de tip **administrator** poate adăuga, modifica şi șterge informații în baza de date , informații legate de utilizatori. De asemenea, va exista şi un rol de super-administrator care poate opera inclusiv asupra utilizatorilor de tip administrator. Administratorii pot sa caute utilizatorii după nume si ii pot filtra după tip, pot asigna profesorii la cursuri si pot face căutare după numele cursului. La căutarea unui curs se afișează si numele profesorilor de la acel curs si un buton care permite vizualizarea tuturor studenților înscriși la cursul respectiv.

Pentru un utilizator de tip **profesor** se vor retine si cursurile predate, numărul minim si numărul maxim de ore pe care le poate preda si departamentul din care face parte.

Pentru un utilizator de tip **student** se va retine si anul de studiu si numărul de ore pe care trebuie sa le susțină.

Aplicația va permite gestiunea cu ușurință a activităților didactice si astfel a interacțiunilor dintre studenți si profesori. Cursurile sunt predate de mai mulți profesori și au una sau mai multe tipuri de activități (curs, seminar, laborator), o descriere, si un numar maxim de studenți participanți. Studenții se pot înscrie la cursuri si sunt asignați profesorului cu cei mai putini studenți la data înscrierii. Aceștia sunt evaluați cu note pentru fiecare tip de activitate si primesc o nota finala ca medie ponderata intre tipurile de activități. Profesorul stabilește din interfața grafica împărțirea procentuala pe tipurile de activități (ex. 20% seminar, 35% laborator, 45% curs/examenul de la curs).

Fiecare activitate se desfășoară recursiv intre doua date, pe o anumita perioada de timp. La asignarea unui profesor la un curs se vor alege tipurile de activități. De exemplu, profesorul X preda cursul Y cu activitățile: curs –săptămânal, laborator –săptămânal. Ulterior, profesorul poate programa activitățile (curs, seminar, laborator, colocviu, examen) într-un calendar, pe zile si ore, specificând si numărul maxim de participanți. Activitățile pot fi programate doar in viitor. Profesorii pot accesa un catalog, unde pot filtra studenții după cursuri si le pot adăuga note. Cataloagele pot fi descărcate sub forma de fișier.

La logare, studenții si profesorii pot sa își vadă activitățile din ziua curenta sau pot accesa o pagina cu toate activitățile la care sunt asignați / înscriși. Aceste liste pot fi descărcate din sistem sub forma unor fișiere.

Studenții se pot înscrie la cursuri, pot renunța la cursuri si își pot vedea notele. Aceștia trebuie sa aleagă activitățile la care vor sa participe si pot participa la ele doar daca mai sunt locuri sau nu exista o suprapunere cu o alta activitate (de exemplu, studentul dorește sa participe la laboratorul de BD marți la ora 10. Se înscrie la acea activitate, iar înscrierea este valida doar daca nu are deja o alta activitate marți la ora 10 sau daca mai sunt locuri disponibile. In caz contrar, se afișează un mesaj de eroare).

Totodată, studenții se pot înscrie in grupuri de studiu pentru o anumita materie, daca sunt înscriși la materia respectiva. Aceștia pot sa vadă toți membrii grupului si sa lase mesaje. Pe grup, studenții pot adaugă activități si sa definească un număr minim de participanți si o perioada in care ceilalți studenți pot sa anunțe participarea (de exemplu, un student adaugă o activitate de aprofundare a cursului pentru data de 12.12.2020, ora 16:00, 2 ore, cu număr minim de participanți 5 si timp de expirare 2 ore). Daca numărul minim nu este atins, activitatea se anulează, iar studenții înscriși la ea primesc un mesaj de informare.

# Organizare structurată a cerințelor utilizatorilor

Baza de date trebuie sa stocheze informații despre:

● Utilizatori (super-admin, admini, profesori, studenți)

● Cursuri

● Profesorii asignați la un curs

● Participanții la un curs

● Grupurile de studiu

● Activitățile fiecărui curs si ale grupelor de studiu

● Participanții la o activitate

● Mesajele de pe grupurile de studiu

● Membrii unui grup

● Invitațiile fiecărui utilizator

Mai mult trebuie sa permită si următoarele operații:

● Adăugarea de utilizatori noi daca nu există

● Adăugarea de cursuri daca nu există

● Adăugarea de activități

● Adăugarea de grupuri

● Ștergerea automata a invitațiilor

● Ștergerea automata a activităților de grup

● Calcularea notei finale a unui student la un curs

● Verifica suprapunerea a doua activități in momentul asignării unui student la o materie

● Calcularea numărului total de participanți la o activitate de grup

● Găsirea profesorului cu numărul minim de participant

Aplicația Java trebuie sa permită prelucrarea informațiilor din baza de date, căutarea, ștergerea, modificarea si afișarea informațiilor si conectarea utilizatorilor înregistrați.

# Determinarea si caracterizarea de profiluri de utilizatori

Orice utilizator deține următoarele funcționalități:

* Autentificare/Deautentificare
* Vizualizare date personale
* Vizualizare activități într-o zi anume sau in toate zilele
* Vizualizare cursuri
* Vizualizare invitații
* Participare la o activitate didactica sau a unei grupei de studiu

Student

Înscriere/Părăsire/Căutare curs

Înscriere activități didactice

Creare/Înscriere/Vizualizare grup de studio

Invitare colegi

Creare/Înscriere activitate de grup

Chat cu grupul de studio

Vizualizare note

Profesor

Programare activități didactice

Setare ponderi activități

Vizualizare studenți si înscriere note

Administrator

Gestionare conturi (Vizualizare, Modificare, Ștergere)

Creare cursuri

Asignarea profesorilor la cursuri

# Modelul de date si descrierea acestuia

# Entități si atributele lor

**Users** - informații generale despre utilizatori. Atribute: username (PK), password, CNP, nume, prenume, adresa, nr\_telefon, email, IBAN, nr\_contract, role.

**Studenți** - informații suplimentare despre student. Atribute: username (PK, FK), an\_studiu, nr\_ore.

**Profesori** - informații suplimentare despre profesor. Atribute: username (PK, FK), departament, min\_ore, max\_ore.

**Cursuri** - informații despre cursuri. Atribute: nume\_curs (PK), descriere, nr\_max\_participanti.

**Asignare Profesori la Cursuri** - informații despre profesorii asignați unor discipline. Atribute: id\_asignare\_prof\_curs (PK), username (FK), nume\_curs (FK), pondere\_curs, pondere\_laborator, pondere\_seminar, bool\_curs, bool\_laborator, bool\_seminar.

**Asignare Studenți la Cursuri** - informații despre studenții înscriși la cursuri. Atribute: id\_asignare\_student\_prof\_curs (PK), username (FK), id\_asignare\_prof\_curs (FK), nota\_finala, nota\_curs, nota\_laborator, nota\_seminar.

**Activitate** - informații despre activități. Atribute: id\_activ (PK), start\_date, final\_date, zi, ora, durata, tip\_activitate, locuri\_max, id\_asignare\_prof\_curs (FK), id\_activ\_grupa (FK), nr\_part\_curent.

**Participanti activitate** - informații despre participanții la o activitate. Atribute: id\_part (PK), id\_activ (FK), username (FK), nota\_activitate.

**Grupa Studiu** - informații generale despre o grupa de studio. Atribute: id\_grupa (PK), nume\_curs (FK).

**Participanți grupa** - informații despre participanții la o activitate de grupa. id\_part\_grupa(PK), username (FK), id\_grupa (FK).

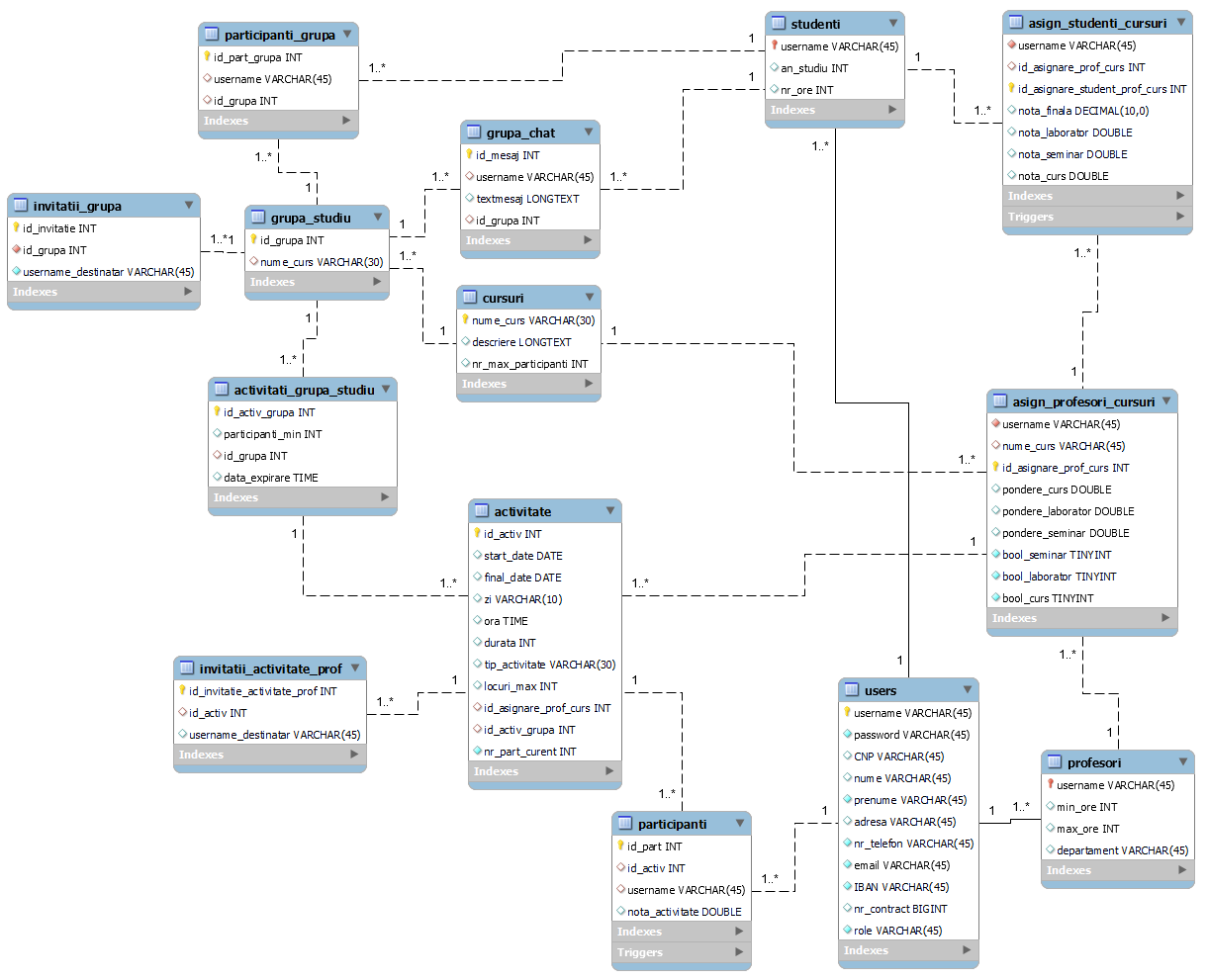
**Activitati grupa** - informații despre activitățile grupelor de studiu. Atribute: id\_activ\_grupa (PK), participant, id\_grupa (FK), data\_expirare.

**Chat grupa ­**- stochează mesajele scrise pe chat-urile grupelor. Atribute: id\_mesaj (PK), username (FK), textmesaj, id\_grupa (FK).

**Invitații profesori la activitate de grupa** - invitațiile la o activitate către profesori. Atribute: id\_invitatie\_activitate\_prof (PK), id\_activ (FK), username\_destinatar.

**Invitații student in grupa** - invitațiile in grupe către studenti. Atribute: id\_invitatie (PK), id\_grupa (FK), username\_destinatar.

# Diagrama EER/UML pentru modelul de date complet



# Normalizarea datelor

Normalizare bazelor de date este un proces de optimizare a bazei de date prin care se încearcă minimizarea redundantei datelor, si a anomaliilor de introducere, actualizare si ștergere.

Pentru a identifica corect o pereche formata dintr-un profesor si cursul pe care îl preda, sau intre student si cursul la care este înscris, si de asemenea, pentru a rezolva relațiile many-to-many (un student se poate înscrie la mai multe cursuri & un curs poate avea mai mulți studenți înscriși si un profesor poate preda mai multe cursuri & un curs poate fi predat de mai mulți profesori), am introdus tabelele auxiliare **asign\_profesori\_cursuri** si **asign\_profesori\_cursuri**. Fiecare pereche profesor-curs, respectiv sintaxa “student X înscris la cursul Y predate de profesorul Z”, are un ID unic prin care este identificata.

Baza noastră de date respectă **BCNF (forma normala Boyce-Codd)**. Atributele fiecărui tabel nu depind de alte atribute, si totodată respecta principiul atomicității. Fiecare tabel are o singură cheie primară după care sunt identificate înregistrările și este suficientă pentru a identifica în mod unic orice înregistrare din baza de date.

# Detalii de implementare MySQL

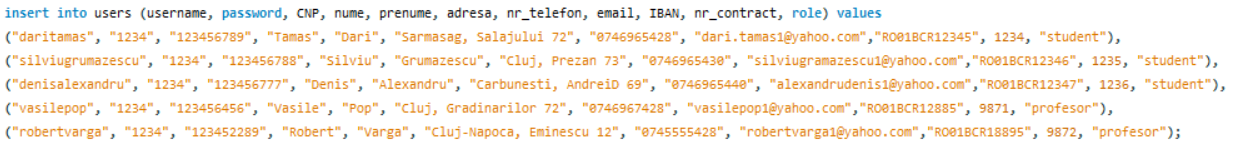
# DDL (Data Definition Language)

* Crearea bazei de date
* O imagine care conține text

  Descriere generată automatCrearea unui table
* O imagine care conține text

  Descriere generată automatAdăugarea cheilor străine si a constrângerilor

# DML (Data Manipulation Language)

* Inserarea datelor in tabele (Popularea)
* Modificarea datelor din tabele

# SQL Programatic

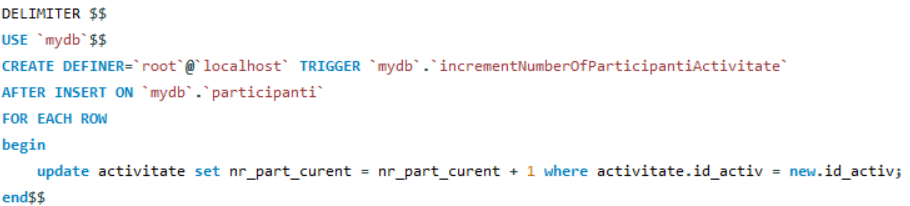
* Proceduri
* O imagine care conține text

  Descriere generată automatÎnregistrarea unui utilizator
* O imagine care conține text

  Descriere generată automatObținerea datelor unui utilizator
* O imagine care conține text

  Descriere generată automatRepartizarea unui student la profesorul cu cei mai putini studenti atribuiti
* O imagine care conține text

  Descriere generată automatVerificarea existentei unei activitati intr-un interval de timp
* O imagine care conține text

  Descriere generată automatObtinerea activitatilor la care poate participa un student
* Triggere
* O imagine care conține text

  Descriere generată automatIncrementarea numărului participanților la o activitate
* Calcularea notei finale pe baza notelor obtinute si ponderilor stabilite
* Event

# Detalii de implementare Java

O imagine care conține text

Descriere generată automatO imagine care conține text

Descriere generată automatInteracțiunea dintre utilizator si baza de date se realizează cu ajutorul interfeței grafice, creata special pentru a satisface nevoile fiecărui utilizator. Ideile centrale in jurul cărora s-a dezvoltat interfața au fost separarea utilizatorilor in roluri si respectarea arhitecturii MVC. Fiecărui obiect de tip **User** ii corespunde un parametru **role**, care oferă utilizatorului acces la **view**-ul corespunzător prin intermediul unui **controller**.

O imagine care conține text

Descriere generată automatPanoul principal al userului de tip student:

Fragmente de cod din interiorul controller-ului studentului:O imagine care conține text

Descriere generată automat

# O imagine care conține text Descriere generată automatIn urma folosirii elementelor interfeței (butoane, liste, tabele, panouri), pentru ca operația ceruta de către utilizator sa poată fi realizata, este nevoie de a interacționa cu baza de date prin intermediul interogărilor sau modificărilor de tip ALTER, INSERT, DELETE. Aceste operații sunt posibile cu ajutorul unor metode definite in clasa **BDConnection**.

Clasa **BDConnection** este responsabila de a stabili conexiunea dintre baza de date si aplicația Java prin intermediul **JDBC**-ului.

O imagine care conține text

Descriere generată automat

# Exemple de metode din cadrul clasei BDConnection (metode apelate din cadrul tuturor controllerelor):

# O imagine care conține text Descriere generată automat O imagine care conține text Descriere generată automat

O imagine care conține text

Descriere generată automatO imagine care conține text

Descriere generată automat

O imagine care conține text

Descriere generată automat

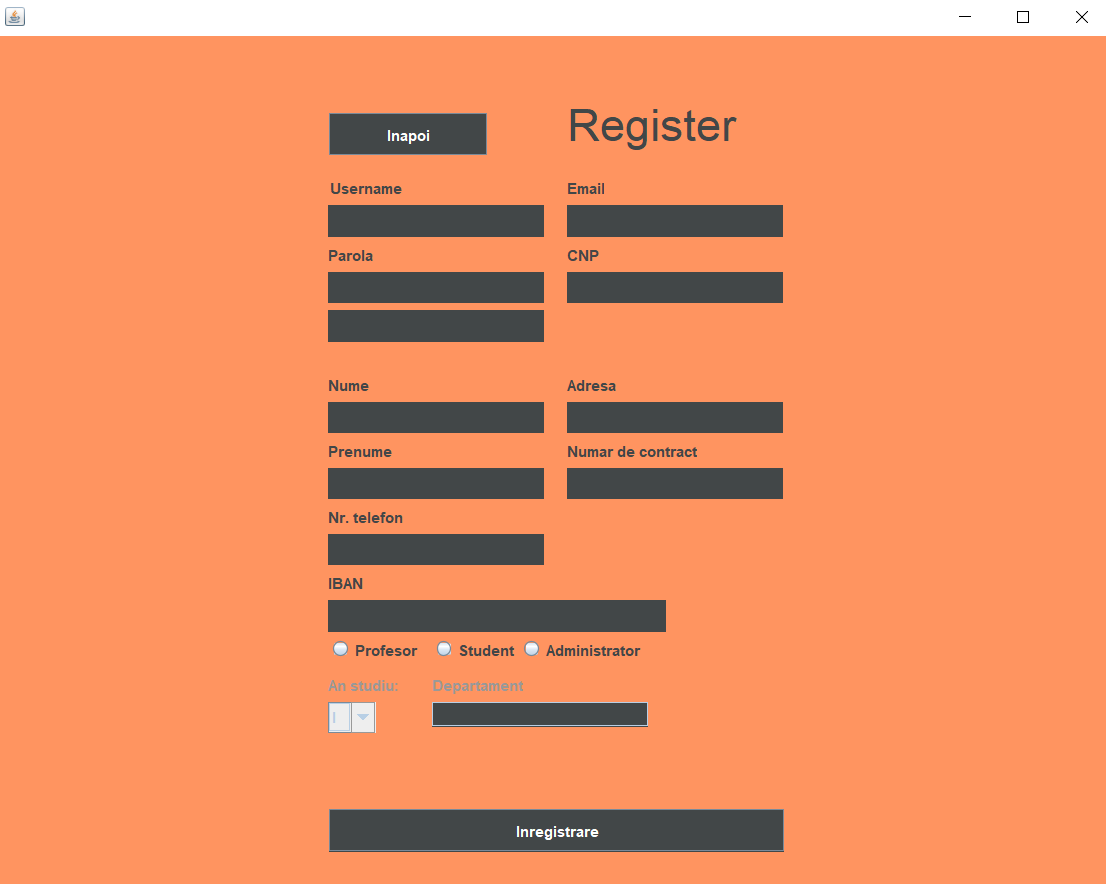
# Utilizarea aplicației

# Pagina de întâmpinare

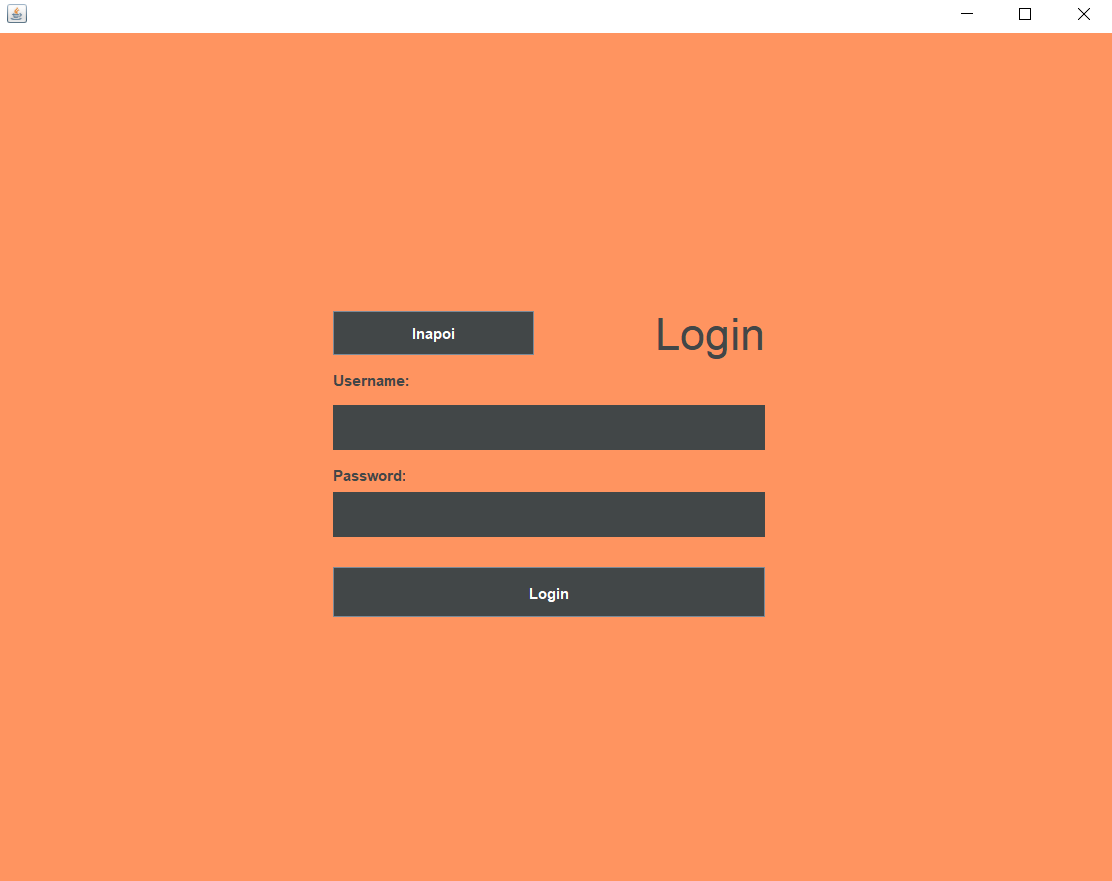
Graphical user interface

Description automatically generated

# Panoul de înregistrare



# Panoul de autentificare



# Graphical user interface Description automatically generatedInterfața studentului

# Elemente ale interfeței studentului

* Graphical user interface, treemap chart, PowerPoint

  Description automatically generatedPanoul grupei de studiu
* Graphical user interface

  Description automatically generated with medium confidencePanoul pentru vizualizarea notelor
* Graphical user interface, application, PowerPoint

  Description automatically generatedPanoul pentru înscrierea la cursuri
* Table

  Description automatically generatedPanoul pentru vizualizarea activităților
* Graphical user interface, timeline

  Description automatically generatedPanoul pentru înscrierea la activități

# Graphical user interface Description automatically generatedInterfața profesorului

* Graphical user interface, chart

  Description automatically generatedPanoul pentru administrarea cursurilor
* Calendar

  Description automatically generatedPanoul pentru vizualizarea activităților
* Graphical user interface, timeline

  Description automatically generatedPanoul pentru vizualizarea si administrarea catalogului
* Graphical user interface

  Description automatically generatedPanoul pentru programarea activităților
* Graphical user interface

  Description automatically generatedPanoul pentru vizualizarea invitațiilor

# Interfața administratorului

Graphical user interface, application

Description automatically generated

# Graphical user interface, chart Description automatically generatedPanoul de date personale

# Concluzii. Limitări si dezvoltări ulterioare

Proiectul prezentat oferă o soluție pentru digitalizarea si gestionarea eficienta a sistemului informațional al unei platforme de studiu al unei universități. Utilizatorii către care este destinata platforma sunt studenții, profesorii si administratorii.

Aplicația oferă diferite servicii in funcție de tipul utilizatorului autentificat, accesarea si corectitudinea acestor servicii fiind facilitata de interfața grafica configurata special in acest scop. Vizualizarea propriilor date este o funcționalitate comuna pentru toate tipurile de utilizator, la fel ca vizualizarea programului de activități.

In timp ce conturile de student, profesor sau administrator sunt generate in urma unei înregistrări, rolul de super-administrator este setat prin popularea implicita a bazei de date. Exista funcționalități particulare determinate de rolul ales la înregistrare (Ex: Un student nu poate modifica ponderile sau notele de la un curs, un profesor nu poate vizualiza chatul unei grupe de studiu). Super-administratorul are, in plus fata de un administrator normal, dreptul de a vizualiza si manipula datele personale ale celorlalți administratori.

In cazul unor dezvoltări ulterioare, platforma poate beneficia de un design atractiv al interfeței grafice prin adăugarea unor pictograme sugestive sau a unei secțiuni de Ajutor. Platforma are deja caracteristica de a programa activități de tip examen sau colocviu, așa ca ar fi destul de intuitiv ca in viitor sa se implementeze si capabilitatea susținerii acestor examene prin intermediul platformei.

Pentru a i se asigura practicitatea, platforma ar putea beneficia in viitor si de o baza de date stocata in mediul CLOUD, lucru ce ar putea facilita accesarea platformei pe dispozitive care nu au baza de data creata in prealabil, deci implicit, de pe telefoane mobile.

# Bibliografie

1. MVC Architecture in Java - <https://www.javatpoint.com/mvc-architecture-in-java>
2. Package javax.swing - <https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html>
3. Event Scheduler Overview - <https://dev.mysql.com/doc/refman/5.7/en/events-overview.html>
4. Apache POI - the Java API for Microsoft Documents - <https://poi.apache.org/>
5. JDBC Basics - <https://docs.oracle.com/javase/tutorial/jdbc/basics/index.html>
6. How To Use MySQL Triggers - <https://phoenixnap.com/kb/mysql-trigger>
7. **MySQL ON DELETE CASCADE -** <https://www.educba.com/mysql-on-delete-cascade/>
8. What is Normalization in SQL and what are its types? <https://www.edureka.co/blog/normalization-in-sql/>
9. Class Collections - <https://docs.oracle.com/javase/7/docs/api/java/util/Collections.html>
10. How to save edited JTable data to database? - <https://stackoverflow.com/questions/18151123/how-to-save-edited-jtable-data-to-database>
11. Introducerea in Baze de Date, Cosmina Ivan - <https://ftp.utcluj.ro/pub/users/civan/IBD/1_CURS_Resurse/>