Cuprins

1. Introducere

1.1 Descrierea Generală a Proiectului

1.2 Scopul și Obiectivele

2. Tema Proiectului

2.1 Prezentarea Aplicației

2.2 Funcționalități și Utilizatori

3. Descrierea Bazei de Date

3.1 Structura generala si Detalii specifice

3.2 Nivelul de Normalizare

4. Elemente de programare a functionalitatilor

4.1 Proceduri și Triggere.Vederi.Lista completa si exemple

4.2. Alte exemple de Cod SQL

5. Interfața Grafică a Utilizatorului (GUI)

5.1 Descrierea functionalitatilor per tip de utilizator

5.2 Detalii de implementare funcționalități specifice

6. Manual de Utilizare

6.1 Înregistrare și Autentificare

6.2 Gestionarea Informațiilor per utilizatori

7. Concluzii și dezvoltari ulterioare

7.1 Analiza funcționalităților curente- cele mai interesante solutii adoptate

7.2 Dezvoltari ulterioare

1.Introducere:

1.1 Descrierea Generală a Proiectului

Proiectul „Introducere în Baze de Date” a avut ca scop dezvoltarea unei aplicații informatice cu interfață grafică, utilizând principii avansate de programare orientată pe obiecte, gestionarea bazelor de date relaționale și implementarea controlului accesului bazat pe roluri (RBAC). Aplicația realizată permite gestionarea eficientă a unei platforme de studiu, oferind funcționalități adaptate fiecărui tip de utilizator, cum ar fi studenți, profesori, administratori și super-administratori.

Aplicația utilizează o bază de date MySQL pentru stocarea informațiilor despre utilizatori, cursuri, activități și alte operațiuni specifice unei platforme academice. Interacțiunea utilizatorilor cu sistemul este realizată prin intermediul unei interfețe grafice intuitive, care facilitează utilizarea aplicației indiferent de nivelul tehnic al utilizatorilor. Sistemul oferă funcționalități complexe, printre care autentificarea utilizatorilor, gestionarea datelor personale și operațiunilor specifice fiecărui rol, administrarea cursurilor și a activităților, precum și exportul de rapoarte și cataloage.

Prin procesul de autentificare, aplicația asigură accesul diferențiat pentru utilizatori, în funcție de drepturile lor. Utilizatorii își pot vizualiza datele personale, însă modificările acestora sunt permise doar administratorilor. De asemenea, super-administratorii au drepturi extinse, incluzând gestionarea conturilor administratorilor. Gestionarea cursurilor și activităților include crearea, modificarea și programarea acestora în calendar, respectând reguli precum evitarea suprapunerilor sau limitarea locurilor disponibile.

Pentru studenți, aplicația facilitează înscrierea la cursuri și activități, vizualizarea notelor și participarea în grupuri de studiu, unde pot organiza activități suplimentare. Profesorii au acces la funcționalități de gestionare a cursurilor predate, adăugarea notelor pentru studenți și generarea cataloagelor descărcabile. Administratorii au rolul de a gestiona utilizatorii, de a aloca profesorii la cursuri și de a coordona operațiunile generale ale platformei.

Implementarea proiectului a utilizat limbajul Java pentru dezvoltarea funcționalităților aplicației, împreună cu MySQL pentru gestionarea bazei de date. Structura aplicației reflectă un design bine definit, bazat pe modele UML și standarde de bune practici în dezvoltarea software. Aplicația reprezintă un instrument util pentru gestionarea activităților academice și demonstrează aplicarea practică a conceptelor teoretice din disciplina „Introducere în Baze de Date”.

1.2 Scopul și Obiectivele

Scopul principal al proiectului este dezvoltarea unei aplicații informatice cu interfață grafică, destinată gestionării eficiente a unei platforme de studiu. Proiectul urmărește aplicarea conceptelor teoretice din domeniul bazelor de date, utilizând tehnologii pentru manipularea acestora și implementând un sistem informatic ce răspunde nevoilor utilizatorilor cu roluri diferite. Aplicația va facilita gestionarea informațiilor academice printr-o interfață ușor de utilizat, oferind acces diferit pentru fiecare tip de utilizator, precum studenți, profesori, administratori și super-administratori.

Primul obiectiv al proiectului este proiectarea unei baze de date relaționale normalizate, care să conțină tabele interconectate, ce vor permite gestionarea eficientă a datelor. Un alt obiectiv important este implementarea unui sistem de control al accesului bazat pe roluri (RBAC), care va asigura permisiuni diferențiate pentru utilizatori în funcție de rolul acestora, fie că sunt studenți, profesori sau administratori. În ceea ce privește dezvoltarea aplicației, proiectul vizează realizarea unei interfețe grafice intuitive și accesibile, care să faciliteze interacțiunea utilizatorilor cu sistemul.

Un alt obiectiv este asigurarea integrității datelor prin implementarea mecanismelor de validare pentru a evita suprapunerile de activități și erorile în gestionarea datelor. În plus, aplicația va oferi opțiuni de export și raportare, permițând utilizatorilor să descarce rapoarte și activități programate pentru o administrare mai ușoară și mai eficientă a platformei. Spre exemplu methoda valid() din clasa LogareBackend.

2. Tema Proiectului

2.1 Prezentarea Aplicației

**Platforma de Studiu** este o aplicație informatică destinată gestionării activităților academice într-un sistem educațional. Aplicația este creată pentru a sprijini interacțiunea dintre studenți, profesori și administratori, oferind un mediu centralizat pentru gestionarea informațiilor educaționale și administrative. Scopul principal este de a eficientiza procesele academice, cum ar fi înscrierea la cursuri, evaluarea studenților, organizarea activităților didactice și gestionarea drepturilor de acces ale utilizatorilor.

**Funcționalități cheie ale aplicației:**

* **Autentificare și controlul accesului**: Sistemul de autentificare permite utilizatorilor să se conecteze pe baza unui nume de utilizator și a unei parole. Controlul accesului este realizat pe baza unui sistem de roluri (RBAC), iar fiecare utilizator are acces la diferite funcționalități în funcție de rolul său (student, profesor, administrator, super-administrator).
* **Gestionarea activităților academice**: Aplicația permite profesorilor să creeze și să administreze cursuri, seminarii și laboratoare, iar studenților să se înscrie la aceste activități. De asemenea, profesorii pot adăuga note și pot evalua performanța studenților pe baza activităților academice.
* **Calendar și programări**: Profesorii pot programa activitățile într-un calendar, iar studenții pot vizualiza aceste activități pentru a se înscrie în funcție de disponibilitatea lor. Suprapunerile de orar sunt prevenite, iar fiecare activitate poate fi limitată în număr de participanți.
* **Grupuri de studiu**: Studenții pot crea grupuri de studiu pentru un anumit curs, unde pot interacționa cu alți colegi, pot organiza sesiuni de aprofundare a materiilor și pot adăuga activități cu un număr minim de participanți.
* **Gestionarea informațiilor utilizatorilor**: Fiecare utilizator poate vizualiza informațiile sale personale (CNP, nume, prenume, adresa, etc.), dar nu poate modifica aceste date decât în anumite condiții stabilite de administrator.

Aplicația asigură o interfață grafică prietenoasă, prin care utilizatorii pot naviga ușor printre funcționalitățile disponibile, beneficiind de o experiență eficientă și organizată. Acest sistem contribuie la reducerea timpilor de gestionare manuală și la centralizarea tuturor informațiilor necesare pentru buna desfășurare a activităților academice.

**Beneficii:**

* **Eficiență în administrarea cursurilor**: Profesorii și administratorii pot gestiona mai ușor activitățile academice și notele studenților.
* **Flexibilitate pentru studenți**: Studenții au posibilitatea de a se înscrie la cursuri, de a vizualiza activitățile și de a urmări progresul lor academic.
* **Acces rapid la informații**: Toți utilizatorii pot accesa rapid datele relevante din platformă, în funcție de drepturile lor de acces.

Aplicația a fost construită folosind tehnologii de ultimă generație, cu un backend bazat pe MySQL și un frontend intuitiv, care permite un management eficient al informațiilor și activităților.

2.2 Funcționalități și Utilizatori

Pentru functionalitati, creearea aplicatiei s-a bazat pe aces Use-Case-Diagram: A diagram of a diagram

Description automatically generated

3. Descrierea Bazei de Date

3.1 Structura generala si Detalii specifice

Baza de date **platforma\_studiu** este concepută pentru a gestiona informațiile referitoare la utilizatori, cursuri, activități academice și interacțiunile dintre studenți și profesori. Ea este structurată pe mai multe tabele, fiecare având rolul de a organiza și stoca informațiile necesare în funcție de tipul de utilizator și de activitatea desfășurată.

**1. Tabele pentru utilizatori:**

* **utilizator:** Stochează informațiile de bază ale utilizatorilor (student, profesor, administrator, super-administrator), inclusiv detalii personale și date de contact.
* **autentificare:** Conține informațiile de autentificare pentru utilizatori, asociind fiecare utilizator cu un username și parolă.
* **student:** Stochează informațiile adiționale pentru studenți, cum ar fi anul de studiu și numărul de ore.
* **profesor:** Conține detalii despre profesori, precum numărul minim și maxim de ore predat, și departamentul din care fac parte.

**2. Tabele pentru gestionarea cursurilor și activităților:**

* **curs:** Conține detalii despre cursuri (descrierea cursului, numărul maxim de studenți).
* **activitate\_profesor:** Leagă activitatiile de profesori, indicând care profesori sunt asignați la ce activitati.
* **curs\_student:** Stochează relațiile dintre studenți și cursuri, incluzând notele atribuite pentru fiecare curs.
* **activitate:** Detaliază activitățile asociate fiecărui curs (tipul activității și ponderea acestora).
* **calendar:** Conține informații despre programul activităților, inclusiv perioada de desfășurare a acestora.

**3. Tabele pentru gestionarea grupurilor de studiu:**

* **grup:** Stochează informații despre grupurile de studiu, inclusiv cursul asociat, numărul minim de participanți și data de expirare.
* **grup\_activitate:** Leagă grupurile de activitățile asociate.
* **grup\_studenti:** Asociază studenții cu grupurile de studiu, permițând managementul participării la activitățile grupului.
* **mesaj:** Permite comunicarea între membrii unui grup de studiu, stocând mesajele lăsate de studenți.

**Relații între tabele:**

* Tabelele sunt interconectate prin chei externe care asigură integritatea datelor și corectitudinea relațiilor dintre entități (ex. relația dintre cursuri și profesori, studenți și activități, etc.).
* De exemplu, tabela **curs\_profesor** asigură legătura între cursuri și profesori

**grup\_studenti** asociază studenții cu grupurile de studiu, iar **mesaj** permite interacțiunea între membrii grupurilor.

Această structură permite gestionarea eficientă a unei platforme de studiu, asigurând accesul diferențiat la informații și funcționalități pentru fiecare tip de utilizator, precum și integritatea datelor prin diverse mecanisme de validare și ștergere automată.

3.2 Nivelul de Normalizare

Normalizarea bazei de date este un proces esențial pentru eliminarea redundanței datelor și pentru asigurarea integrității informațiilor. În acest proiect, structura bazei de date respectă principiile normalizării pentru a obține o bază de date eficientă, flexibilă și ușor de gestionat. Baza de date este normalizată la cel puțin **3NF (Formă Normală de Trei)**, dar este posibil ca unele tabele să fi fost optimizate pentru performanță sau ușurință în utilizare.

Tabelele din acest proiect respectă 3NF, deoarece nu există dependențe tranzitive între atributele non-cheie. De exemplu, în tabela **utilizator**, datele de contact și detaliile legate de utilizator sunt separate în tabele diferite, precum tabela **autentificare** (care conține doar datele de autentificare ale utilizatorilor), eliminând astfel orice dependență tranzitivă.

Deși baza de date este normalizată, în scopul optimizării performanței și a accesului rapid la date, au fost introduse câteva indexuri pentru a accelera căutările și interogările frecvente legate de activitățile programate.

4. Elemente de programare a functionalitatilor

4.1 Proceduri și Triggere.Vederi.Lista completa si exemple

Proceduri: CheckStudentActivity (int, datetime, datetime) – verifica daca studentul cu id,

dat ca paramentru are vreo activitate intre cele 2 datetime-uri

Triggere: delete\_student – sterge informatiile relevante din baza de date odata cu stergerea unei intrari in tabela student

delete\_profesor – sterge informatiile relevante din baza de date odata cu stergerea unei intrari in tabela profesor

delete\_autentificare – sterge intrarea respectiva din autentificare odata cu stergerea intrarii din utilizator

Views: all\_profesors – arata informatiile tuturor profesorilor din utilizator

all\_studenti - arata informatiile tuturor studentiilor din utilizator

4.2. Alte exemple de Cod SQL

SELECT CASE  
 WHEN *EXISTS* (  
 SELECT 1  
 FROM curs\_student  
 JOIN curs ON curs\_student.curs\_id = curs.id  
 JOIN grup ON grup.curs\_id = curs.id  
 WHERE curs\_student.student\_id = 1 AND grup.id = 5  
 )  
 THEN 'Student can join the group'  
 ELSE 'Student cannot join the group'  
END AS result;

* + Verifica daca un student poate intra intrun grup, daca e inscris la curs si nu e intrat in grup, atunci poate altfel nu.

select curs.id,nume,prenume from utilizator

join curs\_student on student\_id=utilizator.id

join curs on curs\_student.curs\_id = curs.id

where utilizator.id=1

- returneaza cursurile unde e studentul cu un anumit id

SELECT  
 CASE  
 WHEN *EXISTS* (  
 SELECT 1  
 FROM calendar c1  
 JOIN activitate\_nota an ON c1.activitate\_id = an.activitate\_id  
 WHERE an.student\_id = 1  
 AND *EXISTS* (  
 SELECT 1  
 FROM calendar c2  
 WHERE c2.activitate\_id = 12  
 AND (  
 c1.data\_inceput < c2.data\_sfarsit  
 AND c1.data\_sfarsit > c2.data\_inceput  
 )  
 )  
 ) THEN TRUE  
 ELSE FALSE  
 END AS overlap\_exists;

-verifica daca exista overlap la o activitate unde vrea sa se inscrie studentul cu restul la care e inscris deja

select distinct c.id from curs as c join  
 activitate as a on a.curs\_id=c.id  
join activitate\_profesor on a.id = activitate\_profesor.activitate\_id  
join profesor p on p.id=activitate\_profesor.profesor\_id  
where profesor\_id=21;

-verifica cursurile la care un profesor preda activitaiile

SELECT  
 mesaj.continut,  
 CASE  
 WHEN mesaj.student\_id = 1 THEN 'Yes'  
 ELSE 'No'  
 END AS is\_from\_student  
FROM mesaj  
where grup\_id=1;

* + Afiseaza mesajele unui grup, fiind scriese dau nu de persoana care s-a logat

5. Interfața Grafică a Utilizatorului (GUI)

5.1 Descrierea functionalitatilor per tip de utilizator

La deschiderea aplicatiei utilizatorul v-a trebui sa se logheze

A screenshot of a login screen

Description automatically generated

Ca si student vei fi intampinat astfel:

A screenshot of a computer

Description automatically generated

Poti accesa cursurile si acttivitatiile tale astfel:

A screenshot of a computer

Description automatically generatedA screenshot of a calendar

Description automatically generated

Te poti inscrie la activitati:

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Poti vedea activitati din toate zilele sau de astazi:

A screenshot of a computer

Description automatically generatedA screenshot of a phone

Description automatically generated

Poti accesa functionalitatiile de grup:

A screenshot of a chat

Description automatically generated

Ca si profesor vei fi intampinat astfel:

A blue and white screen with white text

Description automatically generated

Poti accesa activitatiile astfel unde poti programa activitati:

A screenshot of a computer

Description automatically generatedA screenshot of a calendar

Description automatically generated

In activitati poti nota studentii si schimba ponderea:

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

5.2 Detalii de implementare funcționalități specifice

Functionalitatiile din GUI s-au implementat cu ajutorul JavaFX, si Claselor de Backend.

6. Manual de Utilizare

6.1 Înregistrare și Autentificare

Se deschide aplicatia si se arata ecranul de autentificare. Conturile sunt creeate de catre administrator.

Administratorul va introduce în sistem informațiile necesare pentru crearea unui cont, inclusiv:

* CNP
* Nume și Prenume
* Număr de telefon
* Email
* IBAN (opțional)
* Număr contract (pentru profesori și administratori)
* Tipul de utilizator

După ce contul a fost creat, utilizatorii vor putea să se autentifice în aplicație. Pentru aceasta, este necesar să completeze următoarele câmpuri pe ecranul de autentificare:

* **Username**: Unic pentru fiecare utilizator.
* **Parola**: Parola stabilită în timpul înregistrării.

După introducerea corectă a datelor, utilizatorul va fi redirecționat în funcție de rolul său (Student, Profesor, Administrator sau Super-Administrator) pe pagina principală corespunzătoare.

**3. Funcționalitatea de deautentificare**:  
La finalizarea sesiunii, utilizatorii se pot deautentifica din aplicație prin apăsarea butonului "Deconectare". Aceasta va închide sesiunea curentă și va permite altui utilizator să acceseze aplicația.

6.2 Gestionarea Informațiilor per utilizatori

În aplicație, există patru tipuri de utilizatori: **Student**, **Profesor**, **Administrator** și **Super-Administrator**, fiecare având un set specific de privilegii și funcționalități accesibile. Gestionarea informațiilor pentru fiecare tip de utilizator se face în următoarele moduri:

**1. Gestionarea informațiilor pentru Student:**

* **Vizualizarea informațiilor personale**: Studenții pot vizualiza datele lor personale (CNP, nume, prenume, adresă, telefon, email, IBAN, număr contract), dar nu pot modifica aceste informații.
* **Înscrierea la activitati**: Studenții pot vizualiza cursurile disponibile și se pot înscrie la activitaiile la care sunt accesibile.
* **Participarea la activități/cursuri**: Studentii pot vedea cursurile si activitatiile la care participa
* **Vizualizarea grupurilor**: Studenții pot vizualiza grupurile, pot trimite mesaje, sa creeze activitati,..

**2. Gestionarea informațiilor pentru Profesor:**

* **Vizualizarea informațiilor personale**: Profesorii pot vizualiza informațiile lor personale, inclusiv cursurile pe care le predau și numărul de ore predate.
* **Gestionarea cursurilor**: Profesorii pot vizualiza cursurile și activitățile asociate acestora. De asemenea, pot stabili ponderile pentru fiecare tip de activitate (seminar, laborator, curs) și pot programa activitățile în calendarul platformei.
* **Atribuirea notelor**: Profesorii pot atribui note studenților pentru activitățile la care aceștia au participat. De asemenea, pot vizualiza lista studenților înscriși la cursurile sau activitățile lor.

**3. Gestionarea informațiilor pentru Administrator:**

* **Gestionarea utilizatorilor**: Administratorii pot adăuga, modifica și șterge utilizatori. Aceștia pot modifica informațiile studenților, profesorilor și altor administratori.
* **Cautarea utilizatoriilor:** Se pot cauta utilizatori si filtra dupa tip, se pot inscrie profesori la curs
* **Cautarea cursurilor:** Se pot cauta cursui

**4. Gestionarea informațiilor pentru Super-Administrator:**

* **Gestionarea utilizatorilor**: Administratorii pot adăuga, modifica și șterge utilizatori. Aceștia pot modifica informațiile studenților, profesorilor și altor administratori.
* **Cautarea utilizatoriilor:** Se pot cauta utilizatori si filtra dupa tip (inclusiv administrator), se pot inscrie profesori la curs
* **Cautarea cursurilor:** Se pot cauta cursui

Prin aceste funcționalități, aplicația asigură o gestionare eficientă și flexibilă a datelor utilizatorilor și facilitează administrarea activităților educaționale din cadrul platformei de studiu.

7. Concluzii și dezvoltari ulterioare

7.1 Analiza funcționalităților curente- cele mai interesante solutii adoptate

Una dintre cele mai interesante solutii adoptate este clasa Table care, primind ca input un Querry returneaza o matrice care contine toate valorile si atributele unui tabel returnat de querry.

De asemenea Clasa Querry poate executa operatii de tipul CRUD.

Aceste 2 clase au facut ca realizarea partii de backend a proiectului sa fie citeata si usor de folosit.

7.2 Dezvoltari ulterioare

**Raport de performanță personalizat**: Studenții și profesorii vor putea accesa rapoarte detaliate despre performanțele academice, incluzând note și statistici privind participarea la activități.

**Generarea de statistici avansate**: Platforma va putea genera statistici pentru administratori sau profesori despre rata de absolvire, performanțele pe tipuri de activități (curs, seminar, laborator) și distribuția notelor.

**Recomandări pentru studenți**: Un sistem care sugerează cursuri și activități pe baza performanțelor anterioare ale studentului, contribuind la dezvoltarea unei strategii educaționale personalizate.

**Platformă de lecții video**: Integrarea unui sistem de management al învățării (LMS) care să permită profesorilor să adauge lecții video și material de curs pentru studenți.

**Evaluări și teste online**: Permite realizarea de evaluări online, inclusiv teste automate și examene, cu funcționalități de corectare automată și generare de rapoarte.