

PROIECT LA DISCIPLINA BAZE DE DATE

Aplicatie de gestiune a unei institutii de invatamant

Proiect realizat de: Darius Razvan Radu,
Meze Razvan si Branet Tudor.
Studenti din grupele 30225, 30222, 30224

Cuprins

1. Tema proiectului	2
2. Proiectare conceptuala a bazei de date.....	3
3. Solutia de transformare in relational	4
4. Descrierea bazei de date relationale.....	5
5. Argumentarea nivelului de normalizare.....	7
6. Codul in SQL	8
7. Functionalitatea aplicatiei	9
7.1. ConnectionPackage.....	9
7.2. MainPackage.....	12
7.3. GUIPackage	12
7.4. ModelsPackage	22
8. Manual de utilizare	23
9. Concluzii si dezvoltari ulterioare	24

1. Tema proiectului

Aplicația oferită reprezintă un sistem informatic de gestiune a unei platforme de studiu prin intermediul interfeței grafice. În speță se realizează gestiunea studenților, a profesorilor și administrarea operațiilor curente din cadrul programelor de studiu.

Aplicația poate fi accesată de către utilizatori prin intermediul procesului de autentificare. Tipurile de utilizatori sunt studenți, profesori, administratori. Utilizatorul își poate vedea datele personale (CNP, nume, prenume, adresa, număr de telefon, email, cont IBAN, numărul de contract) în urma accesării sistemului informatic. De asemenea, utilizatorii se pot deautentifica. Utilizatorii de tip administrator pot adăuga, modifica și șterge informații despre utilizatori. Există și rolul de super-administrator care operează exclusiv asupra utilizatorilor de tip administrator, putând modifica datele acestora.

Administratorii pot realiza căutarea utilizatorilor după nume și îi pot filtra după tip, pot asigna profesorii la cursuri și pot face căutare după numele cursului. La căutarea unui curs se afișează inclusiv numele profesorilor de la acel curs și un buton care permite vizualizarea tuturor studenților înscriși la cursul respectiv.

Utilizatorul de tip profesor are următoarele informații suplimentare pe care le poate accesa: cursurile predate, numărul minim și numărul maxim de ore pe care le poate preda și departamentul din care face parte.

Pe de altă parte, utilizatorul de tip student are următoarele informații suplimentare: anul de studiu și numărul de ore pe care trebuie să le susțină.

O materie poate fi predată de mai mulți profesori și fiecare materie are una sau mai multe activități didactice. Studenții se pot înscrie la cursuri și sunt evaluați cu note pentru fiecare activitate didactică. Își pot vedea media finală pe baza notelor, calculată conform formulei stabilite de profesor.

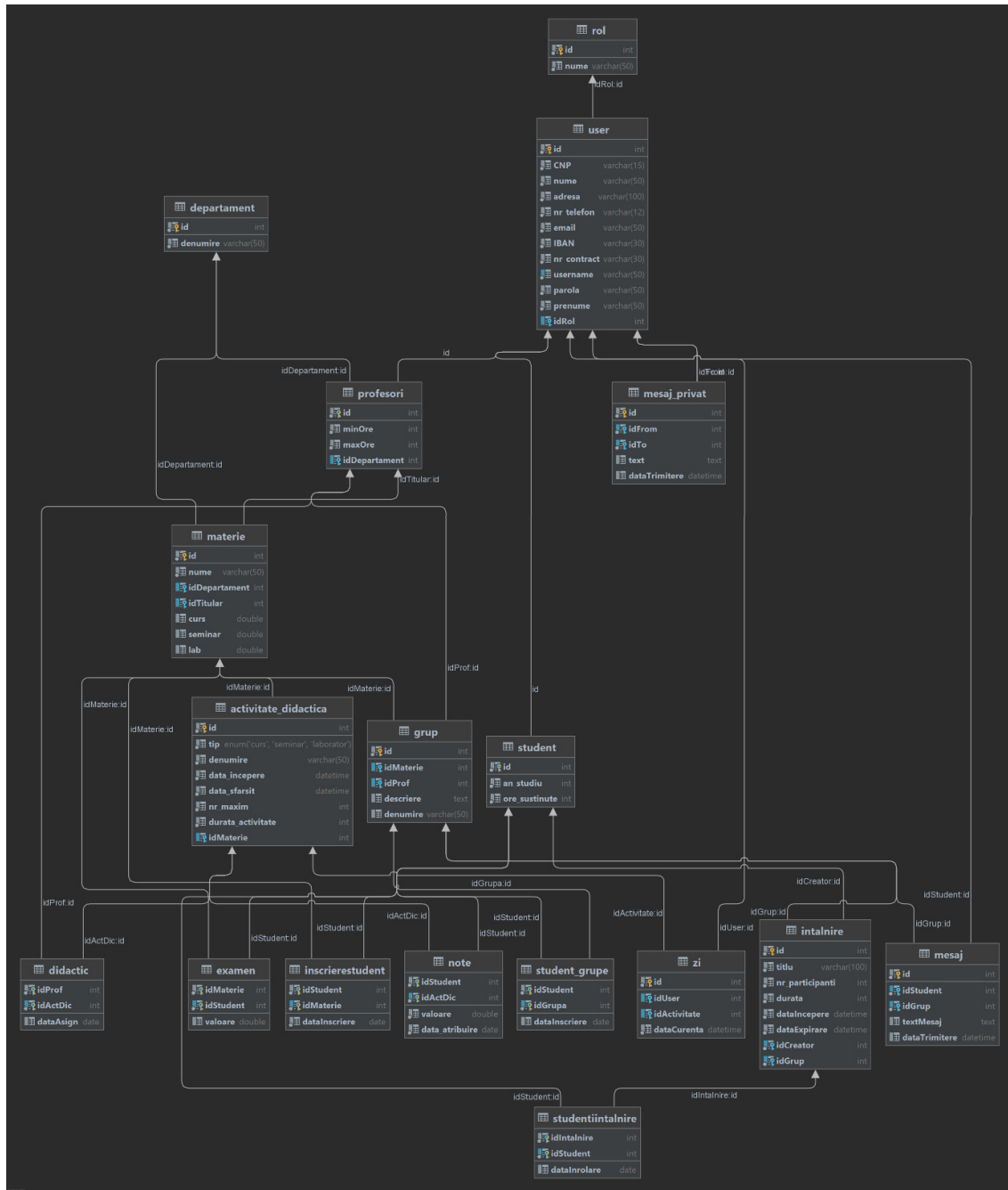
O activitate se desfășoară între două date. Profesorul asignat la o materie este asignat automat și la activitatea de tip curs. De asemenea, trebuie făcută și asignarea de profesori la celelalte tipuri de activități didactice de la materia respectivă. Profesorul poate programa ulterior activitățile într-un calendar. Profesorii pot accesa un catalog unde pot filtra studenții după cursuri și le pot adăuga note. Catalogele pot fi descărcate sub formă de fișier.

La logare, studenții și profesorii pot să își vada activitățile din ziua curentă sau pot accesa o pagină cu toate activitățile la care sunt asignați / înscriși. Aceste liste pot fi descărcate din sistem sub formă unor fișiere.

Studenții se pot înscrie la cursuri, pot renunța la cursuri și își pot vedea notele. Studentul alege activitățile la care vrea să participe și poate participa la ele doar dacă mai sunt locuri sau nu există o suprapunere cu o altă activitate.

Studenții au posibilitatea de a crea un grup de studiu pentru o anumită materie. Ei sunt înscriși automat în grup și pot vedea membrii grupului și pot să lase mesaje. Pe grup, studenții pot savada membrii grupului și să lase mesaje. Un student poate adăuga activități în grup, care vor avea un număr minim de participanți și o perioadă în care ceilalți pot anunța participarea. Dacă numărul minim nu este atins până la încheierea perioadei respective, activitatea se anulează iar studenții înscriși primesc un mesaj de informare.

2. Proiectare conceptuala a bazei de date



3. Solutia de transformare in relational

In modelul de date pe care l-am realizat, se regasesc 5 relati de tipul Many-to-many pe care le-am implementat prin adaugarea cate unui tabel intermediar, dupa cum urmeaza:

- Intre tabela „student” si tabela „materie” am adaugat tabela „inscrierestudent” care are o cheie primara compusa din cele doua chei straine: „idStudent” si „idMaterie”;
- Intre tabela „student” si tabela „activitate_didactica” am adaugat tabela „note” care are o cheie primara compusa din cele doua chei straine: „idStudent” si „idActDic”;
- Intre tabela „profesori” si tabela „activitate_didactica” am adaugat tabela „didactic” care are o cheie primara compusa din cele doua chei straine: „idProf” si „idActDic”;
- Intre tabela „student” si tabela „grup” am adaugat tabela „student_grupe” care are o cheie primara compusa din cele doua chei straine: „idStudent” si „idGrupa”;
- Intre tabela „intalnire” si tabela „student” am adaugat tabela „studentiintalnire” care are o cheie primara compusa din cele doua chei straine: „idIntalnire” si „idStudent”;

4. Descrierea bazei de date relationale

Dupa cum se poate observa si la punctul 2, in cadrul aplicatiei noastre am folosit 18 tabele pentru a retine toate datele necesare in baza noastra de date:

1. Rol(id, nume) – este folosit pentru a retine rolurile pe care le poate avea un utilizator: profesor, student, admin, super admin;
2. User(id, CNP, nume, prenume, adresa, nr_telefon, email, IBAN, nr_contract, username, parola, **idRol**) – tabelul continut toti utilizatorii inregistrati.
3. Profesori(id, minOre, maxOre, **idDepartament**) – tabela este o specializare a tablei „user” ce contine userii ce au rolul de profesor. A fost necesar sa cream acest tabel, deoarece profesorii au nevoie de cele 3 attribute suplimentare;
4. Student(id, ore_sustinute, an_studiu) - tabela este o specializare a tablei „user” ce contine userii ce au rolul de student. A fost necesar sa cream acest tabel, deoarece profesorii au nevoie de cele 2 attribute suplimentare;
5. Departament(id, denumire) – contine cele 3 departamente posibile: automatica, calculatoare si matematica;
6. Materie(id, nume, **idDepartament**, **idTitular**, curs, seminar, lab) – tabela contine materiile care sunt predate in cadrul facultatii. Attributele „curs”, „seminar” si „lab” sunt folosite pentru stocarea ponderilor de calcul a mediei finale la materia respectiva;
7. Activitate_didactica(id, tip, denumire, data_incepe, data_sfarsit, nr_maxim, durata_activitate, **idMaterie**) – tabelul contine toate activitatile didactice de la toate materiile. Fiecare materie poate sa aiba maxim 3 activitati didactice, si anume: curs, seminar si laborator.
8. Inscrierestudent(**idStudent**, **idMaterie**, dataInscriere) – este o tabela intermediara intre tabelele Student si Materie pentru a rezolva relatia Many-to-many;
9. Didactic(**idProf**, **idActDic**, dataAssign) – este o tabela intermediara intre tabelele Profesori si activitate_didactica pentru a rezolva relatia Many-to-many;
10. Note(**idStudent**, **idActDic**, valoare, data_atribuire) - este o tabela intermediara intre tabelele Student si activitate_didactica pentru a rezolva relatia Many-to-many, dar retine pe deasupra valoarea si data cand s-a facut atribuirea;
11. Examen(**idMaterie**, **idStudent**, valoare) – contine notele finale ale studentilor la fiecare disciplina. Inregistrarile se actualizeaza de fiecare data atunci cand o noua nota este adaugata pentru o activitate didactica, prin intermediul unui trigger;
12. Zi(id, **idUser**, **idActivitate**, dataCurenta) – tabela este folosita pentru a stoca toate datele calendaristice in care studentii sau profesori au o anumita activitate didactica, cuprinse intre data de inceput si data de sfarsit a activitatii respective;
13. Grup(id, **idMaterie**, **idProf**, descriere, denumire) - tabela contine grupurile de studiu create de catre elevi, coordonate de catre un profesor.
14. Student_grupe(**idStudent**, **idGrupa**, dataInscriere) - este o tabela intermediara intre tabelele Student si Grup pentru a rezolva relatia Many-to-many, retinand in plus si data la care a avut loc inscrierea;

UNIVERSITATEA TEHNICA “CLUJ-NAPOCA”

15. Intalnire(id, **idCreator**, **idGrup**, titlu, dataIncepere, dataExpirare, durata, nr_participanti) – tabela contine toate intalnirile pe care un student le poate crea in cadrul unui grup.
16. Studentiintalnire(**idIntalnire**, **idStudent**, dataInrolare) - este o tabela intermediara intre tabelele Intalnire si Student pentru a rezolva relatia Many-to-many, retinand in plus si data la care a avut loc inrolarea la intalnire;
17. Mesaj(id, **idStudent**, **idGrup**, textMesaj, dataTrimitere) – tabela contine toate mesajele care au fost trimise in cadrul unui grup;
18. Mesaj_privat(id, **idFrom**, **idTo**, text, dataTrimitere) – tabela contine toate mesajele private trimise intre 2 utilizatori.

5. Argumentarea nivelului de normalizare

In toate tabelele, attributele iau valori atomice si scalare, deci toate relatiile sunt in FN1.

In toate tabelele in care avem cheie primara compusa (din 2 chei), daca exista attribute nonprime atunci ele depind functional de perechea (cheie1,cheie2) dar nu de una din chei. De exemplu, in tabelul studentiintalnire, dataInrolare corespunde datei inrolarii studentului la o intalnire, deci avem doar dependenta functionala idIntalnire,idStudent -> dataInrolare. Acelasi rationament se aplica pentru toate tabelele cu cheie compusa. Astfel, toate relatiile sunt in FN2.

Nu avem dependente tranzitive in tabele. In tabelele cu cheie primara simpla, nu exista dependente tranzitive pentru ca in primul rand nu exista attribute care sa fie determinate in mod unic de id. De exemplu in tabela user, pot exista mai multe conturi pentru o persoana, deci pot exista 2 id-uri diferite cu acelasi CNP, in consecinta nu exista dependenta id-> CNP care ar fi dus intradevar la o dependenta tranzitiva id->CNP->nume. De asemenea in tabelele cu cheie compusa, nu exista attribute nonprime care sa depinda de alt atribut nonprim. In consecinta, relatiile sunt in FN3.

Daca privim tabela user, avem dependenta functionala CNP -> nume. Dar, nu avem si dependenta CNP-> username,parola deoarece o persoana poate sa aiba mai multe conturi. In consecinta, CNP nu este supercheie. Dar faptul ca CNP nu este o supercheie si este termen stang al unei dependente functionale, ne spune ca tabela user nu este in Boyce-Codd. In consecinta, baza de date nu este normalizata Boyce-Codd.

6. Codul in SQL

Userii ce au privilegii asupra bazei de date sunt manager1, user1, admin1. Utilizatorul user1 poate executa exclusiv operatiuni de interogare, manager1 poate executa instructiuni de interogare, inserare, update si de adaugare de indecsi. Admin1 are privilegii maximele.

In cadrul bazei de date, am avut nevoie sa realizam 4 triggere, acestea fiind:

t_zi_alert_profesori se declansaza la inserarea in didactic, adica la atribuirea unui profesor la o activitate. Acesta insereaza in tabela zi remindere despre organizarea unei activitati. Profesorul poate vedea in ce date se tine o anumita activitate didactica, fiecare la o distanta de 7 zile una fata de cealalta. Aceste remindere se insereaza pentru fiecare saptamana, de la data inceperii activitatii si pana la data incheierii acesteia.

t_zi_alert_studenti se declansaza la inserarea in inscrierestudent, adica la inscrierea unui student la o activitate didactica. Acesta insereaza remindere de participare la activitatile de curs, seminar, laborator pentru studenti in tabelul zi. Pentru fiecare dintre aceste 3 tipuri de activitati didactice se insereaza remindere corespunzatoare fiecarei saptamani de la data inceperii si pana la data incheierii activitatii.

t_actualizare_nota_examen_update se declansaza la modificarea unei intrari din tabelul note, adica la inserarea sau modificarea unei note de catre un profesor. Trigger-ul actioneaza asupra tabelului examen. Acesta calculeaza media de examen a studentului la activitatea didactica pentru care i s-a inserat nota. Daca nota nu a fost inserata inca, aceasta e considerata 0. Daca exista deja o nota la examenul respectiv, ea se updateaza, altfel se insereaza.

t_actualizare_nota_examen_insert este exact la fel ca si trigger-ul precedent dar se declansaza la insert pe note.

t_delete_after_delete se declansaza la stergerea din inscrierestudent, adica daca un student renunta la o activitate. Trigger-ul sterge din tabelele note, examen, zi, toate informatiile ce leaga studentul de activitatea respectiva.

7. Functionalitatea aplicatiei

În realizarea aplicației, am structurat codul de Java în 4 pachete: ConnectionPackage (care conține clasa MyConnection și Validator), MainPackage (ce conține clasa Conversie și Main), GUIPackage (care conține clasele utilizate pentru interfața grafică) și ModelsPackage (care este alcătuit din clasele model pentru tabelele din baza de date și câteva enum-uri).

7.1. ConnectionPackage

Acest pachet conține 2 clase: MyConnection, care se ocupă cu legătura dintre aplicația Java și baza noastră de date MySQL, și Validator, care are rolul de a valida datele ce urmează să fie introduse în baza de date.

Clasa de MyConnection are un rol foarte important pentru aplicația noastră. În această clasă se află atât reînnoirea conexiunii cu baza de date, cât și multe metode statice ce sunt folosite pentru apelări de proceduri în SQL pe baza de date.

- Metode de insert:
 - saveUser, apelează procedura sp_InsertUser, ce înserează un user în tabela corespunzătoare;
 - saveStudent, apelează metoda saveUser, apoi procedura sp_InsertStudent care înserează datele în tabela student;
 - saveProfesor, apelează metoda saveUser, apoi procedura sp_InsertProfesor care înserează datele în tabela profesori;
 - saveFormulaMaterie, apelează procedura p_save_formula;
 - saveMesaj, apelează procedura p_insert_mesaj_privat;
 - saveActivitate, apelează procedura p_insert_activitate_didactica;
 - saveNota, apelează procedura p_insert_nota;
 - saveGrup, apelează procedura p_insert_grup;
 - saveIntalnire, apelează procedura p_insert_intalnire;
 - saveStudentIntalnire, apelează procedura p_insert_studentiintalnire;
 - inrolareUser, în funcție de rolul user-ului ce se dorește a fi înrolat, se apelează procedura p_insert_didactic dacă este profesor și p_insert_inscrierestudent dacă este student;
 - inrolareStudentGrup, apelează procedura p_insert_student_grupe
- Metode de update:
 - updateUser, apelează metoda p_update_user;
 - updateStudent, apelează metoda p_update_student;
 - updateProfesor, apelează metoda p_update_profesori

UNIVERSITATEA TEHNICA “CLUJ-NAPOCA”

- Metode de delete:
 - `renuntareStudent`, apeleaza metoda `p_delete_inscrierestudent`;
 - `deleteIntalnire`, apeleaza metoda `p_delete_intalnire`;
 - `renuntareGrup`, apeleaza metoda `p_delete_studenti_intalnire`;
 - `validareIntalniri`, apeleaza metoda `getAllMeetings`, apoi valideaza aceste intalniri, iar daca gaseste vreo intalnire ce este expirata, apeleaza metodele `saveMesaj` pentru salvarea mesajului cum ca meeting-ul a expirat si `deleteIntalnire` pentru a sterge aceasta intalnire.
- Metode ce lucreaza cu fisiere:
 - `descarcareProgram` se creeaza un fisier denumit „program.txt” si se scriu date despre o lista de activitati in acesta;
 - `descarcareProgramAll` se creeaza un fisier denumit „program_tot.txt” si se scriu date despre o lista de activitati in acesta;
 - `descarcareLista`, se creeaza un fisier denumit „activitati.txt” si se scriu date despre o lista de activitati in acesta
- Metode ce executa interogari:
 - `findUser`, executa un query pentru a returna user-ul cu un anumit username. Metoda este folosita preponderant la verificarea daca un user exista in baza de date si se poate loga;
 - `findStudent`, executa un query pentru a returna studentul cu un anumit username;
 - `findProfesor`, executa un query pentru a returna profesorul cu un anumit username
 - `getIdDepartament`, executa un query pentru a returna id-ul unui departament in functie de denumirea departamentului.
 - `getIdUser`, executa un query pentru a returna id-ul unui user in functie de username-ul unui user
 - `getIdRol`, executa un query pentru a returna id-ul unui rol, in functie de denumirea unui rol
 - `getIdGrup`, apeleaza procedura `p_get_id_grup`, ce returneaza id-ul unui grup in functie de profesorul coordinator si cursul pentru care e cursul
 - `getIdIntalnire`, apeleaza procedura `p_get_id_intalnire`, ce returneaza id-ul unei intalniri in functie de toate attributele unei intalniri.
 - `getAllFromDepartament`, executa un query ce returneaza toate campurile din tabela departament
 - `getAllFromRol`, executa un query ce returneaza toate campurile din tabela rol

UNIVERSITATEA TEHNICA “CLUJ-NAPOCA”

- getAllFromUser, executa un query ce returneaza toate campurile din tabela user
- getAllProfesori, apeleaza procedura p_afisare_profi, ce returneaza toate campurile din tabela profesori
- getAllGrups, apeleaza procedura p_afisare_grupuri, ce returneaza grupurile din care face parte un anumit utilizator.
- getAllActivitatiDidactice, apeleaza procedura p_afisare_toate_activitatile_didactice, ce returneaza toate activitatile didactice
- getAllMeetings, apeleaza procedura p_get_meetings, ce returneaza toate intalnirile
- getAllMaterii, avem doua proceduri cu acest nume, una nu primeste niciun parametru, apeland procedura p_afisare_toate_materiile, pentru a returna toate materiile ce exista in tabela materie, iar a doua primeste ca parametru un user si returneaza toate materiile la care este inrolat acesta, apeland procedura p_afisare_materii

- getNumOfStudByActivitate, apeleaza procedura p_num_activitati, ce returneaza numarul de student inscrisi la o activitate, identificata dupa id
- getNumOfStudByMaterie, apeleaza procedura p_num_materii, ce returneaza numarul de student inscrisi la o materie, identificata dupa id
- getNumOfStudByIntalnire, apeleaza procedura p_num_stud_intalnire, ce returneaza numarul de student inrolati la o intalnire, identificata dupa id

- getAllActivitatiDidacticeFromUser, apeleaza procedura p_afisare_activitati_didactice, ce returneaza activitatile didactice ale unui utilizator, identificat dupa username
- getAllActivitatiDidacticeByIdMaterie, apeleaza procedura p_afisare_activitate_didactica_by_id_materie, ce returneaza toate activitatile didactice ale unei materii, identificate dupa id
- getActivitatiByDate, primeste ca parametrii un user si o data, daca data este diferita de null, se apeleaza procedura p_getActivitatiByDate, ce returneaza toate activitatile didactice de la o anumita data, iar daca data trimisa este nula se va afela procedura p_getActivitatiFromZi, ce returneaza toate activitatile din tabela zi ale user-ului trimis (identificat dupa username)
- getMaterieById, apeleaza procedura get_materie_by_id, ce returneaza materia cu un anumit id, trimis ca parametru
- getAllMateriiForEnrol, apeleaza procedura p_afisare_materii_not_stud, ce returneaza toate materiile la care se poate inrola un anumit student trimis ca parametru, identificat dupa username

- getNoteStudentByIdActivitate, apeleaza procedura p_afisare_note_student_by_id_activitate, ce returneaza notele unui student trimis ca parametru la o activitate trimisa ca parametru

UNIVERSITATEA TEHNICA “CLUJ-NAPOCA”

- `getMedieStudentByIdMaterie`, apeleaza procedura `p_afisare_medie_student_by_id_materie`, ce returneaza notele unui student din tabela examen, la o anumita materie trimisa ca parametru, identificata dupa id
- `getMeetingByGrup`, apeleaza procedura `p_afisare_intalniri_from_grup`, ce returneaza intalnirile dintr-un grup, identificat dupa id
- `getDateFromUser`, ce apeleaza procedura `p_get_date_by_user`, ce returneaza toate datele din tabela zi in care are un user activitati
- `getGrupByUser`, apeleaza procedura `p_get_grup_by_user`, ce returneaza grupurile la care este inscris un anumit user, identificat dupa username
- `getMesajeFromUser`, apeleaza procedura `get_all_mesaje_by_id`, ce returneaza toate mesajele dintre 2 utilizatori, sau dintr-un grup
- `getConversationsFromUser`, apeleaza procedura `p_afisare_conversatii`, ce returneaza toate conversatiile pe care le-a avut un user, identificat dupa username (conversatiile insemnand, grupuri in care se afla sau utilizatori cu care a comunicat prin mesaj)
- `getUsersByTip`, apeleaza procedura `p_cautare_dupa_tip`, ce returneaza toti utilizatorii cu un anumit tip, aceasta metoda este folosita pentru filtrarea utilizatorilor dupa tip
- `getStudentByActivitateDidactica`, apeleaza procedura `p_afisare_studenti_by_activitate_didactica`, ce returneaza studentii ce participa la o anumita activitate didactica
- `getUserById`, apeleaza procedura `get_profesor_by_id`, ce returneaza profesorul cu un anumit id trimis ca parametru
- `getAllUsersFromGrup`, ce apeleaza procedura `p_afisare_membrii_grup`, ce returneaza toti membrii dintr-un anumit grup, identificat dupa id
- `getSugestii`, apeleaza procedura `p_get_sugestii`, ce returneaza sugestii pentru inrolarea la grup

7.2. MainPackage

Acest pachet contine 2 clase: Clasa Main care are rolul de a porni executia programului si clasa Conversie ce continue metode statice pentru a converti enum-urile create de noi in stringuri sau invers: `convDepartamentToString`, `convRolToString`, `convStringToDepartament`, `convStringToRol`, `convStringToActivitateDidactica`, `convActivitateDidacticaToString`.

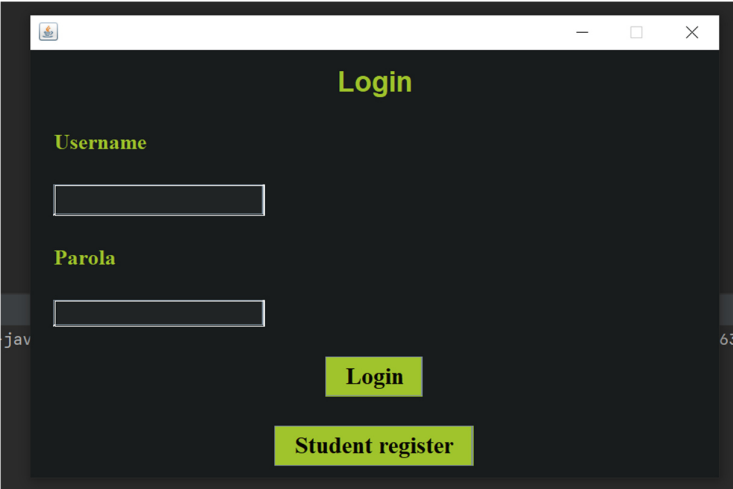
De asemenea, clasa Conversie are 2 metode pentru conversia din string in DateModel: `convStringToDateModel`, `convDateModelToString`.

Aceste conversii sunt necesare datorita faptului ca in baza de date se pot stoca doar string-uri, neputand stoca enum-uri.

7.3. GUIPackage

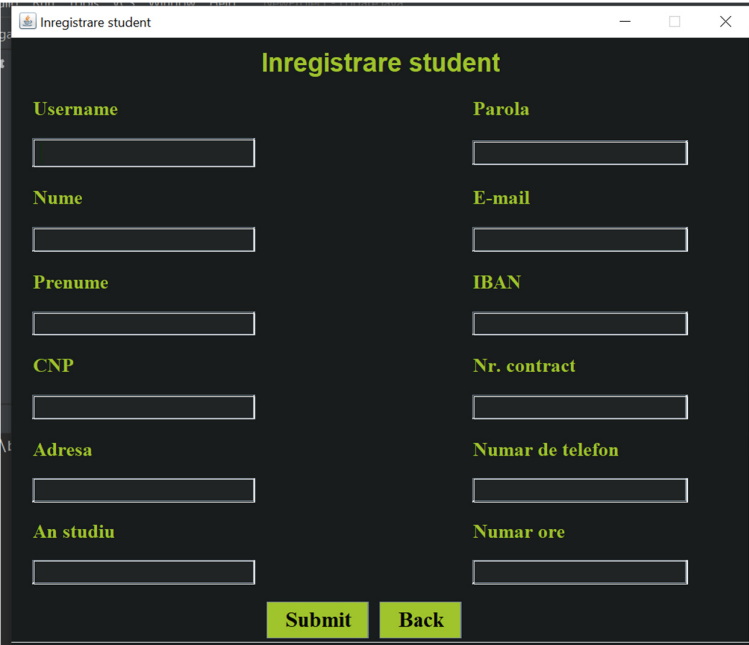
Ferestrele ce urmeaza sa fie afisate in cadrul aplicatiei sunt:

➤ Logare

A screenshot of a Java Swing window titled "Login". The window has a dark background. It contains two text input fields: the first is labeled "Username" and the second is labeled "Parola" (Password). Below the password field is a green button labeled "Login". At the bottom center of the window is another green button labeled "Student register". The window has standard OS window controls (minimize, maximize, close) in the title bar.

Aceasta fereastră apare în momentul rularii aplicației și are ca scop fie logarea unui utilizator, fie înregistrarea unui student. Butonul de „Student register” deschide fereastra „RegisterStudent”, iar la apăsarea butonului de Login se creează un UserModel cu username-ul și parola introduse ce va fi trimis către o metodă din clasa MyConnection pentru a verifica dacă user-ul este deja înregistrat. În caz afirmativ, se deschide fereastra de HomePage, iar în caz contrar se afișează mesajul: "Date incorecte !".

➤ RegisterStudent

A screenshot of a Java Swing window titled "Inregistrare student". The window has a dark background. It contains ten text input fields arranged in two columns. The left column labels are "Username", "Nume", "Prenume", "CNP", and "Adresa". The right column labels are "Parola", "E-mail", "IBAN", "Nr. contract", "Numar de telefon", and "Numar ore". At the bottom of the window are two green buttons: "Submit" and "Back". The window has standard OS window controls in the title bar.

Fereastra este utilizată pentru înregistrarea unui utilizator de tip Student. Butonul de Back ne redirecționează către fereastra de Login, iar la apăsarea butonului de Submit se creează un StudentModel cu datele introduse ce se trimite spre clasa Validator spre a fi validate datele.

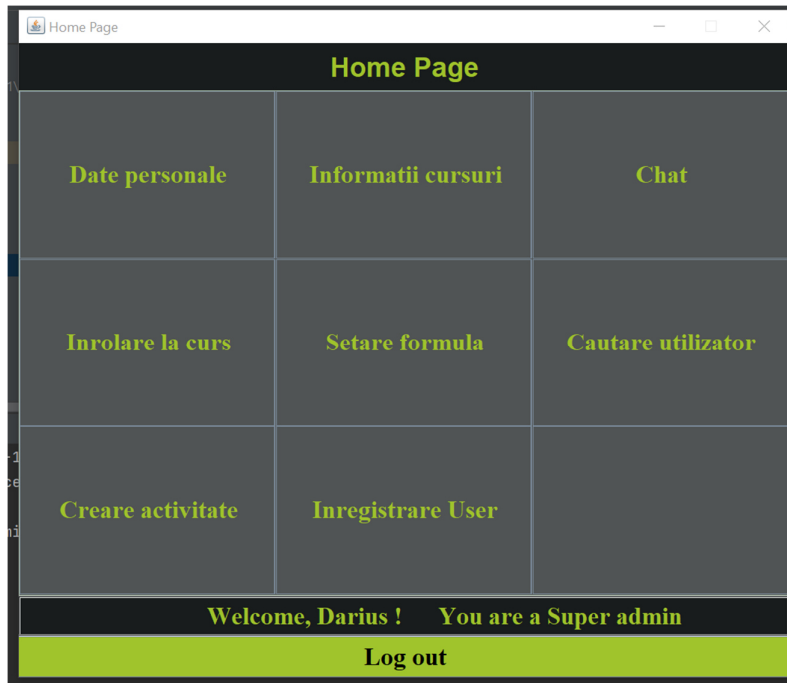
UNIVERSITATEA TEHNICA “CLUJ-NAPOCA”

Daca datele nu sunt valide, se afiseaza un mesaj de eroare, iar daca sunt valide, modelul creat este trimis la o metoda din clasa MyConnection pentru salvarea in baza de date.

➤ HomePage

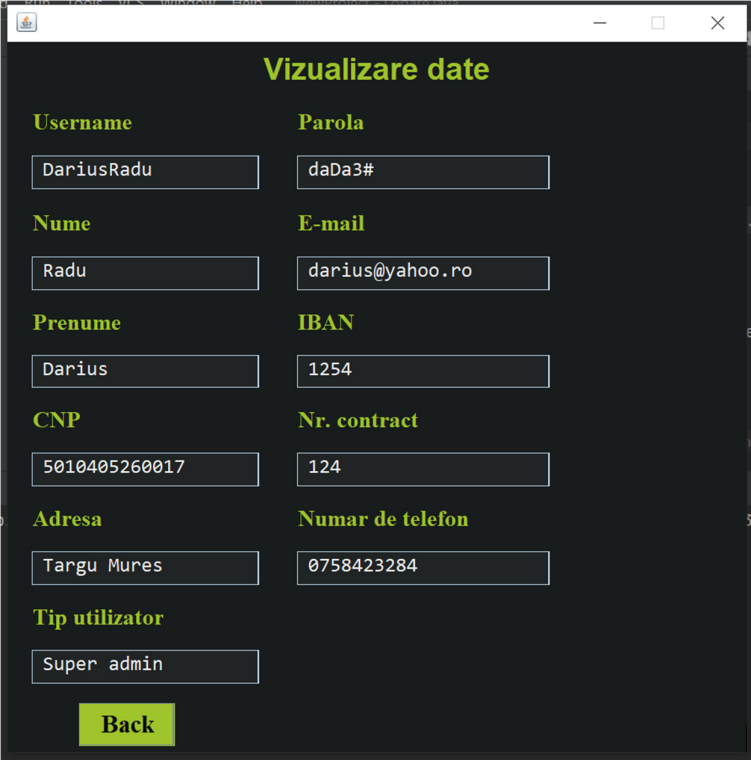
Aceasta fereastră poate arata in 3 moduri diferite, in functie de rolul utilizatorului logat.

➤ Pentru admini sau super-admini



Butonul de „Logout” ne intoarce la fereastra de logare. Odata logati, se afiseaza un mesaj de bun venit si rolul utilizatorului logat. In cazul adminilor, optiunile din meniu sunt:

- Date personale: va deschide o fereastră de „VizualizareDate” ;



The screenshot shows a web application window with a dark background and yellow text. The title is "Vizualizare date". Below the title, there are two columns of labels and input fields. The labels are in yellow, and the input fields contain user data. At the bottom, there is a yellow "Back" button.

Username	Parola
DariusRadu	daDa3#
Nume	E-mail
Radu	darius@yahoo.ro
Prenume	IBAN
Darius	1254
CNP	Nr. contract
5010405260017	124
Adresa	Numar de telefon
Targu Mures	0758423284
Tip utilizator	
Super admin	

Back

Aceasta fereastră este utilizată pentru afișarea datelor de utilizator.

- Informații cursuri: va deschide o fereastră de „VizualizareCursuri”;

Această fereastră este multifuncțională și vom avea un paragraf separat pentru ea.

- Chat: va deschide o fereastră de „PrivatChat”;

Această fereastră este multifuncțională și vom avea un paragraf separat pentru ea.

- Înrolare la curs: va deschide o fereastră de „VizualizareCursuri”;

Opțiunea de înrolare curs pentru utilizatorul User va selecta un student/profesor pe care îl va înrola la o materie/activitate didactică dacă nu există suprapuneri de date.

- Setare formulă: va deschide o fereastră de „VizualizareCursuri”;

În cadrul acestei opțiuni, când selectăm o materie se va deschide o fereastră „SetareFormula”.

Aici, utilizatorul seteaza ponderile pentru fiecare activitate didactica din cadrul materiei. Vor fi afisate campurile Curs, Seminar, Laborator in functie de activitatile de care dispune materia respectiva. La apasarea butonului de „Save” sunt validate, iar daca sunt corecte, datele sunt trimise la o metoda din MyConnection pentru salvarea in baza de date.

- Cautare utilizator: va deschide o fereastră de „VizualizareCursuri”;

Aici, sunt incarcati toti userii din aplicatie. Adminul are posibilitatea de a cauta in lista dupa nume, prenume sau filtra dupa tip. La selectarea unuia dintre utilizatori, se va deschide o fereastră de „VizualizareDate” unde sunt incarcate datele utilizatorului selectat. Adminul are posibilitatea de a schimba anumite date prin apasarea butoanelor „Edit”, respectiv „Save”, in cazul in care utilizatorul selectat este student sau profesor, iar un superadmin poate modifica si datele unui admin.

- Creare activitate: va deschide o fereastră de „VizualizareCursuri”;

Aici, sunt incarcate cursurile. La selectarea unei materii se va deschide o fereastră de „AdaugareActivitati”.

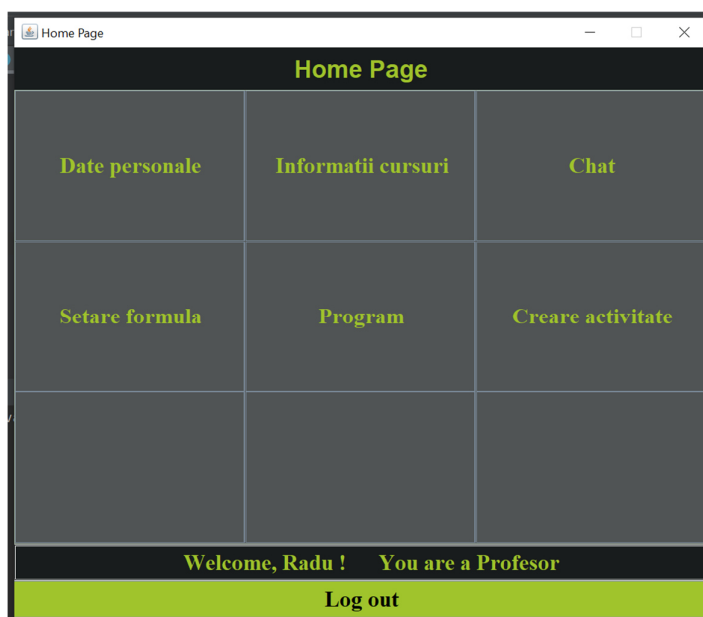
UNIVERSITATEA TEHNICA “CLUJ-NAPOCA”

Aceasta fereastră este folosită pentru adăugarea de activități. La apăsarea butonului Submit, datele introduse sunt validate și sunt trimise către o metodă din MyConnection pentru salvarea în baza de date.

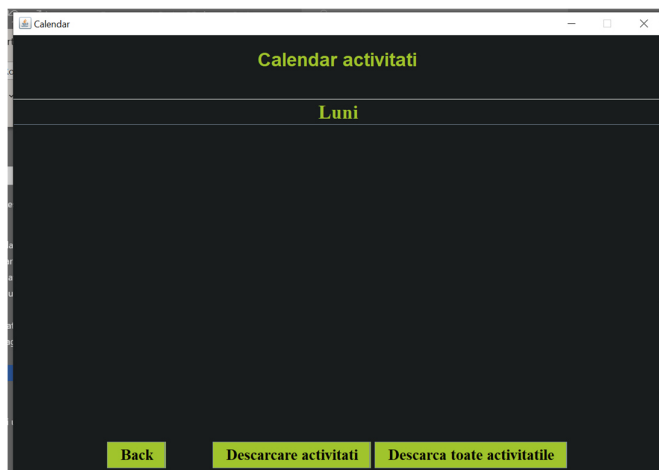
- Inregistrare User: va deschide o fereastră de „RegisterUser”.

Este asemănătoare cu fereastra „RegisterStudent”, doar că în cadrul acestei ferestre există un ComboBox prin care se poate selecta tipul de utilizator. În funcție de acesta apar sau dispar anumite câmpuri ce trebuie completate pentru înregistrarea user-ului.

➤ Pentru profesori



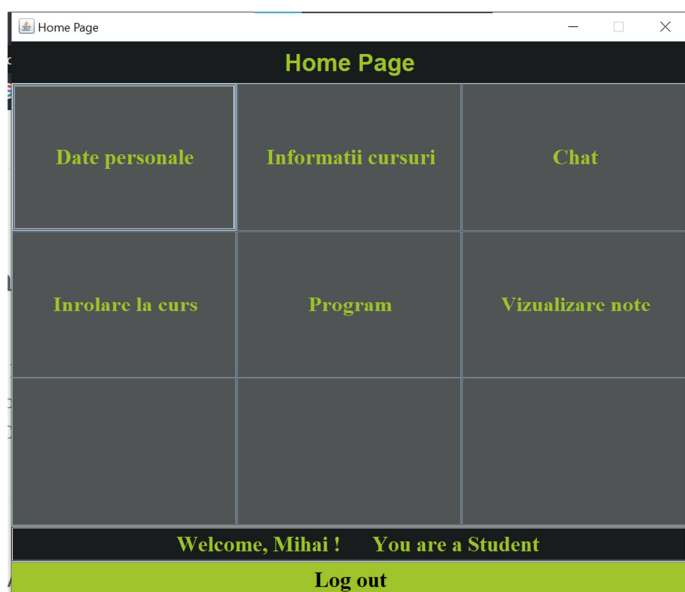
Din această perspectivă, unele opțiuni din meniu sunt identice cu ale adminului. În plus, avem opțiunea de „Program” care va deschide o fereastră „Program”.



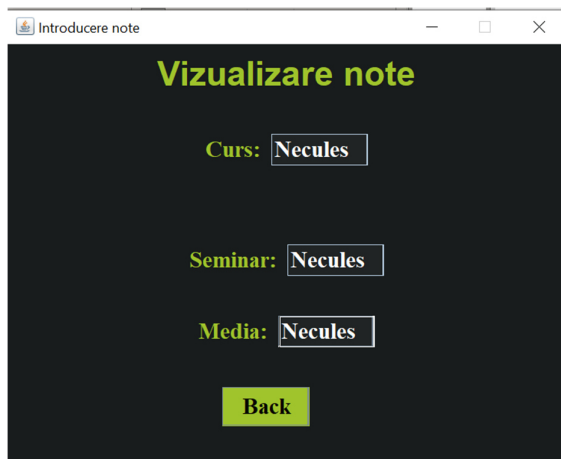
UNIVERSITATEA TEHNICA "CLUJ-NAPOCA"

Aici, sunt incarcate activitatile din ziua curenta. Avem posibilitatea de a descarca activitatile din ziua curenta sau toate activitatile, apasand butoanele „Descarcare activitati”, respectiv „Descarca toate activitatile”.

➤ Pentru student



Exista optiuni comune cu profesori/admini. In plus, fiind Vizualizare note, care deschide o fereastră de tipul „VizualizareCursuri”. Aici, se selecteaza materia la care se doreste a se vedea notele. Se deschide o fereastră de tip „SetareFormula”.



Aici, sunt afisare notele la fiecare activitate ce tine de materia respectiva si media finala. In cazul in care nu avem o nota trecuta se va afisa „Necules”.

- Vizualizare cursuri

Cum am scris mai sus, fereastră „VizualizareCursuri” este o fereastră multifunctionala pe care am folosit-o pentru a incarca date pentru 10 optiuni diferite, prin intermediul unei variabile

UNIVERSITATEA TEHNICA “CLUJ-NAPOCA”

„fereastră”. In toate optiunile exista posibilitatea de a cauta user, curs sau activitatea didactica dupa nume. Optiunile sunt:

1. Pentru incarcarea activitatilor didactice (optiunea Informatii cursuri din HomePage).

La selectarea unei activitati didactice, se afiseaza un mesaj cu informatii despre acea activitate didactica.

Aici, daca utilizatorul este admin/superadmin sau profesor avem un buton de vizualizare studenti ce va afisa studentii inscriși la activitatea selectata. Afisarea se va face deschizand o noua fereastră de tipul „VizualizareCursuri” cu variabila „fereastră” egala cu -1.

Studentii si profesorii pot descarca lista cu activitati ce le-a fost afisata, apasand butonul „Descarca lista activitati”;

2. Pentru incarcarea useriilor (studenti sau profesori) pentru optiunea „Inrolare curs” din HomePage.

Aceasta optiune este diferita in functie de rolul utilizatorului ce a selectat optiunea. In caz ca user-ul este un Admin/Super admin, ne este incarcata o lista cu toti studentii si profesorii. La selectarea unuia dintre ei se deschide o alta fereastră de tip „VizualizareCursuri”, dar cu variabila fereastră = -2, ce va incarca activitatile/cursurile in functie de tipul utilizatorului ce l-am selectat. Dupa ce selectam si o activitate/curs la care sa inrolam user-ul, se vor trimite cele doua modele la o metoda din MyConnection pentru salvarea in baza de date. (in cazul in care student nu are alte activitati ce se suprapun)

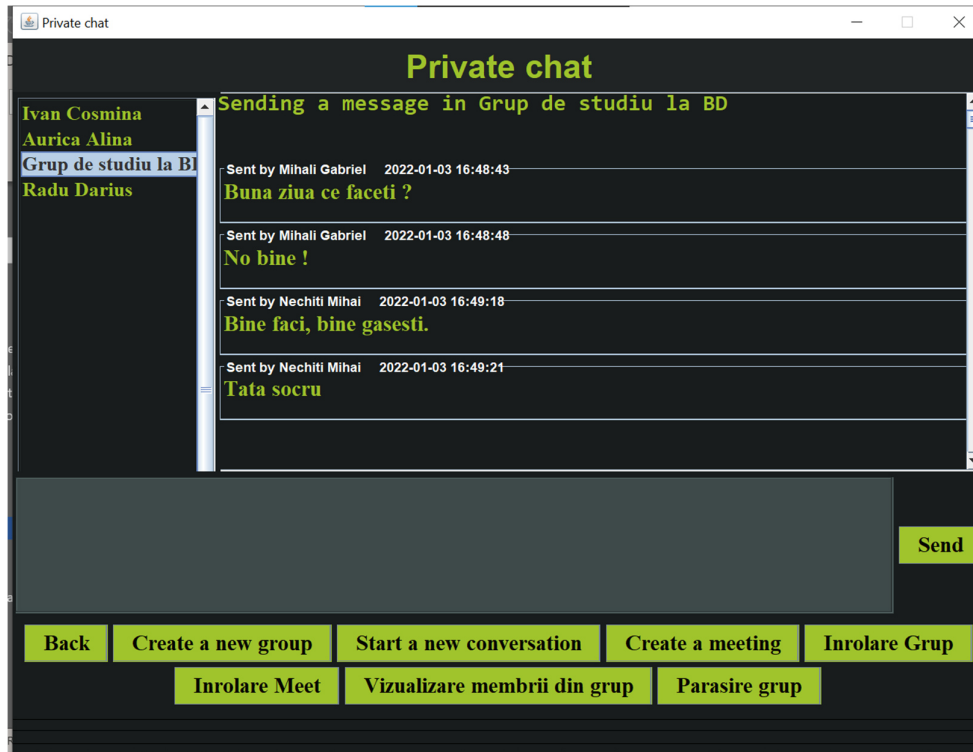
Ca facilitati suplimentare avem posibilitatea de a filtra userii dupa tip.

Daca cel ce a selectat optiunea de inrolare este un student, vor fi incarcate automat doar cursurile la care acesta nu participa, iar ca si in cazul anterior la selectarea unui, se trimite catre o metoda din MyConnection pentru salvarea in baza de date. (in cazul in care student nu are alte activitati ce se suprapun).

3. Pentru setarea formulei. Dupa cum am spus si la fereastră „HomePage”, cand selectam optiunea de „Setare formula”, se deschide o fereastră de tipul „VizualizareCursuri” cu variabila fereastră = 3, unde se incarca cursurile la care se poate seta o formula. In cazul in care utilizatorul ce selecteaza optiunea este profesor, vor fi incarcate doar cursurile la care acesta este titular, iar daca utilizatorul este admin/super admin, se vor afisa toate cursurile existente in baza de date.
4. Pentru cautarea unui utilizator. Aceasta optiune a fost explicata la fereastră de „HomePage”.
5. Pentru vizualizare note. Aceasta optiune a fost explicata la fereastră de „HomePage”.
6. Pentru creare unei activitati. Aceasta optiune a fost explicata la fereastră de „HomePage”.
7. Pentru optiunea de Chat. Aici se va afisa o lista cu toti userii si toate grupurile din care face parte utilizatorul. Odata ce este selectat unul dintre elementele afisate se va deschide o fereastră de tipul „PrivatChat”, ce are ca emitor utilizatorul ce selecteaza elementul si ca receptor persoana selectata sau grupul selectat.

8. Pentru renuntarea unui student la un anumit curs. Aceasta fereastra este incarcata in cazul in care suntem un student si avem deschisa fereastra „VizualizareCursuri” cu variabila „fereastra” = 1 si apasam pe butonul „Renuntare curs”.

- PrivateChat



Aici avem una din cele mai complexe ferestre ce le-am creat in aceasta aplicatie. In primul rand, avem in partea din stanga o lista cu toate conversatiile pe care utilizatorul logat le-a avut. In partea din centru e zona de mesaje, aici sunt incarcate mesajele in functie de destinatarul pe care il selectam noi. In partea de jos avem zona de text si butonul de send (pentru a trimite mesaje), iar si mai jos de ele e zona cu toate meniurile ce le avem in aceasta fereastra.

Butoanele de jos sunt afisate in functie de utilizatorul ce intra in chat si de destinatarul la care doreste sa trimita mesaje.

Optiunile de creare meet, inrolare meet, vizualizare membrii din grup si parasire grup sunt vizibile doar in cazul in care suntem un student si avem deschisa o conversatie intr-un grup. Optiunea de creare grup este disponibila doar pentru studenti, indiferent in ce conversatie se afla.

1. Butonul de send

Cand apasam pe butonul de send, se creaza un „MessagePanel” ce are in interior textul pe care noi il scriem in campul de text. Acest „MessagePanel” are un border, unde se afiseaza si data de trimitere a mesajului si persoana care l-a trimis.

2. Butonul de creare grup

Acest buton va deschide o fereastră de tipul „CreateGroup”.

The screenshot shows a window titled "Create grup". It contains two text input fields labeled "Denumire" and "Descriere". Below these fields are two scrollable lists of suggestions. The left list includes names like Ivan Cosmina, Munteanu Radu, Gavrea Ioan, Rosca Daniela, Oprisa Ciprian, Brehar Raluca, Potolea Rodica, and Iordan Anca. The right list includes roles and names like MSI titular fiind, Gavrea Ioan; BD titular fiind, Ivan Cosmina; MES titular fiind, Munteanu Radu; AF titular fiind, Oprisa Ciprian; MSI titular fiind, Rosca Daniela; POO titular fiind, Brehar Raluca; Programare Logica titular fiind, Potolea Rodica; and RC titular fiind, Oprisa Ciprian. At the bottom of the window are two buttons: "Back" and "Create".

Ce contine campuri pentru introducerea datelor necesare pentru crearea unui grup.

Cand apasam pe butonul de creare, datele introduse vor fi trimise catre clasa de validare, iar in cazul in care sunt valide se trimite catre o metoda din clasa MyConnection pentru salvarea in baza de date. Dupa salvarea in baza de date se deschide o alta fereastră multifunctionala (procedeul este acelasi ca la „VizualizareCursuri”) pentru afisarea sugestiiilor de participanti la acest grup. Fereastră multifunctionala este „InrolareGrup” (cu variabila de control = 4).

In aceasta fereastră ne sunt afisate unele sugestii de studenti ce ar vrea sa participe la acest grup de studiu (in functie de materia pentru care e grupul) si avem optiunea de a trimite mesaj catre toti studentii sugerati, sau in particular doar la unii.

3. Butonul pentru start new conversation

Acest buton ne intoarce la fereastră ce ne afiseaza toti userii si grupurile pentru a selecta un destinatar pentru mesajele noastre.

4. Butonul de creare meeting

Rolul acestui buton este de a crea o intalnire. Cand apasam butonul ni se deschide o fereastră de tipul „AdaugareActivitati” (variabila de control = 2), unde se adauga datele pentru intalnire si la apasarea butonului de creare, datele sunt validate apoi trimise catre o metoda ce le va salva in baza de date.

Cand se deschide fereastră pentru crearea unei intalniri, ne apare un mesaj in care ne sunt sugerate cateva date in care sa punem intalnirea sa nu ni se suprapuna cu activitatile utilizatorilor din grup.

5. Butonul de inrolare la grup

Se deschide o fereastră de tipul „InrolareGrup”.



Aici selectam grupul la care dorim sa ne inrolam, iar la apasarea butonului submit vom fi inrolati.

6. Butonul de inrolare meeting

Se comporta asemenea cu cel de inrolare la grup.

7. Butonul de vizualizare membrii

Va deschide o fereastră de tipul „InrolareGrup” (cu variabila de control = 3).

8. Butonul de parasire grup

La apasarea acestui buton se va apela o metoda din MySqlConnection, ce va sterge inregistrarea studentului din grupul a carui conversatie se afla.

7.4. ModelsPackage

In acest pachet, avem enum-uri si clase-Model. Enum-urile sunt: Departament, Rol, ActivitateDidactica. Am ales sa folosim enum-uri in loc de string-uri pentru evitarea unor erori umane de scriere ce ar putea compromite functionalitatile programului.

Folosind enum-uri, am scapat de posibilitatea aparitiei acestei erori, dar am crescut complexitatea, deoarece avem nevoie de clasa de conversii.

Clasele-model utilizate in program sunt:

- UserModel, ce contine date despre utilizatori preluate din baza de date;
- ProfesorModel si StudentModel, ce sunt subclase ale clasei UserModel;
- RolModel, contine rolurile din baza de date;
- DepartamentModel, contine departamentele din baza de date;
- CursModel, ce contine datele din tabela „Materie”;
- ActivitateDidacticaModel, contine datele din tabela Activitate_didactica;

- GrupModel, contine datele din tabela Grup;
- ConversatieModel, ce este o superclasa pentru clasele UserModel si GrupModel. Am creat aceasta clasa vida pentru a realiza in cadrul functionalitatii de Chat anumite manipulări de obiecte, atat de tip User, cat si de tip Grup;
- MesajPrivatModel, contine datele din tabelele mesaj si mesaj_privat;
- IntalnireModel, contine datele din tabela Intalnire.

Variabilele instante din anumite clase nu coincide in totalitate cu campurile din tabelele din baza de date, deoarece in JAVA avem posibilitatea de a folosi liste.

8. Manual de utilizare

O data pornita aplicatia, se deschide fereastra de login. Daca avem cont, ne logam, in caz contrar, avem posibilitatea de a ne crea un cont de student. Daca ne-am logat, se deschide fereastra de home care are un meniu cu mai multe optiuni in functie de tipul de utilizator.

1. Vizualizare date, ne permite sa vizualizam datele noastre personale fara a le putea modifica.
2. Informatii cursuri, ne afisaza activitatile didactice la care suntem inscrisi, daca suntem student sau profesor, respective toate activitatile existente daca suntem admin/superadmin. Daca suntem profesor/admin/superadmin avem optiunea de a vizualiza studentii inscrisi la o anumita activitate, putand sa le atribuim note, iar daca suntem student putem renunta la un curs la care suntem inscrisi. De asemenea, daca suntem student sau profesor putem descarca lista cu activitatile noastre.
3. Chat, unde putem trimite mesaje la oricare user din aplicatie sau in grupurile in care suntem inscrisi. In grupuri, putem crea intalniri, ne putem inrola la intalniri din cadrul grupului, vizualiza membrii grupului sau parasi grupul. De asemenea, putem crea noi un grup.
4. Inrolare curs, daca suntem student, ne putem inrola la un curs, iar daca suntem admin sau superadmin putem asigna un student/profesor la un curs/activitate didactica.
5. Setare formula, daca suntem profesor/admin/superadmin putem seta formula de calcul pentru o materie.
6. Vizualizare note, daca suntem student putem sa ne vizualizam notele la orice curs la care suntem inscrisi.
7. Program, daca suntem student/profesor putem vizualiza activitatile din ziua curenta si descarca atat activitatile din ziua curenta cat si cele din viitor.
8. Cautare utilizator, daca suntem admin/superadmin, putem sa cautam utilizatorul dupa nume sau sa filtram utilizatorii dupa tip. De asemenea daca selectam pe unul dintre ei putem sa ii vizualizam datele personale si chiar sa le modificam.
9. Inregistrare user, daca suntem admin/superadmin putem inregistra orice tip de user cu prioritate mai mica decat noi.

In cadrul aplicatiei avem butoane de back ce ne trimit la pagina precedenta. In orice fereastra ce se deschide putem cauta useri/activitati/cursuri dupa nume cu ajutorul barii de cautare.

10. Concluzii si dezvoltari ulterioare

Aplicatia are multe functionalitati, un design placut la ochi ce faciliteaza utilizarea cat mai usoara in cadrul unei institutii de invatamant. Aplicatia poate fi imbunatatita in primul rand hostand baza de date pe un server cee ace o va face utilizabila de mai multi useri in timp real (acest lucru va necesita o abordare deosebita a concurentei).

Se pot adauga noi functionalitati precum trimiterea de imagini in grup, intalnirile sa fie video/audio, posibilitatea de customizare a interfetei grafice, criptarea parolelor si mesajelor, posibilitatea de recuperare a parolei prin e-mail.