**PROIECT**

la disciplina

Baze de Date

**Platformă Studiu**

**Proiect realizat de:**

**Oșvat Sergiu-Șerban, CTI ro, grupa 30224**

**Mocoi Ioan-Victor, CTI ro, grupa 30224**

**An academic : 2022 - 2023**

PROIECT de SEMESTRU

Disciplina  : Baze de Date

Coordonator: prof. ing. Cosmina IVAN

Data: 09.01.2023

Cuprins

1. **Introducere**
   * Argumentare, scop si obiective
2. **Specificatii de proiect**
   * Ipoteze specifice domeniului ales pentru proiect
   * Organizarea structurata a cerintelor
   * Determinarea si caracterizarea profilurilor de utilizatori

(admin, profesor, student)

1. **Modelul de date**
   * Entitatile si atributele lor (implementarea)
   * Diagrama EER/UML pentru modelul de date complet
   * Normalizarea datelor
2. **Descrierea functionala a modulelor**
3. **Bonusuri**
4. **Concluzii**
5. **Bibliografie**

**1. Introducere**

Activitatea de desfasurare a programului unei facultati este complexa si orice schimbare care intervine in cadrul acestei activitati necesita o strictete si o atentie sporita avand in vedere ca la mijloc se afla educatia studentilor. Unul din subiectele discutate in cadrul operatiunilor care au loc intr-o facultate este modul de organizare a desfasurari activitatilor didactice, care necesita rapiditate si exactitate.

**2. Specificatii de proiect**

Datorita dorintei noastre de a crea o baza de date care sa fie utila studentilor si profesorilor unei facultati, aceasta trebuie sa fie usor de utilizat, accesibila din punct de vedere a manipularii datelor, si sa nu permita datelor eronate sa fie introduse.

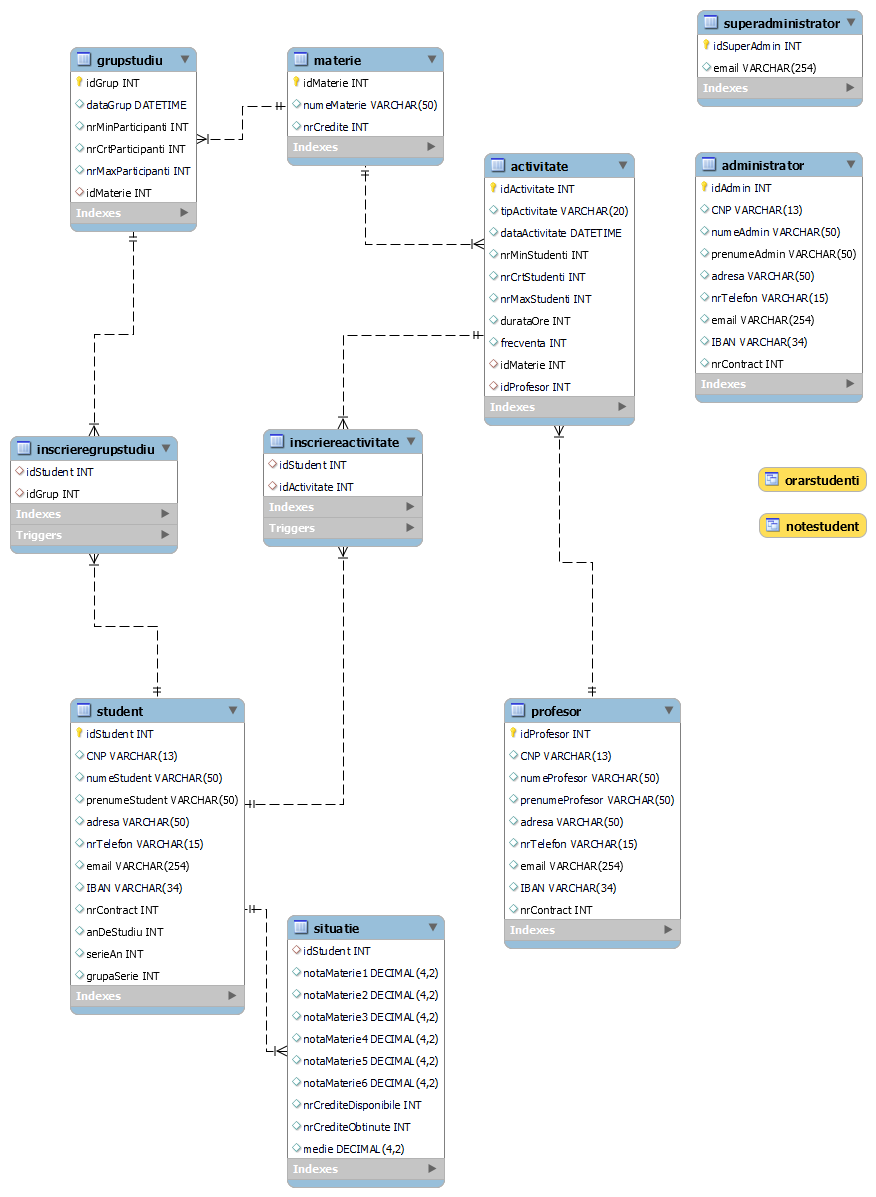
Administratorii pot modifica activitatile intretinute de profesori (Curs, Seminar, Laborator) si inscrierile studentilor la aceste activitati.

Profesorii (si studentii) pot sa isi vada datele personale, propriile activitati si datele despre acestea, si pot nota studentii in functie de procentajele alese pentru fiecare activitate a unei materii (Curs, Seminar, Laborator).

Studentii se pot inscrie la activitati si pot crea grupuri de studiu, pe care doar ei le pot modifica.

**3. Modelul de date**

O prima schema pentru baza e date este reprezentata in diagrama de mai jos:



Entitatea **superAdministrator** are aceleasi permisiuni pe care le are si entitatea **administrator,** atat ca **superAdministratorul** poate modifica si vizualiza datele despre **administrator**

create table if not exists superAdministrator(

idSuperAdmin int primary key unique not null auto\_increment,

email varchar(254)

);

Entitatea **administrator** este entitatea care are control asupra intregii baze de date, si contine datele fiecarui administrator

create table if not exists administrator(

idAdmin int primary key unique not null auto\_increment,

CNP varchar(13),

numeAdmin varchar(50),

prenumeAdmin varchar(50),

adresa varchar(50),

nrTelefon varchar(15),

email varchar(254),

IBAN varchar(34),

nrContract int

);

Entitatea **profesor** contine informatii privind toti profesorii, mai multi profesori pot preda mai multe activitati cat timp acestea nu se suprapun sau nu sunt planificate in trecut

create table if not exists profesor(

idProfesor int primary key unique not null auto\_increment,

CNP varchar(13),

numeProfesor varchar(50),

prenumeProfesor varchar(50),

adresa varchar(50),

nrTelefon varchar(15),

email varchar(254),

IBAN varchar(34),

nrContract int

);

Entitatea **student** contine informatii privind toti studentii, acelasi student poate participa la mai multe activitati si mai multe grupuri de studii cat timp acestea nu se suprapun sau nu sunt programate in trecut si se pot inscrie sau pot renunta la acestea

create table if not exists student(

idStudent int primary key unique not null auto\_increment,

CNP varchar(13),

numeStudent varchar(50),

prenumeStudent varchar(50),

adresa varchar(50),

nrTelefon varchar(15),

email varchar(254),

IBAN varchar(34),

nrContract int,

anDeStudiu int,

serieAn int,

grupaSerie int

);

Entitatea **materie** contine informatii privind toate materiile care sunt predate, materii noi pot fi adaugate doar de administratori si mai multi profesori pot preda aceeasi materie

create table if not exists materie(

idMaterie int primary key unique not null auto\_increment,

numeMaterie varchar(50),

nrCredite int

);

Entitatea **activitate** contine informatii privind activitatile care se desfasoara in momentul actual, acestea apar pe orar si sunt reinnoite pe acesta in fiecare saptamana sau odata la doua saptamani in functie de frecventa activitatii

create table if not exists activitate(

idActivitate int primary key unique not null auto\_increment,

tipActivitate varchar(20),

dataActivitate datetime,

nrMinStudenti int default 10,

nrCrtStudenti int,

nrMaxStudenti int default 150,

durataOre int default 2,

frecventa int default 1,

idMaterie int,

foreign key (idMaterie) references materie (idMaterie) on delete cascade,

idProfesor int,

foreign key (idProfesor) references profesor (idProfesor) on delete cascade

);

Entitatea **grupStudiu** contine informatii privind grupurile de studiu organizate de studenti, acestea pot fi create cat timp nu se suprapun cu alte grupuri sau activitati

create table if not exists grupStudiu(

idGrup int primary key unique not null auto\_increment,

dataGrup datetime,

nrMinParticipanti int default 5,

nrCrtParticipanti int,

nrMaxParticipanti int default 30,

idMaterie int,

foreign key (idMaterie) references materie (idMaterie) on delete cascade

);

Entitatea **inscriereActivitate** contine id-urile tuturor participantilor la activitatile din orar, acestia se pot inscrie o singura data la o anumita activitate cat timp aceasta nu se suprapune cu alte activitati din orarul studentului

create table if not exists inscriereActivitate(

idStudent int,

foreign key (idStudent) references student (idStudent) on delete cascade,

idActivitate int,

foreign key (idActivitate) references activitate (idActivitate) on delete cascade

);

Entitatea **inscriereGrupStudiu** contine informatii privind grupurile de studiu asemanator cu entitatea **inscriereActivitate**

create table if not exists inscriereGrupStudiu(

idStudent int,

foreign key (idStudent) references student (idStudent) on delete cascade,

idGrup int,

foreign key (idGrup) references grupStudiu (idGrup) on delete cascade

);

Entitatea **situatie** contine informatii privind situatiile studentilor, acestia isi pot cauta notele la toate materiile curente prin intermediul id-ului pe care il au

create table if not exists situatie(

idStudent int unique,

foreign key (idStudent) references student (idStudent) on delete cascade,

notaMaterie1 decimal (4,2) default 0.00,

notaMaterie2 decimal (4,2) default 0.00,

notaMaterie3 decimal (4,2) default 0.00,

notaMaterie4 decimal (4,2) default 0.00,

notaMaterie5 decimal (4,2) default 0.00,

notaMaterie6 decimal (4,2) default 0.00,

nrCrediteDisponibile int default 30,

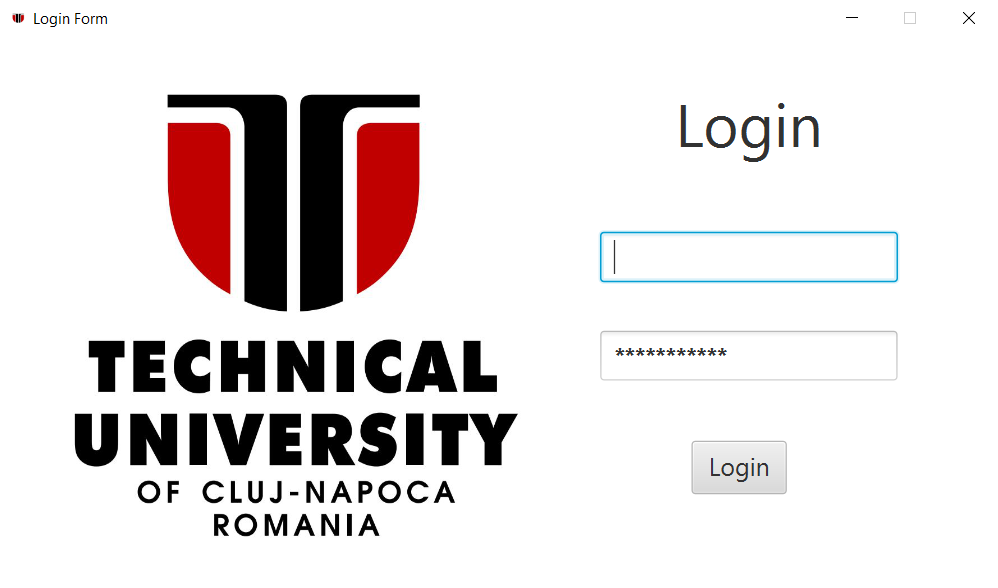
nrCrediteObtinute int default 0.00,

medie decimal(4,2) default 0.00

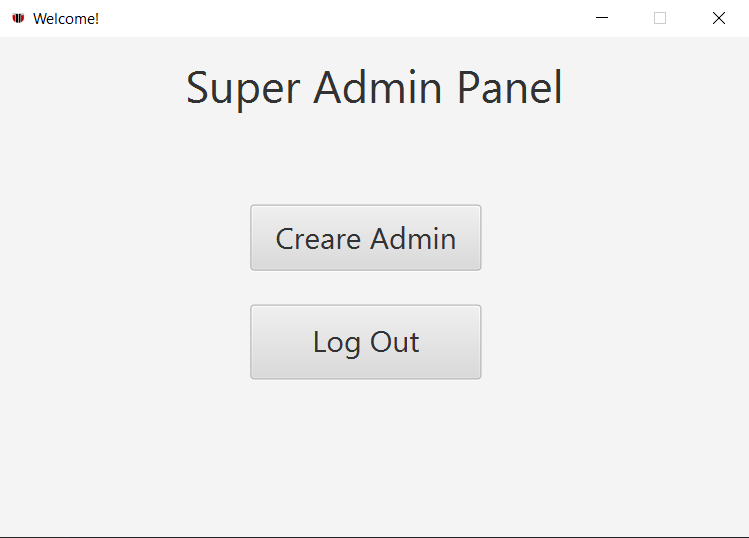
);

**4. Descrierea Functionala a Modulelor**

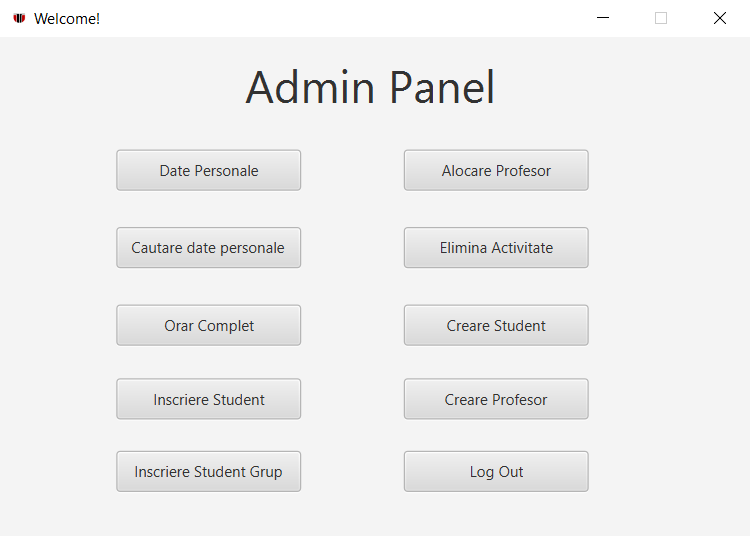
Programul are in total 26 de ferestre care sunt conectate intre ele prin butoane. Am ales sa prezint pe scurt pe cele mai importante dintre ele.



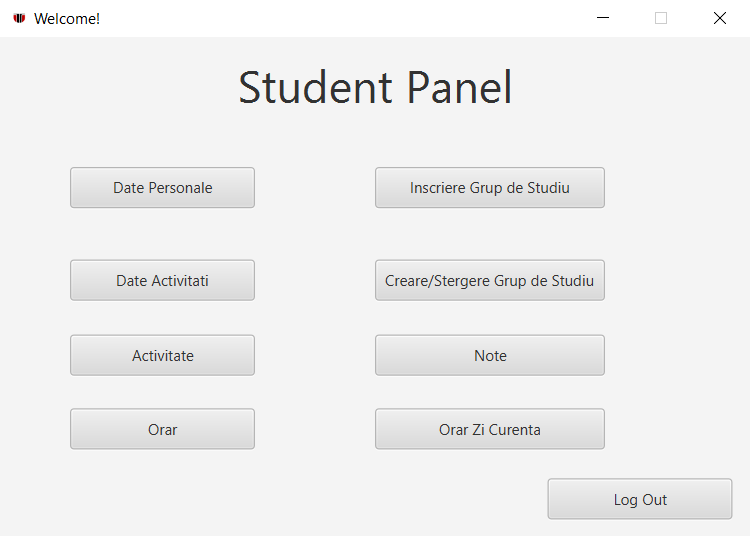
Ecranul de logare unde se pot loga superadministratorii, administratorii, profesorii si studentii cu adresa de mail.



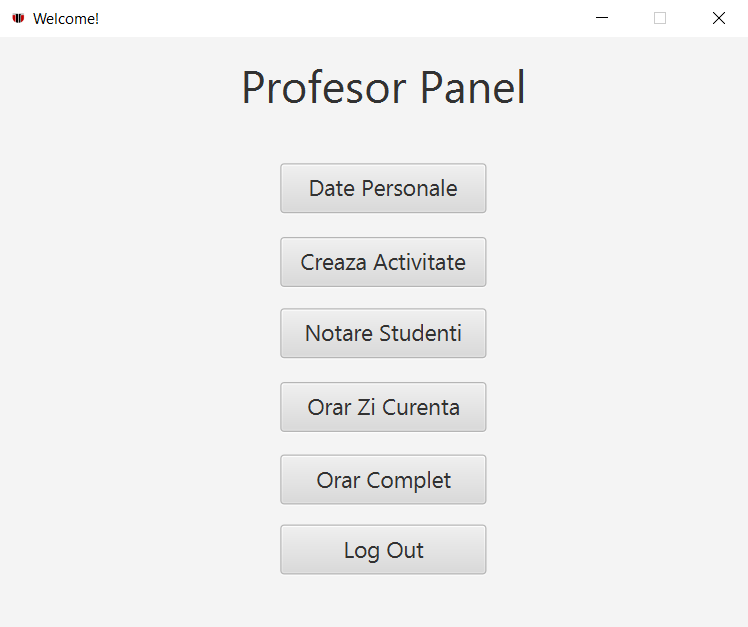
Interfata superadministratorului care ii permite sa creeze conturi de administratori si sa se deconecteze.



Interfata administratorului care ii permite sa isi vada datele personale, sa caute date personale ale altor utilizatori, sa vada toate activitatiile, sa inscrie un student respectiv un profesor la o activitate, sa inscrie un student intr-un grup de studiu, sa elimine o activitate, sa creeze un cont de student sau profesor si in final sa se deconecteze.



Interfata studentului ii permite sa isi vada datele personale, sa vada datele activitatilor la care este inscris, sa vada un orar pe care il poate descarca, se poate inscrie intr-un grup de studiu, poate creea/sterge un grup de studiu, poate sa isi vada notele si orarul din ziua curenta si in final poate sa se deconecteze.

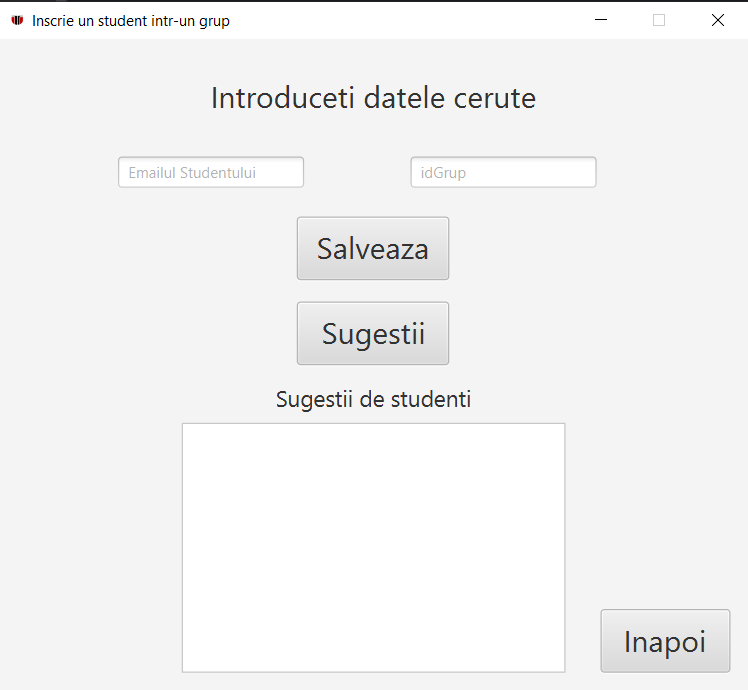


Interfata profesorului ii permite sa isi vada datele personale, sa se inscrie la o activitate creata de el, sa noteze studenti la materia lui, sa isi vada orarul pe ziua curenta cat si cel complet si in final sa se deconecteze.

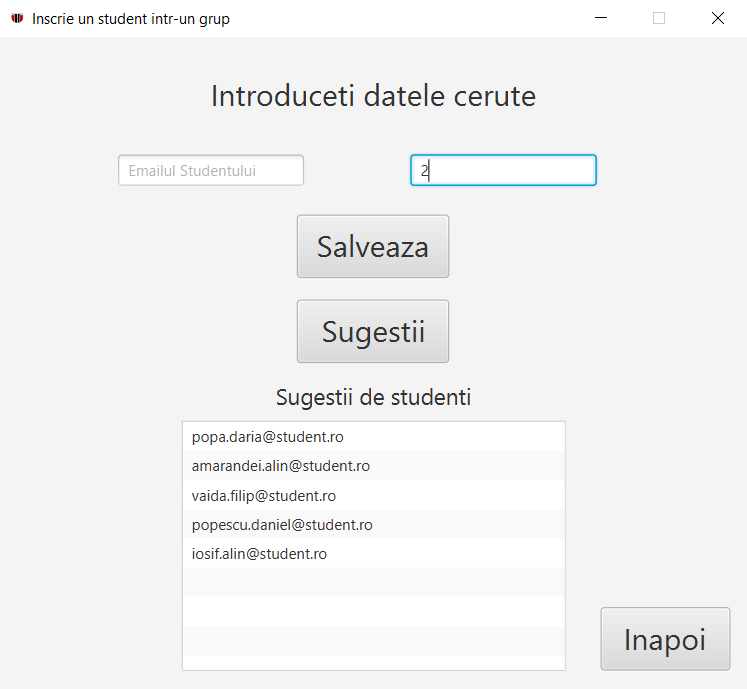
**5. Bonusuri**

* **Afisarea unor sugestii de participanti la grupurile de studiu**

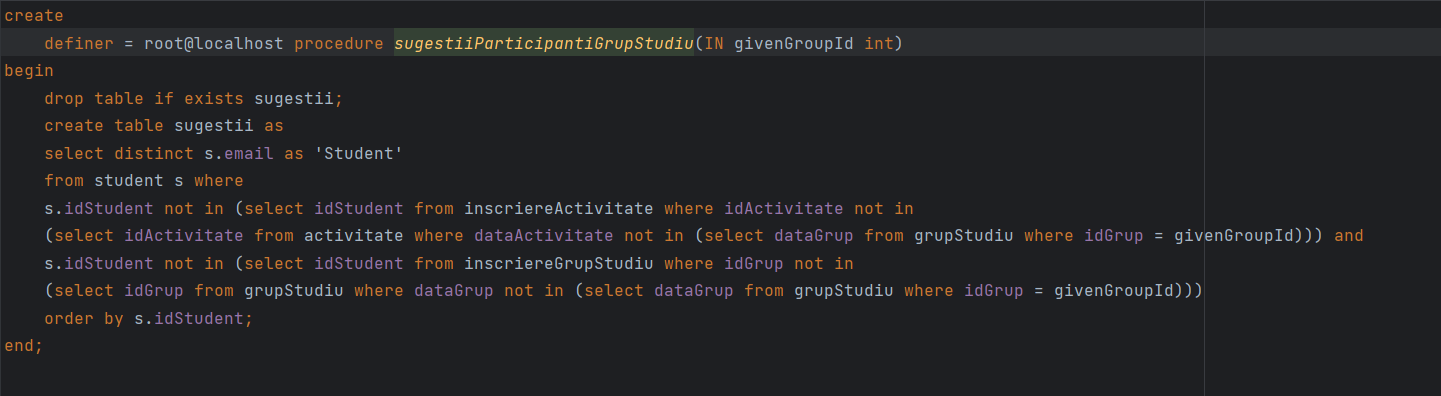
Am decis sa implementam aceasta functionalitate in cadrul optiunii administratorului de a adauga un student la un grup de studiu. Astfel, dupa ce se introduce id-ul grupului dorit, prin apasarea butonului sugestii, lista de sub buton se populeaza cu adresele studentilor care sunt disponibili pentru grupul respectiv. Aceasta functie a fost implementata cu ajutorul unei produceri in SQL si interpretata in Java.



Inaintea apasarii butonului de sugestii



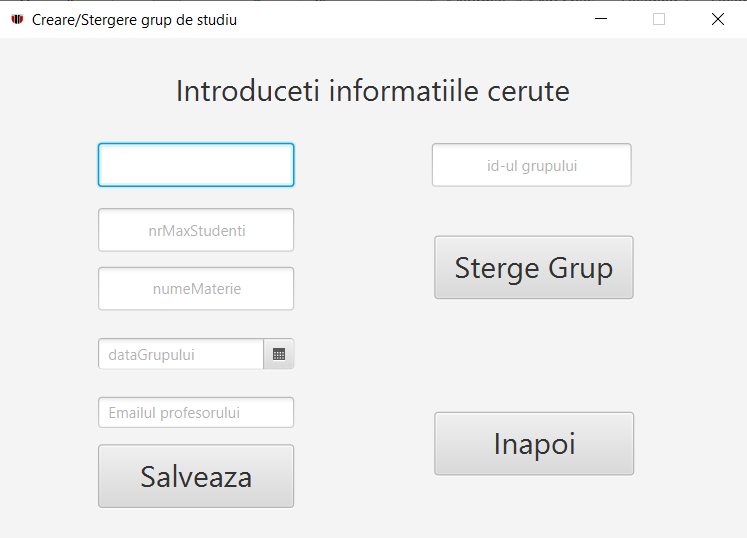
Dupa apasarea butonului de sugestii

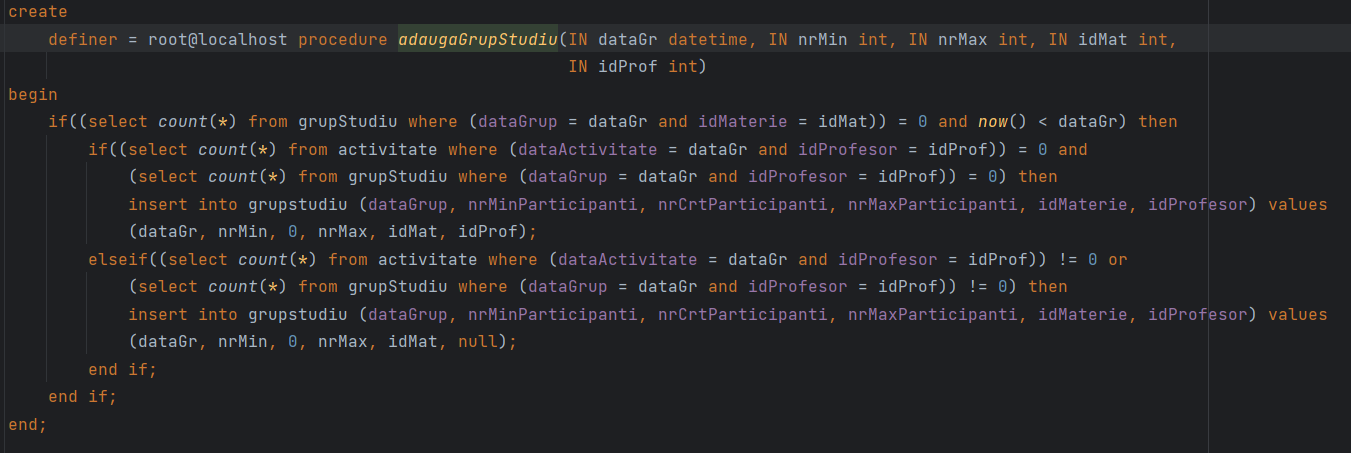


Codul care este responsabil de sugestii

* **Posibilitatea adaugarii unui cadru didactic la o activitate din grupul de studiu.**

Aceasta functionalitate am implementat-o in cadrul optiunii studentului de a crea un grup. Astfel,optional, se poate specifica adresa de mail a profesorului care sa fie alocat la grup.





Codul responsabil pentru functia de creare grup de studiu

**6. Concluzii**

In concluzie, rolul aplicatiei create este acela de a simula o baza de date din cadrul unei facultati. Modul simplificat creat prin intermediul interfetei JavaFX ofera posibilitatea de a vizualiza si descarca date despre toate activitatile care se desfasoara, adauga si elimina participanti la diferitele activitati intretinute precum si notarea studentilor de catre profesori, toate aceste actiuni fiind foarte usor de realizat datorita aplicatiei.

**7. Bibliografie**

* <http://www.mysql.com>
* <http://www.w3schools.com/sql>
* <https://openjfx.io/openjfx-docs/>
* <https://fxdocs.github.io/docs/html5/>