MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

PROIECT

la disciplina
Introducere in Baze de Date

Lanț de policlinici

Cacaraza Tobias Daniel
Costea Delia

An academic

2022-2023

PROIECT de SEMESTRU
Catedra de Calculatoare
Disciplina : Introducere in Baze de Date
Coordonator: s.l. ing. Cosmina IVAN
Data: 20.01.2023

Cuprins

1. Introducere

Proiectul a presupus crearea unei aplicatii care vine in ajutorul angajatilor, dar si a pacientilor din cadrul sistemului medical. Scopul aplicatiei este de a facilita accesul la baza de date a policlinicilor printr-o interfata inutitiva.

Aplicatia poate fi folosita atat de angajatii din cadrul sistemului medical (doctori, asistenti medicali, receptioneri, administratori, inspectori, expert financiar contabil, etc), cat si de catre pacienti.

Instrumentatie software:

Programele folosite în cadrul realizării proiectului sunt:

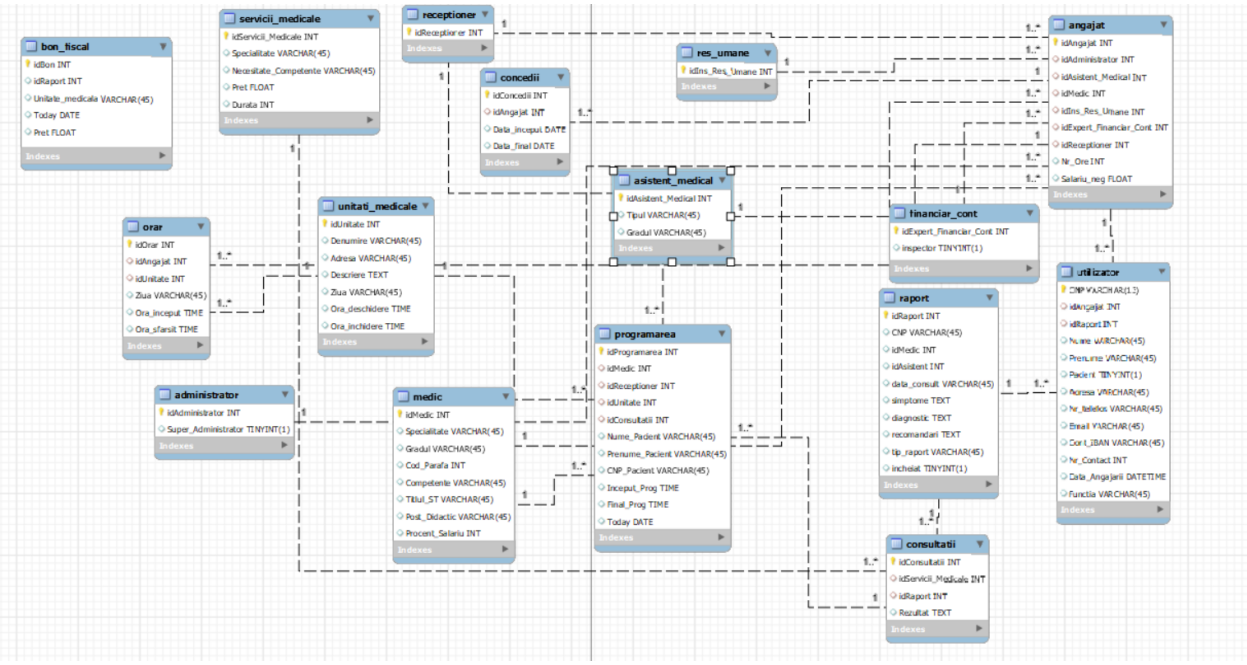
MySQL Workbench - realizarea bazei de date utilizând limbaluj MySQL precum si realizarea diagramei UML a bazei de date

Eclipse – mediu de programare Java

Apache NetBeans IDE – pentru realizarea interfetelor grafice

2. Baza de date

Diagrama EER/UML pentru modelul de date complet



Cod de creare a bazei de date:

Bon Fiscal:

```
DROP TABLE IF EXISTS `policlinica`.`bon_fiscal`;

  CREATE TABLE `policlinica`.`bon_fiscal` (
  `idBon` INT NOT NULL AUTO_INCREMENT,
  `idRaport` INT NULL,
  `Unitate_medicala` VARCHAR(45) NULL,
  `Today` DATE NULL,
  `Pret` FLOAT NULL,
  PRIMARY KEY (`idBon`));
```

Medic:

```
DROP TABLE IF EXISTS `policlinica`.`medic`;

CREATE TABLE `policlinica`.`medic` (
  `idMedic` INT NOT NULL AUTO_INCREMENT,
  `Specialitate` VARCHAR(45) NULL,
```

```
`Gradul` VARCHAR(45) NULL,
`Cod_Parafa` INT NULL,
`Competente` VARCHAR(45) NULL,
`Titlul_ST` VARCHAR(45) NULL,
`Post_Didactic` VARCHAR(45) NULL,
`Procent_Salariu` INT NULL,

PRIMARY KEY (`idMedic`));
```

Administrator:

```
DROP TABLE IF EXISTS `policlinica`.`administrator`;

CREATE TABLE `policlinica`.`administrator` (
`idAdministrator` INT NOT NULL AUTO_INCREMENT,
`Super_Administrator` BOOLEAN NULL,
PRIMARY KEY (`idAdministrator`));
```

Asistent Medical:

```
DROP TABLE IF EXISTS `policlinica`.`asistent_medical`;

CREATE TABLE `policlinica`.`asistent_medical` (
`idAsistent_Medical` INT NOT NULL AUTO_INCREMENT,
`Tipul` VARCHAR(45) NULL,
`Gradul` VARCHAR(45) NULL,
PRIMARY KEY (`idAsistent_Medical`));
```

Resurse Umane:

```
DROP TABLE IF EXISTS `policlinica`.`Res_umane`;

CREATE TABLE `policlinica`.`Res_umane` (
`idIns_Res_Umane` INT NOT NULL AUTO_INCREMENT,
PRIMARY KEY (`idIns_Res_Umane`));
```

Financiar Contabil:

```
DROP TABLE IF EXISTS `policlinica`.`Financiar_cont`;

CREATE TABLE `policlinica`.`Financiar_cont` (
`idExpert_Financiar_Cont` INT NOT NULL AUTO_INCREMENT,
`inspector` BOOLEAN NULL,
PRIMARY KEY (`idExpert_Financiar_Cont`));
```

Receptioner:

```
DROP TABLE IF EXISTS `policlinica`.`receptioner`;

 CREATE TABLE `policlinica`.`receptioner` (
 `idReceptioner` INT NOT NULL AUTO_INCREMENT,
 PRIMARY KEY ( `idReceptioner`));
```

Unitati medicale:

```
DROP TABLE IF EXISTS `policlinica`.`Unitati_medicale`;

 CREATE TABLE `policlinica`.`Unitati_medicale` (
 `idUnitate` INT NOT NULL AUTO_INCREMENT,
 `Denumire` VARCHAR(45) NULL,
 `Adresa` VARCHAR(45) NULL,
 `Descriere` TEXT NULL,
 `Ziua` VARCHAR(45) NULL,
 `Ora_deschidere` TIME NULL,
 `Ora_inchidere` TIME NULL,
 PRIMARY KEY ( `idUnitate`));
```

Raport:

```
DROP TABLE IF EXISTS `policlinica`.`Raport`;

 CREATE TABLE `policlinica`.`Raport` (
 `idRaport` INT NOT NULL AUTO_INCREMENT,
 `CNP` VARCHAR(45) NULL,
 `idMedic` INT NULL,
 `idAsistent` INT NULL,
 `data_consult` VARCHAR(45) NULL,
 `simptome` TEXT NULL,
 `diagnostic` TEXT NULL,
 `recomandari` TEXT NULL,
 `tip_raport` VARCHAR(45) NULL,
 `incheiat` BOOLEAN NULL,
 PRIMARY KEY ( `idRaport`));
```

Servicii medicale:

```
DROP TABLE IF EXISTS `policlinica`.`servicii_medicale`;

 CREATE TABLE `policlinica`.`servicii_medicale` (
 `idServicii_Medicale` INT NOT NULL AUTO_INCREMENT,
```

```
`Specialitate` VARCHAR(45) NULL,
`Necesitate_Competente` VARCHAR(45) NULL,
`Pret` FLOAT NULL,
`Durata` INT NULL,
PRIMARY KEY (`idServicii_Medicale`));
```

Consultatii:

```
DROP TABLE IF EXISTS `policlinica`.`consultatii`;

CREATE TABLE `policlinica`.`consultatii` (
`idConsultatii` INT NOT NULL AUTO_INCREMENT,
`idServicii_Medicale` INT NULL,
`idRaport` INT NULL,
`Rezultat` TEXT NULL,
INDEX `fk_consultatii_servicii_medicale_idx`(`idServicii_Medicale` ASC),
INDEX `fk_consultatii_raport_idx`(`idRaport` ASC),
CONSTRAINT `fk_consultatii_raport_idx`
FOREIGN KEY (`idRaport`)
REFERENCES `policlinica`.`Raport`(`idRaport`),
CONSTRAINT `fk_consultatii_servicii_medicale_idx`
FOREIGN KEY (`idServicii_Medicale`)
REFERENCES `policlinica`.`servicii_medicale`(`idServicii_medicale`),
PRIMARY KEY ( `idConsultatii`));
```

Programari:

```
DROP TABLE IF EXISTS `policlinica`.`programarea`;

CREATE TABLE `policlinica`.`programarea` (
`idProgramarea` INT NOT NULL AUTO_INCREMENT,
`idMedic` INT NULL,
`idReceptioner` INT NULL,
`idUnitate` INT NULL,
`idConsultatii` INT NULL,
`Nume_Pacient` VARCHAR(45) NULL,
`Prenume_Pacient` VARCHAR(45) NULL,
`CNP_Pacient` VARCHAR(45) NULL,
`Inceput_Prog` TIME NULL,
`Final_Prog` TIME NULL,
`Today` DATE NULL,
INDEX `fk_programare_medic_idx` (`idMedic` ASC),
INDEX `fk_programare_receptioner_idx` (`idReceptioner` ASC),
INDEX `fk_programare_unitate_idx` (`idUnitate` ASC),
INDEX `fk_programare_consultatii_idx` (`idConsultatii` ASC),
 CONSTRAINT `fk_programare_receptioner_idx`
```

```
  FOREIGN KEY (`idReceptioner`)
  REFERENCES `policlinica`.`Receptioner` (`idReceptioner`),
 CONSTRAINT `fk_programare_unitate_idx`
  FOREIGN KEY (`idUnitate`)
  REFERENCES `policlinica`.`Unitati_medicale` (`idUnitate`),
 CONSTRAINT `fk_angajat_medic_idx`
  FOREIGN KEY (`idMedic`)
  REFERENCES `policlinica`.`Medic` (`idMedic`),
 CONSTRAINT `fk_programare_consultatii_idx`
  FOREIGN KEY (`idConsultatii`)
  REFERENCES `policlinica`.`consultatii` (`idConsultatii`),
PRIMARY KEY ( `idProgramarea`));
```

Angajat:

```
DROP TABLE IF EXISTS `policlinica`.`angajat`;

 CREATE TABLE `policlinica`.`angajat` (
        `idAngajat` INT(11) NOT NULL AUTO_INCREMENT,
        `idAdministrator` INT(11) NULL DEFAULT NULL,
        `idAsistent_Medical` INT(11) NULL DEFAULT NULL,
        `idMedic` INT(11) NULL,
  `idIns_Res_Umane` INT(11) NULL,
  `idExpert_Financiar_Cont` INT(11) NULL,
  `idReceptioner` INT(11) NULL,
        `Nr_Ore` INT(11) NULL DEFAULT NULL,
        `Salariu_neg` FLOAT NULL DEFAULT NULL,
 INDEX `fk_angajat_administrator1_idx` (`idAdministrator` ASC),
 INDEX `fk_angajat_asistent_medical1_idx` (`idAsistent_Medical` ASC),
 INDEX `fk_angajat_medic1_idx` (`idMedic` ASC),
 INDEX `fk_angajat_res_um_idx` (`idIns_Res_Umane` ASC),
 INDEX `fk_angajat_expert_idx` (`idExpert_Financiar_Cont` ASC),
 INDEX `fk_angajat_receptioner_idx` (`idReceptioner` ASC),
 CONSTRAINT `fk_angajat_receptioner_idx`
  FOREIGN KEY (`idReceptioner`)
  REFERENCES `policlinica`.`Receptioner` (`idReceptioner`),
 CONSTRAINT `fk_angajat_expert_idx`
  FOREIGN KEY (`idExpert_Financiar_Cont`)
  REFERENCES `policlinica`.`Financiar_Cont` (`idExpert_Financiar_Cont`),
 CONSTRAINT `fk_angajat_res_um_idx`
  FOREIGN KEY (`idIns_Res_Umane`)
  REFERENCES `policlinica`.`Res_Umane` (`idIns_Res_Umane`),
 CONSTRAINT `fk_angajat_administrator1`
  FOREIGN KEY (`idAdministrator`)
  REFERENCES `policlinica`.`administrator` (`idAdministrator`),
 CONSTRAINT `fk_angajat_asistent_medical1`
```

```
  FOREIGN KEY (`idAsistent_Medical`)
  REFERENCES `policlinica`.`asistent_medical` (`idAsistent_Medical`),
 CONSTRAINT `fk_angajat_medic1`
  FOREIGN KEY (`idMedic`)
  REFERENCES `policlinica`.`medic` (`idMedic`),
  PRIMARY KEY (`idAngajat`));
```

Utilizator:

```
DROP TABLE IF EXISTS `policlinica`.`utilizator`;

 CREATE TABLE `policlinica`.`utilizator` (
 `CNP` VARCHAR(13) NOT NULL,
 `idAngajat` INT(11) NULL,
 `idRaport` INT(11) NULL,
 `Nume` VARCHAR(45) NULL,
 `Prenume` VARCHAR(45) NULL,
 `Pacient` BOOLEAN NULL,
 `Adresa` VARCHAR(45) NULL,
 `Nr_telefon` VARCHAR(45) NULL,
 `Email` VARCHAR(45) NULL,
 `Cont_IBAN` VARCHAR(45) NULL,
 `Nr_Contact` INT NULL,
 `Data_Angajarii` DATETIME NULL,
 `Functia` VARCHAR(45) NULL,
 INDEX `fk_utilizator_angajat_idx` (`idAngajat` ASC),
 INDEX `fk_utilizator_raport_idx` (`idRaport` ASC),
 CONSTRAINT `fk_utilizator_angajat_idx`
  FOREIGN KEY (`idAngajat`)
  REFERENCES `policlinica`.`angajat` (`idAngajat`),
  CONSTRAINT `fk_utilizator_raport_idx`
  FOREIGN KEY (`idRaport`)
  REFERENCES `policlinica`.`Raport` (`idRaport`),
 PRIMARY KEY (`CNP`) );
```

Orar:

```
DROP TABLE IF EXISTS `policlinica`.`Orar`;

 CREATE TABLE `policlinica`.`Orar` (
 `idOrar` INT NOT NULL AUTO_INCREMENT,
 `idAngajat` INT NULL,
 `idUnitate` INT NULL,
 `Ziua` VARCHAR(45) NULL,
 `Ora_inceput` TIME NULL,
 `Ora_sfarsit` TIME NULL,
```

```
  INDEX `fk_orar_angajat_idx` (`idAngajat` ASC),
  INDEX `fk_orar_unitati_idx` (`idUnitate` ASC),
  CONSTRAINT `fk_orar_angajat_idx`
    FOREIGN KEY (`idAngajat`)
    REFERENCES `policlinica`.`angajat` (`idAngajat`),
  CONSTRAINT `fk_orar_unitati_idx`
    FOREIGN KEY (`idUnitate`)
    REFERENCES `policlinica`.`Unitati_medicale` (`idUnitate`),
  PRIMARY KEY (`idOrar`));
```

Concedii:

```
CREATE TABLE `policlinica`.`concedii` (
  `idConcedii` INT NOT NULL AUTO_INCREMENT,
  `idAngajat` INT NULL,
  `Data_inceput` DATE NULL,
  `Data_final` DATE NULL,
  PRIMARY KEY (`idConcedii`),
  INDEX `fk_concedii_angajat_idx` (`idAngajat` ASC) VISIBLE,
  CONSTRAINT `fk_concedii_angajat`
    FOREIGN KEY (`idAngajat`)
    REFERENCES `policlinica`.`angajat` (`idAngajat`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
```

3. Proceduri. Evenimente

Lista de proceduri:

- Add_policlinica
- Add_utilizator
- Alter_asistent_medical
- Alter_medic
- Cauta_medic
- Cauta_CNP_asistent
- Cauta_CNP_medic
- Drop_policlinica
- Drop_utilizator
- Drop_raport
- Raport

Cateva dintre procedurile folosite sunt:

Adaugarea unei policlinici:

```
CREATE PROCEDURE `policlinica`.`add_policlinica`(Denumire_unitate VARCHAR(45),ad
VARCHAR(45),Descr VARCHAR(45),Zi VARCHAR(45),ora1 TIME,ora2 TIME)
BEGIN
        #set @tp := tip, @col := new_coloana, @tab := tabel, @after_c = dupa_coloana;
   INSERT INTO unitati_medicale
            (Denumire, Adresa, Descriere,Ziua,Ora_deschidere,Ora_inchidere)
            VALUES
            (Denumire_unitate,ad, Descr, Zi,ora1,ora2);

END//

delimiter ;
```

Stergerea unei policlinici:

```
DROP PROCEDURE if EXISTS `policlinica`.`drop_policlinica`;

delimiter //

CREATE PROCEDURE `policlinica`.`drop_policlinica`(Id INT)
BEGIN

   DELETE FROM `policlinica`.`unitati_medicale` WHERE (`idUnitate` = Id);

END//
```

Cautarea CNP-ului unui medic:

```
DROP PROCEDURE IF EXISTS policlinica.cautaCNP_medic;

delimiter //

CREATE PROCEDURE policlinica.cautaCNP_medic(id_M INT)
BEGIN
   SELECT u.nume,u.prenume,u.CNP
   FROM policlinica.medic m inner join policlinica.angajat a using (idMedic) inner join
policlinica.utilizator u using (idAngajat)
   where m.idMedic = id_M;
END//

delimiter ;
```

Adaugarea unui nou utilizator:

```
USE `policlinica`;

DROP procedure IF EXISTS `policlinica`.`add_utilizator`;

DELIMITER $$

USE `policlinica`$$

CREATE DEFINER=`root`@`localhost` PROCEDURE `add_utilizator`(CNP_utilizator
VARCHAR(13),num VARCHAR(45),

pren VARCHAR(45),Paci TINYINT(1),adrs VARCHAR(45),nr_tel VARCHAR(45),em
VARCHAR(45),

IBAN VARCHAR(45),contact INT, Data_ang DATETIME, fct VARCHAR(45) )

BEGIN

        set @f = fct,@id_f = NULL,@id_A = NULL;

        IF @f = "Medic" THEN

        set @f = "Medic";

   INSERT INTO medic

                ()

      VALUES

      ();

   SELECT @id_f := MAX(idMedic) FROM medic;

   INSERT INTO angajat

      (idMedic)

      VALUES

      (@id_f);

   ELSEIF @f = "Asistent Medical" THEN

   set @f = "Asistent Medical";

   INSERT INTO asistent_medical

   ()

   VALUES

   ();
```

```
SELECT @id_f := MAX(idAsistent_Medical) FROM asistent_medical;
INSERT INTO angajat
   (idAsistent_Medical)
   VALUES
   (@id_f);
ELSEIF @f = "Financiar Cont" THEN
set @f = "Financiar Cont";
INSERT INTO financiar_cont
()
VALUES
();
SELECT @id_f := MAX(idExpert_Financiar_Cont) FROM financiar_cont;
INSERT INTO angajat
   (idExpert_Financiar_Cont)
   VALUES
   (@id_f);
      ELSEIF @f = "Administrator" THEN
set @f = "Administrator";
INSERT INTO Administrator
()
VALUES
();
SELECT @id_f := MAX(idAdministrator) FROM administrator;
INSERT INTO angajat
   (idAdministrator)
   VALUES
   (@id_f);
ELSEIF @f = "Receptioner" THEN
```

```
   set @f = "Receptioner";
   INSERT INTO receptioner
   ()
   VALUES
   ();
   SELECT @id_f := MAX(idReceptioner) FROM receptioner;
   INSERT INTO angajat
      (idReceptioner)
      VALUES
      (@id_f);
   ELSE
    SET @f = "Pacient";
   END IF;
   IF @f = "Pacient"
   Then
   set @id_A = NULL;
   else
         SELECT @id_A := MAX(idAngajat) FROM Angajat;
   end if;
   INSERT INTO utilizator
            (CNP,
idAngajat ,Nume ,Prenume ,Pacient ,Adresa ,Nr_telefon ,Email ,Cont_IBAN ,Nr_Contact ,
Data_Angajarii , Functia )
            VALUES
            (CNP_utilizator,@id_A ,num ,pren ,Paci ,adrs ,nr_tel ,em ,IBAN,contact , Data_ang ,
@f  );
END$$
DELIMITER ;
```

Stergerea unui utilizator:

```
DROP PROCEDURE if EXISTS `policlinica`.`drop_utilizator`;
delimiter //
CREATE PROCEDURE `policlinica`.`drop_utilizator`(CNP_U VARCHAR(13))
BEGIN
   SET @a= NULL,@f = NULL, @d = NULL;
   SELECT @a := u.idAngajat from utilizator u where u.CNP = CNP_U;
   Delete FROM orar where idAngajat = @a;
   Delete FROM concedii where idAngajat = @a;
   SELECT @f := u.Functia from utilizator u where u.CNP = CNP_U;
   IF @f = "medic" THEN
      Select @d := m.idMedic from medic m inner join angajat a using (idMedic) where
a.idAngajat = @a;
   ELSEIF @f = "asistent medical" THEN
                Select @d := am.idAsistent_Medical from asistent_medical am inner join angajat
a using (idAsistent_Medical) where a.idAngajat = @a;
   ELSEIF @f = "financiar cont" THEN
                Select @d := f.idExpert_financiar_cont from financiar_cont f inner join angajat a
using (idExpert_financiar_cont) where a.idAngajat = @a;
   ELSEIF @f = "receptioner" THEN
                Select @d := r.idReceptioner from receptioner r inner join angajat a using
(idReceptioner) where a.idAngajat = @a;
         ELSEIF @f = "administrator" THEN
                Select @d := r.idAdministrator from administrator ad inner join angajat a using
(idAdministrator) where a.idAngajat = @a;
   ELSE
    SET @f = " ";
   END IF;
   DELETE FROM `utilizator` u WHERE ( u.`CNP` = CNP_U);
   DELETE FROM angajat a WHERE a.idAngajat = @a;
         IF @f = "medic" THEN
      DELETE FROM medic  m WHERE m.idMedic = @d;
   ELSEIF @f = "asistent medical" THEN
      DELETE FROM  asistent_medical am  WHERE am.idAsistent_Medical = @d;
   ELSEIF @f = "financiar cont" THEN
      DELETE FROM  financiar_cont f WHERE f.idExpert_Financiar_Cont = @d;
   ELSEIF @f = "receptioner" THEN
      DELETE FROM  receptioner r WHERE r.idReceptioner = @d;
         ELSEIF @f = "administrator" THEN
      DELETE FROM  Administrator ad WHERE ad.idAdministrator = @d;
   ELSE
    SET @f = " ";
   END IF;
END//
delimiter ;
```

Exemple de view-uri folosite:

Pentru vizualizarea rapoartelor:

```
USE `policlinica`;

CREATE  OR REPLACE VIEW `view_Raport` AS select * from raport;
```

Pentru vizualizarea profiturilor aduse de medici:

```
USE `policlinica`;

CREATE  OR REPLACE VIEW `Profit_medic` AS select p.idUnitate as unit ,sm.Pret as profit,sm.durata as dur,m.idMedic, (a.Nr_Ore*a.Salariu_neg/10) as salar from servicii_medicale sm inner join consultatii using (idServicii_Medicale) inner join programarea p using (idConsultatii) inner join medic m using (idMedic) inner join angajat a using (idMedic);
```

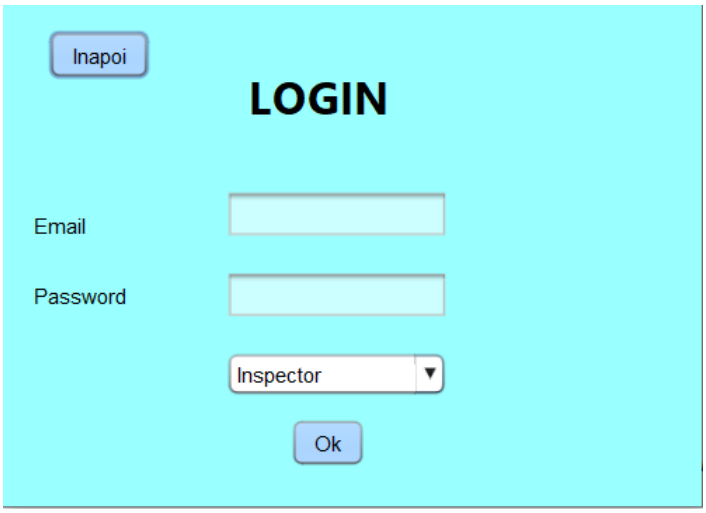
Pentru vizualizarea angajatilor:

```
USE `policlinica`;

CREATE  OR REPLACE VIEW `view_Angajati` AS select * from angajat;
```

4. Interfața grafică

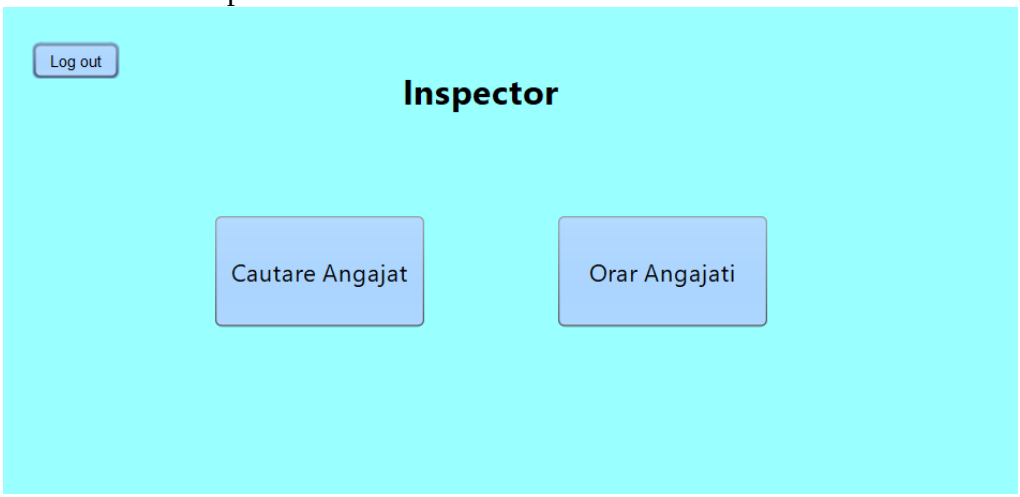
Pagina de conectare, unde persoanele aleg departamentul din care fac parte in cazul in care sunt angajati, iar pacienti dau click pe butonul pacient



Pagina de login, unde persoana care intra in cont trebuie sa selecteze functia pe care o are

Pentru departamentul de gestiune a resurselor umane, interfetele principale sunt:

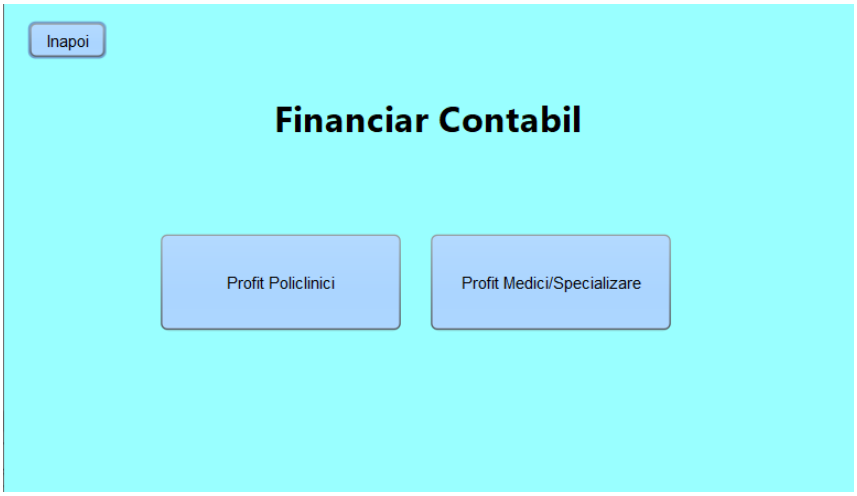
Pentru inspectori:

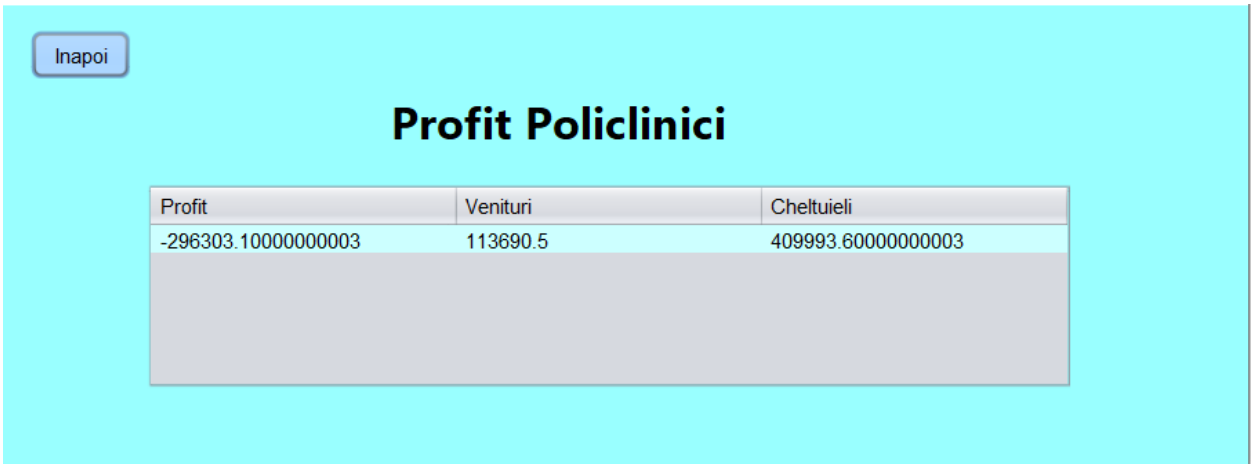


Pentru angajatii de tip: receptioner, medic sau asistent medical

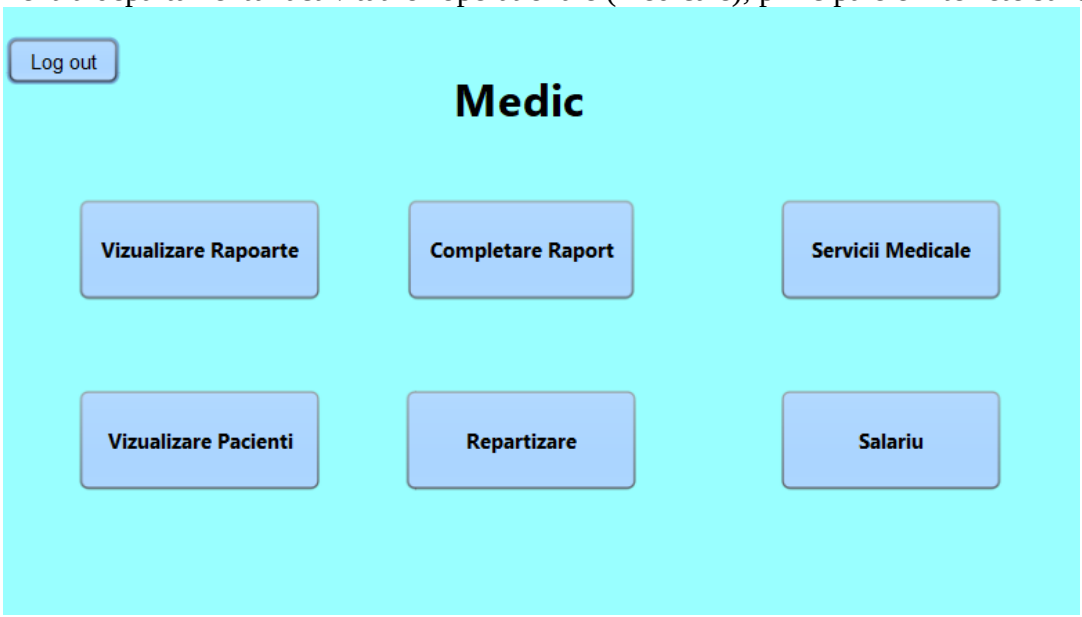
Pentru Expertul financiar contabil:

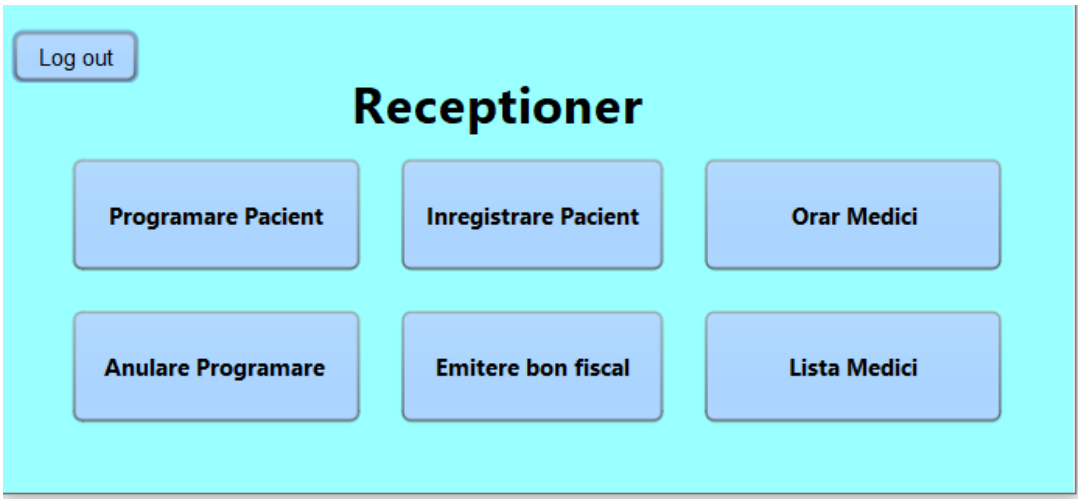
Pentru departamentul de operatii financiar-contabile:

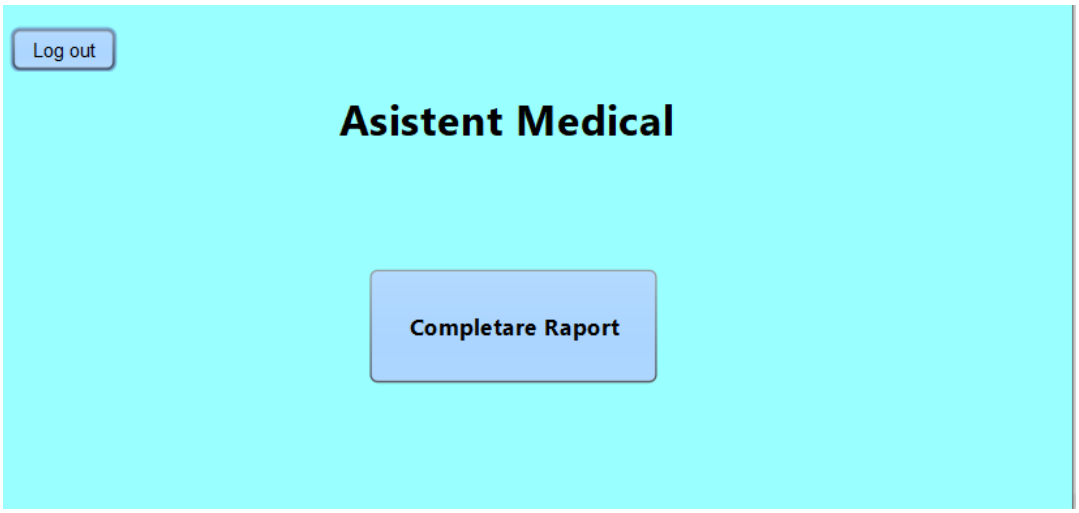




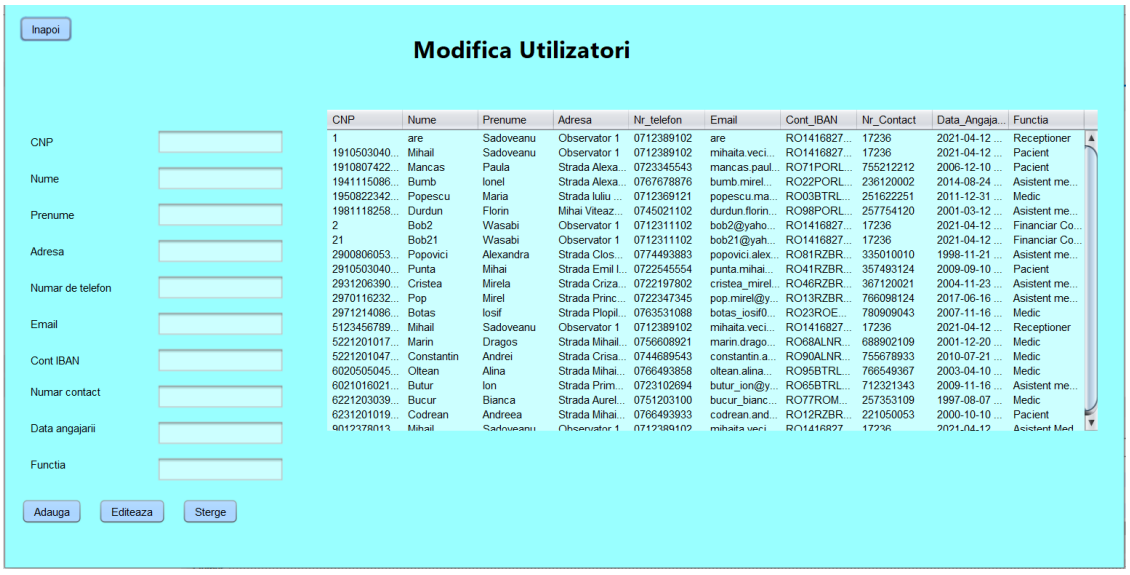
Pentru departamentul activitatilor operationale (medicale), principalele interfete sunt:



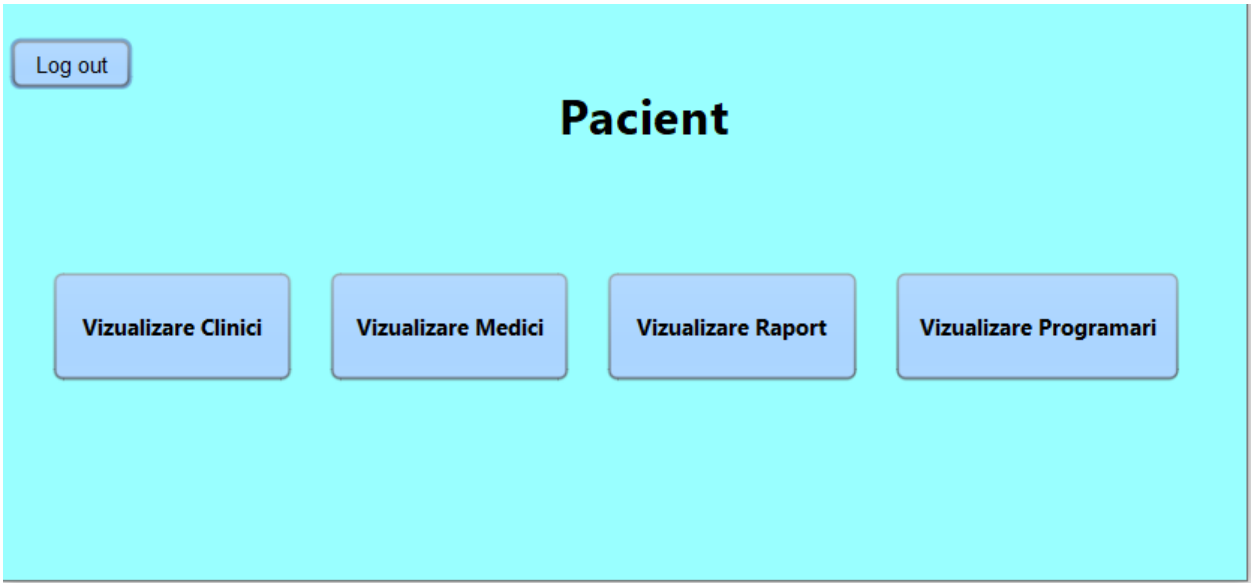




Pentru administratori, interfetele sunt:



Interfetele principale pentru utilizatorul de tip pacient sunt:



idUnitate	Denumire	Adresa	Descriere	Ziua	Ora deschidere	Ora inchidere
1	Dispensar 1	Str. Principala ...	Dispensarul n...	Luni	09:00:00	17:00:00
2	Dispensar 2	Str. Parcului 4...	Ne specializă...	Marți	10:00:00	19:00:00
3	Dispensar 3	Str. Frunzei 789	Personalul nos...	Miercuri	09:00:00	17:00:00
4	Dispensar 4	Str. Stejarului ...	Avem o gamă ...	Joi	10:00:00	19:00:00
5	Dispensar 5	Str. Pinului 222	Dispensarul n...	Vineri	09:00:00	17:00:00
6	Dispensar 6	Str. Cedrului 3...	Avem o gamă ...	Sâmbătă	10:00:00	19:00:00
7	Dispensar 7	Str. Mesteacă...	Dispensarul n...	Duminică	09:00:00	17:00:00
8	Dispensar 8	Str. Aurel Vlaic...	Avem medici s...	Sâmbătă	11:00:00	20:00:00
9	Catena	Strada Mihail-...	Farmacie	Luni-Vineri	08:00:00	17:00:00
10	Catena	Strada Mihail-...	Farmacie	Luni-Vineri	08:00:00	17:00:00

5. Clase și secvențe cod Java

Interfata realizata bazei de date Policlinica, este o interfata simpla cu ajutorul careia utilizatorul poate face anumite operatii.

Interfata contine multiple fereste, fiecare fiind asociata unui departament sau functii ce apartin de utilizator.

Codul de conectare la baza de date:

```
try {
                con = DriverManager.getConnection("jdbc:mysql://localhost/policlinica", "root",
"costea123");

                statement = (Statement) con.createStatement();
                ResultSet rs = statement.executeQuery("Select CNP,Email,Functia from
utilizator where CNP = '" + password
                                + "' and Email = '" + userName + "';");

                if (rs.next()) {
                        if (!rs.getString("Email").equals(null)) {

                                JOptionPane.showMessageDialog(this, "Conectare cu succes!");

                                if (Selectare.getSelectedItem().toString().equals("Pacient")
                                                && rs.getString("Functia").equals("Pacient")) {
                                        UserParola a = new UserParola(userName, password);
                                        Pacient in = new Pacient(a);
                                        in.show();
                                        dispose();

                                } else {

                                        JOptionPane.showMessageDialog(this, "Functia
gresita!");

                                }

                        }
                } else
                        JOptionPane.showMessageDialog(this, "Parola sau email gresite!");

        } catch (

        SQLException sqlException) {
                sqlException.printStackTrace();
        }
```