



**FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE**

**PROIECT**

la disciplina

Baze de Date

**Sistem de gestiune al unui lanț de policlinici**

**Fazacaș Ioana și Fleșeriu Ioan-Rareș**

**An academic :2023 – 2024**

**Grupa: 30225**

PROIECT de SEMESTRU  
Catedra de Calculatoare  
Disciplina : Baze de Date  
Coordonator: s.l. ing. Cosmina Ivan  
Data 17.01.2024

## **CUPRINS**

<b>I. Introducere</b>	<b>3</b>
<b>II. Analiza cerințelor utilizatorilor</b>	<b>4</b>
<b>III. Modelul de date și descrierea acestuia</b>	
1. Entități și atributele lor	5
2. Diagrama UML a bazei de date	7
<b>IV. Detalii ale elementelor din MySQL</b>	
1. Triggere	7
2. Proceduri	8
<b>V. Descrierea funcțională a modulelor</b>	<b>9</b>
<b>VI. Prezentarea sumară a interfeței</b>	<b>10</b>
<b>VII. Prezentarea modelului de interogare și a modului de implementare</b>	<b>14</b>
<b>VIII. Metode de securizare a aplicației</b>	<b>16</b>
<b>IX. Interfața</b>	<b>17</b>
<b>X. Bibliografie</b>	<b>18</b>

## I. Introducere

Am implementat unui sistem informatic destinat gestiunii activităților dintr-un lanț de policlinici.

Lanțul de policlinici este format din mai multe unități medicale, fiecare fiind caracterizată prin denumire, adresă, descrierea serviciilor oferite și programul de funcționare. Funcționalitățile policlinicii și datele din baza sa de date vor fi accesate prin intermediul unei interfețe grafice , care va ușura gestionarea activității policlinicii.

Aplicația va putea fi accesată, pe baza unui proces de autentificare, de către mai multe tipuri de utilizatori, operând în departamentele resurse umane, financiar-contabil sau medical. Pentru fiecare tip de utilizator se vor reține informații precum CNP, nume, prenume, adresa, număr de telefon, email, cont IBAN, numărul de contract, data angajării, funcția deținută în cadrul lanțului de policlinici. Fiecare utilizator își va putea vizualiza datele personale imediat după ce va accesa sistemul informatic, fără a avea însă posibilitatea de a le modifica. În funcție de departamentul din care face parte utilizatorul acesta va putea efectua anumite operații cum ar fi înregistrarea unui pacient, completarea unei programări, completarea unui raport medical, etc.

După terminarea atribuțiilor sale , utilizatorul are posibilitatea de a se deautentifica și să fie trimis din nou la fereastra principală de autentificare.

## **II. Analiza cerintelor utilizatorilor ( Specificatiile de proiect)**

Cerința solicită implementarea unui sistem informatic pentru gestionarea activităților într-un lanț de policlinici. Aplicația va utiliza o bază de date MySQL și va avea o interfață grafică pentru interacțiunea cu utilizatorii. Funcționalitățile principale vizează gestionarea angajaților, aspecte financiar-contabile și administrarea activităților curente ale policlinicii.

Utilizatorii se autentifică în sistem pentru acces, cu mai multe tipuri de utilizatori (administrator, super-administrator și angajat (medic, asistent medical, inspector, contabil, recepționar). Informațiile personale sunt reținute, dar utilizatorii pot doar vizualiza datele proprii, fără a le modifica.

Deautentificarea oferă posibilitatea schimbării utilizatorului fără a reporni aplicația.

În ceea ce privește drepturile de acces ale bazei de date:

- Administratorii pot adăuga, modifica și șterge informații despre utilizatori.
- Există un super-administrator cu privilegii extinse asupra administratorilor.
- Angajații au informații specifice, cum ar fi salariu negociat, număr de ore/lună și funcția ocupată.
- Proiectul se împarte pe module, așadar am abordat problema astfel încât fiecare angajat, în funcție de modulul de care face parte, să fie repartizat în departamentul corespunzător („uman”, „medical”, „financiar”).

### III. Modelul de date si descrierea acestuia

În continuare vom prezenta tabelele create în MySQL, adăugand de asemenea fiecare atribut al fiecărui tabel:

#### Entitati si attributele lor

Baza de date a policlinicii este formata din urmatoarele tabele :

##### **Policlinica**

*Attribute:* id\_policlinica, id\_UM, id\_venit\_policlinica

##### **Adresa**

*Attribute:* id, oras, strada

##### **Persoana**

*Attribute:* CNP, nume, prenume, id\_adresa, nr\_telefon, email, IBAN

##### **Angajat**

*Attribute:* id\_angajat, CNP, nr\_contract, data\_angajarii, departament, functia, salariu, nr\_ore, zile\_ramase\_concediu

##### **Utilizator**

*Attribute:* id\_angajat, username, parola, tip

##### **Asistent\_medical**

*Attribute:* id\_angajat, tip, grad

##### **Medic**

*Attribute:* id\_angajat, id\_specialitati, grad, cod\_parafa, id\_competente, titlu stiintific, postul\_didactic, procent\_aditional

##### **Competente**

*Attribute:* id, ecografie, endoscopie\_digitala, ecocardiografie, cardiologie\_interventionala, bronhoscopie, EEG, EMG, dializa, chirurgie\_laparoscopica, chirurgie\_toracica, chirurgie\_spinala, CT, id\_medic, id\_servicii

### **Specialitati**

*Attribute:* id, boli\_infectioase, cardiologie, chirurgie, dermato\_estetica, dermatovenerologie, diabet\_nutritie, medicina\_generala, ORL, neurologie, obstretica\_ginecologie, pediatrie, ortopedie, oftalmologie

### **Servicii**

*Attribute:* id, nume\_serviciu, pret, durata, id\_competente, id\_specialitate

### **Concedii**

*Attribute:* id, id\_angajat, data\_incepere, data\_finalizare, nr\_zile, zile\_ramase

### **Salarii**

*Attribute:* id\_salarii, id\_angajat, suma, an, luna

### **Programari**

*Attribute:* id\_programare, data\_programarii, id\_servicii, id\_medic, pret\_medic, durata\_consultatie

### **Pacient**

*Attribute:* id\_pacient, cnp, id\_raport

### **Raport\_medical**

*Attribute:* id, id\_pacient, id\_medic, id\_asistent, data\_consult, bon, simptome, id\_investigatii, diagnostic, recomandari, parafa, durata\_consultatie

### **Unitate\_medicala**

*Attribute:* id, denumire, id\_adresa, id\_servicii, program

### **Istoric**

*Attribute:* id, id\_raport



**Calcul\_venit1** : La fiecare nouă programare efectuată se va face un update la venitul policlinicii, adunându-se la profit banii încasați de pe serviciul efectuat în programare și se va face update la salariul medicului care a efectuat serviciul, adăugându-i-se un procent din banii încasați.

**Completare\_salarii** : La fiecare nouă lună se va completa tabela de salarii a angajaților.

**Calcul\_durata\_totala** : Însumează timpul total efectuat pentru toate serviciile dintr-o programare.

**Gasire\_departament** : Completează automat departamentul unui angajat pe baza funcției pe care o are în policlinică.

**Valabilitate\_concediu** : Verifică dacă un angajat și-a epuizat zilele de concediu, iar dacă acesta nu mai are numărul de zile rămase necesare pentru efectuarea unui nou concediu nu se va efectua inserarea în tabela concedii și se va afișa mesajul "'Nu sunt suficiente zile de concediu disponibile!'".

## **Proceduri :**

**Afisari\_profit\_policlinica** : Afișează venitul policlinicii

**Afisari\_salarii\_angajat** : Afișează salariile unui angajat

**Programare2** : Completează o tuplă în tabela programari atunci când se efectuează o programare primind ca parametrii : serviciul , data consultației, prețul și CNP-ul pacientului , căutând medicul cu competențele potrivite pentru efectuarea serviciului selectat și durata consultației tot în funcție de serviciu.

**Print\_pacienti** : Afișează pacienții unui medic pe într-o dată anume.

**Print\_raport\_pacienti** : Afișează raportul medical al unui pacient.

**Completare\_raport** : Efectuează completarea unui raport medical sau modificarea acestuia dacă raportul deja existent nu a fost parafat.

**Afisare\_orar\_inspector** : Afișează orarul unui angajat.

**Verificare\_concediu** : Verifică dacă un angajat se află în concediu sau nu.



**Print\_concedii\_angajat** : Afișează concediile unui angajat.

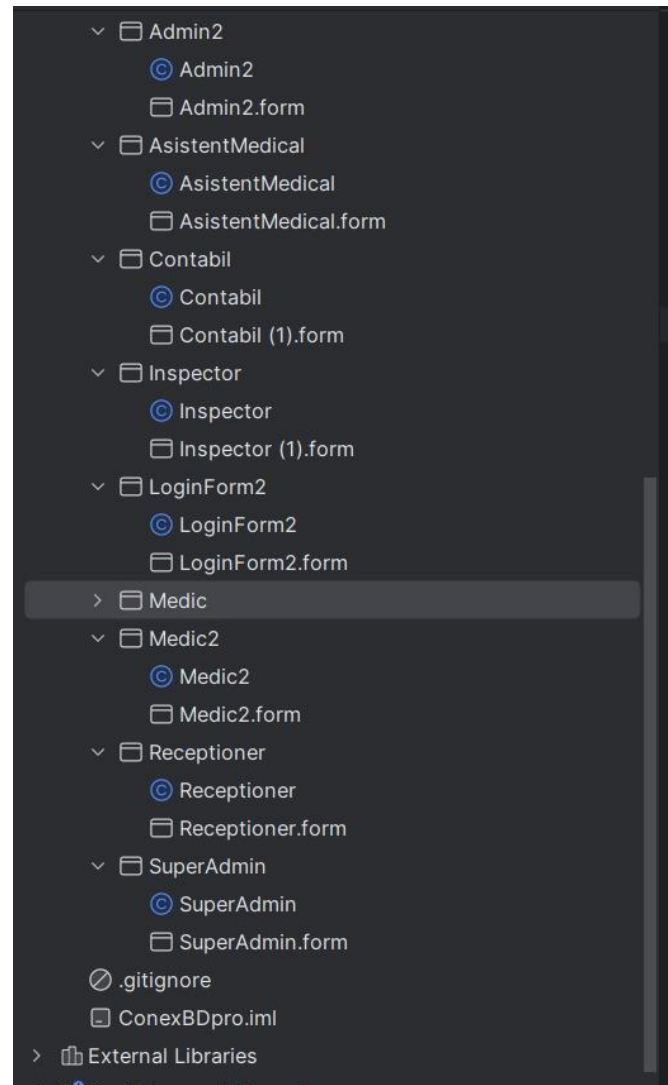
**Update\_concediu** : Modifică zilele rămase de concediu după fiecare concediu efectuat

## V.Descrierea funcțională a modulelor

Proiectul este structurat în mai multe clase care creează o fereastră pentru fiecare tip de angajat în parte. Fiecare clasa conține elementele de interfață corespunzătoare, alături de implementarea tuturor procedurilor și a funcțiilor folosite pentru funcționalitate.

La rularea programului, se va deschide fereastra de LoginForm, care va permite autentificarea angajaților din policlinică.

După conectare, fiecărui angajat i se va deschide fereastra corespunzătoare funcției lui, după cum se observă și dispunerea lor în imaginea alăturată.



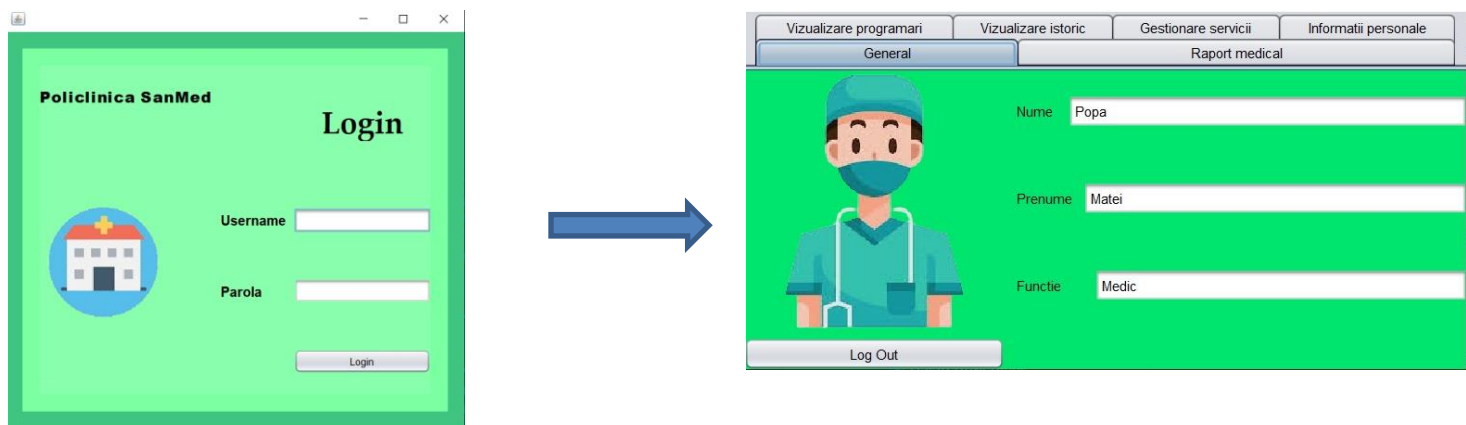
Astfel, conform repartizării pe module, fiecare fereastră va permite accesarea bazei de date în limita atribuțiilor fiecărui angajat (de ex. un medic va putea își vada fiecare salariu

obținut pe parcursul ultimelor luni, dar doar al său, în timp ce un inspector are posibilitatea de a vedea salariul oricărui angajat pe care îl caută).

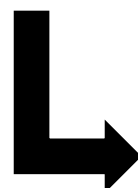
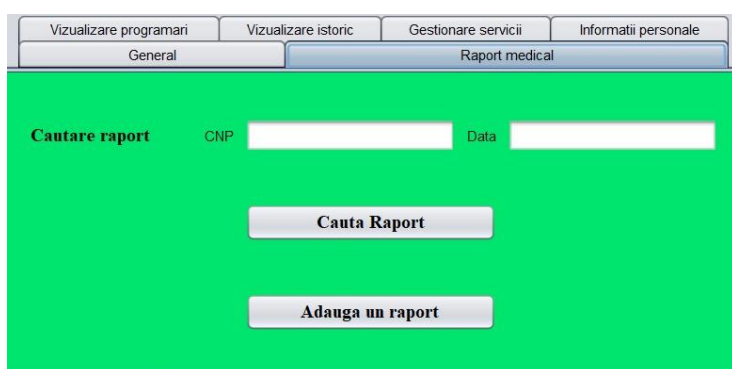
Fiecare clasă din Java este legată de LoginForm(pentru deautentificare), în timp ce clasa LoginForm este la rândul ei legată de fiecare altă clasă(pentru autentificare).

## VI. Prezentarea sumară a interfeței

După logare, medicul Popa Matei va fi trimis la fereastra corespunzătoare medicilor, unde folosind ferestre de tipul "Jtapped Pane", poate accesa fiecare pagină ce îl pune în contact cu baza de date, bineînțeles, având în vedere limitarea accesului acestuia



În exemplul următor, medicul a apăsă butonul "Adaugă un raport", care îi deschide o fereastră cu toate câmpurile necesare completării unui raport.



In continuare sunt prezentate restul de ferestre Jtapped Pane pentru un angajat de tip medic:

Afișarea tuturor programărilor medicului din data introdusă

The screenshot shows a software window titled 'General' and 'Raport medical'. Below the title bar are four tabs: 'Vizualizare programari' (selected), 'Vizualizare istoric', 'Gestionare servicii', and 'Informatii personale'. The main area has a green background. On the left is a clock icon. To its right is a text label 'Data programarii' followed by a white input field. At the bottom center is a button labeled 'Afiseaza programari'.

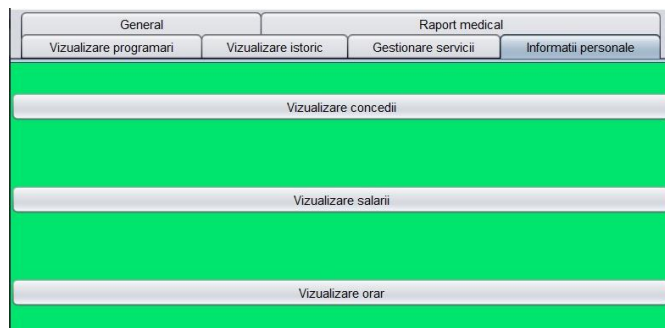
Afișarea istoricului unui pacient

The screenshot shows a software window titled 'General' and 'Raport medical'. Below the title bar are four tabs: 'Vizualizare programari', 'Vizualizare istoric' (selected), 'Gestionare servicii', and 'Informatii personale'. The main area has a green background. On the left is a text label 'CNP' followed by a white input field. At the bottom center is a button labeled 'Cauta pacient'.

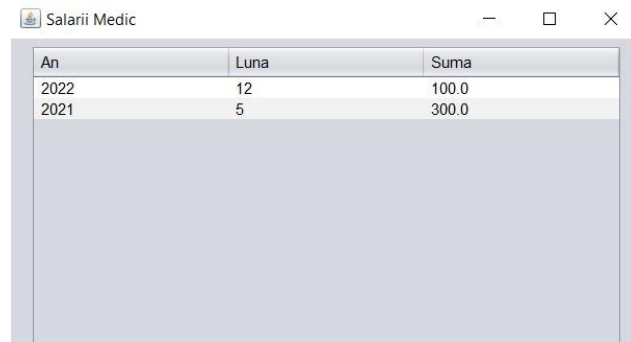
Adăugarea/Ștergerea unui serviciu

The screenshot shows a software window titled 'General' and 'Raport medical'. Below the title bar are four tabs: 'Vizualizare programari', 'Vizualizare istoric', 'Gestionare servicii' (selected), and 'Informatii personale'. The main area has a green background. On the left is a text label 'Serviciu medical'. To its right are two rows of input fields: the first row has 'Denumire' and 'Competente' (with a dropdown arrow), and the second row has 'Pret' and 'Durata'. At the bottom are two buttons: 'Adaugare serviciu' and 'Stergere serviciu'.

## Afișarea informațiilor personale



The screenshot shows a software interface with a menu bar at the top containing 'General' and 'Raport medical'. Below the menu bar are several buttons: 'Vizualizare programari', 'Vizualizare istoric', 'Gestionare servicii', and 'Informatii personale'. Below these buttons are three large green rectangular areas, each containing a button: 'Vizualizare concedii', 'Vizualizare salarii', and 'Vizualizare orar'.

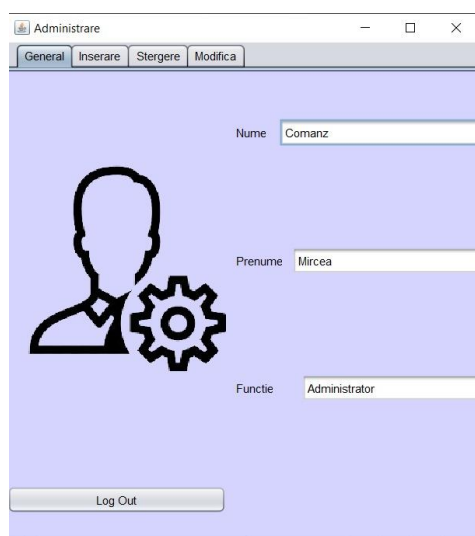


The screenshot shows a window titled 'Salarii Medic' with a table containing salary data for a doctor. The table has three columns: 'An' (Year), 'Luna' (Month), and 'Suma' (Sum).

An	Luna	Suma
2022	12	100.0
2021	5	300.0

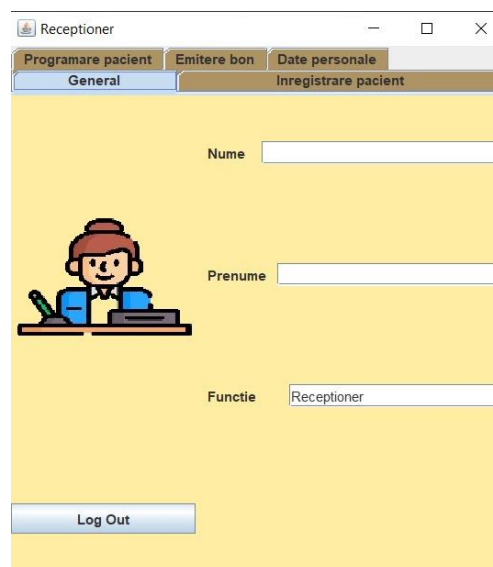
Asemeni modelului prezentat mai sus, fiecare fereastră va avea funcționalități specifice, în același stil de construire al interfeței :

## Fereastra Admin



The screenshot shows a window titled 'Administrare' with a menu bar containing 'General', 'Inserare', 'Stergere', and 'Modifica'. The main area has a light purple background and contains a large icon of a person with a gear. To the right of the icon are three text input fields: 'Nume' (Name) with the value 'Comanz', 'Prenume' (First Name) with the value 'Mircea', and 'Functie' (Function) with the value 'Administrator'. At the bottom left is a 'Log Out' button.

## Fereastra Recepționeer



The screenshot shows a window titled 'Receptioner' with a menu bar containing 'Programare pacient', 'Emitere bon', and 'Date personale'. Below the menu bar are two tabs: 'General' and 'Inregistrare pacient'. The main area has a light yellow background and contains a large icon of a receptionist. To the right of the icon are three text input fields: 'Nume' (Name), 'Prenume' (First Name), and 'Functie' (Function) with the value 'Receptioner'. At the bottom left is a 'Log Out' button.

## Fereastră Super-Admin

Administrare

General Inserare Stergere Modifica

Nume Coman

Prenume Mircea

Funcție Super-Administrator

Log Out

## Fereastră Inspector

Inspector

General Cautare angajat Date personale

Nume

Prenume

Funcție Inspector

Log Out

## Fereastră Contabil

Contabil

General Profit Policlinica Profit Medici Date Personale

Nume

Prenume

Funcție Contabil

Log Out

## Fereastră Asistent Medical

Asistent Medical

General Vizualizare istoric Raport medical Informatii personale

Nume

Prenume

Funcție Asistent medical

Log Out

## VII. Prezentarea modelului de interogare și a modului de implementare

În continuare vom prezenta câteva exemple de cod, cu trigger-e, proceduri și apelarea acestora în Java.

```
87 DELIMITER //
88 ● CREATE TRIGGER Completare_salarii
89 BEFORE INSERT ON venit_lunar
90 FOR EACH ROW
91 BEGIN
92     declare salariu_ang int ;
93     declare id_ang int default 1;
94
95     simple_loop: LOOP
96         set salariu_ang = -1;
97         select salariu into salariu_ang from angajat where id_angajat=id_ang;
98         if(salariu_ang = -1) THEN
99             LEAVE simple_loop;
100         END IF;
101         insert into salarii(id_angajat,suma, luna, an) values (id_ang,salariu_ang, new.luna, new.an);
102         set id_ang=id_ang+1;
103
104     END LOOP simple_loop;
105 END;
106 //
```

Triggerul de completare salarii asigură popularea tabelii "Salarii", astfel încât la fiecare inserare în tabelul „venit\_lunar” se vor completa salariile angajaților în luna respectivă. (Tabela venit\_lunar va fi completată la fiecare inserare a primei programări dintr-o lună.

Trigger-ul verifică salariul de bază al fiecărui angajat și îl adaugă în tabelă, modificări ulterioare fiind facute tot prin trigger-e, folosind update-uri unde e necesar (pentru calcularea salariului medicului în funcție de servicii).

```

68 DELIMITER //
69 • CREATE PROCEDURE print_concedii_angajat(IN id_angajat int,out result int)
70 BEGIN
71     DECLARE id_angajat_val INT ;
72     set id_angajat_val =-1;
73     SELECT id_angajat INTO id_angajat_val FROM angajat a, concedii c WHERE c.id_angajat= id_angajat LIMIT 1;
74
75     if (id_angajat_val=-1) then set result =1;
76     else
77         set result =0;
78         Select * from concedii where id_angajat_val=concedii.id_angajat;
79     end if;
80 END;
81 //
82 DELIMITER ;

```

Procedura de mai sus verifică dacă un angajat, introdus în funcție de angajat\_id se află în concediu, informație care ne ajută la disponibilitatea medicilor la o programare.

Ca mod de funcționare, procedura testează inițial valabilitatea id\_angajat prin care a fost apelată procedura. Dacă se găsește un concediu cu acest id, atunci rezultatul returnat va fi '1' și se vor afișa toate concediile utilizatorului respectiv.

Apelarea în Java a procedurii efectuate mai sus:

```

// Apelul procedurii stocate pentru a obtine concediile angajatului
String callProcedure = "{call print_concedii_angajat(?, ?)}";
CallableStatement callableStatement = connection.prepareCall(callProcedure);
callableStatement.setInt( parameterIndex: 1, idAngajat);
callableStatement.registerOutParameter( parameterIndex: 2, Types.INTEGER);

callableStatement.execute();

```

## VIII. Metode de securizare a aplicației

Aplicatia este securizata prin intermediul procesului de autentificare si autorizare. Conectarea la aplicatie se face prin intermediul unei parole si username. Daca acestea sunt incorecte nu se va permite utilizatorul sa treaca la o alta fereastră. Singurul mod prin care se poate deschide aplicatia este prin intermediul clasei LoginFrom. Daca cumva se reuseste deschiderea unei alte clasa , de exemplu cea de Medic, intrusul nu va putea efectua nici o procedura disponibila medicului precum modificarea sau adaugarea unui raport medical , deoarece toate aceste procese se efectueaza pe baza id-ului de angajat care se obtine in momentul unei logari efectuate cu succes. Accesul la functionalitatile si resursele aplicatiei este controlat in functie de rolul si privilegiile fiecarui utilizator.

Alte idei conceptuale pentru o prevenție bună ar fi dezvoltarea unui algoritm de generare al parolelor, astfel încât pentru fiecare persoana pe care administratorul o introduce ca user în baza de date, va fi generata o parola complexă, diminuând astfel posibilitatea de a fi spart vreun cont.

De asemenea, faptul că doar administratorul și superadministratorul pot face modificări care sa afecteze buna funcționare a bazei de date scade de asemenea posibilitatea ca baza de date sa fie spartă prin interfață.

O alta idee de implementare, care completează algoritmul menționat mai sus, este actualizarea parolelor la o perioada prestabilită.



## IX. Interfață

Pentru crearea interfeței am folosit JavaSwing. Java Swing UI Designer este o facilitate sau unealtă care face parte din medii de dezvoltare integrate (IDE) precum IntelliJ IDEA și Eclipse și care facilitează crearea interfeței utilizator grafice (GUI) pentru aplicații Java bazate pe platforma Swing. Pentru unele ferestre am folosit GUI Designer, iar pentru altele am folosit cod în ceea ce privește coordonatele afișării fieldurilor.

Pentru fiecare buton am scris un ButtonListener care va face modificările cerute în urma apăsării. Pentru fiecare câmp am extras informația acolo unde este necesar și am stocat-o în variabile, sau am trimis-o direct ca rezultat al funcției de extragere.



Folosirea GUI Form

```
public Medic2() {
    System.out.println("(" + id_angajat);
    // initComponents(); // Metoda pentru a initializa componentele din form

    JFrame frame = new JFrame( title: "Medic2");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setContentPane(General);

    frame.pack();
    frame.setVisible(true);

    // Adaugare listener pentru butonul de adaugare serviciu
    AdaugaServiciu.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) { adaugaServiciu(); }
    });
}
```

## X. Bibliografie

[1] Java GUI Tutorial - [https://www3.ntu.edu.sg/home/ehchua/programming/java/jG4a\\_gui.html](https://www3.ntu.edu.sg/home/ehchua/programming/java/jG4a_gui.html)

[2] Teorie Triggere și Proceduri -  
[https://ftp.utcluj.ro/pub/users/civan/IBD/1.LABORATOARE/L6\\_P&T/IBD6\\_Lab.pdf](https://ftp.utcluj.ro/pub/users/civan/IBD/1.LABORATOARE/L6_P&T/IBD6_Lab.pdf)

[3] GUI - <https://www.guru99.com/java-swing-gui.html>

[4] Tutorial GUI - [https://www.youtube.com/watch?v=nIQatlpL\\_GE](https://www.youtube.com/watch?v=nIQatlpL_GE)

[5] Tutorial GUI2 - [https://www.youtube.com/watch?v=whF\\_Qm1epQ8](https://www.youtube.com/watch?v=whF_Qm1epQ8)

[6] Date and TimeStamp MySQL -  
<https://dev.mysql.com/doc/refman/8.0/en/datetime.html>