



**FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE DEPARTAMENTUL CALCULATOARE**

**Proiect**

la disciplina

Baze de Date

Prof. Cosmina Daniela Ivan

**PROIECT la DISCIPLINA INTRODUCERE IN BAZE DE DATE**

**Bogdan Alexandru Birlea**

**Ioan Doru Boca**

**Alexandru Marian**

**Grupa: 30224**

**An Academic: 2023-2027**

Cuprins

[1. Introducere 3](#_Toc187723622)

[1.1 Descrierea Generală a Proiectului 3](#_Toc187723623)

[1.2 Scopul și Obiectivele 3](#_Toc187723624)

[2. Tema Proiectului 4](#_Toc187723625)

[2.1 Prezentarea Aplicației 4](#_Toc187723626)

[2.2 Funcționalități și Utilizatori 5](#_Toc187723627)

[3. Descrierea Bazei de Date 6](#_Toc187723628)

[3.1 Structura Generală 7](#_Toc187723629)

[3.2 Detalii Specifice 8](#_Toc187723630)

[3.3 Nivelul de Normalizare 8](#_Toc187723631)

[4. Elemente de Programare a Funcționalităților 9](#_Toc187723632)

[4.1 Proceduri, Triggere și Vederi 9](#_Toc187723633)

[4.2 Alte exemple de cod SQL 11](#_Toc187723634)

[**5**. Interfața Grafică a Utilizatorului (GUI) - Receptioner 12](#_Toc187723635)

[5.1 Descrierea Funcționalităților- Receptioner 12](#_Toc187723636)

[5.2 Descrierea Funcționalităților - Medic 12](#_Toc187723637)

[5.3 Descrierea Funcționalităților-Asistent medical 13](#_Toc187723638)

[5.4 Descrierea Funcționalităților-Expert financiar 14](#_Toc187723639)

[5.5 Descrierea Funcționalităților 14](#_Toc187723640)

[5.6 Descrierea Funcționalităților-Admin 15](#_Toc187723641)

[5.7 Detalii de Implementare 15](#_Toc187723642)

[6. Manual de Utilizare 16](#_Toc187723643)

[6.1 Înregistrare și Autentificare 16](#_Toc187723644)

[6.2 Gestionarea Informațiilor per Utilizatori 16](#_Toc187723645)

[7. Concluzii și Dezvoltări Ulterioare 17](#_Toc187723646)

[7.1 Analiza Funcționalităților Curente 17](#_Toc187723647)

[7.2 Dezvoltări Ulterioare 18](#_Toc187723648)

# 1. Introducere

## 1.1 Descrierea Generală a Proiectului

Acest proiect își propune dezvoltarea unui sistem informatic complex pentru gestionarea activităților dintr-un lanț de policlinici. Sistemul va fi construit utilizând o bază de date MySQL și o interfață grafică prietenoasă, care va permite utilizatorilor să interacționeze cu sistemul și să efectueze diverse operațiuni în funcție de rolurile și permisiunile lor.

**Caracteristici cheie ale sistemului:**

* **Gestiunea utilizatorilor:** Sistemul va gestiona diferite tipuri de utilizatori, inclusiv administratori, super-administratori, medici, asistenți medicali, recepționeri și personal din departamentele de resurse umane și financiar-contabil. Fiecare tip de utilizator va avea drepturi de acces specifice pentru a asigura securitatea datelor și a funcționalităților.
* **Module funcționale:** Sistemul va fi împărțit în module distincte pentru a gestiona resursele umane, operațiunile financiar-contabile și activitățile operaționale, cum ar fi programările pacienților, înregistrarea pacienților, emiterea bonurilor fiscale, completarea rapoartelor medicale și gestionarea istoricului medical al pacienților.
* **Interfață grafică:** O interfață grafică intuitivă va fi dezvoltată pentru a facilita interacțiunea utilizatorilor cu sistemul, permițându-le să acceseze și să opereze diversele funcționalități într-un mod eficient.
* **Controlul accesului:** Sistemul va implementa un mecanism de control al accesului bazat pe roluri (RBAC), care va asigura că utilizatorii au acces doar la informațiile și funcționalitățile relevante pentru rolurile lor.

## 1.2 Scopul și Obiectivele

**Scopul principal** al acestui proiect este de a crea un sistem informatic complet și robust, care să optimizeze și să automatizeze diversele activități desfășurate într-un lanț de policlinici.

**Obiectivele specifice** ale proiectului includ:

* **Proiectarea unei baze de date relaționale:** Baza de date va fi proiectată utilizând MySQL și va fi normalizată pentru a elimina redundanța datelor și a asigura integritatea datelor.
* **Dezvoltarea unei interfețe grafice:** Interfața grafică va fi dezvoltată utilizând Java și va oferi o experiență de utilizare intuitivă și eficientă.
* **Implementarea controlului accesului bazat pe roluri:** Sistemul va asigura că utilizatorii au acces doar la informațiile și funcționalitățile relevante pentru rolurile lor.
* **Implementarea modulelor funcționale:** Modulele funcționale vor acoperi diverse aspecte ale activităților policlinicii, inclusiv gestiunea resurselor umane, operațiunile financiar-contabile și activitățile operaționale.
* **Testarea și validarea sistemului:** Sistemul va fi testat temeinic pentru a se asigura că îndeplinește cerințele specificate și că funcționează corect.

# 2. Tema Proiectului

Tema proiectului este dezvoltarea unui sistem informatic complex pentru gestionarea activităților dintr-un lanț de policlinici. Sistemul va fi utilizat de către o varietate de utilizatori, inclusiv administratori, medici, asistenți medicali, recepționeri și personal din departamentele de resurse umane și financiar-contabil. Sistemul va oferi o gamă largă de funcționalități, inclusiv:

* **Gestiunea angajaților:** adăugarea, modificarea și ștergerea informațiilor despre angajați, gestionarea programului de lucru și a concediilor, calcularea salariilor.
* **Gestiunea pacienților:** programarea pacienților, înregistrarea pacienților, gestionarea istoricului medical al pacienților.
* **Gestiunea serviciilor medicale:** definirea serviciilor medicale, programarea pacienților pentru servicii medicale, completarea rapoartelor medicale.
* **Gestiunea financiară:** calcularea profitului, generarea de rapoarte financiare.

Sistemul va fi dezvoltat utilizând o bază de date MySQL și o interfață grafică Java. Se va pune accent pe implementarea unui sistem de control al accesului bazat pe roluri (RBAC) pentru a asigura securitatea datelor și a funcționalităților.

**Detalii suplimentare despre tema proiectului:**

* Sistemul va trebui să fie modular, permițând adăugarea de noi funcționalități în viitor.
* Sistemul va trebui să fie robust și fiabil, capabil să gestioneze un volum mare de date.
* Interfața grafică va trebui să fie intuitivă și ușor de utilizat.
* Sistemul va trebui să fie securizat, protejând datele sensibile ale pacienților și angajaților.

Această temă de proiect oferă studenților o oportunitate excelentă de a-și aplica cunoștințele de baze de date, programare Java și interfețe grafice pentru a crea un sistem informatic complex și util. Proiectul va contribui, de asemenea, la dezvoltarea abilităților de lucru în echipă și de gestionare a proiectelor.

* Documente citite

## 2.1 Prezentarea Aplicației

Aplicația dezvoltată în cadrul acestui proiect este un sistem informatic complex pentru gestionarea activităților dintr-un lanț de policlinici. Construită utilizând o bază de date MySQL și o interfață grafică Java, aplicația permite utilizatorilor să interacționeze cu sistemul și să efectueze diverse operațiuni în funcție de rolurile lor.

Aplicația este concepută pentru a fi utilizată de o varietate de utilizatori, inclusiv:

* **Administratori:** care au drepturi complete de gestionare a utilizatorilor și a datelor din sistem.
* **Super-administratori:** care au drepturi suplimentare față de administratori, putând opera și asupra utilizatorilor de tip administrator.
* **Medici:** care pot vizualiza programările, istoricul medical al pacienților și pot completa rapoarte medicale.
* **Asistenți medicali:** care pot vizualiza programările, pacienții înregistrați și pot completa rapoarte de analize medicale.
* **Recepționeri:** care pot gestiona programările pacienților, pot înregistra pacienții și pot emite bonuri fiscale.
* **Personal din departamentele de resurse umane și financiar-contabil:** care au acces la modulele specifice pentru gestiunea resurselor umane, respectiv pentru operațiuni financiar-contabile.

Aplicația oferă o serie de funcționalități menite să optimizeze și să automatizeze activitățile desfășurate într-un lanț de policlinici, cum ar fi:

* **Gestiunea utilizatorilor:** adăugarea, modificarea și ștergerea utilizatorilor, gestionarea rolurilor și a drepturilor de acces.
* **Gestiunea programărilor:** programarea pacienților la medici, vizualizarea programărilor, modificarea și anularea programărilor.
* **Gestiunea istoricului medical:** înregistrarea și vizualizarea informațiilor medicale ale pacienților, inclusiv rapoarte medicale și rezultate ale analizelor.
* **Gestiunea financiară:** generarea de rapoarte financiare, calcularea profitului, vizualizarea salariilor.
* **Gestiunea resurselor umane:** gestionarea programului de lucru al angajaților, gestionarea concediilor, calcularea salariilor.

## 2.2 Funcționalități și Utilizatori

Aplicația este structurată în jurul a trei module principale:

1. **Modulul pentru gestiunea resurselor umane:**
   * Permite inspectorilor de resurse umane să gestioneze programul de lucru al angajaților, să adauge și să editeze concedii.
   * Permite angajaților să vizualizeze propriul orar și informații despre concedii.
2. **Modulul pentru operații financiar-contabile:**
   * Permite experților financiari să vizualizeze profitul realizat de lanțul de policlinici, defalcat pe venituri și cheltuieli.
   * Permite vizualizarea profitului pentru un anumit medic, locație sau specialitate medicală.
   * Permite angajaților să vizualizeze salariile obținute în lunile precedente.
   * Permite medicilor să consulte profitul generat.
3. **Modulul pentru gestiunea activităților operaționale:**
   * Permite recepționerilor să programeze pacienți la medici, să înregistreze pacienții în momentul prezentării la clinică și să emită bonuri fiscale.
   * Permite asistenților medicali să completeze rapoarte pentru analizele medicale.
   * Permite medicilor să vizualizeze pacienții programați, să consulte istoricul medical al pacienților și să completeze rapoarte medicale.

# 3. Descrierea Bazei de Date

Baza de date este implementată în MySQL și conține următoarele tabele:

* **Utilizator:** stochează informații despre utilizatorii aplicației (CNP, nume, prenume, adresă, număr de telefon, email, cont IBAN, număr contract, data angajării, funcție, parolă).
* **Medical:** stochează informații suplimentare despre angajații din domeniul medical (salariu, număr de ore contractuale).
* **Medic:** stochează informații specifice medicilor (specialitate, grad, cod parafă, procent servicii, titlu științific, post didactic).
* **AsistentMedical:** stochează informații specifice asistenților medicali (tip, grad).
* **Receptioner:** stochează informații specifice recepționerilor (id unitate medicală).
* **UnitateMedicala:** stochează informații despre unitățile medicale din lanțul de policlinici (denumire, adresă).
* **CabinetMedical:** stochează informații despre cabinetele medicale din fiecare unitate medicală (id unitate medicală, denumire).
* **ServiciuMedical:** stochează informații despre serviciile medicale oferite de lanțul de policlinici (denumire, necesită competență, preț, durată).
* **Competenta:** stochează informații despre competențele necesare pentru anumite servicii medicale (denumire).
* **Pacient:** stochează informații despre pacienți (nume, prenume, data nașterii, adresă, număr de telefon, email).
* **Programare:** stochează informații despre programările pacienților la medici (zi, oră, CNP medic, CNP recepționer, CNP asistent, id pacient, durată, bon, preț).
* **RaportMedical:** stochează informații despre rapoartele medicale (id pacient, CNP medic, CNP medic recomandare, CNP asistent, zi, simptome, investigații, diagnostic, recomandări, rezultat, cod parafă).
* **Concediu:** stochează informații despre concediile angajaților (CNP angajat, data început, data sfârșit).
* **OrarGeneric:** stochează informații despre orarul generic al angajaților (CNP, zi, oră început, oră sfârșit).
* **OrarSpecific:** stochează informații despre orarul specific al angajaților (CNP, zi, oră început, oră sfârșit).
* **OrarGenericUnitate:** stochează informații despre orarul generic al unităților medicale (id unitate, zi, oră început, oră sfârșit).
* **Medic\_Competenta:** tabel de legătură între Medic și Competență.
* **UnitateMedicala\_Medic:** tabel de legătură între UnitateMedicala și Medic.
* **CabinetMedical\_ServiciuMedical:** tabel de legătură între CabinetMedical și ServiciuMedical.
* **Serviciu\_Programare:** tabel de legătură între ServiciuMedical și Programare.
* **Serviciu\_Raport:** tabel de legătură între ServiciuMedical și RaportMedical.

Relațiile între tabele sunt definite prin intermediul cheilor primare și străine, asigurând integritatea datelor.

* Documente citite

## 3.1 Structura Generală

Baza de date este concepută pentru a stoca informații despre utilizatori, angajați medicali, unități medicale, programări, rapoarte medicale și alte date relevante pentru gestionarea activităților policlinicii.

Diagrama UML a bazei de date este disponibilă în fișierul "proiectuml.pdf".

**Tabele principale:**

* **Utilizator:** Tabelul principal care stochează informații generale despre toți utilizatorii sistemului, indiferent de rolul lor specific.
* **Medical:** Conține informații specifice personalului medical, moștenind și informațiile generale din tabelul 'Utilizator'.
* **Medic:** Include informații suplimentare pentru medici, cum ar fi specialitatea și gradul profesional.
* **AsistentMedical:** Include informații specifice pentru asistenții medicali.
* **Receptioner:** Include informații specifice pentru recepționeri.
* **UnitateMedicala:** Stochează datele despre fiecare unitate medicală din lanțul de policlinici.
* **Pacient:** Conține informații despre pacienții înregistrați în sistem.
* **Programare:** Stochează detaliile despre programările pacienților.
* **RaportMedical:** Salvează informațiile din rapoartele medicale generate.

**Tabele auxiliare:**

* **Concediu:** Înregistrează concediile angajaților.
* **OrarGeneric & OrarSpecific:** Stochează informații despre orarul de lucru.
* **ServiciuMedical:** Descrie serviciile medicale oferite (consultații, analize, etc.).
* **Competenta:** Definește competențele necesare pentru anumite servicii.

## 3.2 Detalii Specifice

* **Chei primare:** Fiecare tabel are o cheie primară pentru a identifica în mod unic fiecare înregistrare.
* **Chei străine:** Se utilizează chei străine pentru a crea relații între tabele și pentru a asigura integritatea referențială a datelor.
* **Tipuri de date:** Sunt utilizate tipuri de date adecvate pentru fiecare coloană, cum ar fi VARCHAR, INT, DATE, TIME, DECIMAL, BOOLEAN.
* **Constrângeri:** Sunt definite constrângeri pentru a asigura validitatea datelor, cum ar fi NOT NULL, UNIQUE.

## 3.3 Nivelul de Normalizare

Baza de date este proiectată să respecte cel puțin **a treia formă normală (3NF)**. Aceasta înseamnă că:

* **1NF:** Nu există grupuri de coloane care se repetă.
* **2NF:** Toate atributele non-cheie sunt complet dependente de cheia primară.
* **3NF:** Nu există dependențe tranzitive între atributele non-cheie.

Normalizarea ajută la:

* **Eliminarea redundanței datelor:** Se reduce spațiul de stocare necesar și se evită inconsistențele datelor.
* **Îmbunătățirea integrității datelor:** Se asigură că datele sunt corecte și consistente.
* **Simplificarea interogărilor:** Se facilitează regăsirea și manipularea datelor.

**Observații:**

* În anumite situații, s-ar putea ca, din motive de performanță, să se aplice denormalizări controlate.
* Pentru a asigura un nivel și mai ridicat de normalizare, s-ar putea lua în considerare și formele normale Boyce-Codd (BCNF) sau a patra formă normală (4NF), însă acest lucru ar putea crește complexitatea bazei de date
* Documente citite

# 4. Elemente de Programare a Funcționalităților

## 4.1 Proceduri, Triggere și Vederi

Pentru a facilita implementarea funcționalităților aplicației și pentru a asigura integritatea datelor, s-au utilizat proceduri stocate, triggere și vederi în baza de date.

**Proceduri stocate:**

Procedurile stocate sunt blocuri de cod SQL precompilate care pot fi apelate după nume. Ele sunt utilizate pentru a automatiza sarcini repetitive, pentru a îmbunătăți performanța și pentru a asigura securitatea datelor.

**Lista completă de proceduri stocate:**

* insert\_utilizator: Inserează un nou utilizator în tabelul 'Utilizator'.
* insert\_medical: Inserează un nou angajat medical în tabelul 'Medical'.
* insert\_medic: Inserează un nou medic în tabelul 'Medic'.
* insert\_asistent: Inserează un nou asistent medical în tabelul 'AsistentMedical'.
* insert\_receptioner: Inserează un nou recepționer în tabelul 'Receptioner'.
* delete\_utilizator\_cascade: Șterge un utilizator și toate datele asociate acestuia din tabelele corelate.
* Updateutilizator: Actualizează informațiile despre un utilizator existent.
* oreLucrate: Calculează numărul de ore lucrate de un angajat.
* CalculProfitMedic1: Calculează profitul generat de un anumit medic într-o lună specificată.
* CalculProfitPeSpecialitate: Calculează profitul total generat de o anumită specialitate medicală într-o lună specificată.
* CalculProfitUnitate1: Calculează profitul total generat de o anumită unitate medicală într-o lună specificată.

**Exemple de proceduri stocate:**

* Procedura insert\_utilizator:

SQL

DELIMITER $$

CREATE PROCEDURE insert\_utilizator(

IN p\_CNP VARCHAR(13),

IN p\_nume VARCHAR(50),

IN p\_prenume VARCHAR(50),

-- ... alte variabile ...

)

BEGIN

INSERT INTO proiect3.utilizator (CNP, nume, prenume, ...)

VALUES (p\_CNP, p\_nume, p\_prenume, ...);

END$$

DELIMITER ;

* Procedura delete\_utilizator\_cascade:

SQL

DELIMITER $$

CREATE PROCEDURE delete\_utilizator\_cascade(IN cnp\_to\_delete VARCHAR(13))

BEGIN

-- Delete from all related tables

DELETE FROM proiect3.serviciu\_programare WHERE id\_programare IN (SELECT id FROM proiect3.programare WHERE CNP\_asistent = cnp\_to\_delete OR CNP\_medic = cnp\_to\_delete OR CNP\_receptioner = cnp\_to\_delete);

-- ... alte ștergeri ...

DELETE FROM proiect3.utilizator WHERE CNP = cnp\_to\_delete;

END$$

DELIMITER ;

**Triggere:**

Triggerele sunt proceduri stocate care se execută automat ca răspuns la anumite evenimente din baza de date, cum ar fi inserarea, actualizarea sau ștergerea unei înregistrări.

**Vederi:**

Vederile sunt tabele virtuale create pe baza rezultatelor unei interogări. Ele sunt utilizate pentru a simplifica interogările complexe și pentru a oferi o modalitate securizată de acces la date.

**Lista completă de vederi:**

* view\_angajati\_disponibili: Afișează angajații care nu sunt programați la o anumită dată și oră.

**Exemplu de vedere:**

SQL

CREATE VIEW view\_angajati\_disponibili AS

SELECT u.CNP, u.nume, u.prenume

FROM Utilizator u

LEFT JOIN Programare p ON u.CNP = p.CNP\_medic

WHERE p.CNP\_medic IS NULL;

## 4.2 Alte exemple de cod SQL

Pe lângă procedurile stocate, triggerele și vederile menționate mai sus, s-au utilizat și alte instrucțiuni SQL pentru a implementa funcționalitățile aplicației.

**Exemple:**

* Interogare pentru a obține lista de pacienți programați la un anumit medic:

SQL

SELECT p.nume, p.prenume

FROM Pacient p

INNER JOIN Programare pr ON p.id = pr.id\_pacient

WHERE pr.CNP\_medic = '1234567890123';

* Interogare pentru a calcula venitul total generat de un anumit serviciu medical:

SQL

SELECT s.denumire, SUM(s.pret) AS venit\_total

FROM ServiciuMedical s

INNER JOIN Serviciu\_Programare sp ON s.id = sp.id\_serviciu

INNER JOIN Programare pr ON sp.id\_programare = pr.id

GROUP BY s.denumire;

Acestea sunt doar câteva exemple de cod SQL utilizat în cadrul aplicației. Baza de date conține o varietate de instrucțiuni SQL pentru a implementa toate funcționalitățile necesare.

* Documente citite

# **5**. Interfața Grafică a Utilizatorului (GUI) - Receptioner

Interfața grafică pentru recepționer este concepută pentru a facilita interacțiunea cu sistemul și a oferi acces rapid la funcționalitățile specifice rolului său.

## 5.1 Descrierea Funcționalităților- Receptioner

GUI-ul recepționerului include următoarele funcționalități cheie:

* **Programarea pacienților:**
  + Permite recepționerului să programeze pacienții la medici.
  + Interfața include câmpuri pentru selectarea pacientului, a medicului, a datei și a orei.
  + Se afișează un calendar interactiv pentru alegerea datei.
  + Recepționerul poate vizualiza disponibilitatea medicilor și poate alege intervalul orar dorit.
* **Înregistrarea pacienților:**
  + Permite recepționerului să înregistreze noi pacienți în sistem.
  + Interfața include câmpuri pentru introducerea datelor personale ale pacientului (nume, prenume, adresă, telefon, etc.).
  + Se realizează validarea datelor introduse pentru a asigura corectitudinea informațiilor.
* **Emiterea de bonuri fiscale:**
  + Permite recepționerului să emită bonuri fiscale pentru serviciile medicale prestate.
  + Interfața afișează lista serviciilor medicale efectuate și permite selectarea celor pentru care se emite bonul.

## 5.2 Descrierea Funcționalităților - Medic

GUI-ul medicului include următoarele funcționalități principale:

* **Vizualizarea programărilor:**
  + Permite medicului să vizualizeze programările sale.
  + Interfața afișează o listă cu programările viitoare, inclusiv data, ora, pacientul programat și serviciile medicale solicitate.
  + Medicul poate filtra programările după dată sau pacient.
* **Completarea rapoartelor medicale:**
  + Permite medicului să completeze rapoarte medicale pentru pacienții consultați.
  + Interfața include câmpuri pentru introducerea diagnosticului, a simptomelor, a investigațiilor și a recomandărilor.
* **Vizualizarea istoricului medical al pacienților:**
  + Permite medicului să acceseze istoricul medical al pacienților.
  + Interfața afișează informații despre diagnosticele anterioare, tratamentele prescrise și rezultatele analizelor medicale.
  + Medicul poate filtra istoricul medical după dată sau tipul de informație.
* **Consultarea profitului generat:**
  + Permite medicului să vizualizeze profitul generat de activitatea sa.
  + Interfața afișează un grafic cu profitul generat pe o perioadă de timp specificată.
  + Medicul poate filtra profitul după specialitate sau unitate medicală.

## 5.3 Descrierea Funcționalităților-Asistent medical

GUI-ul asistentului medical include următoarele funcționalități principale:

* **Gestionarea Pacienților:**
  + Permite asistentului să vizualizeze și să selecteze pacienții.
  + Asistentul poate vizualiza atât pacienții înregistrați în sistem, cât și pe cei programați la medicul la care asistă.
  + Interfața afișează o listă cu pacienții, inclusiv numele, prenumele și alte detalii relevante.
  + Asistentul poate selecta un pacient pentru a accesa informații suplimentare sau pentru a completa un raport de analiză medicală.
* **Completarea Rapoartelor de Analiză Medicală:**
  + Permite asistentului să completeze rapoarte pentru analizele medicale efectuate.
  + Interfața include câmpuri pentru introducerea rezultatelor analizelor și a observațiilor relevante.

## 5.4 Descrierea Funcționalităților-Expert financiar

GUI-ul expertului financiar include următoarele funcționalități principale:

* **Vizualizarea Salariilor Angajaților:**
  + Permite expertului financiar să vizualizeze salariile angajaților.
  + Interfața afișează o listă cu angajații, inclusiv numele, prenumele și salariul lunar.
  + Expertul financiar poate filtra angajații după funcție (medic, asistent medical).
* **Analiza Profitului:**
  + Permite expertului financiar să analizeze profitul generat de diferitele unitati medicale, specializări sau medici.
  + Interfața afișează luyna urmata de profitul pe acea luna.
  + Expertul financiar poate selecta luna pentru analiză .
* **Generarea de Rapoarte Financiare:**
  + Interfața oferă o serie de opțiuni pentru personalizarea rapoartelor, cum ar fi selectarea perioadei.

## 5.5 Descrierea Funcționalităților

GUI-ul Resurse Umane include următoarele funcționalități principale:

* **Gestionarea Angajaților:**
  + Permite vizualizarea și căutarea angajaților.
  + Afișează o listă cu angajații, inclusiv numele, prenumele, funcția și alte detalii relevante.
  + Permite căutarea angajaților după nume, prenume sau funcție.
* **Gestionarea Concediilor:**
  + Permite vizualizarea concediilor angajaților.
  + Afișează o listă cu concediile aprobate,.
  + Permite adăugarea de noi concedii pentru angajați.
  + Oferă opțiunea de a edita sau șterge concedii existente.
* **Gestionarea Orarului de Lucru:**
  + Permite vizualizarea orarului de lucru al angajaților.
  + Afișează orarul generic și specific al fiecărui angajat.
  + Permite modificarea orarului de lucru, atât generic cât și specific.

## 5.6 Descrierea Funcționalităților-Admin

GUI-ul Admin și Super Admin include următoarele funcționalități principale:

* **Gestionarea Utilizatorilor:**
  + Permite vizualizarea, adăugarea, editarea și ștergerea utilizatorilor.
  + Afișează o listă cu toți utilizatorii din sistem, inclusiv numele, prenumele, rolul și alte detalii relevante.
  + Permite adăugarea de noi utilizatori cu diferite roluri (medic, asistent medical, recepționer, etc.).
  + Oferă opțiunea de a edita informațiile utilizatorilor existenți, cum ar fi schimbarea adresei de email sau a rolului.
  + Permite ștergerea utilizatorilor din sistem.
  + Super Admin are drepturi suplimentare față de Admin, putând să editeze și să șteargă utilizatori de tip Admin.

## 5.7 Detalii de Implementare

* **Ferestre modale:** Pentru operații precum programarea pacienților sau înregistrarea de noi pacienți, se utilizează ferestre modale care se suprapun peste fereastra principală.
* **Componente GUI:** Se utilizează o varietate de componente GUI, cum ar fi:
  + JTextField - pentru introducerea de text (nume, prenume, etc.)
  + JButton - pentru acțiuni (programare, înregistrare, emitere bon, etc.)
  + JTable - pentru afișarea de date în format tabelar (lista pacienți, lista programări, etc.)
  + JLabel - pentru afișarea de text informativ.
* **Gestionarea evenimentelor:** Se utilizează ascultători de evenimente (ActionListener, ItemListener, etc.) pentru a gestiona acțiunile utilizatorului (click pe butoane, selectarea de opțiuni, etc.).
* **Validarea datelor:** Se implementează validarea datelor introduse de utilizator pentru a preveni erorile și a asigura integritatea datelor.
* **Conexiunea la baza de date:** Se utilizează o clasă separată (Conexiune.java) pentru gestionarea conexiunii la baza de date.
* **Clasele model:** Se utilizează clase model pentru a reprezenta datele din baza de date (pacienți, programări, etc.).
* **Structura de layout:** Se utilizează layout managers (BorderLayout, FlowLayout, GridLayout, etc.) pentru a aranja componentele GUI într-un mod organizat.

# 6. Manual de Utilizare

## 6.1 Înregistrare și Autentificare

**Înregistrarea:**

* **Administratorul sistemului este responsabil pentru înregistrarea noilor utilizatori.**
* **Administratorul accesează un formular dedicat pentru înregistrarea utilizatorilor.**
* **În formular, administratorul completează informațiile personale și profesionale ale noului utilizator, inclusiv numele, prenumele, adresa de email, funcția, etc.**
* **Administratorul alege un nume de utilizator și o parolă pentru noul utilizator.**
* **Înregistrarea este finalizată prin apăsarea butonului "Înregistrare" din formularul de înregistrare.**

**Autentificarea:**

* Utilizatorii înregistrați se pot autentifica în aplicație utilizând email-ul și parola alese la înregistrare.
* După introducerea datelor de autentificare, utilizatorul trebuie să apese butonul "Autentificare".
* În cazul în care datele de autentificare sunt corecte, utilizatorul va fi redirecționat către pagina principală corespunzătoare rolului său.

## 6.2 Gestionarea Informațiilor per Utilizatori

Aplicația oferă o serie de funcționalități specifice fiecărui tip de utilizator:

* **Pagina principală:**
  + Această pagină este comună tuturor utilizatorilor și oferă acces la informații generale, cum ar fi data și ora curentă, precum și la un meniu cu opțiuni specifice rolului utilizatorului.
  + Utilizatorii se pot deloga din aplicație prin intermediul butonului "Logout".
* **Resurse Umane:**
  + Această pagină este dedicată utilizatorilor de tip Resurse Umane și oferă funcționalități pentru gestionarea angajaților, a concediilor și a orarului de lucru.
  + Utilizatorii pot vizualiza, adăuga, edita și șterge informații despre angajați, pot gestiona concediile și pot modifica orarul de lucru.
* **Expert Financiar:**
  + Această pagină este dedicată utilizatorilor de tip Expert Financiar și oferă funcționalități pentru vizualizarea și analiza informațiilor financiare, cum ar fi salariile angajaților și profitul generat.
  + Utilizatorii pot genera rapoarte detaliate privind salariile și pot analiza profitul pe diferite perioade de timp.
* **Recepționer:**
  + Această pagină este dedicată utilizatorilor de tip Recepționer și oferă funcționalități pentru programarea pacienților, înregistrarea de noi pacienți și emiterea de bonuri fiscale.
  + Utilizatorii pot programa pacienții la medici, pot înregistra noi pacienți în sistem și pot emite bonuri fiscale pentru serviciile medicale prestate.
* **Medic:**
  + Această pagină este dedicată utilizatorilor de tip Medic și oferă funcționalități pentru vizualizarea programărilor, completarea rapoartelor medicale și vizualizarea istoricului medical al pacienților.
  + Utilizatorii pot vizualiza programările, pot completa rapoarte medicale pentru pacienții consultați și pot accesa istoricul medical al pacienților.
* **Asistent Medical:**
  + Această pagină este dedicată utilizatorilor de tip Asistent Medical și oferă funcționalități pentru gestionarea pacienților, completarea rapoartelor de analiză medicală și vizualizarea informațiilor medicale relevante.
  + Utilizatorii pot vizualiza și selecta pacienții, pot completa rapoarte pentru analizele medicale efectuate și pot accesa informații medicale relevante.

Aplicația este concepută pentru a oferi o experiență de utilizare personalizată în funcție de rolul fiecărui utilizator, asigurând accesul la informațiile și funcționalitățile relevante activității lor.

# 7. Concluzii și Dezvoltări Ulterioare

## 7.1 Analiza Funcționalităților Curente

Aplicația dezvoltată în cadrul acestui proiect oferă o gamă largă de funcționalități pentru gestionarea activităților dintr-un lanț de policlinici. Printre cele mai interesante soluții adoptate se numără:

* **Controlul accesului bazat pe roluri (RBAC):** Implementarea RBAC asigură că utilizatorii au acces doar la informațiile și funcționalitățile relevante rolurilor lor, sporind securitatea și eficiența sistemului.
* **Modularizarea aplicației:** Aplicația este structurată în module distincte, ceea ce facilitează dezvoltarea, întreținerea și extinderea ulterioară a sistemului.
* **Utilizarea procedurilor stocate și a triggerelor:** Aceste elemente de programare a bazei de date contribuie la asigurarea integrității datelor și la optimizarea performanței aplicației.
* **Interfața grafică intuitivă:** GUI-ul este conceput pentru a fi ușor de utilizat și de înțeles, oferind o experiență plăcută utilizatorilor.

## 7.2 Dezvoltări Ulterioare

Aplicația poate fi extinsă și îmbunătățită în viitor prin adăugarea de noi funcționalități, cum ar fi:

* **Generarea de rapoarte mai complexe:** Se pot adăuga funcționalități pentru generarea de rapoarte personalizate, care să ofere o analiză mai detaliată a datelor.
* **Integrarea cu alte sisteme:** Aplicația poate fi integrată cu alte sisteme informatice utilizate în domeniul medical, cum ar fi sistemele de imagistică medicală sau sistemele de laborator.
* **Implementarea unui sistem de notificări:** Se poate implementa un sistem de notificări pentru a informa utilizatorii despre evenimente importante, cum ar fi programări noi sau modificări ale orarului.
* **Dezvoltarea unei aplicații mobile:** Se poate dezvolta o aplicație mobilă pentru a permite utilizatorilor să acceseze sistemul de pe dispozitivele mobile.

Prin implementarea acestor dezvoltări ulterioare, aplicația poate deveni un instrument și mai puternic și mai util pentru gestionarea activităților dintr-un lanț de policlinici.