PROIECT LA PA- SDA - JAVA

PROGRAM CONTABILITATE

ECHIPA NUMARUL 10

TEMA 2

Constantin Florentiina-Claudia

Gr.4LF421,AIA II

Diumea Gabriele

Gr.4LF412,AIA II

Pintilie Georgian-Vasile

Gr.4LF412,AIA II

Avasiloaie Stefan

Gr.4LF412,AIA II

Brasov,2023

CUPRINS

CONTENTS

1.INTRODUCERE

Tema noastra consta in realizarea unui program pentru o firma de contabilitate pentru a monitoriza si stoca date despre angajati , clienti si proiectele realizate in cadrul firmei. Pentru realizarea acestui proiect , echipa CaliberTECH a imbinat diverse limbaje de programare si interfete realizand o baza de date pentru stocare pentru care se ofera access prin prisma programului “NetBeans”. Pentru realizarea bazelor de date am folosit programul “XAMPP”.

2.DIAGRAMA DE FLUX A PROCESELOR

2.1.REALIZAREA CONCEPUTLUI

Pentru inceput ne-am gandit la cum ar trebui sa arate aplicatia; ferestrele, butoanele, campurile de text necesare cat si ce ar trebui sa faca acestea si cum sa fie pozitionate in cadrul aplicatie.

Dupa ce am pus la punct aceste aspecte am continuat cu realizarea bazei de date teoretice; am stabilit entitatile necesare unei companii de contabilitate, cat si relatiile dintre acestea, realizand entitati de intersectie in cazul relatiilor many-to-many.

2.2.REALIZAREA DESIGNULUI

Dupa ce am stabilit conceptul aplicatiei, am trecut la realizarea designului in aplicatia Netbeans; am creat ferestrele necesare si le-am creat designul folosind fereastra de design a aplicatiei, ce permite realizarea unei interfete grafice intr-un mod simplu si eficient.

2.3.REALIZAREA BAZEI DE DATE

Odata ce am terminat layout-ul si designul aplicatiei, am inceput crearea bazei de date, prin intermediul aplicatiei XAMPP care ofera posibiltatea crearii acesteia pe un server local.

2.4.REALIZAREA LEGATURII SI A CODULUI

Dupa ce am pregatit baza de date cat si designului aplicatiei, am scris codul pentru realizarea legaturii intre interfata aplicatiei si baza de date pentru fiecare fereastra.

3.PROCESUL COMPLET DE CREARE

3.1 CONCEPTUL

Pentru aspectul aplicatiei in sine am trecut prin multe idei de design pana am ajuns la cel curent, luand in considerare necesitatile unei firme de contabalitate; acesta urmeaza a fi prezentat mai jos.

Pentru baza de date am stabilit de ce entitati vom avea nevoie, si am stabilit o baza de date simpla, care sa contina tabelele de baza necesare unui asemenea proiect. Acestea sunt:

- “angajat”, cu campurile ID_Angajat, nume, prenume, salariu si functie, campul ID_Angajat fiind cheia primara;
- “client”, cu campurile ID_Client, nume, prenume, contact si adresa, avand cheia primara ID_Client;
- “logindata” cu campurile ID_Angajat, username si password; acest tabel stocheaza datele de logare a angajatilor pe baza campului ID_Angajat, acesta fiind cheia straina a tabelului;
- “monthlytracker” cu campurile ID_Angajat, an, luna si interval_lucrat; aici se stocheaza zilele lucrate ale fiecarui angajat pe anul curent si luna introdusa de angajat, ID_Angajat fiind cheia straina a tabelului;
- “proiect” cu campurile ID_Proiect(cheie primara), ID_Angajat(cheie straina), ID_Client(cheie straina), detalii si data_start; acest tabel reprezinta entitatea de legatura intre client si firma noastra;

3.2 DESIGNUL

Am creat mai multe JFrame-uri reprezentand ferestrele programului, acestea fiind:

- LogIn, meniul de logare;
- MainMenu, meniul pentru administrator
- Employee_menu, meniul pentru angajati
- FindEmployee, o fereastra pentru administrator ce permite cautarea unui angajat dupa ID-ul acestuia;
- Afisare_Clienti, o fereastra pentru administrator care ii prezinta acestuia clientii care au apelat la serviciile firmei;
- Pontaj, o fereastra pentru angajati, ce le permite introducerea zilelor lucrate pe luna curenta;

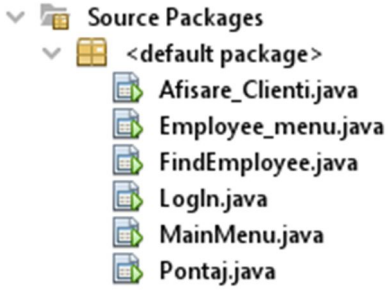

Fig. 1. Jframe-urile utilizate

In cele ce urmeaza vom prezenta designul fiecarui Jframe.

3.2.1 LOGIN

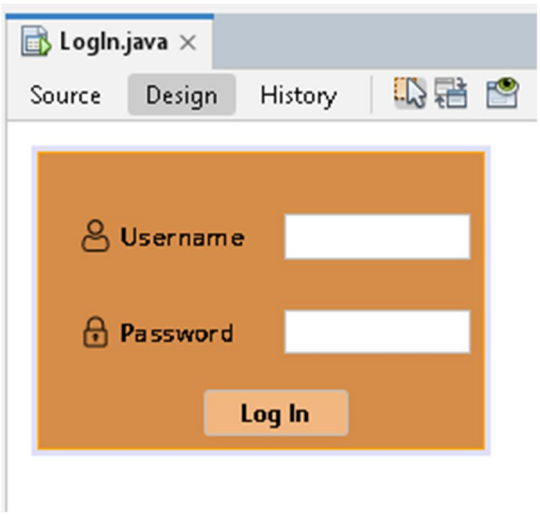

Fig. 2. Jframe-ul pentru meniul de logare

Acest JFrame reprezinta meniul de logare si contine:

- 2 JLable-uri, unul pentru a afisa “Username”, altul pentru a afisa “Password”;
- Langa cele 2 JLableuri am adaugat 2 JTextField-uri pentru a introduce datele de logare;
- Un JButton pentru a ne putea loga;

3.2.2 MainMenu

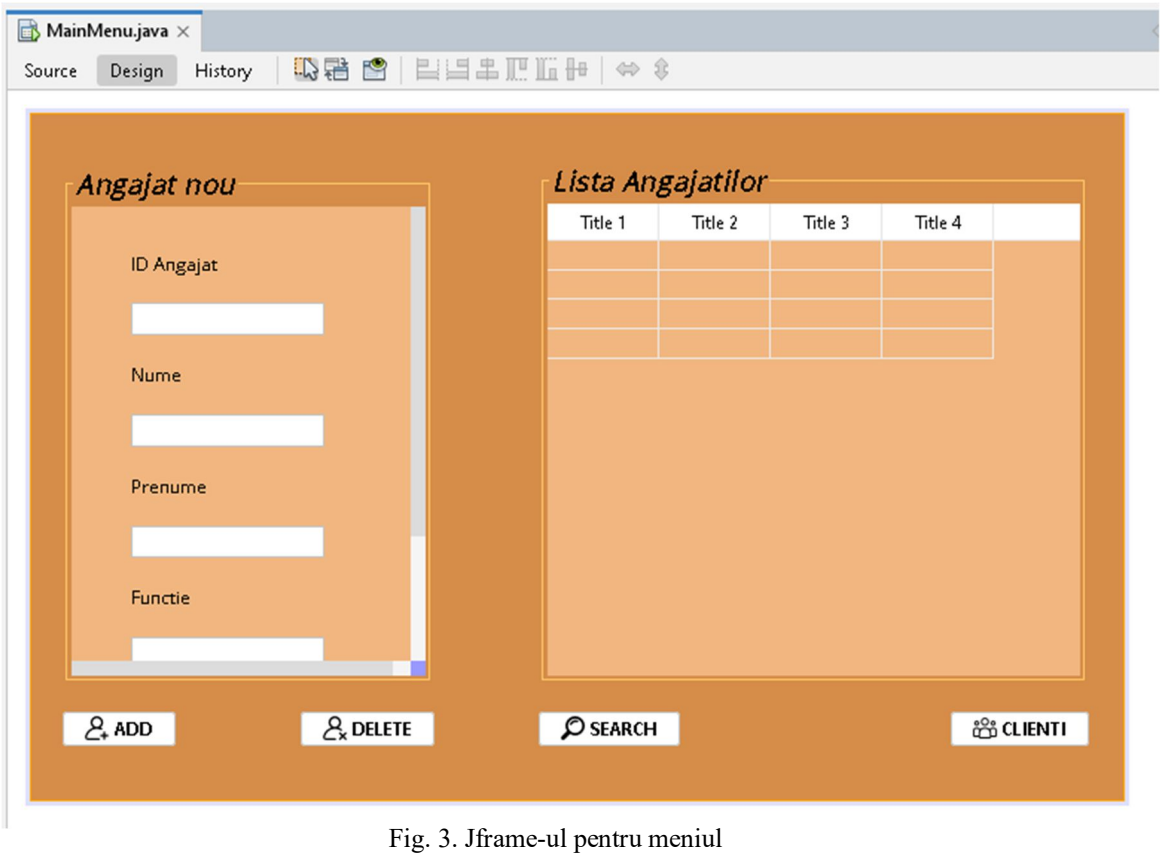

Fig. 3. Jframe-ul pentru meniul principal al administratorului

Acest JFrame reprezinta meniul principal pentru administrator si contine:

- Un JPanel principal ce contine toate celelalte elemente; acesta a fost adaugat pentru a stabili culoarea de fundal;
- Un JscrollPane cu multiple JLable-uri si JTextField-uri pentru a putea introduce datele unui nou angajat;
- Un JButton pentru a adaugat datele introduse anterior in baza noastra de date;
- Un JButton pentru a sterge angajati dupa ID-ul introdus;
- Un alt JPanel ce contine:
    - Un Jtable cu toti angajatii aflati in baza de date;
    - Un JButton pentru a cauta un angajat dupa ID-ul acestuia;
    - Un JButton pentru a afisa clientii ce au apelat la serviciile firmei;

3.2.3 EMPLOYEE_MENU

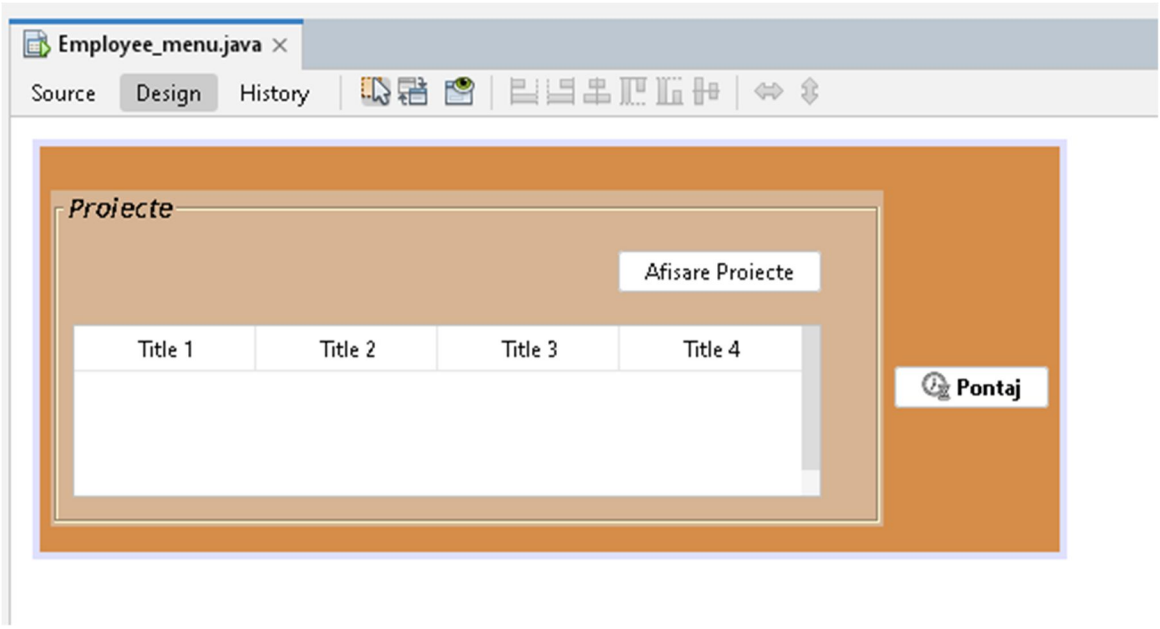

Fig. 4. Jframe-ul pentru meniul principal al angajatului

Acest JFrame reprezinta fereastra pentru afisarea proiectelor angajatului; in el se regasesc urmatoarele:

- Un JButton ce deschide fereastra de ponta;
- Un JTabel ce va afisa proiectele angajatului logat;
- Un JButton ce va realiza afisarea proiectelor;

3.2.4 FINDEMPLOYEE

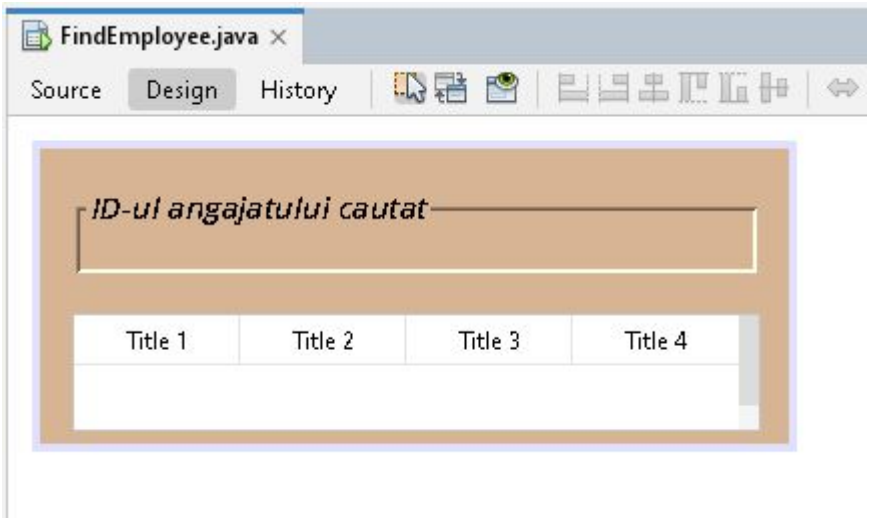

Fig. 5. Jframe-ul pentru fereastra
de cautare a angajatului

Acest JFrame reprezinta fereastra de cautare a angajatului unde am adaugat un JPanel cu urmatoarele:

- Un JTable ce va afisa datele angajatului cautat pe baza ID-ului introduse;
- Un JTextField unde se introduce ID-ul angajatului cautat;

3.2.5 AFISARE_CLIENTI

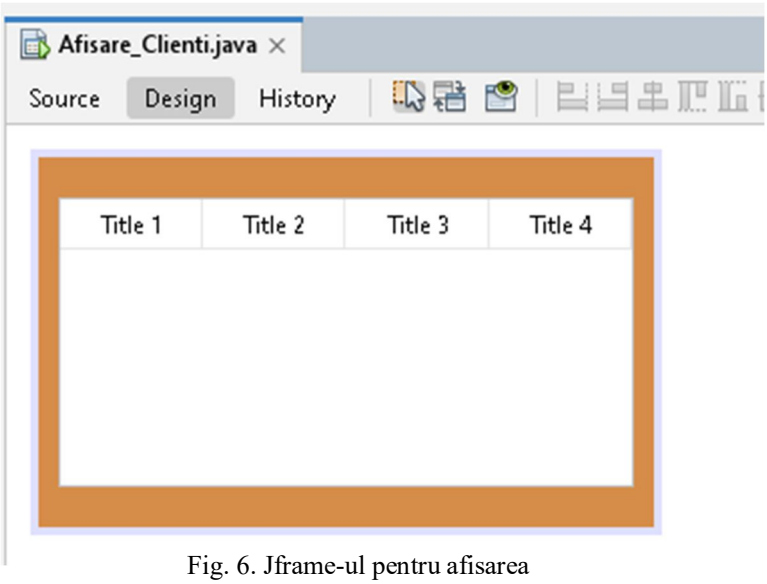

Fig. 6. Jframe-ul pentru afisarea
clientilor

Acesta este un JFrame foarte simplu, continand doar un JPanel cu un JTable unde sunt afisati clientii firmei.

3.2.6 PONTAJ

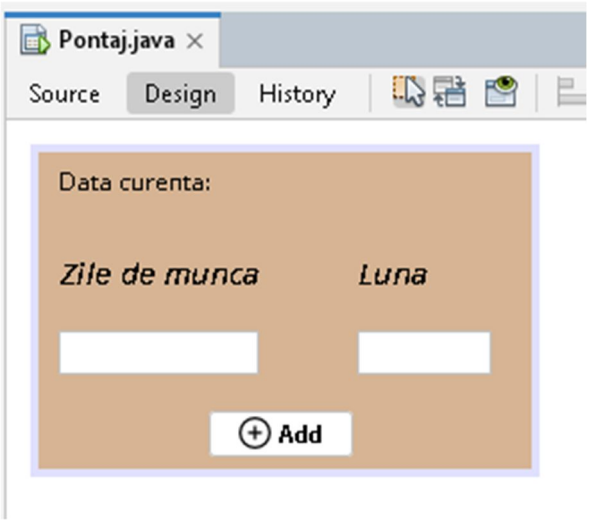

Fig. 7. Jframe-ul pentru introducerea pontajului

In acest JFrame am creat un JPanel ce contine urmatoarele:

- Doua JTextField-uri pentru introducerea zilelor de muna act si a lunii;
- Un JLabel-uri pentru afisarea datei curente;
- Un JButton pentru adaugarea datelor in baza de date;

3.3 REALIZAREA BAZEI DE DATE

Am alcatuit baza de date folosind structura stabilita anterior prin intermediul aplicatiei XAMPP; aceasta creaza un server local unde putem crea tabele folosind cod SQl.

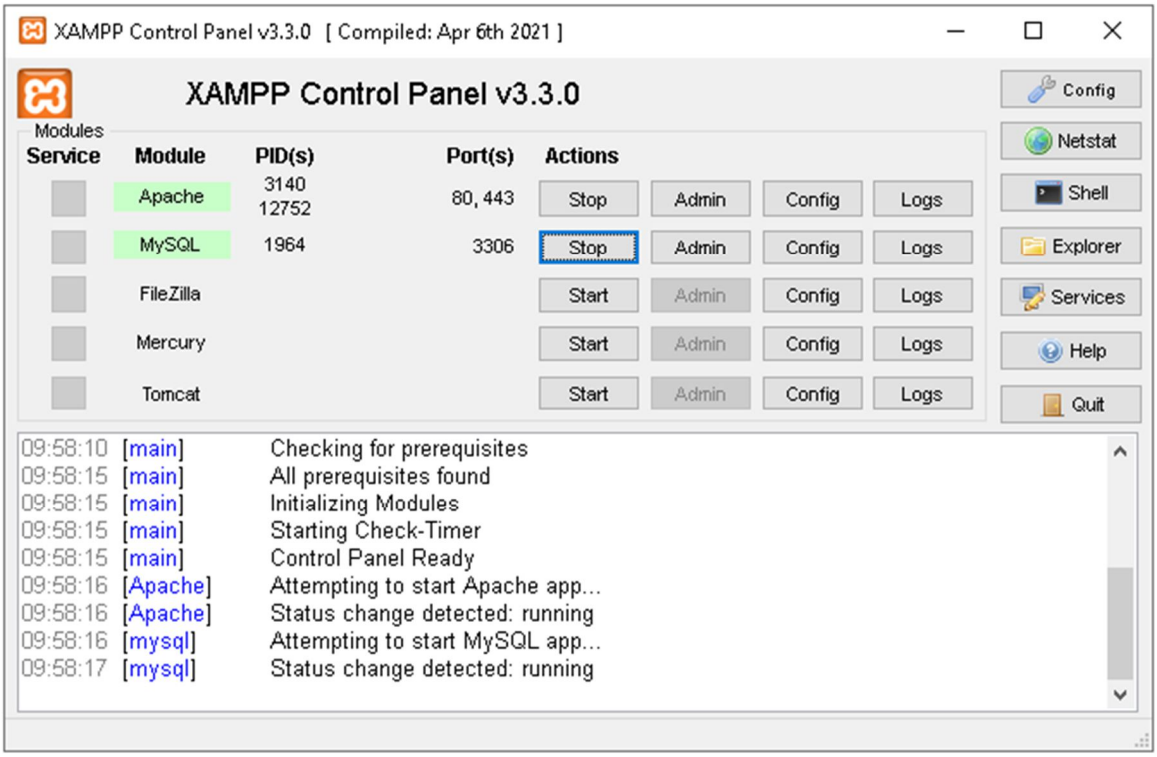

Fig. 8. Lansarea serverului local

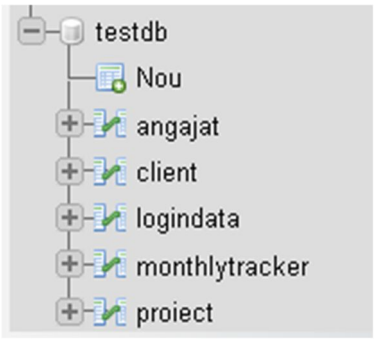

Fig. 9. Tabelele create

	#	Nume	Tip	Colaționare	Atribute	Nul	Implicit	Comentarii	Suplimentar	Acțiune
☐	1	**ID_Angajat**	int(11)			Nu	*Niciuna*			Modifică Elimină Mai mult
☐	2	**Nume**	varchar(25)	utf8mb4_general_ci		Nu	*Niciuna*			Modifică Elimină Mai mult
☐	3	**Prenume**	varchar(40)	utf8mb4_general_ci		Nu	*Niciuna*			Modifică Elimină Mai mult
☐	4	**Salariu**	int(11)			Da	*NULL*			Modifică Elimină Mai mult
☐	5	**Functie**	varchar(30)	utf8mb4_general_ci		Da	*NULL*			Modifică Elimină Mai mult

Fig. 10. Structura tabelei “angajat”

	#	Nume	Tip	Colaționare	Atribute	Nul	Implicit	Comentarii	Suplimentar	Acțiune
☐	1	**ID_Client**	int(11)			Nu	*Niciuna*			Modifică Elimină Mai mult
☐	2	**nume**	varchar(20)	utf8mb4_general_ci		Nu	*Niciuna*			Modifică Elimină Mai mult
☐	3	**prenume**	varchar(30)	utf8mb4_general_ci		Nu	*Niciuna*			Modifică Elimină Mai mult
☐	4	**contact**	varchar(40)	utf8mb4_general_ci		Nu	*Niciuna*			Modifică Elimină Mai mult
☐	5	**adresa**	varchar(50)	utf8mb4_general_ci		Nu	*Niciuna*			Modifică Elimină Mai mult

Fig. 11. Structura tabelei “client”

	#	Nume	Tip	Colaționare	Atribute	Nul	Implicit	Comentarii	Suplimentar	Acțiune
☐	1	**ID_Angajat**	int(11)			Nu	*Niciuna*			Modifică Elimină Mai mult
☐	2	**username**	varchar(20)	utf8mb4_general_ci		Nu	*Niciuna*			Modifică Elimină Mai mult
☐	3	**password**	varchar(15)	utf8mb4_general_ci		Nu	*Niciuna*			Modifică Elimină Mai mult

Fig. 12. Structura tabelei “logindata”

	#	Nume	Tip	Colaționare	Atribute	Nul	Implicit	Comentarii	Suplimentar	Acțiune
☐	1	**ID_Angajat**	int(11)			Nu	*Niciuna*			Modifică Elimină Mai mult
☐	2	**an**	int(11)			Nu	*Niciuna*			Modifică Elimină Mai mult
☐	3	**luna**	int(11)			Nu	*Niciuna*			Modifică Elimină Mai mult
☐	4	**interval_lucrat**	varchar(60)	utf8mb4_general_ci		Nu	*Niciuna*			Modifică Elimină Mai mult

Fig. 13. Structura tabelei “monthlytracker”

	#	Nume	Tip	Colaționare	Atribute	Nul	Implicit	Comentarii	Suplimentar	Acțiune
☐	1	ID_Proiect	int(11)			Nu	*Niciuna*			Modifică Elimină Mai mult
☐	2	ID_Angajat	int(11)			Nu	*Niciuna*			Modifică Elimină Mai mult
☐	3	ID_Client	int(11)			Nu	*Niciuna*			Modifică Elimină Mai mult
☐	4	detalii	varchar(100)	utf8mb4_general_ci		Da	*NULL*			Modifică Elimină Mai mult
☐	5	data_start	varchar(20)	utf8mb4_general_ci		Da	*NULL*			Modifică Elimină Mai mult

Fig. 14. Structura tabelei “proiect”

3.4 REALIZAREA LEGATURII SI A CODULUI

Pentru accesarea bazei de date prin intermediul aplicatiei NetBeans, am folosit secvente de cod similare cu urmatoarele:

- Folosind libraria importata java.sql.Connection am creat un obiect “con” reprezentand conexiunea la baza de date;
- Am folosit libraria java.sql.PreparedStatement pentru a crea un obiect “pst” ce reprezinta o interogare precompilata in SQL;
- Cu ajutorul librariei java.sql.ResultSet am un creat un tabel de date reprezentand un set de date, generat pe baza interogarii precompilate “pst”;
- Executarea interogarii precompilate “pst” prin comanda “pst.executeQuery()”;

Toate acestea se afla intr-un bloc try-catch pentru evitarea eventualelor erori la conectarea cu baza de date.

MainMenu.java

```
public MainMenu() {
        initComponents();
        this.setLocationRelativeTo(null);

        try{
            Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb", "root",
"");

            PreparedStatement pst;

            pst = con.prepareStatement("Select * from angajat");

            ResultSet rs = pst.executeQuery();

            Angajati.setModel(DbUtils.resultSetToTableModel(rs));

        }catch(Exception e){
            JOptionPane.showMessageDialog(null, e);
```

```
        }
    }
```

Mai jos vom prezenta codul pentru realizarea JFrame-ului LogIn, celelalte fiind similare, avand actiuni si interogari sql diferite.

LogIn.java

```
import java.awt.Color;
import javax.swing.JOptionPane;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.Connection;
import java.sql.ResultSet;

public class LogIn extends javax.swing.JFrame {

    public LogIn() {
        initComponents();
        this.setLocationRelativeTo(null);
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        username = new javax.swing.JTextField();
        password = new javax.swing.JPasswordField();
        login = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setAutoRequestFocus(false);
        setResizable(false);

        jPanel1.setBackground(new java.awt.Color(214, 141, 73));

        jLabel1.setFont(new java.awt.Font("Open Sans", 1, 12)); // NOI18N
        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/user.png"))); // NOI18N
        jLabel1.setText("Username");

        jLabel2.setFont(new java.awt.Font("Open Sans", 1, 12)); // NOI18N
        jLabel2.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/padlock.png"))); // NOI18N
        jLabel2.setText("Password");

        username.setFont(new java.awt.Font("Open Sans", 0, 12)); // NOI18N
        username.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                usernameActionPerformed(evt);
            }
        });

        password.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                passwordActionPerformed(evt);
            }
```

```
        });

        login.setBackground(new java.awt.Color(242, 183, 128));
        login.setFont(new java.awt.Font("Segoe UI", 1, 12)); // NOI18N
        login.setText("Log In");
        login.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                loginActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout          jPanel1Layout          =          new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(20, 20, 20)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
                    .addComponent(jLabel1)
                    .addComponent(jLabel2))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,      20,
Short.MAX_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
                    .addComponent(password,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    93,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(username,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    93,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(82, 82, 82)
                .addComponent(login)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(30, 30, 30)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.BASELINE)
                    .addComponent(jLabel1)
                    .addComponent(username,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(25, 25, 25)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.BASELINE)
```

```
                    .addComponent(jLabel2)
                    .addComponent(password,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)
                .addComponent(login)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        javax.swing.GroupLayout            layout            =            new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(0, 0, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(0, 0, 0)
                .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(0, 0, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void loginActionPerformed(java.awt.event.ActionEvent evt) {
        try{
            Connection            con            =            (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb",     "root",
"");
            String  sql  =  "Select  *  from  logindata  where  username=?  and
password=?";

            PreparedStatement pst;
            pst = con.prepareStatement(sql);
            pst.setString(1, username.getText());
            pst.setString(2, password.getText());
            ResultSet rs = pst.executeQuery();

            if(rs.next()){
                JOptionPane.showMessageDialog(null, "LOGGED IN");

                if("admin".equals(username.getText())){

                    MainMenu menu = new MainMenu();

                    setVisible(false);
                    menu.setVisible(true);
                }else
                {
```

```
                    String id_ratat = rs.getString(1);

                    Employee_menu emp = new Employee_menu();

                    emp.id_emp.setText(id_ratat);

                    setVisible(false);
                    emp.setVisible(true);
                }
            }
            else{
                JOptionPane.showMessageDialog(null,                    "Wrong
username/password");

                username.setText("");
                password.setText("");
            }

            con.close();
        }
        catch(Exception e){
            JOptionPane.showMessageDialog(null, e);
        }
    }

    public static void main(String args[]) {
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting
code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the
default look and feel.
         *                  For                  details                  see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for      (javax.swing.UIManager.LookAndFeelInfo      info      :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(LogIn.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(LogIn.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(LogIn.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(LogIn.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
        }
        //</editor-fold>
```

```
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {

                LogIn log_screen = new LogIn();
                log_screen.setVisible(true);

                log_screen.getContentPane().setBackground( Color.PINK );
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JButton login;
    private javax.swing.JPasswordField password;
    private javax.swing.JTextField username;
    // End of variables declaration
}
```

Acesta este un fragment de cod Java care creeaza o interfata grafica de utilizator (GUI) pentru un formular de autentificare. GUI-ul include un camp de text pentru utilizator pentru a introduce numele de utilizator, un camp pentru parola pentru utilizator pentru a introduce parola si un buton de conectare. Atunci cand se apasa butonul de conectare, se executa codul din metoda loginActionPerformed Codul utilizeaza mai multe biblioteci Java, inclusiv java.awt pentru crearea GUI-ului, javax.swing pentru crearea componentelor GUI si java.sql pentru conectarea la o baza de date. Codul utilizeaza, de asemenea, mai multi manageri de layout Swing pentru a pozitiona componentele pe GUI.

Un exemplu mai complicat pentru conectarea la baza de date il constituie codul folosit la realizarea butonului “DELETE” din cadrul ferestrei “MainMenu”; acesta s-a prezentat a fi mai dificil, deoarece am avut nevoie de doua interogari precompilate, una pentru a verifica daca angajatul pe care dorim sa il stergem exista, si alta pentru a realiza stergerea.

MainMenu.java

```
private void Delete_buttonActionPerformed(java.awt.event.ActionEvent evt) {
        String deleted_id = JOptionPane.showInputDialog("Introduceti ID - ul
anagajatului.");

        int d_id = Integer.parseInt(deleted_id);

        String sql = "DELETE FROM angajat WHERE Id_Angajat = '" + d_id + "'";
        String test = "SELECT Id_Angajat FROM Angajat WHERE Id_Angajat like
'%" + d_id + "%'";


           try{
```

```
                Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb", "root", "");

                PreparedStatement pst = con.prepareStatement(sql);
                PreparedStatement psTest = con.prepareStatement(test);

                ResultSet rsTest = psTest.executeQuery();

                if(rsTest.next()){


                    pst.execute();

                    JOptionPane.showMessageDialog(null,"Sucessfully Deleted!");

                    pst = con.prepareStatement("Select * from angajat");

                    ResultSet rs = pst.executeQuery();

                    Angajati.setModel(DbUtils.resultSetToTableModel(rs));
                }
                else{
                    JOptionPane.showMessageDialog(null, "Angajatul nu exista!");
                }

            }
            catch(Exception e){
                JOptionPane.showMessageDialog(null, e);
            }
    }
```

4.CONCLUZII

4.1 CONCLUZII GENERALE

Programul prezentat are ca scop monitorizarea activității firmei de contabilitate, fiind proiectată să corespundă nevoilor angajaților.

Dezvoltarea structurii flexibile este ușor de utilizat, beneficiile unei asemenea abordări sunt:

- Supervizarea activității angajaților și posibilitatea de a schimba anumite sarcini între aceștia
- Desfășurarea fără complicații a lucrărilor
- Îmbunătățirea activităților administrative și decizionale
- Posibilitatea extinderii afacerii fără a se depune muncă suplimentară

4.2 ELEMENTE DE ORIGINALITATE

Programul de contabilitate propus aduce elemente de originalitate prin abordarea sa unică a proceselor de lucru. Acesta se bazează pe o structură flexibilă, care permite supervizarea activității angajaților și schimbarea sarcinilor între aceștia în mod eficient. De asemenea, programul oferă posibilitatea desfășurării lucrărilor fără complicații și îmbunătățește activitățile administrative și decizionale. Aceasta permite extinderea afacerii fără a se depune munca suplimentară. În plus, programul promovează o comunicare eficientă între membrii echipei și celelalte echipe, contribuind la dezvoltarea personală și profesională a angajaților. Aceste elemente de originalitate fac din acest program un instrument valoros pentru orice firmă de contabilitate.

4.3 DIRECTII VIITOARE DE DEZVOLTARE

Directiile viitoare de dezvoltare pentru programul nostru de contabilitate vizeaza imbunatatirea automatizarii proceselor, implementarea tehnologiilor inteligente de analiza a datelor si crearea de integrari cu alte sisteme de gestiune financiara pentru o mai buna eficienta si precizie in activitatile contabile.De asemenea, ne concentram pe dezvoltarea de noi caracteristici care sa raspunda nevoilor specifice ale diferitelor industrii, precum si pe crearea unei interfete ale utilizatorului mai intuitive si usor de utilizat. In plus, vom continua sa investim in siguranta si protectia datelor pentru a asigura confidentialitatea si securitatea informatiilor clientilor nostri. Aceste directii de dezvoltare vor fi realizate prin colaborarea

cu echipele de cercetare si dezvoltare, precum si prin feedback-ul si sugestiile clientilor nostri pentru a ne asigura ca raspundem nevoilor si asteptarilor lor.

4.4 IMPRESII

Acest proiect, pentru membri echipa noastră, ne-a permis să ne dezvoltăm atât din punct de vedere profesional cât și personal. Din punct de vedere profesional, am reușit să colaborăm ca o echipă și să punem accentul pe informatică și mai multe limbaje de programare, fiind nevoiți să învățăm și să ne descurcăm în situații de împas și blocaj. Din punct de vedere personal, am observat că acest proiect ne-a permis să dezvoltăm o comunicare atât cu membrii echipei cât și cu celelalte echipe.