package src.db;

import java.sql.\*;

import java.text.MessageFormat;

import java.time.LocalDate;

/\*\*

 \* JSqlCon

 \*/

public class JSqlCon {

    static final String DB\_URL = "jdbc:mysql://localhost/clinicaeit";

    static final String USER = "root";

    static final String PASS = "";

    static final String QUERY = "SELECT \* FROM funcionario, Paciente";

    static Connection conn;

    public static boolean connectDb() {

        try {// Open a connection

            Class.forName("com.mysql.cj.jdbc.Driver");

            conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/clinicaeit", "root", "");

            return true;

        } catch (Exception e) {

            System.out.println("Occoreu um erro ao se conectar a base de dados!");

        }

        return false;

    }

    public static int getId(String table) {

        int id = 0;

        connectDb();

        String query = MessageFormat.format("SELECT ID FROM {0} ORDER BY ID DESC LIMIT 1", table);

        try {

            Statement stmt = conn.createStatement();

            ResultSet rs = stmt.executeQuery(query);

            // Extract data from result set

            while (rs.next()) {

                // Retrieve by column name

                id = rs.getInt("ID");

            }

        } catch (SQLException e) {

            e.printStackTrace();

        }

        return id;

    }

    // Testing DB connection

    public static void getData() {

        connectDb();

        try {

            Statement stmt = conn.createStatement();

            ResultSet rs = stmt.executeQuery(QUERY);

            // Extract data from result set

            while (rs.next()) {

                // Retrieve by column name

                System.out.print("ID: " + rs.getInt("ID"));

                System.out.print(", Login Name: " + rs.getString("LoginName"));

                System.out.print(", Name: " + rs.getString("Nome") + "\n");

                System.out.print(", Tipo de perfil: " + rs.getString("TipoUsuarioID") + "\n");

            }

        } catch (SQLException e) {

            e.printStackTrace();

        }

    }

    public static void insertIntoPaciente(int id, String loginName, String nome, int numeroSeguroSaude,

            Date dataNascimento, String genero, String nacionalidade, String endereco, String email, String celular,

            boolean status, String observacao, int tipoUsuarioID) {

        connectDb();

        try {

            String update = MessageFormat.format(

                    "INSERT INTO `Paciente` (`ID`, `LoginName`, `Nome`, `NumeroSeguroSaude`, `DataNascimento`, `Genero`, `Nacionalidade`, `Endereco`, `Email`, `Celular`, `Status`, `Observacao`, `TipoUsuarioID`) VALUES ({0}, {1}, {2}, {3}, {4}, {5}, {6}, {7}, {8}, {9}, {10}, {11}, {12})",

                    id, loginName, nome, numeroSeguroSaude, dataNascimento, genero, nacionalidade, endereco, email,

                    celular, status, observacao, tipoUsuarioID);

            Statement stmt = conn.createStatement();

            stmt.executeUpdate(update);

            System.out.println("Sucesso");

        } catch (Exception e) {

            System.out.println(e);

        }

    }

    public static void insertIntoFuncionario(String table, int id, String especialidade, int CRM, String cargo,

            LocalDate dataContratacao, float salario, String nome, Date dataNascimento, String genero,

            String nacionalidade, String endereco, String email, String celular, boolean status, String observacao,

            int tipoUsuarioID, String horarioTrabalho, String loginName) {

        connectDb();

        try {

            String update = MessageFormat.format(

                    "INSERT INTO {0} (`ID`, `Especialidade`, `CRM`, `Cargo`, `DataContratacao`, `Salario`, `Nome`, `DataNascimento`,"+

                    " `Genero`, `Nacionalidade`, `Endereco`, `Email`, `Celular`, `Status`, `Observacao`, `TipoUsuarioID`, `HorarioTrabalho`,"+

                    " `LoginName`) VALUES ('"+id+"', '"+especialidade+"', '"+CRM+"', '"+cargo+"', '"+dataContratacao+"', '"+salario+"', '"+"'"+nome+"'"+"', '"+dataNascimento+"', '"+"'"+genero+"'"+"', '"+nacionalidade+"',"+

                    " '"+endereco+"', '"+email+"', '"+celular+"', '"+status+"', '"+"'"+observacao+"'"+"', '"+tipoUsuarioID+"', '"+horarioTrabalho+"', '"+"'"+loginName+"'"+"')", table

                );

            Statement stmt = conn.createStatement();

            stmt.executeUpdate(update);

            System.out.println("Sucesso");

        } catch (Exception e) {

            System.out.println(e);

        }

    }

}

package src.operations;

import java.util.Date;

public class Consulta {

    private Date dataConsulta;

    private int paciente;//id do paciente

    private int medico;//id do medico

    private String consultaStatus; //agendada, confirmada, realizada, cancelada

    private String obs;//observacoes

    public Consulta(Date dataConsulta, int paciente, int medico, String consultaStatus, String obs) {

        this.dataConsulta = dataConsulta;

        this.paciente = paciente;

        this.medico = medico;

        this.consultaStatus = consultaStatus;

        this.obs = obs;

    }

    public Consulta(){}

    public Date getDataConsulta() {

        return this.dataConsulta;

    }

    public void setDataConsulta(Date dataConsulta) {

        this.dataConsulta = dataConsulta;

    }

    public int getPaciente() {

        return this.paciente;

    }

    public void setPaciente(int paciente) {

        this.paciente = paciente;

    }

    public int getMedico() {

        return this.medico;

    }

    public void setMedico(int medico) {

        this.medico = medico;

    }

    public String getconsultaStatus() {

        return this.consultaStatus;

    }

    public void setconsultaStatus(String consultaStatus) {

        this.consultaStatus = consultaStatus;

    }

    public String getObs() {

        return this.obs;

    }

    public void setObs(String obs) {

        this.obs = obs;

    }

    public void agendarConsulta(){//cria uma nova consulta no sitema

    }

    public void confirmarConsulta(){//confirmar uma consulta agendada

    }

    public void realizaConsulta(){//Registar detalhes da consulta e marcar como realizada

    }

    public String imprimeDados() {

        return

            "Data da consulta:\t "+getDataConsulta()+ "\n" +

            "Paciente:\t\t "+getPaciente()+ "\n" +

            "Medico:\t\t\t "+getMedico()+ "\n" +

            "Estado da consulta:\t "+getconsultaStatus()+ "\n" +

            "Observacoes:\t\t " + getObs() + "\n";

    }

}

package src.operations;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class LoginForm extends JFrame implements ActionListener {

    //initialize button, panel, label, and text field

    JButton b1;

    JPanel newPanel;

    JLabel userLabel, passLabel;

    final JTextField  textField1, textField2;

    //calling constructor

    public LoginForm() {

        setDefaultCloseOperation(javax.swing.

        WindowConstants.DISPOSE\_ON\_CLOSE);

        //create label for username

        userLabel = new JLabel();

        userLabel.setText("Username");//set label value for textField1

        //create text field to get username from the user

        textField1 = new JTextField(15);//set length of the text

        //create label for password

        passLabel = new JLabel();

        passLabel.setText("Password");//set label value for textField2

        //create text field to get password from the user

        textField2 = new JPasswordField(15);//set length for the password

        //create submit button

        b1 = new JButton("SUBMIT");//set label to button

        //create panel to put form elements

        newPanel = new JPanel(new GridLayout(3, 1));

        newPanel.add(userLabel);//set username label to panel

        newPanel.add(textField1);//set text field to panel

        newPanel.add(passLabel);//set password label to panel

        newPanel.add(textField2);//set text field to panel

        newPanel.add(b1);//set button to panel

        //set border to panel

        add(newPanel, BorderLayout.CENTER);

        //perform action on button click

        b1.addActionListener(this);     //add action listener to button

        setTitle("LOGIN FORM");         //set title to the login form

    }

    //define abstract method actionPerformed() which will be called on button click

    public void actionPerformed(ActionEvent ae){//pass action listener as a parameter

        String userValue = textField1.getText();//get user entered username from the textField1

        String passValue = textField2.getText();//get user entered pasword from the textField2

        //check whether the credentials are authentic or not

        if (userValue.equals("test@gmail.com") && passValue.equals("test")) {  //if authentic, navigate user to a new page

            //create instance of the NewPage

            NewPage page = new NewPage();

            //make page visible to the user

            page.setVisible(true);

            //create a welcome label and set it to the new page

            JLabel wel\_label = new JLabel("Welcome: "+userValue);

            page.getContentPane().add(wel\_label);

        }

        else{

            //show error message

            System.out.println("Please enter valid username and password");

        }

    }

}

class NewPage extends JFrame

{

    //constructor

    NewPage()

    {

        setDefaultCloseOperation(javax.swing.

        WindowConstants.DISPOSE\_ON\_CLOSE);

        setTitle("Welcome");

        setSize(400, 200);

    }

}

package src.personel;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectOutputStream;

import java.io.Serializable;

import java.time.LocalDate;

import java.util.\*;

import java.sql.Date;

import src.db.JSqlCon;

/\*\*\*

 \*

 \*/

public class Funcionario extends Pessoa  implements Serializable{

    private int funcId;//numero de funcionario

    private String endereco;

    private String celular;

    private String email;

    private String cargo;

    private LocalDate dataContratacao;

    private float salario;

    private String horarioTrabalho;

    private boolean status; //funcionario no ativo ou nao!

    private String loginName;//nome de utilizador de acesso ao sistema

    private String loginPwd;//Palavra-passe de acesso ao sistema

    private boolean accessPermiss;//permissao de acesso ao sistema!

    private String historico;

    private String obs;//observacaoes

    public static ArrayList<Funcionario> funcionarios = new ArrayList<>();

    Scanner sc= new Scanner(System.in);

    public Funcionario(){}

    public Funcionario(int funcId, String nome, Date dataNasc,String genero, String bi, String nacionalidade, String endereco, String celular, String email,String cargo, LocalDate dataContratacao, float salario, String horarioTrabalho, boolean status, String loginName, String loginPwd, boolean accessPermiss, String historico, String obs) {

        super(nome, dataNasc, genero, bi, nacionalidade);

        this.funcId = funcId;

        this.endereco = endereco;

        this.celular = celular;

        this.email = email;

        this.cargo=cargo;

        this.dataContratacao = dataContratacao;

        this.salario = salario;

        this.horarioTrabalho = horarioTrabalho;

        this.status = status;

        this.loginName = loginName;

        this.loginPwd = loginPwd;

        this.accessPermiss = accessPermiss;

        this.historico = historico;

        this.obs = obs;

    }

    //Metodos acessores, getters and setters

    public String getEndereco() {

        return this.endereco;

    }

    public void setEndereco(String endereco) {

        this.endereco = endereco;

    }

    public String getCelular() {

        return this.celular;

    }

    public void setCelular(String celular) {

        this.celular = celular;

    }

    public String getEmail() {

        return this.email;

    }

    public void setEmail(String email) {

        this.email = email;

    }

    public String getCargo() {

        return this.cargo;

    }

    public void setCargo(String cargo) {

        this.cargo = cargo;

    }

    public LocalDate getDataContratacao() {

        return this.dataContratacao;

    }

    public void setDataContratacao(LocalDate dataContratacao) {

        this.dataContratacao = dataContratacao;

    }

    public float getSalario() {

        return this.salario;

    }

    public void setSalario(float salario) {

        this.salario = salario;

    }

    public String getHorarioTrabalho() {

        return this.horarioTrabalho;

    }

    public void setHorarioTrabalho(String horarioTrabalho) {

        this.horarioTrabalho = horarioTrabalho;

    }

    public boolean isStatus() {

        return this.status;

    }

    public boolean getStatus() {

        return this.status;

    }

    public void setStatus(boolean status) {

        this.status = status;

    }

    public String getLoginName() {

        return this.loginName;

    }

    public void setLoginName(String loginName) {

        this.loginName = loginName;

    }

    public String getLoginPwd() {

        return this.loginPwd;

    }

    public void setLoginPwd(String loginPwd) {

        this.loginPwd = loginPwd;

    }

    public boolean isAccessPermiss() {

        return this.accessPermiss;

    }

    public boolean getAccessPermiss() {

        return this.accessPermiss;

    }

    public void setAccessPermiss(boolean accessPermiss) {

        this.accessPermiss = accessPermiss;

    }

    public String getHistorico() {

        return this.historico;

    }

    public void setHistorico(String historico) {

        this.historico = historico;

    }

    public String getObs() {

        return this.obs;

    }

    public void setObs(String obs) {

        this.obs = obs;

    }

    public int getFuncId() {

        return this.funcId;

    }

    public void setFuncId(int funcId) {

        this.funcId = funcId;

    }

    public String adicionarPaciente(ArrayList<Paciente> pacienteList, int tipoUsuarioID){//adicionar paciente ao sistema

        try {

            int id = JSqlCon.getId("paciente");

            Paciente paciente = new Paciente();

            paciente.setPacId(id);

            System.out.println("Digite o nome: ");paciente.setNome(sc.next());

            paciente.setDataNasc(null);

            //System.out.println("Digite o genero: ");paciente.setGenero(sc.next());

            /\*System.out.println("Digite o numero de BI: ");paciente.setBi(sc.next());

            System.out.println("Digite a nacionalidade: ");paciente.setNacionalidade(sc.next());

            System.out.println("Digite o estado civil: ");paciente.setEstadoCivil(sc.next());

            System.out.println("Digite o numero de celular: ");paciente.setCelular(sc.nextInt());

            System.out.println("Digite o email ");paciente.setEmail(sc.next());

            System.out.println("Digite o nome de utilizador: ");paciente.setLoginName(sc.next());

            System.out.println("Crie sua password: ");paciente.setLoginPwd(sc.next());

            System.out.println("Digite sua informacao de emergencia: ");paciente.setEmergencyInfo(sc.next());

            System.out.println("Digite o seu endereco: ");paciente.setEndereco(sc.next());

            System.out.println("Digite o seu numero de seguro: ");paciente.setNumeroSegSaude(sc.nextInt()); \*/

            paciente.setAccessPermiss(true);

            paciente.setStatus(true);

            //System.out.println("Insira seu historico Medico: "); paciente.setHistoricoMedico(sc.next());

            System.out.println("Insira uma observacao: "); paciente.setObs(sc.next());

            Paciente.pacientes.add(paciente);

            new Paciente(++id, paciente.getNome(), paciente.getDataNasc(), paciente.getGenero(), paciente.getBi(), paciente.getNacionalidade(), paciente.getEmergencyInfo(), paciente.getEndereco(), paciente.getEmail(), paciente.getCelular(), paciente.getEstadoCivil(),0 , paciente.getHistoricoMedico(), paciente.getLoginName(), paciente.getLoginPwd(), paciente.getAccessPermiss(), paciente.getStatus(), paciente.getObs());

            salvarDados(null, Paciente.pacientes, "src/File/ficheiros/paciente.dat");

            JSqlCon.insertIntoPaciente(++id, getLoginName(), paciente.getNome(), paciente.getNumeroSegSaude(), paciente.getDataNasc(), paciente.getGenero(), paciente.getNacionalidade(), paciente.getEndereco(), paciente.getEmail(), paciente.getCelular(), paciente.getStatus(), paciente.getObs(), tipoUsuarioID);

        } catch (Exception e) {

            // TODO: handle exception

        }

        return "Erro ao adicionar Paciente";

    }

    public void confimarConsulta(){//confirmar umna consulta agendada (confirmada, cancelada, adiada..)

    }

    public void monitorarSistema(){//mostrar uma visao geral so sistema

    }

    //metodo para imprimir os dados do Funcionario

    @Override

    public void imprimeDados(){

        for(Funcionario fc: funcionarios){

            System.out.println("\n"+

                "Codigo de funcionario:\t\t "+fc.getFuncId()+ "\n" +

                "Nome:\t\t\t\t "+fc.getNome()+ "\n" +

                "Data de nascimento:\t\t "+fc.getDataNasc()+ "\n" +

                "Genero:\t\t\t\t "+fc.getGenero()+ "\n" +

                "Numero de bilhete de identidade: "+fc.getBi()+ "\n" +

                "Nacionalidade:\t\t\t "+fc.getNacionalidade()+ "\n" +

                "Endereco:\t\t\t " + fc.getEndereco() + "\n" +

                "Celular:\t\t\t " + fc.getCelular() + "\n" +

                "Email:\t\t\t\t " + fc.getEmail() + "\n" +

                "Data de contratacao:\t\t " + fc.getDataContratacao() + "\n" +

                "Salario:\t\t\t " + fc.getSalario() + "\n" +

                "Horario de trabalho:\t\t " + fc.getHorarioTrabalho() + "\n" +

                "Status:\t\t\t\t " + fc.isStatus() + "\n" +

                "Nome de utilizador:\t\t\t " + fc.getLoginName() + "\n" +

                "Permissao de acesso:\t\t " + fc.isAccessPermiss() + "\n" +

                "Observacoes:\t\t\t " + fc.getObs() + "\n"

            );

        }

    }

    public static void salvarDados(ArrayList<Funcionario> obj1, ArrayList<Paciente> obj2, String fileName){

        try {

            // Criar um FileOutputStream para escrever no arquivo .dat

            FileOutputStream fileOutputStream = new FileOutputStream(fileName);

            // Criar um ObjectOutputStream para escrever objetos no FileOutputStream

            ObjectOutputStream objectOutputStream = new ObjectOutputStream(fileOutputStream);

            if(obj1.equals(null)){

             // Escrever objeto no arquivo

                objectOutputStream.writeObject(obj2);

                System.out.println("Objeto Paciente salvo em "+fileName);

            }

            else if(obj2.equals(null)){

                // Escrever objeto no arquivo

                objectOutputStream.writeObject(obj1);

                System.out.println("Objeto Funcionario salvo em "+fileName);

            }

            else

                System.out.println("Nao pode gravar dois ojectos em simultaneo. Atencao!");

            // Fechar fluxos

            objectOutputStream.close();

            fileOutputStream.close();

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

}

package src.personel;

import java.time.LocalDate;

import java.sql.Date;

public class Medico extends Funcionario{

    private String especialidade;

    private int CRM;//numero de registo medico

    public Medico(int funcId, String nome, Date dataNasc, String genero, String bi, String nacionalidade, String endereco, String celular, String email,String cargo, LocalDate dataContratacao, int salario, String horarioTrabalho, boolean status, String loginName, String loginPwd, boolean accessPermiss, String historico, String obs, String especialidade, int CRM){

        super(funcId, nome, dataNasc, genero, bi, nacionalidade, endereco, celular, email, cargo, dataContratacao, salario, horarioTrabalho, status, loginName, loginPwd, accessPermiss, historico, obs);

        this.especialidade = especialidade;

        this.CRM = CRM;

    }

    public Medico(){}

    public String getEspecialidade() {

        return this.especialidade;

    }

    public void setEspecialidade(String especialidade) {

        this.especialidade = especialidade;

    }

    public int getCRM() {

        return this.CRM;

    }

    public void setCRM(int CRM) {

        this.CRM = CRM;

    }

    public void gerirAgenda(){//Permitir ao medico, ver e gerir a sua agenda de consultas

    }

    public void aceitarConsulta(){//aceitar uma consulta agenda (apos a confirmacao)

    }

    public void rejeitarConsulta(){//rejeitar uma consulta agendada

    }

}

package src.personel;

import java.io.Serializable;

import java.util.ArrayList;

import java.sql.Date;

public class Paciente extends Pessoa implements Serializable{

    private String EmergencyInfo; //informacao de emergencia

    private int PacId;//numero de paciente

    private String endereco;

    private String estadoCivil;

    private String email;

    private String celular;

    private int numeroSegSaude;

    private String historicoMedico;

    private String loginName;

    private String loginPwd;

    private boolean accessPermiss;

    private boolean status;

    private String obs; //observacaoes

    public static ArrayList<Paciente> pacientes = new ArrayList<>();

    public Paciente() {

    }

    public Paciente(int PacId, String nome, Date dataNasc,String genero, String bi, String nacionalidade, String EmergencyInfo, String endereco,String email, String celular, String estadoCivil, int numeroSegSaude, String historicoMedico, String loginName, String loginPwd, boolean accessPermiss, boolean status, String obs) {

        super(nome, dataNasc, genero, bi, nacionalidade);

        this.EmergencyInfo = EmergencyInfo;

        this.PacId = PacId;

        this.endereco = endereco;

        this.estadoCivil = estadoCivil;

        this.email = email;

        this.celular = celular;

        this.numeroSegSaude = numeroSegSaude;

        this.historicoMedico = historicoMedico;

        this.loginName = loginName;

        this.loginPwd = loginPwd;

        this.accessPermiss = accessPermiss;

        this.status = status;

        this.obs = obs;

    }

    public boolean isStatus() {

        return this.status;

    }

    public String getEmergencyInfo() {

        return this.EmergencyInfo;

    }

    public void setEmergencyInfo(String EmergencyInfo) {

        this.EmergencyInfo = EmergencyInfo;

    }

    public int getPacId() {

        return this.PacId;

    }

    public void setPacId(int PacId) {

        this.PacId = PacId;

    }

    public String getEndereco() {

        return this.endereco;

    }

    public void setEndereco(String endereco) {

        this.endereco = endereco;

    }

    public String getEstadoCivil() {

        return this.estadoCivil;

    }

    public void setEstadoCivil(String estadoCivil) {

        this.estadoCivil = estadoCivil;

    }

    public int getNumeroSegSaude() {

        return this.numeroSegSaude;

    }

    public void setNumeroSegSaude(int numeroSegSaude) {

        this.numeroSegSaude = numeroSegSaude;

    }

    public String getHistoricoMedico() {

        return this.historicoMedico;

    }

    public void setHistoricoMedico(String historicoMedico) {

        this.historicoMedico = historicoMedico;

    }

    public boolean isAccessPermiss() {

        return this.accessPermiss;

    }

    public boolean getAccessPermiss() {

        return this.accessPermiss;

    }

    public void setAccessPermiss(boolean accessPermiss) {

        this.accessPermiss = accessPermiss;

    }

    public boolean isActive() {

        return this.status;

    }

    public boolean getStatus() {

        return this.status;

    }

    public void setStatus(boolean status) {

        this.status = status;

    }

    public String getLoginName() {

        return this.loginName;

    }

    public void setLoginName(String loginName) {

        this.loginName = loginName;

    }

    public String getLoginPwd() {

        return this.loginPwd;

    }

    public void setLoginPwd(String loginPwd) {

        this.loginPwd = loginPwd;

    }

    public String getEmail() {

        return this.email;

    }

    public void setEmail(String email) {

        this.email = email;

    }

    public String getCelular() {

        return this.celular;

    }

    public void setCelular(String celular) {

        this.celular = celular;

    }

    public String getObs() {

        return this.obs;

    }

    public void setObs(String obs) {

        this.obs = obs;

    }

    @Override

    public void imprimeDados(){

        for(Paciente pc: pacientes){

            System.out.println("\n"+

                "Numero de Paciente: \t\t" + pc.getPacId() + "\n" +

                "Nome: \t\t\t\t" + pc.getNome() + "\n" +

                "Data de nascimento: \t\t" + pc.getDataNasc() + "\n" +

                "Genero: \t\t\t" + pc.getGenero() + "\n" +

                "Nacionalidade: \t\t\t" + pc.getNacionalidade() + "\n" +

                "Estado Civil: \t\t\t" + pc.getEstadoCivil() + "\n" +

                "Endereco: \t\t\t" + pc.getEndereco() + "\n" +

                "Celular: \t\t\t" + pc.getCelular() + "\n" +

                "Email: \t\t\t\t" + pc.getEmail() + "\n" +

                "Informacao de Emergencia: \t" + pc.getEmergencyInfo() + "\n" +

                "Numero de SegSaude: \t\t" + pc.getNumeroSegSaude() + "\n" +

                "Nome de utilizador: \t\t" + pc.getLoginName() + "\n" +

                "Permissao de acesso: \t\t" + pc.isAccessPermiss() + "\n" +

                "Status: \t\t\t" + pc.isStatus() + "\n" +

                "Historico Medico: \t\t" + pc.getHistoricoMedico() + "\n" +

                "Observacao: \t\t\t" + pc.getObs() + "\n"

            );

        }

    }

    /\*@Override

    public void salvarDados(Pacie) {

        try (ObjectOutputStream outputStream = new ObjectOutputStream(new FileOutputStream("dados.dat"))) {

            outputStream.writeObject(pessoas);

            JOptionPane.showMessageDialog(null, "Dados salvos com sucesso!");

        } catch (IOException e) {

            e.printStackTrace();

        }

    }\*/

    @Override

    public String toString() {

        return "{" +

            " EmergencyInfo='" + getEmergencyInfo() + "'" +

            ", PacId='" + getPacId() + "'" +

            ", endereco='" + getEndereco() + "'" +

            ", estadoCivil='" + getEstadoCivil() + "'" +

            ", email='" + getEmail() + "'" +

            ", celular='" + getCelular() + "'" +

            ", numeroSegSaude='" + getNumeroSegSaude() + "'" +

            ", historicoMedico='" + getHistoricoMedico() + "'" +

            ", loginName='" + getLoginName() + "'" +

            ", loginPwd='" + getLoginPwd() + "'" +

            ", accessPermiss='" + isAccessPermiss() + "'" +

            ", status='" + isStatus() + "'" +

            ", obs='" + getObs() + "'" +

            "}";

    }

}

package src.personel;

import java.sql.Date;

/\*\*\*\*

 \* @author

 \*\*\*/

public abstract class Pessoa{

    protected String nome;

    protected Date dataNasc;

    protected String genero;

    protected String bi;//numero de bilhete de de identidade

    protected String nacionalidade;

    public Pessoa(){}//metodo contrutor vazio

    public Pessoa(String nome, Date dataNasc, String genero, String bi, String nacionalidade){//metodo construtor com parametros

        this.nome = nome;

        this.dataNasc = dataNasc;

        this.genero = genero;

        this.bi = bi;

        this.nacionalidade = nacionalidade;

    }

    public String getNome(){

        return this.nome;

    }

    public Date getDataNasc(){

        return this.dataNasc;

    }

    public String getBi(){

        return this.bi;

    }

    public String getNacionalidade(){

        return this.nacionalidade;

    }

    public void setNome(String nome){

        this.nome = nome;

    }

    public String getGenero() {

        return this.genero;

    }

    public void setGenero(String genero) {

        this.genero = genero;

    }

    public void setDataNasc(Date dataNasc){

        this.dataNasc = dataNasc;

    }

    public void setBi(String bi){

        this.bi = bi;

    }

    public void setNacionalidade(String nacionalidade){

        this.nacionalidade = nacionalidade;

    }

    //metodo abstracto

    public abstract void imprimeDados();

    //public abstract int pinGen();

    //protected abstract void salvarDados(Pessoa obj, String fileName);

}

package src;

import src.db.JSqlCon;

import src.personel.\*;

import java.sql.\*;

import java.text.MessageFormat;

import java.util.Scanner;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

/\*\*

 \* App

 \*/

public class App extends JPanel {

    private JLabel usernameLabel;

    private JTextField usernameField;

    private JLabel passwordLabel;

    private JPasswordField passwordField;

    private JButton loginButton;

    private JButton cancelButton;

    public App() {

        // Set the layout manager n

        setLayout(new GridBagLayout());

        // Initialize components

        usernameLabel = new JLabel("Username:");

        usernameField = new JTextField(20);

        passwordLabel = new JLabel("Password:");

        passwordField = new JPasswordField(20);

        loginButton = new JButton("Login");

        cancelButton = new JButton("Cancel");

        // Configure components

        usernameLabel.setHorizontalAlignment(SwingConstants.RIGHT);

        passwordLabel.setHorizontalAlignment(SwingConstants.RIGHT);

        // Add components to the panel

        add(usernameLabel, new GridBagConstraints(0, 0, 1, 1, 0.0, 0.0,

                GridBagConstraints.EAST, GridBagConstraints.NONE,

                new Insets(10, 10, 5, 5), 0, 0));

        add(usernameField, new GridBagConstraints(1, 0, 1, 1, 1.0, 0.0,

                GridBagConstraints.WEST, GridBagConstraints.HORIZONTAL,

                new Insets(10, 0, 5, 10), 0, 0));

        add(passwordLabel, new GridBagConstraints(0, 1, 1, 1, 0.0, 0.0,

                GridBagConstraints.EAST, GridBagConstraints.NONE,

                new Insets(5, 10, 5, 5), 0, 0));

        add(passwordField, new GridBagConstraints(1, 1, 1, 1, 1.0, 0.0,

                GridBagConstraints.WEST, GridBagConstraints.HORIZONTAL,

                new Insets(5, 0, 5, 10), 0, 0));

        add(loginButton, new GridBagConstraints(0, 2, 1, 1, 0.0, 0.0,

                GridBagConstraints.CENTER, GridBagConstraints.NONE,

                new Insets(5, 10, 10, 5), 0, 0));

        add(cancelButton, new GridBagConstraints(1, 2, 1, 1, 0.0, 0.0,

                GridBagConstraints.CENTER, GridBagConstraints.NONE,

                new Insets(5, 5, 10, 10), 0, 0));

        // Add action listeners

        loginButton.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                String user = usernameField.getText();

                String password = new String(passwordField.getPassword());

                if (user.equals("admin") && password.equals("admin")) {

                    // if login is successful, dispose the login dialog and open the main panel

                    // Dispose of the App frame

                    Window window = SwingUtilities.windowForComponent(App.this);

                    if (window instanceof Frame) {

                        // Dispose of the parent frame

                        ((Frame) window).dispose();

                    } else if (window instanceof Dialog) {

                        // Dispose of the parent dialog

                        ((Dialog) window).dispose();

                    }

                    // Open the main panel

                    MainPanel mainPanel = new MainPanel();

                    mainPanel.setVisible(true);

                    //frame.dispose();

                } else {

                    JOptionPane.showMessageDialog(null, "Invalid login. Please try again.", "Error", JOptionPane.ERROR\_MESSAGE);

                }

            }

        });

        cancelButton.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                // Here, you can handle the cancel button action

                // For example, you can close the application or

                // the login dialog

                System.exit(0);

            }

        });

    }

    public static void main(String[] args) {

        // Create a frame to hold the panel

        JFrame frame = new JFrame("Login Panel");

        frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        // Create a panel and add it to the frame

        App panel = new App();

        frame.getContentPane().add(panel);

        // Display the frame

        frame.pack();

        frame.setVisible(true);

    }

    public static void registorUtilizador(){

        Scanner sc =  new Scanner(System.in);

        System.out.println(

            "\nSistema de agendamento de consultas\n\n"+

            "1.Registar Paciente\n"+

            "2.Registar Funcionario\n\n"+

            "Escolha uma opcao"

        );

        int escolha;

        do{

            escolha = sc.nextInt();

            switch(escolha){

                case 1:

                    System.out.println("Brevemente");

                    escolha=0;

                    break;

                case 2:

                    System.out.println("Brevemente");

                    escolha=0;

                    break;

                case 0:

                    break;

                default:

                    System.out.println("Opcao invalida!");

            }

        }while(escolha!=0);

        sc.close();

    }

}

package src;

import javax.swing.\*;

import src.personel.Paciente;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class MainPanel extends JFrame {

    public MainPanel() {

        setTitle("Sistema de agendamento de consultas");

        setSize(800, 600);

        JPanel panel = new JPanel();

        panel.setLayout(new BorderLayout());

        JButton regFuncBtn = new JButton("Registrar funcionario");

        JButton visAgendaBtn =  new JButton("Visualizar agenda");

        JButton regPacBtn = new JButton("Registrar paciente");

        JButton confConsultBtn = new JButton("Confirmar consultas");

        JButton hisMedBtn = new JButton("Historico medico");

        JToolBar toolbar = new JToolBar();

        toolbar.add(regFuncBtn);

        toolbar.add(regPacBtn);

        toolbar.add(confConsultBtn);

        toolbar.add(confConsultBtn);

        toolbar.add(hisMedBtn);

        panel.add(toolbar, BorderLayout.NORTH);

        JTable table = new JTable();

        JScrollPane scrollPane = new JScrollPane(table);

        panel.add(scrollPane, BorderLayout.CENTER);

        // Adding action listeners to the buttons

        regPacBtn.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                 //dispose of the panel

                dispose();

                // Open panel for registering func

                MainPanel mainPanel = new MainPanel();

                mainPanel.setVisible(false);

                JFrame frame = new JFrame("Painel Registar de Paciente");

                frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

                frame.getContentPane().add(new PainelRegistarPac());

                frame.pack();

                frame.setVisible(true);

            }

        });

        regFuncBtn.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                //dispose of the panel

                dispose();

                // Open panel for registering func

                MainPanel mainPanel = new MainPanel();

                mainPanel.setVisible(false);

                JFrame frame = new JFrame("Painel Registar de Funcionario");

                frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

                frame.getContentPane().add(new PainelRegistarFunc());

                frame.pack();

                frame.setVisible(true);

            }

        });

        hisMedBtn.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                // Open panel for viewing schedule

                Paciente paciente=  new Paciente();

                int pacId = Integer.parseInt(JOptionPane.showInputDialog("Numero do Paciente"));

                for (Paciente Pacient : Paciente.pacientes) {

                    if(paciente.getPacId()==pacId)

                        JOptionPane.showMessageDialog(null, paciente.getHistoricoMedico(), "Historico Medico", ABORT);

                }

            }

        });

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        add(panel);

    }

}

package src;

import javax.swing.\*;

import src.db.JSqlCon;

import src.personel.Funcionario;

import src.personel.Medico;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class PainelRegistarFunc extends JPanel implements ActionListener {

    private JButton saveButton;

    private JButton cancelButton;

    private JTextField nomeField = new JTextField();

    private JTextField generoField = new JTextField();

    private JTextField dataNascimentoField = new JTextField();

    private JTextField numeroBIField = new JTextField();

    private JTextField nacionalidadeField = new JTextField();

    private JTextField numeroCelularField = new JTextField();

    private JTextField emailField = new JTextField();

    private JTextField enderecoField = new JTextField();

    private JTextField cargoField = new JTextField();

    private JTextField salarioField = new JTextField("0.00");

    private JTextField dataContratacaoField = new JTextField();

    private JTextField horarioTrabalhoField = new JTextField();

    private JTextField nomeUtilizadorField = new JTextField();

    private JTextField passwordField = new JTextField();

    private JTextField observacoesField = new JTextField();

    public PainelRegistarFunc() {

        setLayout(new GridLayout(22, 2));

        GridBagConstraints constraints = new GridBagConstraints();

        constraints.anchor = GridBagConstraints.WEST;

        constraints.insets = new Insets(10, 10, 10, 10);

        add(new JLabel("Nome:"));

        add(nomeField);

        add(new JLabel("Genero:"));

        add(generoField);

        add(new JLabel("Data de Nascimento:"));

        add(dataNascimentoField);

        add(new JLabel("Numero de BI:"));

        add(numeroBIField);

        add(new JLabel("Nacionalidade:"));

        add(nacionalidadeField);

        add(new JLabel("Numero de Celular:"));

        add(numeroCelularField);

        add(new JLabel("Email:"));

        add(emailField);

        add(new JLabel("Endereco:"));

        add(enderecoField);

        add(new JLabel("Cargo:"));

        add(cargoField);

        add(new JLabel("Salario:"));

        add(salarioField);

        add(new JLabel("Data de Contratacao:"));

        add(dataContratacaoField);

        add(new JLabel("Horario de Trabalho:"));

        add(horarioTrabalhoField);

        add(new JLabel("Nome de Utilizador:"));

        add(nomeUtilizadorField);

        add(new JLabel("Password:"));

        add(passwordField);

        add(new JLabel("Observacoes:"));

        add(observacoesField);

        saveButton = new JButton("Save");

        cancelButton = new JButton("Cancel");

        constraints.gridx = 0;

        constraints.gridy = 12;

        add(saveButton, constraints);

        constraints.gridx = 1;

        constraints.gridy = 12;

        add(cancelButton, constraints);

        saveButton.setActionCommand("save");

        cancelButton.setActionCommand("cancel");

        saveButton.addActionListener(this);

        cancelButton.addActionListener((ActionListener) this);

        setBorder(BorderFactory.createTitledBorder("Registar Funcionario"));

    }

    //@Override

    public void actionPerformed(ActionEvent e) {

        if (e.getActionCommand().equals("save")) {

            // ... handle save button action here ...

            Funcionario funcionario = new Funcionario();

            System.out.println(funcionario.getLoginName());

            int id = JSqlCon.getId("funcionario");

            Medico med = new Medico();

            funcionario.setNome(nomeField.getText());

            funcionario.setDataNasc(null);

            funcionario.setGenero(generoField.getText());

            funcionario.setBi(numeroBIField.getText());

            funcionario.setNacionalidade(nacionalidadeField.getText());

            funcionario.setEndereco(enderecoField.getText());

            funcionario.setCelular(numeroCelularField.getText());

            funcionario.setEmail(emailField.getText());

            funcionario.setCargo(cargoField.getText());

            funcionario.setDataContratacao(null);

            funcionario.setSalario(Float.parseFloat(salarioField.getText()));

            funcionario.setHorarioTrabalho(horarioTrabalhoField.getText());

            funcionario.setStatus(true);

            funcionario.setLoginName(nomeUtilizadorField.getText());

            funcionario.setLoginPwd(passwordField.getText());

            funcionario.setAccessPermiss(true);

            funcionario.setHistorico(null);

            funcionario.setObs(observacoesField.getText());

            funcionario.setFuncId(id);

            funcionario = new Funcionario(++id,funcionario.getNome(),funcionario.getDataNasc(),funcionario.getGenero(),funcionario.getBi(),

            funcionario.getNacionalidade(),funcionario.getEndereco(), funcionario.getCelular(), funcionario.getEmail(), funcionario.getCargo(),

            funcionario.getDataContratacao(),funcionario.getSalario(),funcionario.getHorarioTrabalho(), funcionario.getStatus(), funcionario.getLoginName(),

            funcionario.getLoginPwd(),funcionario.getAccessPermiss(), funcionario.getHistorico(), funcionario.getObs());

            Funcionario.funcionarios.add(funcionario);//salvar funcionarios no array

            JSqlCon.insertIntoFuncionario("funcionario", ++id, med.getEspecialidade(), med.getCRM() , funcionario.getCargo(), funcionario.getDataContratacao(),

            funcionario.getSalario(), funcionario.getNome(), funcionario.getDataNasc(), funcionario.getGenero(), funcionario.getNacionalidade(), funcionario.getEndereco(),

            funcionario.getEmail(), funcionario.getCelular(), funcionario.getStatus(), funcionario.getObs(), 1, funcionario.getHorarioTrabalho(), funcionario.getLoginName());

            Funcionario.salvarDados(Funcionario.funcionarios, null, "src/File/ficheiros/funcionarios.dat");

        } else if (e.getActionCommand().equals("cancel")) {

            // ... handle cancel button action here ...

            //dispose panel

            Window parentWindow = SwingUtilities.getWindowAncestor(PainelRegistarFunc.this);

            if (parentWindow != null) {

                // Dispose of the parent window

                parentWindow.dispose();

        }

            //open main panel

            MainPanel mainPanel = new MainPanel();

            mainPanel.setVisible(true);

        }

    }

}

package src;

import javax.swing.\*;

import src.db.JSqlCon;

import src.personel.Funcionario;

import src.personel.Paciente;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class PainelRegistarPac extends JPanel implements ActionListener {

    private JButton saveButton;

    private JButton cancelButton;

    private JTextField nomeField = new JTextField();

    private JTextField dataNascField = new JTextField();

    private JTextField generoField = new JTextField();

    private JTextField biField = new JTextField();

    private JTextField nacionalidadeField = new JTextField();

    private JTextField emergencyInfoField = new JTextField();

    private JTextField enderecoField = new JTextField();

    private JTextField emailField = new JTextField();

    private JTextField celularField = new JTextField();

    private JTextField estadoCivilField = new JTextField();

    private JTextField numeroSegSaudeField = new JTextField("0");

    private JTextField historicoMedicoField = new JTextField();

    private JTextField loginNameField = new JTextField();

    private JTextField loginPwdField = new JTextField();

    private JTextField accessPermissField = new JTextField();

    private JTextField statusField = new JTextField();

    private JTextField obsField = new JTextField();

    public PainelRegistarPac() {

        setLayout(new GridLayout(22, 2));

        GridBagConstraints constraints = new GridBagConstraints();

        constraints.anchor = GridBagConstraints.WEST;

        constraints.insets = new Insets(10, 10, 10, 10);

        add(new JLabel("Nome:"));

        add(nomeField);

        add(new JLabel("Genero:"));

        add(generoField);

        add(new JLabel("Data de Nascimento:"));

        add(dataNascField);

        add(new JLabel("Numero de BI:"));

        add(biField);

        add(new JLabel("Nacionalidade:"));

        add(nacionalidadeField);

        add(new JLabel("Numero de Celular:"));

        add(celularField);

        add(new JLabel("Email:"));

        add(emailField);

        add(new JLabel("Endereco:"));

        add(enderecoField);

        add(new JLabel("Estado civil:"));

        add(estadoCivilField);

        add(new JLabel("Numero de suguro de saude"));

        add(numeroSegSaudeField);

        add(new JLabel("Historico medico:"));

        add(historicoMedicoField);

        add(new JLabel("Informcacao de emergencia:"));

        add(emergencyInfoField);

        add(new JLabel("Nome de Utilizador:"));

        add(loginNameField);

        add(new JLabel("Password:"));

        add(loginPwdField);

        add(new JLabel("Observacoes:"));

        add(obsField);

        saveButton = new JButton("Save");

        cancelButton = new JButton("Cancel");

        constraints.gridx = 0;

        constraints.gridy = 12;

        add(saveButton, constraints);

        constraints.gridx = 1;

        constraints.gridy = 12;

        add(cancelButton, constraints);

        saveButton.setActionCommand("save");

        cancelButton.setActionCommand("cancel");

        saveButton.addActionListener(this);

        cancelButton.addActionListener((ActionListener) this);

        setBorder(BorderFactory.createTitledBorder("Registar Paciente"));

    }

    //@Override

    public void actionPerformed(ActionEvent e) {

        if (e.getActionCommand().equals("save")) {

            // ... handle save button action here ...

            Paciente paciente = new Paciente();

            int id = JSqlCon.getId("paciente");

            paciente.setNome(nomeField.getText());

            paciente.setDataNasc(null);

            paciente.setGenero(generoField.getText());

            paciente.setBi(biField.getText());

            paciente.setNacionalidade(nacionalidadeField.getText());

            paciente.setEndereco(enderecoField.getText());

            paciente.setCelular(celularField.getText());

            paciente.setEmail(emailField.getText());

            paciente.setEstadoCivil(estadoCivilField.getText());

            paciente.setNumeroSegSaude(Integer.parseInt(numeroSegSaudeField.getText()));

            paciente.setHistoricoMedico(historicoMedicoField.getText());

            paciente.setStatus(true);

            paciente.setEmergencyInfo(emergencyInfoField.getText());

            paciente.setLoginPwd(loginNameField.getText());

            paciente.setAccessPermiss(true);

            paciente.setLoginPwd(loginPwdField.getText());

            paciente.setObs(obsField.getText());

            paciente.setPacId(id);

            paciente = new Paciente(++id,

            paciente.getNome(),paciente.getDataNasc(),paciente.getGenero(),paciente.getBi(),paciente.getNacionalidade(),

            paciente.getEmergencyInfo(),paciente.getEndereco(),paciente.getEmail(),paciente.getCelular(),paciente.getEstadoCivil(),

            paciente.getNumeroSegSaude(),paciente.getHistoricoMedico(),paciente.getLoginName(),paciente.getLoginPwd(),

            paciente.getAccessPermiss(),paciente.getStatus(),paciente.getObs());

            Paciente.pacientes.add(paciente);//salvar Pacientes no array

            JSqlCon.insertIntoPaciente(id, TOOL\_TIP\_TEXT\_KEY, TOOL\_TIP\_TEXT\_KEY, id, null, TOOL\_TIP\_TEXT\_KEY, TOOL\_TIP\_TEXT\_KEY, TOOL\_TIP\_TEXT\_KEY, TOOL\_TIP\_TEXT\_KEY, TOOL\_TIP\_TEXT\_KEY, getFocusTraversalKeysEnabled(), TOOL\_TIP\_TEXT\_KEY, id);

            Funcionario.salvarDados(null, Paciente.pacientes, "src/File/ficheiros/pacientes.dat");

        } else if (e.getActionCommand().equals("cancel")) {

            // ... handle cancel button action here ...

            //dispose panel

            Window parentWindow = SwingUtilities.getWindowAncestor(PainelRegistarPac.this);

            if (parentWindow != null) {

                // Dispose of the parent window

                parentWindow.dispose();

        }

            //open main panel

            MainPanel mainPanel = new MainPanel();

            mainPanel.setVisible(true);

        }

    }

}

-- phpMyAdmin SQL Dump

-- version 5.2.1

-- https://www.phpmyadmin.net/

--

-- Host: 127.0.0.1

-- Generation Time: Oct 01, 2023 at 09:12 PM

-- Server version: 10.4.28-MariaDB

-- PHP Version: 8.2.4

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

START TRANSACTION;

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8mb4 \*/;

create database `clinicaeit`;

use `clinicaeit`;

--

-- Database: `clinicaeit`

--

-- --------------------------------------------------------

--

-- Table structure for table `consulta`

--

CREATE TABLE `consulta` (

  `ID` int(11) NOT NULL,

  `DataConsulta` date DEFAULT NULL,

  `PacienteID` int(11) DEFAULT NULL,

  `MedicoID` int(11) DEFAULT NULL,

  `ConsultaStatus` varchar(20) DEFAULT NULL,

  `Observacao` text DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Table structure for table `funcionario`

--

CREATE TABLE `funcionario` (

  `ID` int(11) NOT NULL,

  `Especialidade` varchar(255) DEFAULT NULL,

  `CRM` varchar(20) DEFAULT NULL,

  `Cargo` varchar(50) DEFAULT NULL,

  `DataContratacao` date DEFAULT NULL,

  `Salario` decimal(10,2) DEFAULT NULL,

  `Nome` varchar(60) NOT NULL,

  `DataNascimento` date DEFAULT NULL,

  `Genero` enum('Masculino','Feminino') DEFAULT NULL,

  `Nacionalidade` varchar(50) DEFAULT NULL,

  `Endereco` varchar(255) DEFAULT NULL,

  `Email` varchar(255) DEFAULT NULL,

  `Celular` varchar(20) DEFAULT NULL,

  `Status` varchar(20) DEFAULT NULL,

  `Observacao` text DEFAULT NULL,

  `TipoUsuarioID` int(11) DEFAULT NULL,

  `HorarioTrabalho` varchar(50) DEFAULT NULL,

  `LoginName` varchar(10) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

--

-- Dumping data for table `funcionario`

--

INSERT INTO `funcionario` (`ID`, `Especialidade`, `CRM`, `Cargo`, `DataContratacao`, `Salario`, `Nome`, `DataNascimento`, `Genero`, `Nacionalidade`, `Endereco`, `Email`, `Celular`, `Status`, `Observacao`, `TipoUsuarioID`, `HorarioTrabalho`, `LoginName`) VALUES

(1000, NULL, NULL, NULL, '2023-09-27', 113000.30, 'Joao Mateus', '1973-09-05', 'Masculino', 'Mocambicano', 'Cidade da Matola', 'joao.mateus@clinicaeit.co.mz', NULL, NULL, NULL, 3, NULL, 'JMateus1'),

(1001, 'Cardiologia', '1487565', 'Medico', '2023-09-27', 350000.00, 'John Rambo', '1965-09-28', 'Masculino', 'Mocambicano', 'Casa Branca', 'rambo@clinicaeit.co.mz', NULL, NULL, NULL, 1, NULL, 'JRambo1');

-- --------------------------------------------------------

--

-- Table structure for table `historicomedico`

--

CREATE TABLE `historicomedico` (

  `IDHistoricoMedico` int(11) NOT NULL,

  `IDPaciente` int(11) DEFAULT NULL,

  `IDMedico` int(11) DEFAULT NULL,

  `DataConsulta` date DEFAULT NULL,

  `Diagnostico` text DEFAULT NULL,

  `Tratamento` text DEFAULT NULL,

  `ResultadosExames` text DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Table structure for table `paciente`

--

CREATE TABLE `paciente` (

  `ID` int(11) NOT NULL,

  `LoginName` varchar(10) DEFAULT NULL,

  `Nome` varchar(255) NOT NULL,

  `NumeroSeguroSaude` varchar(20) DEFAULT NULL,

  `DataNascimento` date DEFAULT NULL,

  `Genero` enum('Masculino','Feminino') DEFAULT NULL,

  `Nacionalidade` varchar(50) DEFAULT NULL,

  `Endereco` varchar(255) DEFAULT NULL,

  `Email` varchar(255) DEFAULT NULL,

  `Celular` varchar(20) DEFAULT NULL,

  `Status` varchar(20) DEFAULT NULL,

  `Observacao` text DEFAULT NULL,

  `TipoUsuarioID` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

--

-- Dumping data for table `paciente`

--

INSERT INTO `paciente` (`ID`, `LoginName`, `Nome`, `NumeroSeguroSaude`, `DataNascimento`, `Genero`, `Nacionalidade`, `Endereco`, `Email`, `Celular`, `Status`, `Observacao`, `TipoUsuarioID`) VALUES

(999999, 'CJ01', 'Carl Jonson', '547821235', '1998-01-01', 'Masculino', 'Mocambicano', NULL, NULL, '+2584552562', NULL, NULL, 2);

-- --------------------------------------------------------

--

-- Table structure for table `tipoperfil`

--

CREATE TABLE `tipoperfil` (

  `ID` int(11) NOT NULL,

  `NomePerfil` varchar(50) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

--

-- Dumping data for table `tipoperfil`

--

INSERT INTO `tipoperfil` (`ID`, `NomePerfil`) VALUES

(1, 'medico'),

(2, 'paciente'),

(3, 'admin');

--

-- Indexes for dumped tables

--

--

-- Indexes for table `consulta`

--

ALTER TABLE `consulta`

  ADD PRIMARY KEY (`ID`),

  ADD KEY `PacienteID` (`PacienteID`),

  ADD KEY `MedicoID` (`MedicoID`);

--

-- Indexes for table `funcionario`

--

ALTER TABLE `funcionario`

  ADD PRIMARY KEY (`ID`),

  ADD KEY `TipoUsuarioID` (`TipoUsuarioID`);

--

-- Indexes for table `historicomedico`

--

ALTER TABLE `historicomedico`

  ADD PRIMARY KEY (`IDHistoricoMedico`),

  ADD KEY `IDPaciente` (`IDPaciente`),

  ADD KEY `IDMedico` (`IDMedico`);

--

-- Indexes for table `paciente`

--

ALTER TABLE `paciente`

  ADD PRIMARY KEY (`ID`),

  ADD KEY `TipoUsuarioID` (`TipoUsuarioID`);

--

-- Indexes for table `tipoperfil`

--

ALTER TABLE `tipoperfil`

  ADD PRIMARY KEY (`ID`);

--

-- AUTO\_INCREMENT for dumped tables

--

--

-- AUTO\_INCREMENT for table `consulta`

--

ALTER TABLE `consulta`

  MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT for table `funcionario`

--

ALTER TABLE `funcionario`

  MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=1002;

--

-- AUTO\_INCREMENT for table `historicomedico`

--

ALTER TABLE `historicomedico`

  MODIFY `IDHistoricoMedico` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT for table `paciente`

--

ALTER TABLE `paciente`

  MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=1000000;

--

-- AUTO\_INCREMENT for table `tipoperfil`

--

ALTER TABLE `tipoperfil`

  MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=4;

--

-- Constraints for dumped tables

--

--

-- Constraints for table `consulta`

--

ALTER TABLE `consulta`

  ADD CONSTRAINT `consulta\_ibfk\_1` FOREIGN KEY (`PacienteID`) REFERENCES `paciente` (`ID`),

  ADD CONSTRAINT `consulta\_ibfk\_2` FOREIGN KEY (`MedicoID`) REFERENCES `funcionario` (`ID`);

--

-- Constraints for table `funcionario`

--

ALTER TABLE `funcionario`

  ADD CONSTRAINT `funcionario\_ibfk\_1` FOREIGN KEY (`TipoUsuarioID`) REFERENCES `tipoperfil` (`ID`);

--

-- Constraints for table `historicomedico`

--

ALTER TABLE `historicomedico`

  ADD CONSTRAINT `historicomedico\_ibfk\_1` FOREIGN KEY (`IDPaciente`) REFERENCES `paciente` (`ID`),

  ADD CONSTRAINT `historicomedico\_ibfk\_2` FOREIGN KEY (`IDMedico`) REFERENCES `funcionario` (`ID`);

--

-- Constraints for table `paciente`

--

ALTER TABLE `paciente`

  ADD CONSTRAINT `paciente\_ibfk\_1` FOREIGN KEY (`TipoUsuarioID`) REFERENCES `tipoperfil` (`ID`);

COMMIT;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;