MANUAL TÉCNICO

DAVID BORREGO ASENCIO

ÍNDICE

MANUAL TÉCNICO

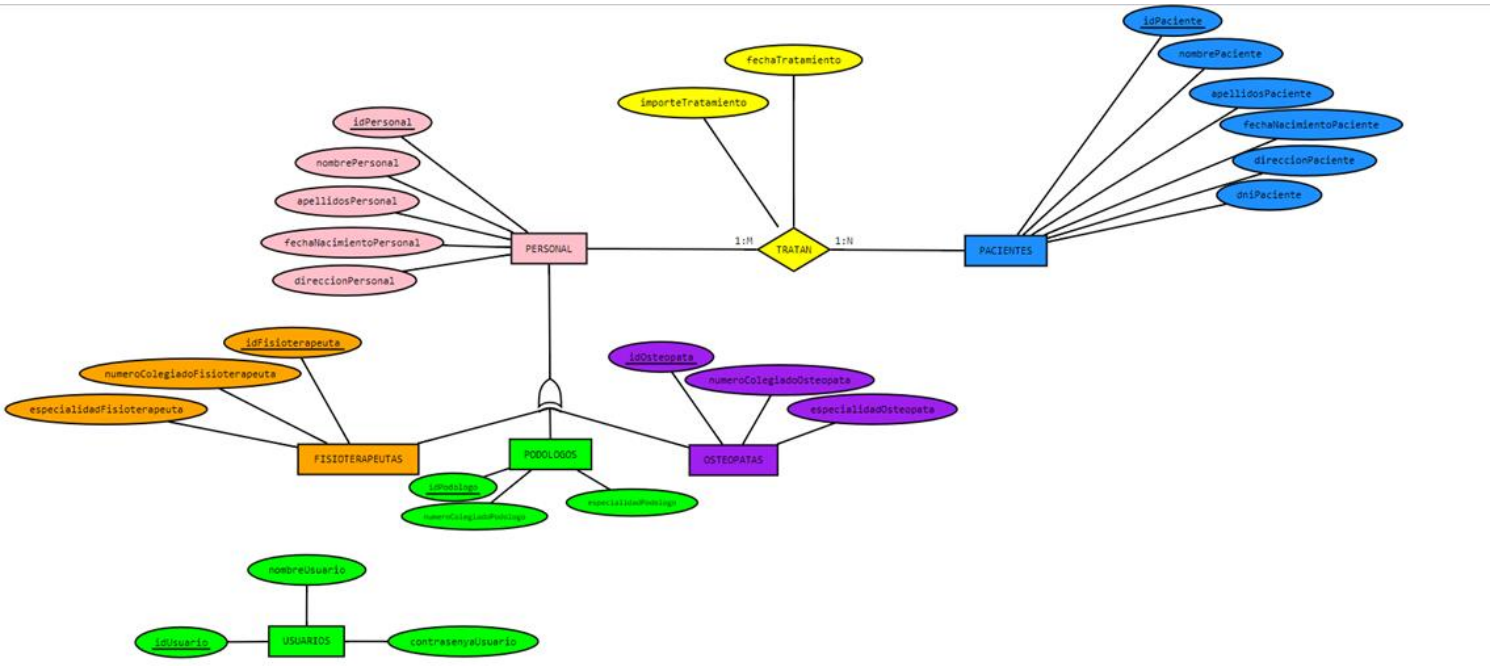
Introducción

En el presente **manual técnico**, se describen los siguientes aspectos referentes a la aplicación desarrollada para la gestión de la base de datos de una políclína de fisioterapia y pododología:

- ✓ Modelo de datos (diagrama E-R, esquema relacional y diagrama en Workbench).
- ✓ Sentencias SQL de creación de la BD y las tablas.
- ✓ Librerías externas utilizadas para el desarrollo de la aplicación.
- ✓ Requisitos técnicos imprescindibles para la ejecución del programa.
- ✓ Explicación de las clases JAVA desarrolladas.

Funcionalidad

Diagrama E-R



Esquema Relacional

PERSONAL (**idPersonal**, nombrePersonal, apellidosPersonal, fechaNacimientoPersonal, direccionPersonal)

PACIENTES (**idPaciente**, nombrePaciente, apellidosPaciente, fechaNacimientoPaciente, direccionPaciente, dniPaciente)

FISIOTERAPEUTAS **(idFisioterapeuta**, numeroColegiadoFisioterapeuta, especialidadFisioterapeuta, IdPersonalFK)

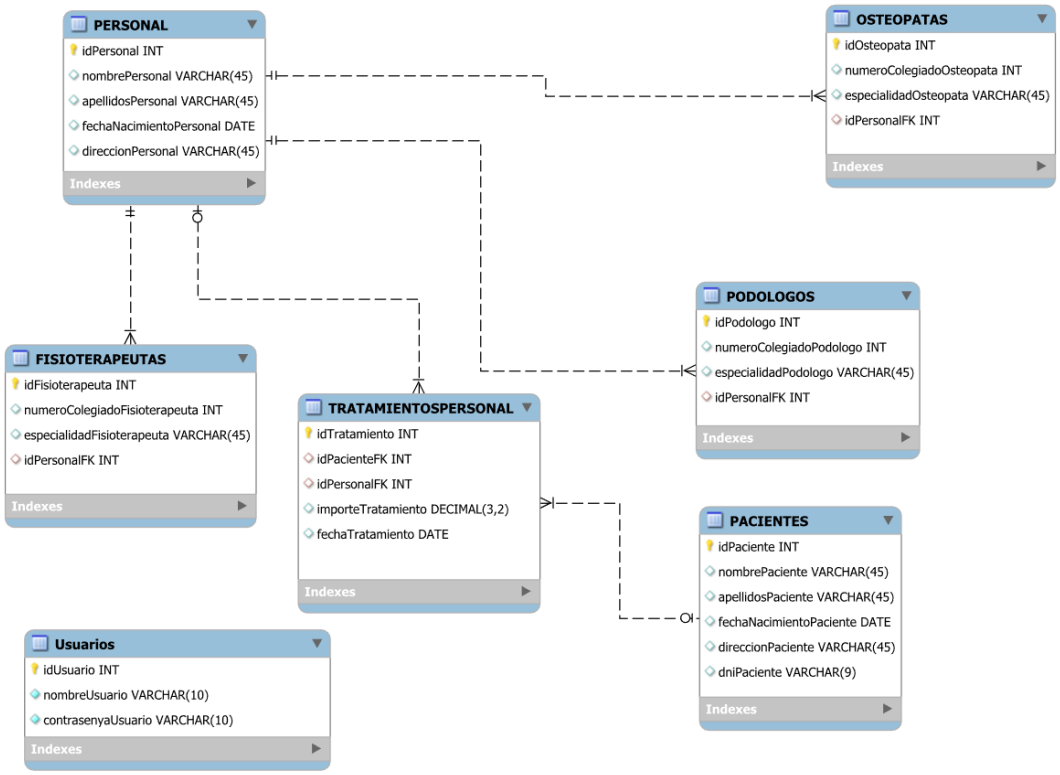
PODOLOGOS (**idPodologo**, numeroColegiadoPodologo, especialidadPodologo, IdPersonalFK)

OSTEOPATAS (**idOsteopata**, numeroColegiadoOsteopata, especialidadOsteopata, IdPersonalFK)

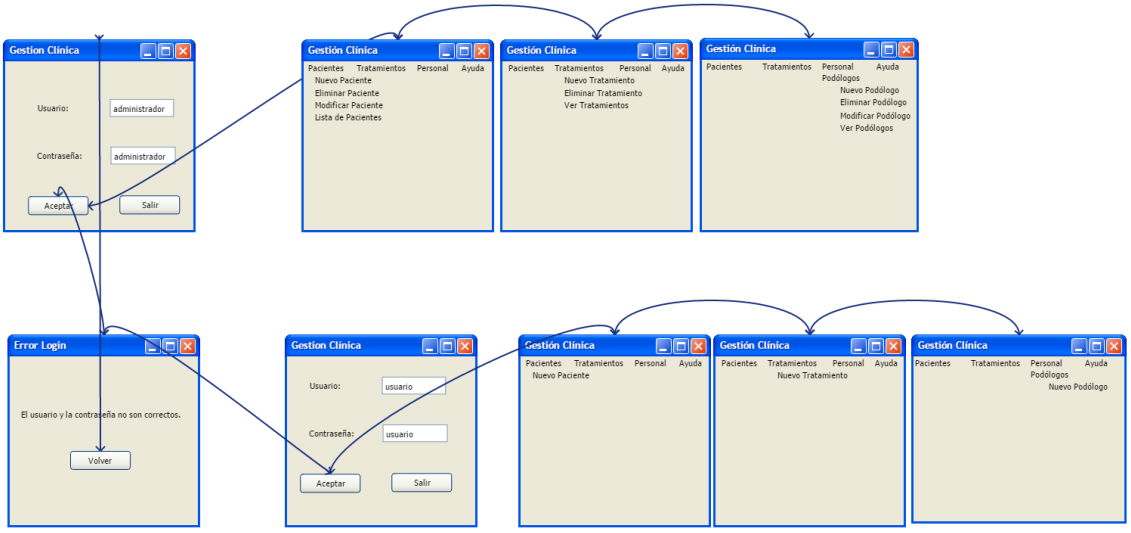
TRATAMIENTOSPERSONAL (**idTratamiento,** idPacienteFK, idPersonalFK, importeTratamiento, fechaTratamiento)

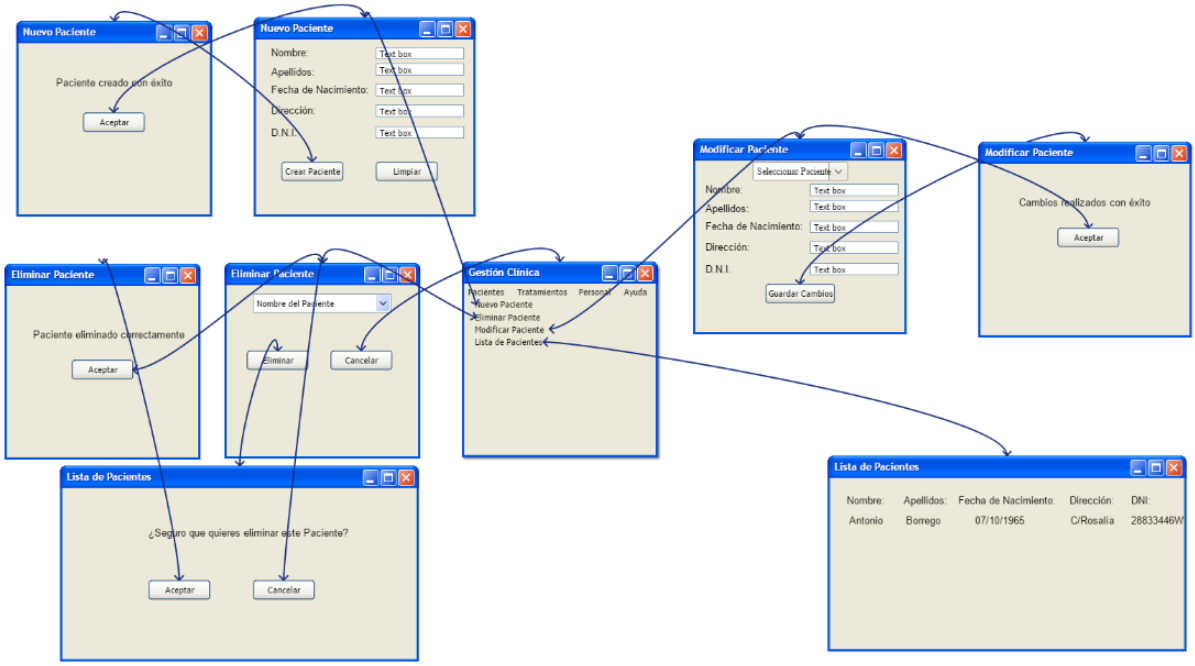
USUARIOS (**idUsuario**, nombreUsuario, contrasenyaUsuario)

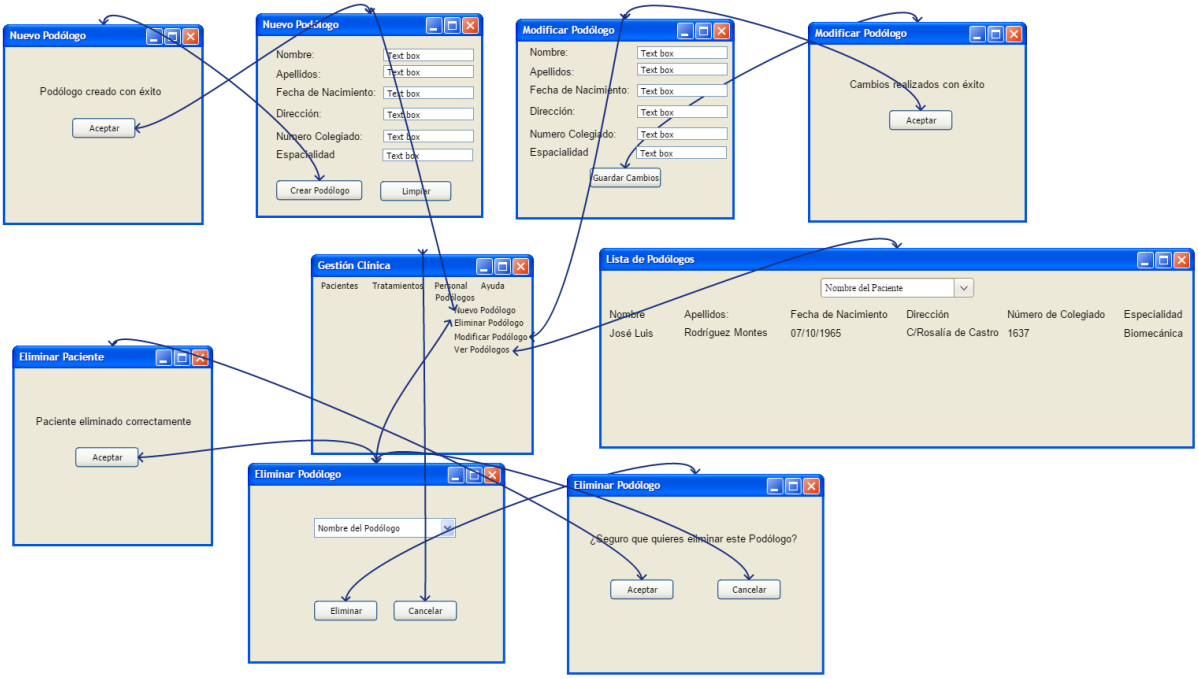
Diagrama Workbench

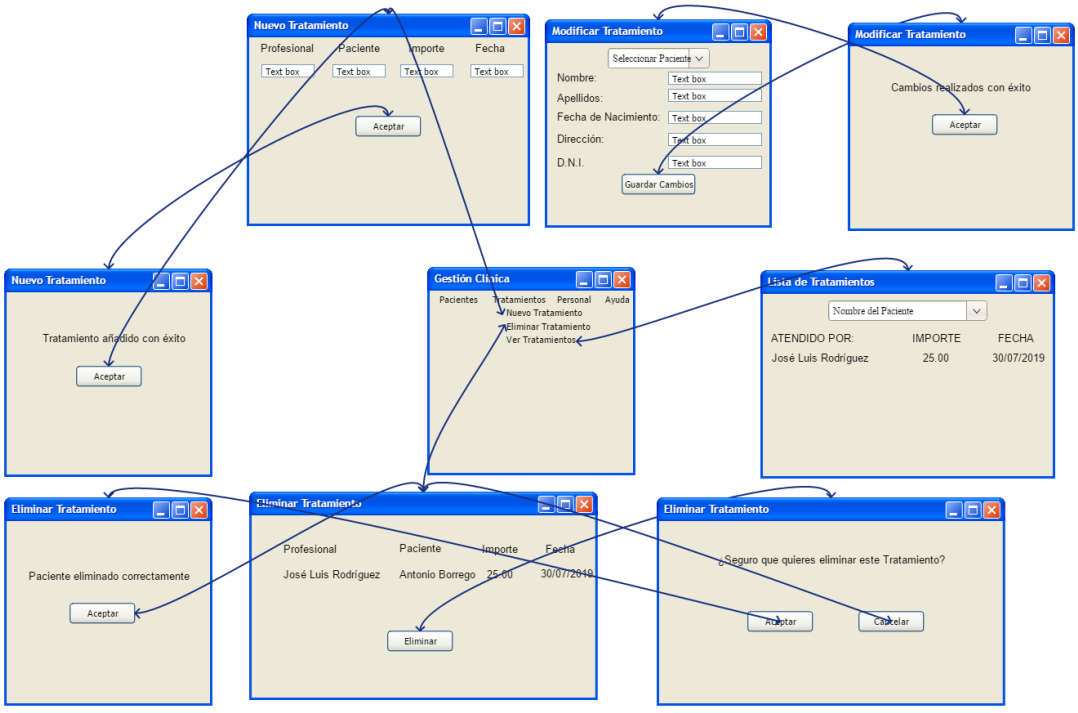


Diseño de Interfaz






Nuevo Podólogo
Podólogo creado con éxito
Aceptar
Nuevo Podólogo
Nombre:
Apellidos:
Fecha de Nacimiento:
Dirección:
Numero Colegiado:
Especialidad
Text box
Crear Podólogo
Limpiar
Modificar Podólogo
Guardar Cambios
Cambios realizados con éxito
Gestión Clínica
Pacientes
Tratamientos
Personal
Ayuda
Podólogos
Nuevo Podólogo
Eliminar Podólogo
Modificar Podólogo
Ver Podólogos
Lista de Podólogos
Nombre del Paciente
Nombre
Apellidos:
Fecha de Nacimiento
Dirección
Número de Colegiado
Especialidad
José Luis
Rodríguez Montes
07/10/1965
C/Rosalía de Castro
1637
Biomecánica
Eliminar Paciente
Paciente eliminado correctamente
Eliminar Podólogo
Nombre del Podólogo
Eliminar
Cancelar
¿Seguro que quieres eliminar este Podólogo?


Nuevo Tratamiento
Profesional
Paciente
Importe
Fecha
Text box
Aceptar
Modificar Tratamiento
Seleccionar Paciente
Nombre:
Apellidos:
Fecha de Nacimiento:
Dirección:
D.N.I.
Guardar Cambios
Cambios realizados con éxito
Tratamiento añadido con éxito
Gestión Clínica
Pacientes
Tratamientos
Personal
Ayuda
Nuevo Tratamiento
Eliminar Tratamiento
Ver Tratamientos
Lista de Tratamientos
Nombre del Paciente
ATENDIDO POR:
IMPORTE
FECHA
José Luis Rodríguez
25.00
30/07/2019
Eliminar Tratamiento
Paciente eliminado correctamente
Profesional
Paciente
Importe
Fecha
José Luis Rodríguez
Antonio Borrego
25.00
30/07/2019
Eliminar
¿Seguro que quieres eliminar este Tratamiento?
Cancelar

Sentencias SQL creación de la BD

Sentencia create database

```
CREATE DATABASE policlinica CHARACTER SET utf8mb4 COLLATE utf8mb4_spanish2_ci;
```

Sentencias create table

```
USE policlinica;
CREATE TABLE personal (
   idPersonal INT(11) AUTO_INCREMENT,
   nombrePersonal VARCHAR(45),
   apellidosPersonal VARCHAR(45),
   fechaNacimientoPersonal DATE,
   direccionPersonal VARCHAR(45),
   PRIMARY KEY (idPersonal)
);
CREATE TABLE pacientes (
   idPaciente INT(11) AUTO_INCREMENT,
   nombrePaciente VARCHAR(45),
   apellidosPaciente VARCHAR(45),
   fechaNacimientoPaciente DATE,
   direccionPaciente VARCHAR(45),
   dniPaciente VARCHAR(9),
   PRIMARY KEY (idPaciente)
);
```

```
CREATE TABLE fisioterapeutas (
   idFisioterapeuta INT(11) AUTO_INCREMENT,
   numeroColegiadoFisioterapeuta VARCHAR(45),
   especialidadFisioterapeuta VARCHAR(45),
   idPersonalFK INT(11),
   PRIMARY KEY (idFisioterapeuta),
   FOREIGN KEY (idPersonalFK)
      REFERENCES personal (idPersonal)
);
CREATE TABLE podologos (
   idPodologo INT(11) AUTO_INCREMENT,
   numeroColegiadoPodologo VARCHAR(45),
   especialidadPodologo VARCHAR(45),
   idPersonalFK INT(11),
   PRIMARY KEY (idPodologo),
   FOREIGN KEY (idPersonalFK)
      REFERENCES personal (idPersonal)
);
CREATE TABLE osteopatas (
   idOsteopata INT(11) AUTO_INCREMENT,
   numeroColegiadoOsteopata VARCHAR(45),
   especialidadOsteopata VARCHAR(45),
   idPersonalFK INT(11),
   PRIMARY KEY (idOsteopata),
   FOREIGN KEY (idPersonalFK)
      REFERENCES personal (idPersonal)
);
```

```
CREATE TABLE tratamientoPersonal (
   idTratamiento INT(11)  AUTO_INCREMENT,
   idPacienteFK INT(11),
   idPersonalFK INT(11),
   importeTratamiento DECIMAL(10,2),
   fechaTratamiento DATE,
   PRIMARY KEY (idTratamiento),
   FOREIGN KEY (idPacienteFK)
      REFERENCES pacientes (idPaciente),
          FOREIGN KEY (idPersonalFK)
      REFERENCES personal (idPersonal)
);

CREATE TABLE usuarios (
  idUsuario INT(11) AUTO_INCREMENT,
  nombreUsuario VARCHAR(15),
  contrasenyaUsuario VARCHAR(15),
  PRIMARY KEY (idUsuario)
);

INSERT INTO  usuarios (nombreUsuario, contrasenyaUsuario) VALUES ('admin', 'admin');
INSERT INTO  usuarios (nombreUsuario, contrasenyaUsuario) VALUES ('user', 'user');
```

Código fuente en JAVA

Clase Ayuda

```
package es.grupostudium.GestionClinica;


import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.IOException;
import javax.swing.JFrame;

public class Ayuda extends JFrame implements WindowListener, ActionListener
{
	private static final long serialVersionUID = 1L;


	public Ayuda()
	{


		try
		{
		Runtime.getRuntime().exec("hh.exe Ayuda.chm");
		}
		catch (IOException e)
		{
		e.printStackTrace();
		}


	}


	public void windowActivated(WindowEvent we){}

	public void windowClosed(WindowEvent we){}

	public void windowClosing(WindowEvent we)

	{

		this.setVisible(false);

	}

	public void windowDeactivated(WindowEvent we){}

	public void windowDeiconified(WindowEvent we){}

	public void windowIconified(WindowEvent we){}

	public void windowOpened(WindowEvent we){}
```

```
	public void actionPerformed(ActionEvent ae)
	{

	}
}
```

Clase ConectaBBDD

```
package es.grupostudium.GestionClinica;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class ConectaBBDD
{


	String driver = "com.mysql.jdbc.Driver";
	String url
="jdbc:mysql://localhost:3306/policlinica?autoReconnect=true&useSSL=false";
	String login = "administrativo";
	String password = "studium2019;";
	String sentencia = null;
	Connection connection = null;
	Statement statement = null;
	ResultSet rs = null;

	public ConectaBBDD()
	{

		try
		{
			Class.forName(driver);
		}
		catch(ClassNotFoundException e)
		{
			System.out.println("Se ha producido un error al cargar el
Driver");
		}

		try
		{
			connection = DriverManager.getConnection(url, login,
password);

		}
		catch(SQLException e)
		{
			System.out.println("Se produjo un error al conectar a la
Base de Datos");
		}
	}
```

```
public boolean agregar_paciente(String sentencia)
{

	try
	{
		statement=connection.createStatement();
		statement.executeUpdate(sentencia);
		statement.close();
		connection.close();
		return true;
	}
	catch(SQLException e)
	{
		System.out.println("Error en la sentencia SQL");
		return false;
	}
}

public boolean agregar_personal(String sentencia)
{

	try
	{
		statement=connection.createStatement();
		statement.executeUpdate(sentencia);
		statement.close();
		connection.close();
		return true;
	}
	catch(SQLException e)
	{
		System.out.println("Error en la sentencia SQL");
		return false;
	}
}

public boolean agregar_podologo(String sentencia)
{

	try
	{
		statement=connection.createStatement();
		statement.executeUpdate(sentencia);
		statement.close();
		connection.close();
		return true;
	}
	catch(SQLException e)
	{
		System.out.println("Error AQUÍ");
		return false;
	}
}

public Object obtener_id_de_los_pacientes(String sentencia)
{
```

```
        try
        {
                statement=connection.createStatement();
                rs= statement.executeQuery(sentencia);
                return rs;
        }
        catch(SQLException e)
        {
                System.out.println("Error en la sentencia SQL");
                return rs;
        }
}

public Object obtener_nombres_de_los_pacientes(String sentencia)
{

        try
        {
                statement=connection.createStatement();
                rs= statement.executeQuery(sentencia);
                return rs;
        }
        catch(SQLException e)
        {
                System.out.println("Error en la sentencia SQL");
                return rs;
        }
}
public Object obtener_apellidos_de_los_pacientes(String sentencia)
{

        try
        {
                statement=connection.createStatement();
                rs= statement.executeQuery(sentencia);
                return rs;
        }
        catch(SQLException e)
        {
                System.out.println("Error en la sentencia SQL");
                return rs;
        }
}

public Object obtener_id_del_personal(String sentencia)
{

        try
        {
                statement=connection.createStatement();
                rs= statement.executeQuery(sentencia);
                return rs;
        }
        catch(SQLException e)
        {
                System.out.println("Error en la sentencia SQL");
                return rs;
```

```
		}
	}

	public Object obtener_id_de_los_podologos(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			rs= statement.executeQuery(sentencia);
			return rs;
		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return rs;
		}
	}

	public Object obtener_nombres_del_personal(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			rs= statement.executeQuery(sentencia);
			return rs;
		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return rs;
		}
	}
	public Object obtener_apellidos_del_personal(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			rs= statement.executeQuery(sentencia);
			return rs;
		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return rs;
		}
	}

	public Object nombre_personal_que_atiende(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			rs= statement.executeQuery(sentencia);
			return rs;
```

```
		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return rs;
		}
	}

	public Object obtener_todos_los_datos_de_los_pacientes(String
sentencia)
	{

		try
		{
			statement=connection.createStatement();
			rs= statement.executeQuery(sentencia);
			return rs;
		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return rs;
		}
	}

	public Object obtener_tratamientos(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			rs= statement.executeQuery(sentencia);
			return rs;
		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return rs;
		}
	}

	public Object obtener_todos_los_datos_del_personal(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			rs= statement.executeQuery(sentencia);
			return rs;
		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return rs;
		}
	}
```

```
	public Object obtener_todos_los_datos_de_los_podologos(String
sentencia)
	{

		try
		{
			statement=connection.createStatement();
			rs= statement.executeQuery(sentencia);
			return rs;
		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return rs;
		}
	}

	public void eliminar_paciente(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			statement.executeUpdate(sentencia);
			statement.close();
			connection.close();

		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");

		}
	}
	public boolean eliminar_podologo(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			statement.executeUpdate(sentencia);
			statement.close();
			connection.close();
			return true;

		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return false;
		}
	}

	public boolean eliminar_personal(String sentencia)
	{

		try
		{
```

```
                statement=connection.createStatement();
                statement.executeUpdate(sentencia);
                statement.close();
                connection.close();
                return true;
        }
        catch(SQLException e)
        {
                System.out.println("Error en la sentencia SQL");
                return false;
        }
}
public void eliminar_tratamiento(String sentencia)
{

        try
        {
                statement=connection.createStatement();
                statement.executeUpdate(sentencia);
                statement.close();
                connection.close();

        }
        catch(SQLException e)
        {
                System.out.println("Error en la sentencia SQL");

        }
}

public boolean modificar_paciente(String sentencia)
{

        try
        {
                statement=connection.createStatement();
                statement.executeUpdate(sentencia);
                statement.close();
                connection.close();
                return true;
        }
        catch(SQLException e)
        {
                System.out.println("Error en la sentencia SQL");
                return false;
        }
}

public boolean modificar_podologo(String sentencia)
{

        try
        {
                statement=connection.createStatement();
                statement.executeUpdate(sentencia);
                statement.close();
                connection.close();
                return true;
```

```
		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return false;
		}
	}

	public boolean modificar_personal(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			statement.executeUpdate(sentencia);
			statement.close();
			connection.close();
			return true;

		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL");
			return false;
		}
	}

	public boolean agregar_tratamiento(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			statement.executeUpdate(sentencia);
			statement.close();
			connection.close();
			return true;

		}
		catch(SQLException e)
		{
			System.out.println("Error en la sentencia SQL agregar
tratamiento.");
			return false;
		}
	}

	public boolean comprobar_Usuario(String sentencia)
	{

		try
		{
			statement=connection.createStatement();
			rs= statement.executeQuery(sentencia);

			if (rs.next())
			{
```

```
                        return true;
                }
                else
                {
                        return false;
                }

        }
        catch(SQLException e)
        {
                System.out.println("Error en la sentencia SQL");
                return false;
        }
    }
}
```

Clase GestionClinica1

```
package es.grupostudium.GestionClinica;

import java.awt.FlowLayout;
import java.awt.Graphics2D;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.awt.image.BufferedImage;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;

public class GestionClinica1 extends JFrame implements WindowListener,
ActionListener
{

    private static final long serialVersionUID = 1L;

    JMenuBar barraMenu = new JMenuBar();

    JMenu pacientes = new JMenu("Pacientes");
    JMenu tratamientos = new JMenu("Tratamientos");
    JMenu personal = new JMenu("Personal");
    JMenu ayuda = new JMenu("Ayuda");

    JMenuItem nuevoPaciente = new JMenuItem("Nuevo Paciente");
    JMenuItem eliminarPaciente = new JMenuItem("Eliminar Paciente");
    JMenuItem modificarPaciente = new JMenuItem("Modificar Paciente");
    JMenuItem listadePacientes = new JMenuItem("Lista de Pacientes");

    JMenuItem nuevoTratamiento = new JMenuItem("Nuevo Tratamiento");
    JMenuItem eliminarTratamiento = new JMenuItem("Eliminar Tratamiento");
    JMenuItem verTratamientos = new JMenuItem("Ver Tratamientos");

    JMenu podologosPersonal = new JMenu("Podólogos");
    JMenuItem nuevoPodologo = new JMenuItem("Nuevo Podólogo");
    JMenuItem eliminarPodologo = new JMenuItem("Eliminar Podólogo");
```

```
JMenuItem modificarPodologo = new JMenuItem("Modificar Podólogos");
JMenuItem verPodologos = new JMenuItem("Ver Podólogos");

JMenuItem manual = new JMenuItem("Ver Manual de Usuario");



public GestionClinica1()
{

	setLayout(new FlowLayout());
	setTitle("CLÍNICA SALLE");
	setSize(800,600);
	setLocationRelativeTo(null);

	setJMenuBar(barraMenu);

	pacientes.add(nuevoPaciente);
	pacientes.add(eliminarPaciente);
	pacientes.add(modificarPaciente);
	pacientes.add(listadePacientes);

	nuevoPaciente.addActionListener(this);
	eliminarPaciente.addActionListener(this);
	modificarPaciente.addActionListener(this);
	listadePacientes.addActionListener(this);

	tratamientos.add(nuevoTratamiento);
	tratamientos.add(eliminarTratamiento);
	tratamientos.add(verTratamientos);

	nuevoTratamiento.addActionListener(this);
	eliminarTratamiento.addActionListener(this);
	verTratamientos.addActionListener(this);

	podologosPersonal.add(nuevoPodologo);
	podologosPersonal.add(eliminarPodologo);
	podologosPersonal.add(modificarPodologo);
	podologosPersonal.add(verPodologos);
	personal.add(podologosPersonal);

	nuevoPodologo.addActionListener(this);
	eliminarPodologo.addActionListener(this);
	modificarPodologo.addActionListener(this);
	verPodologos.addActionListener(this);

	ayuda.add(manual);
	manual.addActionListener(this);

	barraMenu.add(pacientes);
	barraMenu.add(tratamientos);
	barraMenu.add(personal);
	barraMenu.add(ayuda);



	ImageIcon image = new ImageIcon("PRINCIPAL1.png");
```

```
            int scale = 1;

            int width = image.getIconWidth();
            int height = image.getIconHeight();
            BufferedImage buffer = new BufferedImage(scale * width, scale *
height, BufferedImage.TYPE_INT_ARGB);
            Graphics2D graphics = buffer.createGraphics();

            graphics.scale(scale,scale);
            image.paintIcon(null, graphics, 0, 0);
            graphics.dispose();
            JLabel label = new JLabel(new ImageIcon(buffer));
            add(label);


            addWindowListener(this);
            setVisible(true);


      }



      public void actionPerformed(ActionEvent ae)
      {
            Object a;
            a = ae.getSource();

            if(a.equals(manual))
            {
                  new Ayuda();

            }

            if(a.equals(nuevoPaciente))
            {
                  new VentanaNuevoPaciente();
            }else
            {
                  if(a.equals(eliminarPaciente))
                  {
                        new VentanaEliminarPaciente();

                  }else
                  {
                        if(a.equals(modificarPaciente))
                        {
                              new VentanaModificarPaciente();
                        }else
                        {
                              if(a.equals(listadePacientes))
                              {
                                    new VentanaListaPacientes();
                              }
                        }
                  }
            }
```

```
			if(a.equals(nuevoPodologo))
			{
				new VentanaNuevoPodologo();
			}else
			{
				if(a.equals(eliminarPodologo))
				{
					new VentanaEliminarPodologo();
				}
				else
				{
					if(a.equals(modificarPodologo))
					{
						new VentanaModificarPodologo();
					}
					else
					{
						if(a.equals(verPodologos))
						{
							new VentanaListaPodologos();
						}
					}
				}
			}

			if(a.equals(nuevoTratamiento))
			{
				new VentanaNuevoTratamiento();

			}else
				if(a.equals(eliminarTratamiento))
				{
					new VentanaEliminarTratamiento();

				}
				else
				{
					if(a.equals(verTratamientos))
					{
						new VentanaListaTratamientos();
					}
				}

		}
		public void windowActivated(WindowEvent we){}
		public void windowClosed(WindowEvent we)
		{
			System.exit(0);
		}
		public void windowDeactivated(WindowEvent we){}
		public void windowDeiconified(WindowEvent we){}
		public void windowIconified(WindowEvent we){}
		public void windowOpened(WindowEvent we){}
		public void windowClosing(WindowEvent we){}

}
```

Clase GestionClinica2

```
package es.grupostudium.GestionClinica;

import java.awt.FlowLayout;
import java.awt.Graphics2D;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.awt.image.BufferedImage;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;

public class GestionClinica2 extends JFrame implements WindowListener,
ActionListener
{

	private static final long serialVersionUID = 1L;

	JMenuBar barraMenu = new JMenuBar();

	JMenu pacientes = new JMenu("Pacientes");
	JMenu tratamientos = new JMenu("Tratamientos");
	JMenu personal = new JMenu("Personal");
	JMenu ayuda = new JMenu("Ayuda");

	JMenuItem nuevoPaciente = new JMenuItem("Nuevo Paciente");

	JMenuItem nuevoTratamiento = new JMenuItem("Nuevo Tratamiento");

	JMenu podologosPersonal = new JMenu("Podólogos");
	JMenuItem nuevoPodologo = new JMenuItem("Nuevo Podólogo");

	JMenuItem manual = new JMenuItem("Ver Manual de Usuario");



	public GestionClinica2()
	{

		setLayout(new FlowLayout());
		setTitle("Gestión Clínica");
		setSize(800,600);
		setLocationRelativeTo(null);

		setJMenuBar(barraMenu);

		nuevoPaciente.addActionListener(this);
		pacientes.add(nuevoPaciente);

		nuevoTratamiento.addActionListener(this);
		tratamientos.add(nuevoTratamiento);
```

```
            nuevoPodologo.addActionListener(this);
            podologosPersonal.add(nuevoPodologo);

            personal.add(podologosPersonal);

            ayuda.add(manual);
            manual.addActionListener(this);

            barraMenu.add(pacientes);
            barraMenu.add(tratamientos);
            barraMenu.add(personal);
            barraMenu.add(ayuda);


            ImageIcon image = new ImageIcon("PRINCIPAL1.png");
            int scale = 1;

            int width = image.getIconWidth();
            int height = image.getIconHeight();
            BufferedImage buffer = new BufferedImage(scale * width, scale *
height, BufferedImage.TYPE_INT_ARGB);
            Graphics2D graphics = buffer.createGraphics();

            graphics.scale(scale,scale);
            image.paintIcon(null, graphics, 0, 0);
            graphics.dispose();
            JLabel label = new JLabel(new ImageIcon(buffer));
            add(label);


            addWindowListener(this);
            setVisible(true);



      }


      public void actionPerformed(ActionEvent ae)
      {
            Object a;
            a = ae.getSource();

            if(a.equals(manual))
            {
                  new Ayuda();

            }

            if(a.equals(nuevoPaciente))
            {
                  new VentanaNuevoPaciente();
            }

            if(a.equals(nuevoPodologo))
            {
                  new VentanaNuevoPodologo();
            }
```

```
			if(a.equals(nuevoTratamiento))
			{
				new VentanaNuevoTratamiento();

			}
		}
		public void windowActivated(WindowEvent we){}
		public void windowClosed(WindowEvent we){}
		public void windowClosing(WindowEvent we)
		{

			System.exit(0);
		}
		public void windowDeactivated(WindowEvent we){}
		public void windowDeiconified(WindowEvent we){}
		public void windowIconified(WindowEvent we){}
		public void windowOpened(WindowEvent we){}

}
```

Clase Login

```
package es.grupostudium.GestionClinica;

import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.Graphics2D;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.awt.image.BufferedImage;
import java.sql.ResultSet;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;

public class Login extends JFrame implements WindowListener, ActionListener
{
	private static final long serialVersionUID = 1L;

	GridBagLayout gridbag = new GridBagLayout();
	GridBagConstraints constraints = new GridBagConstraints();

	JPanel panel1 = new JPanel();
	JPanel panel2 = new JPanel();

	JLabel usuario = new JLabel("Usuario:");
	JLabel contrasenya = new JLabel("Contraseña:");
```

```
	JLabel derechosAutor = new JLabel("Desarrollado por: David Borrego
Asencio - Todos los derechos reservados © Version: 1.0");

	JTextField textousuario = new JTextField(15);
	JPasswordField textocontrasenya = new JPasswordField(15);

	JButton btnAceptar = new JButton("Aceptar");
	JButton btnLimpiar = new JButton(" Limpiar ");


	JDialog dialogo0 = new JDialog(this, "LOGIN – CLÍNICA SALLE", false);
	JLabel mensaje0 = new JLabel("Usuario y Contraseña Incorrectos.");
	JButton btnAceptar0= new JButton("Volver a intentar");

	ResultSet usuarios_contraseñas = null;

	Login()
	{
		setLayout(new BorderLayout());
		setTitle("LOGIN – CLÍNICA SALLE");
		setResizable(true);
		setSize(800, 400);
		setLocationRelativeTo(null);
		setResizable(false);

		panel1.setLayout(gridbag);
		panel2.setLayout(new FlowLayout());


		//INSERTAR LA IMAGEN EN EL LAYOUT
		ImageIcon image = new ImageIcon("SIMBOLOCLINICA.png");
		int scale = 1;

		int width = image.getIconWidth();
		int height = image.getIconHeight();
		BufferedImage buffer = new BufferedImage(scale * width, scale *
height, BufferedImage.TYPE_INT_ARGB);
		Graphics2D graphics = buffer.createGraphics();

		graphics.scale(scale,scale);
		image.paintIcon(null, graphics, 0, 0);
		graphics.dispose();
		JLabel label = new JLabel(new ImageIcon(buffer));
		panel2.add(label);
		add("East", panel2);

		// CONFIGURAR EL GRIDBAGLAYOUT

		constraints.insets = new Insets(0,0,0,40); //MARGEN A LA DERECHA
		constraints.anchor =  GridBagConstraints.LAST_LINE_END;
		constraints.fill =  GridBagConstraints.NONE;
		constraints.weighty=0.6;
		constraints.weightx=0.1;
		constraints.gridx = 0 ; // Establece gridx -->COLUMNA
		constraints.gridy = 0 ; // Establece gridy --> FILA
		constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
		constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
		panel1.add(usuario, constraints);
```

```
            constraints.insets = new Insets(0,0,0,0);
            constraints.anchor =  GridBagConstraints.LAST_LINE_START;
            constraints.fill =  GridBagConstraints.NONE;
            constraints.weighty=0.6;
            constraints.weightx=0.1;
            constraints.gridx = 1 ; // Establece gridx -->COLUMNA
            constraints.gridy = 0 ; // Establece gridy --> FILA
            constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
            constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
            panel1.add(textousuario, constraints);

            constraints.insets = new Insets(0,0,0,40); //MARGEN A LA DERECHA
            constraints.anchor =  GridBagConstraints.LINE_END;
            constraints.fill =  GridBagConstraints.NONE;
            constraints.weighty=0.2;
            constraints.weightx=0.1;
            constraints.gridx = 0 ; // Establece gridx -->COLUMNA
            constraints.gridy = 1 ; // Establece gridy --> FILA
            constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
            constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
            panel1.add(contrasenya, constraints);

            constraints.insets = new Insets(0,0,0,0);
            constraints.anchor =  GridBagConstraints.LINE_START;
            constraints.fill =  GridBagConstraints.NONE;
            constraints.weighty=0.2;
            constraints.weightx=0.1;
            constraints.gridx = 1 ; // Establece gridx -->COLUMNA
            constraints.gridy = 1 ; // Establece gridy --> FILA
            constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
            constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
            panel1.add(textocontrasenya, constraints);


            constraints.insets = new Insets(60,0,0,40); //MARGEN A LA
DERECHA
            constraints.anchor =  GridBagConstraints.LINE_END;
            constraints.weighty=0.0;
            constraints.weightx=0.0;
            constraints.fill =  GridBagConstraints.NONE;
            constraints.gridx = 0 ; // Establece gridx -->COLUMNA
            constraints.gridy = 2 ; // Establece gridy --> FILA
            constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
            constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
            btnAceptar.addActionListener(this);
            panel1.add(btnAceptar, constraints);


            constraints.insets = new Insets(60,0,0,0); //MARGEN A LA
IZQUIERDA
            constraints.anchor =  GridBagConstraints.LINE_START;
            constraints.fill =  GridBagConstraints.NONE;
            constraints.weighty=0.0;
            constraints.weightx=0.0;
            constraints.gridx = 1 ; // Establece gridx -->COLUMNA
            constraints.gridy = 2 ; // Establece gridy --> FILA
```

```
        constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
        constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
        btnLimpiar.addActionListener(this);
        panel1.add(btnLimpiar, constraints);

        constraints.insets = new Insets(0,0,0,0);
        constraints.anchor =  GridBagConstraints.PAGE_END;
        constraints.fill =  GridBagConstraints.NONE;
        constraints.weighty=1.0;
        constraints.weightx=1.0;
        constraints.gridx = 0 ; // Establece gridx -->COLUMNA
        constraints.gridy = 3 ; // Establece gridy --> FILA
        constraints.gridwidth = 2 ; // Establece gridwidth --> ANCHURA
        constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
        panel1.add(derechosAutor, constraints);

        add("Center", panel1);
        textousuario.requestFocus();


        addWindowListener(this);
        setVisible(true);

        dialogo0.setLayout(new FlowLayout());
        dialogo0.setSize(220,150);
        dialogo0.setResizable(false);
        dialogo0.setLocationRelativeTo(null);
        dialogo0.addWindowListener(this);
        dialogo0.add(mensaje0);
        btnAceptar0.addActionListener(this);
        dialogo0.add(btnAceptar0);


}


public void windowActivated(WindowEvent we){}

public void windowClosed(WindowEvent we){}

public void windowClosing(WindowEvent we)

{

       this.setVisible(false);

}

public void windowDeactivated(WindowEvent we){}

public void windowDeiconified(WindowEvent we){}

public void windowIconified(WindowEvent we){}

public void windowOpened(WindowEvent we){}

public static void main(String[] args)
{
```

```
            new Login();
      }

      public void actionPerformed(ActionEvent ae)
      {
            Object a;
            a=ae.getSource();
            char[] arrayC = textocontrasenya.getPassword();
            String pass = new String(arrayC);
            boolean llave=false;

            String Login = "SELECT * FROM usuarios WHERE nombreUsuario='"+
textousuario.getText() + "' AND contrasenyaUsuario='" + pass +"';";

            if(a.equals(btnLimpiar))
            {
                  llave=true;
                  textousuario.selectAll();
                  textousuario.setText("");
                  textocontrasenya.selectAll();
                  textocontrasenya.setText("");
                  textousuario.requestFocus();
            }

            if(a.equals(btnAceptar))
            {

                  if(textousuario.getText().equals("admin") && new
ConectaBBDD().comprobar_Usuario(Login)==true)
                  {
                        new Movimientos().guardar(textousuario.getText(),
"Login");
                        new GestionClinica1();
                        this.setVisible(false);
                        llave=true;
                  }else
                  {
                        if(textousuario.getText().equals("user") && new
ConectaBBDD().comprobar_Usuario(Login)==true)
                        {
                              new
Movimientos().guardar(textousuario.getText(), "Login");
                              new GestionClinica2();
                              this.setVisible(false);
                              llave=true;
                        }
                  }
            }

            if(llave==false)
            {
                  dialogo0.setVisible(true);
                  if(a.equals(btnAceptar0))
                  {
                        dialogo0.setVisible(false);
                  }
            }
```

```
	}


}
```

Clase Movimientos

```
package es.grupostudium.GestionClinica;

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Calendar;
import java.util.GregorianCalendar;

public class Movimientos
{

	public Movimientos()
	{
	}
	public void guardar(String usuario, String accion)
	{

		try
		{
			Calendar calendario = new GregorianCalendar();

			FileWriter fw = new FileWriter("Archivo.log", true);

			BufferedWriter bw = new BufferedWriter(fw);

			PrintWriter salida = new PrintWriter(bw);


			if(calendario.get(Calendar.DAY_OF_MONTH) < 10 &&
calendario.get(Calendar.MONTH) < 10)
			{
				salida.print("[" +
calendario.get(Calendar.HOUR_OF_DAY) +":"+ calendario.get(Calendar.MINUTE) +
":"+ calendario.get(Calendar.SECOND) + "]" );

	salida.print("["+"0"+calendario.get(Calendar.DAY_OF_MONTH)+"/"+"0"+((c
alendario.get(Calendar.MONTH))+1)+"/"+calendario.get(Calendar.YEAR)+"]");
				salida.print("["+usuario+"]");
				salida.print("["+accion+"]");
			}else
			{
				if(calendario.get(Calendar.DAY_OF_MONTH) < 10)
				{
					salida.print("[" +
calendario.get(Calendar.HOUR_OF_DAY) +":"+ calendario.get(Calendar.MINUTE) +
":"+ calendario.get(Calendar.SECOND) + "]" );
```

```
	salida.print("["+"0"+calendario.get(Calendar.DAY_OF_MONTH)+"/"+((calen
dario.get(Calendar.MONTH))+1)+"/"+calendario.get(Calendar.YEAR)+"]");
                                    salida.print("["+usuario+"]");
                                    salida.print("["+accion+"]");
                              }else
                              {
                                    if(calendario.get(Calendar.MONTH) < 10)
                                    {
                                          salida.print("[" +
calendario.get(Calendar.HOUR_OF_DAY) +":"+ calendario.get(Calendar.MINUTE) +
":"+ calendario.get(Calendar.SECOND) + "]" );

	salida.print("["+calendario.get(Calendar.DAY_OF_MONTH)+"/"+"0"+((calen
dario.get(Calendar.MONTH))+1)+"/"+calendario.get(Calendar.YEAR)+"]");
                                          salida.print("["+usuario+"]");
                                          salida.print("["+accion+"]");
                                    }
                              }
                        }


                        salida.println();
                        salida.close();
                        bw.close();
                        fw.close();
                        System.out.println("Archivo creado correctamente!");
                  }
                  catch(IOException i)
                  {
                        System.out.println("Se produjo un error de Archivo");
                  }
            }

            public String recoge_fecha_actual()
            {

                  String fecha = new String();

                  Calendar calendario = new GregorianCalendar();

                  if(calendario.get(Calendar.DAY_OF_MONTH) < 10 &&
calendario.get(Calendar.MONTH) < 10)
                  {
                        fecha = (calendario.get(Calendar.YEAR)+ "/" +
"0"+((calendario.get(Calendar.MONTH))+1) + "/" + "0"+
calendario.get(Calendar.DAY_OF_MONTH));

                  }else
                  {
                        if(calendario.get(Calendar.DAY_OF_MONTH) < 10)
                        {
                              fecha = (calendario.get(Calendar.YEAR)+ "/" +
((calendario.get(Calendar.MONTH))+1) + "/" + "0"+
calendario.get(Calendar.DAY_OF_MONTH));

                        }else
                        {
```

```
                        if(calendario.get(Calendar.MONTH) < 10)
                        {
                                fecha = (calendario.get(Calendar.YEAR)+ "/" +
"0" + ((calendario.get(Calendar.MONTH))+1) + "/" +
calendario.get(Calendar.DAY_OF_MONTH));

                        }
                }
        }

        return fecha;
    }

    public void recoge_sentencia(String sentencia)
    {
        try
        {
                Calendar calendario = new GregorianCalendar();

                FileWriter fw = new FileWriter("Archivo.log", true);

                BufferedWriter bw = new BufferedWriter(fw);

                PrintWriter salida = new PrintWriter(bw);


                if(calendario.get(Calendar.DAY_OF_MONTH) < 10 &&
calendario.get(Calendar.MONTH) < 10)
                {
                        salida.print("[" +
calendario.get(Calendar.HOUR_OF_DAY) +":"+ calendario.get(Calendar.MINUTE) +
":"+ calendario.get(Calendar.SECOND) + "]" );

    salida.print("["+"0"+calendario.get(Calendar.DAY_OF_MONTH)+"/"+"0"+((c
alendario.get(Calendar.MONTH))+1)+"/"+calendario.get(Calendar.YEAR)+"]");
                        salida.print("["+ sentencia +"]");
                }else
                {
                        if(calendario.get(Calendar.DAY_OF_MONTH) < 10)
                        {
                                salida.print("[" +
calendario.get(Calendar.HOUR_OF_DAY) +":"+ calendario.get(Calendar.MINUTE) +
":"+ calendario.get(Calendar.SECOND) + "]" );

    salida.print("["+"0"+calendario.get(Calendar.DAY_OF_MONTH)+"/"+((calen
dario.get(Calendar.MONTH))+1)+"/"+calendario.get(Calendar.YEAR)+"]");
                                salida.print("["+ sentencia +"]");
                        }else
                        {
                                if(calendario.get(Calendar.MONTH) < 10)
                                {
                                        salida.print("[" +
calendario.get(Calendar.HOUR_OF_DAY) +":"+ calendario.get(Calendar.MINUTE) +
":"+ calendario.get(Calendar.SECOND) + "]" );

    salida.print("["+calendario.get(Calendar.DAY_OF_MONTH)+"/"+"0"+((calen
dario.get(Calendar.MONTH))+1)+"/"+calendario.get(Calendar.YEAR)+"]");
                                        salida.print("["+ sentencia +"]");
```

```
                                }
                        }
                }

                salida.println();
                salida.close();
                bw.close();
                fw.close();

        }
        catch(IOException i)
        {
                System.out.println("Se produjo un error de Archivo Log");
        }

    }
}
```

Clase VentanaEliminarPaciente

```
package es.grupostudium.GestionClinica;

import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;

public class VentanaEliminarPaciente extends JFrame implements
WindowListener, ActionListener
{
    private static final long serialVersionUID = 1L;

    GridBagConstraints constraints = new GridBagConstraints();
    GridBagLayout gridbag = new GridBagLayout();

    JComboBox choice = new JComboBox();

    String str1 = "Elegir Paciente...";

    JPanel panel1 = new JPanel();
    JPanel panel2 = new JPanel();
    JPanel panel3 = new JPanel();
    JPanel panel4 = new JPanel();
    JPanel panel5 = new JPanel();
    JPanel panel6 = new JPanel();
```

```
	JButton btnB1 = new JButton("Eliminar");
	JButton btnB2 = new JButton("Cancelar");
	JButton btnB3 = new JButton("Aceptar"); //
	JButton btnB4 = new JButton("Cancelar");
	JButton btnB5 = new JButton("Aceptar");

	JDialog dialogo1 = new JDialog(this, "ELIMINAR PACIENTE", true);
	JLabel mensaje1 = new JLabel("¿Estás seguro que quieres eliminar este
paciente?");
	JDialog dialogo2 = new JDialog(this, "ELIMINAR PACIENTE", true);
	JLabel mensaje2 = new JLabel("Paciente eliminado con éxito.");
	ResultSet rs_id_pacientes = null;
	ResultSet rs_nombres_pacientes = null;
	ResultSet rs_apellidos_pacientes = null;

	public VentanaEliminarPaciente()
	{

		btnB1.addActionListener(this);
		btnB2.addActionListener(this);
		btnB3.addActionListener(this);
		btnB4.addActionListener(this);
		btnB5.addActionListener(this);


		setLayout(gridbag);
		setSize(600,400);
		setResizable(false);
		setLocationRelativeTo(null);
		setTitle("ELIMINAR PACIENTE");

		rs_id_pacientes = (ResultSet) new
ConectaBBDD().obtener_id_de_los_pacientes("SELECT idPaciente FROM
pacientes;");
		rs_nombres_pacientes = (ResultSet) new
ConectaBBDD().obtener_nombres_de_los_pacientes("SELECT nombrePaciente FROM
pacientes;");
		rs_apellidos_pacientes = (ResultSet) new
ConectaBBDD().obtener_apellidos_de_los_pacientes("SELECT apellidosPaciente
from pacientes;");

		choice.addItem(str1);


		try
		{
			while(rs_id_pacientes.next() &&
rs_nombres_pacientes.next() && rs_apellidos_pacientes.next())
			{

	choice.addItem(rs_id_pacientes.getString("idPaciente") +"-"+
rs_nombres_pacientes.getString("nombrePaciente") +" "+
rs_apellidos_pacientes.getString("apellidosPaciente"));
			}
		} catch (SQLException e)
		{
			e.printStackTrace();
```

```
}

try
{
	rs_id_pacientes.close();
	rs_nombres_pacientes.close();
	rs_apellidos_pacientes.close();

} catch (SQLException e)
{
	e.printStackTrace();
}



panel1.setLayout(new FlowLayout());
panel1.add(btnB1);
panel1.add(btnB2);


constraints.anchor =  GridBagConstraints.CENTER;
constraints.fill =  GridBagConstraints.NONE;
constraints.weighty=0.6;
constraints.weightx=0.6;
constraints.gridx = 0 ; // Establece gridx -->COLUMNA
constraints.gridy = 0 ; // Establece gridy --> FILA
constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
add(choice, constraints);

constraints.anchor =  GridBagConstraints.PAGE_START;
constraints.fill =  GridBagConstraints.NONE;
constraints.weighty=0.9;
constraints.weightx=0.9;
constraints.gridx = 0 ; // Establece gridx -->COLUMNA
constraints.gridy = 1 ; // Establece gridy --> FILA
constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
add(panel1, constraints);


addWindowListener(this);
setVisible(true);


dialogo1.setLayout(new BorderLayout());
panel2.setLayout(new FlowLayout());
panel3.setLayout(new FlowLayout());
dialogo1.setSize(600,400);
dialogo1.setResizable(false);
dialogo1.setLocationRelativeTo(null);
dialogo1.addWindowListener(this);
panel2.add(mensaje1);
dialogo1.add("North",panel2);
panel3.add(btnB3);
panel3.add(btnB4);
dialogo1.add("Center", panel3);
```

```
		dialogo2.setLayout(new BorderLayout());
		panel4.setLayout(new FlowLayout());
		panel5.setLayout(new FlowLayout());
		dialogo2.setSize(600,400);
		dialogo2.setResizable(false);
		dialogo2.setLocationRelativeTo(null);
		dialogo2.addWindowListener(this);
		panel4.add(mensaje2);
		dialogo2.add("North",panel4);
		panel5.add(btnB5);
		dialogo2.add("Center", panel5);

	}

	public static void main(String[] args)
	{
		new VentanaEliminarPaciente();
	}

	public void windowActivated(WindowEvent we){}

	public void windowClosed(WindowEvent we){}

	public void windowClosing(WindowEvent we)

	{

		this.setVisible(false);

	}

	public void windowDeactivated(WindowEvent we){}

	public void windowDeiconified(WindowEvent we){}

	public void windowIconified(WindowEvent we){}

	public void windowOpened(WindowEvent we){}



	public void actionPerformed(ActionEvent ae)
	{
		Object a;
		a = ae.getSource();
		String nombre_completo_paciente = (String)
choice.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

		if(a.equals(btnB1) && !nombre_completo_paciente.equals("Elegir
Paciente..."))
		{
			dialogo1.setVisible(true);
		}
		if(a.equals(btnB2))
		{
			this.setVisible(false);
		}
```

```
            if(a.equals(btnB3))
            {

                  String[] seleciona_id = nombre_completo_paciente.split("-
");

                  String sentencia_eliminar_paciente = "DELETE FROM
pacientes WHERE idPaciente='"+ seleciona_id[0] +"';";

                  new
ConectaBBDD().eliminar_paciente(sentencia_eliminar_paciente);

                  new
Movimientos().recoge_sentencia(sentencia_eliminar_paciente);


                  dialogo1.setVisible(false);
                  dialogo2.setVisible(true);



                  choice.removeAllItems();

                  rs_id_pacientes = (ResultSet) new
ConectaBBDD().obtener_id_de_los_pacientes("SELECT idPaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
                  rs_nombres_pacientes = (ResultSet) new
ConectaBBDD().obtener_nombres_de_los_pacientes("SELECT nombrePaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
                  rs_apellidos_pacientes = (ResultSet) new
ConectaBBDD().obtener_apellidos_de_los_pacientes("SELECT apellidosPaciente
from pacientes;");//DEVUELVE RS CON LOS APELLIDOS DE LOS PACIENTES

                  choice.addItem(str1);


                  try //VOLVEMOS A USAR UN WHILE PARA RELLENAR EL JCOMBOX
CON LOS RESULTADOS DEL RESULSET
                  {
                        while(rs_id_pacientes.next() &&
rs_nombres_pacientes.next() && rs_apellidos_pacientes.next())
                        {

      choice.addItem(rs_id_pacientes.getString("idPaciente") +"-"+
rs_nombres_pacientes.getString("nombrePaciente") +" "+
rs_apellidos_pacientes.getString("apellidosPaciente"));//AÑADE NOMBRE Y
APELLIDOS DE LOS RS USADOS ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX
                        }
                  } catch (SQLException e)
                  {
                        e.printStackTrace();
                  }

                  try //VOLVEMOS A CERRAR LOS RESULTSET USADOS
                  {
                        rs_id_pacientes.close();
                        rs_nombres_pacientes.close();
                        rs_apellidos_pacientes.close();
```

```
                } catch (SQLException e)
                {
                        e.printStackTrace();
                }

            }
            if(a.equals(btnB4))
            {
                dialogo1.setVisible(false);
            }
            if(a.equals(btnB5))
            {
                dialogo2.setVisible(false);

            }

        }
}
```

Clase VentanaEliminarPodologo

```
package es.grupostudium.GestionClinica;

import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;

public class VentanaEliminarPodologo extends JFrame implements
WindowListener, ActionListener
{
        private static final long serialVersionUID = 1L;

        GridBagConstraints constraints = new GridBagConstraints(); //Crear la
distribución
        GridBagLayout gridbag = new GridBagLayout(); //Crear las restricciones

        JComboBox choice = new JComboBox();

        String str1 = "Elegir Podólogo...";

        JPanel panel1 = new JPanel();
        JPanel panel2 = new JPanel();
```

```
	JPanel panel3 = new JPanel();
	JPanel panel4 = new JPanel();
	JPanel panel5 = new JPanel();
	JPanel panel6 = new JPanel();

	JButton btnB1 = new JButton("Eliminar");
	JButton btnB2 = new JButton("Cancelar");
	JButton btnB3 = new JButton("Aceptar"); //
	JButton btnB4 = new JButton("Cancelar");
	JButton btnB5 = new JButton("Aceptar");

	JDialog dialogo1 = new JDialog(this, "ELIMINAR PODÓLOGO", true);
	JLabel mensaje1 = new JLabel("<html><body>¿Estás seguro que quieres
eliminar este Podólogo?<br>ADVERTENCIA: Se eliminarán todos los tratamientos
asociados a este Profesional.</body></html>");
	JDialog dialogo2 = new JDialog(this, "ELIMINAR PODÓLOGO", true);
	JLabel mensaje2 = new JLabel("Podólogo eliminado con éxito.");

	ResultSet rs_podologos = null;
	ResultSet rs_personal = null;

	public VentanaEliminarPodologo()
	{

		btnB1.addActionListener(this);
		btnB2.addActionListener(this);
		btnB3.addActionListener(this);
		btnB4.addActionListener(this);
		btnB5.addActionListener(this);


		setLayout(gridbag);
		setSize(600,400);
		setResizable(false);
		setLocationRelativeTo(null);
		setTitle("ELIMINAR PODÓLOGO");


		choice.addItem(str1);

		try
		{
			rs_podologos = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_de_los_podologos("SELECT idPodologo,
numeroColegiadoPodologo, especialidadPodologo, idPersonalFK FROM
podologos;");

			if(rs_podologos.next())
			{
				rs_podologos.beforeFirst(); //LO DEVOLVEMOS A SU
ESTADO INICIAL DESPUES DEL "IF"

				while(rs_podologos.next())
				{
					rs_personal = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_del_personal("SELECT idPersonal,
nombrePersonal, apellidosPersonal, fechaNacimientoPersonal, direccionPersonal
```

```
FROM personal WHERE idPersonal='"+ rs_podologos.getString("idPersonalFK") +
"';");

                                    rs_personal.next();


      choice.addItem(rs_personal.getString("idPersonal") +"-"+
rs_personal.getString("nombrePersonal") +" "+
rs_personal.getString("apellidosPersonal"));//AÑADE NOMBRE Y APELLIDOS DE LOS
RS USADOS ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX

                              }

                              rs_personal.close(); //CERRAMOS LOS RESULTSET
USADOS
                              rs_podologos.close();

                        }

            } catch (SQLException e)
            {
                  System.out.println("Error al cargar el JCOMBOBOX con las
consultas de la Base de Datos");
                  e.printStackTrace();
            }


            panel1.setLayout(new FlowLayout());
            panel1.add(btnB1);
            panel1.add(btnB2);


            constraints.anchor =  GridBagConstraints.CENTER;
            constraints.fill =  GridBagConstraints.NONE;
            constraints.weighty=0.6;
            constraints.weightx=0.6;
            constraints.gridx = 0 ; // Establece gridx -->COLUMNA
            constraints.gridy = 0 ; // Establece gridy --> FILA
            constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
            constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
            add(choice, constraints);

            constraints.anchor =  GridBagConstraints.PAGE_START;
            constraints.fill =  GridBagConstraints.NONE;
            constraints.weighty=0.9;
            constraints.weightx=0.9;
            constraints.gridx = 0 ; // Establece gridx -->COLUMNA
            constraints.gridy = 1 ; // Establece gridy --> FILA
            constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
            constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
            add(panel1, constraints);


            addWindowListener(this);
            setVisible(true);


            dialogo1.setLayout(new BorderLayout());
```

```
        panel2.setLayout(new FlowLayout());
        panel3.setLayout(new FlowLayout());
        dialogo1.setSize(600,400);
        dialogo1.setResizable(false);
        dialogo1.setLocationRelativeTo(null);
        dialogo1.addWindowListener(this);
        panel2.add(mensaje1);
        dialogo1.add("North",panel2);
        panel3.add(btnB3);
        panel3.add(btnB4);
        dialogo1.add("Center", panel3);

        dialogo2.setLayout(new BorderLayout());
        panel4.setLayout(new FlowLayout());
        panel5.setLayout(new FlowLayout());
        dialogo2.setSize(600,400);
        dialogo2.setResizable(false);
        dialogo2.setLocationRelativeTo(null);
        dialogo2.addWindowListener(this);
        panel4.add(mensaje2);
        dialogo2.add("North",panel4);
        panel5.add(btnB5);
        dialogo2.add("Center", panel5);

}

public static void main(String[] args)
{
        new VentanaEliminarPaciente();
}

public void windowActivated(WindowEvent we){}

public void windowClosed(WindowEvent we){}

public void windowClosing(WindowEvent we)

{

        this.setVisible(false);

}

public void windowDeactivated(WindowEvent we){}

public void windowDeiconified(WindowEvent we){}

public void windowIconified(WindowEvent we){}

public void windowOpened(WindowEvent we){}



public void actionPerformed(ActionEvent ae)
{
        Object a;
        a = ae.getSource();
```

```
            if(a.equals(btnB1))
            {
                  dialogo1.setVisible(true);
            }
            if(a.equals(btnB2))
            {
                  this.setVisible(false);
            }
            if(a.equals(btnB3))
            {

                  String nombre_completo_podologo = (String)
choice.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

                  String[] seleciona_id = nombre_completo_podologo.split("-
");  //METODO PARA SEPARAR EL ID DEL RESTO DEL NOMBRE

                  String sentencia_eliminar_podologo = "DELETE FROM
podologos WHERE idPodologo='"+ seleciona_id[0] +"';"; //SENTENCIA PARA
ELIMINAR PODOLOGO EN LA BD

                  String sentencia_eliminar_personal = "DELETE FROM personal
WHERE idPersonal='"+ seleciona_id[0] +"';"; //SENTENCIA PARA ELIMINAR
PACIENTE EN LA BD


                  if(new
ConectaBBDD().eliminar_podologo(sentencia_eliminar_podologo)==true )
                  {

                        //SENTENCIA QUE ELIMINA TODOS LOS TTOS DEL PODÓLOGO
QUE ESTÁS BORRANDO

                        String sentencia_eliminar_tratamiento = "DELETE
FROM tratamientopersonal WHERE idPersonalFK='"+ seleciona_id[0] + "';";
//SENTENCIA PARA ELIMINAR TRATAMIENTO EN LA BD

                        new
ConectaBBDD().eliminar_tratamiento(sentencia_eliminar_tratamiento);


                        if (new
ConectaBBDD().eliminar_personal(sentencia_eliminar_personal)==true)
                        {

                              new
Movimientos().recoge_sentencia(sentencia_eliminar_podologo + " " +
sentencia_eliminar_tratamiento + " " + sentencia_eliminar_personal);
//ESCRIBE EN EL LOG LA SENTENCIA, ELIMINA TANTO EL PODÓLOGO LOS TTOS
ASOCIADOS COMO EL PERSONAL CORRESPONDIENTE
                              dialogo1.setVisible(false);
                              dialogo2.setVisible(true);
                        }

                  }
                  else
                  {
```

```
                        dialogo1.setVisible(false);
                        mensaje2.setText("ERROR AL ELIMINAR PODÓLOGO");
                        dialogo2.setVisible(true);
                }



                //LIMPIAR EL JCOMBOBOX Y ACTUALIZAR LOS DATOS

                choice.removeAllItems();

                choice.addItem(str1);

                try
                {
                        rs_podologos = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_de_los_podologos("SELECT idPodologo,
numeroColegiadoPodologo, especialidadPodologo, idPersonalFK FROM
podologos;");

                        if(rs_podologos.next())
                        {

                                rs_podologos.beforeFirst(); //LO DEVOLVEMOS A
SU ESTADO INICIAL DESPUES DEL "IF"

                                while(rs_podologos.next())
                                {
                                        rs_personal = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_del_personal("SELECT idPersonal,
nombrePersonal, apellidosPersonal, fechaNacimientoPersonal, direccionPersonal
FROM personal WHERE idPersonal='"+ rs_podologos.getString("idPersonalFK") +
"';");

                                        rs_personal.next();


      choice.addItem(rs_personal.getString("idPersonal") +"-"+
rs_personal.getString("nombrePersonal") +" "+
rs_personal.getString("apellidosPersonal"));//AÑADE NOMBRE Y APELLIDOS DE LOS
RS USADOS ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX

                                }

                                rs_personal.close(); //CERRAMOS LOS RESULTSET
USADOS
                                rs_podologos.close();

                        }
                } catch (SQLException e)
                {
                        System.out.println("Error al cargar el JCOMBOBOX
con las consultas de la Base de Datos");
                        e.printStackTrace();
                }
        }

        if(a.equals(btnB4))
```

```
		{
			dialogo1.setVisible(false);
		}
		if(a.equals(btnB5))
		{
			dialogo2.setVisible(false);

		}

	}
}
```

Clase VentanaEliminarTratamiento

```
package es.grupostudium.GestionClinica;

import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;

public class VentanaEliminarTratamiento extends JFrame implements
WindowListener, ActionListener
{

	private static final long serialVersionUID = 1L;

	GridBagConstraints constraints = new GridBagConstraints(); //Crear la
distribución
	GridBagLayout gridbag = new GridBagLayout(); //Crear las restricciones

	JComboBox choice1 = new JComboBox();
	JComboBox choice2 = new JComboBox();

	String str1 = "Elegir Paciente...";
	String str2 = "Elegir Tratamiento...";

	JPanel panel1 = new JPanel();
	JPanel panel2 = new JPanel();
	JPanel panel3 = new JPanel();
	JPanel panel4 = new JPanel();
	JPanel panel5 = new JPanel();
	JPanel panel6 = new JPanel();
	JPanel panel7 = new JPanel();
	JPanel panel8 = new JPanel();
```

```
	JButton btnB1 = new JButton("Seleccionar");
	JButton btnB2 = new JButton("Cancelar");
	JButton btnB3 = new JButton("Eliminar");
	JButton btnB4 = new JButton("Cancelar");
	JButton btnB5 = new JButton("Aceptar");
	JButton btnB6 = new JButton("Cancelar");
	JButton btnB7 = new JButton("Aceptar");

	JDialog dialogo1 = new JDialog(this, "ELIMINAR TRATAMIENTO", true);
	JDialog dialogo2 = new JDialog(this, "ELIMINAR TRATAMIENTO", true);
	JDialog dialogo3 = new JDialog(this, "ELIMINAR TRATAMIENTO", true);

	JLabel mensaje1 = new JLabel("¿Estás seguro que quieres eliminar este
Tratamiento?");
	JLabel mensaje2 = new JLabel("Tratamiento eliminado con éxito.");

	ResultSet rs_podologos = null;
	ResultSet rs_tratamientos = null;

	ResultSet rs_id_pacientes = null;
	ResultSet rs_nombres_pacientes = null;
	ResultSet rs_apellidos_pacientes = null;



	public VentanaEliminarTratamiento()
	{

		btnB1.addActionListener(this);
		btnB2.addActionListener(this);
		btnB3.addActionListener(this);
		btnB4.addActionListener(this);
		btnB5.addActionListener(this);
		btnB6.addActionListener(this);
		btnB7.addActionListener(this);


		setLayout(gridbag);
		setSize(600,400);
		setResizable(false);
		setLocationRelativeTo(null);
		setTitle("ELIMINAR TRATAMIENTO");


		rs_podologos = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_de_los_podologos("SELECT idPodologo,
numeroColegiadoPodologo, especialidadPodologo, idPersonalFK FROM
podologos;");

		choice1.addItem(str1);
		choice2.addItem(str2);

		rs_id_pacientes = (ResultSet) new
ConectaBBDD().obtener_id_de_los_pacientes("SELECT idPaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
```

```
            rs_nombres_pacientes = (ResultSet) new
ConectaBBDD().obtener_nombres_de_los_pacientes("SELECT nombrePaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
            rs_apellidos_pacientes = (ResultSet) new
ConectaBBDD().obtener_apellidos_de_los_pacientes("SELECT apellidosPaciente
from pacientes;");//DEVUELVE RS CON LOS APELLIDOS DE LOS PACIENTES


            try //USAMOS UN WHILE PARA RELLENAR EL JCOMBOX CON LOS
RESULTADOS DEL RESULSET
            {
                  while(rs_id_pacientes.next() &&
rs_nombres_pacientes.next() && rs_apellidos_pacientes.next())
                  {

      choice1.addItem(rs_id_pacientes.getString("idPaciente") +"-"+
rs_nombres_pacientes.getString("nombrePaciente") +" "+
rs_apellidos_pacientes.getString("apellidosPaciente"));//AÑADE NOMBRE Y
APELLIDOS DE LOS RS USADOS ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX
                  }
            } catch (SQLException e)
            {
                  e.printStackTrace();
            }

            try //CERRAMOS LOS RESULTSET USADOS
            {
                  rs_id_pacientes.close();
                  rs_nombres_pacientes.close();
                  rs_apellidos_pacientes.close();

            } catch (SQLException e)
            {
                  e.printStackTrace();
            }


            panel1.setLayout(new FlowLayout());
            panel1.add(btnB1);
            panel1.add(btnB2);


            constraints.anchor =  GridBagConstraints.CENTER;
            constraints.fill =  GridBagConstraints.NONE;
            constraints.weighty=0.6;
            constraints.weightx=0.6;
            constraints.gridx = 0 ; // Establece gridx -->COLUMNA
            constraints.gridy = 0 ; // Establece gridy --> FILA
            constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
            constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
            add(choice1, constraints);

            constraints.anchor =  GridBagConstraints.PAGE_START;
            constraints.fill =  GridBagConstraints.NONE;
            constraints.weighty=0.9;
            constraints.weightx=0.9;
            constraints.gridx = 0 ; // Establece gridx -->COLUMNA
            constraints.gridy = 1 ; // Establece gridy --> FILA
```

```
        constraints.gridwidth = 1 ; // Establece gridwidth --> ANCHURA
        constraints.gridheight = 1 ; // Establece gridheight --> ALTURA
        add(panel1, constraints);


        addWindowListener(this);
        setVisible(true);


        dialogo1.setLayout(new BorderLayout());
        panel2.setLayout(new FlowLayout());
        panel3.setLayout(new FlowLayout());
        dialogo1.setSize(600,400);
        dialogo1.setResizable(false);
        dialogo1.setLocationRelativeTo(null);
        dialogo1.addWindowListener(this);
        panel2.add(choice2);
        dialogo1.add("North",panel2);
        panel3.add(btnB3);
        panel3.add(btnB4);
        dialogo1.add("Center", panel3);



        dialogo2.setLayout(new BorderLayout());
        panel4.setLayout(new FlowLayout());
        panel5.setLayout(new FlowLayout());
        dialogo2.setSize(600,400);
        dialogo2.setResizable(false);
        dialogo2.setLocationRelativeTo(null);
        dialogo2.addWindowListener(this);
        panel4.add(mensaje1);
        dialogo2.add("North",panel4);
        panel5.add(btnB5);
        panel5.add(btnB6);
        dialogo2.add("Center", panel5);

        dialogo3.setLayout(new BorderLayout());
        panel7.setLayout(new FlowLayout());
        panel8.setLayout(new FlowLayout());
        dialogo3.setSize(600,400);
        dialogo3.setResizable(false);
        dialogo3.setLocationRelativeTo(null);
        dialogo3.addWindowListener(this);
        panel7.add(mensaje2);
        dialogo3.add("North",panel7);
        panel8.add(btnB7);
        dialogo3.add("Center", panel8);


}


public void windowActivated(WindowEvent we){}

public void windowClosed(WindowEvent we){}
```

```
	public void windowClosing(WindowEvent we)

	{

		this.setVisible(false);

	}

	public void windowDeactivated(WindowEvent we){}

	public void windowDeiconified(WindowEvent we){}

	public void windowIconified(WindowEvent we){}

	public void windowOpened(WindowEvent we){}



	public void actionPerformed(ActionEvent ae)
	{
		Object a;
		a = ae.getSource();

		if(a.equals(btnB1))
		{
			String nombre_completo_paciente = (String)
choice1.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

			if(!nombre_completo_paciente.equals("Elegir Paciente..."))
			{

				String[] seleciona_id =
nombre_completo_paciente.split("-");  //METODO PARA SEPARAR EL ID DEL RESTO
DEL NOMBRE

				String sentencia_obtener_tratamientos = "SELECT *
FROM tratamientopersonal WHERE idPacienteFK='"+ seleciona_id[0] + "';";

				rs_tratamientos = (ResultSet)new
ConectaBBDD().obtener_tratamientos(sentencia_obtener_tratamientos);

				try //USAMOS UN WHILE PARA RELLENAR EL JCOMBOX CON
LOS RESULTADOS DEL RESULSET
				{
					while(rs_tratamientos.next())
					{

	choice2.addItem(rs_tratamientos.getString(1) + "-IMPORTE: " +
rs_tratamientos.getString(4) +"€  FECHA: "+
rs_tratamientos.getString(5));//AÑADE NOMBRE Y APELLIDOS DE LOS RS USADOS
ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX
					}
				} catch (SQLException e)
				{
					e.printStackTrace();
				}
				dialogo1.setVisible(true);
```

```
			}
		}

		if(a.equals(btnB2))
		{
			this.setVisible(false);
		}


		if(a.equals(btnB3))
		{
			String nombre_tratamiento = (String)
choice2.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

			if(!nombre_tratamiento.equals("Elegir Tratamiento..."))
			{
			dialogo2.setVisible(true);
			}
		}

		if(a.equals(btnB4))
		{
			this.setVisible(false);
		}

		if(a.equals(btnB5))
		{

			String nombre_tratamiento = (String)
choice2.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

			if(!nombre_tratamiento.equals("Elegir Tratamiento..."))
			{

			String[] seleciona_id =  nombre_tratamiento.split("-");
//METODO PARA SEPARAR EL ID DEL RESTO DEL NOMBRE

			String sentencia_eliminar_tratamiento = "DELETE FROM
tratamientopersonal WHERE idTratamiento='"+ seleciona_id[0] +"'"; //SENTENCIA
PARA ELIMINAR TRATAMIENTO EN LA BD

			new
ConectaBBDD().eliminar_tratamiento(sentencia_eliminar_tratamiento);

			new
Movimientos().recoge_sentencia(sentencia_eliminar_tratamiento);

			dialogo3.setVisible(true);

			}
		}
		if(a.equals(btnB6))
		{

		this.setVisible(false);
```

```
            }
            if(a.equals(btnB7))
            {

            this.setVisible(false);
            }


      }


}
```

Clase VentanaListaPacientes

```
import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.Graphics;
import java.awt.Image;
import java.awt.Panel;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.FileOutputStream;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JTable;
import javax.swing.SwingConstants;
import javax.swing.table.DefaultTableCellRenderer;
import javax.swing.table.DefaultTableModel;
import com.itextpdf.text.BaseColor;
import com.itextpdf.text.Document;
import com.itextpdf.text.Font;
import com.itextpdf.text.FontFactory;
import com.itextpdf.text.Paragraph;
import com.itextpdf.text.pdf.PdfPTable;
import com.itextpdf.text.pdf.PdfWriter;


public class VentanaListaPacientes extends JFrame implements WindowListener,
ActionListener
{
      private static final long serialVersionUID = 1L;

      JButton imprime = new JButton("Exportar a PDF");
      Panel panel1 = new Panel();
      DefaultTableModel modelo = new DefaultTableModel();
      JTable tabla = new JTable(modelo);
      ResultSet rs_lista_pacientes = null;

      VentanaListaPacientes()
      {
```

```
            setLayout(new BorderLayout());
            setTitle("LISTA DE PACIENTES");
            setSize(800,600);
            setLocationRelativeTo(null);
            setResizable(true);

            //AÑADIMOS LAS COLUMNAS AL JTABLE
            modelo.addColumn("ID");
            modelo.addColumn("Nombre");
            modelo.addColumn("Apellidos");
            modelo.addColumn("Fecha de Nacimiento");
            modelo.addColumn("Dirección");
            modelo.addColumn("DNI");

            //HACEMOS QUE EL TEXTO DENTRO DE LA TABLA SALGA CENTRADO
            DefaultTableCellRenderer tcr = new DefaultTableCellRenderer();
            tcr.setHorizontalAlignment(SwingConstants.CENTER); //ALINEA LOS
STRINGS EN EL CENTRO DE LA CELDA
            tabla.getColumnModel().getColumn(0).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(1).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(2).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(3).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(4).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(5).setCellRenderer(tcr);

            rs_lista_pacientes = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_de_los_pacientes("SELECT idPaciente,
nombrePaciente, apellidosPaciente, fechaNacimientoPaciente,
direccionPaciente, dniPaciente FROM pacientes;");

            new Movimientos().recoge_sentencia("SELECT idPaciente,
nombrePaciente, apellidosPaciente, fechaNacimientoPaciente,
direccionPaciente, dniPaciente FROM pacientes;"); //ESCRIBE EN EL LOG

            Object [] encabezado = {"ID", "NOMBRE", "APELLIDOS", "FECHA
NACIMIENTO", "DIRECCIÓN", "DNI"};
            modelo.addRow(encabezado);

            try
            {
                  // BUCLE PARA EL RESULSET CON LOS DATOS DE LOS PACIENTES

                  while (rs_lista_pacientes.next())
                  {
                        // SE CREA UN ARRAY DE TIPO OBJETO QUE SERÁN LAS
FILAS DE LA TABLA

                        Object [] fila = new Object[6];         // HAY 6
COLUMNAS

                        // SE RELLENA CADA POSICIÓN DEL ARRAY CON UNA DE
LAS COLUMNAS DE LA TABLA EN LA BASE DE DATOS

                        for (int i=0;i<6;i++)
                        {

                              fila[i] = rs_lista_pacientes.getObject(i+1);
      // EL PRIMER INDICE EN EL RS ES EL 1, NO EL 0, POR ESO SE SUMA 1.
```

```
                        }

                        modelo.addRow(fila);
// SE AÑADE AL MODELO LA FILA COMPLETA

                }
            }catch(SQLException e)
            {
                System.out.println("Error en la sentencia SQL" +
e.getMessage());
            }


            imprime.addActionListener(this);
            panel1.setLayout(new FlowLayout());
            panel1.add(imprime);
            add(tabla, "North");
            add(panel1, "Center");

            addWindowListener(this);
            setVisible(true);


      }

      public void windowActivated(WindowEvent we){}

      public void windowClosed(WindowEvent we){}

      public void windowClosing(WindowEvent we)

      {

            this.setVisible(false);

      }

      public void windowDeactivated(WindowEvent we){}

      public void windowDeiconified(WindowEvent we){}

      public void windowIconified(WindowEvent we){}

      public void windowOpened(WindowEvent we){}

      public void actionPerformed(ActionEvent ae){


            Document documento = new Document();
            String [] encabezado = {"ID", "NOMBRE", "APELLIDOS", "FECHA
NACIMIENTO", "DIRECCIÓN", "DNI"};

            try
            {
                rs_lista_pacientes.beforeFirst();
```

```
                    // Se crea el OutputStream para el fichero donde queremos
dejar el pdf.
                    FileOutputStream ficheroPdf = new
FileOutputStream("Listado de Pacientes.pdf");

                    // Se asocia el documento al OutputStream y se indica que
el espaciado entre
                    // lineas sera de 20. Esta llamada debe hacerse antes de
abrir el documento

      PdfWriter.getInstance(documento,ficheroPdf).setInitialLeading(20);

                    // Se abre el documento.
                    documento.open();
                    documento.add(new Paragraph("  LISTADO DE PACIENTES   -
CLÍNICA SALLE",FontFactory.getFont("arial", // fuente
                                  22, // tamaño
                                  Font.ITALIC, // estilo
                                  BaseColor.BLACK)));// color
                    documento.add(new Paragraph("  "));

                    /*
                     Image foto = Image.getInstance("SIMBOLOCLINICA.png");
                    foto.scaleToFit(100, 100);
                    foto.setAlignment(Chunk.ALIGN_MIDDLE);
                    documento.add(foto);
                    PdfPTable tabla = new PdfPTable(3);
                     */

                    PdfPTable tabla = new PdfPTable(6);

                    for( int i = 0 ; i < 6; i++)
                    {
                          tabla.addCell(encabezado[i]);
                    }


                    while(rs_lista_pacientes.next())
                    {
                          for (int i = 0; i < 6; i++)
                          {

      tabla.addCell(rs_lista_pacientes.getString(i+1));
                          }
                          ;
                    }

                    documento.add(tabla);
                    documento.close();
              }
              catch ( Exception e )
              {
                    e.printStackTrace();
              }

      }
}
```

Clase VentanaListaPodologos

```
package es.grupostudium.GestionClinica;

import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.Panel;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.FileOutputStream;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JTable;
import javax.swing.SwingConstants;
import javax.swing.table.DefaultTableCellRenderer;
import javax.swing.table.DefaultTableModel;
import com.itextpdf.text.BaseColor;
import com.itextpdf.text.Document;
import com.itextpdf.text.Font;
import com.itextpdf.text.FontFactory;
import com.itextpdf.text.Paragraph;
import com.itextpdf.text.pdf.PdfPTable;
import com.itextpdf.text.pdf.PdfWriter;

public class VentanaListaPodologos extends JFrame implements WindowListener,
ActionListener
{
	private static final long serialVersionUID = 1L;

	DefaultTableModel modelo = new DefaultTableModel();
	JTable tabla = new JTable(modelo);
	ResultSet rs_personal = null;
	ResultSet rs_podologos = null;
	Panel panel1 = new Panel();
	JButton imprime = new JButton("Exportar a PDF");
	ResultSet rs_id_pacientes = null;
	ResultSet rs_nombres_pacientes = null;
	ResultSet rs_apellidos_pacientes = null;
	JComboBox choice = new JComboBox();

	VentanaListaPodologos()
	{
		setLayout(new BorderLayout());
		setTitle("LISTA DE PODÓLOGOS");
		setSize(800,600);
		setLocationRelativeTo(null);
		setResizable(true);

		//AÑADIMOS LAS COLUMNAS AL JTABLE
		modelo.addColumn("ID PERSONAL");
		modelo.addColumn("NOMBRE");
		modelo.addColumn("APELLIDOS");
		modelo.addColumn("FECHA DE NACIMIENTO");
```

```
            modelo.addColumn("DIRECCIÓN");
            modelo.addColumn("NÚMERO COLEGIADO");
            modelo.addColumn("ESPECIALIDAD");

            //HACEMOS QUE EL TEXTO DENTRO DE LA TABLA SALGA CENTRADO
            DefaultTableCellRenderer tcr = new DefaultTableCellRenderer();
            tcr.setHorizontalAlignment(SwingConstants.CENTER); //ALINEA LOS
STRINGS EN EL CENTRO DE LA CELDA
            tabla.getColumnModel().getColumn(0).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(1).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(2).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(3).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(4).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(5).setCellRenderer(tcr);
            tabla.getColumnModel().getColumn(6).setCellRenderer(tcr);

            rs_personal = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_del_personal("SELECT idPersonal,
nombrePersonal, apellidosPersonal, fechaNacimientoPersonal, direccionPersonal
FROM personal;");
            rs_podologos = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_de_los_podologos("SELECT
numeroColegiadoPodologo, especialidadPodologo FROM podologos;");

            new Movimientos().recoge_sentencia("SELECT idPersonal,
nombrePersonal, apellidosPersonal, fechaNacimientoPersonal, direccionPersonal
FROM personal; " + "SELECT numeroColegiadoPodologo, especialidadPodologo FROM
podologos;"); //ESCRIBE EN EL LOG

            Object [] encabezado = {"ID PERSONAL", "NOMBRE", "APELLIDOS",
"FECHA NACIMIENTO", "DIRECCIÓN", "NÚMERO COLEGIADO", "ESPECIALIDAD"};
            modelo.addRow(encabezado);

            rs_id_pacientes = (ResultSet) new
ConectaBBDD().obtener_id_de_los_pacientes("SELECT idPaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
            rs_nombres_pacientes = (ResultSet) new
ConectaBBDD().obtener_nombres_de_los_pacientes("SELECT nombrePaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
            rs_apellidos_pacientes = (ResultSet) new
ConectaBBDD().obtener_apellidos_de_los_pacientes("SELECT apellidosPaciente
from pacientes;");//DEVUELVE RS CON LOS APELLIDOS DE LOS PACIENTES


            try //USAMOS UN WHILE PARA RELLENAR EL JCOMBOX CON LOS
RESULTADOS DEL RESULSET
            {
                  while(rs_id_pacientes.next() &&
rs_nombres_pacientes.next() && rs_apellidos_pacientes.next())
                  {

      choice.addItem(rs_id_pacientes.getString("idPaciente") +"-"+
rs_nombres_pacientes.getString("nombrePaciente") +" "+
rs_apellidos_pacientes.getString("apellidosPaciente"));//AÑADE NOMBRE Y
APELLIDOS DE LOS RS USADOS ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX
                  }
            } catch (SQLException e)
            {
```

```
                e.printStackTrace();
            }

            try //CERRAMOS LOS RESULTSET USADOS
            {
                rs_id_pacientes.close();
                rs_nombres_pacientes.close();
                rs_apellidos_pacientes.close();

            } catch (SQLException e)
            {
                e.printStackTrace();
            }

            try
            {
                // BUCLE PARA EL RESULSET CON LOS DATOS DE LOS PACIENTES



                while (rs_personal.next() && rs_podologos.next())
                {
                    // SE CREA UN ARRAY DE TIPO OBJETO QUE SERÁN LAS
FILAS DE LA TABLA

                    Object [] fila = new Object[7];         // HAY 7
COLUMNAS


                    // SE RELLENA CADA POSICIÓN DEL ARRAY CON UNA DE
LAS COLUMNAS DE LA TABLA EN LA BASE DE DATOS


                    for (int i = 0; i < 7 ; i++)
                    {

                                if( i < 5)
                                {
                                    fila[i] =
rs_personal.getObject(i+1);      // EL PRIMER INDICE EN EL RS ES EL 1, NO EL
0, POR ESO SE SUMA 1.
                                }
                                else
                                {
                                    fila[i] =
rs_podologos.getObject(i-4);          //SE LE RESTA 4 PARA OBTENER EL VALOR
1 Y 2 DEL RESULSET
                                }
                    }


                    modelo.addRow(fila);                    // SE AÑADE
AL MODELO LA FILA COMPLETA

                }
            }catch(SQLException e)
            {
```

```
                    System.out.println("Error en la sentencia SQL" +
e.getMessage());
                    e.printStackTrace();
             }


             imprime.addActionListener(this);
             panel1.setLayout(new FlowLayout());
             panel1.add(imprime);
             add(tabla, "North");
             add(panel1, "Center");


             addWindowListener(this);
             setVisible(true);


      }

      public void windowActivated(WindowEvent we){}

      public void windowClosed(WindowEvent we){}

      public void windowClosing(WindowEvent we)

      {

             this.setVisible(false);

      }

      public void windowDeactivated(WindowEvent we){}

      public void windowDeiconified(WindowEvent we){}

      public void windowIconified(WindowEvent we){}

      public void windowOpened(WindowEvent we){}



      public void actionPerformed(ActionEvent ae)
      {
             String [] encabezado = {"ID PERSONAL", "NOMBRE", "APELLIDOS",
"FECHA NACIMIENTO", "DIRECCIÓN", "NÚMERO COLEGIADO", "ESPECIALIDAD"};
             Document documento = new Document();

             try
             {
                    rs_personal.beforeFirst();
                    rs_podologos.beforeFirst();

                    // Se crea el OutputStream para el fichero donde queremos
dejar el pdf.
                    FileOutputStream ficheroPdf = new
FileOutputStream("Listado de Podólogos.pdf");
```

```
                    // Se asocia el documento al OutputStream y se indica que
el espaciado entre
                    // lineas sera de 20. Esta llamada debe hacerse antes de
abrir el documento

      PdfWriter.getInstance(documento,ficheroPdf).setInitialLeading(20);

                    // Se abre el documento.
                    documento.open();
                    documento.add(new Paragraph("  LISTADO DE PODÓLOGOS   -
CLÍNICA SALLE",FontFactory.getFont("arial", // fuente
                                  22, // tamaño
                                  Font.ITALIC, // estilo
                                  BaseColor.BLACK)));// color
                    documento.add(new Paragraph("  "));

                    /*
                     Image foto = Image.getInstance("SIMBOLOCLINICA.png");
                    foto.scaleToFit(100, 100);
                    foto.setAlignment(Chunk.ALIGN_MIDDLE);
                    documento.add(foto);
                    PdfPTable tabla = new PdfPTable(3);
                     */

                    PdfPTable tabla = new PdfPTable(7);

                    for( int i = 0 ; i < 7; i++)
                    {
                          tabla.addCell(encabezado[i]);
                    }


                    while(rs_personal.next() && rs_podologos.next())
                    {
                          for (int i = 0; i < 7 ; i++)
                          {

                                      if( i < 5)
                                      {

      tabla.addCell(rs_personal.getString(i+1));    // EL PRIMER INDICE EN
EL RS ES EL 1, NO EL 0, POR ESO SE SUMA 1.
                                      }
                                      else
                                      {

      tabla.addCell(rs_podologos.getString(i-4));        //SE LE RESTA 4
PARA OBTENER EL VALOR 1 Y 2 DEL RESULSET
                                      }
                          }

                    }

                    documento.add(tabla);
                    documento.close();
              }
              catch ( Exception e )
              {
```

```
                    e.printStackTrace();
                }
        }
}
```

Clase VentanaListaTratamientos

```
package es.grupostudium.GestionClinica;

import java.awt.BorderLayout;
import java.awt.Choice;
import java.awt.FlowLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.FileOutputStream;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JTable;
import javax.swing.SwingConstants;
import javax.swing.table.DefaultTableCellRenderer;
import javax.swing.table.DefaultTableModel;
import com.itextpdf.text.BaseColor;
import com.itextpdf.text.Document;
import com.itextpdf.text.Font;
import com.itextpdf.text.FontFactory;
import com.itextpdf.text.Paragraph;
import com.itextpdf.text.pdf.PdfPTable;
import com.itextpdf.text.pdf.PdfWriter;

public class VentanaListaTratamientos extends JFrame implements
WindowListener, ActionListener, ItemListener
{

        private static final long serialVersionUID = 1L;

        JPanel panel1 = new JPanel();
        JPanel panel2 = new JPanel();
        JPanel panel3 = new JPanel();
        JButton imprime = new JButton("Exportar a PDF");

        DefaultTableModel modelo = new DefaultTableModel();
        Choice choice = new Choice();

        String str1 = "Elegir Paciente...";
        JTable tabla = new JTable(modelo);

        ResultSet rs_id_pacientes = null;
        ResultSet rs_nombres_pacientes = null;
        ResultSet rs_apellidos_pacientes = null;
```

```
	ResultSet rs_datos_tratamientos = null;

	Object [] encabezado = {"ID DEL PERSONAL QUE ATIENDE", "IMPORTE",
"FECHA"};

	public VentanaListaTratamientos()
	{
		setLayout(new BorderLayout());
		setSize(800,600);
		setTitle("LISTA DE TRATAMIENTOS");
		setResizable(true);


		//AÑADIMOS LAS COLUMNAS AL JTABLE
		modelo.addColumn("ID DEL PERSONAL QUE ATIENDE");
		modelo.addColumn("IMPORTE");
		modelo.addColumn("FECHA");

		//HACEMOS QUE EL TEXTO DENTRO DE LA TABLA SALGA CENTRADO
		DefaultTableCellRenderer tcr = new DefaultTableCellRenderer();
		tcr.setHorizontalAlignment(SwingConstants.CENTER); //ALINEA LOS
STRINGS EN EL CENTRO DE LA CELDA
		tabla.getColumnModel().getColumn(0).setCellRenderer(tcr);
		tabla.getColumnModel().getColumn(1).setCellRenderer(tcr);
		tabla.getColumnModel().getColumn(2).setCellRenderer(tcr);


		choice.addItemListener(this);
		choice.addItem(str1);


		rs_id_pacientes = (ResultSet) new
ConectaBBDD().obtener_id_de_los_pacientes("SELECT idPaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
		rs_nombres_pacientes = (ResultSet) new
ConectaBBDD().obtener_nombres_de_los_pacientes("SELECT nombrePaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
		rs_apellidos_pacientes = (ResultSet) new
ConectaBBDD().obtener_apellidos_de_los_pacientes("SELECT apellidosPaciente
from pacientes;");//DEVUELVE RS CON LOS APELLIDOS DE LOS PACIENTES


		try //USAMOS UN WHILE PARA RELLENAR EL JCOMBOX CON LOS
RESULTADOS DEL RESULSET
		{
			while(rs_id_pacientes.next() &&
rs_nombres_pacientes.next() && rs_apellidos_pacientes.next())
			{

	choice.addItem(rs_id_pacientes.getString("idPaciente") +"-"+
rs_nombres_pacientes.getString("nombrePaciente") +" "+
rs_apellidos_pacientes.getString("apellidosPaciente"));//AÑADE NOMBRE Y
APELLIDOS DE LOS RS USADOS ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX
			}
		} catch (SQLException e)
		{
```

```
                    System.out.println("Error al rellenar el JComboBox con los
datos de las consultas.");
                }

                try //CERRAMOS LOS RESULTSET USADOS
                {
                        rs_id_pacientes.close();
                        rs_nombres_pacientes.close();
                        rs_apellidos_pacientes.close();

                } catch (SQLException e)
                {
                        System.out.println("Error al cerrar los ResultSet de las
consultas.");
                }



                panel1.setLayout( new FlowLayout());


                panel1.add(choice);

                panel2.setLayout( new BorderLayout());
                panel2.add(tabla, "Center");

                imprime.addActionListener(this);
                panel3.setLayout(new FlowLayout());
                panel3.add(imprime);
                add(panel1,"North");
                add(panel2,"Center");
                add(panel3, "South");

                modelo.addRow(encabezado); //AÑADE ENCABEZADO DESPUÉS DE LIMPIAR




                addWindowListener(this);
                setLocationRelativeTo(null);
                setVisible(true);


        }

        public void windowActivated(WindowEvent we){}

        public void windowClosed(WindowEvent we){}

        public void windowClosing(WindowEvent we)

        {

                this.setVisible(false);

        }

        public void windowDeactivated(WindowEvent we){}
```

```
	public void windowDeiconified(WindowEvent we){}

	public void windowIconified(WindowEvent we){}

	public void windowOpened(WindowEvent we){}

	public void actionPerformed(ActionEvent ae)
	{

		if(ae.getSource().equals(imprime))
		{
			String [] encabezado = {"ID PERSONAL QUE ATIENDE",
"IMPORTE", "FECHA NACIMIENTO"};
			Document documento = new Document();

			try
			{
				rs_datos_tratamientos.beforeFirst();

				String nombre_completo_paciente = (String)
choice.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

				String[] seleciona_nombre =
nombre_completo_paciente.split("-");

				seleciona_nombre[1].toUpperCase();

				// Se crea el OutputStream para el fichero donde
queremos dejar el pdf.
				FileOutputStream ficheroPdf = new
FileOutputStream("Listado de Tratamientos.pdf");

				// Se asocia el documento al OutputStream y se
indica que el espaciado entre
				// lineas sera de 20. Esta llamada debe hacerse
antes de abrir el documento

	PdfWriter.getInstance(documento,ficheroPdf).setInitialLeading(20);

				// Se abre el documento.
				documento.open();
				documento.add(new Paragraph("  LISTADO DE
TRATAMIENTOS - "+seleciona_nombre[1].toUpperCase() +" - CLÍNICA
SALLE",FontFactory.getFont("arial", // fuente
					22, // tamaño
					Font.ITALIC, // estilo
					BaseColor.BLACK)));// color
				documento.add(new Paragraph("  "));

				/*
			 Image foto = Image.getInstance("SIMBOLOCLINICA.png");
			foto.scaleToFit(100, 100);
			foto.setAlignment(Chunk.ALIGN_MIDDLE);
			documento.add(foto);
			PdfPTable tabla = new PdfPTable(3);
				 */
```

```
                        PdfPTable tabla = new PdfPTable(3);

                        for( int i = 0 ; i < 3; i++)
                        {
                              tabla.addCell(encabezado[i]);
                        }


                        while(rs_datos_tratamientos.next())
                        {
                              for (int i = 0; i < 3 ; i++)
                              {


      tabla.addCell(rs_datos_tratamientos.getString(i+1));

                              }

                        }

                        documento.add(tabla);
                        documento.close();
                  }
                  catch ( Exception e )
                  {
                        e.printStackTrace();
                  }
            }
      }

      public void itemStateChanged(ItemEvent ie)
      {

            String nombre_completo_paciente = (String)
choice.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

            String[] seleciona_id = nombre_completo_paciente.split("-");
//METODO PARA SEPARAR EL ID DEL RESTO DEL NOMBRE


            if(!nombre_completo_paciente.equals("Elegir Paciente..."))
            {

                  try
                  {

                        //LIMPIAR EL JTABLE
                        for (int i = 0; i < tabla.getRowCount(); i++)
                        {
                              modelo.removeRow(i);
                              i -= 1;
                        }
                        modelo.addRow(encabezado); //AÑADE ENCABEZADO
DESPUÉS DE LIMPIAR
```

```
                        rs_datos_tratamientos = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_de_los_pacientes("SELECT idPersonalFK,
importeTratamiento, fechaTratamiento FROM tratamientopersonal WHERE
idPacienteFK = '"+ seleciona_id[0] + "';");  //SENTENCIA QUE RECOGE LOS DATOS
DE LOS TRATAMIENTOS

                        new Movimientos().recoge_sentencia("SELECT
idPersonalFK, importeTratamiento, fechaTratamiento FROM tratamientopersonal
WHERE idPacienteFK = '"+ seleciona_id[0] + "';"); //ESCRIBE EN EL LOG

                        // BUCLE PARA EL RESULSET CON LOS DATOS DEL
PACIENTE

                        while (rs_datos_tratamientos.next())
                        {

                              Object [] fila = new Object[3];


                              for (int i = 0; i < 3; i++)
                              {

                                    fila[i] =
rs_datos_tratamientos.getObject(i+1);  //SE SUMA 1 PORQUE EL RESULTSET NO
EMPIEZA EN 0
                              }


                              modelo.addRow(fila);
// SE AÑADE AL MODELO LA FILA COMPLETA

                        }

                  }
                  catch(SQLException e)
                  {
                        System.out.println("Error en la sentencia SQL" +
e.getMessage());
                  }

            }else
            {
                  //LIMPIAR EL JTABLE
                  for (int i = 0; i < tabla.getRowCount(); i++)
                  {
                        modelo.removeRow(i);
                        i -= 1;
                  }
                  modelo.addRow(encabezado); //AÑADE ENCABEZADO DESPUÉS DE
LIMPIAR
            }
      }
}
```

Clase VentanaModificarPaciente

```
package es.grupostudium.GestionClinica;

import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class VentanaModificarPaciente extends JFrame implements
WindowListener, ActionListener, ItemListener
{
	private static final long serialVersionUID = 1L;

	String str1 = "Elegir Paciente...";
	GridBagConstraints constraints = new GridBagConstraints(); //Crear la
distribución
	GridBagLayout gridbag = new GridBagLayout(); //Crear las restricciones

	JPanel panel1 = new JPanel();
	JPanel panel2 = new JPanel();
	JPanel panel3 = new JPanel();
	JPanel panel4 = new JPanel();
	JPanel panel5 = new JPanel();

	JLabel nombre = new JLabel("Nombre:");
	JLabel apellidos = new JLabel("Apellidos:");
	JLabel fechaNacimiento = new JLabel("Fecha de Nacimiento:");
	JLabel direccion = new JLabel("Dirección:");
	JLabel dni= new JLabel("DNI:");

	JTextField textonombre = new JTextField(14);
	JTextField textoapellidos = new JTextField(14);
	JTextField textofechaNacimiento = new JTextField(7);
	JTextField textodireccion = new JTextField(14);
	JTextField textodni = new JTextField(7);

	JButton btnGuardarCambios = new JButton("Guardar Cambios");
	JButton btnLimpiar = new JButton("Limpiar");

	JDialog dialogo3 = new JDialog(this,"MODIFICAR PACIENTE", true);
	JLabel mensaje3 = new JLabel("");
```

```
JButton btnAceptar= new JButton("Aceptar");

JComboBox choice = new JComboBox();

ResultSet rs_id_pacientes = null;
ResultSet rs_nombres_pacientes = null;
ResultSet rs_apellidos_pacientes = null;

ResultSet rs_datos_pacientes = null;

VentanaModificarPaciente()
{

	setLayout(new BorderLayout());
	setSize(600,400);
	setTitle("MODIFICAR PACIENTE");
	setResizable(false);
	setLocationRelativeTo(null);

	panel1.setLayout(gridbag);
	panel2.setLayout(new FlowLayout());


	constraints.gridx = 0;
	constraints.gridy = 0;
	constraints.gridwidth = 1;
	constraints.gridheight = 1;
	constraints.weighty=0.5;
	panel1.add(nombre,constraints);

	constraints.weighty=0.0;

	constraints.gridx = 0;
	constraints.gridy = 1;
	constraints.gridwidth = 1;
	constraints.gridheight = 1;
	constraints.weighty=0.5;
	panel1.add(apellidos,constraints);

	constraints.weighty=0.0;

	constraints.gridx = 0;
	constraints.gridy = 2;
	constraints.gridwidth = 1;
	constraints.gridheight = 1;
	constraints.weighty=0.5;
	panel1.add(fechaNacimiento,constraints);

	constraints.weighty=0.0;

	constraints.gridx = 0;
	constraints.gridy = 3;
	constraints.gridwidth = 1;
	constraints.gridheight = 1;
	constraints.weighty=0.5;
	panel1.add(direccion,constraints);

	constraints.weighty=0.0;
```

```
constraints.gridx = 0;
constraints.gridy = 4;
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
panel1.add(dni,constraints);

constraints.weighty=0.0;

constraints.gridx = 1;
constraints.gridy = 0;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(textonombre,constraints);

constraints.gridx = 1;
constraints.gridy = 1;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(textoapellidos,constraints);

constraints.gridx = 1;
constraints.gridy = 2;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(textofechaNacimiento,constraints);


constraints.gridx = 1;
constraints.gridy = 3;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(textodireccion,constraints);

constraints.gridx = 1;
constraints.gridy = 4;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(textodni,constraints);


btnGuardarCambios.addActionListener(this);
panel2.add(btnGuardarCambios,constraints);

add(panel1,"Center");
add(panel2, "South");

panel3.setLayout(new FlowLayout());
choice.addItemListener(this);
choice.addItem("Elegir Paciente...");
panel3.add(choice);
add(panel3, "North");


dialogo3.setLayout(new BorderLayout());
dialogo3.setSize(600,400);
dialogo3.setResizable(false);
```

```
            dialogo3.setLocationRelativeTo(null);
            dialogo3.addWindowListener(this);
            btnAceptar.addActionListener(this);
            panel4.setLayout(new FlowLayout());
            panel5.setLayout(new FlowLayout());
            panel4.add(mensaje3);
            panel5.add(btnAceptar);
            dialogo3.add(panel4, "North");
            dialogo3.add(panel5, "Center");


            rs_id_pacientes = (ResultSet) new
ConectaBBDD().obtener_id_de_los_pacientes("SELECT idPaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
            rs_nombres_pacientes = (ResultSet) new
ConectaBBDD().obtener_nombres_de_los_pacientes("SELECT nombrePaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
            rs_apellidos_pacientes = (ResultSet) new
ConectaBBDD().obtener_apellidos_de_los_pacientes("SELECT apellidosPaciente
from pacientes;");//DEVUELVE RS CON LOS APELLIDOS DE LOS PACIENTES


            try //USAMOS UN WHILE PARA RELLENAR EL JCOMBOX CON LOS
RESULTADOS DEL RESULSET
            {
                  while(rs_id_pacientes.next() &&
rs_nombres_pacientes.next() && rs_apellidos_pacientes.next())
                  {

      choice.addItem(rs_id_pacientes.getString("idPaciente") +"-"+
rs_nombres_pacientes.getString("nombrePaciente") +" "+
rs_apellidos_pacientes.getString("apellidosPaciente"));//AÑADE NOMBRE Y
APELLIDOS DE LOS RS USADOS ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX
                  }
            } catch (SQLException e)
            {
                  e.printStackTrace();
            }

            try //CERRAMOS LOS RESULTSET USADOS
            {
                  rs_id_pacientes.close();
                  rs_nombres_pacientes.close();
                  rs_apellidos_pacientes.close();

            } catch (SQLException e)
            {
                  e.printStackTrace();
            }


            addWindowListener(this);
            setVisible(true);
      }

      public void windowActivated(WindowEvent we){}

      public void windowClosed(WindowEvent we){}
```

```
	public void windowClosing(WindowEvent we)

	{

		this.setVisible(false);

	}

	public void windowDeactivated(WindowEvent we){}

	public void windowDeiconified(WindowEvent we){}

	public void windowIconified(WindowEvent we){}

	public void windowOpened(WindowEvent we){}



	public void actionPerformed(ActionEvent ae)
	{
		Object a;
		a=ae.getSource();

		String nombre_completo_paciente = (String)
choice.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

		if(a.equals(btnGuardarCambios) &&
!nombre_completo_paciente.equals("Elegir Paciente...")) //La segunda parte
del AND lo hacemos para que no puedas realizar cambios cuando esta marcada la
opcion "elegir paciente".
		{
			try
			{

				String[] seleciona_id =
nombre_completo_paciente.split("-");  //METODO PARA SEPARAR EL ID DEL RESTO
DEL NOMBRE

				String sentencia_modificar_paciente = "UPDATE
pacientes SET nombrePaciente = '" + textonombre.getText() + "',
apellidosPaciente = '" + textoapellidos.getText() + "',
fechaNacimientoPaciente = '" + textofechaNacimiento.getText() + "',
direccionPaciente = '" + textodireccion.getText() + "', dniPaciente = '" +
textodni.getText() + "' WHERE idPaciente = '" + seleciona_id[0] + "';";
//SENTENCIA PARA ACTUALIZAR PACIENTE EN LA BD

				if(new
ConectaBBDD().modificar_paciente(sentencia_modificar_paciente)==true)
				{

					new
Movimientos().recoge_sentencia(sentencia_modificar_paciente); //ESCRIBE EN EL
LOG LA SENTENCIA

					mensaje3.setText("Cambios realizados con
éxito");
```

```
                                dialogo3.setVisible(true);
                        }else
                        {
                                System.out.println("Error en la sentencia
UPDATE");
                                mensaje3.setText("ERROR AL ACTUALIZAR LOS
DATOS");
                                dialogo3.setVisible(true);
                        }


                }catch(Exception e)
                {
                        System.out.println(e);
                }


        }


        if(a.equals(btnAceptar))
        {
                dialogo3.setVisible(false);
        }
    }

    public void itemStateChanged(ItemEvent ie)
    {

        String nombre_completo_paciente = (String)
choice.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

        String[] seleciona_id = nombre_completo_paciente.split("-");
//METODO PARA SEPARAR EL ID DEL RESTO DEL NOMBRE

        rs_datos_pacientes = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_de_los_pacientes("SELECT
nombrePaciente, apellidosPaciente, fechaNacimientoPaciente,
direccionPaciente, dniPaciente FROM pacientes WHERE idPaciente = '" +
seleciona_id[0] +"';");   //SENTENCIA QUE RECOGE LOS DATOS DE LOS PACIENTES
MENOS EL ID PARA LUEGO RELLENAR LOS JTEXTFIELD

        if(!nombre_completo_paciente.equals("Elegir Paciente..."))
        {
                try
                {
                        // BUCLE PARA EL RESULSET CON LOS DATOS DE LOS
PACIENTES

                        while (rs_datos_pacientes.next())
                        {

      textonombre.setText(rs_datos_pacientes.getString(1));  //ESTO VA
RELLENANDO LOS JTEXTFIELD CON LOS DATOS DEL PACIENTE SELECCIONADO EN EL
JCOMBOBOX
```

```
		textoapellidos.setText(rs_datos_pacientes.getString(2));

		textofechaNacimiento.setText(rs_datos_pacientes.getString(3));


		textodireccion.setText(rs_datos_pacientes.getString(4));

		textodni.setText(rs_datos_pacientes.getString(5));
                        }
                    }catch(SQLException e)
                    {
                        System.out.println("Error en la sentencia SQL" +
e.getMessage());
                    }

                }else
                {
                    textonombre.setText("");
                    textoapellidos.setText("");
                    textofechaNacimiento.setText("");

                    textodireccion.setText("");
                    textodni.setText("");
                }

        }

}
```

Clase VentanaModificarPodologo

```
package es.grupostudium.GestionClinica;

import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class VentanaModificarPodologo extends JFrame implements
WindowListener, ActionListener, ItemListener
{
```

```
	private static final long serialVersionUID = 1L;

	String str1 = "Elegir Podólogo...";
	GridBagConstraints constraints = new GridBagConstraints(); //Crear la
distribución
	GridBagLayout gridbag = new GridBagLayout(); //Crear las restricciones

	JPanel panel1 = new JPanel();
	JPanel panel2 = new JPanel();
	JPanel panel3 = new JPanel();
	JPanel panel4 = new JPanel();
	JPanel panel5 = new JPanel();

	JLabel nombre = new JLabel("Nombre:");
	JLabel apellidos = new JLabel("Apellidos:");
	JLabel fechaNacimiento = new JLabel("Fecha de Nacimiento:");
	JLabel direccion = new JLabel("Dirección:");
	JLabel numerocolegiado = new JLabel("Número Colegiado:");
	JLabel especialidad = new JLabel("Especialidad:");

	JTextField textonombre = new JTextField(14);
	JTextField textoapellidos = new JTextField(14);
	JTextField textofechaNacimiento = new JTextField(7);
	JTextField textodireccion = new JTextField(14);
	JTextField textonumerocolegiado = new JTextField(7);
	JTextField textoespecialidad = new JTextField(14);

	JButton btnGuardarCambios = new JButton("Guardar Cambios");
	JButton btnLimpiar = new JButton("Limpiar");

	JDialog dialogo3 = new JDialog(this,"MODIFICAR PODÓLOGO", true);
	JLabel mensaje3 = new JLabel("");
	JButton btnAceptar= new JButton("Aceptar");

	JComboBox choice = new JComboBox();

	ResultSet rs_podologos = null;
	ResultSet rs_personal = null;

	VentanaModificarPodologo()
	{

		setLayout(new BorderLayout());
		setSize(600,400);
		setTitle("MODIFICAR PODÓLOGO");
		setResizable(false);
		setLocationRelativeTo(null);


		panel1.setLayout(gridbag);
		panel2.setLayout(new FlowLayout());


		constraints.gridx = 0;
		constraints.gridy = 0;
		constraints.gridwidth = 1;
		constraints.gridheight = 1;
		constraints.weighty=0.5;
```

```
panel1.add(nombre,constraints);

constraints.weighty=0.0;

constraints.gridx = 0;
constraints.gridy = 1;
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
panel1.add(apellidos,constraints);

constraints.weighty=0.0;

constraints.gridx = 0;
constraints.gridy = 2;
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
panel1.add(fechaNacimiento,constraints);

constraints.weighty=0.0;

constraints.gridx = 0;
constraints.gridy = 3;
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
panel1.add(direccion,constraints);

constraints.weighty=0.0;

constraints.gridx = 0;
constraints.gridy = 4;
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
panel1.add(numerocolegiado,constraints);

constraints.weighty=0.0;

constraints.gridx = 0;
constraints.gridy = 5;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(especialidad,constraints);

constraints.weighty=0.0;

constraints.gridx = 1;
constraints.gridy = 0;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(textonombre,constraints);

constraints.weighty=0.0;
constraints.gridx = 1;
constraints.gridy = 1;
constraints.gridwidth = 1;
```

```
constraints.gridheight = 1;
panel1.add(textoapellidos,constraints);

constraints.gridx = 1;
constraints.gridy = 2;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(textofechaNacimiento,constraints);


constraints.gridx = 1;
constraints.gridy = 3;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(textodireccion,constraints);

constraints.gridx = 1;
constraints.gridy = 4;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(textonumerocolegiado,constraints);

constraints.gridx = 1;
constraints.gridy = 5;
constraints.gridwidth = 1;
constraints.gridheight = 1;
panel1.add(textoespecialidad,constraints);


btnGuardarCambios.addActionListener(this);
panel2.add(btnGuardarCambios,constraints);

add(panel1,"Center");
add(panel2, "South");

panel3.setLayout(new FlowLayout());
choice.addItemListener(this);
choice.addItem("Elegir Podólogo...");
panel3.add(choice);
add(panel3, "North");

dialogo3.setLayout(new BorderLayout());
dialogo3.setSize(600,400);
dialogo3.setResizable(false);
dialogo3.setLocationRelativeTo(null);
dialogo3.addWindowListener(this);
btnAceptar.addActionListener(this);
panel4.setLayout(new FlowLayout());
panel5.setLayout(new FlowLayout());
panel4.add(mensaje3);
panel5.add(btnAceptar);
dialogo3.add(panel4, "North");
dialogo3.add(panel5, "Center");



try
{
```

```
                    rs_podologos = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_de_los_podologos("SELECT idPodologo,
numeroColegiadoPodologo, especialidadPodologo, idPersonalFK FROM
podologos;");

                    if(rs_podologos.next())
                    {
                          rs_podologos.beforeFirst(); //LO DEVOLVEMOS A SU
ESTADO INICIAL DESPUES DEL "IF"

                          while(rs_podologos.next())
                          {
                                rs_personal = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_del_personal("SELECT idPersonal,
nombrePersonal, apellidosPersonal, fechaNacimientoPersonal, direccionPersonal
FROM personal WHERE idPersonal='"+ rs_podologos.getString("idPersonalFK") +
"';");

                                rs_personal.next();


      choice.addItem(rs_personal.getString("idPersonal") +"-"+
rs_personal.getString("nombrePersonal") +" "+
rs_personal.getString("apellidosPersonal"));//AÑADE NOMBRE Y APELLIDOS DE LOS
RS USADOS ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX
                          }


                          rs_personal.close(); //CERRAMOS RESULSETS
                          rs_podologos.close();

                    }

             } catch (SQLException e)
             {
                    System.out.println("Error al cargar el JCOMBOBOX con las
consultas de la Base de Datos");
                    e.printStackTrace();
             }


             addWindowListener(this);
             setVisible(true);
      }


      public void windowActivated(WindowEvent we){}

      public void windowClosed(WindowEvent we){}

      public void windowClosing(WindowEvent we)

      {

            this.setVisible(false);

      }
```

```
	public void windowDeactivated(WindowEvent we){}

	public void windowDeiconified(WindowEvent we){}

	public void windowIconified(WindowEvent we){}

	public void windowOpened(WindowEvent we){}



	public void actionPerformed(ActionEvent ae)
	{
		Object a;
		a=ae.getSource();

		String nombre_completo_podologo = (String)
choice.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

		if(a.equals(btnGuardarCambios) &&
!nombre_completo_podologo.equals("Elegir Podólogo..."))
		{

			String[] seleciona_id = nombre_completo_podologo.split("-
"); //METODO PARA SEPARAR EL ID DEL RESTO DEL NOMBRE


			String sentencia_modificar_podologo = "UPDATE podologos
SET numeroColegiadoPodologo = '" + textonumerocolegiado.getText() + "',
especialidadPodologo = '" + textoespecialidad.getText() + "' WHERE
idPodologo='"+ seleciona_id[0] +"';";

			if(new
ConectaBBDD().modificar_podologo(sentencia_modificar_podologo)==true)
			{
				String sentencia_modificar_personal = "UPDATE
personal SET nombrePersonal = '" + textonombre.getText() + "',
apellidosPersonal = '" + textoapellidos.getText() + "',
fechaNacimientoPersonal = '" + textofechaNacimiento.getText() + "',
direccionPersonal = '" + textodireccion.getText() + "' WHERE idPersonal='"+
seleciona_id[0] +"';";

				new
Movimientos().recoge_sentencia(sentencia_modificar_personal); //ESCRIBE EN EL
LOG LA SENTENCIA

				if(new
ConectaBBDD().modificar_personal(sentencia_modificar_personal)==true)
				{
					mensaje3.setText("Cambios realizados con
éxito");
					dialogo3.setVisible(true);
				}
				else
				{
					mensaje3.setText("ERROR AL ACTUALIZAR LOS
DATOS");
```

```
                                dialogo3.setVisible(true);
                        }

                }else
                {
                        mensaje3.setText("ERROR AL ACTUALIZAR LOS DATOS");
                        dialogo3.setVisible(true);
                }


        }
        if(a.equals(btnAceptar))
        {
                dialogo3.setVisible(false);
        }
    }


    public void itemStateChanged(ItemEvent ie)
    {

        String nombre_completo_podologo = (String)
choice.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y LO
CONVIERTE EN STRING

        String[] seleciona_id = nombre_completo_podologo.split("-");
//METODO PARA SEPARAR EL ID DEL RESTO DEL NOMBRE


        if(!nombre_completo_podologo.equals("Elegir Podólogo..."))
        {
                try
                {

                        rs_podologos = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_de_los_podologos("SELECT
numeroColegiadoPodologo, especialidadPodologo, idPersonalFK FROM podologos
WHERE idPodologo='"+ seleciona_id[0] +"';");

                        rs_podologos.next();

                        rs_personal = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_del_personal("SELECT nombrePersonal,
apellidosPersonal, fechaNacimientoPersonal, direccionPersonal FROM personal
WHERE idPersonal='"+ rs_podologos.getString("idPersonalFK") +"';");

                        rs_personal.next();

                }
                catch (SQLException e)
                {
                        e.printStackTrace();
                        System.out.println("Error al capturar los datos
para el JCOMBOBOX");
                }

                try
                {
```

```
                        //ESTO VA RELLENANDO LOS JTEXTFIELD CON LOS DATOS
DEL PODÓLOGO SELECCIONADO EN EL JCOMBOBOX

                        textonombre.setText(rs_personal.getString(1));
                        textoapellidos.setText(rs_personal.getString(2));

      textofechaNacimiento.setText(rs_personal.getString(3));

                        textodireccion.setText(rs_personal.getString(4));

      textonumerocolegiado.setText(rs_podologos.getString(1));

      textoespecialidad.setText(rs_podologos.getString(2));

                        rs_podologos.close();
                        rs_personal.close();

                  }
                  catch (Exception e)
                  {
                        System.out.println("Error al rellenar los
JTEXTFIELD con los datos del RESULTSET");
                  }

            }else
            {
                  textonombre.setText("");
                  textoapellidos.setText("");
                  textofechaNacimiento.setText("");

                  textodireccion.setText("");
                  textonumerocolegiado.setText("");
                  textoespecialidad.setText("");
            }

      }

}
```

Clase VentanaNuevoPaciente

```
package es.grupostudium.GestionClinica;

import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
```

```
import javax.swing.JPanel;
import javax.swing.JTextField;

public class VentanaNuevoPaciente extends JFrame implements WindowListener,
ActionListener
{

	private static final long serialVersionUID = 1L;

	GridBagConstraints constraints = new GridBagConstraints(); //Crear la
distribución
	GridBagLayout gridbag = new GridBagLayout(); //Crear las restricciones

	JPanel panel1 = new JPanel();
	JPanel panel2 = new JPanel();
	JPanel panel3 = new JPanel();
	JPanel panel4 = new JPanel();

	JLabel nombre = new JLabel("Nombre:");
	JLabel apellidos = new JLabel("Apellidos:");
	JLabel fechaNacimiento = new JLabel("Fecha de Nacimiento:");
	JLabel direccion = new JLabel("Dirección:");
	JLabel dni= new JLabel("DNI:");

	JTextField textonombre = new JTextField(14);
	JTextField textoapellidos = new JTextField(14);
	JTextField textofechaNacimiento = new JTextField(7);
	JTextField textodireccion = new JTextField(14);
	JTextField textodni = new JTextField(7);

	JButton btnCrear = new JButton("Crear Paciente");
	JButton btnLimpiar = new JButton("Limpiar");

	JDialog dialogo1 = new JDialog(this, "NUEVO PACIENTE", true);
	JLabel mensaje1 = new JLabel("");
	JButton btnAceptar1= new JButton("Aceptar");

	JDialog dialogo2 = new JDialog(this, "NUEVO PACIENTE", true);
	JLabel mensaje2 = new JLabel("");
	JButton btnAceptar2= new JButton("Aceptar");



	VentanaNuevoPaciente()
	{
		setLayout(gridbag);
		setTitle("NUEVO PACIENTE");
		setResizable(false);
		setSize(600,400);
		setLocationRelativeTo(null);


		constraints.insets = new Insets(0,0,0,50); //MARGEN A LA DERECHA
		constraints.anchor =  GridBagConstraints.LINE_END;


		constraints.gridx = 0;
		constraints.gridy = 0;
```

```
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
add(nombre,constraints);

constraints.weighty=0.0;

constraints.gridx = 0;
constraints.gridy = 1;
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
add(apellidos,constraints);

constraints.weighty=0.0;

constraints.gridx = 0;
constraints.gridy = 2;
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
add(fechaNacimiento,constraints);

constraints.weighty=0.0;

constraints.gridx = 0;
constraints.gridy = 3;
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
add(direccion,constraints);

constraints.weighty=0.0;

constraints.gridx = 0;
constraints.gridy = 4;
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
add(dni,constraints);

constraints.weighty=0.0;

constraints.gridx = 1;
constraints.gridy = 0;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textonombre,constraints);

constraints.gridx = 1;
constraints.gridy = 1;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textoapellidos,constraints);

constraints.anchor =  GridBagConstraints.LINE_START;
constraints.gridx = 1;
constraints.gridy = 2;
```

```
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textofechaNacimiento,constraints);
constraints.anchor =  GridBagConstraints.LINE_END;

constraints.gridx = 1;
constraints.gridy = 3;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textodireccion,constraints);

constraints.anchor =  GridBagConstraints.LINE_START;
constraints.gridx = 1;
constraints.gridy = 4;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textodni,constraints);
constraints.anchor =  GridBagConstraints.LINE_END;

constraints.insets = new Insets(0,0,0,0); //MARGEN A LA DERECHA
constraints.anchor =  GridBagConstraints.CENTER;
constraints.fill = GridBagConstraints.NONE;

constraints.gridx = 0;
constraints.gridy = 5;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(btnCrear,constraints);
btnCrear.addActionListener(this);

constraints.gridx = 1;
constraints.gridy = 5;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(btnLimpiar,constraints);
btnLimpiar.addActionListener(this);

dialogo1.setLayout(new BorderLayout());
dialogo1.setSize(600,400);
dialogo1.setResizable(false);
dialogo1.setLocationRelativeTo(null);
dialogo1.addWindowListener(this);
btnAceptar1.addActionListener(this);
panel1.setLayout(new FlowLayout());
panel2.setLayout(new FlowLayout());
panel1.add(mensaje1);
panel2.add(btnAceptar1);
dialogo1.add(panel1, "North");
dialogo1.add(panel2, "Center");

dialogo2.setLayout(new BorderLayout());
dialogo2.setSize(600,400);
dialogo2.setResizable(true);
dialogo2.setLocationRelativeTo(null);
dialogo2.addWindowListener(this);
btnAceptar2.addActionListener(this);
panel3.setLayout(new FlowLayout());
panel4.setLayout(new FlowLayout());
```

```
            panel3.add(mensaje2);
            panel4.add(btnAceptar2);
            dialogo2.add(panel3, "North");
            dialogo2.add(panel4, "Center");

            addWindowListener(this);
            setVisible(true);

      }


      public void windowActivated(WindowEvent we){}

      public void windowClosed(WindowEvent we){}

      public void windowClosing(WindowEvent we)
      {
            this.setVisible(false);
      }

      public void windowDeactivated(WindowEvent we){}

      public void windowDeiconified(WindowEvent we){}

      public void windowIconified(WindowEvent we){}

      public void windowOpened(WindowEvent we){}


      public void actionPerformed(ActionEvent ae)
      {
            Object a;
            a=ae.getSource();
            if(a.equals(btnLimpiar))
            {
                  textonombre.selectAll();
                  textonombre.setText("");
                  textoapellidos.selectAll();
                  textoapellidos.setText("");
                  textofechaNacimiento.selectAll();
                  textofechaNacimiento.setText("");
                  textodireccion.selectAll();
                  textodireccion.setText("");
                  textodni.selectAll();
                  textodni.setText("");
                  textonombre.requestFocus();
            }
            if(a.equals(btnCrear))
            {
                  String sentencia_nuevo_paciente = "INSERT INTO pacientes
(nombrePaciente, apellidosPaciente, fechaNacimientoPaciente,
direccionPaciente, dniPaciente) VALUES('" +textonombre.getText()+ "','" +
textoapellidos.getText() + "','" + textofechaNacimiento.getText() + "','" +
textodireccion.getText() + "','" + textodni.getText() + "');";

                  if(new
ConectaBBDD().agregar_paciente(sentencia_nuevo_paciente)==true)  //SENTENCIA
PARA AGREGAR UN NUEVO PACIENTE A LA BD
```

```
                    {
                        new Movimientos().recoge_sentencia("INSERT INTO
pacientes (nombrePaciente, apellidosPaciente, fechaNacimientoPaciente,
direccionPaciente, dniPaciente) VALUES('" +textonombre.getText()+ "','" +
textoapellidos.getText() + "','" + textofechaNacimiento.getText() + "','" +
textodireccion.getText() + "','" + textodni.getText() + "');"); //ESCRIBE EN
EL LOG LA SENTENCIA

                        mensaje1.setText("Paciente creado con éxito");
                        dialogo1.setVisible(true);

                        //LIMPIAMOS TODOS LOS JTEXTFIELD PARA INTRODUCIR
OTRO PACIENTE
                        textonombre.selectAll();
                        textonombre.setText("");
                        textoapellidos.selectAll();
                        textoapellidos.setText("");
                        textofechaNacimiento.selectAll();
                        textofechaNacimiento.setText("");
                        textodireccion.selectAll();
                        textodireccion.setText("");
                        textodni.selectAll();
                        textodni.setText("");
                        textonombre.requestFocus();

                    }else
                    {
                        mensaje2.setText("<html><body>Error al insertar los
datos por parte del Usuario.<br> EJEMPLO DE FORMATO DE FECHA VÁLIDO AAAA-MM-
DD --> 2019-04-18<br>EJEMPLO DE FORMATO DE DNI VÁLIDO (9 CARACTERES) -->
00000000A <br>INSERTAR TODOS LOS CAMPOS</body></html>");
                        dialogo2.setVisible(true);
                    }

            }
            if(a.equals(btnAceptar1))
            {
                    dialogo1.setVisible(false);
            }
            if(a.equals(btnAceptar2))
            {
                    dialogo2.setVisible(false);
            }

      }

}
```

Clase VentanaNuevoPodologo

```
package es.grupostudium.GestionClinica;

import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class VentanaNuevoPodologo extends JFrame implements WindowListener,
ActionListener
{
	private static final long serialVersionUID = 1L;

	GridBagConstraints constraints = new GridBagConstraints(); //Crear la
distribución
	GridBagLayout gridbag = new GridBagLayout(); //Crear las restricciones

	JPanel panel1 = new JPanel();
	JPanel panel2 = new JPanel();
	JPanel panel3 = new JPanel();
	JPanel panel4 = new JPanel();

	JLabel nombre = new JLabel("Nombre:");
	JLabel apellidos = new JLabel("Apellidos:");
	JLabel fechaNacimiento = new JLabel("Fecha de Nacimiento:");
	JLabel direccion = new JLabel("Dirección:");
	JLabel numeroColegiado= new JLabel("Número de Colegiado:");
	JLabel especialidad = new JLabel("Especialidad:");

	JTextField textonombre = new JTextField(14);
	JTextField textoapellidos = new JTextField(14);
	JTextField textofechaNacimiento = new JTextField(7);
	JTextField textodireccion = new JTextField(14);
	JTextField textonumeroColegiado = new JTextField(7);
	JTextField textoespecialidad= new JTextField(14);

	JButton btnCrear = new JButton("Crear Podólogo");
	JButton btnLimpiar = new JButton("Limpiar");

	JDialog dialogo1 = new JDialog(this, "NUEVO PODÓLOGO", true);
	JLabel mensaje1 = new JLabel("");
	JButton btnAceptar= new JButton("Aceptar");

	JDialog dialogo2 = new JDialog(this, "NUEVO PODÓLOGO", true);
```

```
JLabel mensaje2 = new JLabel("");
JButton btnAceptar2= new JButton("Aceptar");

ResultSet rs = null;

VentanaNuevoPodologo()
{
	setLayout(gridbag);
	setTitle("NUEVO PODÓLOGO");
	setResizable(false);
	setSize(600,400);
	setLocationRelativeTo(null);


	constraints.gridx = 0;
	constraints.gridy = 0;
	constraints.gridwidth = 1;
	constraints.gridheight = 1;
	constraints.weighty=0.5;
	add(nombre,constraints);

	constraints.weighty=0.0;

	constraints.gridx = 0;
	constraints.gridy = 1;
	constraints.gridwidth = 1;
	constraints.gridheight = 1;
	constraints.weighty=0.5;
	add(apellidos,constraints);

	constraints.weighty=0.0;

	constraints.gridx = 0;
	constraints.gridy = 2;
	constraints.gridwidth = 1;
	constraints.gridheight = 1;
	constraints.weighty=0.5;
	add(fechaNacimiento,constraints);

	constraints.weighty=0.0;

	constraints.gridx = 0;
	constraints.gridy = 3;
	constraints.gridwidth = 1;
	constraints.gridheight = 1;
	constraints.weighty=0.5;
	add(direccion,constraints);

	constraints.weighty=0.0;

	constraints.gridx = 0;
	constraints.gridy = 4;
	constraints.gridwidth = 1;
	constraints.gridheight = 1;
	constraints.weighty=0.5;
	add(numeroColegiado,constraints);

	constraints.weighty=0.0;
```

```
constraints.gridx = 0;
constraints.gridy = 5;
constraints.gridwidth = 1;
constraints.gridheight = 1;
constraints.weighty=0.5;
add(especialidad,constraints);

constraints.weighty=0.0;

constraints.gridx = 1;
constraints.gridy = 0;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textonombre,constraints);

constraints.gridx = 1;
constraints.gridy = 1;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textoapellidos,constraints);

constraints.gridx = 1;
constraints.gridy = 2;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textofechaNacimiento,constraints);

constraints.gridx = 1;
constraints.gridy = 3;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textodireccion,constraints);

constraints.gridx = 1;
constraints.gridy = 4;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textonumeroColegiado,constraints);

constraints.gridx = 1;
constraints.gridy = 5;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(textoespecialidad,constraints);

constraints.gridx = 0;
constraints.gridy = 6;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(btnCrear,constraints);
btnCrear.addActionListener(this);

constraints.gridx = 1;
constraints.gridy = 6;
constraints.gridwidth = 1;
constraints.gridheight = 1;
add(btnLimpiar,constraints);
```

```
        btnLimpiar.addActionListener(this);

        dialogo1.setLayout(new        BorderLayout());
        dialogo1.setSize(600,400);
        dialogo1.setResizable(false);
        dialogo1.setLocationRelativeTo(null);
        dialogo1.addWindowListener(this);
        btnAceptar.addActionListener(this);
        panel1.setLayout(new FlowLayout());

        panel2.setLayout(new FlowLayout());
        panel1.add(mensaje1);
        panel2.add(btnAceptar);
        dialogo1.add(panel1, "North");
        dialogo1.add(panel2,"Center");

        dialogo2.setLayout(new BorderLayout());
        dialogo2.setSize(600,400);
        dialogo2.setResizable(true);
        dialogo2.setLocationRelativeTo(null);
        dialogo2.addWindowListener(this);
        btnAceptar2.addActionListener(this);
        panel3.setLayout(new FlowLayout());
        panel4.setLayout(new FlowLayout());
        panel3.add(mensaje2);
        panel4.add(btnAceptar2);
        dialogo2.add(panel3, "North");
        dialogo2.add(panel4, "Center");

        addWindowListener(this);
        setVisible(true);
}


public void windowActivated(WindowEvent we){}

public void windowClosed(WindowEvent we){}

public void windowClosing(WindowEvent we)

{

        this.setVisible(false);

}

public void windowDeactivated(WindowEvent we){}

public void windowDeiconified(WindowEvent we){}

public void windowIconified(WindowEvent we){}

public void windowOpened(WindowEvent we){}


public void actionPerformed(ActionEvent ae)
```

```
	{
		Object a;
		a=ae.getSource();
		if(a.equals(btnLimpiar))
		{
			textonombre.selectAll();
			textonombre.setText("");
			textoapellidos.selectAll();
			textoapellidos.setText("");
			textofechaNacimiento.selectAll();
			textofechaNacimiento.setText("");
			textodireccion.selectAll();
			textodireccion.setText("");
			textonumeroColegiado.selectAll();
			textonumeroColegiado.setText("");
			textoespecialidad.selectAll();
			textoespecialidad.setText("");
			textonombre.requestFocus();
		}
		if(a.equals(btnCrear))
		{

			String sentencia_nuevo_personal = "INSERT INTO personal
(nombrePersonal, apellidosPersonal, fechaNacimientoPersonal,
direccionPersonal) VALUES('" +textonombre.getText()+ "','" +
textoapellidos.getText() + "','" + textofechaNacimiento.getText() + "','" +
textodireccion.getText() + "');";

			if(new
ConectaBBDD().agregar_personal(sentencia_nuevo_personal)==true)  //SENTENCIA
PARA AGREGAR UN NUEVO PACIENTE A LA BD
			{

				new
Movimientos().recoge_sentencia(sentencia_nuevo_personal); //ESCRIBE EN EL LOG
LA SENTENCIA

				mensaje1.setText("Podólogo creado con éxito.");
				dialogo1.setVisible(true);


				String sentencia_datos_personal = "SELECT
idPersonal, apellidosPersonal, fechaNacimientoPersonal, direccionPersonal
from personal;";

				rs = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_del_personal(sentencia_datos_personal);

				try
				{
					rs.last();

					String sentencia_nuevo_podologo = "INSERT
INTO podologos (numeroColegiadoPodologo,  especialidadPodologo,
idPersonalFK) VALUES ('" + textonumeroColegiado.getText() +"', '" +
textoespecialidad.getText() + "', '" + rs.getString(1) + "');";
```

```
                                    new
ConectaBBDD().agregar_podologo(sentencia_nuevo_podologo);

                                }
                                catch (SQLException e)
                                {
                                        System.out.println("Error al asociar con el
ID del Personal");
                                }


                                //VACIAMOS LOS JTEXTFIELD PARA EL SIGUIENTE
REGISTRO
                                textonombre.selectAll();
                                textonombre.setText("");
                                textoapellidos.selectAll();
                                textoapellidos.setText("");
                                textofechaNacimiento.selectAll();
                                textofechaNacimiento.setText("");
                                textodireccion.selectAll();
                                textodireccion.setText("");
                                textonumeroColegiado.selectAll();
                                textonumeroColegiado.setText("");
                                textoespecialidad.selectAll();
                                textoespecialidad.setText("");
                                textonombre.requestFocus();

                        }
                        else
                        {
                                mensaje2.setText("<html><body>Error al insertar los
datos por parte del Usuario.<br> EJEMPLO DE FORMATO DE FECHA VÁLIDO AAAA-MM-
DD --> 2019-04-18</html></body>");
                                dialogo2.setVisible(true);
                        }


                }
                if(a.equals(btnAceptar))
                {
                        dialogo1.setVisible(false);
                }
                if(a.equals(btnAceptar2))
                {
                        dialogo2.setVisible(false);
                }


        }

}
```

Clase VentanaNuevoTratamiento

```
package es.grupostudium.GestionClinica;


import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class VentanaNuevoTratamiento extends JFrame implements
WindowListener, ActionListener, MouseListener
{
	private static final long serialVersionUID = 1L;

	String recoge_fecha = new String();
	String fecha_invertida = new String();

	JPanel panel1 = new JPanel();
	JPanel panel2 = new JPanel();
	JPanel panel3 = new JPanel();
	JPanel panel4 = new JPanel();

	JLabel profesional = new JLabel("Profesional");
	JLabel paciente = new JLabel("Paciente");
	JLabel importe = new JLabel("Importe del Tratamiento");
	JLabel fecha = new JLabel("Fecha del Tratamiento");

	JComboBox textoprofesional = new JComboBox();
	JComboBox textopaciente = new JComboBox();
	JTextField textoimporte = new JTextField(30);
	JTextField textofecha = new JTextField(30);

	JButton btnAceptar= new JButton("Aceptar");
	JButton btnEliminar = new JButton("Eliminar");

	JDialog dialogo1 = new JDialog(this, "NUEVO TRATAMIENTO", true);
	JLabel mensaje1 = new JLabel("Tratamiento añadido con éxito.");
	JButton btnAceptar1= new JButton("Aceptar");

	ResultSet rs_personal = null;

	ResultSet rs_id_pacientes = null;
```

```
	ResultSet rs_nombres_pacientes = null;
	ResultSet rs_apellidos_pacientes = null;


	VentanaNuevoTratamiento()
	{
		setLayout(new BorderLayout());
		setTitle("NUEVO TRATAMIENTO");
		setResizable(true);
		setSize(800,300);
		setLocationRelativeTo(null);


		panel1.setLayout(new GridLayout(2,4));
		panel2.setLayout(new FlowLayout());

		btnAceptar.addActionListener(this);
		panel2.add(btnAceptar);

		panel1.add(profesional);
		panel1.add(paciente);
		panel1.add(importe);
		panel1.add(fecha);
		panel1.add(textoprofesional);
		panel1.add(textopaciente);
		panel1.add(textoimporte);
		panel1.add(textofecha);


		add(panel1,"North");
		add(panel2, "Center");

		dialogo1.setLayout(new BorderLayout());
		dialogo1.setSize(800,300);
		dialogo1.setTitle("NUEVO TRATAMIENTO");
		dialogo1.setResizable(true);
		dialogo1.setLocationRelativeTo(null);
		dialogo1.addWindowListener(this);
		btnAceptar1.addActionListener(this);
		panel3.add(mensaje1);
		panel4.add(btnAceptar1);
		dialogo1.add(panel3, "North");
		dialogo1.add(panel4, "Center");

		rs_personal = (ResultSet) new
ConectaBBDD().obtener_todos_los_datos_del_personal("SELECT * FROM
personal;");

		try
		{

			textoprofesional.addItem("Elegir Profesional...");

			while(rs_personal.next())
			{


	textoprofesional.addItem(rs_personal.getString("idPersonal") +"-"+
```

```
rs_personal.getString("nombrePersonal") +" "+
rs_personal.getString("apellidosPersonal"));//AÑADE NOMBRE Y APELLIDOS DE LOS
RS USADOS ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX

                    }

                    rs_personal.close();


              } catch (SQLException e)
              {
                    System.out.println("Error al cargar el JCOMBOBOX 1 con las
consultas de la Base de Datos");

              }


              rs_id_pacientes = (ResultSet) new
ConectaBBDD().obtener_id_de_los_pacientes("SELECT idPaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
              rs_nombres_pacientes = (ResultSet) new
ConectaBBDD().obtener_nombres_de_los_pacientes("SELECT nombrePaciente FROM
pacientes;"); //DEVUELVE RS CON LOS NOMBRES DE LOS PACIENTES
              rs_apellidos_pacientes = (ResultSet) new
ConectaBBDD().obtener_apellidos_de_los_pacientes("SELECT apellidosPaciente
from pacientes;");//DEVUELVE RS CON LOS APELLIDOS DE LOS PACIENTES


              try //USAMOS UN WHILE PARA RELLENAR EL JCOMBOX CON LOS
RESULTADOS DEL RESULSET
              {
                    textopaciente.addItem("Elegir Paciente...");

                    while(rs_id_pacientes.next() &&
rs_nombres_pacientes.next() && rs_apellidos_pacientes.next())
                    {

      textopaciente.addItem(rs_id_pacientes.getString("idPaciente") +"-"+
rs_nombres_pacientes.getString("nombrePaciente") +" "+
rs_apellidos_pacientes.getString("apellidosPaciente"));//AÑADE NOMBRE Y
APELLIDOS DE LOS RS USADOS ANTERIORMENTE Y LOS METE EN UN JCOMBOBOX
                    }

                    rs_id_pacientes.close();
                    rs_nombres_pacientes.close();
                    rs_apellidos_pacientes.close();

              }
              catch (SQLException e)
              {

                    System.out.println("Error al cargar el JCOMBOBOX 2 con las
consultas de la Base de Datos");
              }


              recoge_fecha = new Movimientos().recoge_fecha_actual();
              textofecha.setText(recoge_fecha);
```

```
		textoimporte.setText("0.00");

		textoimporte.addMouseListener(this);
		addWindowListener(this);
		setVisible(true);
	}



	public void windowActivated(WindowEvent we){}

	public void windowClosed(WindowEvent we){}

	public void windowClosing(WindowEvent we)

	{

		this.setVisible(false);

	}

	public void windowDeactivated(WindowEvent we){}

	public void windowDeiconified(WindowEvent we){}

	public void windowIconified(WindowEvent we){}

	public void windowOpened(WindowEvent we){}



	public void actionPerformed(ActionEvent ae)
	{

		Object a;
		a=ae.getSource();


		if(a.equals(btnAceptar))
		{

			try {


				String nombre_completo_paciente = (String)
textopaciente.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL JCOMBOBOX Y
LO CONVIERTE EN STRING

				String[] seleciona_id_1 =
nombre_completo_paciente.split("-");  //METODO PARA SEPARAR EL ID DEL RESTO
DEL NOMBRE

				String nombre_completo_podologo = (String)
textoprofesional.getSelectedItem(); //RECOGE LO QUE SELECCIONA EN EL
JCOMBOBOX Y LO CONVIERTE EN STRING
```

```
                        String[] seleciona_id_2 =
nombre_completo_podologo.split("-");  //METODO PARA SEPARAR EL ID DEL RESTO
DEL NOMBRE

                        double importe =
Double.parseDouble(textoimporte.getText());

                        if(new ConectaBBDD().agregar_tratamiento("INSERT
INTO tratamientopersonal (idPacienteFK, idPersonalFK, importeTratamiento,
fechaTratamiento) VALUES ("+seleciona_id_1[0]+", "+ seleciona_id_2[0] +", "+
importe +", '"+ textofecha.getText()+"');") == true)
                        {
                              new Movimientos().recoge_sentencia("INSERT
INTO tratamientopersonal (idPacienteFK, idPersonalFK, importeTratamiento,
fechaTratamiento) VALUES ("+seleciona_id_1[0]+", "+ seleciona_id_2[0] +", "+
importe +", '"+ textofecha.getText()+"');"); //ESCRIBE EN EL LOG LA SENTENCIA
                              dialogo1.setVisible(true);
                        }
                        else
                        {
                              mensaje1.setText("<html><body>Error al
insertar los datos por parte del Usuario.<br> EJEMPLO DE FORMATO DE FECHA
VÁLIDO AAAA-MM-DD --> 2019-04-18<br>EJEMPLO DE FORMATO DE IMPORTE VÁLIDO
00.00 --> 50.05<br>SE DEBE ELEGIR UN PROFESIONAL Y UN PACIENTE<br>TODOS LOS
CAMPOS SON OBLIGATORIOS</html></body>");
                              dialogo1.setVisible(true);
                        }


                  }
                  /*catch(Exception e)
                  {
                        e.printStackTrace();
                        mensaje1.setText("<html><body>Error al insertar los
datos por parte del Usuario.<br> EJEMPLO DE FORMATO DE FECHA VÁLIDO AAAA-MM-
DD --> 2019-04-18<br>EJEMPLO DE FORMATO DE IMPORTE VÁLIDO   00.00 -->
50.05<br>SE DEBE ELEGIR UN PROFESIONAL Y UN PACIENTE<br>TODOS LOS CAMPOS SON
OBLIGATORIOS</html></body>");
                        dialogo1.setVisible(true);
                  }*/
                  catch(NumberFormatException e)
                  {
                        textoimporte.setText("");
                        mensaje1.setText("<html><body>Error al insertar los
datos por parte del Usuario.<br> EJEMPLO DE FORMATO DE FECHA VÁLIDO AAAA-MM-
DD --> 2019-04-18<br>EJEMPLO DE FORMATO DE IMPORTE VÁLIDO   00.00 -->
50.05<br>SE DEBE ELEGIR UN PROFESIONAL Y UN PACIENTE<br>TODOS LOS CAMPOS SON
OBLIGATORIOS</html></body>");
                        dialogo1.setVisible(true);
                  }


            }

            if(a.equals(btnAceptar1))
            {
                  dialogo1.setVisible(false);
            }
```

```
        }

        public void mouseClicked(MouseEvent me){}
        public void mouseEntered(MouseEvent me){}
        public void mouseExited(MouseEvent me){}
        public void mouseReleased(MouseEvent me){}

        public void mousePressed(MouseEvent me)
        {
                if(textoimporte.requestFocus(true))
                {
                        textoimporte.setText("");
                }
        }

}
```

Librerías

Para el desarrollo de la aplicación, se han utilizado las siguientes librerías externas, que proporcionan las utilidades necesarias para conectarse a la BD.

- **MySQL Connector/J 5.1.46**: http://dev.mysql.com/downloads/connector/j/
- **IText 5.5.5:** http://itextpdf.com/release/iText555

Instalación

Para poder ejecutar la aplicación, se requieren los siguientes componentes:

- MySQL Server (usuario: administrativo / contraseña: studium2019;).
- BD de la Policlínica.
- Java SE Development Kit 8.
- Archivo JAR de la aplicación.

Una vez que se cumplan estos requisitos, simplemente tenemos que abrir un terminal y, estando en la ruta pertinente, escribir la siguiente sentencia:

java –jar policlinica.jar

Clases Java

A continuación, se realiza una breve explicación sobre cada una de las clases desarrolladas en la presente práctica.

Clase Ayuda: Clase que contiene un menú de ayuda para el usuario.

Clase ConectaBBDD: Conecta con la Base de Datos y contiene todos los métodos necesarios para interactuar con ella.

Clase GestionClinica1: Contiene la interfaz gráfica de las distintas ventanas con las que podrá interactuar el usuario “admin”.

Clase GestionClinica2: Contiene la interfaz gráfica de las distintas ventanas con las que podrá interactuar el usuario “user”.

Clase Login: Contiene la ventana principal que aparece cuando se ejecuta el programa. En ella tenemos los campos necesarios para que los usuarios se autentifiquen.

Clase Movimientos: Clase que registra todas las acciones CRUD por parte del usuario en un archivo “.txt”.

Clase VentanaEliminarPaciente: Clase que permite borrar datos de la tabla “pacientes” mediante una interfaz gráfica.

Clase VentanaEliminarPodologo: Clase que permite borrar datos de la tabla “podologos” y “personal” mediante una interfaz gráfica.

Clase VentanaEliminarTratamiento: Clase que permite borrar datos de la tabla “tratamientoPersonal” mediante una interfaz gráfica.

Clase VentanaListaPacientes: Clase que muestra en una interfaz gráfica todos los registros de la tabla “pacientes”. Con la posibilidad de exportar a PDF los listados.

Clase VentanaListaPodologos: Clase que muestra en una interfaz gráfica todos los registros de la tabla “podologos” y “personal” combinadas entre ambas. Con la posibilidad de exportar a PDF los listados.

Clase VentanaListaTratamientos: Clase que muestra en una interfaz gráfica todos los registros de la tabla “tratamientoPersonal”. Con la posibilidad de exportar a PDF los listados.

Clase VentanaModificarPaciente: Clase que muestra los registros de la tabla “pacientes” y nos permite hacer modificaciones en sus datos mediante una interfaz gráfica.

Clase VentanaModificarPodologo: Clase que muestra los registros de la tabla “podólogos” y nos permite hacer modificaciones en sus datos mediante una interfaz gráfica.

Clase VentanaNuevoPaciente: Clase que nos permite crear un registro en la tabla “pacientes” mediante una interfaz gráfica.

Clase VentanaNuevoPodologo: Clase que nos permite crear un registro en la tabla “podologos” mediante una interfaz gráfica.

Clase VentanaNuevoTratamiento: Clase que nos permite crear un registro en la tabla “tratamientoPersonal” mediante una interfaz gráfica.