***Documentación del sistema médico***

Contenido

[Introducción. 2](#_Toc24556692)

[Análisis 2](#_Toc24556693)

[Objetivo. 2](#_Toc24556694)

[Método. 2](#_Toc24556695)

[Estrategia 2](#_Toc24556696)

[Pantalla de inicio. 2](#_Toc24556697)

[Pacientes 3](#_Toc24556698)

[Medico 3](#_Toc24556699)

[Planteamiento de la solución e inicio de proyecto 3](#_Toc24556700)

[Diseño 4](#_Toc24556701)

[Base de datos 4](#_Toc24556702)

[Creación de BD. 4](#_Toc24556703)

[Recolección de datos en tabla 4](#_Toc24556704)

[INSERTAR LA BD A NUESTRO LOCAL HOST 6](#_Toc24556705)

[Interfaces: 13](#_Toc24556706)

[Implementación: 13](#_Toc24556707)

# Introducción.

Hoy el uso de los avances de las tecnologías de la información en la sociedad es imprescindible, de ahí la necesidad de sistemas automatizados que permitan la gestión de un gran volumen de datos. En cuanto a la cuestión médica se ha dado pasos importantes en la informatización con la construcción de infraestructuras de telecomunicaciones actualmente asumidas por sistemas manuales con sus posibles sesgos humanos y de tiempo.

# Análisis

## Objetivo.

Diseñar el prototipo de una aplicación informática que muestre la información que se requiere proporcionar dentro de un centro médico de forma interoperable, como son el caso de:

* **Expediente (historial clínico y reportes).**
* **Médicos.**
* **Interconsultas (citas a consultorio).**
* **Agenda.**
* **Pacientes.**
* **Facturación (pagos).**
* **Asistente (servicios y ayuda).**
* **Medicamentos.**
* **Inicio de sesión (Información del centro médico).**

## Método.

Se trata de una Investigación de Innovación Tecnológica sobre el sistema movimiento hospitalario en el sector salud, utilizando la metodología Programación Paralela, modelando con el Lenguaje Unificado de Modelado la aplicación, ajustando el desarrollo de los flujos de trabajo según las especificaciones de UML para el desarrollo.

## Estrategia

Para llevar a cabo el desarrollo de la aplicación es necesario realizar un análisis con el cual se encuentran aspectos importantes que servirán como cimiento para el desarrollo de la aplicación. Dentro de las cuales podemos destacar los componentes que la aplicación requerirá para poder funcionar de manera correcta.

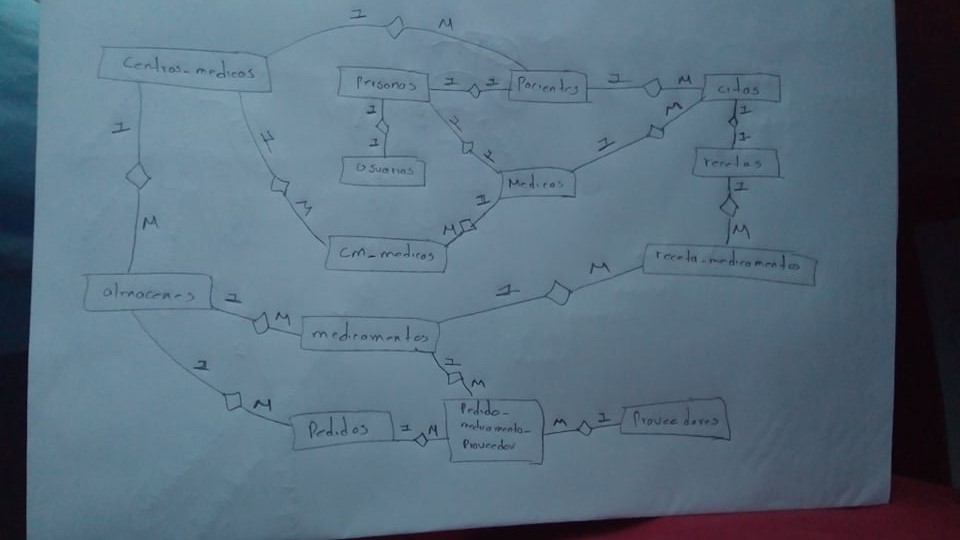
Dentro de los componentes que se ha podido extraer dentro del análisis se encuentra:

## Pantalla de inicio.

La pantalla de inicio servirá para mostrar información de acerca del centro médico, como lo son; la visión, la misión, los valores, los servicios, los especialistas, horas de servicio, información de contacto y de ubicación.

Dentro del panel de inicio se puede crear un carrusel que muestre imágenes sobre el centro de salud y junto con estos a parezcan en un jlabel la información que se presenta en imagen.

Se hará uso de separadores para separar la información principal del centro médico, la cual se podrá mostrar en una tabla para organizar de mejor manera la información de contacto y de ubicación.



## Pacientes

Para tener registro de los pacientes se requiere el uso de formularios, los cuales sirven para obtener datos importantes del paciente, como lo pueden ser: nombre, apellido paterno, apellido materno, edad, sexo, fecha de nacimiento, número de seguro social (si lo tiene), ocupación, nacionalidad, localidad, código postal, domicilio, teléfono, correo electrónico, (doctor que atiende, pero este se ve reflejado en el expediente y/o historial clínico).

## Medico

Para el registro del médico son requeridos datos como: nombre, apellido paterno y materno, rfc, curp, cedula profesional, edad, sexo, teléfono particular y de oficina, correo y datos de contacto y ubicación.

.

## Planteamiento de la solución e inicio de proyecto

El proyecto está pensado realizarse en un entorno java con métodos de programación paralela para facilitar y optimizar los procesos del sistema.

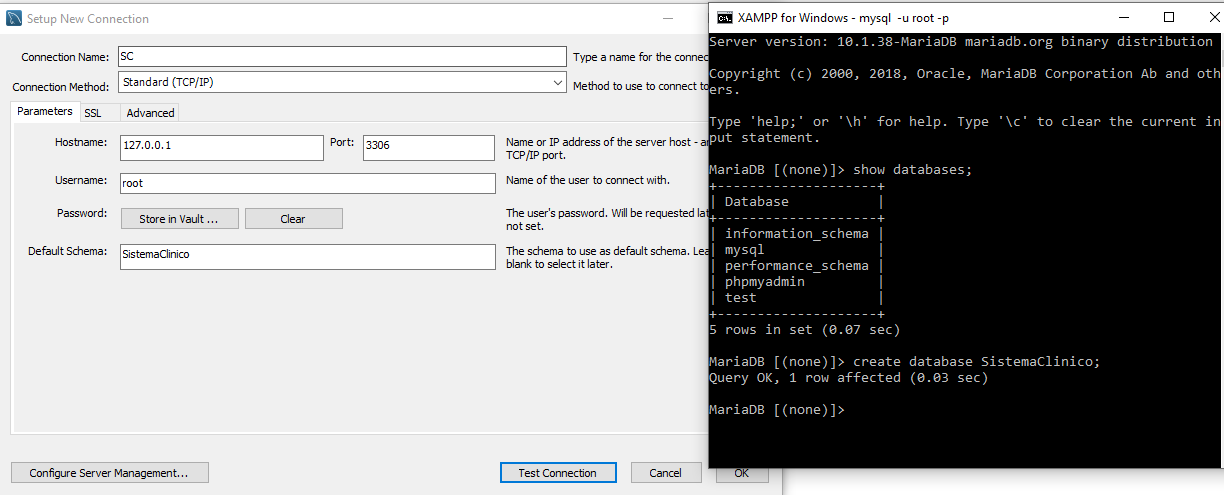
# Diseño

## Base de datos

En el sistema se hara uso de base de datos y sus gestores, como de inicio se comenzara de forma local se recomienda usar xampp, o MySql Workbench para crear y trabajar en ellas.

### Creación de BD.

Se creara una BD que haga referencia al sistema clínico, esta base de datos puede ser creado por medio de consola y o forma gráfica, después se continuara con la prueba de conexión y la conexión de la BD dentro del gestor.



Creación de la BD y prueba de conexión

Posterior mente se da paso a la creación de las tablas que contendrán la información, no sin antes haber realizado el análisis para ver campos necesitaran ser creados y bajo que especificaciones deben de ser creados.

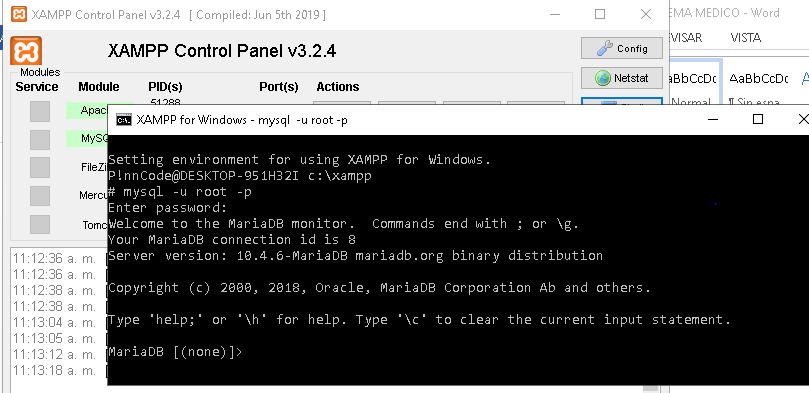
### Recolección de datos en tabla

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **paciente** | **doctor** | **Centro medico** | **medicamentos** | **factura** | **expediente** |
| **Nombre** | **x** | **x** |  | **x** |  | **x** |
| **Apellido paterno** | **x** | **x** |  |  |  | **x** |
| **Apellido materno** | **x** | **x** |  |  |  | **x** |
| **Teléfono** | **x** | **x** | **x** |  |  | **x** |
| **celular** | **x** | **x** |  |  |  | **x** |
| **Correo** | **x** | **x** | **x** |  |  | **x** |
| **Dirección** | **x** | **x** | **x** |  |  | **x** |
| **nacionalidad** | **x** | **x** |  |  |  | **x** |
| **Edad** | **x** | **x** |  |  |  | **x** |
| **Sexo** | **x** | **x** |  |  |  | **x** |
| **Curp** | **x** | **x** |  |  |  | **x** |
| **Rfc** | **x** | **x** |  |  |  | **x** |
| **Cedula profesional** | **x** | **x** |  |  |  | **x** |
| **NSS** | **x** | **x** |  |  |  | **x** |
| **Código postal** | **x** | **x** | **x** |  |  | **x** |
| **Fechas** |  |  |  | **x** | **x** | **x** |
| **precio** |  |  |  | **x** | **x** |  |
| **usos** |  |  |  | **x** |  | **x** |
| **descripción** |  |  | **x** |  |  | **x** |

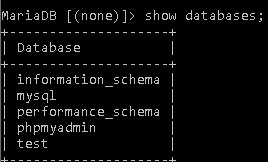
### INSERTAR LA BD A NUESTRO LOCAL HOST

1.- tener instalado XAMPP

2.- Abrir la consola de xampp, dentro de la consola de xampp escribir mysql –u root –p y escribir enter, en caso de que tenga contraseña ingresarla.

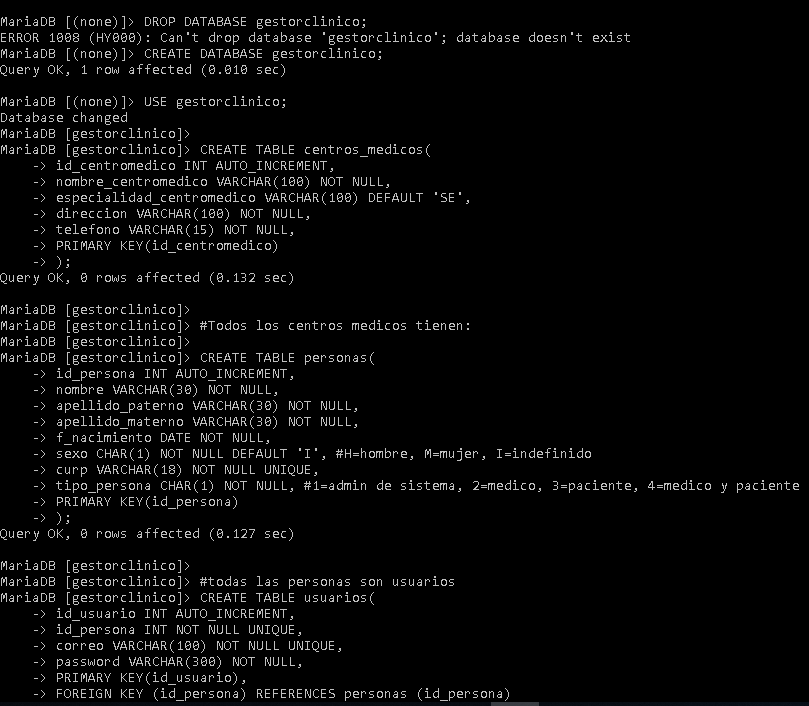


3.- Una vez dentro de MariaDB podemos consultar las bases de datos existentes con el comando show databases;



4.- Verificamos que nuestra base de datos BD\_gestorclinico, y damos paso a crear las tablas a usar dentro del proyecto (en caso de existir la BD la podremos borrar con DROP DATABASE gestorclinico;).

5.- Una vez verificado esto copiamos el contenido del archivo de BD\_gestorclinico.sql para crear la BD

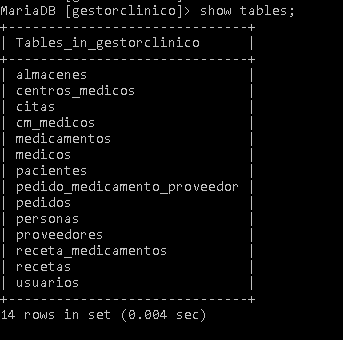


6.- Después de insertar las tablas podremos consultar si se han creado todo de manera correcta con los comandos:

Show databases;

Use nameDB; (use BD\_gestorclinico;)

Show tables; (para mostrar las tablas que se han creado en la Bd)



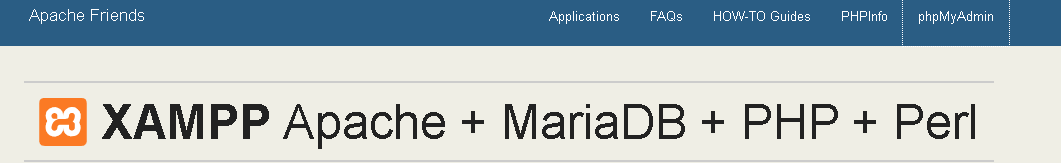
8.- otra forma de agregar la BD es desde el navegador se teclea la ip del servidor local (local host) 127.0.0.1



O escribiendo localhost



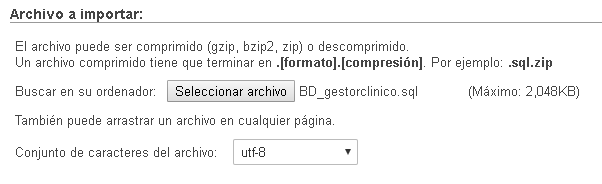
Accedemos al apartado de phpMyadmin



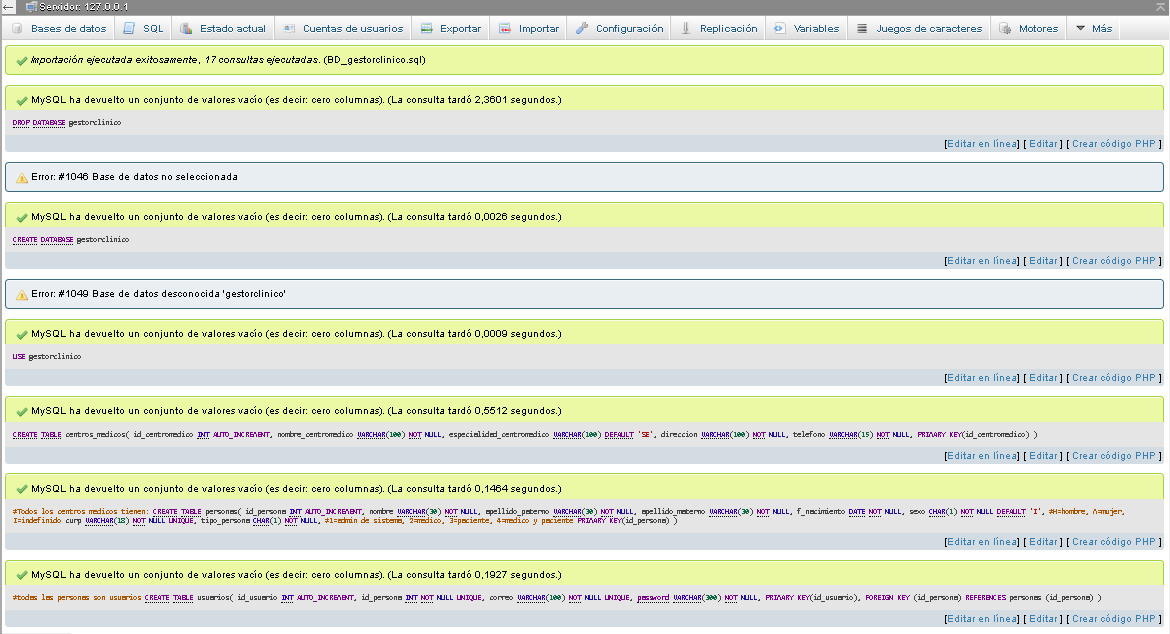
Seleccionamos la opción de importar



Seleccionamos la Bd con la que se trabajara y seleccionamos el cotejamiento utf-8

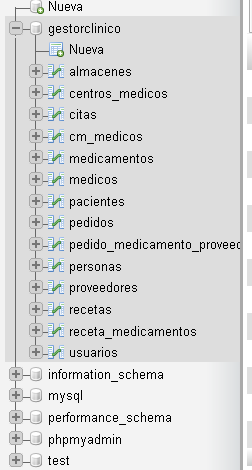


Y le damos en continuar, después de esto se insertara la BD y nos mostrara un pantalla igual a esto



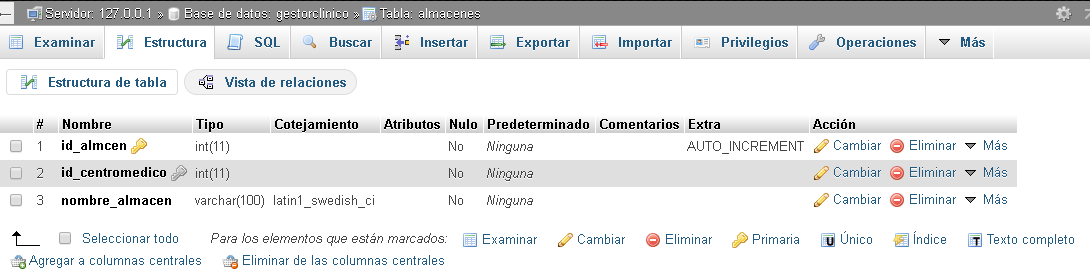
En la cual dirá las tablas que se insertaron de manera exitosa.

Podremos ver las tablas que se generaron seleccionando la BD en el menú que ofrece PhpMyAdmin



Podremos modificar las tablas seleccionado cada una de estas y dando en la opción editar (cambiar).





Para conectar a la BD creamos una clase de tipo conexión

public class Conexion

{

Connection conectar=null;

public Connection conexionbd(){

try{

Class.forName("com.mysql.jdbc.Driver"); //Libreria usada para conectar la base de datos a java

conectar=DriverManager.getConnection("jdbc:mysql://localhost:3306/BD\_gestorclinico", "root", "");

System.out.println("Conexion establecida");

}catch(Exception e){

System.out.println("Error de conexion: "+e.getMessage());

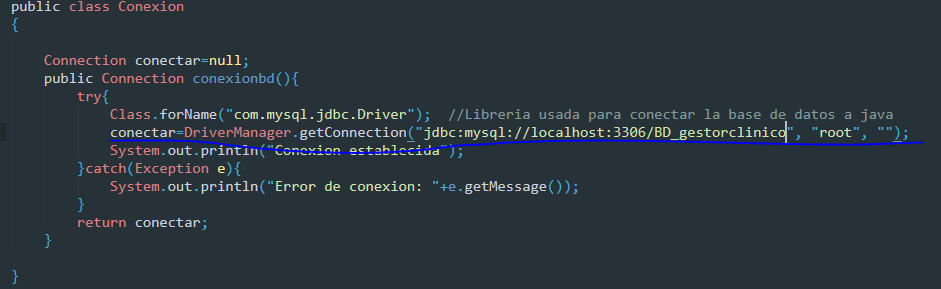
}

return conectar;

}

}

Dentro de esta especificaremos el jar o drive de conexión a usar, el puerto del servidor local y el nombre de la BD, usuario y contraseña de este.



Podemos hacer testeo de que la conexión es correcta, en caso contrario verificar los pasos anteriores.

Y para comenzar con la conexión con la interfaces crearemos archivos de tipo Jframe y se crearan correspondiente a BD.

Para la inserción de a la BD se usara:

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

**conexión e inserción para centros\_medicos**

private void ClaveActionPerformed(java.awt.event.ActionEvent evt)

{

try{

java.sql.Statement at=cn.createStatement();

//ResultSet rs=at.executeQuery("SELECT centros\_medicos FROM centros\_medicos WHERE telefono=(SELECT MAX(telefono) from centros\_medicos)");

ResultSet rs=at.executeQuery("SELECT telefono FROM centros\_medicos WHERE telefono=(SELECT MAX(telefono) from centros\_medicos)");

}catch(SQLException ex){

Logger.getLogger(c.class.getName()).log(Level.SEVERE,null, ex);

}

}

private void AceptarActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event\_AceptarActionPerformed

{

try

{

//INSERT INTO centros\_medicos('nombre\_centromedico','especialidad\_centromedico','direccion','telefono')VALUES('hospital adolfo','cirujia','toluca de lerdo','1234567890');

PreparedStatement pst=cn.prepareStatement("INSERT INTO centros\_medicos(nombre\_centromedico,especialidad\_centromedico,direccion,telefono)VALUES(?,?,?,?)");

pst.setString(1, nombre\_centromedico.getText());

pst.setString(2, especialidad\_centromedico.getText());

pst.setString(3, direccion.getText());

pst.setInt(4, (Integer.parseInt(telefono.getText())));

pst.executeUpdate();

JOptionPane.showMessageDialog(null, "Insercion Correcta");

new c().setVisible(true);

} catch (SQLException ex)

{

try

{

java.sql.Statement at = null;

at = cn.createStatement();

ResultSet rs=at.executeQuery("SELECT telefono FROM centros\_medicos WHERE telefono=(SELECT MAX(telefono) from centros\_medicos)");

JOptionPane.showConfirmDialog(null, "Error de insercion verifique sus datos: "+rs);

} catch (SQLException ex1)

{

Logger.getLogger(a.class.getName()).log(Level.SEVERE, null, ex1);

}

JOptionPane.showMessageDialog(null, "Insercion incorrecta");

}

}

private void CancelarActionPerformed(java.awt.event.ActionEvent evt)

{

nombre\_centromedico.setText("");

especialidad\_centromedico.setText("");

direccion.setText("");

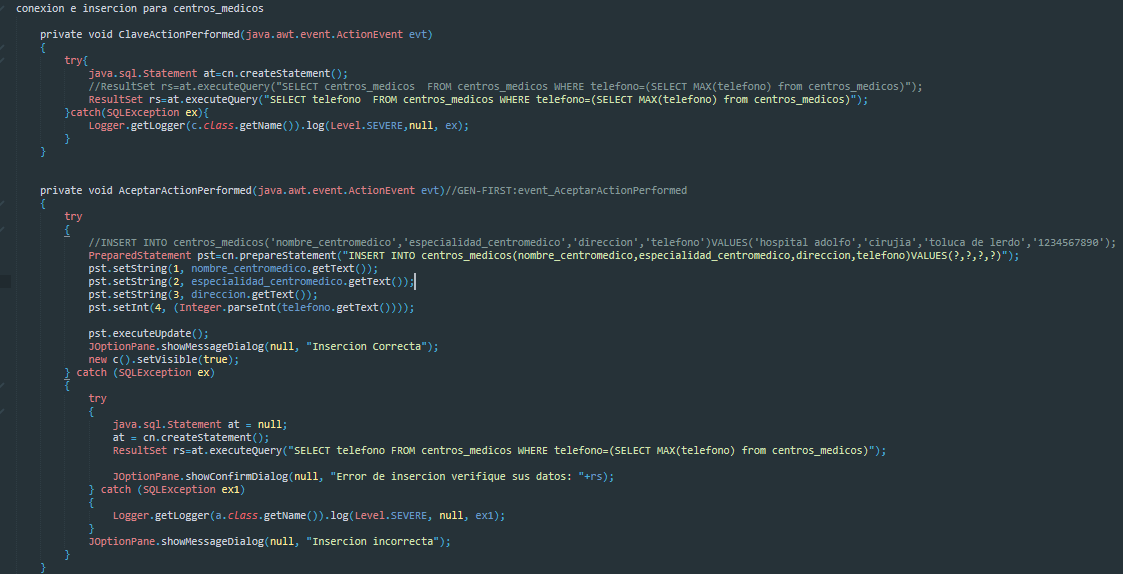
telefono.setText("");

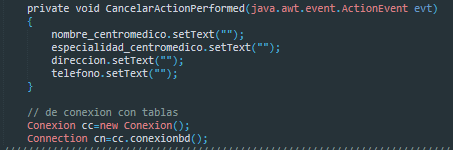
}

// de conexion con tablas

Conexion cc=new Conexion();

Connection cn=cc.conexionbd();



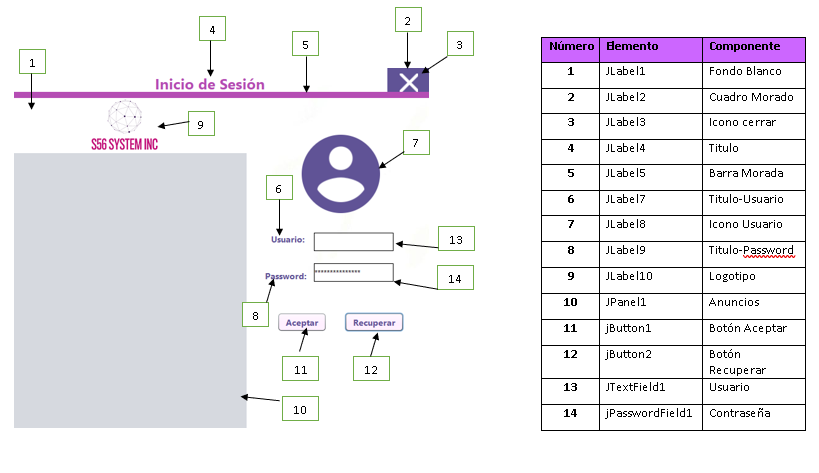


## Interfaces:

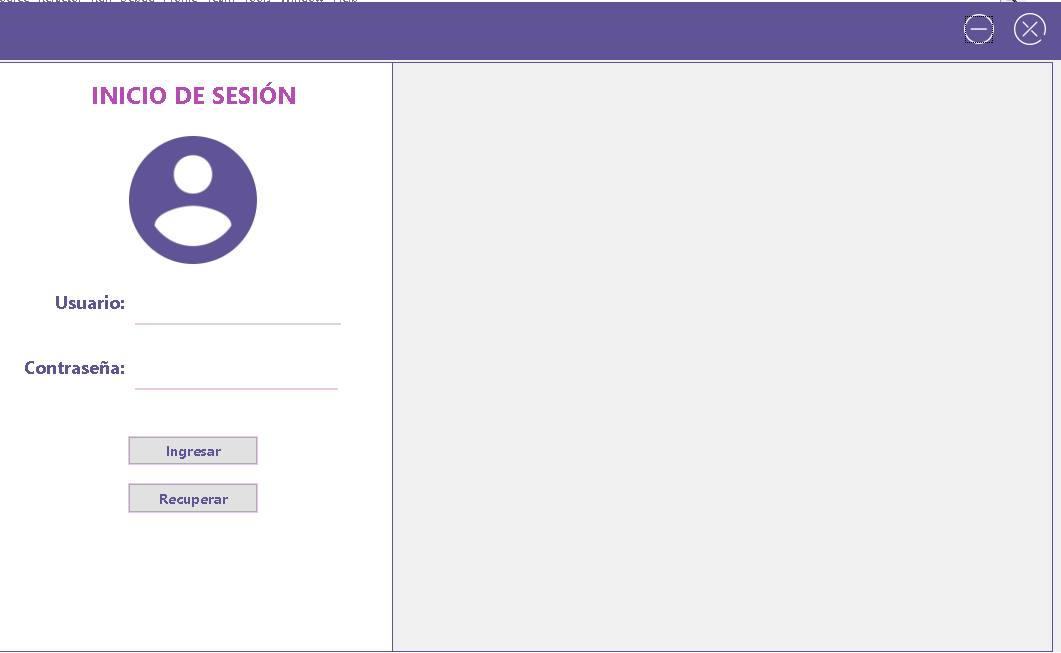
Para las interfaces se eligieron colores fríos, principalmente morado, blanco y rosado.

### Descripción de inicio de sesión:

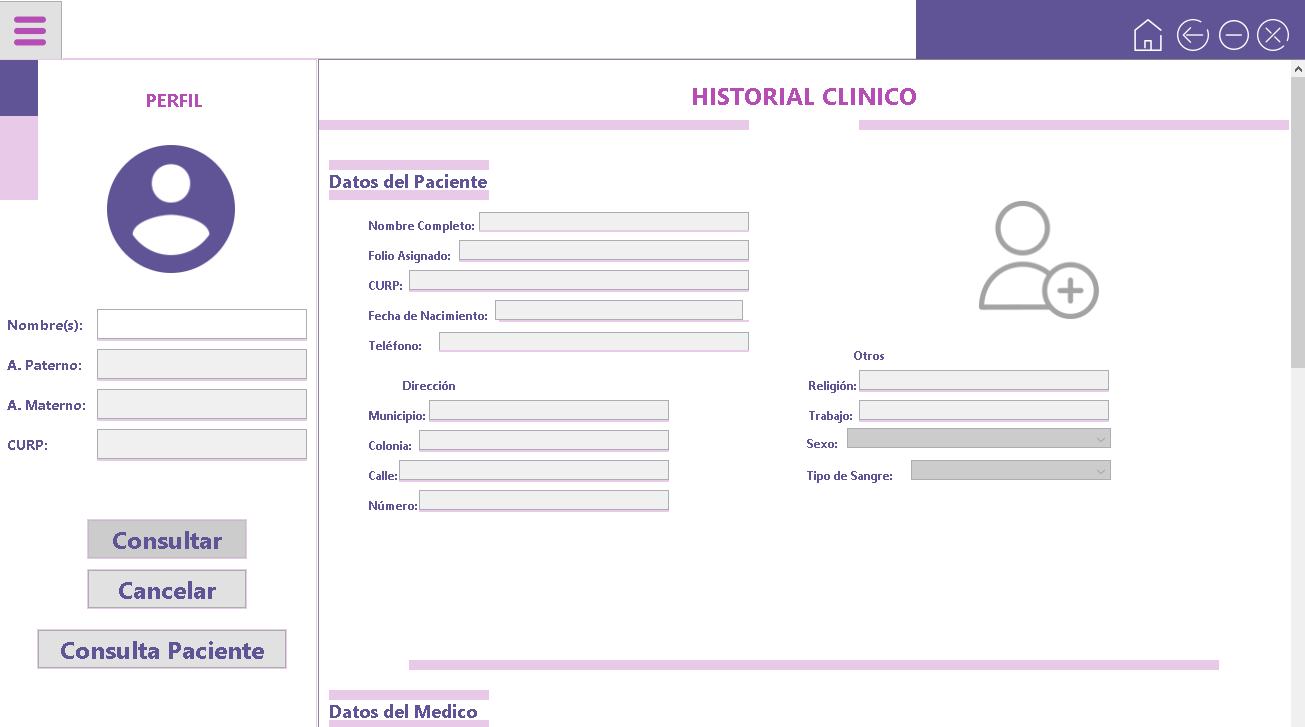


Posteriormente fueron removidos los elementos de la empresa junto con la etiqueta de inicio de sesión:

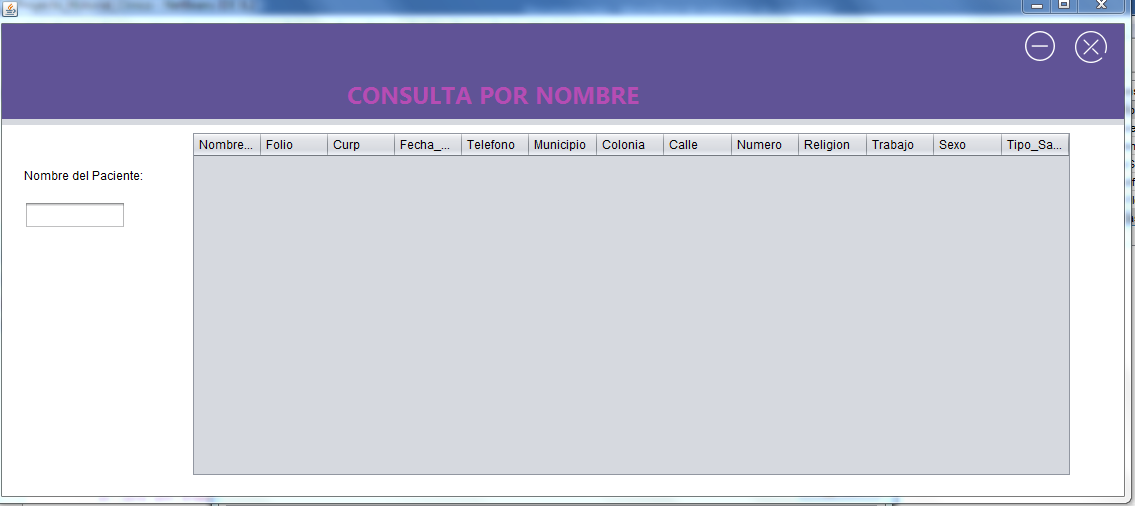


Al pasar la ventana de login, muestra un conjunto de campos que permiten buscar un paciente, y si no se encuntra, darlo de alta:





Se implementa así mismo una ventana de consultas que permite visualizar los datos de los pacientes en una tabla, así mismo, permite buscar por nombre a través de una caja de texto, la cual actualiza la tabla por cada nueva letra añadida.



# Implementación:

**Ventana:ConsultaPaciente**

package Interfaz;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

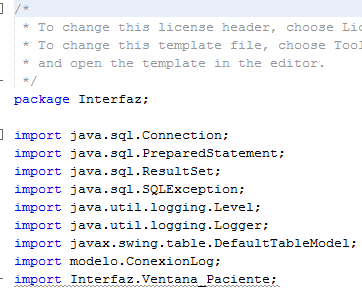
import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.table.DefaultTableModel;

import modelo.ConexionLog;

import Interfaz.Ventana\_Paciente;



public class ConsultaPaciente extends javax.swing.JFrame {

Connection con;

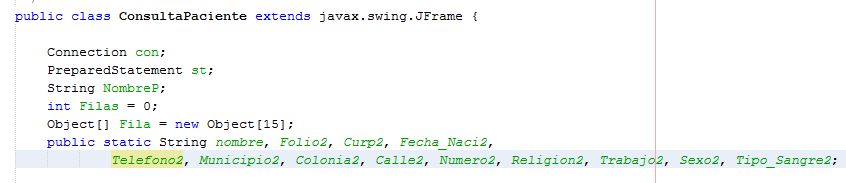
PreparedStatement st;

String NombreP;

int Filas = 0;

Object[] Fila = new Object[15];

public static String nombre, Folio2, Curp2, Fecha\_Naci2, Telefono2, Municipio2, Colonia2, Calle2, Numero2, Religion2, Trabajo2, Sexo2, Tipo\_Sangre2;



private void formWindowOpened(java.awt.event.WindowEvent evt) {

// try {

//Vacía el modelo de la tabla y la reinicia.

DefaultTableModel modelo = (DefaultTableModel) Tabla.getModel();

modelo.setRowCount(0);

Tabla.setModel(modelo);

//Llenar la tabla con las columnas

modelo.addColumn("Nombre\_Completo");

modelo.addColumn("Folio");

modelo.addColumn("Curp");

modelo.addColumn("Fecha\_Naci");

modelo.addColumn("Telefono");

modelo.addColumn("Municipio");

modelo.addColumn("Colonia");

modelo.addColumn("Calle");

modelo.addColumn("Numero");

modelo.addColumn("Religion");

modelo.addColumn("Trabajo");

modelo.addColumn("Sexo");

modelo.addColumn("Tipo\_Sangre");

try {

con = ConexionLog.getConexion();

st = con.prepareStatement("SELECT Nombre\_Completo,Folio,Curp,Fecha\_Naci,Telefono,Municipio,Colonia,Calle,Numero,Religion,Trabajo,Sexo,Tipo\_Sangre FROM datos\_paciente");

ResultSet resultado = st.executeQuery();

while (resultado.next()) {

// Filas++;

String Nombre = resultado.getString("Nombre\_Completo");

Fila[Filas] = Nombre;

Filas++;

String Folio = resultado.getString("Folio");

Fila[Filas] = Folio;

Filas++;

String Curp = resultado.getString("Curp");

Fila[Filas] = Curp;

Filas++;

String Fecha = resultado.getString("Fecha\_Naci");

Fila[Filas] = Fecha;

Filas++;

String Telefono = resultado.getString("Telefono");

Fila[Filas] = Telefono;

Filas++;

String Municipio = resultado.getString("Municipio");

Fila[Filas] = Municipio;

Filas++;

String Colonia = resultado.getString("Colonia");

Fila[Filas] = Colonia;

Filas++;

String Calle = resultado.getString("Calle");

Fila[Filas] = Calle;

Filas++;

String Numero = resultado.getString("Numero");

Fila[Filas] = Numero;

Filas++;

String Religion = resultado.getString("Religion");

Fila[Filas] = Religion;

Filas++;

String Trabajo = resultado.getString("Trabajo");

Fila[Filas] = Trabajo;

Filas++;

String Sexo = resultado.getString("Sexo");

Fila[Filas] = Sexo;

Filas++;

String Tipo\_Sangre = resultado.getString("Tipo\_Sangre");

Fila[Filas] = Tipo\_Sangre;

Filas = 0;

System.out.println("Nombre\_Completo" + Nombre + " Folio " + Folio + " Curp " + Curp + "Fecha\_Naci" + Fecha + " Telefono " + Telefono

+ "Municipio" + Municipio + "Colonia" + Colonia + "Calle" + Calle + "Numero" + Numero + "Religion" + Religion + "Trabajo" + Trabajo + " Sexo " + Sexo + "Tipo\_Sangre" + Tipo\_Sangre);

modelo.addRow(Fila);

Filas = 0;

}

Filas = 0;

Tabla.setModel(modelo);// TODO add your handling code here:

} catch (SQLException ex) {

Logger.getLogger(ConsultaPaciente.class.getName()).log(Level.SEVERE, null, ex);

}}







private void TNombreKeyTyped(java.awt.event.KeyEvent evt) {

DefaultTableModel modelo = (DefaultTableModel) Tabla.getModel();

modelo.setRowCount(0);

Tabla.setModel(modelo);

String Busqueda = TNombre.getText();

try {

con = ConexionLog.getConexion();

st = con.prepareStatement("SELECT Nombre\_Completo,Folio,Curp,Fecha\_Naci,Telefono,Municipio,Colonia,Calle,Numero,Religion,Trabajo,Sexo,Tipo\_Sangre FROM datos\_paciente WHERE Nombre\_Completo LIKE '%" + Busqueda + "%'");

ResultSet resultado = st.executeQuery();

while (resultado.next()) {

// Filas++;

String Nombre = resultado.getString("Nombre\_Completo");

Fila[Filas] = Nombre;

Filas++;

String Folio = resultado.getString("Folio");

Fila[Filas] = Folio;

Filas++;

String Curp = resultado.getString("Curp");

Fila[Filas] = Curp;

Filas++;

String Fecha = resultado.getString("Fecha\_Naci");

Fila[Filas] = Fecha;

Filas++;

int Telefono = resultado.getInt("Telefono");

Fila[Filas] = Telefono;

Filas++;

String Municipio = resultado.getString("Municipio");

Fila[Filas] = Municipio;

Filas++;

String Colonia = resultado.getString("Colonia");

Fila[Filas] = Colonia;

Filas++;

String Calle = resultado.getString("Calle");

Fila[Filas] = Calle;

Filas++;

String Numero = resultado.getString("Numero");

Fila[Filas] = Numero;

Filas++;

String Religion = resultado.getString("Religion");

Fila[Filas] = Religion;

Filas++;

String Trabajo = resultado.getString("Trabajo");

Fila[Filas] = Trabajo;

Filas++;

String Sexo = resultado.getString("Sexo");

Fila[Filas] = Sexo;

Filas++;

String Tipo\_Sangre = resultado.getString("Tipo\_Sangre");

Fila[Filas] = Tipo\_Sangre;

Filas = 0;

System.out.println("Nombre\_Completo" + Nombre + " Folio " + Folio + " Curp " + Curp + "Fecha\_Naci" + Fecha + " Telefono " + Telefono

+ "Municipio" + Municipio + "Colonia" + Colonia + "Calle" + Calle + "Numero" + Numero + "Religion" + Religion + "Trabajo" + Trabajo + "Sexo " + Sexo + "Tipo\_Sangre" + Tipo\_Sangre);

modelo.addRow(Fila);

Filas = 0;

}

Filas = 0;

Tabla.setModel(modelo);// TODO add your handling code here:

} catch (SQLException ex) {

Logger.getLogger(ConsultaPaciente.class.getName()).log(Level.SEVERE, null, ex);

}

}







private void TablaMouseClicked(java.awt.event.MouseEvent evt) {

int fila = Tabla.getSelectedRow();

String dato = Tabla.getValueAt(fila, 0).toString();

try {

con = ConexionLog.getConexion();

PreparedStatement ps;

ps = con.prepareStatement("SELECT Nombre\_Completo,Folio,Curp,Fecha\_Naci,Telefono,Municipio,Colonia,Calle,Numero,Religion,Trabajo,Sexo,Tipo\_Sangre FROM datos\_paciente WHERE Nombre\_Completo LIKE '" + dato + "'");

ResultSet resultado = ps.executeQuery();

if (resultado.next()) {

nombre = resultado.getString("Nombre\_Completo");

Folio2 = resultado.getString("Folio");

Curp2 = resultado.getString("Curp");

Fecha\_Naci2 = resultado.getString("Fecha\_Naci");

Telefono2 = resultado.getString("Telefono");

Municipio2 = resultado.getString("Municipio");

Colonia2 = resultado.getString("Colonia");

Calle2 = resultado.getString("Calle");

Numero2 = resultado.getString("Numero");

Religion2 = resultado.getString("Religion");

Trabajo2 = resultado.getString("Trabajo");

Sexo2 = resultado.getString("Sexo");

Tipo\_Sangre2 = resultado.getString("Tipo\_Sangre");

//Calle,Numero,Religion,Trabajo,Sexo,Tipo\_Sangre

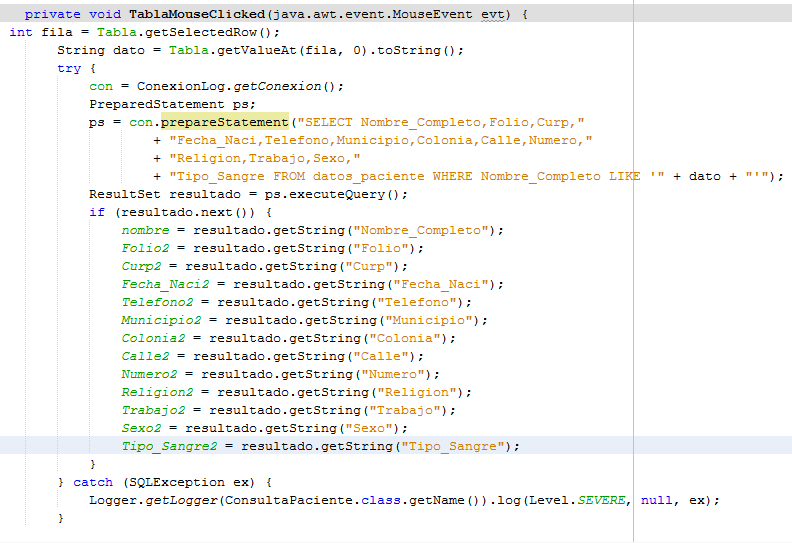
}

} catch (SQLException ex) {

Logger.getLogger(ConsultaPaciente.class.getName()).log(Level.SEVERE, null, ex);

}

}





private void jBC\_PacienteFocusGained(java.awt.event.FocusEvent evt) {

// TODO add your handling code here:

String nombre = ConsultaPaciente.nombre;

jTFNomCompPac.setText(nombre);

String folio = ConsultaPaciente.Folio2;

jTFFolio.setText(folio);

String curp = ConsultaPaciente.Curp2;

jTFCurp2.setText(curp);

String fecha = ConsultaPaciente.Fecha\_Naci2;

jTFFecNac.setText(fecha);

String tel = ConsultaPaciente.Telefono2;

jTFTelefono.setText(tel);

String municipio = ConsultaPaciente.Municipio2;

jTFMunicipio.setText(municipio);

String colonia = ConsultaPaciente.Colonia2;

jTFColonia.setText(colonia);

String calle = ConsultaPaciente.Calle2;

jTFCalle.setText(calle);

String num = ConsultaPaciente.Numero2;

jTFNumero.setText(num);

String religion = ConsultaPaciente.Religion2;

jTFReligion.setText(religion);

String trabajo = ConsultaPaciente.Trabajo2;

jTFTrabajo.setText(trabajo);

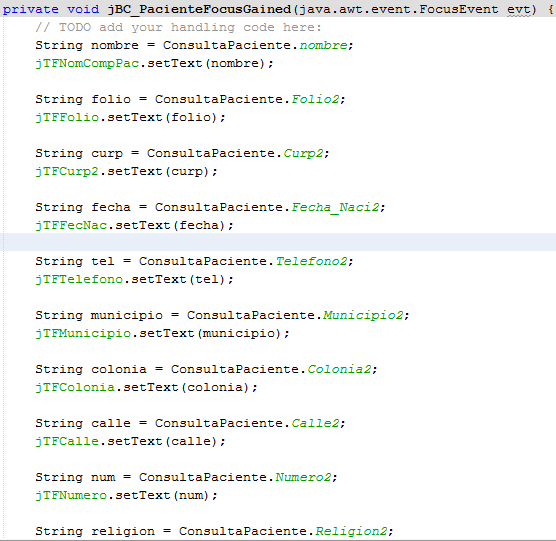
String sexo = ConsultaPaciente.Sexo2;

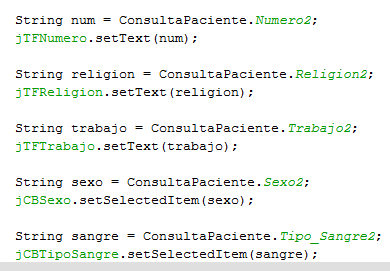
jCBSexo.setSelectedItem(sexo);

String sangre = ConsultaPaciente.Tipo\_Sangre2;

jCBTipoSangre.setSelectedItem(sangre);

}





private void jBC\_PacienteActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

ConsultaPaciente cp = new ConsultaPaciente();

cp.setVisible(true);

}

