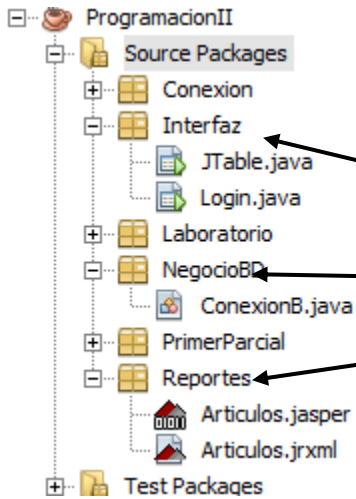


# MANUAL PROGRAMADOR

## Paquetes

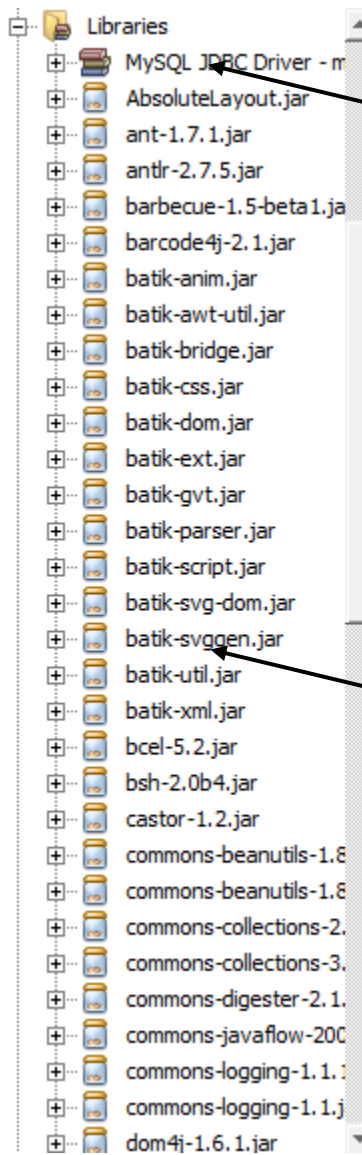


Interfaz: Dentro de este paquete se encuentran los formularios para la app

NegocioBD: incluye el código para poder conectar nuestra base de datos en mysql a netbeans.

Reportes: contiene el diseño de nuestro reporte,

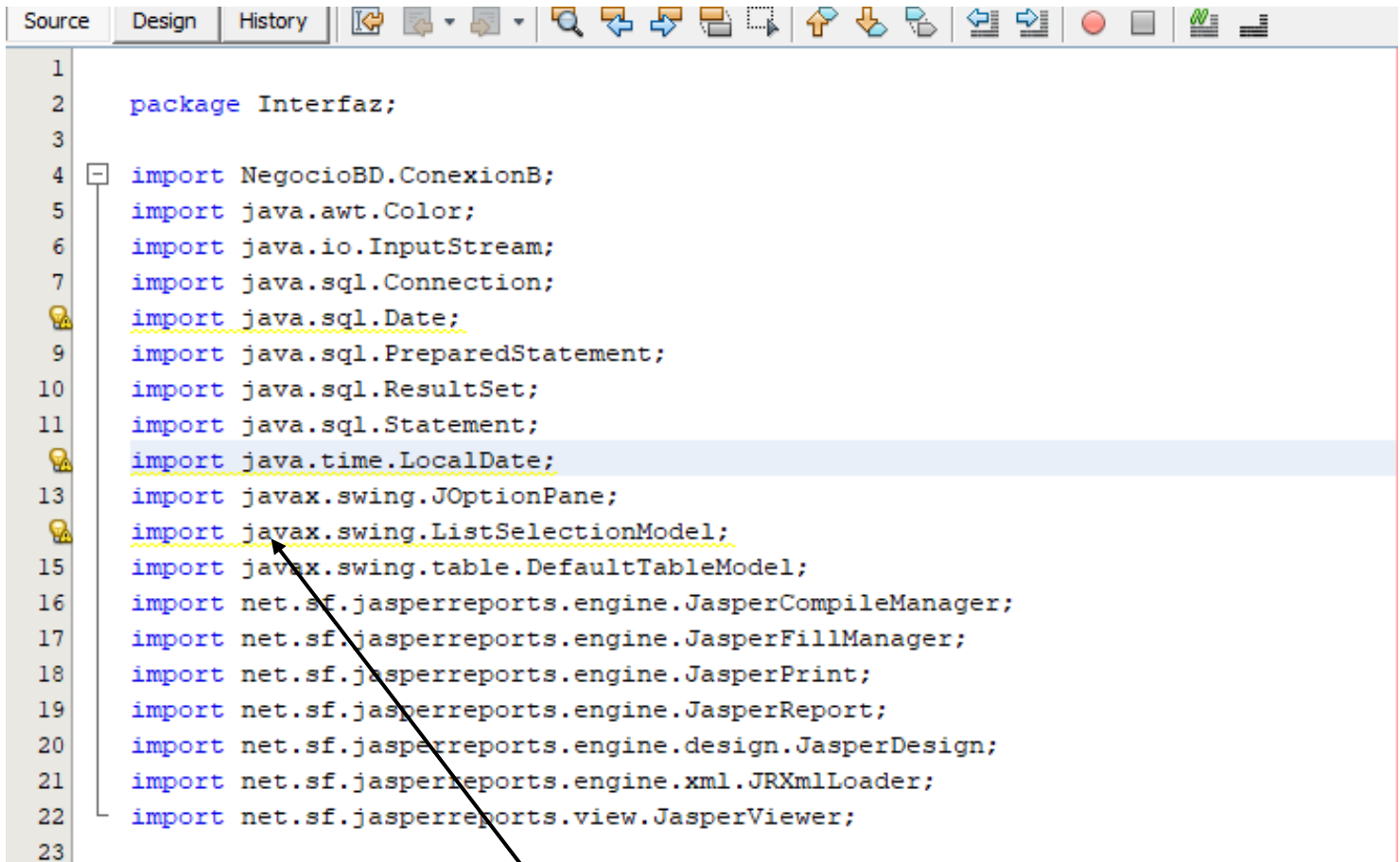
## Librerías



Para que se pueda ejecutar el código de conexión correctamente.

Todas las librerías que nos permite la ejecución correcta de los reportes en pdf

## Código



```
1
2 package Interfaz;
3
4 import NegocioBD.ConexionB;
5 import java.awt.Color;
6 import java.io.InputStream;
7 import java.sql.Connection;
8 import java.sql.Date;
9 import java.sql.PreparedStatement;
10 import java.sql.ResultSet;
11 import java.sql.Statement;
12 import java.time.LocalDate;
13 import javax.swing.JOptionPane;
14 import javax.swing.ListSelectionModel;
15 import javax.swing.table.DefaultTableModel;
16 import net.sf.jasperreports.engine.JasperCompileManager;
17 import net.sf.jasperreports.engine.JasperFillManager;
18 import net.sf.jasperreports.engine.JasperPrint;
19 import net.sf.jasperreports.engine.JasperReport;
20 import net.sf.jasperreports.engine.design.JasperDesign;
21 import net.sf.jasperreports.engine.xml.JRXmlLoader;
22 import net.sf.jasperreports.view.JasperViewer;
23
```

The image shows a code editor window with a toolbar at the top. The code is in a Java file named 'Interfaz.java'. It starts with a package declaration 'package Interfaz;'. Then, there are 19 'import' statements. The first is 'import NegocioBD.ConexionB;'. The next eight are imports from the 'java' package: 'java.awt.Color', 'java.io.InputStream', 'java.sql.Connection', 'java.sql.Date', 'java.sql.PreparedStatement', 'java.sql.ResultSet', 'java.sql.Statement', and 'java.time.LocalDate'. The next two are imports from the 'javax.swing' package: 'JOptionPane' and 'ListSelectionModel'. The last eight are imports from the 'net.sf.jasperreports' package: 'JasperCompileManager', 'JasperFillManager', 'JasperPrint', 'JasperReport', 'JasperDesign', 'JRXmlLoader', and 'JasperViewer'. A callout box with a black border and white background points to the 'import' keyword in the 14th line, stating that this part of the code specifies classes or packages to be used later without including their package names.

Esta parte del código "import" especifica clases o paquetes completos de Java para consultarlos más adelante sin incluir sus nombres de paquete

```
public class JTable extends javax.swing.JFrame {  
    //conexion  
  
    int filas;  
    String cod;  
    ConexionB conl=new ConexionB();  
    Connection conet;  
    DefaultTableModel modelo;  
    Statement st;  
    ResultSet rs;  
    //int idc;  
    PreparedStatement ps;  
  
    public JTable() {  
        initComponents();  
        //para que centre al ejecutar  
        setLocationRelativeTo(null);  
        consultar();  
    }  
}
```

SetLocation nos permite colocar nuestro form en una posición céntrica al ejecutar.

Creación de variables locales

```
*/  
int fila = Tabla.getSelectedRow();  
txtcodigo.setText(Tabla.getValueAt(fila, 0) .toString());  
txtnombre.setText(Tabla.getValueAt(fila, 1) .toString());  
txtpunitario.setText(Tabla.getValueAt(fila, 2) .toString());  
txtcantidad.setText(Tabla.getValueAt(fila, 3) .toString());  
txtfecha.setText(Tabla.getValueAt(fila, 4) .toString());  
filas = fila;  
}
```

Permite que al dar clic a cualquier fila de nuestra tabla, el contenido de la selección sea mostrado en los campos de nuestro formulario.

```

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
    // BUSCAAR
    Connection conet = null;

    try{
        conet= con1.getConnection();
        ps = conet.prepareStatement("SELECT *FROM producto WHERE codigoProducto= ?");
        ps.setString(1, txtcodigo.getText());
        rs= ps.executeQuery();
        if (rs.next()){
            txtcodigo.setText(rs.getString("codigoProducto"));
            txtnombre.setText(rs.getString("nombreProducto"));
            txtpunitario.setText(rs.getString("precioUnitario"));
            txtcantidad.setText(rs.getString("cantidadProducto"));
            txtfecha.setText(rs.getString("fechaVencimiento"));

        } else{
            JOptionPane.showMessageDialog(null,"No existe ese producto");
        }
    } catch (Exception e){
        //System.err.println(e);
    }
}

```

Dentro del botón buscar se inserta el siguiente código que nos permite buscar un producto, el cual se está llamando desde nuestra BD, por lo tanto debemos agregar siempre nuestro conet.

```
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    jButton4.setBackground(Color.GREEN);  
    nuevo();  
}  
  
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {  
    // ELIMINAR  
    eliminar();  
    consultar();  
    nuevo();  
}  
  
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  
    //MODIFICAR  
    modificar();  
    consultar();  
    nuevo();  
}  
  
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    // BOTON AGREGAR  
    agregar();  
    consultar();  
    nuevo();  
}
```

Dentro de cada botón de funciones, se llaman los métodos previamente realizados para que al dar clic este pueda funcionar, el método por sí solo no funcionara al menos que se esté llamando desde el botón correspondiente.

```
void modificar(){
    String cod = txtcodigo.getText();
    String nombre = txtnombre.getText();
    String precio = txtpuntario.getText();
    String cantidad = txtcantidad.getText();
    String fecha = txtfecha.getText();

    try{
        if ( cod.equals("") || nombre.equals("") || precio.equals("") || cantidad.equals("") || fecha.equals("")){
            JOptionPane.showMessageDialog(null , "Faltan ingresar datos");
            limpiarTabla();
        } else{
            String sql= "Update producto set codigoProducto='"+cod+"', nombreProducto='"+nombre+"', precioUnitario='"+precio+"', cantidadProducto='"+cantidad+"', fecha='"+fecha+"'";

            conet= conl.getConnection();
            st= conet.createStatement();
            st.executeUpdate(sql);
            JOptionPane.showMessageDialog(null , "Producto Modificado");
            limpiarTabla();
        }
    } catch(Exception e){
    }
}
```

Se observan las variables previamente declaradas en nuestras cajas de texto, seguidamente de una sentencia de que si algún campo esta vacío este debe ser llenado obligadamente, si estos están llenos correctamente entonces nuestra BD se va a modificar mediante nuestro Update producto set, finalmente se llama a nuestro conet.



```
//METODO CONSULTAR
void consultar(){
    String sql="select * from producto";
    try {
        conet= conl.getConnection();
        st= conet.createStatement();
        rs= st.executeQuery(sql);
        Object [] producto = new Object [5];
        modelo = (DefaultTableModel) Tabla.getModel();
        while (rs.next()) {
            producto [0] = rs.getInt("codigoProducto");
            producto [1] = rs.getString("nombreProducto");
            producto [2] = rs.getFloat("precioUnitario");
            producto [3] = rs.getInt("cantidadProducto");
            producto [4] = rs.getDate("fechaVencimiento");

            modelo.addRow(producto);
        }
        Tabla.setModel(modelo);
    } catch (Exception e){
    }
}
```

El método consultar nos muestra en una tabla los productos desde nuestra BD. Esto mediante "select \* from producto", siempre debemos llamar a nuestra conexión.

```

void agregar() {

    String cod = txtcodigo.getText();
    String nombre = txtnombre.getText();
    String precio = txtpuntario.getText();
    String cantidad = txtcantidad.getText();
    String fecha = txtfecha.getText();

    try{
        if ( cod.equals("") || nombre.equals("") || precio.equals("") || cantidad.equals("") || fecha.equals("") ){
            JOptionPane.showMessageDialog(null , "Faltan ingresar datos");
            limpiarTabla();
        } else{
            String sql= "insert into producto (codigoProducto,nombreProducto,precioUnitario,cantidadProducto,fechaVencimiento)

            conet= conl.getConnection();
            st= conet.createStatement();
            st.executeUpdate(sql);
            JOptionPane.showMessageDialog(null , "Producto registrado");
            limpiarTabla();
        }

    } catch(Exception e){

    }
}

```

Método agregar, la sentencia if identifica que todos los campos deben estar llenos para que el programa siga su secuencia de funcionamiento, luego nuestro else dice que al cumplir con todos los campos llenos se van a guardar los datos dentro de nuestra BD, todo esto mediante el insert into.

```

//METODO LIMPIART
void limpiarTabla(){
    //para no repetir datos anteriores
    for(int i=0; i<= Tabla.getRowCount(); i++){
        modelo.removeRow(i);
        i= i-1;
    }
}
}

```

Permite que no haya datos repetidos.

```

void eliminar() {
    int fila = Tabla.getSelectedRow();
    String cod = Tabla.getValueAt(fila, 0) .toString();
    try{
        PreparedStatement elimi= conet.prepareStatement("DELETE FROM producto WHERE codigoProducto='"+cod+"' ");
        elimi.executeUpdate();

        JOptionPane.showMessageDialog(null, "Eliminado");
    }catch(Exception e){
        JOptionPane.showMessageDialog(null, " No Eliminado");
        limpiarTabla();
    }
}

```

Se realiza a través de un delete from “nombre de la base de datos “where, le indicamos a la bd que se guie por el código de producto y así no eliminar por error a otro,

```

void nuevo() {
    txtcodigo.setText("");
    txtnombre.setText("");
    txtcantidad.setText("");
    txtpunitario.setText("");
    txtfecha.setText("");
    txtcodigo.requestFocus();
}

```

Limpia las cajas de texto y permite agregar un producto nuevo

Realizar el nombramiento de las variables

The screenshot shows a web application interface for 'EVSO CORP' within a development tool. The interface includes a header with the company name and logo, a search section with a 'Buscar' button, and a section for product management with buttons for 'Agregar', 'Modificar', 'Eliminar', 'Nuevo', and 'Reportes'. Below this is a table titled 'Informacion Productos EVSO CORP' with columns for 'Codigo Producto', 'Nombre Producto', 'Precio Unitario', 'Cantidad de producto', and 'Fecha de vencimiento'. Annotations with arrows point to specific elements: one points to the 'Codigo producto' input field, another points to the 'Modificar' button, and a third points to the 'Properties' panel on the right side of the development tool.

Properties

background #0...  
border (No Ex...  
foreground #0...  
selectedIndex 0  
tabLayoutPol...  
tabPlacement ...  
toolTipText ...  
Other Properties  
UIClassID Tab...  
alignmentX 0.5  
alignmentY 0.5  
autoscrolls  
baselineReso...  
componentPo...  
cursor ...  
debugGraphi...  
doubleBuffer  
enabled  
focusCycleR...  
focusTravers...  
focusTravers...  
focusable  
font Cenb...  
inheritsPopu...  
inputVerifier ...  
inact [0, 0...  
(TabbedPanel1 [J...

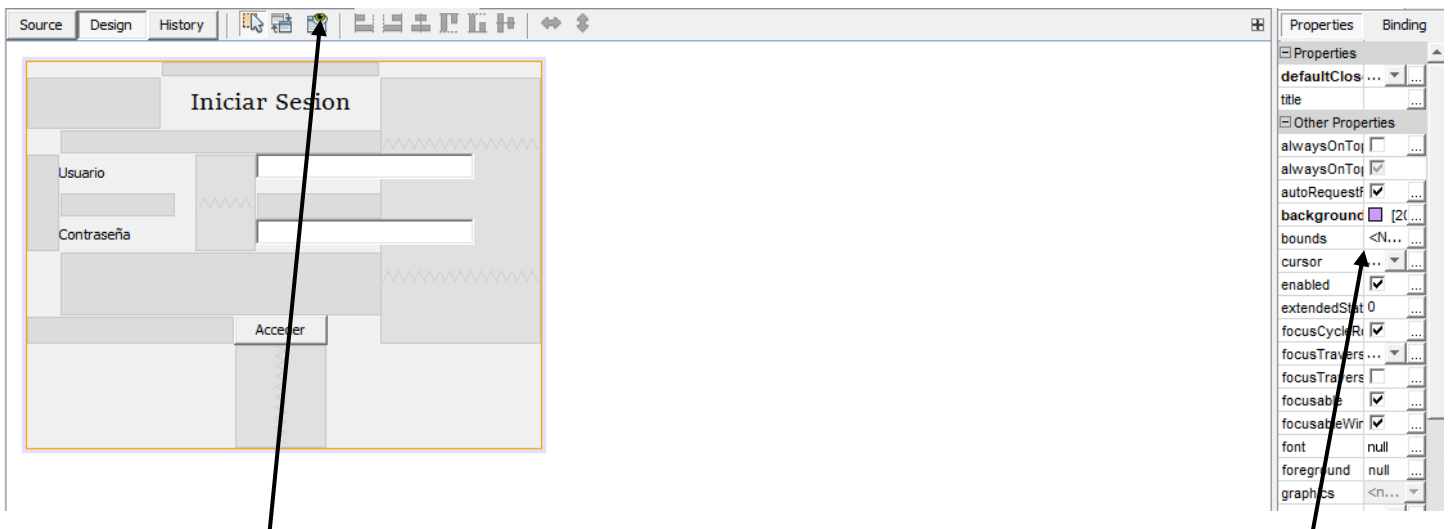
Dentro de los botones se ingresa el código de funcionalidad o se puede llamar algún método

Propiedades y funciones que se pueden agregar al formulario

# Login

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    // INGRESOO  
    String user,pass;  
  
    user =jTFuser.getText();  
    pass=jPFpass.getText();  
  
    if (user.equals("") || pass.equals("")) {  
        JOptionPane.showMessageDialog(this, "Ingresa todos los datos ", "ERROR ", JOptionPane.ERROR_MESSAGE);  
    }else if (user.equals("usuario") || pass.equals("user123")){  
        JOptionPane.showMessageDialog(this, "Acceso concedido");  
        b= new JTable();  
        this.setVisible(false);  
        b.setVisible(true);  
    } else {  
        JOptionPane.showMessageDialog(this, "Datos no coincide","mensaje de error", JOptionPane.ERROR_MESSAGE);  
    }  
}
```

Código que valida al usuario y su password y que llama a la página principal a la cual se desea logear.



Visualizar antes de ejecutar

Propiedades en las cuales se pueden realizar diversos cambios al form

## Conexión

```
package NegocioBD;

import java.sql.Connection;
import java.sql.DriverManager;

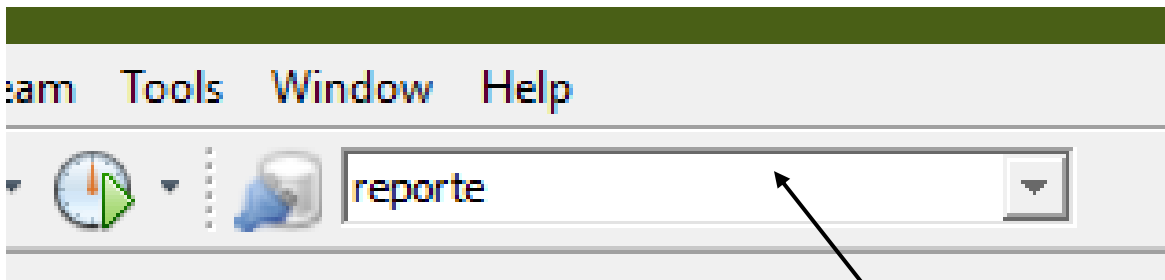
public class ConexionB {
    Connection con;

    public ConexionB() {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            con=(com.mysql.jdbc.Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/bdnegocio","root","");
        } catch (Exception e) {
            System.err.println("Error:" +e);
        }
    }

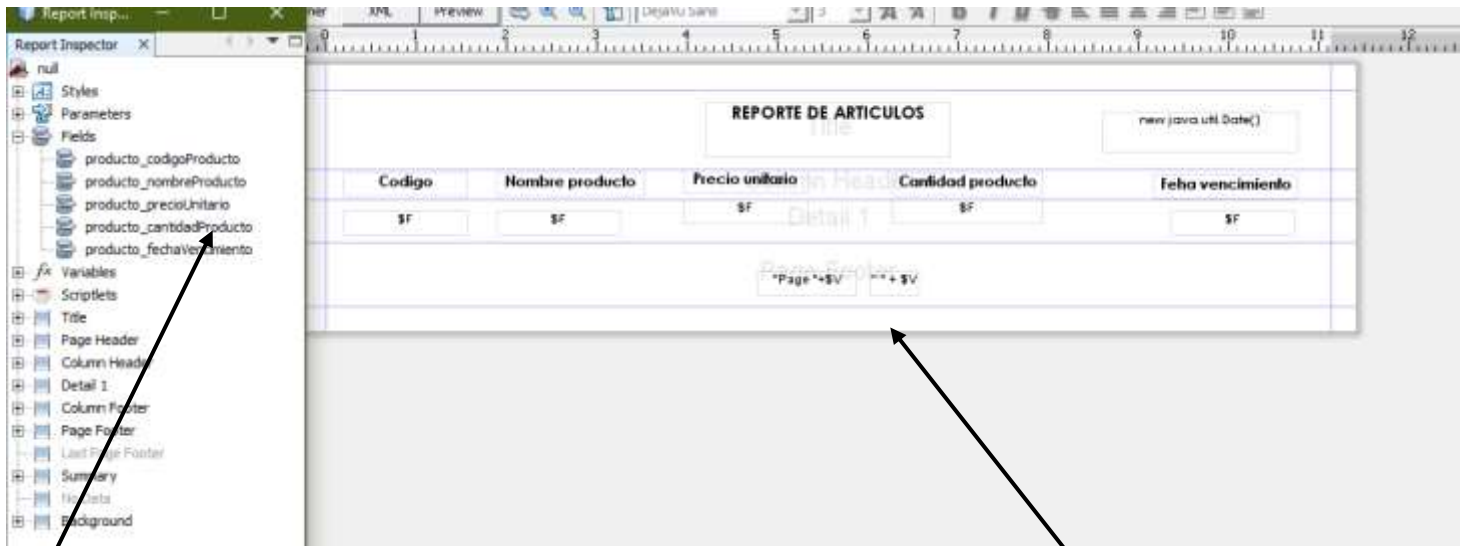
    public Connection getConnection(){
        return con;
    }
}
```

Dentro de este código se agrega la dirección de la bd, para así poder llamarla y asegurarse de que va a conectarse con la aplicación de netbeans.

## Reportes



BD desde la cual se van a llamar los datos que nos mostrara el reporte



Variables llamadas desde la BD, lo cual permite que se puedan insertar en el modelo el reporte

Diseño del reporte

```
private void BtnReporteActionPerformed(java.awt.event.ActionEvent evt) {
    // REPORTE
    try {
        InputStream archivo= getClass().getResourceAsStream("/Reportes/Articulos.jrxml");
        JasperDesign dise= JRXmlLoader.load(archivo);
        JasperReport jr= JasperCompileManager.compileReport(dise);
        JasperPrint jp= JasperFillManager.fillReport(jr,null, conet);
        JasperViewer.viewReport(jp);
    }

    catch (Exception e){

    }

}
```

Código dentro del botón reporte en la página principal, desde el cual se llama nuestro jrxml.

## Base de Datos



Nombre de la base de datos

Tablas que contiene la BD

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	codigoProducto	int(10)			No	Ninguna			Cambiar  Eliminar  Más
<input type="checkbox"/> 2	nombreProducto	varchar(100)	utf8mb4_general_ci		No	Ninguna			Cambiar  Eliminar  Más
<input type="checkbox"/> 3	precioUnitario	float			No	Ninguna			Cambiar  Eliminar  Más
<input type="checkbox"/> 4	cantidadProducto	int(7)			No	Ninguna			Cambiar  Eliminar  Más
<input type="checkbox"/> 5	fechaVencimiento	date			No	Ninguna			Cambiar  Eliminar  Más

↑ ☐ Seleccionar todo Para los elementos que están marcados: Examinar Cambiar Eliminar Primaria Único Índice

Tabla productos en la que se visualiza el contenido con el respectivo tipo de dato y sus atributos correspondientes



#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar  Eliminar  Más
<input type="checkbox"/> 2	Total	decimal(10,0)			No	Ninguna			Cambiar  Eliminar  Más
<input type="checkbox"/> 3	fecha	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()	Cambiar  Eliminar  Más

Ambas tablas muestran su contenido y atributos, y estas tablas cuentan con una entidad relación ya que son utilizadas para generar ventas.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	cod	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar  Eliminar  Más
<input type="checkbox"/> 2	codV	int(11)			No	Ninguna			Cambiar  Eliminar  Más
<input type="checkbox"/> 3	precio	decimal(10,0)			No	Ninguna			Cambiar  Eliminar  Más
<input type="checkbox"/> 4	cantidad	int(11)			No	Ninguna			Cambiar  Eliminar  Más
<input type="checkbox"/> 5	subtotal	decimal(10,0)			No	Ninguna			Cambiar  Eliminar  Más
<input type="checkbox"/> 6	fecha	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()	Cambiar  Eliminar  Más