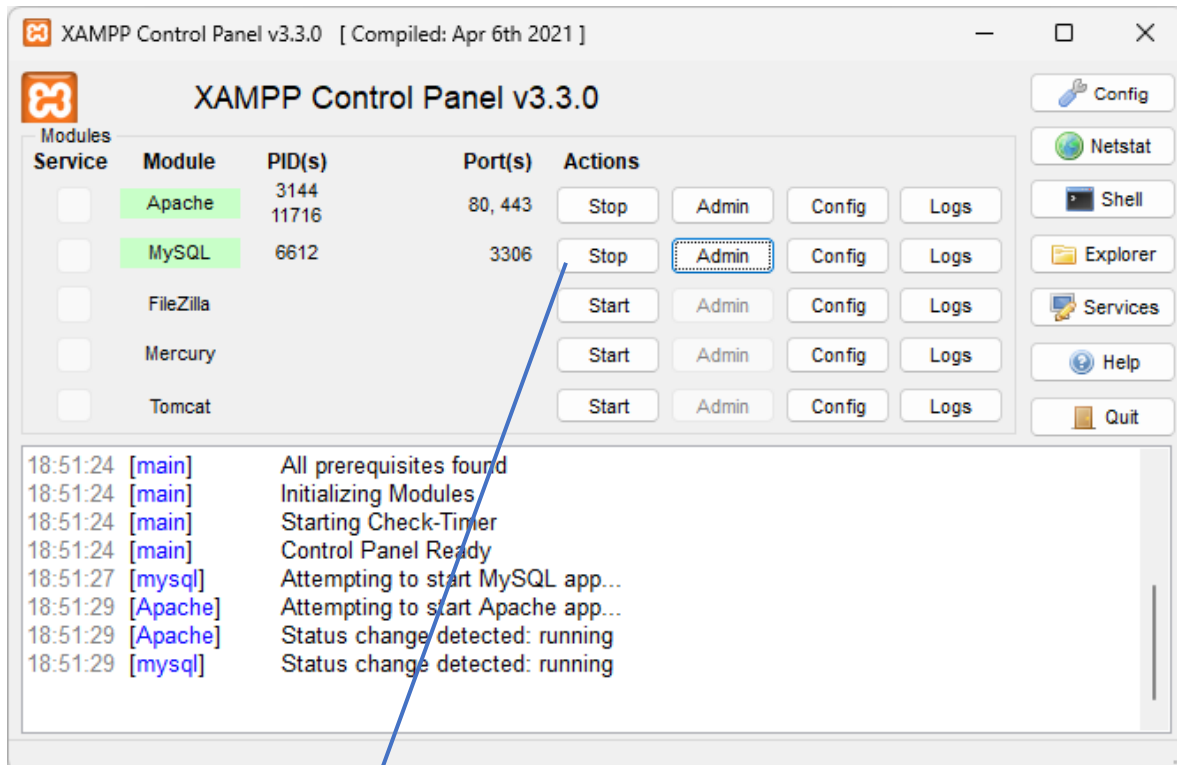


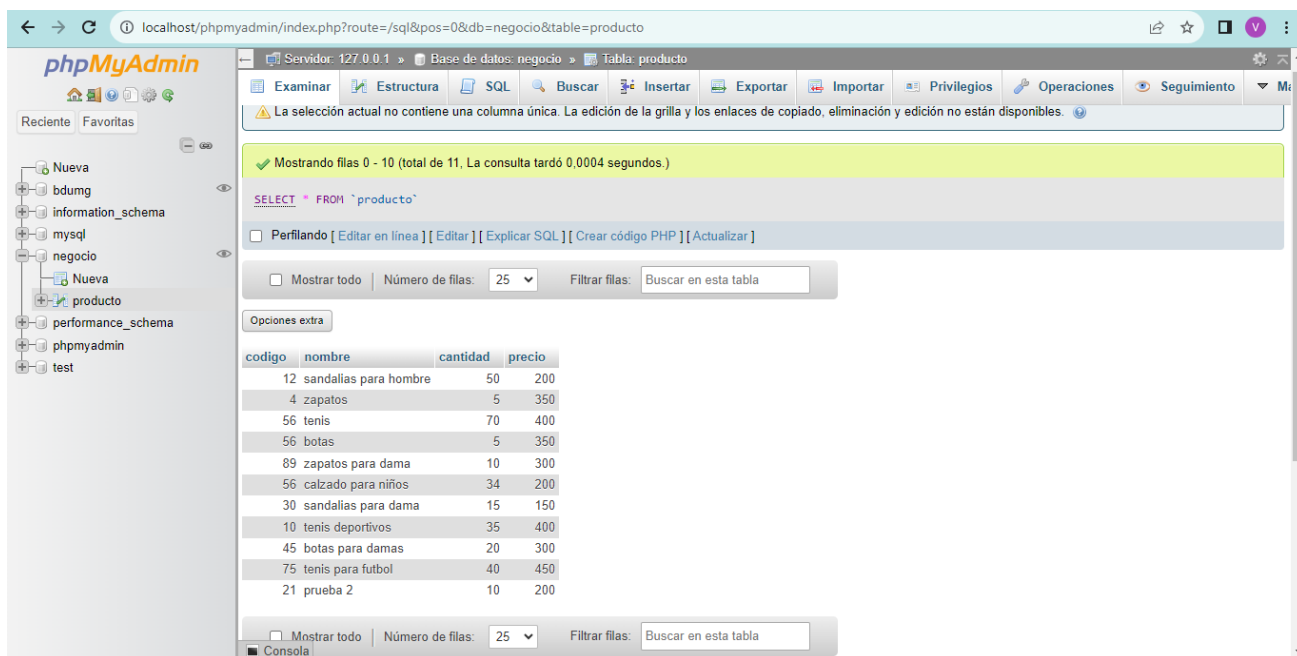
# -MANUAL TECNICO DEL PROYECTO MiZAPATERIA-

Utilizamos XAMPP como conector de Java Netbeans con la Base de Datos MySQL.



INICIALIZAMOS APACHE Y MySQL

Verificar que nuestra Base de Datos funcione bien.



La selección actual no contiene una columna única. La edición de la grilla y los enlaces de copiado, eliminación y edición no están disponibles.

Mostrando filas 0 - 10 (total de 11, La consulta tardó 0,0004 segundos.)

```
SELECT * FROM `producto`
```

Perfilando [ Editar en línea ] [ Editar ] [ Explicar SQL ] [ Crear código PHP ] [ Actualizar ]

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

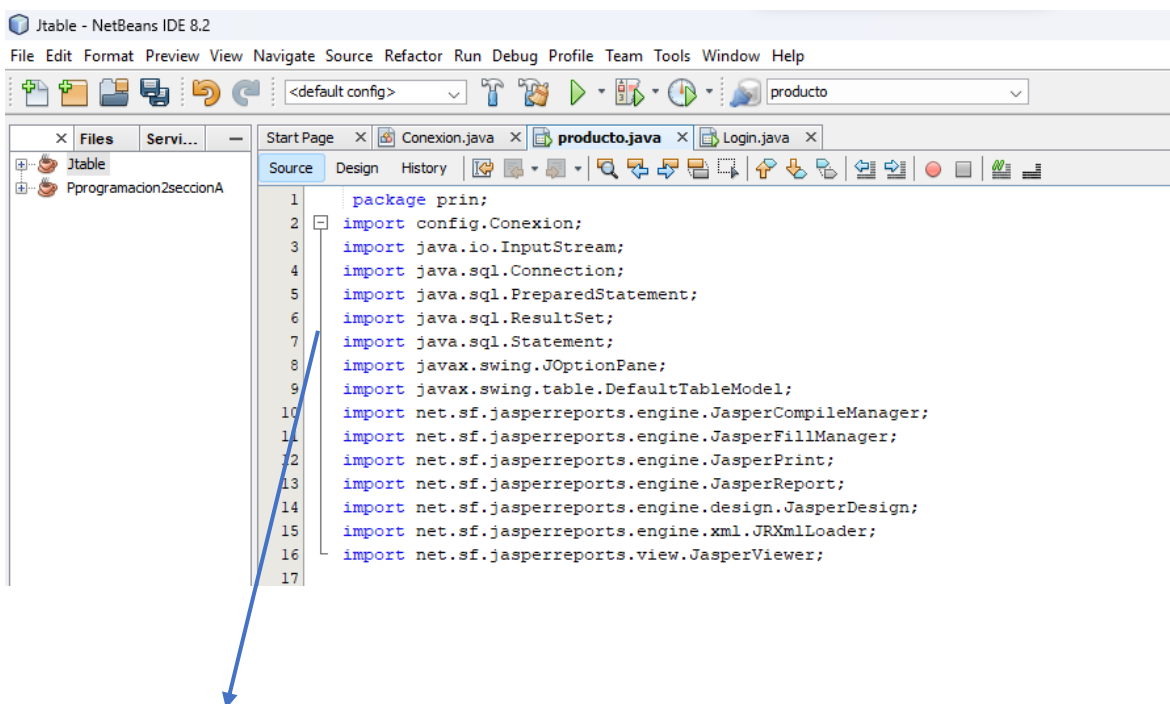
Opciones extra

codigo	nombre	cantidad	precio
12	sandalias para hombre	50	200
4	zapatos	5	350
56	tenis	70	400
56	botas	5	350
89	zapatos para dama	10	300
56	calzado para niños	34	200
30	sandalias para dama	15	150
10	tenis deportivos	35	400
45	botas para damas	20	300
75	tenis para futbol	40	450
21	prueba 2	10	200

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

Consola

## Comenzando con el código



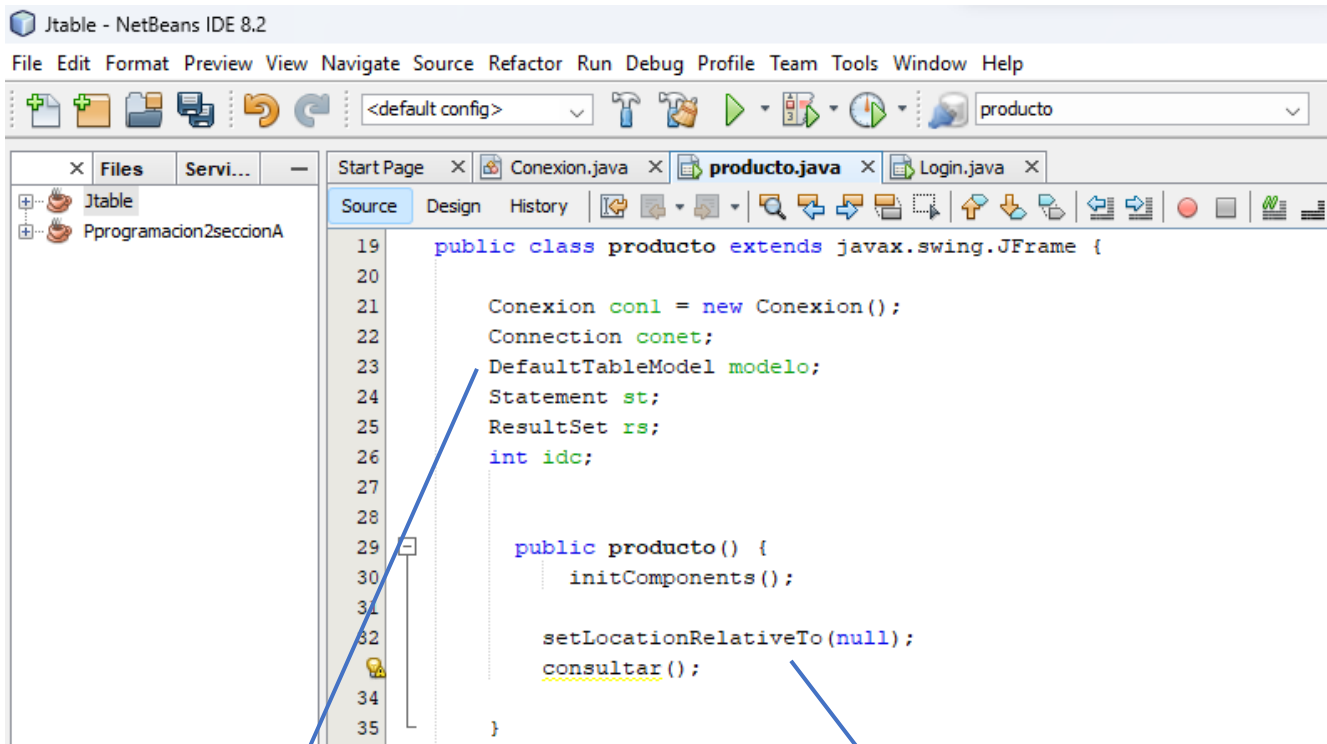
NetBeans IDE 8.2

File Edit Format Preview View Navigate Source Refactor Run Debug Profile Team Tools Window Help

StartPage Conexion.java producto.java Login.java

```
1 package prin;
2 import config.Conexion;
3 import java.io.InputStream;
4 import java.sql.Connection;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import java.sql.Statement;
8 import javax.swing.JOptionPane;
9 import javax.swing.table.DefaultTableModel;
10 import net.sf.jasperreports.engine.JasperCompileManager;
11 import net.sf.jasperreports.engine.JasperFillManager;
12 import net.sf.jasperreports.engine.JasperPrint;
13 import net.sf.jasperreports.engine.JasperReport;
14 import net.sf.jasperreports.engine.design.JasperDesign;
15 import net.sf.jasperreports.engine.xml.JRXmlLoader;
16 import net.sf.jasperreports.view.JasperViewer;
17
```

SE AGREGAN LAS LIBRERIAS QUE SE UTILIZAN DE FORMA UTIL EN TODO EL CODIGO.



DEFINIR LAS VARIABLES LOCALES Y LA CLASE.

-HACE QUE TODAS LAS FUNCIONES  
LOS COMPONENTES EJECUTEN.  
-CENTRA EL PANEL EN PANTALLA.

```
320
321 private void ModificarActionPerformed(java.awt.event.ActionEvent evt) {
322     Modificar();
323     consultar();
324     Nuevo();
325 }
326
327 void Modificar() {
328     String cod = txtCodigo.getText();
329     String nombre = txtNombre.getText();
330     String cantidad = txtCantidad.getText();
331     String precio = txtPrecio.getText();
332
333     try{
334         if ( ((cod.equals("") || nombre.equals("")) || cantidad.equals("")) || precio.equals("")) ){
335             JOptionPane.showMessageDialog(null, "Faltan ingresar datos");
336             limpiarTabla();
337         } else{
338             String sql= "Update producto set codigo='"+cod+"', nombre='"+nombre+"', cantidad='"+cantidad+"', precio='"+precio+"';
339
340             conet= conl.getConnection();
341             st= conet.createStatement();
342             st.executeUpdate(sql);
343             JOptionPane.showMessageDialog(null, " Datos Modificados.");
344             limpiarTabla();
345         }
346     } catch (Exception e) {
347
348     }
349
350 }
```

CODIGO PARA REALIZAR LAS  
MODIFICACIONES EN EL PRODUCTO,  
VINCULADO A CONSULTAR Y NUEVO.

```
Start Page x Conexion.java x producto.java x Login.java x
Source Design History
353
354
355 private void AgregarActionPerformed(java.awt.event.ActionEvent evt) {
356     Agregar();
357     consultar();
358     Nuevo();
359 }
360
361 private void txtCodigoActionPerformed(java.awt.event.ActionEvent evt) {
362     // TODO add your handling code here:
363 }
364
365 private void TablaMouseClicked(java.awt.event.MouseEvent evt) {
366     int fila = Tabla.getSelectedRow();
367     if (fila == -1) {
368         JOptionPane.showMessageDialog(null, "No se seleccionó ninguna fila.");
369     } else {
370         String cod = Tabla.getValueAt(fila, 0).toString();
371         String nombre = (String) Tabla.getValueAt(fila, 1);
372         String cantidad = Tabla.getValueAt(fila, 2).toString();
373         String precio = Tabla.getValueAt(fila, 3).toString();
374
375         txtCodigo.setText(cod);
376         txtNombre.setText(nombre);
377         txtCantidad.setText(cantidad);
378         txtPrecio.setText(precio);
379     }
380
381 }
382
```

ESTO SIRVE PARA QUE AL SELECCIONAR UNA LINEA EN LA BASE DE DATOS, LA INFORMACION SE TRANSFIERA A LOS CAMPOS VACIOS DEL PANEL CON LOS DATOS EXACTOS.

```
Start Page x Conexion.java x producto.java x Login.java x
Source Design History
392
393
394 private void BuscarActionPerformed(java.awt.event.ActionEvent evt) {
395     // BUSCAR
396     Connection conet = null;
397
398
399     try{
400         conet= con1.getConnection();
401         PreparedStatement ps = conet.prepareStatement("SELECT *FROM producto WHERE codigo= ?");
402         ps.setString(1, txtCodigo.getText());
403         rs= ps.executeQuery();
404         if (rs.next()){
405             txtCodigo.setText(rs.getString("Codigo"));
406             txtNombre.setText(rs.getString("Nombre"));
407             txtCantidad.setText(rs.getString("Cantidad"));
408             txtPrecio.setText(rs.getString("Precio"));
409
410
411         } else{
412             JOptionPane.showMessageDialog(null,"No existe ese producto");
413         }
414     } catch(Exception e){
415         //System.err.println(e);
416
417     }
418 }
```

BUSQUEDA EN LA BASE DE DATOS  
CON EL CODIGO DEL PRODUCTO.

```

420 private void ReporteActionPerformed(java.awt.event.ActionEvent evt) {
421     try {
422         InputStream archivo= getClass().getResourceAsStream("/Reportes/Articulos.jrxml");
423         JasperDesign dise= JRXmlLoader.load(archivo);
424         JasperReport jr= JasperCompileManager.compileReport(dise);
425         JasperPrint jp= JasperFillManager.fillReport(jr,null, conet);
426         JasperViewer.viewReport(jp);
427     }
428     catch (Exception e){
429     }
430 }
431
432
433

```

PARA GENERAR EL REPORTE EN PDF.

```

433 private void btnSalirActionPerformed(java.awt.event.ActionEvent evt) {
434
435     int opcion = JOptionPane.showConfirmDialog(this, "¿Desea salir del programa?", "Confirmar salida", JOptionPane.YES_NO_OPTION);
436
437     if (opcion == JOptionPane.YES_OPTION) {
438         // Cierra la aplicación
439         System.exit(0);
440     }
441 }
442
443

```

BOTON PARA SALIR DE FORMA SEGURA DEL PROGRAMA.

```

480 void consultar() {
481     String sql= "select * from producto";
482     try {
483         conet= con1.getConnection();
484         st= conet.createStatement();
485         rs= st.executeQuery(sql);
486         Object [] producto = new Object [4];
487         modelo = (DefaultTableModel) Tabla.getModel();
488         while (rs.next()) {
489             producto [0] = rs.getDouble("codigo");
490             producto [1] = rs.getString ("nombre");
491             producto [2] = rs.getDouble("cantidad");
492             producto [3] = rs.getFloat("Precio");
493
494
495             modelo.addRow(producto);
496         }
497         Tabla.setModel(modelo);
498     } catch (Exception e){
499
500     }
501 }

```

METODO CONSULTA QUE VINCULA  
CON ELIMINAR, NUEVO,  
MODIFICAR Y AGREGAR.



```
Source Design History
502
503 void Agregar() {
504     String cod = txtCodigo.getText();
505     String nombre = txtNombre.getText();
506     String cantidad = txtCantidad.getText();
507     String precio = txtPrecio.getText();
508
509     try{
510         if ( ((cod.equals("") || nombre.equals("")) || cantidad.equals("")) || precio.equals("")) ){
511             JOptionPane.showMessageDialog(null , "Faltan ingresar datos");
512             limpiarTabla();
513
514         } else{
515             String sql= "insert into producto (codigo,nombre,cantidad,precio) values ('"+cod+"','"+nombre+"','"+cantidad+"','"+precio+"')";
516
517             conet= con1.getConnection();
518             st= conet.createStatement();
519             st.executeUpdate(sql);
520             JOptionPane.showMessageDialog(null , "Producto registrado");
521             limpiarTabla();
522         }
523     } catch (Exception e){
524
525     }
526
527
528
529 }
```

BOTON AGREGAR NUEVO PRODUCTO

```
Source Design History
void limpiarTabla() {
    DefaultTableModel model = (DefaultTableModel) Tabla.getModel();
    model.setRowCount(0); // Elimina todas las filas de la tabla
}

//ELIMINAR.
void Eliminar() {
    String cod = txtCodigo.getText();

    if (cod.isEmpty()) {
        JOptionPane.showMessageDialog(null, "Ingrese un código válido para eliminar un producto.");
    } else {
        try {
            String sql = "DELETE FROM producto WHERE codigo = '" + cod + "'";
            conet = con1.getConnection();
            st = conet.createStatement();
            st.executeUpdate(sql);
            JOptionPane.showMessageDialog(null, "Producto eliminado correctamente.");
            limpiarTabla();
        } catch (Exception e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(null, "Error al eliminar el producto.");
        }
    }
}
```

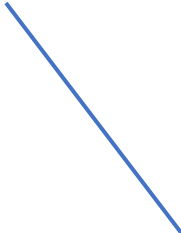
METODO ELIMINAR PRODUCTO

ELIMINA TODAS LAS FILAS DE LA TABLA.

```

558
559 void Nuevo() {
560     txtCodigo.setText("");
561     txtNombre.setText("");
562     txtCantidad.setText("");
563     txtPrecio.setText("");
564     txtCodigo.requestFocus();
565 }

```




BOTON NUEVO.

```

1 package config;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6
7 public class Conexion {
8     Connection con;
9
10    public Conexion() {
11        try {
12            Class.forName("com.mysql.jdbc.Driver");
13            con=(com.mysql.jdbc.Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/negocio","root","");
14        } catch (ClassNotFoundException | SQLException e) {
15            System.err.println("Error:" + e);
16        }
17    }
18
19
20
21    public Connection getConnection() {
22        return con;
23    }
24
25 }
26

```



CONEXIÓN CON LA BASE DE DATOS MySQL

```
Start Page x Conexion.java x producto.java x Login.java x
Source Design History
Generated Code
26
27 @SuppressWarnings("unchecked")
28
101
102 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
103     String user, pass;
104
105     user = jTextField1.getText();
106     pass = jPasswordField1.getText();
107
108     if (user.equals("") || pass.equals("")) {
109         JOptionPane.showMessageDialog(this, "Debe llenar los campos necesario", "Mensaje de error", JOptionPane.ERROR_MESSAGE);
110     } else if (user.equals("usuario") || pass.equals("123456789")) {
111         JOptionPane.showMessageDialog(this, "Acceso Concedido");
112         b = new producto();
113         this.setVisible(false);
114         b.setVisible(true);
115     } else {
116         JOptionPane.showMessageDialog(this, "Los datos no coinciden, ACCESO DENEGADO", "Mensaje de error", JOptionPane.ERROR_MESSAGE);
117     }
118 }
119
```

CODIGO DEL LOGIN Y PASSWORD

Start Page x Conexion.java x producto.java x Login.java x Articulos.jrxml x

Designer XML Preview DejaVu Sans 3

REPOTE DE LOS ARTICULOS			
CODIGO	NOMBRE	CANTIDAD	PRECIO
\$F{producto_codigo}	\$F	\$F	\$F{producto_precio}
"Page "+\$V " "+ \$V			

PANEL DE EDICION DE REPORTES