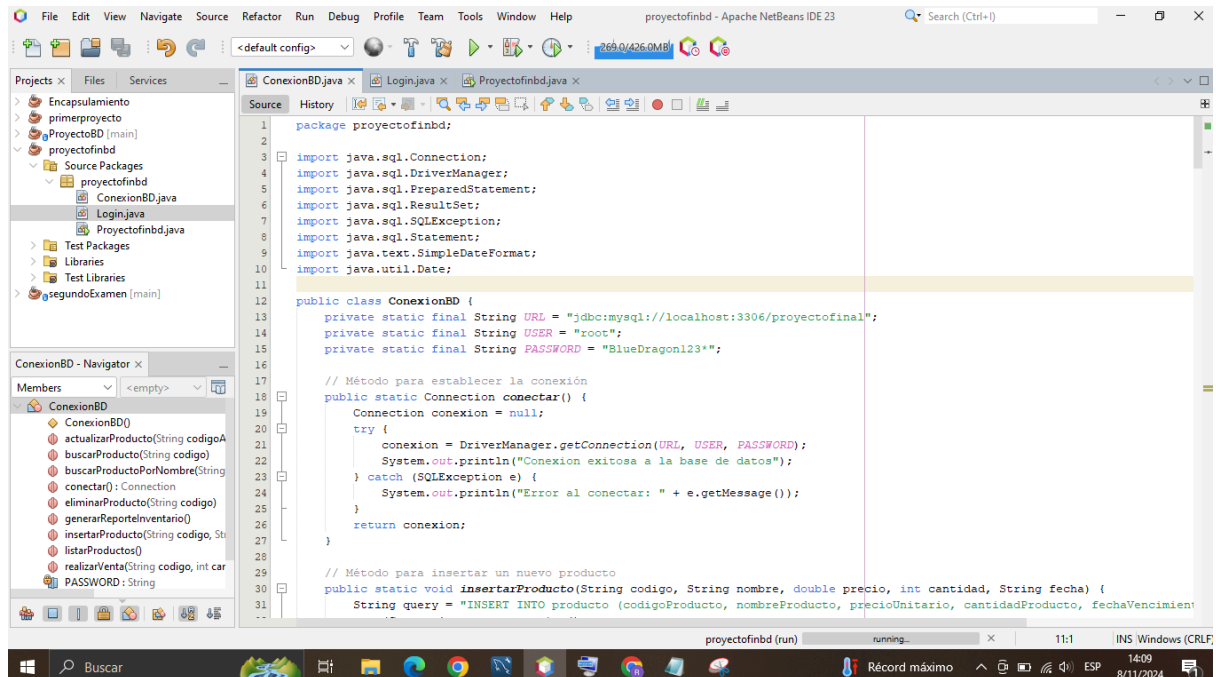
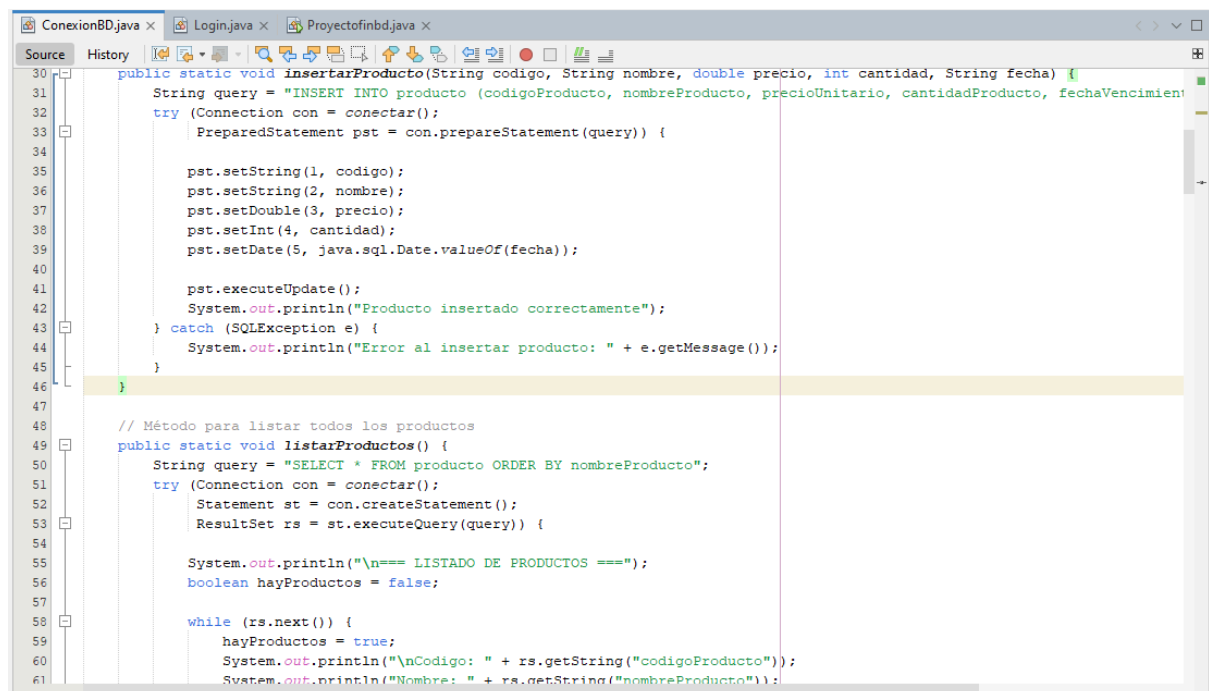


CLASE 1 ConexionBD



```
1 package proyectofinbd;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import java.sql.SQLException;
8 import java.sql.Statement;
9 import java.text.SimpleDateFormat;
10 import java.util.Date;
11
12 public class ConexionBD {
13     private static final String URL = "jdbc:mysql://localhost:3306/proyectofinal";
14     private static final String USER = "root";
15     private static final String PASSWORD = "BlueDragon123*";
16
17     // Método para establecer la conexión
18     public static Connection conectar() {
19         Connection conexion = null;
20         try {
21             conexion = DriverManager.getConnection(URL, USER, PASSWORD);
22             System.out.println("Conexión exitosa a la base de datos");
23         } catch (SQLException e) {
24             System.out.println("Error al conectar: " + e.getMessage());
25         }
26         return conexion;
27     }
28
29     // Método para insertar un nuevo producto
30     public static void insertarProducto(String codigo, String nombre, double precio, int cantidad, String fecha) {
31         String query = "INSERT INTO producto (codigoProducto, nombreProducto, precioUnitario, cantidadProducto, fechaVencimiento) VALUES (" +
32             codigo + ", " + nombre + ", " + precio + ", " + cantidad + ", " + fecha + ")";
33     }
34 }
```



```
30 public static void insertarProducto(String codigo, String nombre, double precio, int cantidad, String fecha) {
31     String query = "INSERT INTO producto (codigoProducto, nombreProducto, precioUnitario, cantidadProducto, fechaVencimiento) VALUES (" +
32         codigo + ", " + nombre + ", " + precio + ", " + cantidad + ", " + fecha + ")";
33     try (Connection con = conectar();
34         PreparedStatement pst = con.prepareStatement(query)) {
35         pst.setString(1, codigo);
36         pst.setString(2, nombre);
37         pst.setDouble(3, precio);
38         pst.setInt(4, cantidad);
39         pst.setDate(5, java.sql.Date.valueOf(fecha));
40
41         pst.executeUpdate();
42         System.out.println("Producto insertado correctamente");
43     } catch (SQLException e) {
44         System.out.println("Error al insertar producto: " + e.getMessage());
45     }
46 }
47
48 // Método para listar todos los productos
49 public static void listarProductos() {
50     String query = "SELECT * FROM producto ORDER BY nombreProducto";
51     try (Connection con = conectar();
52         Statement st = con.createStatement();
53         ResultSet rs = st.executeQuery(query)) {
54
55         System.out.println("\n== LISTADO DE PRODUCTOS ==");
56         boolean hayProductos = false;
57
58         while (rs.next()) {
59             hayProductos = true;
60             System.out.println("\nCodigo: " + rs.getString("codigoProducto"));
61             System.out.println("Nombre: " + rs.getString("nombreProducto"));
62         }
63     }
64 }
```

The image shows a screenshot of an IDE with a Java file named `ProyectoFinbd.java` open. The code is for a database application and includes the following logic:

- Lines 61-65: Print product details (Nombre, Precio, Cantidad, Fecha de Vencimiento) using `rs.getString`, `rs.getDouble`, `rs.getInt`, and `rs.getDate`.
- Lines 66-70: A closing brace for a loop or block.
- Lines 71-75: A `catch` block for `SQLException` with a message "Error al listar productos".
- Lines 76-77: A comment: `// Método para buscar producto por código`.
- Lines 78-79: A public static method `buscarProducto(String codigo)`.
- Lines 80-81: A query string: `"SELECT * FROM producto WHERE codigoProducto = ?";`.
- Lines 82-84: Database connection and query execution logic using `Connection`, `PreparedStatement`, and `ResultSet`.
- Lines 85-92: A loop to iterate through the results, printing each product's details.

The IDE interface includes a toolbar at the top, a sidebar on the left with "Source" and "History" tabs, and a status bar at the bottom showing "proyectoFinbd (run)" and "running...".

```
123         System.out.println("No se encontraron productos con ese nombre.");
124     }
125     } catch (SQLException e) {
126         System.out.println("Error al buscar producto: " + e.getMessage());
127     }
128 }
129
130 // Método para actualizar producto
131 // Método para actualizar producto
132 public static void actualizarProducto(String codigoActual, String nuevoCodigo, String nombre, double precio, int cantidad) {
133     String query = "UPDATE producto SET codigoProducto = ?, nombreProducto = ?, precioUnitario = ?, cantidadProducto = ? WHERE";
134     try (Connection con = conectar();
135         PreparedStatement pst = con.prepareStatement(query)) {
136
137         pst.setString(1, nuevoCodigo);
138         pst.setString(2, nombre);
139         pst.setDouble(3, precio);
140         pst.setInt(4, cantidad);
141         pst.setString(5, codigoActual);
142
143         int filasActualizadas = pst.executeUpdate();
144         if (filasActualizadas > 0) {
145             System.out.println("Producto actualizado correctamente");
146         } else {
147             System.out.println("No se encontro el producto con el codigo especificado");
148         }
149     } catch (SQLException e) {
150         System.out.println("Error al actualizar producto: " + e.getMessage());
151     }
152 }
153
154 // Método para eliminar producto
```

projectofinbd (run) running... x 114:37 INS Windows (CRLF)

ConexionBD.java Login.java Projectofinbd.java

Source History

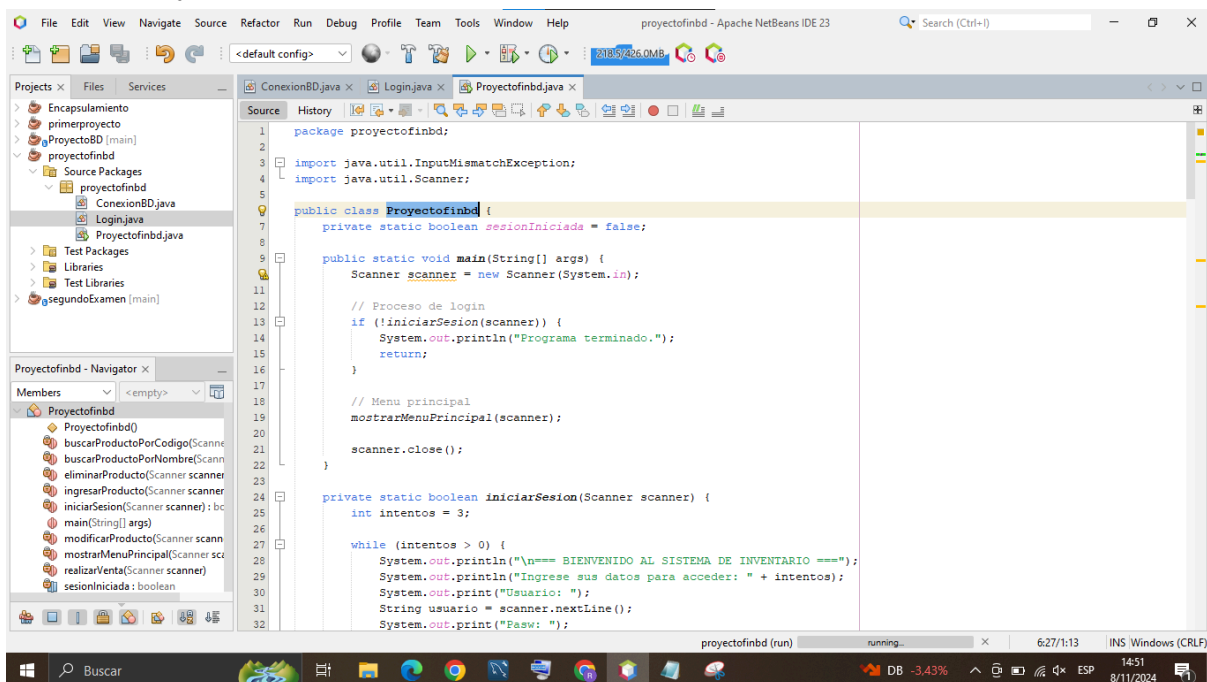
```
160         pst.setString(1, codigo);
161
162         int filasEliminadas = pst.executeUpdate();
163         if (filasEliminadas > 0) {
164             System.out.println("Producto eliminado correctamente");
165         } else {
166             System.out.println("No se encontró el producto con el codigo especificado");
167         }
168     } catch (SQLException e) {
169         System.out.println("Error al eliminar producto: " + e.getMessage());
170     }
171 }
172
173 // Método para realizar una venta
174 public static void realizarVenta(String codigo, int cantidadVenta) {
175     Connection con = null;
176     try {
177         con = conectar();
178         con.setAutoCommit(false); // Inicio de la transacción
179
180         // Verificar existencia y stock
181         String queryStock = "SELECT cantidadProducto, precioUnitario, nombreProducto FROM producto WHERE codigoProducto = ";
182         PreparedStatement pstStock = con.prepareStatement(queryStock);
183         pstStock.setString(1, codigo);
184         ResultSet rs = pstStock.executeQuery();
185
186         if (rs.next()) {
187             int stockActual = rs.getInt("cantidadProducto");
188             double precio = rs.getDouble("precioUnitario");
189             String nombreProducto = rs.getString("nombreProducto");
190
191             if (stockActual >= cantidadVenta) {
```

```
ConexionBD.java x Login.java x Proyectofinbd.java x
Source History
194 PreparedStatement pstupdate = con.prepareStatement(updatestock);
195
196 pstUpdate.setInt(1, cantidadVenta);
197 pstUpdate.setString(2, codigo);
198 pstUpdate.executeUpdate();
199
200 // Calcular total
201 double total = precio * cantidadVenta;
202
203 // Imprimir factura
204 System.out.println("\n===== FACTURA =====");
205 System.out.println("Fecha: " + new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").format(new Date()));
206 System.out.println("-----");
207 System.out.println("Producto: " + nombreProducto);
208 System.out.println("Codigo: " + codigo);
209 System.out.println("Cantidad: " + cantidadVenta);
210 System.out.println("Precio unitario: $" + precio);
211 System.out.println("-----");
212 System.out.println("Total a pagar: $" + total);
213 System.out.println("=====");
214
215 con.commit(); // Confirmar transacción
216 System.out.println("\nVenta realizada con éxito.");
217 } else {
218 System.out.println("Error: Stock insuficiente.");
219 System.out.println("Stock actual: " + stockActual);
220 System.out.println("Cantidad solicitada: " + cantidadVenta);
221 con.rollback();
222 }
223 } else {
224 System.out.println("Error: Producto no encontrado.");
225 con.rollback();
226 }
227 }
```

```
ConexionBD.java x Login.java x Proyectofinbd.java x
Source History
225 }
226 } catch (SQLException e) {
227 try {
228 if (con != null) con.rollback();
229 } catch (SQLException ex) {
230 System.out.println("Error al realizar rollback: " + ex.getMessage());
231 }
232 System.out.println("Error al realizar la venta: " + e.getMessage());
233 } finally {
234 try {
235 if (con != null) {
236 con.setAutoCommit(true);
237 con.close();
238 }
239 } catch (SQLException e) {
240 System.out.println("Error al cerrar la conexion: " + e.getMessage());
241 }
242 }
243 }
244
245 // Método para generar reporte de inventario
246 public static void generarReporteInventario() {
247 String query = "SELECT codigoProducto, nombreProducto, precioUnitario, cantidadProducto, " +
248 "fechaVencimiento FROM producto ORDER BY nombreProducto";
249 try (Connection con = conectar();
250 Statement st = con.createStatement();
251 ResultSet rs = st.executeQuery(query)) {
252
253 System.out.println("\n===== REPORTE DE INVENTARIO =====");
254 System.out.println("Fecha: " + new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").format(new Date()));
255 System.out.println("=====");
256 }
```

```
ConexionBD.java x Login.java x Proyectofinbd.java x
Source History
257 "CODIGO", "NOMBRE", "PRECIO", "STOCK", "VENCIMIENTO");
258 System.out.println("-----");
259
260 double valorTotal = 0;
261 int totalProductos = 0;
262
263 while (rs.next()) {
264     String codigo = rs.getString("codigoProducto");
265     String nombre = rs.getString("nombreProducto");
266     double precio = rs.getDouble("precioUnitario");
267     int cantidad = rs.getInt("cantidadProducto");
268     Date fechaVenc = rs.getDate("fechaVencimiento");
269
270     valorTotal += precio * cantidad;
271     totalProductos += cantidad;
272
273     System.out.printf("%-12s %-25s $%-9.2f %-10d %-12s\n",
274         codigo, nombre, precio, cantidad,
275         new SimpleDateFormat("yyyy-MM-dd").format(fechaVenc));
276 }
277
278 System.out.println("=====");
279 System.out.printf("Total de productos en inventario: %d\n", totalProductos);
280 System.out.printf("Valor total del inventario: $%.2f\n", valorTotal);
281 System.out.println("=====");
282
283 } catch (SQLException e) {
284     System.out.println("Error al generar reporte: " + e.getMessage());
285 }
286 }
287 }
```

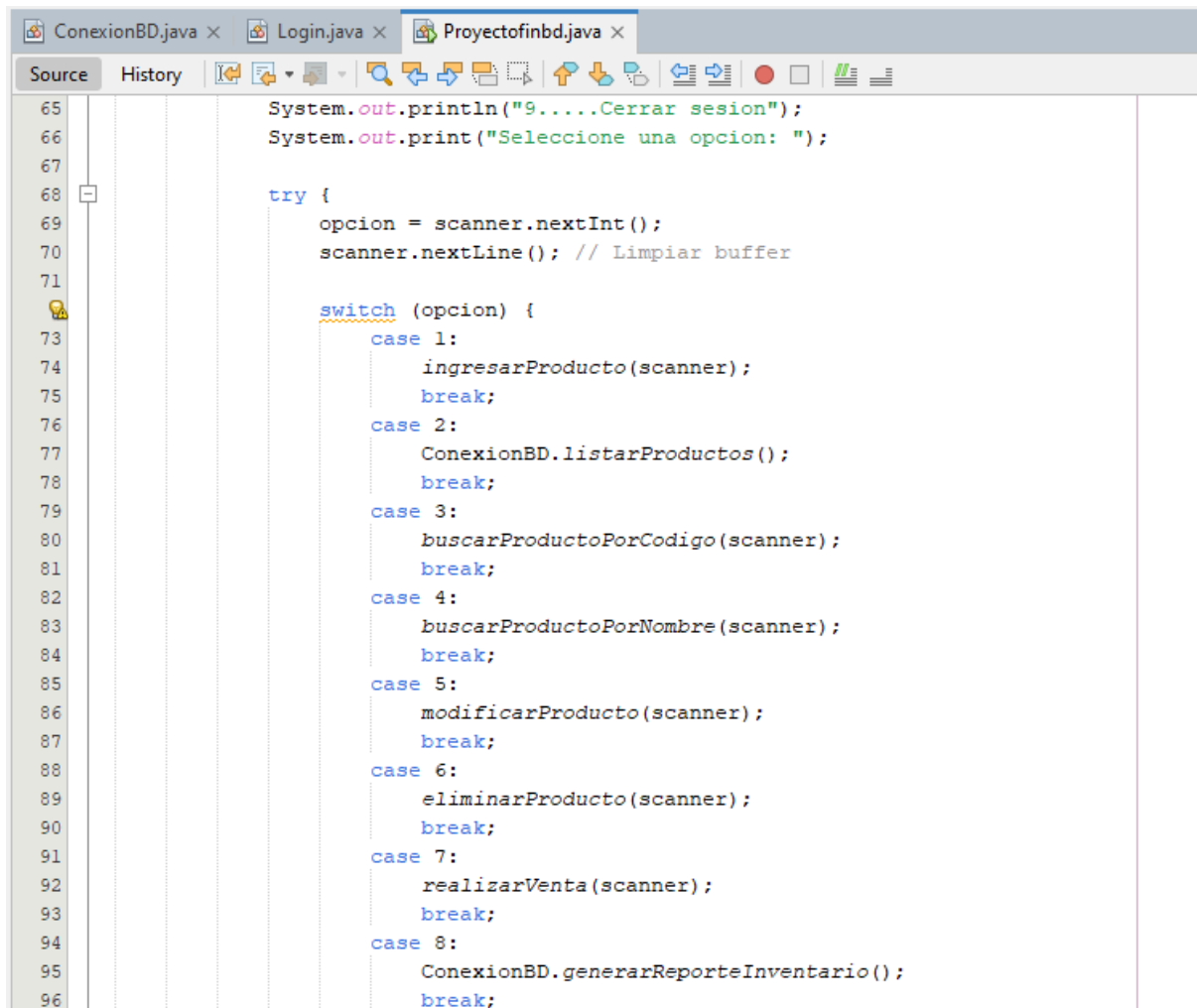
CLASE 2 Proyectofinbd



The screenshot shows the Apache NetBeans IDE with the ProjectoFinbd project open. The left sidebar displays the project structure, including the source packages and the main classes. The main editor window shows the source code for the ProyectoFinbd class, which includes package declarations, imports, and the main method. The code is written in Java and includes comments in Spanish.

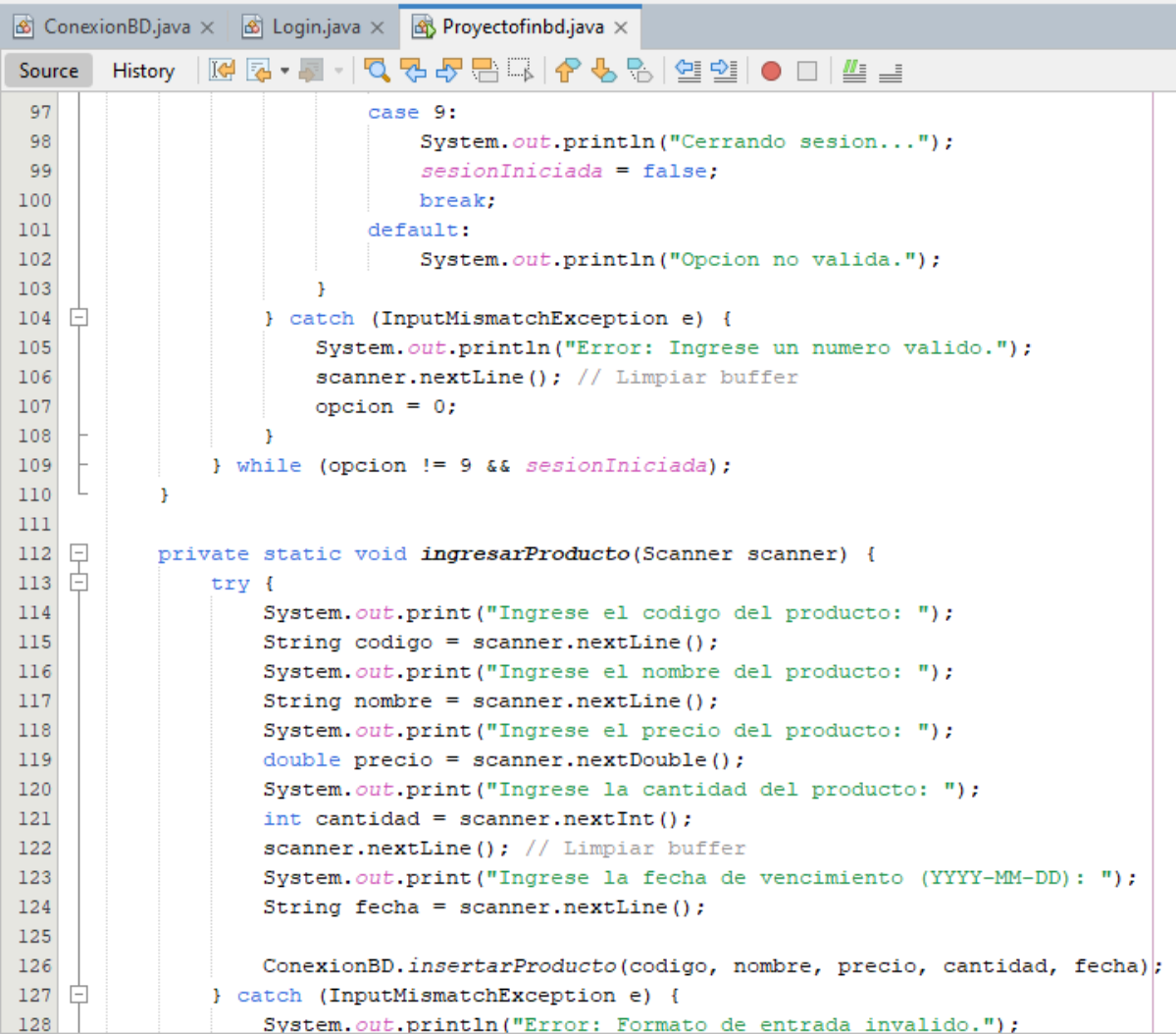
```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
projectofinbd - Apache NetBeans IDE 23
Search (Ctrl+I)
2018/07/26 OMB
ProjectoFinbd
package projectofinbd;
import java.util.InputMismatchException;
import java.util.Scanner;
public class ProyectoFinbd {
    private static boolean sesionIniciada = false;
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Proceso de login
        if (!iniciarSesion(scanner)) {
            System.out.println("Programa terminado.");
            return;
        }
        // Menu principal
        mostrarMenuPrincipal(scanner);
        scanner.close();
    }
    private static boolean iniciarSesion(Scanner scanner) {
        int intentos = 3;
        while (intentos > 0) {
            System.out.println("\n=== BIENVENIDO AL SISTEMA DE INVENTARIO ===");
            System.out.println("Ingrese sus datos para acceder: " + intentos);
            System.out.print("Usuario: ");
            String usuario = scanner.nextLine();
            System.out.print("Psw: ");
            String psw = scanner.nextLine();
            if (usuario.equals("admin") && psw.equals("1234")) {
                sesionIniciada = true;
                return true;
            }
            intentos--;
        }
        return false;
    }
}
```

```
ConexionBD.java x Login.java x Proyectofinbd.java x
Source History
33 String password = scanner.nextLine();
34
35 if (Login.validarUsuario(usuario, password)) {
36     System.out.println("Inicio de sesion exitoso!");
37     sesionIniciada = true;
38     return true;
39 } else {
40     intentos--;
41     if (intentos > 0) {
42         System.out.println("Credenciales incorrectas. Por favor, intente nuevamente.");
43     }
44 }
45
46
47 System.out.println("Numero maximo de intentos alcanzado.");
48 return false;
49
50
51 private static void mostrarMenuPrincipal(Scanner scanner) {
52     int opcion = 0;
53     do {
54         System.out.println("\n*****");
55         System.out.println("      Menu Principal      ");
56         System.out.println("*****");
57         System.out.println("1.....Ingresar producto");
58         System.out.println("2.....Mostrar productos");
59         System.out.println("3.....Buscar producto por codigo");
60         System.out.println("4.....Buscar producto por nombre");
61         System.out.println("5.....Modificar producto");
62         System.out.println("6.....Eliminar producto");
63         System.out.println("7.....Realizar venta");
64         System.out.println("8.....Generar reporte de inventario");
```



The screenshot shows an IDE with three tabs: 'ConexionBD.java', 'Login.java', and 'ProyectoFinbd.java'. The 'ProyectoFinbd.java' tab is active, displaying a Java source file. The code is as follows:

```
65      System.out.println("9.....Cerrar sesion");
66      System.out.print("Seleccione una opcion: ");
67
68      try {
69          opcion = scanner.nextInt();
70          scanner.nextLine(); // Limpiar buffer
71
72          switch (opcion) {
73              case 1:
74                  ingresarProducto(scanner);
75                  break;
76              case 2:
77                  ConexionBD.listarProductos();
78                  break;
79              case 3:
80                  buscarProductoPorCodigo(scanner);
81                  break;
82              case 4:
83                  buscarProductoPorNombre(scanner);
84                  break;
85              case 5:
86                  modificarProducto(scanner);
87                  break;
88              case 6:
89                  eliminarProducto(scanner);
90                  break;
91              case 7:
92                  realizarVenta(scanner);
93                  break;
94              case 8:
95                  ConexionBD.generarReporteInventario();
96                  break;
```



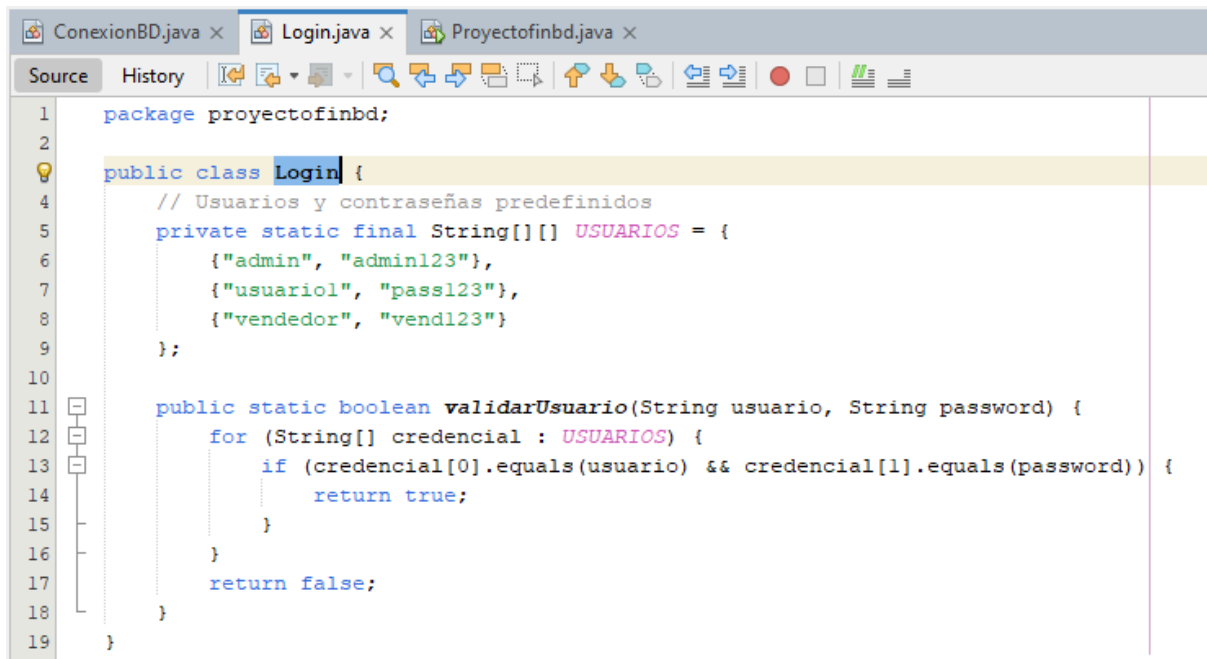
```
ConexionBD.java x Login.java x Proyectofinbd.java x
Source History
97         case 9:
98             System.out.println("Cerrando sesion...");
99             sesionIniciada = false;
100             break;
101         default:
102             System.out.println("Opcion no valida.");
103     }
104 } catch (InputMismatchException e) {
105     System.out.println("Error: Ingrese un numero valido.");
106     scanner.nextLine(); // Limpiar buffer
107     opcion = 0;
108 }
109 } while (opcion != 9 && sesionIniciada);
110 }
111
112 private static void ingresarProducto(Scanner scanner) {
113     try {
114         System.out.print("Ingrese el codigo del producto: ");
115         String codigo = scanner.nextLine();
116         System.out.print("Ingrese el nombre del producto: ");
117         String nombre = scanner.nextLine();
118         System.out.print("Ingrese el precio del producto: ");
119         double precio = scanner.nextDouble();
120         System.out.print("Ingrese la cantidad del producto: ");
121         int cantidad = scanner.nextInt();
122         scanner.nextLine(); // Limpiar buffer
123         System.out.print("Ingrese la fecha de vencimiento (YYYY-MM-DD): ");
124         String fecha = scanner.nextLine();
125
126         ConexionBD.insertarProducto(codigo, nombre, precio, cantidad, fecha);
127     } catch (InputMismatchException e) {
128         System.out.println("Error: Formato de entrada invalido.");
```



```
ConexionBD.java x Login.java x Proyectofinbd.java x
Source History
128 System.out.println("Error: Formato de entrada invalido.");
129 scanner.nextLine(); // Limpiar buffer
130 }
131 }
132
133 private static void buscarProductoPorCodigo(Scanner scanner) {
134     System.out.print("Ingrese el codigo del producto: ");
135     String codigo = scanner.nextLine();
136     ConexionBD.buscarProducto(codigo);
137 }
138
139 private static void buscarProductoPorNombre(Scanner scanner) {
140     System.out.print("Ingrese el nombre del producto: ");
141     String nombre = scanner.nextLine();
142     ConexionBD.buscarProductoPorNombre(nombre);
143 }
144
145 private static void modificarProducto(Scanner scanner) {
146     try {
147         System.out.print("Ingrese el codigo del producto a modificar: ");
148         String codigoActual = scanner.nextLine();
149
150         // Primero verificamos si el producto existe
151         ConexionBD.buscarProducto(codigoActual);
152
153         System.out.println("\nIngrese los nuevos datos del producto:");
154         System.out.print("Nuevo codigo del producto (Enter para mantener el actual): ");
155         String nuevoCodigo = scanner.nextLine();
156         // Si no se ingresa un nuevo código, mantenemos el actual
157         if (nuevoCodigo.trim().isEmpty()) {
158             nuevoCodigo = codigoActual;
159         }
160     }
161 }
```

```
159
160
161     System.out.print("Nuevo nombre del producto: ");
162     String nombre = scanner.nextLine();
163
164     System.out.print("Nuevo precio del producto: ");
165     double precio = scanner.nextDouble();
166
167     System.out.print("Nueva cantidad en stock: ");
168     int cantidad = scanner.nextInt();
169     scanner.nextLine(); // Limpiar buffer
170
171     ConexionBD.actualizarProducto(codigoActual, nuevoCodigo, nombre, precio, cantidad);
172 } catch (InputMismatchException e) {
173     System.out.println("Error: Formato de entrada invalido.");
174     scanner.nextLine(); // Limpiar buffer
175 }
176
177
178 private static void eliminarProducto(Scanner scanner) {
179     System.out.print("Ingrese el codigo del producto a eliminar: ");
180     String codigo = scanner.nextLine();
181     ConexionBD.eliminarProducto(codigo);
182 }
183
184 private static void realizarVenta(Scanner scanner) {
185     try {
186         System.out.print("Ingrese el codigo del producto: ");
187         String codigo = scanner.nextLine();
188         System.out.print("Ingrese la cantidad a vender: ");
189         int cantidad = scanner.nextInt();
190         scanner.nextLine(); // Limpiar buffer
191
192         ConexionBD.realizarVenta(codigo, cantidad);
193     } catch (InputMismatchException e) {
194         System.out.println("Error: Formato de entrada invalido.");
195         scanner.nextLine(); // Limpiar buffer
196     }
197 }
198
199
```

CLASE 3 Login



```
1 package proyectoфинbd;
2
3 public class Login {
4     // Usuarios y contraseñas predefinidos
5     private static final String[][] USUARIOS = {
6         {"admin", "admin123"},
7         {"usuariol", "pass123"},
8         {"vendedor", "vend123"}
9     };
10
11     public static boolean validarUsuario(String usuario, String password) {
12         for (String[] credencial : USUARIOS) {
13             if (credencial[0].equals(usuario) && credencial[1].equals(password)) {
14                 return true;
15             }
16         }
17         return false;
18     }
19 }
```