

SEMANA 8 – TALLER DE DESARROLLO DE APLICACIONES 1

ENTREGADO POR: Mendoza Ricse Gimena Cristel

CÓDIGO: Q00076H

CATEDRÁTICO: Fernández Bejarano Raúl Enrique

PROYECTO FINA – MANUAL

Paso 1. Creación de la Base de Datos

Descripción

Se crea la base de datos gestion_carnets y sus tablas principales (facultades, carreras, carnets). El script incluye un DROP DATABASE IF EXISTS para poder ejecutarlo múltiples veces sin errores.

```
-- Eliminar la base de datos si ya existe
DROP DATABASE IF EXISTS gestion_carnets;

-- Crear la base de datos
CREATE DATABASE gestion_carnets;
USE gestion_carnets;

-- Crear tabla de facultades
CREATE TABLE facultades (
    codigofacultad CHAR(8) PRIMARY KEY,
    nombrefacultad VARCHAR(100) NOT NULL
);

-- Crear tabla de carreras
CREATE TABLE carreras (
    codigocarrera CHAR(8) PRIMARY KEY,
    nombrecarrera VARCHAR(100) NOT NULL,
    codigofacultad CHAR(8) NOT NULL,
    FOREIGN KEY (codigofacultad) REFERENCES facultades(codigofacultad)
);

-- Crear tabla de carnets (estudiantes)
CREATE TABLE carnets (
    codigo VARCHAR(10) PRIMARY KEY,
    dni VARCHAR(15) NOT NULL UNIQUE,
    nombre_completo VARCHAR(100) NOT NULL,
    codigocarrera CHAR(8) NOT NULL,
    FOREIGN KEY (codigocarrera) REFERENCES carreras(codigocarrera)
);
```

```

-- Insertar facultades
INSERT INTO facultades (codigofacultad, nombrefacultad) VALUES
('FAC001', 'Ciencias administrativas y contables'),
('FAC002', 'Derecho y ciencias políticas'),
('FAC003', 'Ingeniería'),
('FAC004', 'Ciencias de la Salud'),
('FAC005', 'Medicina Humana');

-- Insertar carreras para la facultad de Ciencias administrativas y contables
INSERT INTO carreras (codigocarrera, nombrecarrera, codigofacultad) VALUES
('CAR001', 'Administración', 'FAC001'),
('CAR002', 'Administración e Inteligencia de Negocios', 'FAC001'),
('CAR003', 'Administración y Negocios Globales', 'FAC001'),
('CAR004', 'Administración y Gestión del Talento Humano', 'FAC001'),
('CAR005', 'Contabilidad y Finanzas', 'FAC001');

-- Insertar carreras para la facultad de Derecho y ciencias políticas
INSERT INTO carreras (codigocarrera, nombrecarrera, codigofacultad) VALUES
('CAR006', 'Derecho', 'FAC002'),
('CAR007', 'Educación Inicial', 'FAC002'),
('CAR008', 'Educación Primaria', 'FAC002');

-- Insertar carreras para la facultad de Ingeniería
INSERT INTO carreras (codigocarrera, nombrecarrera, codigofacultad) VALUES
('CAR009', 'Arquitectura', 'FAC003'),
('CAR010', 'Ingeniería Civil', 'FAC003'),
('CAR011', 'Ingeniería Industrial', 'FAC003'),
('CAR012', 'Ingeniería del Medio Ambiente y Desarrollo', 'FAC003'),
('CAR013', 'Ingeniería de Sistemas y Computación', 'FAC003');

-- Insertar carreras para la facultad de Ciencias de la Salud
INSERT INTO carreras (codigocarrera, nombrecarrera, codigofacultad) VALUES
('CAR014', 'Enfermería', 'FAC004'),
('CAR015', 'Farmacia y Bioquímica', 'FAC004'),
('CAR016', 'Medicina Veterinaria y Zootecnia', 'FAC004'),
('CAR017', 'Nutrición Humana', 'FAC004'),
('CAR018', 'Obstetricia', 'FAC004'),
('CAR019', 'Odontología', 'FAC004'),
('CAR020', 'Psicología', 'FAC004'),
('CAR021', 'Tecnología Médica', 'FAC004');

-- Insertar carreras para la facultad de Medicina Humana
INSERT INTO carreras (codigocarrera, nombrecarrera, codigofacultad) VALUES
('CAR022', 'Medicina Humana', 'FAC005');

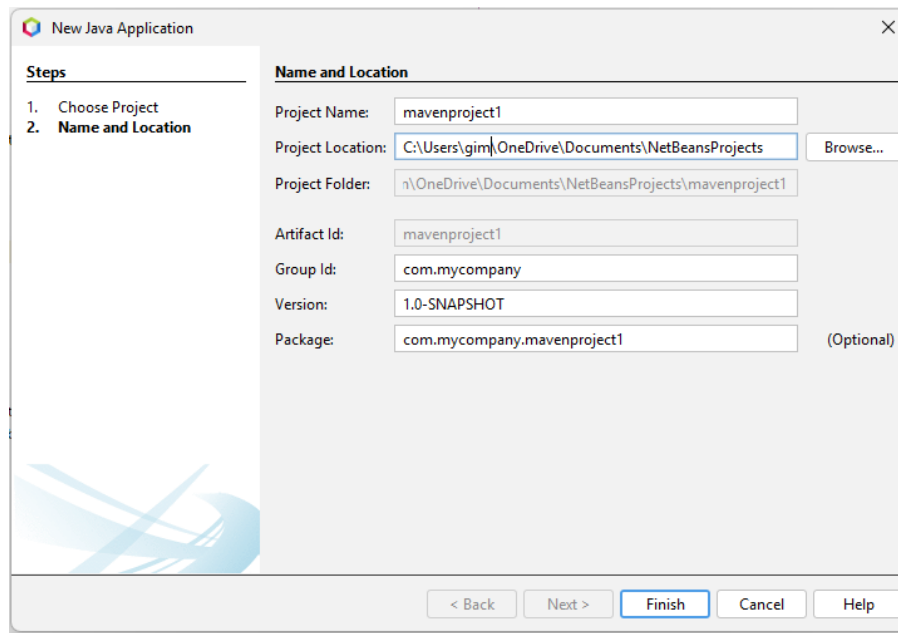
```

Paso 2. Creación del Proyecto en Java (Maven)

Descripción

Se genera un proyecto Maven llamado proyecto-carnets para manejar dependencias y estructura estándar.

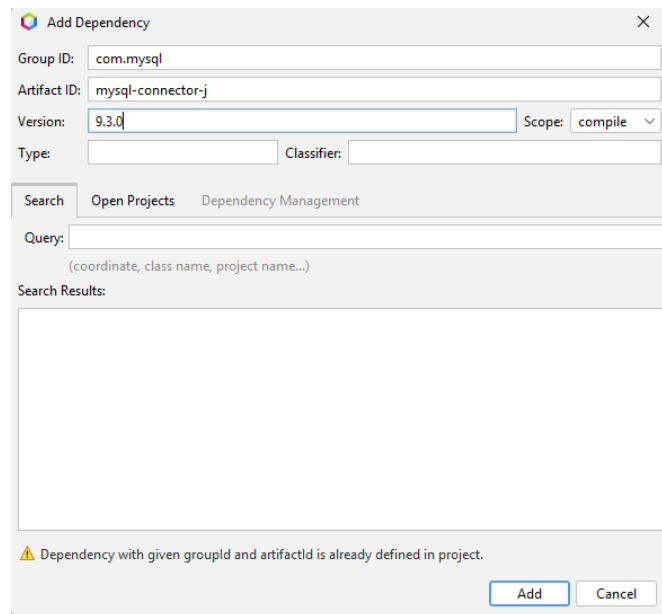
1. En tu IDE (IntelliJ/Eclipse/NetBeans) eliges “New → Maven Project”.
2. Asignas groupId = com.universidad y artifactId = proyecto-carnets.
3. Se crea la estructura de carpetas src/main/java, src/main/resources, etc.



Paso 3. Descarga de Dependencias

Descripción

En el pom.xml se añaden las dependencias para MySQL Connector/J y



Paso 4. Clase de Conexión (conexion.java)

Descripción

Clase utilitaria para abrir la conexión JDBC a la base de datos MySQL:

```
public class conexion {  
    private static final String URL = "jdbc:mysql://localhost:3306/gestion_carnets";  
    private static final String USUARIO = "root"; // Cambia si es distinto  
    private static final String CLAVE = "root123"; // Cambia si tienes contraseña  
  
    public static Connection conectar() throws SQLException {  
        return DriverManager.getConnection(URL, USUARIO, CLAVE);  
    }  
}
```

Paso 5. Modelo de Datos (Carnet.java)

Descripción

Definición de la clase Carnet, que representa un registro de estudiante:

```

public class Carnet {

    private String codigo;
    private String dni;
    private String nombreCompleto;
    private String facultad;
    private String carrera;

    public Carnet(String codigo, String dni) {
        this.codigo = codigo;
        this.dni = dni;
    }

    public Carnet(String codigo, String dni, String nombreCompleto, String facultad, String carrera) {
        this.codigo = codigo;
        this.dni = dni;
        this.nombreCompleto = nombreCompleto;
        this.facultad = facultad;
        this.carrera = carrera;
    }

    public String getCodigo() {
        return codigo;
    }

    public String getDni() {
        return dni;
    }

    public String getNombreCompleto() {
        return nombreCompleto;
    }

    public String getFacultad() {
        return facultad;
    }

    public String getCarrera() {
        return carrera;
    }

    public void actualizarDatos(String nombreCompleto, String facultad, String carrera) {
        this.nombreCompleto = nombreCompleto;
        this.facultad = facultad;
        this.carrera = carrera;
    }

}

```

Paso 6. Interfaz Principal (CMenu)

Descripción

Ventana Swing que muestra:

- Un campo de búsqueda.
- Una tabla con columnas: Código, DNI, Nombre, Facultad, Carrera.
- Botones **CREAR**, **ACTUALIZAR**, **ELIMINAR** y **SALIR**.

SISTEMA CARNET

BUSCA SU CARNET:

Title 1	Title 2	Title 3	Title 4

CREAR

ACTUALIZAR

ELIMINAR


SALIR

Paso 7. Formulario de Carnet (CCarnet)

Descripción

Diálogo Swing usado para crear o editar un carnet. Contiene:

- Text fields para DNI, Código, Nombre completo.
- Combo boxes para Facultad y Carrera (el segundo se actualiza dinámicamente).
- Botones **CREAR/ACTUALIZAR** y **VOLVER**.



DNI : **NOMBRES:**

CODIGO: **APELLIDOS:**

FACULTAD: **CARRERA:**

CREAR

VOLVER

Paso 8. Controlador del Menú (MenuControlador.java)

Descripción

Clase que:

1. Inicializa la vista CMenu.
2. Implementa cargarCarnetsDesdeBD() y buscarCarnet(), que leen de carnets y, mediante subconsultas a carreras y facultades, obtienen los nombres.
3. Gestiona eventos de los botones y la selección de filas.

```
19 public class MenuControlador implements ActionListener, MouseListener, KeyListener {
20
21     private final CMenu vista;
22     public static Carnet carnetSeleccionado = null;
23
24     public MenuControlador(CMenu vista) {
25         this.vista = vista;
26
27         this.vista.btnCrear.addActionListener(this);
28         this.vista.btnActualizar.addActionListener(this);
29         this.vista.btnEliminar.addActionListener(this);
30         this.vista.btnSalir.addActionListener(this);
31
32         this.vista.tblDatos.addMouseListener(this);
33         this.vista.txtBuscar.addKeyListener(this);
34
35         this.vista.btnActualizar.setEnabled(false);
36         this.vista.btnEliminar.setEnabled(false);
37
38         // Cargar los datos de la base al iniciar
39         cargarCarnetsDesdeBD();
40     }
```

```

42 private void cargarCarnetsDesdeBD() {
43     DefaultTableModel modelo = new DefaultTableModel();
44     modelo.addColumn("Código");
45     modelo.addColumn("DNI");
46     modelo.addColumn("Nombre Completo");
47     modelo.addColumn("Facultad");
48     modelo.addColumn("Carrera");
49
50     try (Connection con = conexion.conectar()) {
51         String sql = "SELECT * FROM carnets";
52         PreparedStatement ps = con.prepareStatement(sql);
53         ResultSet rs = ps.executeQuery();
54
55         while (rs.next()) {
56             String codigo = rs.getString("codigo");
57             String dni = rs.getString("dni");
58             String nombre = rs.getString("nombre_completo");
59             String codCarrera = rs.getString("codigocarrera");
60
61             String nombreCarrera = obtenerNombreCarrera(con, codCarrera);
62             String nombreFacultad = obtenerNombreFacultadDesdeCarrera(con, codCarrera);
63
64             modelo.addRow(new Object[]{codigo, dni, nombre, nombreFacultad, nombreCarrera});
65         }
66
67         vista.tblDatos.setModel(modelo);
68     } catch (Exception e) {
69         JOptionPane.showMessageDialog(vista, "Error al cargar datos: " + e.getMessage());
70     }
71 }
72

```

```

73 private String obtenerNombreCarrera(Connection con, String codCarrera) {
74     try {
75         String sql = "SELECT nombrecarrera FROM carreras WHERE codigocarrera = ?";
76         PreparedStatement ps = con.prepareStatement(sql);
77         ps.setString(1, codCarrera);
78         ResultSet rs = ps.executeQuery();
79         if (rs.next()) {
80             return rs.getString("nombrecarrera");
81         }
82     } catch (Exception e) {
83         // manejar si quieres
84     }
85     return "";
86 }
87
88 private String obtenerNombreFacultadDesdeCarrera(Connection con, String codCarrera) {
89     try {
90         String sqlCarrera = "SELECT codigofacultad FROM carreras WHERE codigocarrera = ?";
91         PreparedStatement psCarrera = con.prepareStatement(sqlCarrera);
92         psCarrera.setString(1, codCarrera);
93         ResultSet rsCarrera = psCarrera.executeQuery();
94
95         if (rsCarrera.next()) {
96             String codFacultad = rsCarrera.getString("codigofacultad");
97
98             String sqlFacultad = "SELECT nombrefacultad FROM facultades WHERE codigofacultad = ?";
99             PreparedStatement psFacultad = con.prepareStatement(sqlFacultad);
100             psFacultad.setString(1, codFacultad);
101             ResultSet rsFacultad = psFacultad.executeQuery();
102
103             if (rsFacultad.next()) {
104                 return rsFacultad.getString("nombrefacultad");
105             }
106         }
107     } catch (Exception e) {
108         // manejar si quieres
109     }
110     return "";
111 }

```



```

88 private String obtenerNombreFacultadDesdeCarrera(Connection con, String codCarrera) {
89     try {
90         String sqlCarrera = "SELECT codigofacultad FROM carreras WHERE codigocarrera = ?";
91         PreparedStatement psCarrera = con.prepareStatement(sqlCarrera);
92         psCarrera.setString(1, codCarrera);
93         ResultSet rsCarrera = psCarrera.executeQuery();
94
95         if (rsCarrera.next()) {
96             String codFacultad = rsCarrera.getString("codigofacultad");
97
98             String sqlFacultad = "SELECT nombrefacultad FROM facultades WHERE codigofacultad = ?";
99             PreparedStatement psFacultad = con.prepareStatement(sqlFacultad);
100             psFacultad.setString(1, codFacultad);
101             ResultSet rsFacultad = psFacultad.executeQuery();
102
103             if (rsFacultad.next()) {
104                 return rsFacultad.getString("nombrefacultad");
105             }
106         }
107     } catch (Exception e) {
108         // manejar si quieres
109     }
110     return "";
111 }
112
113 private void buscarCarnet() {
114     String texto = vista.txtBuscar.getText().toLowerCase();
115
116     DefaultTableModel modelo = new DefaultTableModel();
117     modelo.addColumn("Código");
118     modelo.addColumn("DNI");
119     modelo.addColumn("Nombre Completo");
120     modelo.addColumn("Facultad");
121     modelo.addColumn("Carrera");
122
123     try (Connection con = conexion.conectar()) {
124         String sql = "SELECT * FROM carnets WHERE LOWER(dni) LIKE ? OR LOWER(codigo) LIKE ? OR LOWER(nombre_completo) LIKE ?";
125         PreparedStatement ps = con.prepareStatement(sql);
126         String searchTerm = "%" + texto + "%";
127         ps.setString(1, searchTerm);
128         ps.setString(2, searchTerm);
129         ps.setString(3, searchTerm);
130
131         ResultSet rs = ps.executeQuery();
132
133         while (rs.next()) {
134             String codigo = rs.getString("codigo");
135             String dni = rs.getString("dni");
136             String nombre = rs.getString("nombre_completo");
137             String codCarrera = rs.getString("codigocarrera");
138
139             String nombreCarrera = obtenerNombreCarrera(con, codCarrera);
140             String nombreFacultad = obtenerNombreFacultadDesdeCarrera(con, codCarrera);
141
142             modelo.addRow(new Object[]{codigo, dni, nombre, nombreFacultad, nombreCarrera});
143         }
144
145         vista.tblDatos.setModel(modelo);
146     } catch (Exception e) {
147         JOptionPane.showMessageDialog(vista, "Error al buscar: " + e.getMessage());
148     }
149 }
150

```

```

151     @Override
152     public void actionPerformed(ActionEvent e) {
153         Object source = e.getSource();
154
155         if (source == vista.btnCrear) {
156             crear();
157         } else if (source == vista.btnActualizar) {
158             actualizar();
159         } else if (source == vista.btnEliminar) {
160             eliminar();
161         } else if (source == vista.btnSalir) {
162             salir();
163         }
164     }
165
166     private void crear() {
167         CCarnet dialogo = new CCarnet(vista, true, "crear");
168         dialogo.setVisible(true);
169         cargarCarnetsDesdeBD();
170         limpiarSeleccion();
171     }
172
173     private void actualizar() {
174         if (carnetSeleccionado != null) {
175             CCarnet dialogo = new CCarnet(vista, true, "actualizar");
176             dialogo.setVisible(true);
177             cargarCarnetsDesdeBD();
178             limpiarSeleccion();
179         } else {
180             JOptionPane.showMessageDialog(vista, "Seleccione un carnet primero.");
181         }
182     }
183
184     private void eliminar() {
185         if (carnetSeleccionado != null) {
186             int confirm = JOptionPane.showConfirmDialog(vista, "¿Seguro de eliminar este carnet?", "Confirmar", JOptionPane.YES_NO_OPTION);
187             if (confirm == JOptionPane.YES_OPTION) {
188                 try (Connection con = conexion.conectar()) {
189                     String sql = "DELETE FROM carnets WHERE codigo = ?";
190                     PreparedStatement ps = con.prepareStatement(sql);
191                     ps.setString(1, carnetSeleccionado.getCodigo());
192                     ps.executeUpdate();
193                     cargarCarnetsDesdeBD();
194                     carnetSeleccionado = null;
195                     limpiarSeleccion();
196                 } catch (Exception e) {
197                     JOptionPane.showMessageDialog(vista, "Error al eliminar: " + e.getMessage());
198                 }
199             } else {
200                 JOptionPane.showMessageDialog(vista, "Seleccione un carnet primero.");
201             }
202         }
203     }
204
205     private void salir() {
206         int respuesta = JOptionPane.showOptionDialog(vista, "¿Estás seguro de salir?", "Salir", JOptionPane.YES_NO_OPTION, JOptionPane.QUESTION_MESS.
207         if (respuesta == 0) {
208             System.exit(0);
209         }
210         vista.txtBuscar.requestFocus();
211     }
212
213     private void limpiarSeleccion() {
214         carnetSeleccionado = null;
215         vista.btnActualizar.setEnabled(false);
216         vista.btnEliminar.setEnabled(false);
217         vista.tblDatos.clearSelection();
218     }
219

```

```

220     @Override
221     public void mouseClicked(MouseEvent e) {
222         int fila = vista.tblDatos.getSelectedRow();
223         if (fila >= 0) {
224             String codigo = (String) vista.tblDatos.getValueAt(fila, 0); // columna 0 = código
225
226             try (Connection con = conexion.conectar()) {
227                 String sql = "SELECT * FROM carnets WHERE codigo = ?";
228                 PreparedStatement ps = con.prepareStatement(sql);
229                 ps.setString(1, codigo);
230                 ResultSet rs = ps.executeQuery();
231                 if (rs.next()) {
232                     carnetSeleccionado = new Carnet(
233                         rs.getString("codigo"),
234                         rs.getString("dni"),
235                         rs.getString("nombre_completo"),
236                         rs.getString("facultad"),
237                         rs.getString("carrera")
238                     );
239                 }
240                 vista.btnActualizar.setEnabled(true);
241                 vista.btnEliminar.setEnabled(true);
242             } catch (Exception ex) {
243                 JOptionPane.showMessageDialog(vista, "Error al seleccionar: " + ex.getMessage());
244             }
245         }
246     }
247
248
249
250     @Override
251     public void keyReleased(KeyEvent e) {
252         if (e.getSource() == vista.txtBuscar) {
253             buscarCarnet();
254         }
255     }
256
257
258

```

Paso 9. Controlador del Formulario (CarnetControlador.java)

Descripción

Gestiona el diálogo CCarnet en dos modos:

- **Crear:** valida campos, obtiene codigocarrera a partir del nombre seleccionado y hace INSERT en la tabla carnets.
- **Actualizar:** precarga datos desde la base (lee codigocarrera, traduce a nombres), permite editar y hace UPDATE.

```
18 public class CarnetControlador implements ActionListener {
19
20     private final CCarnet vista;
21     private final String tipo;
22
23     public CarnetControlador(CCarnet vista, String tipo) {
24         this.vista = vista;
25         this.tipo = tipo;
26
27         this.vista.btnCrear.addActionListener(this);
28         this.vista.btnVolver.addActionListener(this);
29
30         if (tipo.equalsIgnoreCase("actualizar")) {
31             cargarDatos();
32         }
33     }
34 }
```

```

35 private void cargarDatos() {
36     String codigo = MenuControlador.carnetSeleccionado.getCodigo();
37     String dni = MenuControlador.carnetSeleccionado.getDni();
38
39     String sql = "SELECT * FROM carnets WHERE codigo = ? AND dni = ?";
40
41     try (Connection conn = conexion.conectar(); PreparedStatement stmt = conn.prepareStatement(sql)) {
42
43         stmt.setString(1, codigo);
44         stmt.setString(2, dni);
45
46         ResultSet rs = stmt.executeQuery();
47
48         if (rs.next()) {
49             String nombre = rs.getString("nombre_completo");
50             String codCarrera = rs.getString("codigocarrera");
51
52             String nombreCarrera = obtenerNombreCarrera(conn, codCarrera);
53             String nombreFacultad = obtenerNombreFacultadDesdeCarrera(conn, codCarrera);
54
55             vista.txtCodigo.setText(codigo);
56             vista.txtDni.setText(dni);
57             vista.txtNombre.setText(nombre);
58
59             vista.cmbFacultad.setSelectedItem(nombreFacultad);
60             vista.actualizarCarreras();
61             vista.cmbCarrera.setSelectedItem(nombreCarrera);
62
63             vista.txtCodigo.setEditable(false);
64             vista.txtDni.setEditable(false);
65         } else {
66             JOptionPane.showMessageDialog(vista, "Carnet no encontrado.", "Aviso", JOptionPane.WARNING_MESSAGE);
67             vista.dispose();
68         }
69
70     } catch (SQLException e) {
71         JOptionPane.showMessageDialog(vista, "Error al cargar datos:\n" + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
72     }
73 }
74
75 private String obtenerNombreCarrera(Connection con, String codCarrera) {
76     String sql = "SELECT nombrecarrera FROM carreras WHERE codigocarrera = ?";
77     try (PreparedStatement ps = con.prepareStatement(sql)) {
78         ps.setString(1, codCarrera);
79         ResultSet rs = ps.executeQuery();
80         if (rs.next()) {
81             return rs.getString("nombrecarrera");
82         }
83     } catch (SQLException e) {
84         JOptionPane.showMessageDialog(vista, "Error al obtener nombre de carrera:\n" + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
85     }
86     return null;
87 }
88
89 private String obtenerNombreFacultadDesdeCarrera(Connection con, String codCarrera) {
90     String codFacultad = null;
91     String sqlCarrera = "SELECT codigofacultad FROM carreras WHERE codigocarrera = ?";
92     try (PreparedStatement psCarrera = con.prepareStatement(sqlCarrera)) {
93         psCarrera.setString(1, codCarrera);
94         ResultSet rsCarrera = psCarrera.executeQuery();
95         if (rsCarrera.next()) {
96             codFacultad = rsCarrera.getString("codigofacultad");
97         }
98     } catch (SQLException e) {
99         JOptionPane.showMessageDialog(vista, "Error al obtener código de facultad:\n" + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
100     }
101     return null;
102
103     if (codFacultad != null) {
104         String sqlFacultad = "SELECT nombrefacultad FROM facultades WHERE codigofacultad = ?";
105         try (PreparedStatement psFacultad = con.prepareStatement(sqlFacultad)) {
106             psFacultad.setString(1, codFacultad);
107             ResultSet rsFacultad = psFacultad.executeQuery();
108             if (rsFacultad.next()) {
109                 return rsFacultad.getString("nombrefacultad");
110             }
111         } catch (SQLException e) {
112             JOptionPane.showMessageDialog(vista, "Error al obtener nombre de facultad:\n" + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
113         }
114     }
115
116     return null;
117 }

```

```

119 private boolean validarCampos() {
120     if (vista.txtCodigo.getText().trim().isEmpty()
121         || vista.txtDni.getText().trim().isEmpty()
122         || vista.txtNombre.getText().trim().isEmpty()
123         || vista.cmbFacultad.getSelectedIndex() == -1
124         || vista.cmbCarrera.getSelectedIndex() == -1) {
125         JOptionPane.showMessageDialog(vista, "Todos los campos deben estar completos.", "Error", JOptionPane.ERROR_MESSAGE);
126         return false;
127     }
128     return true;
129 }
130
131 private void crearCarnet() {
132     String codigoCarrera = obtenerCodigoCarrera((String) vista.cmbCarrera.getSelectedItem());
133
134     if (codigoCarrera == null) {
135         JOptionPane.showMessageDialog(vista, "No se encontró el código de la carrera.", "Error", JOptionPane.ERROR_MESSAGE);
136         return;
137     }
138
139     String sql = "INSERT INTO carnets (codigo, dni, nombre_completo, codigocarrera) VALUES (?, ?, ?, ?)";
140
141     try (Connection conn = conexion.conectar(); PreparedStatement stmt = conn.prepareStatement(sql)) {
142
143         stmt.setString(1, vista.txtCodigo.getText());
144         stmt.setString(2, vista.txtDni.getText());
145         stmt.setString(3, vista.txtNombre.getText());
146         stmt.setString(4, codigoCarrera);
147
148         stmt.executeUpdate();
149
150         JOptionPane.showMessageDialog(vista, "Carnet creado exitosamente.");
151         vista.dispose();
152     } catch (SQLException e) {
153         JOptionPane.showMessageDialog(vista, "Error al crear carnet:\n" + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
154     }
155 }
156
157
158 private void actualizarCarnet() {
159     String codigoCarrera = obtenerCodigoCarrera((String) vista.cmbCarrera.getSelectedItem());
160
161     if (codigoCarrera == null) {
162         JOptionPane.showMessageDialog(vista, "No se encontró el código de la carrera.", "Error", JOptionPane.ERROR_MESSAGE);
163         return;
164     }
165
166     String sql = "UPDATE carnets SET nombre_completo = ?, codigocarrera = ? WHERE codigo = ? AND dni = ?";
167
168     try (Connection conn = conexion.conectar(); PreparedStatement stmt = conn.prepareStatement(sql)) {
169
170         stmt.setString(1, vista.txtNombre.getText());
171         stmt.setString(2, codigoCarrera);
172         stmt.setString(3, vista.txtCodigo.getText());
173         stmt.setString(4, vista.txtDni.getText());
174
175         stmt.executeUpdate();
176
177         JOptionPane.showMessageDialog(vista, "Carnet actualizado exitosamente.");
178         vista.dispose();
179     } catch (SQLException e) {
180         JOptionPane.showMessageDialog(vista, "Error al actualizar carnet:\n" + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
181     }
182 }
183
184 private String obtenerCodigoCarrera(String nombreCarrera) {
185     String sql = "SELECT codigocarrera FROM carreras WHERE nombrecarrera = ?";
186     try (Connection conn = conexion.conectar(); PreparedStatement stmt = conn.prepareStatement(sql)) {
187         stmt.setString(1, nombreCarrera);
188         ResultSet rs = stmt.executeQuery();
189         if (rs.next()) {
190             return rs.getString("codigocarrera");
191         }
192     } catch (SQLException e) {
193         JOptionPane.showMessageDialog(vista, "Error al obtener código de carrera:\n" + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
194     }
195     return null;
196 }

```

```
8
9      @Override
10     public void actionPerformed(ActionEvent e) {
11         Object source = e.getSource();
12
13         if (source == vista.btnVolver) {
14             vista.dispose();
15         } else if (source == vista.btnCrear) {
16             if (validarCampos()) {
17                 if (tipo.equalsIgnoreCase("crear")) {
18                     crearCarnet();
19                 } else if (tipo.equalsIgnoreCase("actualizar")) {
20                     actualizarCarnet();
21                 }
22             }
23         }
24     }
25 }
```