TALLER DE DESARROLLO DE APLICACIONES 1

ENTREGADO POR: Mescua Segovia Marino

CÓDIGO: N04058C

CATEDRÁTICO: Fernández Bejarano Raúl Enrique

PROYECTO FINAL

ESTRUCTURA:

- **modelo:** Contiene las clases que representan las entidades (Modelo) y la conexión a la base de datos.
- **controlador:** Contiene la lógica de negocio y la conexión con la base de datos (Controlador).
- **vista:** Ventanas de interfaz gráfica (Vista).

BASE DE DATOS:

```sql
CREATE TABLE facultades (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL
);

CREATE TABLE carreras (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    facultad_id INT,
    FOREIGN KEY (facultad_id) REFERENCES facultades(id)
);

CREATE TABLE carnets (
    codigo VARCHAR(10) PRIMARY KEY,
    dni VARCHAR(15) NOT NULL,
    nombre VARCHAR(100) NOT NULL,
    apellido VARCHAR(100) NOT NULL,
    facultad_id INT,
    carrera_id INT,
    FOREIGN KEY (facultad_id) REFERENCES facultades(id),
    FOREIGN KEY (carrera_id) REFERENCES carreras(id)
);
```

MODELO - CARNETS:

```java
public class Carnet {

    private String codigo;
    private String dni;
    private String nombre;
    private String apellido;
    private int facultadId;
    private int carreraId;

    public Carnet() {
    }

    public Carnet(String codigo, String dni, String nombre, String apellido, int facultadId, int carreraId) {
        this.codigo = codigo;
        this.dni = dni;
        this.nombre = nombre;
        this.apellido = apellido;
        this.facultadId = facultadId;
        this.carreraId = carreraId;
    }

    // Getters y Setters
    public String getCodigo() {
        return codigo;
    }

    public void setCodigo(String codigo) {
        this.codigo = codigo;
    }

    public String getDni() {
        return dni;
    }

    public void setDni(String dni) {
        this.dni = dni;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getApellido() {
        return apellido;
    }

    public void setApellido(String apellido) {
        this.apellido = apellido;
    }

    public int getFacultadId() {
        return facultadId;
    }

    public void setFacultadId(int facultadId) {
        this.facultadId = facultadId;
    }

    public int getCarreraId() {
        return carreraId;
    }

    public void setCarreraId(int carreraId) {
        this.carreraId = carreraId;
    }
}
```

MODELO - CARRERA:

```java
public class Carrera {

    private int id;
    private String nombre;
    private int facultadId;

    public Carrera() {
    }

    public Carrera(int id, String nombre, int facultadId) {
        this.id = id;
        this.nombre = nombre;
        this.facultadId = facultadId;
    }

    // Getters y Setters
    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public int getFacultadId() {
        return facultadId;
    }

    public void setFacultadId(int facultadId) {
        this.facultadId = facultadId;
    }

    @Override
    public String toString() {
        return nombre;
    }
}
```

MODELO - CONEXION:

```java
public class Conexion {

    private static final String URL = "jdbc:mysql://localhost:3306/carnetsmarino";
    private static final String USER = "root"; // Cambia esto si tu usuario es diferente
    private static final String PASSWORD = "root123"; // Cambia esto si tu contraseña no está vacía

    public static Connection getConexion() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            return DriverManager.getConnection(URL, USER, PASSWORD);
        } catch (ClassNotFoundException | SQLException e) {
            e.printStackTrace();
            return null;
        }
    }
}
```

MODELO - FACULTAD:

```java
public class Facultad {

    private int id;
    private String nombre;

    public Facultad() {
    }

    public Facultad(int id, String nombre) {
        this.id = id;
        this.nombre = nombre;
    }

    // Getters y Setters
    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    @Override
    public String toString() {
        return nombre;
    }
}
```

CONTROLADOR - CARNETCONTROLADOR:

```java
public class CarnetControlador {

    public boolean crearCarnet(Carnet carnet) {
        String sql = "INSERT INTO carnets (codigo, dni, nombre, apellido, facultad_id, carrera_id) VALUES (?, ?, ?, ?, ?, ?)";
        try (Connection conn = Conexion.getConexion(); PreparedStatement ps = conn.prepareStatement(sql)) {

            ps.setString(1, carnet.getCodigo());
            ps.setString(2, carnet.getDni());
            ps.setString(3, carnet.getNombre());
            ps.setString(4, carnet.getApellido());
            ps.setInt(5, carnet.getFacultadId());
            ps.setInt(6, carnet.getCarreraId());

            return ps.executeUpdate() > 0;

        } catch (Exception e) {
            e.printStackTrace();
        }
        return false;
    }
```

```java
    public List<Carnet> listarCarnets() {
        List<Carnet> lista = new ArrayList<>();
        String sql = "SELECT * FROM carnets";

        try (Connection conn = Conexion.getConexion(); PreparedStatement ps = conn.prepareStatement(sql); ResultSet rs = ps.executeQuery()) {

            while (rs.next()) {
                Carnet c = new Carnet();
                c.setCodigo(rs.getString("codigo"));
                c.setDni(rs.getString("dni"));
                c.setNombre(rs.getString("nombre"));
                c.setApellido(rs.getString("apellido"));
                c.setFacultadId(rs.getInt("facultad_id"));
                c.setCarreraId(rs.getInt("carrera_id"));
                lista.add(c);
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
        return lista;
    }
```

```java
    public boolean eliminarCarnet(String codigo) {
        String sql = "DELETE FROM carnets WHERE codigo = ?";

        try (Connection conn = Conexion.getConexion(); PreparedStatement ps = conn.prepareStatement(sql)) {

            ps.setString(1, codigo);
            return ps.executeUpdate() > 0;

        } catch (Exception e) {
            e.printStackTrace();
        }
        return false;
    }
```

```java
    public boolean actualizarCarnet(Carnet carnet) {
        String sql = "UPDATE carnets SET dni=?, nombre=?, apellido=?, facultad_id=?, carrera_id=? WHERE codigo=?";
        try (Connection conn = Conexion.getConexion(); PreparedStatement ps = conn.prepareStatement(sql)) {

            ps.setString(1, carnet.getDni());
            ps.setString(2, carnet.getNombre());
            ps.setString(3, carnet.getApellido());
            ps.setInt(4, carnet.getFacultadId());
            ps.setInt(5, carnet.getCarreraId());
            ps.setString(6, carnet.getCodigo());

            return ps.executeUpdate() > 0;

        } catch (Exception e) {
            e.printStackTrace();
        }
        return false;
    }
```

CONTROLADOR - CARRERACONTROLADOR:

```java
public class CarreraControlador {

    public List<Carrera> listarCarreras() {
        List<Carrera> lista = new ArrayList<>();
        String sql = "SELECT * FROM carreras";

        try (Connection conn = Conexion.getConexion(); PreparedStatement ps = conn.prepareStatement(sql); ResultSet rs = ps.executeQuery()) {

            while (rs.next()) {
                Carrera c = new Carrera();
                c.setId(rs.getInt("id"));
                c.setNombre(rs.getString("nombre"));
                c.setFacultadId(rs.getInt("facultad_id"));
                lista.add(c);
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
        return lista;
    }

    public boolean agregarCarrera(String nombre, int facultadId) {
        String sql = "INSERT INTO carreras (nombre, facultad_id) VALUES (?, ?)";

        try (Connection conn = Conexion.getConexion(); PreparedStatement ps = conn.prepareStatement(sql)) {

            ps.setString(1, nombre);
            ps.setInt(2, facultadId);
            return ps.executeUpdate() > 0;

        } catch (Exception e) {
            e.printStackTrace();
        }
        return false;
    }
```

CONTROLADOR - FACULTADCONTROLADOR:

```java
public class FacultadControlador {

    public List<Facultad> listarFacultades() {
        List<Facultad> lista = new ArrayList<>();
        String sql = "SELECT * FROM facultades";

        try (Connection conn = Conexion.getConexion(); PreparedStatement ps = conn.prepareStatement(sql); ResultSet rs = ps.executeQuery()) {

            while (rs.next()) {
                Facultad f = new Facultad();
                f.setId(rs.getInt("id"));
                f.setNombre(rs.getString("nombre"));
                lista.add(f);
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
        return lista;
    }

    public boolean agregarFacultad(String nombre) {
        String sql = "INSERT INTO facultades (nombre) VALUES (?)";
        try (Connection conn = Conexion.getConexion(); PreparedStatement ps = conn.prepareStatement(sql)) {

            ps.setString(1, nombre);
            return ps.executeUpdate() > 0;

        } catch (Exception e) {
            e.printStackTrace();
        }
        return false;
    }
}
```

VISTA - MAIN:

```java
public class Main extends javax.swing.JFrame {

    /**
     * Creates new form Main
     */
    FacultadControlador facultadControlador = new FacultadControlador();
    CarreraControlador carreraControlador = new CarreraControlador();
    CarnetControlador carnetControlador = new CarnetControlador();

    public Main() {
        initComponents();
        configurarTablaCarnets();
        cargarFacultadesYCarreras();
        cargarTablaCarnets();
    }

    private void configurarTablaCarnets() {
        String[] columnas = {
            "Código", "DNI", "Nombre", "Apellido", "ID Facultad", "ID Carrera"
        };

        DefaultTableModel modelo = new DefaultTableModel(null, columnas) {
            @Override
            public boolean isCellEditable(int row, int column) {
                return false; // hace que ninguna celda sea editable
            }
        };

        tblCarnets.setModel(modelo);
    }

    private void cargarTablaCarnets() {
        DefaultTableModel modelo = (DefaultTableModel) tblCarnets.getModel();
        modelo.setRowCount(0); // Limpiar tabla antes de llenar

        List<Carnet> lista = carnetControlador.listarCarnets();
        for (Carnet c : lista) {
            Object[] fila = {
                c.getCodigo(),
                c.getDni(),
                c.getNombre(),
                c.getApellido(),
                c.getFacultadId(),
                c.getCarreraId()
            };
            modelo.addRow(fila);
        }
    }
```

```java
private void cargarFacultadesYCarreras() {
    cmbFacultad.removeAllItems();
    cmbCarrera.removeAllItems();

    List<Facultad> facultades = facultadControlador.listarFacultades();
    for (Facultad f : facultades) {
        cmbFacultad.addItem(f);
    }

    // Si hay facultades, cargar carreras de la primera
    if (!facultades.isEmpty()) {
        cargarCarrerasPorFacultad(facultades.get(0).getId());
    }
}

private void limpiarCampos() {
    txtCodigo.setText("");
    txtDni.setText("");
    txtNombres.setText("");
    txtApellidos.setText("");
    cmbFacultad.setSelectedIndex(0);
    cmbCarrera.setSelectedIndex(0);
    tblCarnets.clearSelection();
}

private void cargarCarrerasPorFacultad(int facultadId) {
    cmbCarrera.removeAllItems();

    List<Carrera> lista = carreraControlador.listarCarreras();
    for (Carrera c : lista) {
        if (c.getFacultadId() == facultadId) {
            cmbCarrera.addItem(c);
        }
    }
}
```

```java
private void cmbFacultadActionPerformed(java.awt.event.ActionEvent evt) {
    Facultad facultadSeleccionada = (Facultad) cmbFacultad.getSelectedItem();
    if (facultadSeleccionada != null) {
        cargarCarrerasPorFacultad(facultadSeleccionada.getId());
    }
}

private void txtNombresActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void txtApellidosActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void tblCarnetsMouseClicked(java.awt.event.MouseEvent evt) {
    int fila = tblCarnets.getSelectedRow();
    if (fila != -1) {
        // Obtener datos de la fila
        String codigo = tblCarnets.getValueAt(fila, 0).toString();
        String dni = tblCarnets.getValueAt(fila, 1).toString();
        String nombre = tblCarnets.getValueAt(fila, 2).toString();
        String apellido = tblCarnets.getValueAt(fila, 3).toString();
        int facultadId = Integer.parseInt(tblCarnets.getValueAt(fila, 4).toString());
        int carreraId = Integer.parseInt(tblCarnets.getValueAt(fila, 5).toString());

        txtCodigo.setText(codigo);
        txtDni.setText(dni);
        txtNombres.setText(nombre);
        txtApellidos.setText(apellido);

        // Seleccionar en combos
        for (int i = 0; i < cmbFacultad.getItemCount(); i++) {
            if (cmbFacultad.getItemAt(i).getId() == facultadId) {
                cmbFacultad.setSelectedIndex(i);
                break;
            }
        }

        for (int i = 0; i < cmbCarrera.getItemCount(); i++) {
            if (cmbCarrera.getItemAt(i).getId() == carreraId) {
                cmbCarrera.setSelectedIndex(i);
                break;
            }
        }
    }
}
```

```java
private void btnCrearActionPerformed(java.awt.event.ActionEvent evt) {
    String codigo = txtCodigo.getText().trim();
    String dni = txtDni.getText().trim();
    String nombre = txtNombres.getText().trim();
    String apellido = txtApellidos.getText().trim();
    Facultad facultad = (Facultad) cmbFacultad.getSelectedItem();
    Carrera carrera = (Carrera) cmbCarrera.getSelectedItem();

    // Validar campos obligatorios
    if (codigo.isEmpty() || dni.isEmpty() || nombre.isEmpty() || apellido.isEmpty()
            || facultad == null || carrera == null) {
        JOptionPane.showMessageDialog(this, "Por favor, completa todos los campos.");
        return;
    }

    // Crear el objeto Carnet
    Carnet carnet = new Carnet(
            codigo,
            dni,
            nombre,
            apellido,
            facultad.getId(),
            carrera.getId()
    );

    boolean creado = carnetControlador.crearCarnet(carnet);

    if (creado) {
        JOptionPane.showMessageDialog(this, "Carnet creado exitosamente.");
        cargarTablaCarnets(); // Actualiza la tabla con el nuevo registro
        limpiarCampos(); // Vuelve al estado inicial
    } else {
        JOptionPane.showMessageDialog(this, "Error al crear el carnet. Verifica que el código no exista.");
    }
}
```

```java
private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    String codigo = txtCodigo.getText().trim();

    if (codigo.isEmpty()) {
        JOptionPane.showMessageDialog(this, "Selecciona un carnet primero.");
        return;
    }

    int confirmacion = JOptionPane.showConfirmDialog(
            this,
            "¿Estás seguro de eliminar el carnet con código: " + codigo + "?",
            "Confirmar eliminación",
            JOptionPane.YES_NO_OPTION
    );

    if (confirmacion == JOptionPane.YES_OPTION) {
        boolean eliminado = carnetControlador.eliminarCarnet(codigo);

        if (eliminado) {
            JOptionPane.showMessageDialog(this, "Carnet eliminado correctamente.");
            cargarTablaCarnets();
            limpiarCampos(); // Deshabilita campos y botones
        } else {
            JOptionPane.showMessageDialog(this, "Error al eliminar el carnet.");
        }
    }
}
```

```
    private void btnRecargarActionPerformed(java.awt.event.ActionEvent evt) {
        cargarFacultadesYCarreras(); // Recarga desde la BD
        JOptionPane.showMessageDialog(this, "Facultades y carreras actualizadas.");
    }

    private void txtBuscarCarnetKeyReleased(java.awt.event.KeyEvent evt) {
        String texto = txtBuscarCarnet.getText().trim().toLowerCase();
        DefaultTableModel modelo = (DefaultTableModel) tblCarnets.getModel();
        modelo.setRowCount(0); // limpia tabla

        List<Carnet> lista = carnetControlador.listarCarnets();

        for (Carnet c : lista) {
            if (c.getCodigo().toLowerCase().contains(texto) || c.getDni().toLowerCase().contains(texto)) {
                Object[] fila = {
                    c.getCodigo(),
                    c.getDni(),
                    c.getNombre(),
                    c.getApellido(),
                    c.getFacultadId(),
                    c.getCarreraId()
                };
                modelo.addRow(fila);
            }
        }

        if (modelo.getRowCount() == 0) {
            JOptionPane.showMessageDialog(this, "No se encontraron coincidencias.");
        }
    }

    private void btnAbrirCarreraActionPerformed(java.awt.event.ActionEvent evt) {
        new GuiCarrera(this, true).setVisible(true);
    }

    private void btnAbrirFacultadActionPerformed(java.awt.event.ActionEvent evt) {
        new GuiFacultad(this, true).setVisible(true);
    }
```

VISTA - GUICARRERA:

```
public class GuiCarrera extends javax.swing.JDialog {

    /**
     * Creates new form Carrera
     */
    CarreraControlador carreraControlador = new CarreraControlador();
    FacultadControlador facultadControlador = new FacultadControlador();

    public GuiCarrera(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
        cargarFacultades(); // llenar combo
        cargarTablaCarreras();
    }

    private void cargarFacultades() {
        cmbFacultadCarrera.removeAllItems();
        for (Facultad f : facultadControlador.listarFacultades()) {
            cmbFacultadCarrera.addItem(f);
        }
    }

    private void cargarTablaCarreras() {
        String[] columnas = {"ID", "Nombre", "ID Facultad"};
        DefaultTableModel modelo = new DefaultTableModel(null, columnas) {
            @Override
            public boolean isCellEditable(int row, int column) {
                return false;
            }
        };

        List<Carrera> lista = carreraControlador.listarCarreras();

        for (Carrera c : lista) {
            Object[] fila = {
                c.getId(),
                c.getNombre(),
                c.getFacultadId()
            };
            modelo.addRow(fila);
        }

        tblCarreras.setModel(modelo);
    }

```

```java
    private void btnCrearCarreraActionPerformed(java.awt.event.ActionEvent evt) {
        String nombre = txtNombreCarrera.getText().trim();
        Facultad facultad = (Facultad) cmbFacultadCarrera.getSelectedItem();

        if (nombre.isEmpty() || facultad == null) {
            JOptionPane.showMessageDialog(this, "Completa el nombre y selecciona una facultad."
            return;
        }

        boolean creada = carreraControlador.agregarCarrera(nombre, facultad.getId());

        if (creada) {
            JOptionPane.showMessageDialog(this, "Carrera registrada.");
            txtNombreCarrera.setText("");
            cargarTablaCarreras();
        } else {
            JOptionPane.showMessageDialog(this, "Error al registrar la carrera.");
        }
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        Look and feel setting code (optional)

        /* Create and display the dialog */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                GuiCarrera dialog = new GuiCarrera(new javax.swing.JFrame(), true);
                dialog.addWindowListener(new java.awt.event.WindowAdapter() {
                    @Override
                    public void windowClosing(java.awt.event.WindowEvent e) {
                        System.exit(0);
                    }
                });
                dialog.setVisible(true);
            }
        });
    }
```

VISTA - GUIFACULTAD:

```java
public class GuiFacultad extends javax.swing.JDialog {

    /**
     * Creates new form GuiFacultad
     */
    FacultadControlador facultadControlador = new FacultadControlador();

    public GuiFacultad(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
        cargarTablaFacultades();
    }

    private void cargarTablaFacultades() {
        String[] columnas = {"ID", "Nombre"};
        DefaultTableModel modelo = new DefaultTableModel(null, columnas) {
            @Override
            public boolean isCellEditable(int row, int column) {
                return false;
            }
        };

        List<Facultad> lista = facultadControlador.listarFacultades();
        for (Facultad f : lista) {
            Object[] fila = {f.getId(), f.getNombre()};
            modelo.addRow(fila);
        }

        tblFacultades.setModel(modelo);
    }
```

```java
    private void btnCrearFacultadActionPerformed(java.awt.event.ActionEvent evt) {
        String nombre = txtNombreFacultad.getText().trim();

        if (nombre.isEmpty()) {
            JOptionPane.showMessageDialog(this, "Ingrese el nombre de la facultad.");
            return;
        }

        boolean creada = facultadControlador.agregarFacultad(nombre);

        if (creada) {
            JOptionPane.showMessageDialog(this, "Facultad creada correctamente.");
            txtNombreFacultad.setText("");
            cargarTablaFacultades();
        } else {
            JOptionPane.showMessageDialog(this, "Error al crear facultad.");
        }
    }
```