Index

Introducción

A "students" database was designed in java, which allows us to make changes such as: Add, Edit, Visualize, Delete and Update data. All the information that we add and it does it in an organized way so that later we can find such data.
Our database is composed of a table in which each one of the columns keeps a part of the information, on each element that we want to add in our table and each row conforms a record.

Diagramas de clase

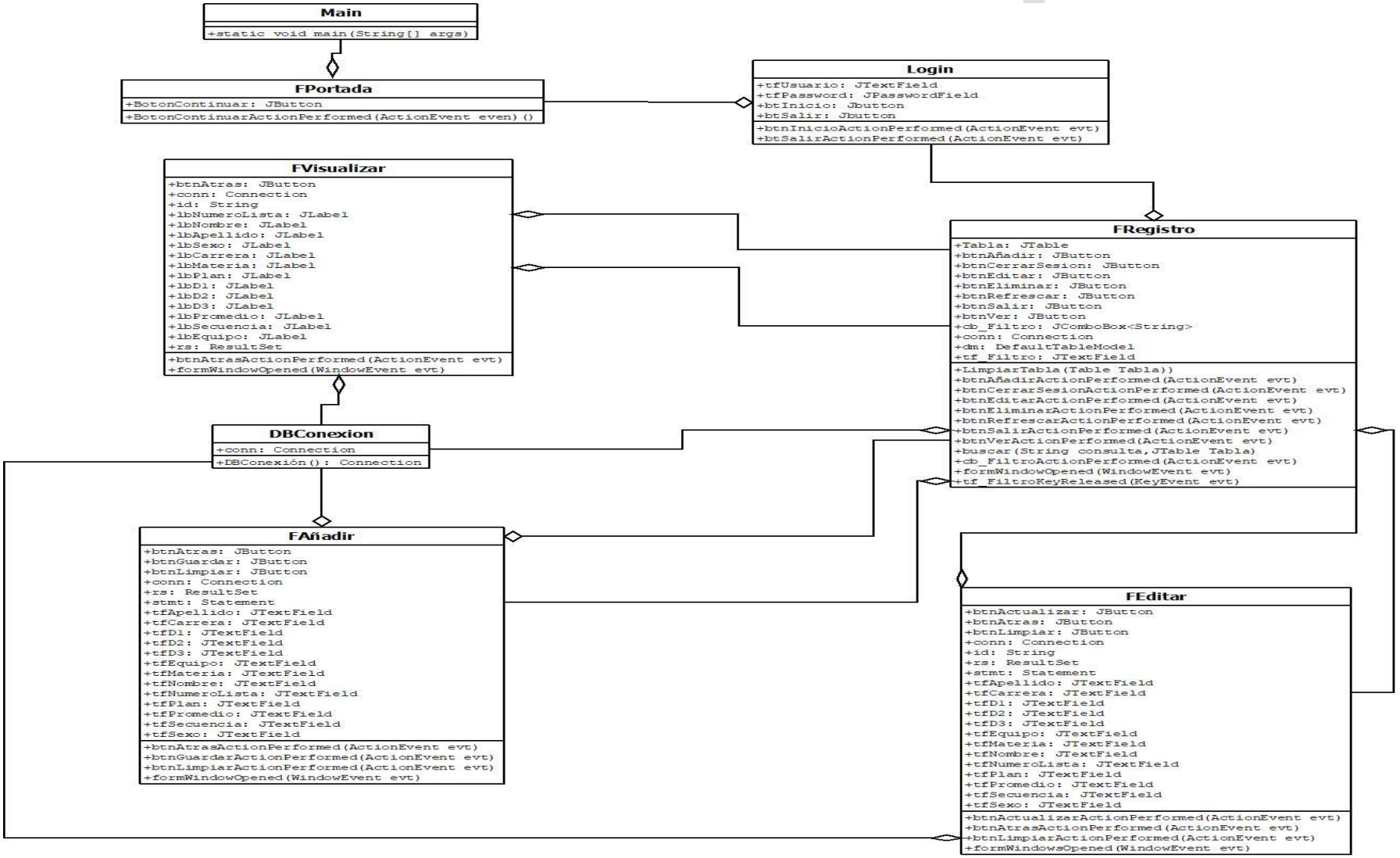


Diagrama de composición de clases

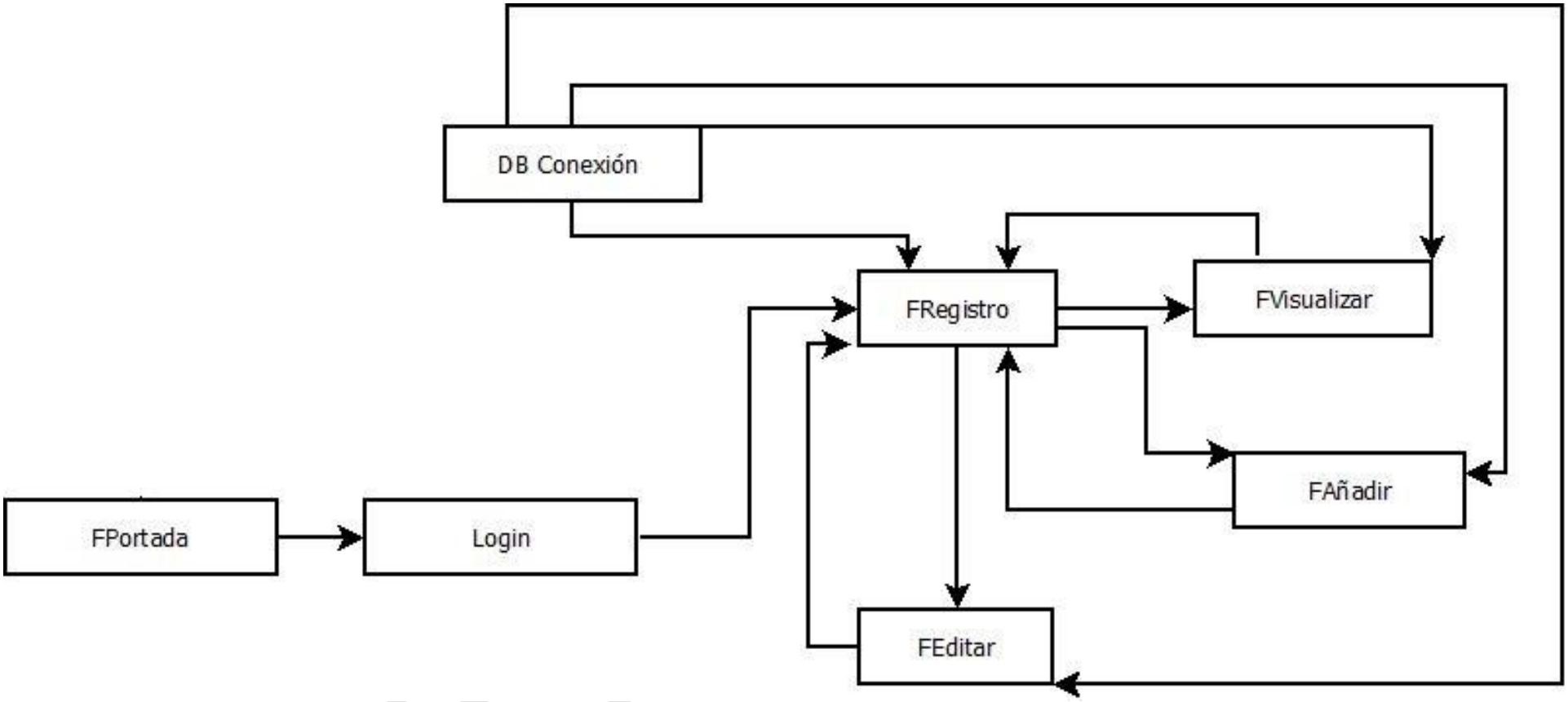
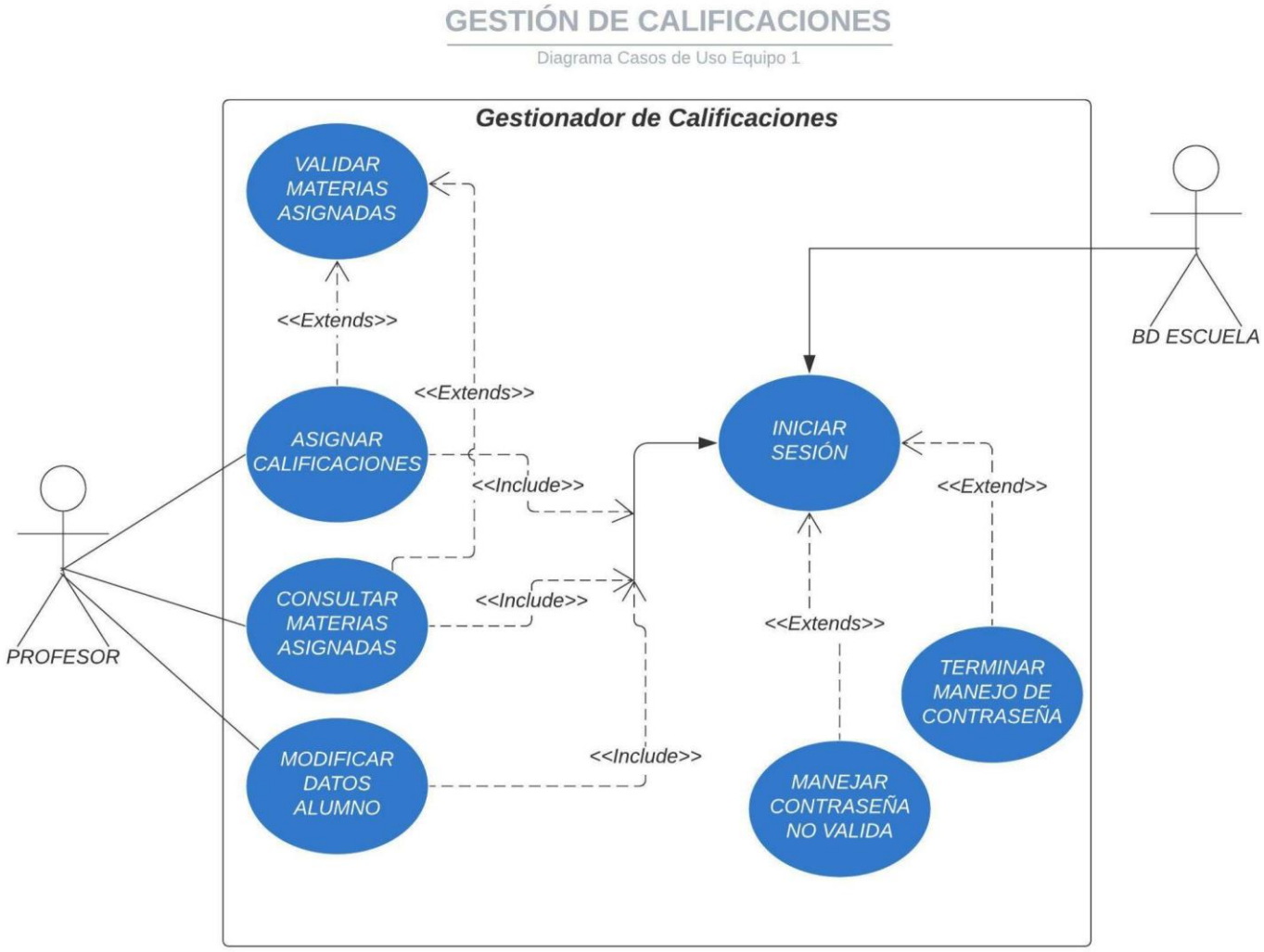


Diagrama de Casos de Uso



Implementación de todos los métodos

CLASE MAIN

public static void main(String[] args)

- Es la clase de ejecución principal
- Inicia la entrada por FPortada.
    - public static void main(String[] args) {
    - FPortada Portada = new FPortada();
    - Portada.setVisible(true);
    - }

CLASE DBCONEXION

public static Connection DBConexión()

- Contiene la dirección, nombre, usuario y contraseña de la base de datos
- Recibe la conexión con la base de datos
- Su tipo de dato es String
- Retorna un valor nulo de conexión exitosa o un Exception si la conexión fracasó.
    - try {
    - Class.forName("com.mysql.jdbc.Driver");
    - Connection conn = DriverManager.getConnection("jdbc:mysql://localhost/dbcalificaciones","root","");
    - return conn;
    - }catch(Exception e) {
    - System.out.println("Error en la bd: " + e);
    - return null;
    - }

CLASE PORTADA

private void BotonContinuarActionPerformed(java.awt.event.ActionEvent evt)

- Muestra la portada con el nombre de todos los integrantes
- Recibe el click del botón continuar
- Click del usuario
- Al dar click al botón de tipo JButton este nos redirige a la clase FLogin
    - FLogin Login = new FLogin();
    - Login.setVisible(true);
    - this.setVisible(false);
    - this.dispose();
- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 25)

CLASE FLOGIN

private void btnInicioActionPerformed(java.awt.event.ActionEvent evt)

- Muestra un botón para iniciar sesión del usuario
- Recibe usuario y password proporcionados de tipo String
- Click del usuario a los campos de usuario y contraseña, estas son validadas con la base de datos, si los datos existen y coinciden el boton se procederá a ejecutar
- Al dar click al botón de tipo JButton este nos redirige a la clase FRegisto

```
String user = tfUsuario.getText();
String password = tfPassword.getText();
try {
    Connection connection = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/dbcalificaciones",
    "root", "");
deberan coincidir con los ingresados en los campos de texto.
    PreparedStatement st = (PreparedStatement) connection
    .prepareStatement("SELECT user, password FROM login
WHERE user = ? AND password = ?");
    st.setString(1, user);
    st.setString(2, password);
    ResultSet rs = st.executeQuery();
    if (rs.next()) {
    FRegistro Registro = new FRegistro();
    Registro.setVisible(true);
    this.dispose();
    } else {
    JOptionPane.showMessageDialog(btnInicio, "Usuario y/ó
Contraseña Incorrectos","ALERTA",JOptionPane.ERROR_MESSAGE);
    }
} catch (SQLException sqlException) {
    sqlException.printStackTrace();
}
```

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 25)

CLASE FREGISTRO

private void formWindowOpened(java.awt.event.WindowEvent evt)

- Muestra la tabla con los datos de la base de datos automáticamente al ser abierto el formulario
- Al ejecutarse se llenan los datos de la tabla cada dato es de tipo String y la tabla es de tipo JTable
- Al dar click al botón de tipo JButton este nos redirige a la clase FRegisto

```
conn = DBConexion.DBConexión();
 // Desde sql usando SELECT para la tabla alumnos y ordenando los datos
desde el ultimo agregado (orden descendente)
String sql = "SELECT * FROM alumnos ORDER BY NumeroLista DESC";
try {
PStatement = conn.prepareStatement(sql);
 rs = PStatement.executeQuery();
DefaultTableModel model = (DefaultTableModel) Tabla.getModel();
while(rs.next()){
// Se llama a los campos de la tabla de bd a la tabla "Tabla".
String id = rs.getString("NumeroLista");
String lnombre = rs.getString("Nombre");
String lApellido = rs.getString("ApellidoPaterno");
```

```
String lSexo = rs.getString("Sexo");
String lcarrera = rs.getString("Carrera");
String lmateria = rs.getString("Materia");
String lplan = rs.getString("PlanEstudios");
String lD1 = rs.getString("D1");
String lD2 = rs.getString("D2");
String lD3 = rs.getString("D3");
String lpromedio = rs.getString("PromedioR");
String lsecuencia = rs.getString("Secuencia");
String lequipo = rs.getString("NoEquipo");
//String Sexo = rs.getString("sexo");
 // Se añaden los valores a cada columna
            Object[] row = { id, lnombre,lApellido, lSexo, lcarrera, lmateria,
lplan, lD1, lD2, lD3, lpromedio, lsecuencia, lequipo };
            model.addRow(row);
String lnombre = rs.getString("Nombre");
            String lApellido = rs.getString("ApellidoPaterno");
            String lSexo = rs.getString("Sexo");
            String lcarrera = rs.getString("Carrera");
            String lmateria = rs.getString("Materia");
            String lplan = rs.getString("PlanEstudios");
            String lD1 = rs.getString("D1");
            String lD2 = rs.getString("D2");
            String lD3 = rs.getString("D3");
            String lpromedio = rs.getString("PromedioR");
            String lsecuencia = rs.getString("Secuencia");
            String lequipo = rs.getString("NoEquipo");
            //String Sexo = rs.getString("sexo");
            // Se añaden los valores a cada columna
            Object[] row = { id, lnombre,lApellido, lSexo, lcarrera, lmateria,
lplan, lD1, lD2, lD3, lpromedio, lsecuencia, lequipo };
            model.addRow(row);
          }
          // Si existiera un error este se debera mostrar en consola como una
excepcion
       }catch(Exception ex) {
          System.out.println("Error: "+ex);
        }
           //Mustra el promedio grupal al momento en un label
           String sql2 = "SELECT AVG(PromedioR) AS PromedioR FROM
alumnos WHERE PromedioR";
           try {
              PStatement = conn.prepareStatement(sql2);
              rs = PStatement.executeQuery();
              while(rs.next()){
```

```
                // Se llama al resultado de la operac´on sql
                String Media = rs.getString("PromedioR");
                lb_PrGeneral1.setText(Media);
            }
        }catch(Exception ex) {
            System.out.println("Error: "+ex);
        }
```

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 26)

private void btnVerActionPerformed(java.awt.event.ActionEvent evt)

- Muestra un botón para redireccionar al formulario de ver
- Recibe de parte del usuario el campo seleccionado en la tabla para redirigir con esos datos.

```
        int row = Tabla.getSelectedRow();
        String id = Tabla.getModel().getValueAt(row, 0).toString();
        FVisualizar Ver = new FVisualizar();
        Ver.id = id;
        Ver.setVisible(true);
        this.setVisible(false);
        this.dispose();
```

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 26)

private void btnEditarActionPerformed(java.awt.event.ActionEvent evt)

- Muestra un botón para redireccionar al formulario de editar
- Recibe de parte del usuario el campo seleccionado en la tabla, para redirigir con esos datos.

```
        int row = Tabla.getSelectedRow();
        String id = Tabla.getModel().getValueAt(row, 0).toString();
        FEditar Editar = new FEditar();
        Editar.id = id;
        Editar.setVisible(true);
        this.setVisible(false);
        this.dispose();
```

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 26)

private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt)

- Muestra un botón llamado eliminar de tipo JButton que ejecuta una serie de acciones
- Recibe el click del botón y hace una operación sql a la tabla con la instrucción DELETE dependiendo el item seleccionado de la tabla

```
    int row = Tabla.getSelectedRow();
        String id = Tabla.getModel().getValueAt(row, 0).toString();
        String sql = "DELETE FROM alumnos WHERE NumeroLista ="+id+"";
        int dialogButton = JOptionPane.showConfirmDialog (null, "¿Esta
    Seguro?","ALERTA",JOptionPane.YES_NO_OPTION);
        if(dialogButton == 0){
            try {
                stmt = conn.createStatement();
                if(stmt.executeUpdate(sql) == 1){
```

- JOptionPane.showMessageDialog(null,"Se elimino con exito","ALERTA",JOptionPane.INFORMATION_MESSAGE);
- // Refrescar la ventana para mostrar los dataos actualizados
- this.setVisible(false);
- new FRegistro().setVisible(true);
- } else {
- JOptionPane.showMessageDialog(null,"Error al eliminar","ALERTA",JOptionPane.WARNING_MESSAGE);
- }
- } catch(Exception ex) {
- System.out.println("Error: "+ex);
- }

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 26)

private void btnAñadirActionPerformed(java.awt.event.ActionEvent evt)

- Muestra la portada con el nombre de todos los integrantes
- Recibe el click del botón continuar
- Click del usuario
- Al dar click al botón de tipo JButton este nos redirige a la clase FLogin
  - new FAñadir().setVisible(true);
  - this.setVisible(false);
  - this.dispose();
- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 27)

private void btnRefrescarActionPerformed(java.awt.event.ActionEvent evt)

- Abre y cierra el formulario
- Recibe el click del botón refrescar
- Click del usuario
  - this.setVisible(false);
  - new FRegistro().setVisible(true);
  - this.dispose();

private void btnSalirActionPerformed(java.awt.event.ActionEvent evt)

- Sale del programa
- Recibe el click del botón continuar
- Click del usuario
- Al dar click al botón de tipo JButton cierra el programa por completo.
  - System.exit(0);
- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 26)

private void btnCerrarSesionActionPerformed(java.awt.event.ActionEvent evt)

- Sale del formulario FRegistro sin cerrar el programa
- Recibe el click del botón.
- Click del usuario
- Al dar click al botón de tipo JButton este nos redirige a la clase FLogin nuevamente, se deberá iniciar sesión de nuevo para ir a FRegistro.
  - this.setVisible(false);
  - new FLogin().setVisible(true);
  - this.dispose();

private void cb_FiltroActionPerformed(java.awt.event.ActionEvent evt)

- Consta del filtro de ComboBox con Mayor promedio, menor promedio, mayor a la media, etc, para efectos de simplificación solo se pusieron 3 en este documento.
- Recibe el dato del ComboBox y la JTabla los datos de la consulta sql, llenandola cada vez cuando el ComboBox cambia de opción.
- Recibe click del usuario, la conexión de la base de datos conn, los datos de la tabla de la base de datos,.

```
        if(cb_Filtro.getSelectedItem().toString().equals("Mayor Promedio")){
            // Este metodo se usa para limpiar la tabla definido con anterioridad
            LimpiarTabla(Tabla);
JOptionPane.showMessageDialog(null,"TEST","ALERTA",JOptionPane.WARNING_MESSAGE);
            conn = DBConexion.DBConexión();
            // Se hace una consulta mysql de la tabla alumnos de la columna PromedioR pero imprimiendo en tabla los mayores a 7
            String sql = "SELECT * FROM alumnos WHERE PromedioR > 7";
            try {
               PStatement = conn.prepareStatement(sql);
               rs = PStatement.executeQuery();
               DefaultTableModel model = (DefaultTableModel) Tabla.getModel();
               while(rs.next()){
                  // Se llama a los campos de la tabla de bd a la tabla "Tabla".
                  String id = rs.getString("NumeroLista");
                  String lnombre = rs.getString("Nombre");
                  String lApellido = rs.getString("ApellidoPaterno");
                  String lSexo = rs.getString("Sexo");
                  String lcarrera = rs.getString("Carrera");
                  String lmateria = rs.getString("Materia");
                  String lplan = rs.getString("PlanEstudios");
                  String lD1 = rs.getString("D1");
                  String lD2 = rs.getString("D2");
                  String lD3 = rs.getString("D3");
                  String lpromedio = rs.getString("PromedioR");
                  String lsecuencia = rs.getString("Secuencia");
                  String lequipo = rs.getString("NoEquipo");
                  Object[] row = { id, lnombre,lApellido, lSexo, lcarrera, lmateria, lplan, lD1, lD2, lD3, lpromedio, lsecuencia, lequipo };
                  model.addRow(row);
               }
               // Si existiera un error este se debera mostrar en consola como una excepcion
            } catch(Exception ex) {
               System.out.println("Error: "+ex);
            }
```

```
        }
        else if(cb_Filtro.getSelectedItem().toString().equals("Menor Promedio")){
            LimpiarTabla(Tabla);
            conn = DBConexion.DBConexión();
            // Se hace una consulta mysql de la tabla alumnos de la columna PromedioR pero imprimiendo en tabla los menores o iguales a 6
            String sql = "SELECT * FROM alumnos WHERE PromedioR <= 6";
            try {
               PStatement = conn.prepareStatement(sql);
               rs = PStatement.executeQuery();
               DefaultTableModel model = (DefaultTableModel) Tabla.getModel();
               while(rs.next()){
                  // Se llama a los campos de la tabla de bd a la tabla "Tabla".
                  String id = rs.getString("NumeroLista");
                  String lnombre = rs.getString("Nombre");
                  String lApellido = rs.getString("ApellidoPaterno");
                  String lSexo = rs.getString("Sexo");
                  String lcarrera = rs.getString("Carrera");
                  String lmateria = rs.getString("Materia");
                  String lplan = rs.getString("PlanEstudios");
                  String lD1 = rs.getString("D1");
                  String lD2 = rs.getString("D2");
                  String lD3 = rs.getString("D3");
                  String lpromedio = rs.getString("PromedioR");
                  String lsecuencia = rs.getString("Secuencia");
                  String lequipo = rs.getString("NoEquipo");
                  // Se añaden los valores a cada columna
                  Object[] row = { id, lnombre,lApellido, lSexo, lcarrera, lmateria, lplan, lD1, lD2, lD3, lpromedio, lsecuencia, lequipo };
                  model.addRow(row);
               }
            } catch(Exception ex) {
               System.out.println("Error: "+ex);
            }
        }
        else if(cb_Filtro.getSelectedItem().toString().equals("Mayor a la M")){
            LimpiarTabla(Tabla);
            conn = DBConexion.DBConexión();
            // Se hace una comsulta mysql con una subconsulta de promedio grupal y posteriormente por alumno de la columna PromedioR
            // Se imprimen los mayores al promedio grupal
            String sql = "SELECT * FROM alumnos WHERE PromedioR > (SELECT AVG(PromedioR) FROM alumnos)";
```

```
            try {
               PStatement = conn.prepareStatement(sql);
               rs = PStatement.executeQuery();
               DefaultTableModel model = (DefaultTableModel)
Tabla.getModel();
               while(rs.next()){
                  String id = rs.getString("NumeroLista");
                     String lnombre = rs.getString("Nombre");
                  String lApellido = rs.getString("ApellidoPaterno");
                  String lSexo = rs.getString("Sexo");
                  String lcarrera = rs.getString("Carrera");
                  String lmateria = rs.getString("Materia");
                  String lplan = rs.getString("PlanEstudios");
                  String lD1 = rs.getString("D1");
                  String lD2 = rs.getString("D2");
                  String lD3 = rs.getString("D3");
                  String lpromedio = rs.getString("PromedioR");
                  String lsecuencia = rs.getString("Secuencia");
                  String lequipo = rs.getString("NoEquipo");
                  Object[] row = { id, lnombre,lApellido, lSexo, lcarrera, lmateria,
lplan, lD1, lD2, lD3, lpromedio, lsecuencia, lequipo };
                  model.addRow(row);
               }
            } catch(Exception ex) {
            System.out.println("Error: "+ex);
            }
         }
```

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 25)

public void LimpiarTabla(JTable Tabla)

- Vacía la tabla a su originalidad.
- Recibe la Tabla JTable
- Se invoca al método en el filtro de la tabla de Registro.

```
            try {
               DefaultTableModel modelo=(DefaultTableModel) Tabla.getModel();
               int filas=Tabla.getRowCount();
               for (int i = 0;filas>i; i++) {
                  modelo.removeRow(0);
               }
         } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "Error al limpiar la tabla.");
         }
   …
```

private void buscar (String consulta, JTable Tabla)

- Realiza una pequeña búsqueda en la tabla de FRegistro

- Recibe el dato proporcionado por el usuario en tf_Filtro de tipo JTextField, número o letra.
- Devuelve los datos reflejados en la tabla de FRegistro.
- Al dar click al botón de tipo JButton cierra el programa por completo.
    - dm = (DefaultTableModel) Tabla.getModel();
    - TableRowSorter<DefaultTableModel> tr = new TableRowSorter<>(dm);
    - Tabla.setRowSorter(tr);
    - tr.setRowFilter(RowFilter.regexFilter(consulta));
    - Las pruebas del método se adjuntan en el apartado de capturas (Pag. 28)

CLASE FAÑADIR

private void formWindowOpened(java.awt.event.WindowEvent evt)

- Se inicia al abrir el formulario automáticamente.
- Al iniciar carga la conexión inicial de la base de datos definida en DBConexion.
    - conn = DBConexion.DBConexión();

private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt)

- Registra la instrucion cargada en los JTextField
- Recibe los datos de tipo String de los campos de texto.
- Al dar click al botón de guardar se registra en la tabla de la base de datos para posteriormente regresar a FRegistro si la operación se realizó exitosamente.
    - String lNumeroLista = tfNumeroLista.getText();
    - String lNombre = tfNombre.getText();
    - String lApellido = tfApellido.getText();
    - String lSexo = tfSexo.getText();
    - String lCarrera = tfCarrera.getText();
    - String lMateria = tfMateria.getText();
    - String lPlan = tfPlan.getText();
    - String lD1 = tfD1.getText();
    - String lD2 = tfD2.getText();
    - String lD3 = tfD3.getText();
    - String lPromedio = tfPromedio.getText();
    - String lSecuencia = tfSecuencia.getText();
    - String lEquipo = tfEquipo.getText();
    - String sql = "INSERT INTO alumnos(NumeroLista, Nombre, ApellidoPaterno, Sexo, Carrera, Materia, PlanEstudios, D1, D2, D3, PromedioR, Secuencia, Noequipo) VALUES('"+lNumeroLista+"','"+lNombre+"','"+lApellido+"','"+lSexo+"','"+lCarrera+"','"+lMateria+"','"+lPlan+"','"+lD1+"','"+lD2+"','"+lD3+"','"+lPromedio+"','"+lSecuencia+"','"+lEquipo+"')";
    - try {
    - stmt = conn.createStatement();
    - if(stmt.executeUpdate(sql) == 1){
    - JOptionPane.showMessageDialog(null,"Se añadio con exito","ALERTA",JOptionPane.INFORMATION_MESSAGE);
    - // Automaticamente regresara a FRegistro para ver el nuevo dato agregado

```
            new FRegistro().setVisible(true);
            this.setVisible(false);
            this.dispose();
          } else {
            JOptionPane.showMessageDialog(null,"Error al
añadir","ALERTA",JOptionPane.WARNING_MESSAGE);
          }

        } catch(Exception ex) {
          System.out.println("Error: "+ex);
        }
```

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 30)

private void btnAtrasActionPerformed(java.awt.event.ActionEvent evt)

- Regresa a FRegistro
- Recibe el click del botón atras de tipo JButton
- Click del usuario
  - new FRegistro().setVisible(true);
  - this.setVisible(false);
  - this.dispose();
- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 39)

private void btnLimpiarActionPerformed(java.awt.event.ActionEvent evt)

- Limpia los campos de cada JTextField.
- Recibe el click del botón limpiar.
  - tfNumeroLista.setText(null);
  - tfNombre.setText(null);
  - tfApellido.setText(null);
  - tfSexo.setText(null);
  - tfCarrera.setText(null);
  - tfMateria.setText(null);
  - tfPlan.setText(null);
  - tfD1.setText(null);
  - tfD2.setText(null);
  - tfD3.setText(null);
  - tfPromedio.setText(null);
  - tfSecuencia.setText(null);
  - tfEquipo.setText(null);
  - //Nos localiza en el campo 1
  - tfNumeroLista.requestFocus();
- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 29)

CLASE FEDITAR

private void formWindowOpened(java.awt.event.WindowEvent evt)

- Se inicializa al abrir el formulario
- Recibe la conexión definida en DBConexion y los datos tipo String seleccionados por el usuario de la Tabla de FRegistro realizando una consulta sql a la base de datos.
  - conn = DBConexion.DBConexión();

```
        String sql = "SELECT * FROM alumnos WHERE NumeroLista =
"+id+"";
        try {
          PStatement = conn.prepareStatement(sql);
          rs = PStatement.executeQuery();
          if(rs.next()){
            tfNumeroLista.setText(rs.getString("NumeroLista"));
            tfNombre.setText(rs.getString("Nombre"));
            tfApellido.setText(rs.getString("ApellidoPaterno"));
            tfSexo.setText(rs.getString("Sexo"));
            tfCarrera.setText(rs.getString("Carrera"));
            tfMateria.setText(rs.getString("Materia"));
            tfPlan.setText(rs.getString("PlanEstudios"));
            tfD1.setText(rs.getString("D1"));
            tfD2.setText(rs.getString("D2"));
            tfD3.setText(rs.getString("D3"));
            tfPromedio.setText(rs.getString("PromedioR"));
            tfSecuencia.setText(rs.getString("Secuencia"));
            tfEquipo.setText(rs.getString("NoEquipo"));
          }
        } catch(Exception ex) {
          System.out.println("Error: "+ex);
        }
```

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 30)

private void btnActualizarActionPerformed(java.awt.event.ActionEvent evt)

- Actualiza los datos de un usuario seleccionado en la tabla de FRegistro
- Recibe los datos de tipo String y los imprime en los JTextField, para su posterior edición, una vez editados al pulsar el botón estos datos son enviados nuevamente y sql realiza la operación update en la tabla de la base de datos, una vez concluido esto regresa a FRegistro automáticamente.

```
        String lNumeroLista = tfNumeroLista.getText();
        String lNombre = tfNombre.getText();
        String lApellido = tfApellido.getText(); //Aqui
        String lSexo = tfSexo.getText();
        String lCarrera = tfCarrera.getText();
        String lMateria = tfMateria.getText();
        String lPlan = tfPlan.getText();
        String lD1 = tfD1.getText();
        String lD2 = tfD2.getText();
        String lD3 = tfD3.getText();
        String lPromedio = tfPromedio.getText();
        String lSecuencia = tfSecuencia.getText();
        String lEquipo = tfEquipo.getText();
        String sql = "UPDATE alumnos SET NumeroLista='"+lNumeroLista+"',
Nombre='"+lNombre+"', ApellidoPaterno='"+lApellido+"', Sexo='"+lSexo+"',
```

Carrera='"+lCarrera+"', Materia='"+lMateria+"', PlanEstudios='"+lPlan+"', D1='"+lD1+"', D2='"+lD2+"', D3='"+lD3+"', PromedioR='"+lPromedio+"', Secuencia='"+lSecuencia+"', NoEquipo='"+lEquipo+"' WHERE NumeroLista = "+id+"";

```
        try {
          stmt = conn.createStatement();
          if(stmt.executeUpdate(sql) == 1){
            JOptionPane.showMessageDialog(null,"Se actualizo con exito","ALERTA",JOptionPane.INFORMATION_MESSAGE);
            // Automaticamente regresara a FRegistro una vez actualizado el dato.
            new FRegistro().setVisible(true);
            this.setVisible(false);
            this.dispose();
          } else {
            JOptionPane.showMessageDialog(null,"Error al editar","ALERTA",JOptionPane.WARNING_MESSAGE);
          }
        } catch(Exception ex) {
          System.out.println("Error: "+ex);
        }
```

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 30)

private void btnAtrasActionPerformed(java.awt.event.ActionEvent evt)

- Regresa a FRegistro
- Recibe el click del botón atrás de tipo JButton

```
        new FRegistro().setVisible(true);
        this.setVisible(false);
        this.dispose();
```

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 30)

private void btnLimpiarActionPerformed(java.awt.event.ActionEvent evt)

- Limpia los campos de cada JTextField

```
        tfNumeroLista.setText(null);
        tfNombre.setText(null);
        tfApellido.setText(null);
        tfSexo.setText(null);
        tfCarrera.setText(null);
        tfMateria.setText(null);
        tfPlan.setText(null);
        tfD1.setText(null);
        tfD2.setText(null);
        tfD3.setText(null);
        tfPromedio.setText(null);
        tfSecuencia.setText(null);
        tfEquipo.setText(null);
        tfNumeroLista.requestFocus();
```

- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 30)

CLASE FVISUALIZAR

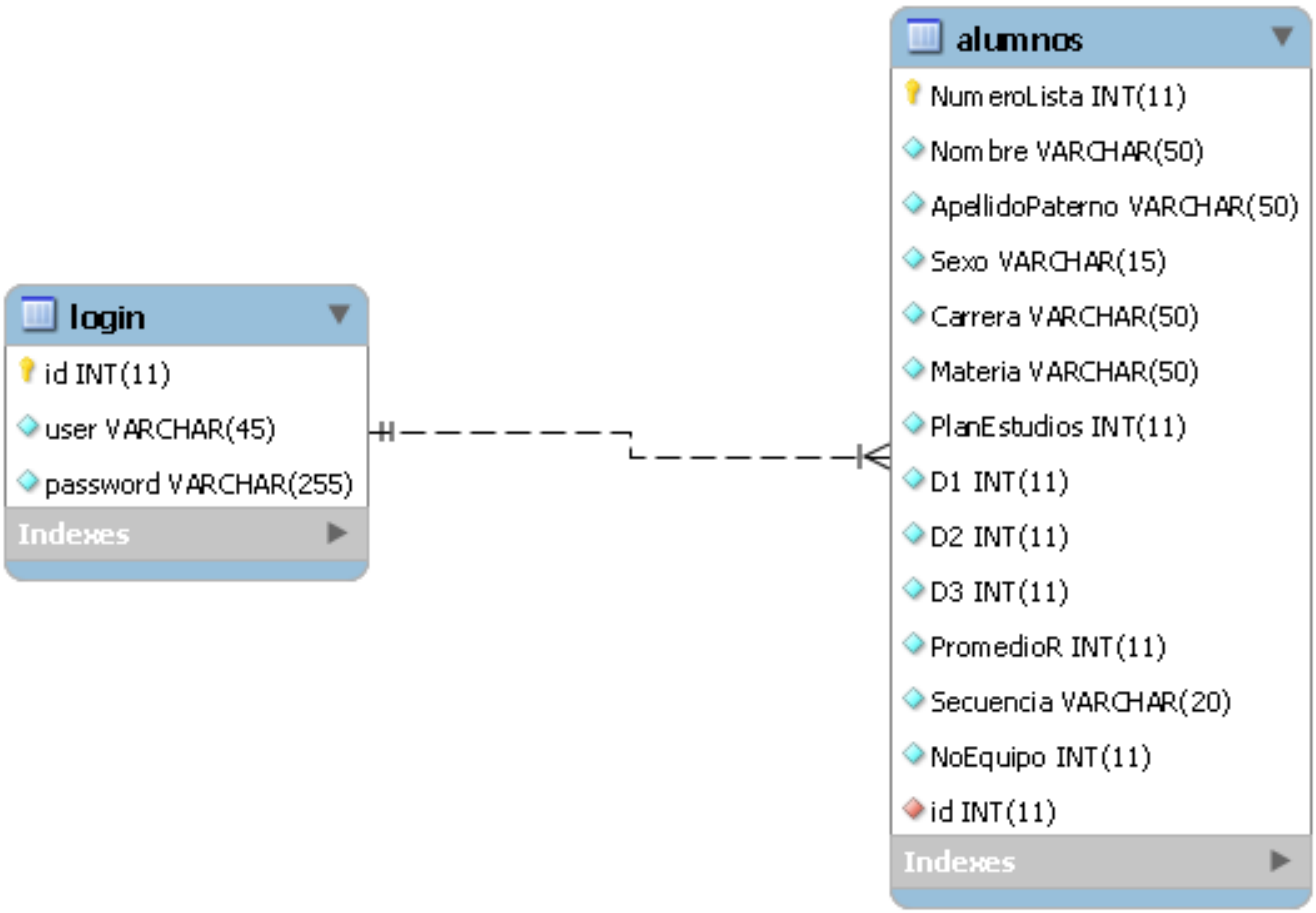
private void formWindowOpened(java.awt.event.WindowEvent evt)

- Se inicializa al abrir el formulario y muestra los seleccionados en FRegistro, estos datos no pueden ser modificados solo vistos ya que se imprimen en JLabel
- Recibe los datos de tipo String de la tabla de FRegistro, hace una consulta sql y los imprime en cada JLabel con la conexión definida en DBConexion.
    - conn = DBConexion.DBConexión();
    - String sql = "SELECT * FROM alumnos WHERE NumeroLista = "+id+"";
    - try {
    - PStatement = conn.prepareStatement(sql);
    - rs = PStatement.executeQuery();
    - if(rs.next()){
    - lbNumeroLista.setText(rs.getString("NumeroLista"));
    - lbNombre.setText(rs.getString("Nombre"));
    - lbApellido.setText(rs.getString("ApellidoPaterno"));
    - lbSexo.setText(rs.getString("Sexo"));
    - lbCarrera.setText(rs.getString("Carrera"));
    - lbMateria.setText(rs.getString("Materia"));
    - lbPlan.setText(rs.getString("PlanEstudios"));
    - lbD1.setText(rs.getString("D1"));
    - lbD2.setText(rs.getString("D2"));
    - lbD3.setText(rs.getString("D3"));
    - lbPromedio.setText(rs.getString("PromedioR"));
    - lbSecuencia.setText(rs.getString("Secuencia"));
    - lbEquipo.setText(rs.getString("NoEquipo"));
    - }
    - } catch(Exception e) {
    - System.out.println("Error en: "+e);
    - }
- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 30)

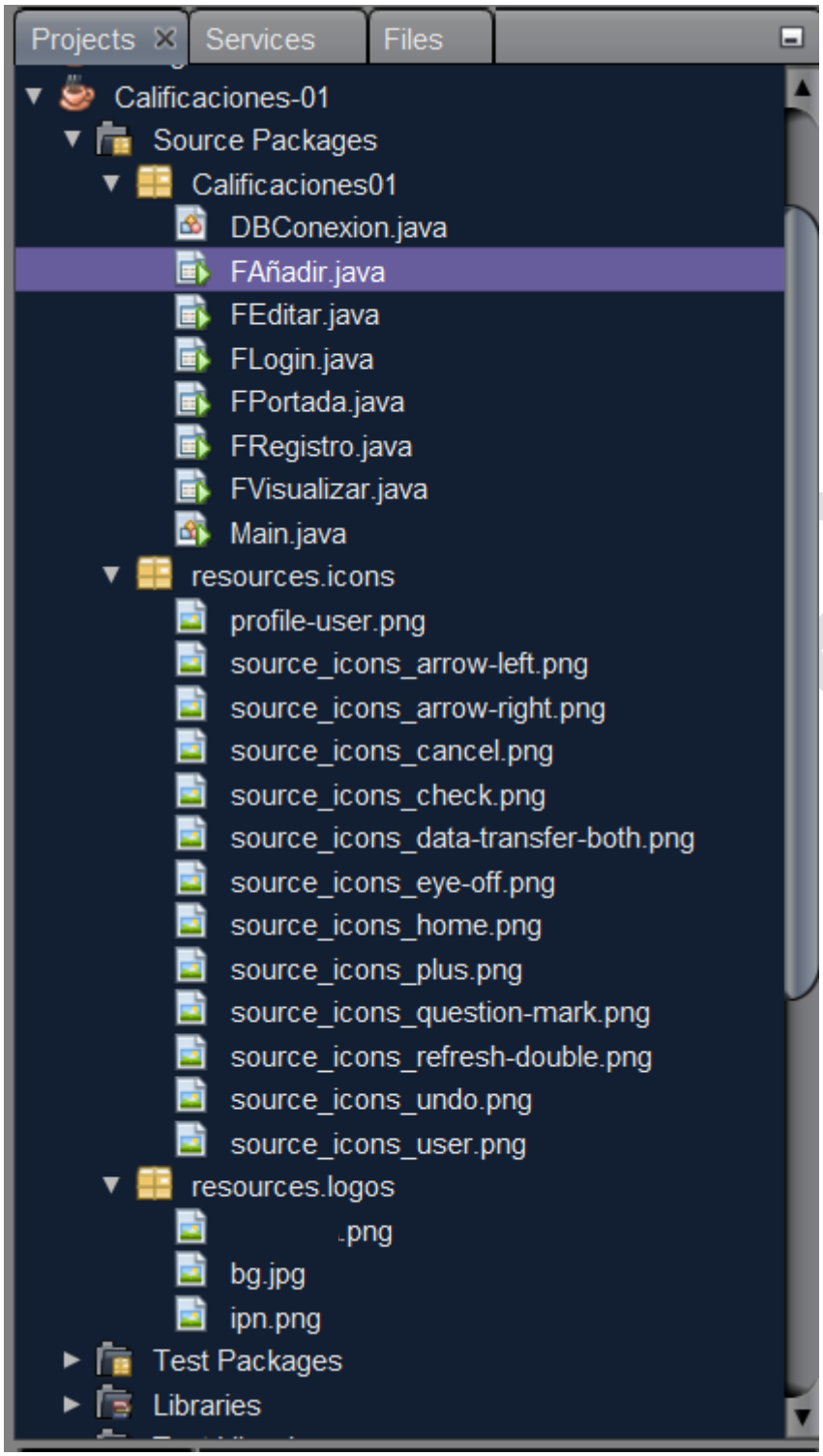
private void btnAtrasActionPerformed(java.awt.event.ActionEvent evt)

- Regresa a FRegistro
- Recibe el click del botón atrás de tipo JButton
    - new FRegistro().setVisible(true);
    - this.setVisible(false);
    - this.dispose();
- Las pruebas del método se adjuntan en el apartado de capturas (Pag. 30)

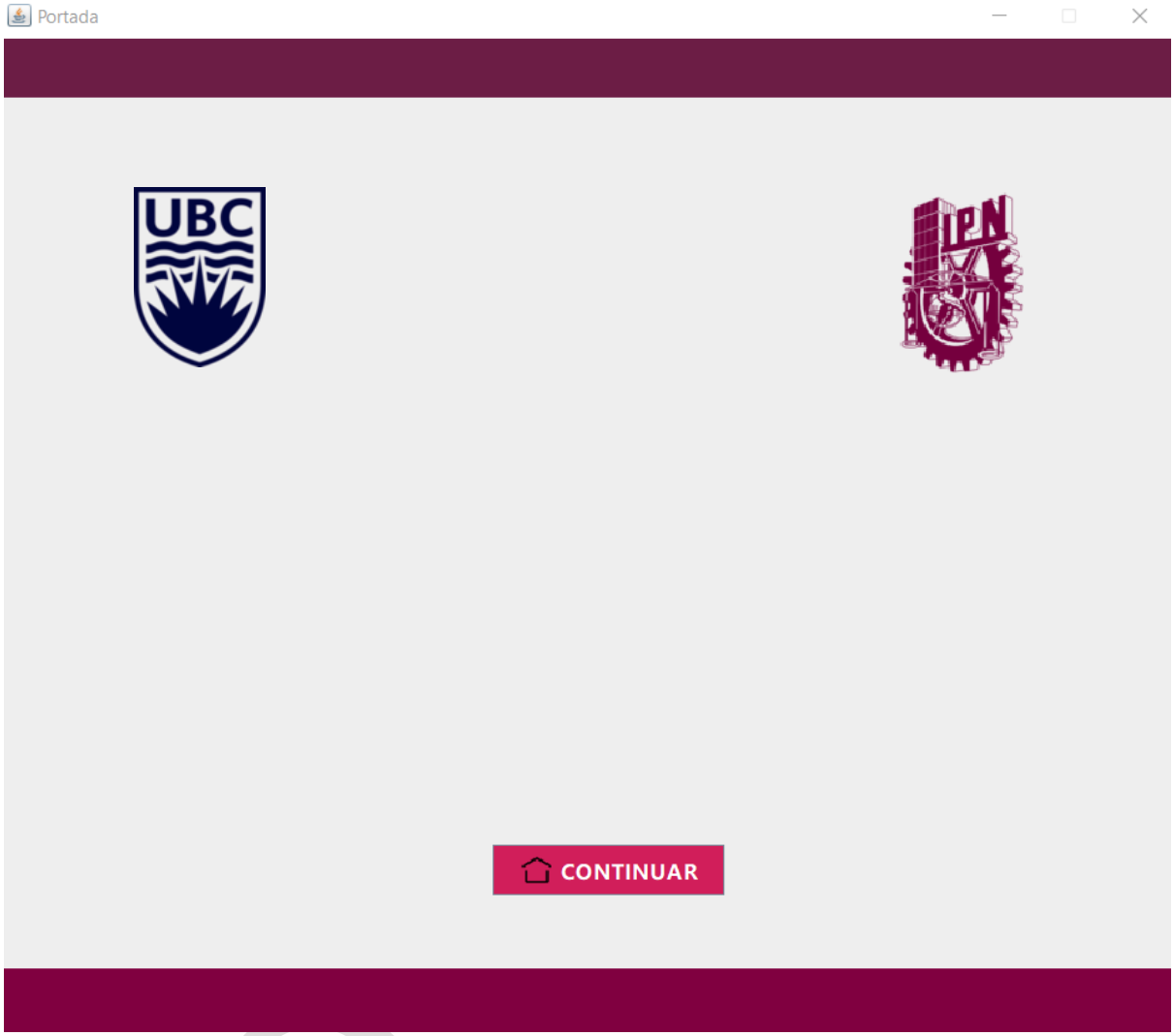
Tablas de la Base de Datos



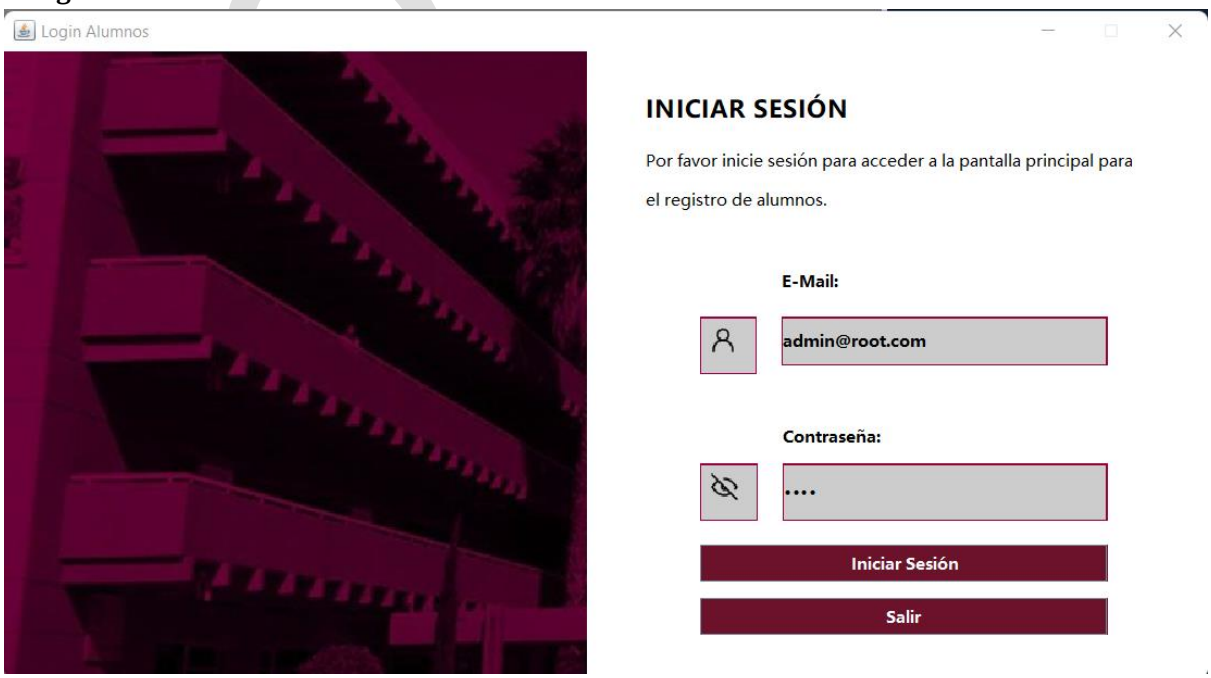
Capturas de pantalla



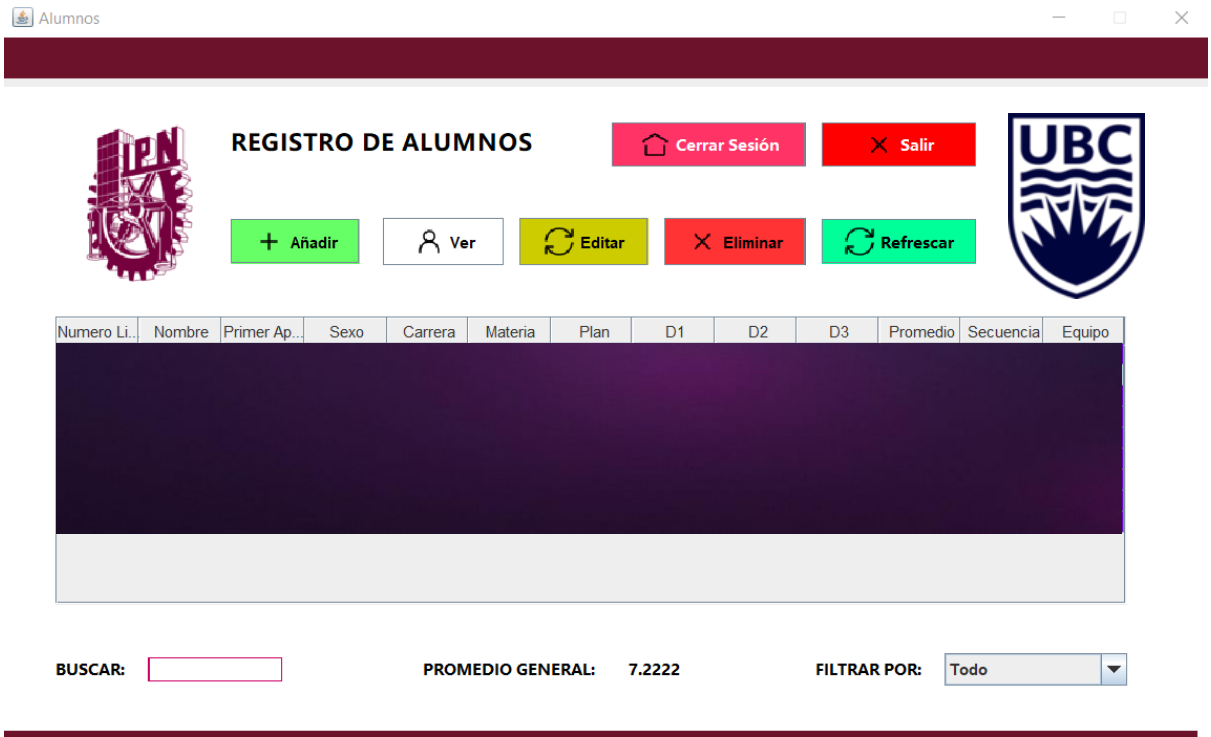
FPortada

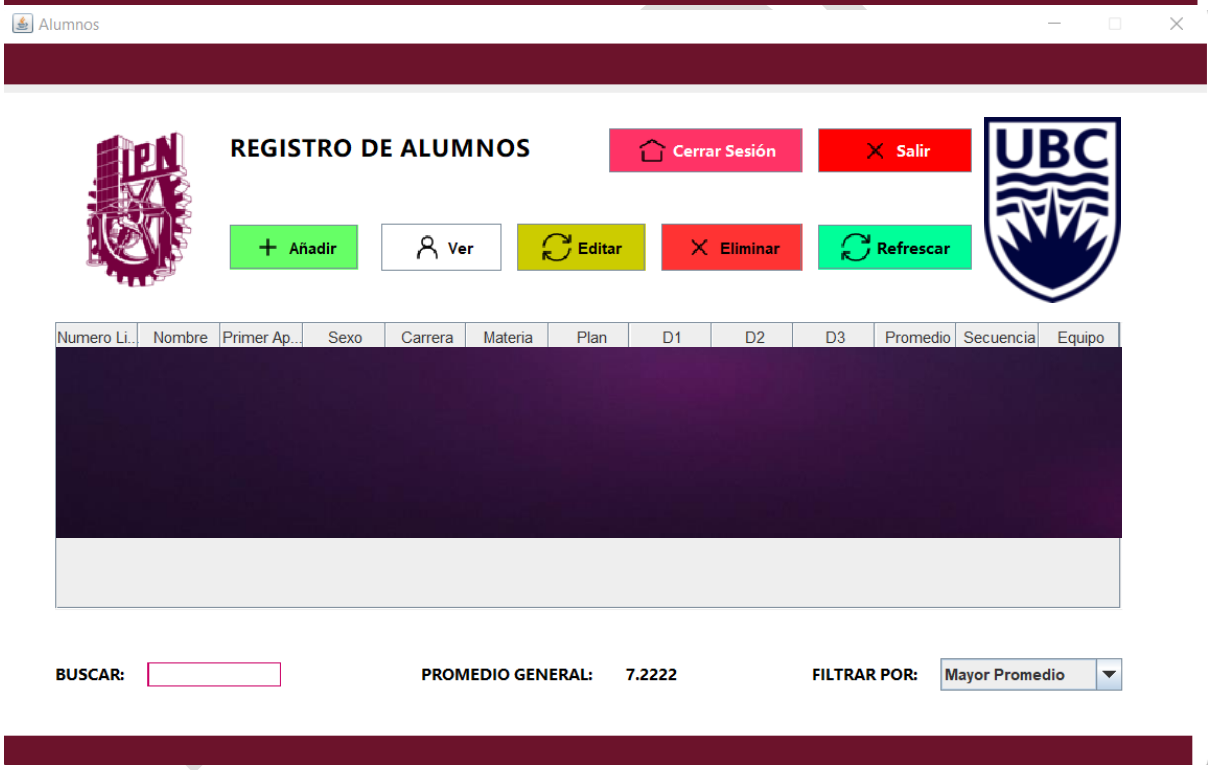


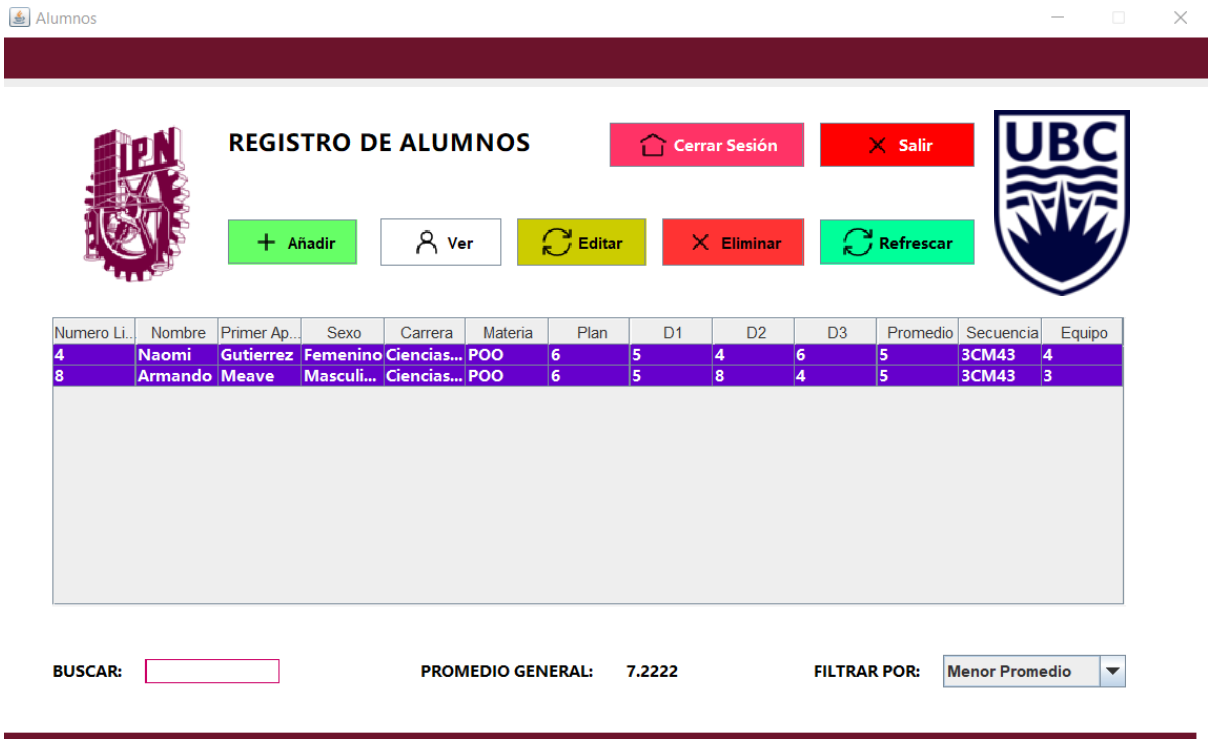
FLogin

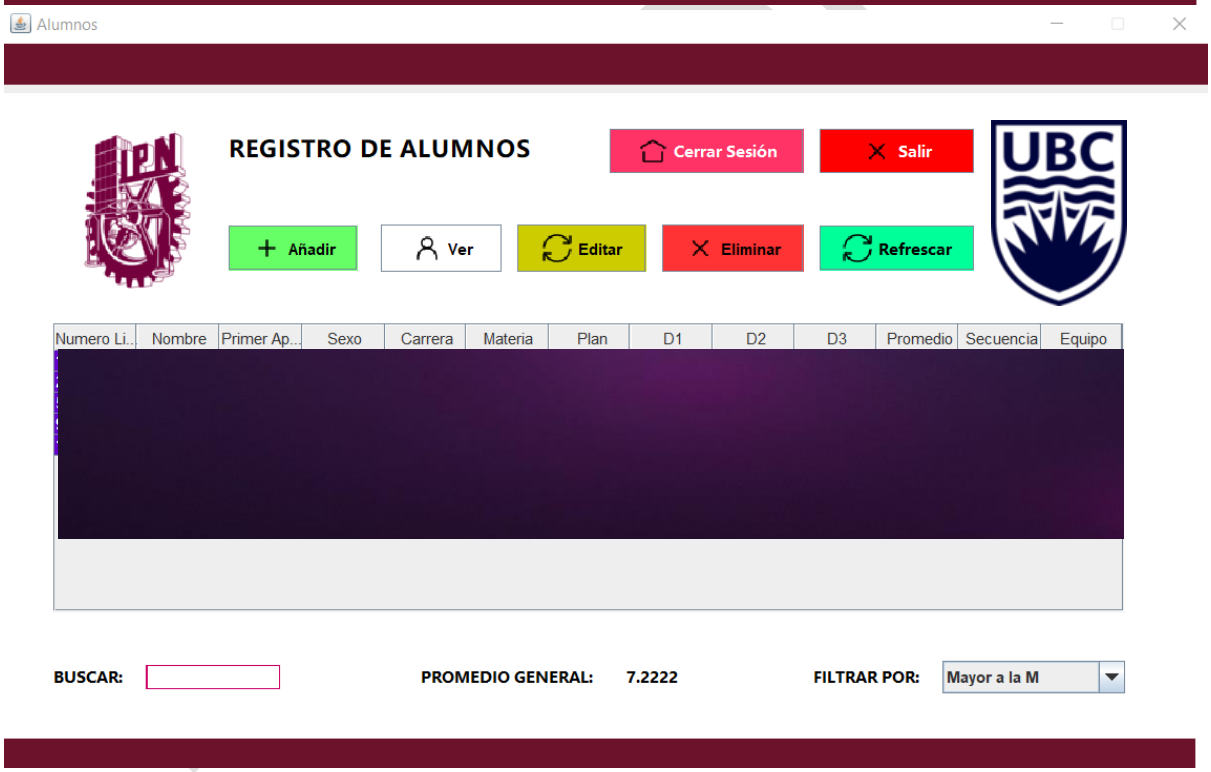


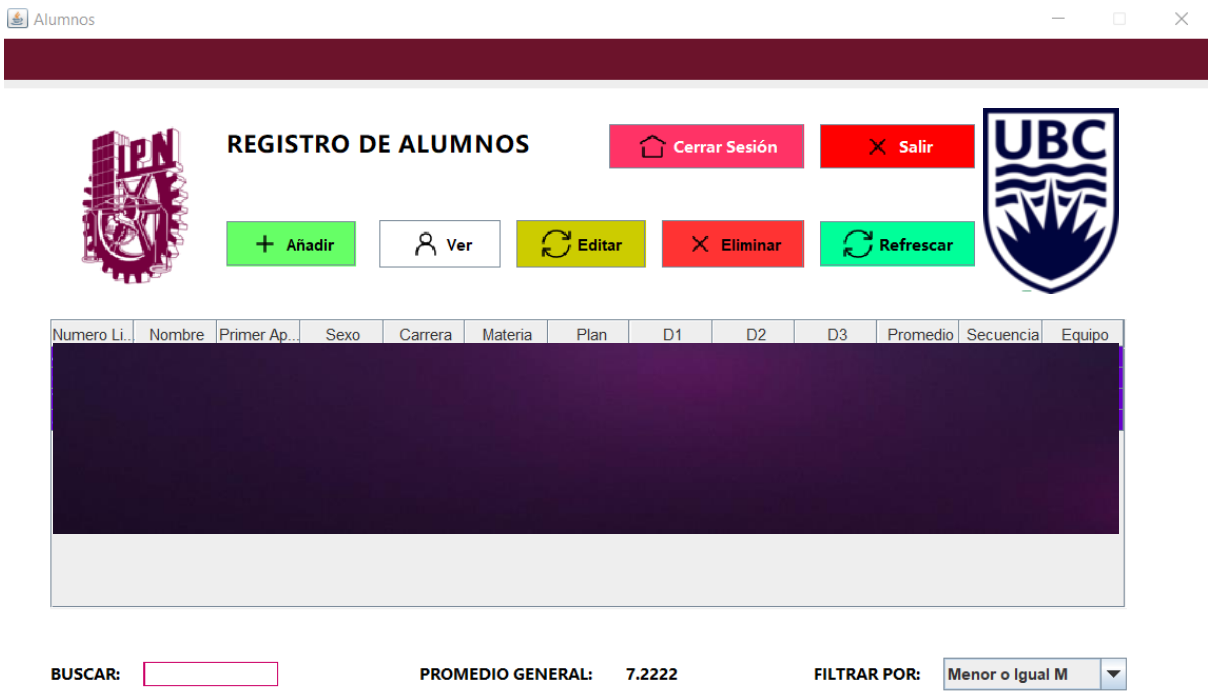
FRegistro


Alumnos
REGISTRO DE ALUMNOS
Cerrar Sesión
Salir
Añadir
Ver
Editar
Eliminar
Refrescar
UBC
Numero Li..
Nombre
Primer Ap...
Sexo
Carrera
Materia
Plan
D1
D2
D3
Promedio
Secuencia
Equipo
BUSCAR:
PROMEDIO GENERAL:
7.2222
FILTRAR POR:
Todo

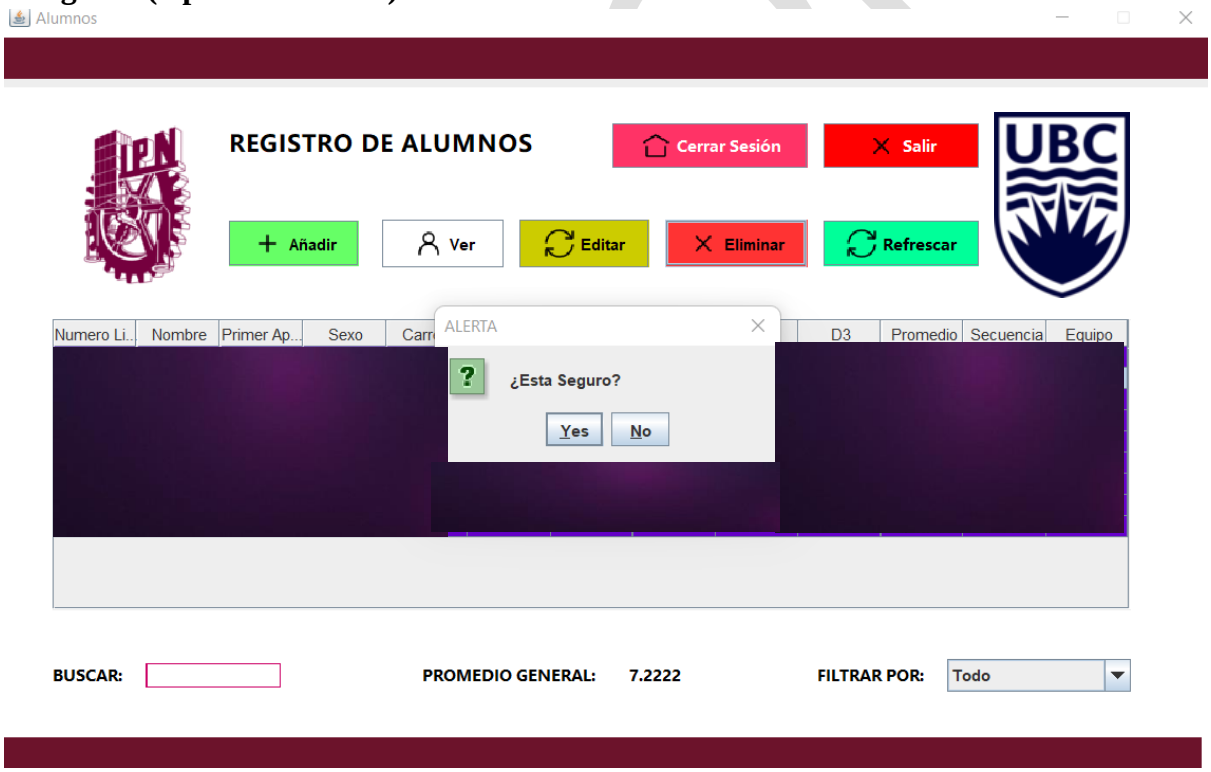

Alumnos
REGISTRO DE ALUMNOS
Cerrar Sesión
Salir
Añadir
Ver
Editar
Eliminar
Refrescar
UBC
Numero Li..
Nombre
Primer Ap...
Sexo
Carrera
Materia
Plan
D1
D2
D3
Promedio
Secuencia
Equipo
BUSCAR:
PROMEDIO GENERAL:
7.2222
FILTRAR POR:
Mayor Promedio


Alumnos
REGISTRO DE ALUMNOS
Cerrar Sesión
Salir
Añadir
Ver
Editar
Eliminar
Refrescar
Numero Li.. Nombre Primer Ap... Sexo Carrera Materia Plan D1 D2 D3 Promedio Secuencia Equipo
4 Naomi Gutierrez Femenino Ciencias... POO 6 5 4 6 5 3CM43 4
8 Armando Meave Masculi... Ciencias... POO 6 5 8 4 5 3CM43 3
BUSCAR:
PROMEDIO GENERAL: 7.2222
FILTRAR POR: Menor Promedio


Alumnos
REGISTRO DE ALUMNOS
Cerrar Sesión
Salir
Añadir
Ver
Editar
Eliminar
Refrescar
Numero Li.. Nombre Primer Ap... Sexo Carrera Materia Plan D1 D2 D3 Promedio Secuencia Equipo
BUSCAR:
PROMEDIO GENERAL: 7.2222
FILTRAR POR: Mayor a la M

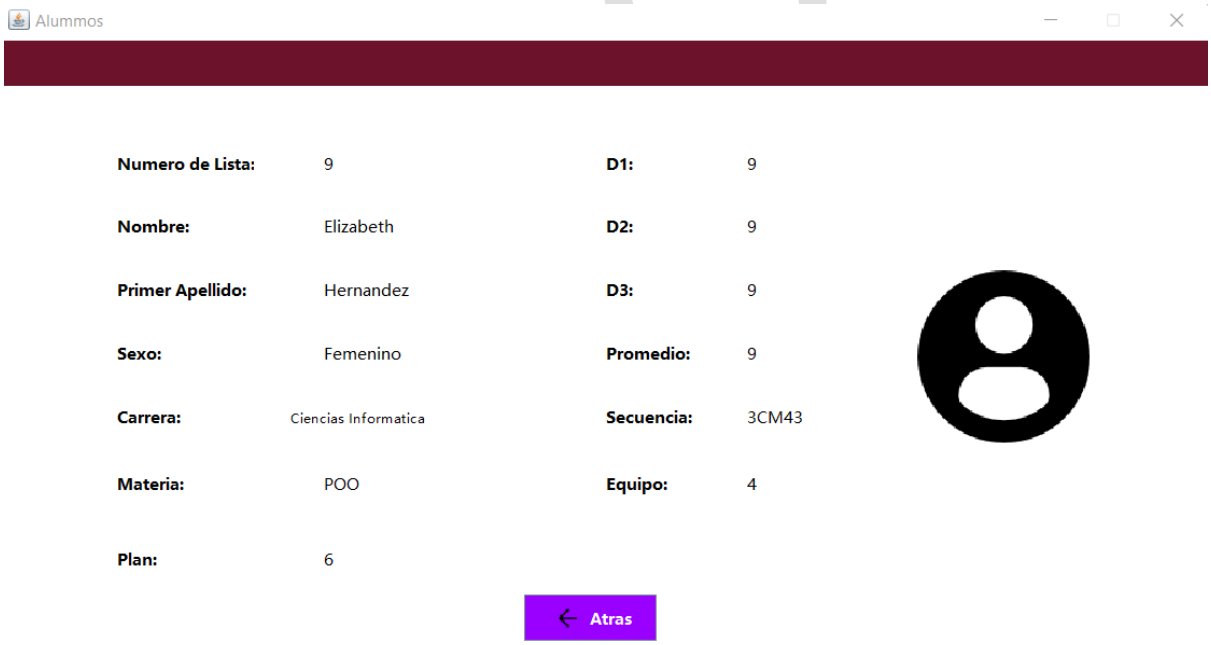


FRegistro (Opción Eliminar)

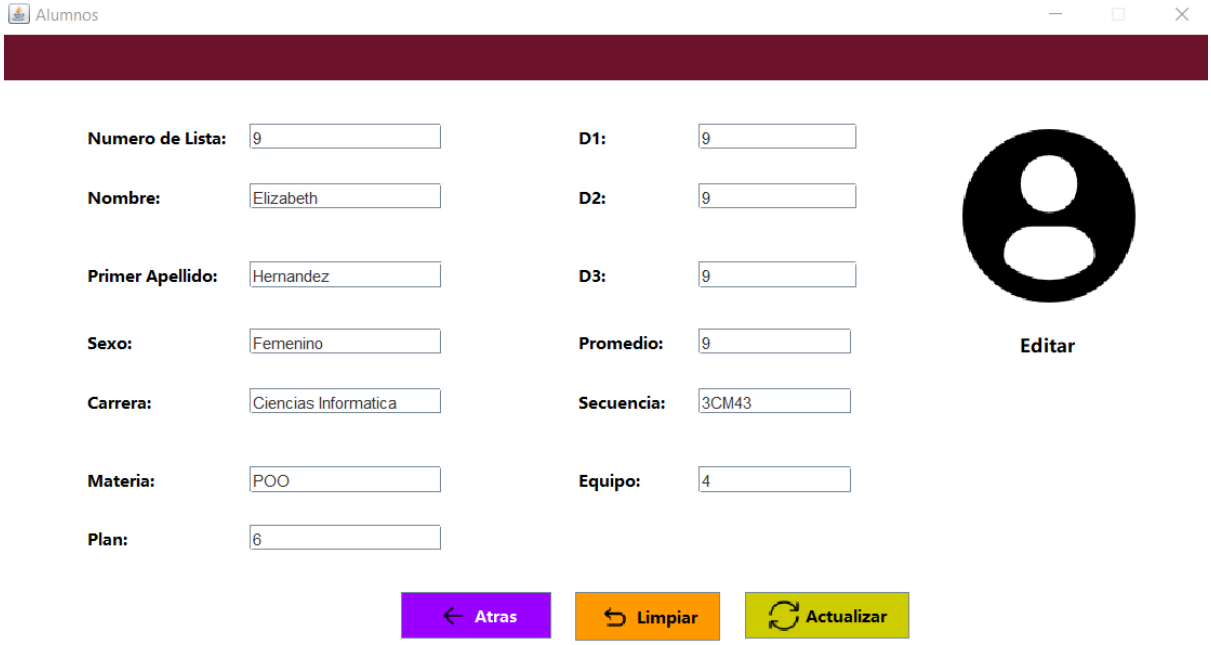


FRegistro (Buscador)

FVisualizar



FEditar



FAñadir