Documentación Técnica

# DESCRIPCIÓN GENERAL DE LA APLICACIÓN:

Se trata de un programa de escritorio escrito en Java, y con una interfaz intuitiva de Swing, con una conexión a base de datos, con el driver de JDBC para mysql.

El programa tiene como función principal la gestión de un registro de estudiantes, con las diferentes funcionalidades necesarias para una gestión simple pero eficaz, es decir, se pueden crear, eliminar, modificar, consultar y filtrar registros.

# ESTRUCTURA DEL PROYECTO:

Diagrama de paquetes y clases:

* Principal
  + ConexiónDataBase.java
  + Principal.java
* Pantallas
  + Detalle.java
  + Inicio.java
  + Opciones.java
  + Registro.java
  + RenderCeldas.java

Build path:

* JRE JavaSE-19 (JDK 19)
* mysql-connector-j-8.0.33.jar

Extra:

* conexion.properties

Como se puede observar en los listados anteriores el proyecto tiene una estructura, libre pero organizada donde existen dos paquetes diferenciados para separar la parte visual y sus derivados y la parte trasera y sus derivados, pantallas y principal, respectivamente.

Comenzando por las pantallas, hay 4 ficheros que extienden la clase Jframe, que son la parte front de la aplicación, la de inicio utiliza un layaout de bordes, para fragmentar y ordenar las partes de la pantalla por paneles, estos paneles son un Scrollpanel y un panel con grid layout. La pantalla de detalle tiene un layout absolute y a sido ajustada a mano, esta tiene en su constructor como obligación obtener ciertos datos, como la posición a la que debe aparecer cuando se instancia, (que es desde principal en el submenú de opciones), o los datos que debe mostrar es decir, en función de la fila donde se accede a las opciones. La de opciones es una pantalla auxiliar de la pantalla principal, que ocupa poco espacio y solo tiene tres botones. La de registro es una pantalla con layout absolute ajustada a mano, donde podemos encontrar un formulario.

Aparte de las pantallas en el paquete de pantallas también se encuentra la clase RenderCeldas, que implementa TableCellEditor, KeyListener, y tiene como función exclusiva modificar las acciones que suceden al clicar en la tabla, esto se consigue modificando los diferentes métodos que se implementan. Esta clase tiene su aparición cuando se crea la tabla de inicio.

Por ultimo en el otro paquete, encontramos la clase ConexiónDataBase que generara la conexión hacia la base de datos usando el driver de JDBC, y que mediante a la clase principal, será la que le de a las pantallas la opción de obteber o modificar datos de la base de datos. Y la clase principal que es donde se encuentra el main, y las principales constante y variables que serán comunes para las diferentes clases de la aplicación, es decir que hará de nexo entre las pantallas, la gestión de las misma, y la base de datos. Cabe destacar que existirán en esta las instancias de la pantalla de inicio, de la vista que se muestra por encima de la inicial, declarada como cache. Y la conexión a la base de datos.

# Instrucciones de instalación y configuración (incluyendo la base de datos):

Para la instalación del proyecto en Windows, es necesario tener instalado java, al menos de la versión del JDk 19, y para utilizar la base de datos, que debe estar en local, debes abrir abrir y ejecutar el archivo sql en tu administrador de my sql, aparte deberar modificar el archivo configuración.properties, para que los datos coincidan, puesto que si no dará error al ejecutar el programa.

# Explicación detallada de cómo se establece la conexión JDBC y se realizan las operaciones CRUD:

La conexión sql se establece como antes menciones en la explicación de las clases en la clase ConexionDataBase, utilizando el driver JDBC para mysql. El objeto de esta clase que se encuentra en una constante de la clase principal, devolverá con el método get conexión la conexión, de tal manera que cada clase pueda realizar sus propias consultas o actualizaciones.

Se destaca que ya que no existe riego de inserción de sql, puesto a que se trata de un administrador donde no son necesario ni permisos ni existen restricciones a la hora de acceder a los datos, se ha utilizado statements base.

Se enumeran todas las sentencias utilizadas en el código, y se marcan con un asterisco las palabras que se tratan de variables para su posterior explicación.

1. Consulta de la clase inicio:
   1. Código:

java.sql.Statement st = Principal.***BD***.getCon().createStatement();

consulta = st.executeQuery(C\*1);

java.sql.ResultSetMetaData metaData = consulta.getMetaData();

* 1. Consultas: \*1 :
     1. Select \* from estudiantes(inicialización de la pantalla)
     2. "select \* from estudiantes where " + selector.getSelectedItem()\*2 + " Like '%"+ buscador.getText()\*3 + "%'"
     3. Select \* from estudiantes (Método recargar)

\*2  = Selector.getSelectedItem() hace referencia a la opción seleccionada en el JCombox que se encuentra cerca del buscador.

\*3 = buscador.getText() hace referencia al texto escrito por el usuario en el Jtextfield que se usa de buscador.

1. Consultas clase detalle:
   1. Código:

try {

java.sql.Statement st = Principal.***BD***.getCon().createStatement();

consulta = st.executeQuery(C\*4);

columns.add("Datos");

columns.add("Informacion");

for (int i = 0; i < 4; i++) {

Vector<String> row = new Vector<>();

data.add(row);

}

data.get(0).add("ID");

data.get(1).add("Nombre");

data.get(2).add("Edad");

data.get(3).add("Curso");

while (consulta.next()) {

for (int i = 1; i <= 4; i++) {

data.get(i - 1).add(consulta.getString(i));

}}}

* 1. Consultas:\*4:
     1. Select \* from estudiantes where ID=" + id\*5

\*5 = id es un parámetro del constructor de detalle que hace referencia al id del registro a mostrar.

1. Actualizaciones de tabla en registro:
   1. Código:

java.sql.Statement st = Principal.***BD***.getCon().createStatement();

String query = "INSERT INTO Estudiantes (Nombre,edad,curso) VALUES ('" + Nombre.getText()\*6 + "','"

+ Edad.getText()\*7 + "','" + Curso.getText()\*8 + "')";

st.executeUpdate(query);

Principal.*PantallaCache*.setVisible(false);

Principal.*PantallaCache* = null;

* 1. Updates:
     1. INSERT INTO Estudiantes (Nombre,edad,curso) VALUES ('" + Nombre.getText() + "','"+ Edad.getText() + "','" + Curso.getText() + "')";

\*6 = Nombre.getText() hace referencia al valor introduccido por el usuario en el TextFiled con sobre nombre Nombre.

\*7 = Edad.getText() hace referencia al valor introduccido por el usuario en el TextFiled con sobre nombre Edad.

\*8 = Curso.getText() hace referencia al valor introduccido por el usuario en el TextFiled con sobre nombre Curso.

# Referencias:

A la hora de hablar de las referencias he de hacer especial mención a chatGPT puesto a que me ha ayudado a la hora de generar los datos de ejemplo, revisar las sentencias SQL, y a crear la clase RenderCeldas. Aparte de la IA, también consulte varios blogs, de los cuales ya no tengo los nombres para solucionar conflictos con el driver de JDBC.