Manual de técnico del sistema biblioteca

Hilary Jazmin Rompich Pirir
Carné: 7590-21-3903
Programación I
31 de mayo de 2024

Introducción

El propósito de esta guía técnica es proporcionar instrucciones paso a paso sobre cómo desarrollar, configurar y mantener un proyecto JavaFX conectado a una base de datos utilizando principios de programación orientada a objetos (POO). Este proyecto, llamado Biblioteca, es una aplicación de administración de libros y usuarios, que permite al administrador realizar operaciones CRUD (crear, leer, actualizar, eliminar) en elementos de libros y usuarios.
Permite al cliente crear cuenta, ingresar con su usuario y contraseña, realizar búsqueda de libros a la base de datos y finalmente realizar una transacción de préstamo de libros.

El objetivo principal de este proyecto es crear una aplicación de escritorio eficiente y fácil de usar para la gestión de bibliotecas. Esta aplicación permite a los administradores de bibliotecas gestionar libros y usuarios de forma eficaz. Con JavaFX para la interfaz gráfica de usuario (GUI), PostgreSQL como sistema de gestión de bases de datos y principios de programación orientada a objetos, este proyecto tiene como objetivo proporcionar una solución a la administración de libros, clientes y transacciones.

Alcance

Esta guía está dirigida a desarrolladores y técnicos que puedan mejorar o realizar dicho sistema. El manual está dividido en secciones claramente definidas para facilitar la navegación y la referencia. Cada sección incluye explicaciones detalladas, ejemplos de código, diagramas, para ilustrar conceptos y procesos. Esta guía técnica tiene como objetivo proporcionar un recurso completo y útil para cualquier desarrollador o técnico que trabaje en un proyecto de Biblioteca, brindando una comprensión profunda de su arquitectura y creando condiciones favorables para futuros mantenimientos y mejoras del sistema.

Diagramas de casos de uso

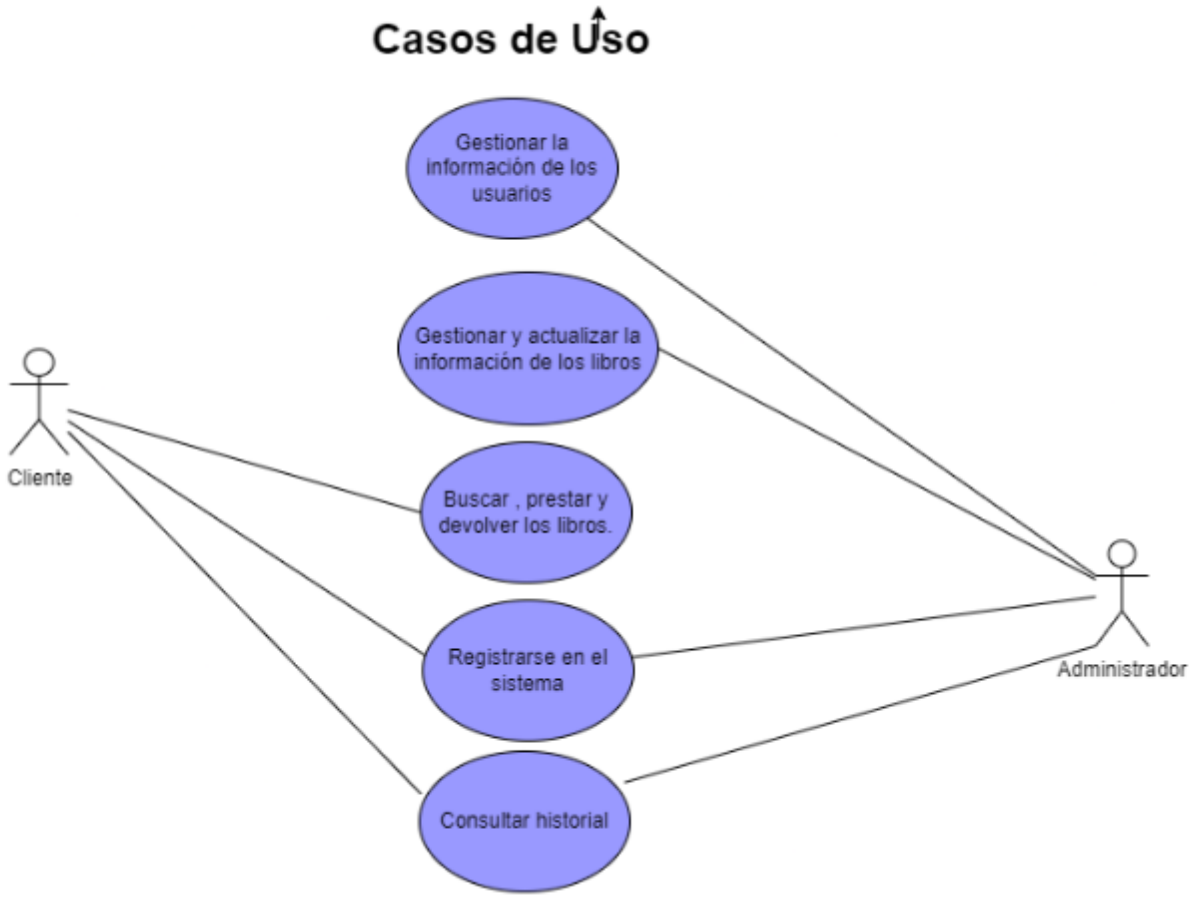


Diagrama de clases

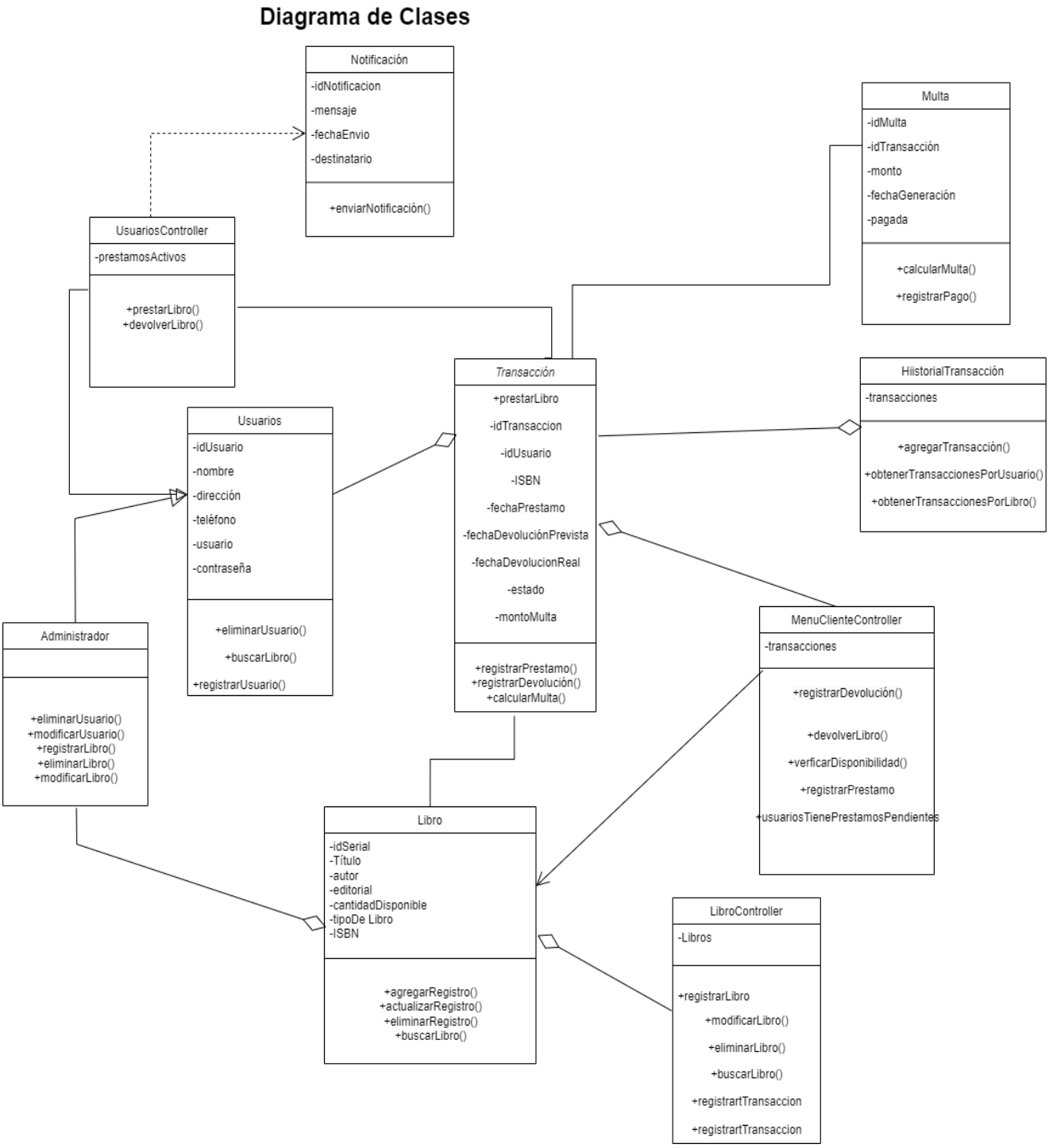
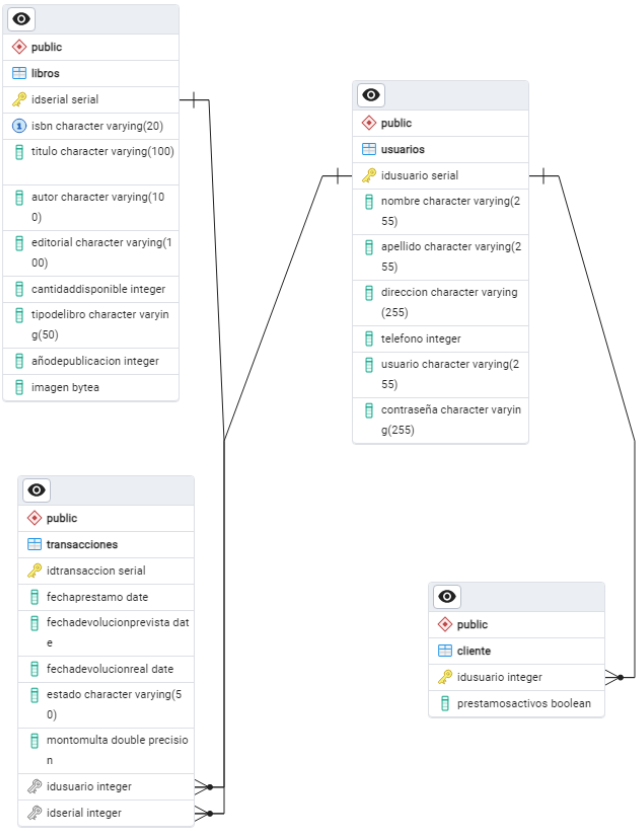


Diagrama de base de datos



Tecnologías utilizadas

1. Java: Lenguaje de programación principal utilizado para desarrollar la lógica de la aplicación.
2. JavaFX: Biblioteca gráfica utilizada para crear la interfaz del usuario.
3. PostgreSQL: Sistema de gestión de bases de datos relacional utilizado para almacenar y gestionar los datos.
4. JDBC (Java Database Connectivity): API utilizada para conectar la aplicación Java con la base de datos PostgreSQL.
5. NeatBeans: Entorno de desarrollo integrado (IDE) utilizado para el desarrollo del proyecto.
6. Git y GitHub: Herramientas utilizadas para el control de versiones y la colaboración en el desarrollo del código fuente.

Instrucciones de configuración del entorno de trabajo

Requisitos previos

- Java Development Kit (JDK) 22: Asegúrate de tener JDK 22 instalado en tu sistema.
- NetBeans 21: Instala NetBeans 21 en tu computadora.
- Scene Builder: Descarga e instala Scene Builder desde el sitio oficial de Gluon (https://gluonhq.com/products/scene-builder/).

Procesos de instalación

Paso 1: Descargar el Proyecto desde el Repositorio

1. Accede al repositorio que contiene el proyecto JavaFX Maven en GitHub.
   Enlace:
   https://github.com/HilaryRompich2021/PROYECTO_BIBLIOTECA

2. Haz clic en el botón "Code" y selecciona "Download ZIP" para descargar el proyecto como un archivo ZIP.
3. Descomprime el archivo ZIP descargado en una ubicación de tu elección en tu computadora.

Paso 2: Importar el Proyecto en NetBeans

1. Abre NetBeans 21 en tu computadora.
2. En el menú principal, selecciona File -> Open Project.
3. Navega hasta el directorio donde descomprimiste el proyecto descargado y haz clic en "Open Project".
4. NetBeans importará automáticamente el proyecto y lo mostrará en la ventana del Proyecto.

Paso 3: Configurar la Plataforma Java y JDK en NetBeans

1. Verifica que NetBeans esté configurado para utilizar el JDK adecuado:
2. Ve a Tools -> Options -> Java.
3. En la pestaña Java Platform, asegúrate de que el JDK 22 esté seleccionado como la Plataforma Java predeterminada.

Paso 4: Descargar PostgreSQL

1. Dirigirse al sitio oficial: https://www.postgresql.org/download/
2. Seleccione el sistema operativo de su máquina y siga las instrucciones para descargar el instalador adecuado.

Paso 5: Descargar el controlador JDBC

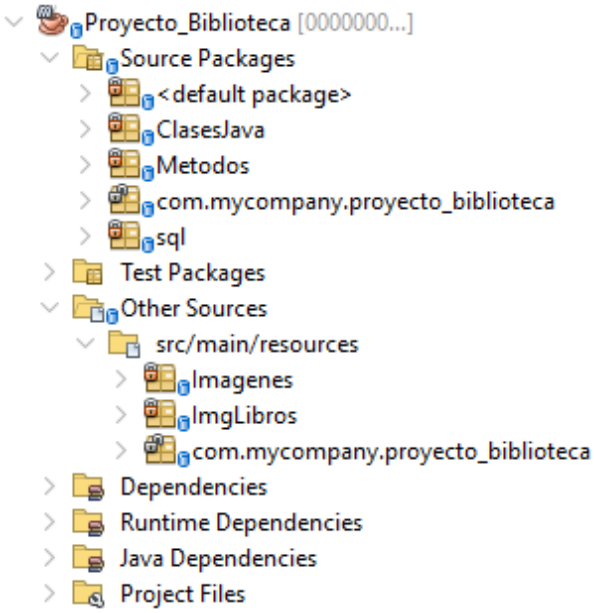
1. Dirigirse al sitio oficial de PostgreSQL
   https://jdbc.postgresql.org/download.html
2. Descargue el archivo ‘postgresql-.jar’

Paso 6: Compilar y Ejecutar el Proyecto

1. Una vez que el proyecto esté importado, haz clic derecho en el proyecto en la ventana del Proyecto y selecciona Build para compilar el proyecto.
2. Después de compilar con éxito, haz clic derecho en el proyecto en la ventana del Proyecto y selecciona Run para ejecutar el proyecto.

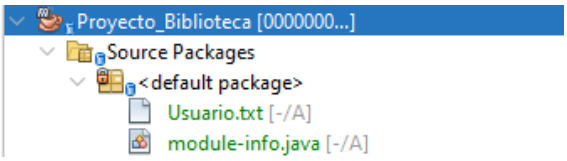
Estructura del proyecto

Tenemos los paquetes que organizan las diferentes clases.

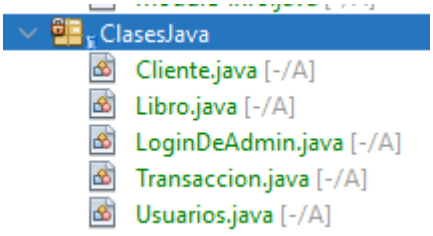


Este proyecto fue trabajado en Maven por lo cual se estuvo organizando de esta forma:

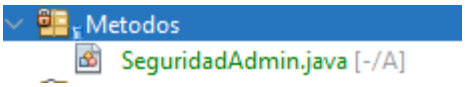
- En el paquete <default package>, en donde tenemos almacenado la clase module-info, en la cual descargamos las librerías.
- Así mismo tenemos un archivo de texto que tiene almacenado el usuario y contraseña del administrador.



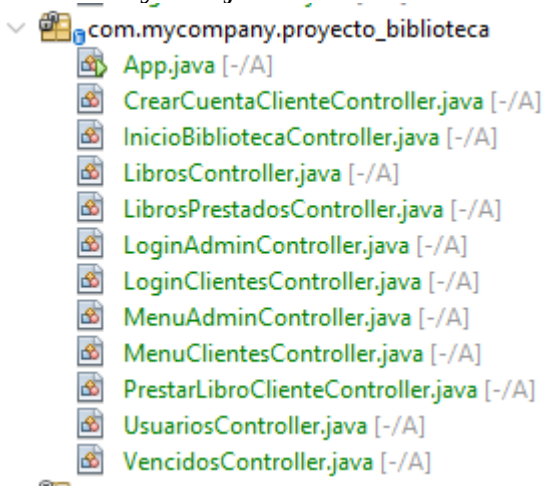
- En el paquete Clases java, encontramos clases en donde almacenamos atributos.



- Paquete Métodos, tiene almacenado una clase java, en el cual se realizo la validación de logueo únicamente de el administrador.

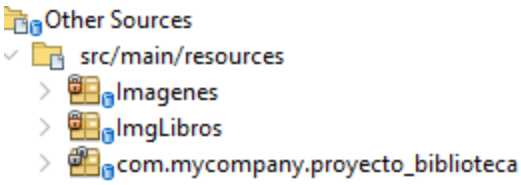


- Paquete com.my.company.proyecto_biblioteca, se encuentran las clases controladoras de las interfaces, en las cuales se manipularon los atributos y se ejecutaron los métodos como acciones.

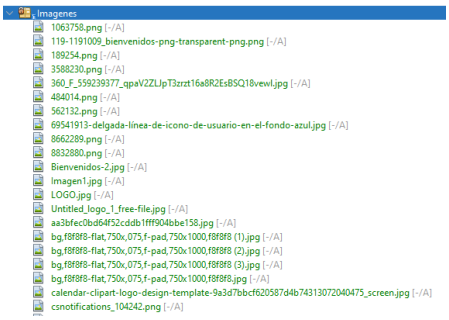


- Paquete sql, contiene la conexión hacia la base de datos PostgreSQL.

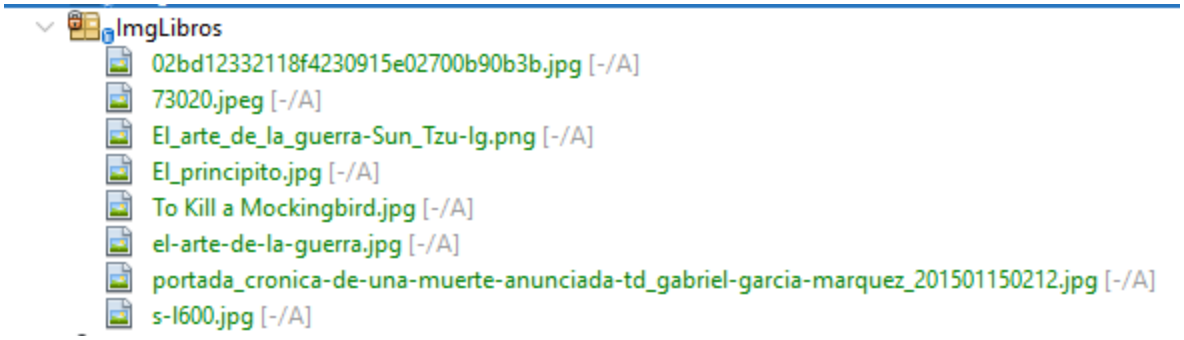
- Tenemos el archivo que almacena otros paquetes que se utilizaron para la interfaz gráfica:



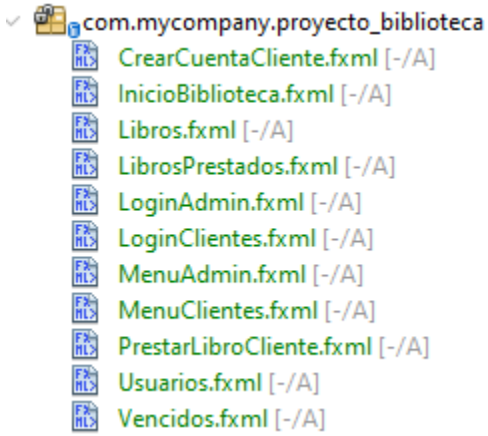
- El paquete Imágenes contiene imágenes que se utilizaron en las interfaces.



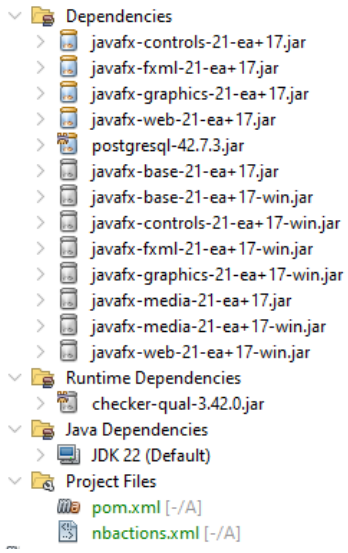
- El paquete ImgLibros, contiene imagénes de las portadas de los libros que se usaron para registrarlos:



- El paquete com.mycompany.proyecto_biblioteca, se almacenan las clases FXML, las cuales son interfaces que se muestran en el proyecto.



- Como se observa en los archivos siguientes se encuentran las dependencias, el archivo pom y el jdk que se utilizó en el proyecto.



Proceso de Compilación

1. Configuración del Entorno de Desarrollo:

Asegúrate de tener instalado Java Development Kit (JDK) en tu sistema y configurado correctamente.
Instala Maven en tu sistema y verifica que esté correctamente configurado.
Estructura del Proyecto Maven:

2. Crea un proyecto Maven utilizando el arquetipo adecuado para una aplicación JavaFX.

Organiza tu proyecto Maven siguiendo las convenciones estándar de directorios de Maven, como src/main/java para el código fuente y src/main/resources para los recursos.

3. Configuración de Dependencias:

Define las dependencias necesarias en el archivo pom.xml de Maven.
Incluye las dependencias de JavaFX, JDBC u otros frameworks que necesites para tu proyecto.

4. Desarrollo de Código:

Escribe el código fuente de tu aplicación JavaFX y la lógica de negocio utilizando POO, siguiendo los requisitos y funcionalidades descritas para la aplicación Biblioteca.

5. Abrir NetBeans:
   Abre NetBeans IDE en tu sistema.
   Importar el Proyecto Maven:
   En NetBeans, selecciona File > Open Project.
   Navega hasta el directorio raíz de tu proyecto Maven y selecciona el archivo pom.xml.
   Haz clic en "Open Project".
   Esperar a que NetBeans Cargue el Proyecto:
   NetBeans cargará automáticamente el proyecto Maven y descargará las dependencias necesarias.

6. Compilar el Proyecto:
   Una vez que NetBeans haya cargado el proyecto, puedes compilarlo seleccionando Run > Build Project o simplemente presionando F11.

7. Ejecutar el Proyecto:
   Después de compilar con éxito, puedes ejecutar el proyecto seleccionando Run > Run Project o haciendo clic en el botón de ejecución verde en la barra de herramientas.
   NetBeans ejecutará el proyecto y mostrará la salida en la consola de salida o en una ventana de la aplicación, dependiendo de cómo esté configurado el proyecto.
   Depurar el Proyecto (Opcional):

   Si necesitas depurar el proyecto, puedes establecer puntos de interrupción en el código y ejecutar el proyecto en modo de depuración seleccionando Debug > Debug Project o presionando Ctrl + F5.
   Detener la Ejecución (Opcional):

   Si en algún momento deseas detener la ejecución del proyecto, puedes hacerlo seleccionando Run > Stop o simplemente presionando Ctrl + F2.

Código de las implementaciones de programación orientada a objetos.

```
 */
public class Libro {
    //Abstracción son los atributos
   private int idSerial;
   private String ISBN;
   private String titulo;
   private String autor;
   private String editorial;
   private int cantidadDisponible;
   private String tipoDelibro;
   private int añoDepublicacion;
   private byte[] imagen;

   //Encapsulamiento lo privado y publico de las clases, por medio del constructor utilizo los atributos privados
public Libro( int idSerial, String ISBN, String titulo, String autor, String editorial,int cantidadDisponible, String tipoDelibro,  int añoDePublicacion,  byte
       this.idSerial = idSerial;
       this.titulo = titulo;
       this.autor = autor;
       this.editorial = editorial;
       this.cantidadDisponible = cantidadDisponible;
       this.tipoDelibro = tipoDelibro;
       this.ISBN = ISBN;
       this.añoDepublicacion = añoDePublicacion;
       this.imagen = (imagen != null) ? imagen : new byte[0]; // Maneja el caso de imagen nula
   }
```

Código de implementación de CRUD

```
//------------------------------------ Agregar registro-----------------------------------------
public void agregarRegistro(){

 // Verifica si se ha seleccionado una imagen
 if (selectedFile == null) {
     Alert alert = new Alert(Alert.AlertType.WARNING);
     alert.setTitle("Advertencia");
     alert.setHeaderText(null);
     alert.setContentText("Por favor, seleccione una imagen.");
     alert.showAndWait();
     return;
 }

 // Consulta SQL para insertar un nuevo libro en la base de datos
 String query = "INSERT INTO libros (ISBN, titulo, autor, editorial, cantidadDisponible, tipoDeLibro, añoDePublicacion, imagen) VALUES (?, ?, ?, ?, ?, ?, ?,
```

```
//---------------------------------Actualizar registro-----------------------------------------
private void actualizarRegistro (){
String query = "UPDATE libros SET ISBN = '" + ISBNtxt.getText() + "', titulo= '" + titulotxt.getText() +
"', autor= '" + autortxt.getText() + "', editorial= '" + editorialtxt.getText() + "', cantidadDisponible= '" + cantidadDisponibletxt.getText() +
"', tipoDeLibro= '" + tipoDeLibrotxt.getText() + "', añoDePublicacion= '"+ añoDePublicaciontxt.getText() + "' WHERE idSerial = " + idSerialtxt.getText();
ejecutarQuery(query, () ->{
      // mostrarUsuarios();

       // Actualizar el registro en la lista observable
    Libro selectedLibro = TbLibros.getSelectionModel().getSelectedItem();
    if (selectedLibro != null) {
        selectedLibro.setISBN(ISBNtxt.getText());
        selectedLibro.setTitulo(titulotxt.getText());
        selectedLibro.setAutor(autortxt.getText());
        selectedLibro.setEditorial(editorialtxt.getText());
        selectedLibro.setCantidadDisponible(Integer.parseInt(cantidadDisponibletxt.getText()));
        selectedLibro.setTipoDeLibro(tipoDeLibrotxt.getText());
        selectedLibro.setAñoDePublicacion(Integer.parseInt(añoDePublicaciontxt.getText()));

        // Refrescar la tabla para mostrar los cambios
        TbLibros.refresh();

    // Limpiar los campos de texto
    limpiarCampos();
});
}
```

```
//-----------------------------------Eliminar registro-----------------------------------------
private void eliminarRegistro(){
Libro selectedLibro = TbLibros.getSelectionModel().getSelectedItem();
if (selectedLibro != null) {
  int id = selectedLibro.getIdSerial();

String query = "DELETE FROM libros WHERE idSerial =" + idSerialtxt.getText();
ejecutarQuery(query, () ->{
//actualizarIds();
mostrarLibros();
// Limpiar los campos de texto
    limpiarCampos();
});

}
}
```

Sección de solución de problemas comunes.

Las interfaces suelen tardarse, unas más que otras, pero si ejecuta lo indicado y muestra la pantalla.

• Documentación extra • Documentación de JavaFX: https://openjfx.io/ • NeatBeans Documentation: https://netbeans.apache.org/kb/ • En la descarga del JDK 22 y dependencias. Asegúrese de configurar las dependencias en el archivo pom.xml así mismo implementar las librerías en la clase module info.

Información de contacto

Correo electrónico: hrompichp1@miumg.edu.g