Escuela Politécnica Nacional

Facultad de Ingeniería de Sistemas

Programación II

Juego musical de guitarra controlado por un mando de Xbox

Grupo #4

Integrantes:

Sebastián Oña

Richard Tipantiza

Rommel Rivera

Carlos Troya

Docente:

Ing. Patricio Pacha

Índice

Introducción

En un mundo donde la música es la clave para conectar emociones y experiencias, nuestro proyecto busca innovar en el ámbito de los videojuegos musicales. Con la fusión de la tradicional guitarra y la tecnología moderna del mando de Xbox, ofrecemos una experiencia única que invita a los usuarios a explorar su creatividad musical. Este juego no solo es una herramienta para aprender y practicar acordes de guitarra, sino también un medio para expresarse artísticamente y compartir creaciones con otros. A través de una interfaz intuitiva y una jugabilidad envolvente, nuestro juego musical se convierte en un espacio donde la música y el entretenimiento se entrelazan para crear momentos memorables.

Se dirige principalmente a miembros de nuestra clase de programación II quienes necesitarán comprender la arquitectura y los estándares de codificación para desarrollar y mantener el juego. También es relevante para usuarios finales interesados en la experiencia musical y educativa que ofrece el juego. Además, puede ser útil para educadores que buscan herramientas innovadoras para enseñar música y para investigadores que estudian el impacto de los videojuegos en la educación musical.

Requisitos del sistema

2.1. Hardware

- **CPU:** Procesador de 64 bits con al menos 2.5 GHz.
- **RAM:** Memoria RAM de 8 GB o más.
- **Almacenamiento:** Al menos 20 GB de espacio libre en el disco duro.
- **Tarjeta gráfica:** Compatible con DirectX 11 o superior.
- **Mando de Xbox:** Mando de Xbox One o Xbox Series X|S para la selección de acordes.

2.2. Software

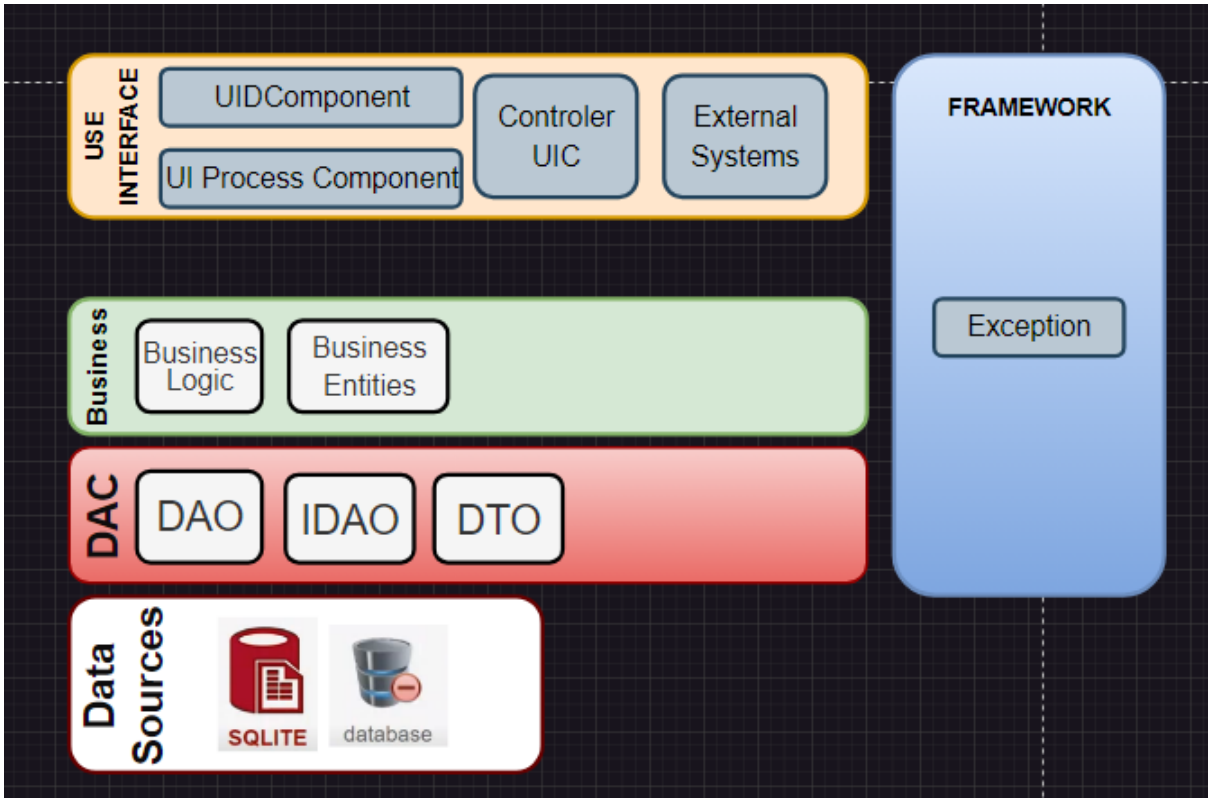
- **Sistema Operativo:** Windows 10 o superior, macOS Catalina o superior, Linux (Ubuntu 18.04 LTS o superior).
- **Frameworks:** Java Development Kit (JDK) versión 8 o superior.
- **Otros:** Visual Studio Code para el desarrollo y depuración.

2.3 Dependencias

- **Bibliotecas Java:** JavaFX para la interfaz gráfica, Swing para componentes adicionales.
- **Librerías específicas del juego:** JLabel, UserInterface, awt, net y Scanner.

Arquitectura del sistema

3.1. Diagrama de Arquitectura



3.2. Descripción de componentes

El diagrama de arquitectura del juego musical se compone de varios componentes clave que trabajan juntos para proporcionar una experiencia de usuario completa y eficiente. Los **Data Sources** incluyen SQLite, una base de datos ligera y autocontenido que almacena datos del juego en el dispositivo del usuario, y opcionalmente una base de datos más robusta para proyectos más grandes o para compartir datos entre múltiples usuarios.

El **Data Access Component (DAC)** es fundamental para gestionar la interacción con la base de datos. Dentro del DAC, el **DAO (Data Access Object**) encapsula la lógica de acceso a la base de datos, permitiendo operaciones CRUD sin exponer detalles específicos de la base de datos. El **IDAO (Interface Data Access Object)** define la interfaz que los DAO deben implementar, asegurando la compatibilidad con diferentes tipos de bases de datos. Además, el **Data Helper** proporciona métodos auxiliares para

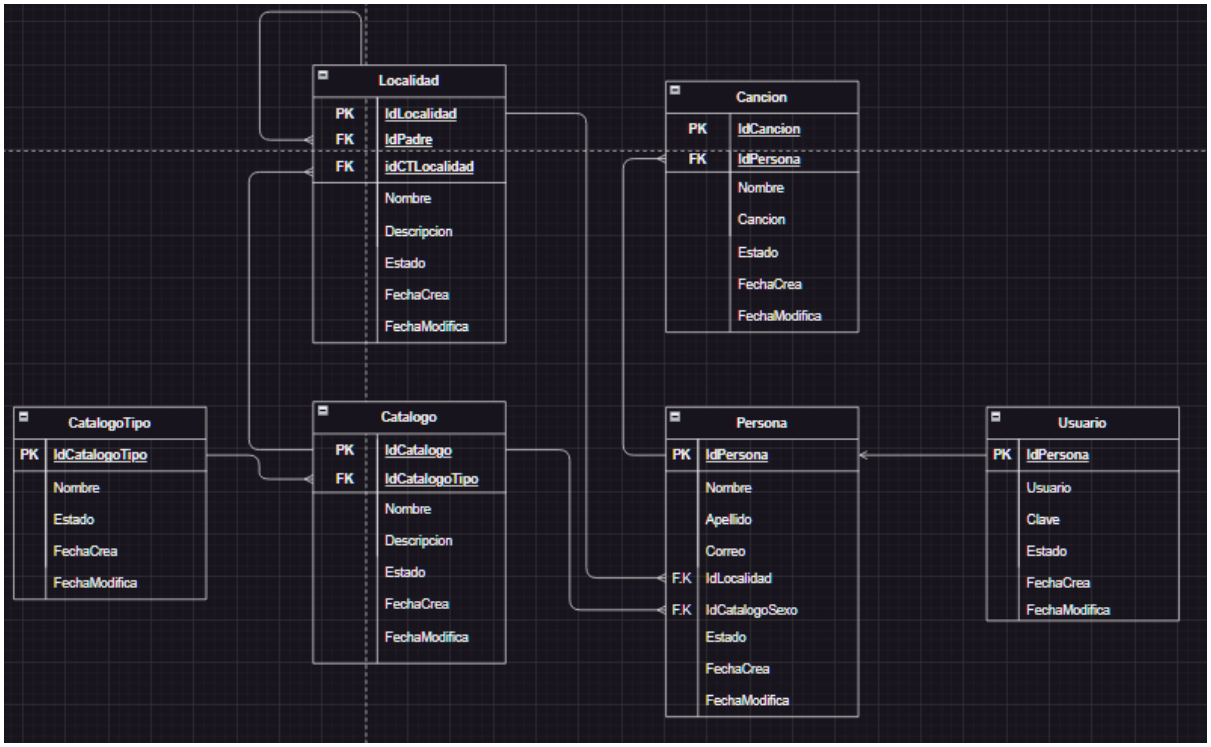
tareas comunes y el **DTO (Data Transfer Object)** facilita la transferencia de datos entre capas del sistema.

En el aspecto del negocio, el **Business Logic** contiene las reglas y lógica principal del juego, mientras que los **Business Entities** representan los objetos del mundo real del juego, como canciones y acordes. La **Use Interface** define las operaciones que los usuarios pueden realizar en el juego.

Finalmente, el **UI Process Component** se encarga de la gestión del flujo de trabajo en la interfaz gráfica. Dentro de este componente, el **UID Component** es responsable de la interfaz gráfica del usuario y la interacción con el mando de Xbox, mientras que el **UI Process Component** maneja eventos como mostrar mensajes al usuario y otros procesos relacionados con la interfaz.

Diseño de la Base de Datos

4.1. Modelo Entidad - Relación



4.2. Estructura de las Tablas

Para la base de datos del proyecto, se desarrollaron seis tablas:

CatalogoTipo

- **Primary Key (PK):** IdCatalogoTipo
- **Columnas:** Nombre, Estado, FechaCrea, FechaModifica
- **Descripción:** Esta tabla contiene información sobre los tipos de catálogos. Las columnas Estado y FechaCrea se autollenan por defecto; Estado se llena con un carácter para indicar disponibilidad

y FechaCrea se llena al momento de ingresar datos en cada registro.

Catalogo

- **Primary Key (PK):** IdCatalogo
- **Foreign Key (FK):** IdCatalogoTipo (conectada a la tabla CatalogoTipo)
- **Columnas:** Nombre, Descripción, Estado, FechaCrea, FechaModifica
- **Descripción:** Esta tabla almacena los catálogos y está relacionada con la tabla CatalogoTipo.

Localidad

- **Primary Key (PK):** IdLocalidad
- **Foreign Key (FK):** IdPadre (conectada consigo misma), IdCTLocalidad (conectada a la tabla Catalogo)
- **Columnas:** Nombre, Descripción, Estado, FechaCrea, FechaModifica
- **Descripción:** Esta tabla contiene información sobre las localidades y tiene una relación jerárquica consigo misma.

Persona

- **Primary Key (PK):** IdPersona
- **Foreign Keys (FK):** IdLocalidad (conectada a la tabla Localidad), IdCatalogoSexo (conectada a la tabla Catalogo)
- **Columnas:** Nombre, Apellido, Correo, Estado, FechaCrea, FechaModifica
- **Descripción:** Esta tabla almacena información sobre las personas y está relacionada con las tablas Localidad y Catalogo.

Cancion

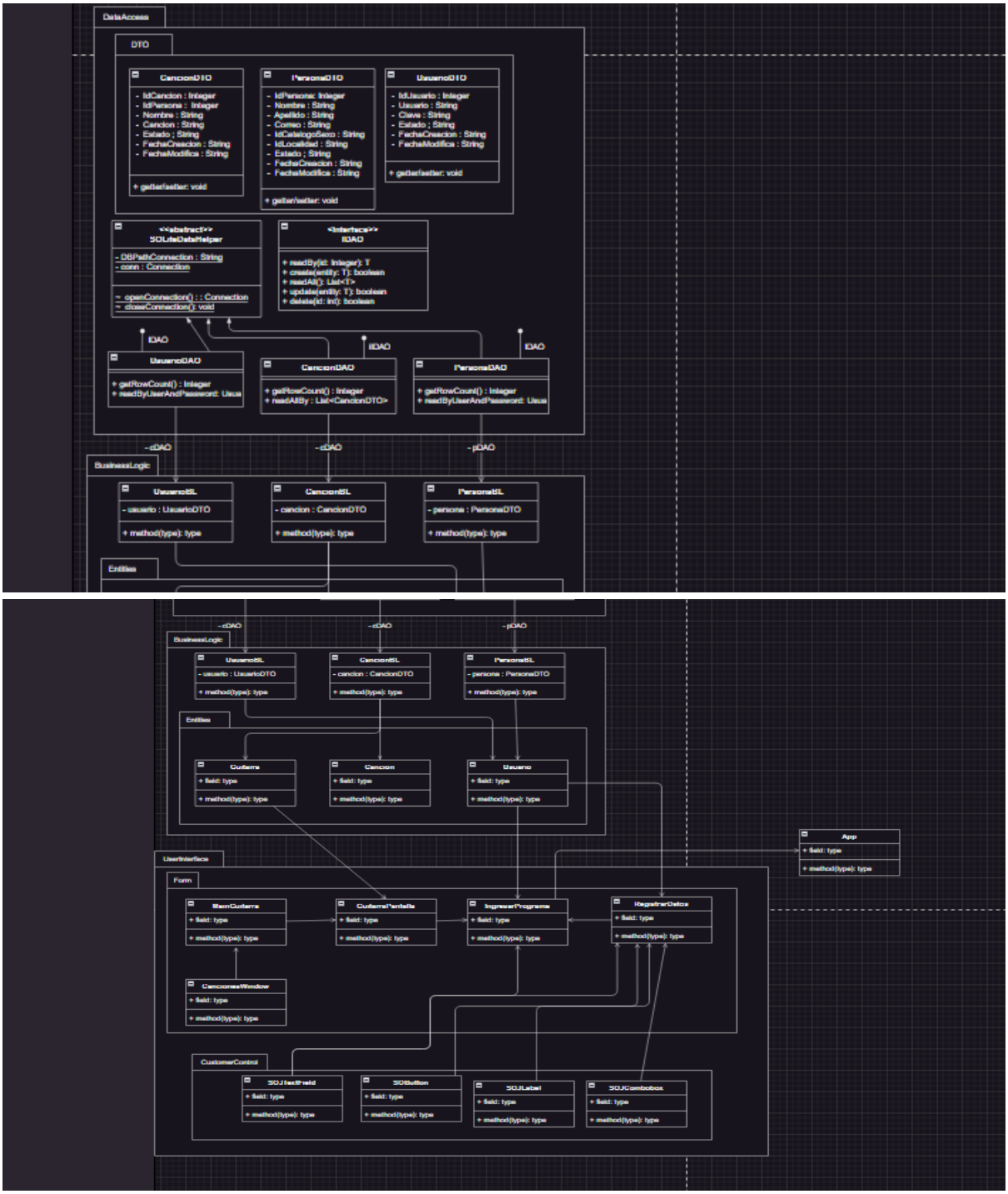
- **Primary Key (PK):** IdCancion
- **Foreign Key (FK):** IdPersona (conectada a la tabla Persona)
- **Columnas:** Nombre, Cancion, Estado, FechaCrea, FechaModifica
- **Descripción:** Esta tabla contiene información sobre las canciones y está relacionada con la tabla Persona.

Usuario

- **Primary Key (PK):** IdPersona (compartida con la tabla Persona)
- **Columnas:** Usuario, Clave, Estado, FechaCrea, FechaModifica
- **Descripción:** Esta tabla almacena información sobre los usuarios y comparte la clave primaria con la tabla Persona.

Diagrama de Clases

5.1. Diagrama de UML de Clases

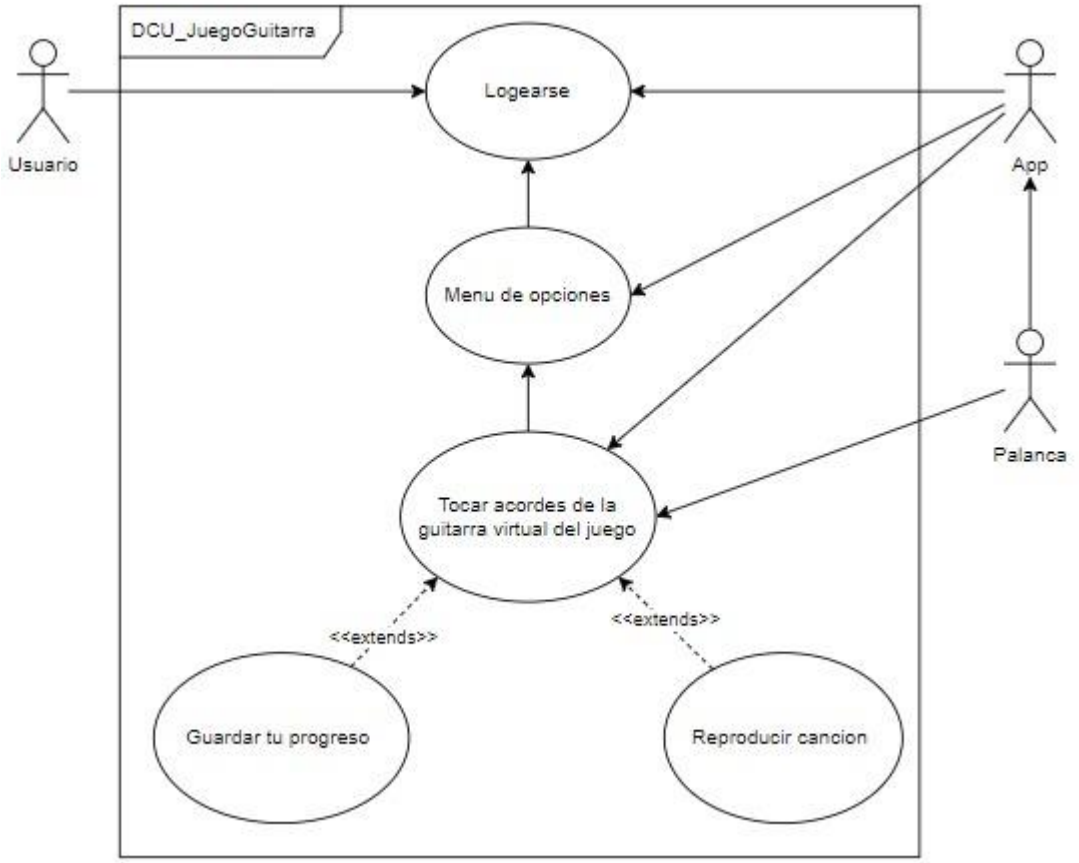


5.2. Descripción de clases

- **Guitarra**: Esta clase maneja la presentación de notas, canciones y la biblioteca del usuario. Tiene métodos para interactuar con el juego (`pantallaJuego`), reproducir canciones (`reproducir`), editar información (`editar`) y guardar datos (`guardar`).
- **Juego**: Esta clase actúa como el núcleo del sistema, gestionando la biblioteca y el menú principal. Está asociada tanto con `Guitarra` como con `Usuario`, lo que indica que puede interactuar con ambas clases.

- **Usuario**: Esta clase gestiona la información del usuario, incluyendo el nombre de usuario y la clave. Tiene métodos para ingresar datos y registrarse, lo que sugiere funcionalidades de autenticación.
- **App**: Esta clase contiene el método principal (`main`) y un objeto `Scanner` para la entrada de datos. Está conectada a `Juego`, lo que indica que `App` puede iniciar y gestionar el menú principal del juego.

5.3. Diagrama de caso de uso



Interfaz Gráfica de Usuario

6.1. Descripción de Componentes de la Interfaz

La interfaz gráfica de usuario (GUI) para registrar datos de un usuario utilizando Java Swing. La clase RegistrarDatos extiende JPanel y organiza varios campos de texto, cuadros de selección y botones en una cuadrícula para capturar información como nombre, apellido, correo, país, ciudad, usuario y contraseña. Al seleccionar un país, las opciones de ciudad se actualizan dinámicamente. Al presionar el botón de “Confirmar Registro”, se valida que todos los campos estén completos y que las contraseñas coincidan. Si todo es correcto, se crea una instancia de la clase Usuario para registrar los datos en el sistema, mostrando un mensaje de éxito o error según el resultado.

La interfaz gráfica de usuario (GUI) para el ingreso y registro de usuarios en un programa, utilizando Java Swing. La clase IngresarPrograma extiende JFrame y configura una ventana con campos de texto para el nombre de usuario y la contraseña, además de botones para ingresar y registrar. La interfaz incluye un fondo personalizado y utiliza un diseño de cuadrícula para organizar los componentes.

Al presionar el botón “Ingresar”, se valida el nombre de usuario y la contraseña mediante el método LlevarVariable de la clase Usuario. Si las credenciales son correctas, se oculta la ventana actual y se muestra la pantalla del juego (GuitarraPantalla). Si las credenciales son incorrectas, se muestra un mensaje de error. El botón “Registrar” despliega un formulario de registro (RegistrarDatos) en un cuadro de diálogo para que el usuario pueda registrarse en el sistema. La ventana está configurada para no ser redimensionable y se centra en la pantalla al abrirse.

La ventana de aplicación para una “Guitarra Virtual” utilizando Java Swing. La clase GuitarraPantalla extiende JFrame y se encarga de configurar la ventana principal de la aplicación. Al instanciarse, recibe un IdUsuario que se almacena como un atributo de la clase. La ventana se configura con un título, tamaño fijo, y se centra en la pantalla. Además, se crea y añade un panel personalizado (MainGuitarraPanel) al centro de la ventana, utilizando el IdUsuario para personalizar el contenido del panel. La ventana se hace visible y se asegura que la aplicación se cierre correctamente al cerrar la ventana.

El panel principal para una aplicación de “Guitarra Virtual” utilizando Java Swing. La clase MainGuitarraPanel extiende JPanel y configura una interfaz gráfica donde los usuarios pueden grabar, guardar, borrar y reproducir canciones. Al iniciar, se muestra un mensaje de bienvenida y un botón para cargar canciones. Los botones de grabar, guardar, borrar y reproducir permiten al usuario interactuar con la guitarra virtual. Al presionar “Grabar”, se inicia la grabación de una canción, que se detiene al presionar nuevamente el botón. El botón “Guardar” permite al usuario guardar la canción en la base de datos, solicitando un nombre para la canción. El botón “Borrar” limpia la canción actual, y el botón “Reproducir” permite escuchar la canción grabada. La interfaz también incluye un área de texto para mostrar los acordes de la canción y un panel de desplazamiento para facilitar la visualización.

La ventana para gestionar una lista de canciones utilizando Java Swing. La clase CancionesWindow extiende JFrame y configura una interfaz gráfica donde los usuarios pueden seleccionar y abrir canciones de una lista. La ventana incluye una lista (JList) para mostrar las canciones, y dos botones: “Abrir” y “Cancelar”. Al presionar “Abrir”, se verifica si se ha seleccionado una canción y, de ser así, se imprime el nombre de la canción seleccionada y se cierra la ventana. Si no se ha seleccionado ninguna canción, se muestra un mensaje de error. El botón “Cancelar” simplemente cierra la ventana. La interfaz está diseñada con un BorderLayout, colocando la lista de canciones en el centro y los botones en la parte inferior. La ventana se centra en la pantalla al abrirse y se configura para cerrarse al seleccionar “Cancelar”.

Referencias

Banerjee, I., Nguyen, B., Garousi, V., & Memon, A. (2013). Graphical user interface (GUI) testing: Systematic mapping and repository. *Information and Software Technology*, *55*(10), 1679-1694.

MUÐO, D., JAVIER, F., GUTIERREZ LOPEZ, F. R. A. N. C. I. S. C. O., & PIMENTEL SANCHEZ, E. R. N. E. S. T. O. (2007). *Programación orientada a objetos con Java*. Ediciones Paraninfo, SA.

Owens, M. (2003). SQLite: embebiendo una base de datos SQL. *Mundo Linux: Sólo programadores Linux*, (56), 54-59.