UNIVERSIDAD NACIONAL DE COLOMBIA

ENTREGABLE 2: PROTOTIPO FUNCIONAL DEL SISTEMA

Jefferson Jair Figueroa Escobar
Juan David Beltran Orjuela
John Jairo Paez Albino
Juan Sebastian Leguizamón Silva

13 de Noviembre del 2025

UNIVERSIDAD NACIONAL DE COLOMBIA

Universidad Nacional de Colombia
Sistemas de Información

Índice

UNIVERSIDAD NACIONAL DE COLOMBIA

Levantamiento de Requerimientos

Técnicas Aplicadas

En nuestro caso, la técnica que aplicamos fue la observación.

Para el caso de la observación fuimos a un bar de aquí en la ciudad de Bogotá, que cumpliera más o menos con los requisitos que estábamos planteando para el ERP, un bar que requiera de un sistema de información, que requiera utilizar la API de Spotify, tanto para el control de las canciones, como para la obtención y representación de métricas que pueden ayudar a que el Dj o la persona encargada de la música ponga canciones acorde a la situación.

En nuestras observaciones encontramos que actualmente el manejo de la música suele estar ligado a la percepción del Dj y a las canciones que suelen pedir las personas, sin embargo, en ciertas ocasiones el aforo parece variar en función a las canciones o al tipo de canciones que están sonando. Por consiguiente suponemos que nuestro sistema ERP propuesto para este negocio puede ser factible para obtener más aforo y mayor consumo en base a la música presentada.

Requerimientos Funcionales

Tabla 1. Requisitos Funcionales priorizados con MoSCoW

ID	Nombre	Descripción	Prioridad	Fuente	¿Implementado?
RF 1	Como usuario quiero visualizar una página de inicio antes de iniciar sesión, para ver que si estoy entrando a el aplicativo correcto	Básicamente el sistema debe tener una clase de página de inicio o Landing Page, para que el usuario, administrador o la persona que vaya a ingresar, puede observar que esta entrando a el aplicativo	Must	Este requisito se debe puesto que al momento de utilizar la API de Spotify hay que realizar un inicio de sesión y no se vería del todo bien, iniciar el aplicativo y que de la nada haga la solicitud de inicio de sesión.	Si
RF 2	Como usuario quiero ver un DashBoard, para ingresar a los apartados de canciones, listas, métricas u otros.	Es el panel principal del aplicativo, donde se podrá acceder a las diferentes opciones que proporciona el sistema, como el apartado de canciones, el apartado de listas, y el apartado de BI.	Must	Es necesario algo así, para establecer el flujo que se desea seguir ya sea para un evento, alistar las listas del día, o hacer inteligencia de negocio.	Si
RF 3	Como usuario, quiero	En algunos casos es	Must	Es necesario tener control	Si

	tener control sobre las canciones, para pausar o cambiar de canción.	necesario tener control sobre las canciones, poder cambiar una canción o pausar en caso de ser necesario, básicamente tener un control sobre las canciones.		sobre las canciones que están sonando, puesto que en algunos caso es necesario poner otra canción, o simplemente cambiarla, incluso en algunos casos, es necesario pausar la canción.	
RF 4	Como usuario, quiero visualizar el nombre de la canción en la interfaz gráfica para identificar la canción que está sonando.	Es necesario saber el nombre de la canción que se está escuchando en ese momento, tanto para hacer una calificación, como para hacer una retroalimentación.	Must	Es necesario saber qué canción está sonando en cierto momento específico, por lo cual necesitamos que se muestre el nombre de la canción cerca del reproductor.	Si
RF 5	Como usuario, quiero visualizar el nombre de la lista de reproducción, para identificar si es la lista de reproducción programada.	Saber el nombre de la lista de reproducción es crucial, aún más, cuando las listas están dadas por días o eventos especiales.	Must	Como el bar suele manejar ciertas canciones por eventos o por días, es necesario visualizar la lista de reproducción de donde se está escuchando la canción, ya sea una propia o otra lista.	Si
RF 6	Como usuario, quiero calificar las canciones, para plasmar el ambiente que se vivió mientras estas estaban sonando	La persona que tiene el control de la música generalmente percibe que tan bien recibida o no es una canción, la calificación permite determinar que tan buena es una canción para el bar en base a la percepción del DJ o la persona a cargo de la música.	Should	Dado que queremos hacer inteligencia de negocio o BI, es necesario recoger métricas, en nuestro caso, necesitamos que el Dj o la persona encargada de la música, haga una calificación de la canción que está sonando, mediante estrellas, dado el ambiente que está viendo el Dj por lo cual se basará en su percepción.	No
RF 7	Como usuario, quiero llenar un formulario sobre las canciones presentadas en ese dia para plasmar qué tan útiles pueden ser estas canciones en el bar	El formulario es algo que se debería llenar todos los días al cerrar el bar, de tal manera que se puede tener más información sobre las canciones que se presentation ese día, debería tener campos	Must	Es necesario recoger más métricas, por este motivo, al finalizar el evento o al cierre de el var en cada noche, es necesario rellenar un formulario con diferentes datos, incluyendo cosas como percepción general, datos	No

		como: Percepción general, Aforo promedio, Aforo Maximo, Canciones cantadas, Consumo de la barra.		de consumo, mejores canciones, etc.	
RF 8	Como Administrador, quiero tener un panel de métricas, para analizarlas y tomar decisiones sobre las canciones que deberían estar posteriormente en las listas de reproducción.	Un panel de métricas es muy útil, sobre todo teniendo en cuentas las KPIs que se establecieron, esto permite que el bar tome mejores decisiones sobre las canciones que deberían sonar o no en base a el análisis de los resultados obtenidos.	Must	Un panel de métricas es necesario para visualizar las diferentes gráficas y las KPIs definidas, para tomar decisiones acorde a las canciones y tipos de canciones que generan y no generan un mayor consumo y aforo en el bar.	Si
RF 9	Como usuario, quiero visualizar una BlackList, para mover las canciones que tienen un desempeño muy bajo o su reproducción afecta de forma negativa a el bar.	Una BlackList, ayuda a definir que tipo de canciones no pueden ser reproducidas en el bar, puesto que este tipo de canciones o la canción en si, no es de agrado para la gran mayoría de personas que asisten a el bar.	Must	Muchas canciones no son de agrado para las personas que frecuentan el bar, por eso es necesario evitar que a toda costa este tipo de canciones suenen o se reproduzcan (Generalmente estas son las que van a tener un puntaje muy bajo)	No

Requerimientos No Funcionales

Tabla 2. Requisitos no funcionales

ID	CATEGORÍA	DESCRIPCIÓN	CRITERIO DE ACEPTACIÓN
RNF 1	Rendimiento	El sistema debe responder en menos de 2 segundos	Tiempo de respuesta < 2 segundos en todas las acciones clave
RNF 2	Seguridad	Las contraseñas deben encriptarse con SHA-256	Confirmar almacenado sólo hashes seguros
RNF 3	Usabilidad	La interfaz debe ser accesible desde dispositivos móviles	dispositivos móviles Pruebas en mobile con UI adaptables y sin errores
RNF 4	Disponibilidad	El sistema debe tener 99.9% de disponibilidad	Monitoreo uptime y reportes mensuales
RNF 5	Mantenibilidad	Código modular, claro, documentado para fácil actualización	Documentación actualizada y modularización del código
RNF 6	Seguridad	Implementar autenticación y autorización robusta para controlar accesos	Usuarios sin permiso no pueden acceder a recursos restringidos
RNF 7	Confiabilidad	El sistema debe garantizar integridad de datos y evitar pérdidas	Backups regulares y manejo adecuado de transacciones
RNF 8	Usabilidad	La experiencia de usuario debe ser intuitiva y fácil de usar	Feedback positivo en evaluaciones de usuarios

Reglas de Negocio

1. Alcance y Objetivo
El sistema recopila información musical para conocer los gustos de los clientes y asigna géneros a días específicos con el objetivo de aumentar las ventas y mejorar la experiencia en La Iglesia Bar.

2. Usuarios y Roles
Está pensado para ser usado principalmente por el personal administrativo, especialmente el jefe de barra y gerente, asegurando que sólo ellos manipulen las listas y métricas.

3. Control de Acceso y Seguridad
El acceso se realiza a través de la cuenta oficial de Spotify del bar y sólo usuarios autorizados pueden acceder para evitar pérdidas o manipulaciones indebidas. La encuesta final la realiza exclusivamente el jefe de barra.

4. Normativa y Cumplimiento
El sistema también cumple con la legislación colombiana vigente en materia de protección de datos y seguridad, garantizando privacidad y protección de la información.

5. Funcionalidades Principales
Permite reproducir música, administrar listas organizadas por días y géneros, y generar métricas y encuestas que evalúan la efectividad musical en la experiencia y ventas.

Casos de Uso

CU 1: Visualizar DashBoard	
Actores ● Usuario del Sistema	**Requisito** RF 2: Como usuario quiero ver un DashBoard, para ingresar a los apartados de canciones, listas, métricas u otros
Descripción Debería ser el panel principal del aplicativo, donde se podrá acceder a las diferentes opciones que proporciona el sistema, como el apartado de canciones, el apartado de listas, y el apartado de BI.	
Precondiciones ● El usuario ya debe haber iniciado sesión con su cuenta de spotify compatible con la API de Spotify	
Flujo Normal 1. El sistema presenta la pantalla principal. 2. El sistema presenta varios botones para acceder a diferentes funciones del sistema. 3. El usuario hace click en un botón que representa la función que necesita realizar. 4. El sistema muestra la información acorde a la función solicitada. 5. El usuario puede hacer clic nuevamente en otro botón.	
Postcondiciones Se ha mostrado la vista relacionada o función acorde a el botón con el cual hubo interacción.	

CU 2: Calificar Canciones	
Actores ● Usuario del Sistema ● Sistema	**Requisito** RF 6: Como usuario, quiero calificar las canciones, para plasmar el ambiente que se vivió mientras estas estaban sonando
Descripción El usuario puede realizar una calificación de la canción que está sonando con una clase de calificación por estrellas básicamente de 1 a 5, esta calificación la puede realizar la persona encargada de la música.	
Precondiciones ● El usuario ya debe haber iniciado sesión con su cuenta de spotify compatible con la API de Spotify ● El usuario tiene que estar en la opción de las listas de reproducción o las canciones, visualizando la lista de reproducción principal.	
Flujo Normal 1. El sistema proporciona en el objeto representativo de la canción cinco estrellas sin relleno. 2. El usuario interactúa con las estrellas haciendo click sobre la calificación deseada. 3. El sistema obtiene el id de la canción y la calificación. 4. El sistema guarda la calificación asociada a la canción en la base de datos. a. Si no es posible realizar este paso, el sistema muestra un mensaje informando sobre el error. b. El sistema limpia los campos y restablece el valor de la calificación. → Vuelve al punto 1 5. El usuario puede calificar nuevamente la canción o calificar otra canción	
Postcondiciones Se ha calificado la canción y esta calificación se habrá guardado en la base de datos.	

CU 3: Llenar Formulario del Día	
Actores ● Usuario del Sistema	**Requisito** RF 7: Como usuario, quiero llenar un formulario sobre las canciones presentadas en ese dia para plasmar qué tan útiles pueden ser estas canciones en el bar

Descripción

Es un formulario que se debe llenar todos los días al cerrar el bar, este tendrá en cuenta las canciones que se presentan ese día. En este se podrán tener en cuenta más aspectos, como:

- Percepción general.
- Aforo promedio.
- Aforo Maximo.
- Canciones cantadas.
- Consumo de la barra.
- etc.

Precondiciones

- El usuario ya debe haber iniciado sesión con su cuenta de spotify compatible con la API de Spotify
- El usuario tiene que estar en la opción de calificación diaria accediendo al formulario.

Flujo Normal

1. El sistema proporciona el formulario a llenar.
2. El usuario llena o selecciona los campos correspondientes.
3. El sistema valida la información diligenciada.
    a. Si el usuario deja campos obligatorios sin completar, el sistema mostrará un mensaje de error indicando qué campos faltan y no permite continuar.
    → Vuelve al punto 2
4. El sistema guarda la información registrada en la base de datos
5. El sistema mostrará un mensaje de “guardado correctamente”.
6. El sistema retorna al usuario a **{Visualizar DashBoard}**

Postcondiciones

Se ha registrado la información del formulario en la base de datos.

Matriz de Trazabilidad

Tabla 3. Matriz de trazabilidad con requisitos funcionales, no funcionales y casos de uso

ID	Descripción	Caso de Uso	Componentes del Sistema
RF 1	Como usuario quiero visualizar una página de inicio antes de iniciar sesión, para ver que si estoy entrando a el aplicativo correcto	**CU 1: Visualizar DashBoard**	● Landing Page personalizado ● Botón de inicio.
RF 2	Como usuario quiero ver un DashBoard, para ingresar a los apartados de canciones, listas, métricas u otros.	N/A	● Dashboard ● Botones de ingreso a los demás módulos ● Pantalla principal
RF 3	Como usuario, quiero tener control sobre las canciones, para pausar o cambiar de canción.	N/A	● Botones de control entre canciones. ● Lista de reproducción principal. ● Consultas a la API para obtener canciones.
RF 4	Como usuario, quiero visualizar el nombre de la canción en la interfaz gráfica para identificar la canción que está sonando.	N/A	● Consulta a la API para obtener información de la canción ● Panel para visualizar el nombre de la canción.
RF 5	Como usuario, quiero visualizar el nombre de la lista de reproducción, para identificar si es la lista de reproducción programada.	N/A	● Consulta a la API para obtener información de la lista de reproducción. ● Panel para visualizar el nombre o el identificador de la lista de reproducción.
RF 6	Como usuario, quiero calificar las canciones, para plasmar el ambiente que se vivió mientras estas estaban sonando	**CU 2: Calificar Canciones**	● Agrupación de botones en forma de estrellas para calificar las canciones.
RF 7	Como usuario, quiero llenar un formulario sobre las canciones presentadas en ese dia para plasmar qué tan útiles pueden ser estas canciones en el bar	**CU 3: Llenar Formulario del Día**	● Formulario de canciones diarias. ● Guardado en la base de datos. ● Renderización de Métricas.
RF 8	Como Administrador, quiero tener un panel de métricas, para analizarlas y tomar decisiones sobre	N/A	● Consulta a la DB para las métricas. ● Visualización de gráficas.

	las canciones que deberían estar posteriormente en las listas de reproducción.		
RF 9	Como usuario, quiero visualizar una BlackList, para mover las canciones que tienen un desempeño muy bajo o su reproducción afecta de forma negativa a el bar.	N/A	● Lista para la Blacklist ● Marcado de canciones en la DB
RNF 1	El sistema debe responder en menos de 2 segundos	N/A	● En cualquier interacción con el sistema, este se debe demorar menos de 2 segundos en responder.
RNF 2	Las contraseñas deben encriptarse con SHA-256	N/A	● Guardado en base de datos y consulta de contraseñas
RNF 3	La interfaz debe ser accesible desde dispositivos móviles	N/A	● Diseño responsive
RNF 4	El sistema debe tener 99.9% de disponibilidad	N/A	● Servidor con Hosting confiable. ● Servidor propio funcionando siempre
RNF 5	Código modular, claro, documentado para fácil actualización	N/A	● Estructura del proyecto por módulos ● Comentarios y documentación técnica ● Uso de Git ● Código limpio y buenas prácticas
RNF 6	Implementar autenticación y autorización robusta para controlar accesos	N/A	● Login ● Encriptación de contraseñas
RNF 7	El sistema debe garantizar integridad de datos y evitar pérdidas	N/A	● Base de datos.
RNF 8	La experiencia de usuario debe ser intuitiva y fácil de usar	N/A	● Paneles de usuario claros y entendibles ● Navegación por el sistema clará.

Modelo del Sistema

Diagrama de Casos de Uso

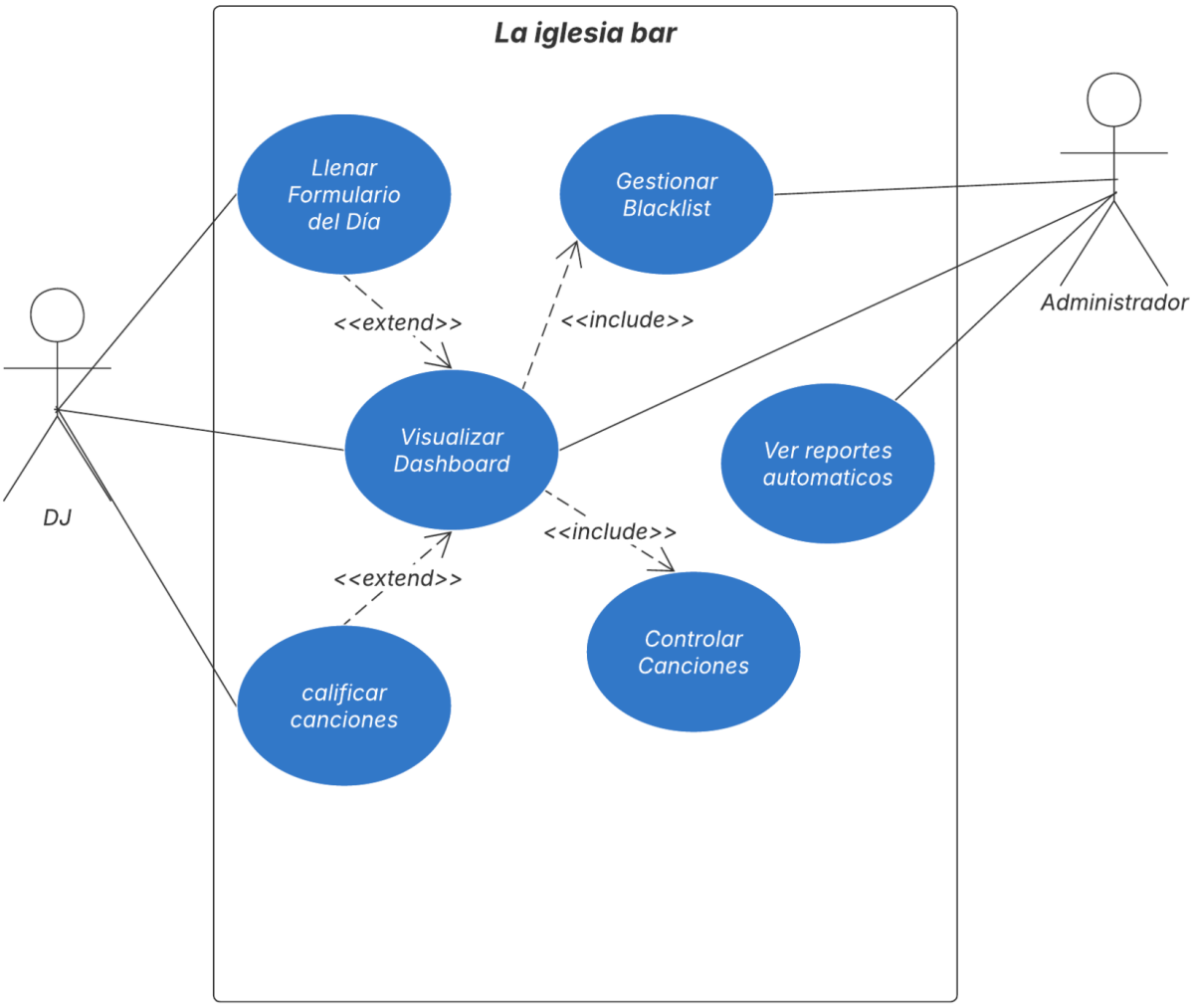

Figura 1. Diagrama de Casos de Uso del Sistema de Gestión de DJs y Administradores

Diagramas de Clases

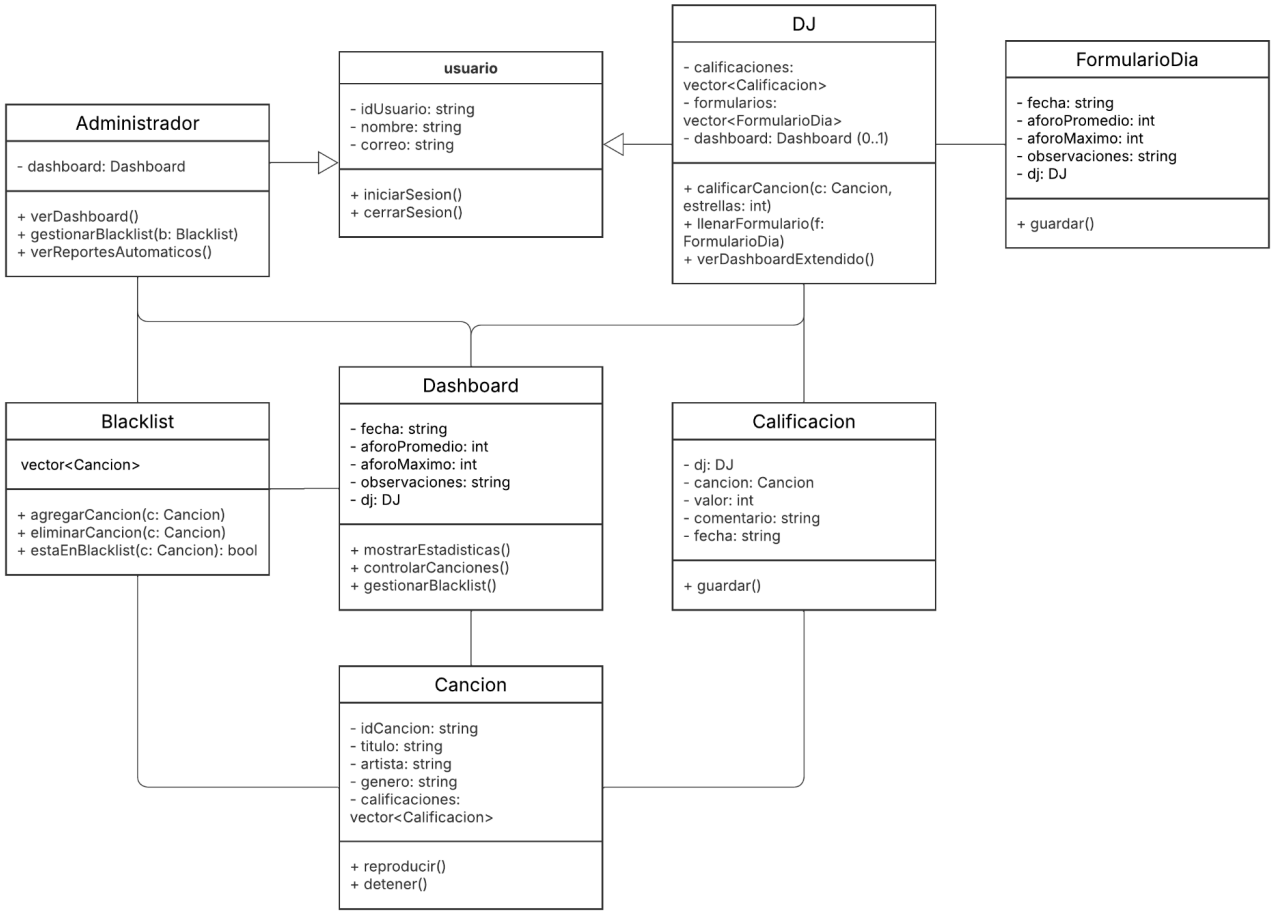

Figura 2. Diagrama de Clases del Sistema

Diagramas de Secuencia

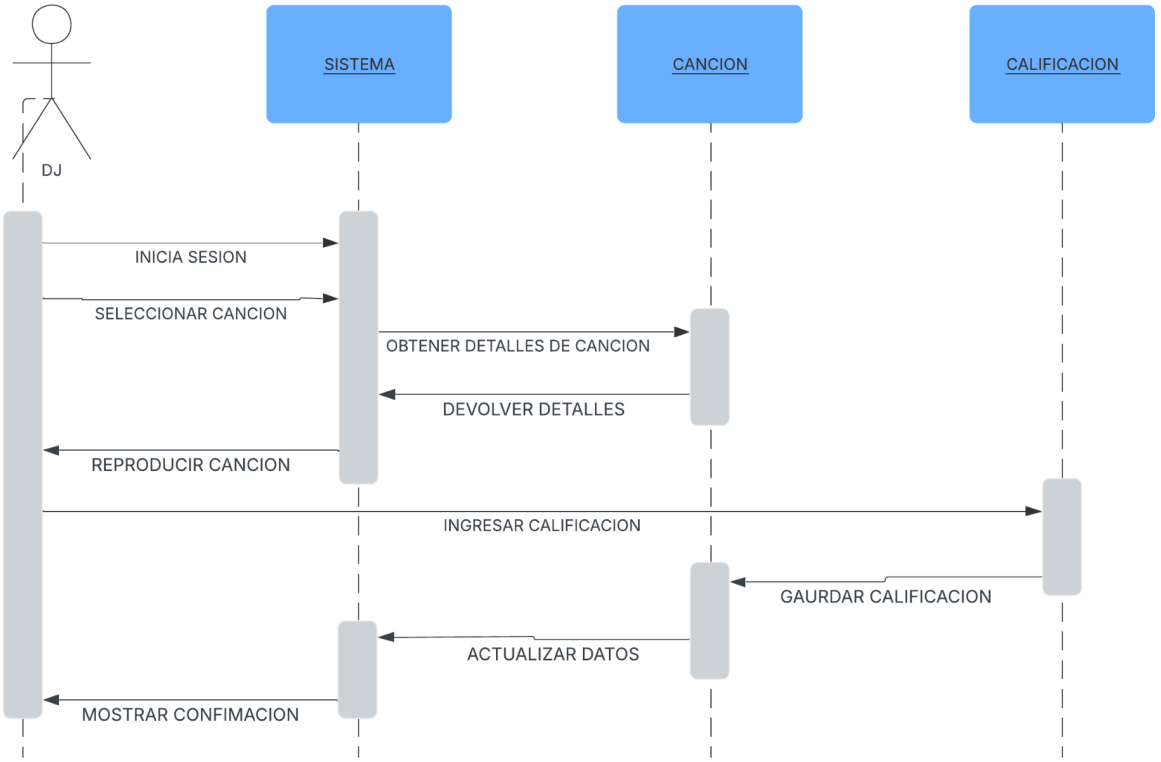

Figura 3. Diagrama de Secuencia – Calificación de Canción

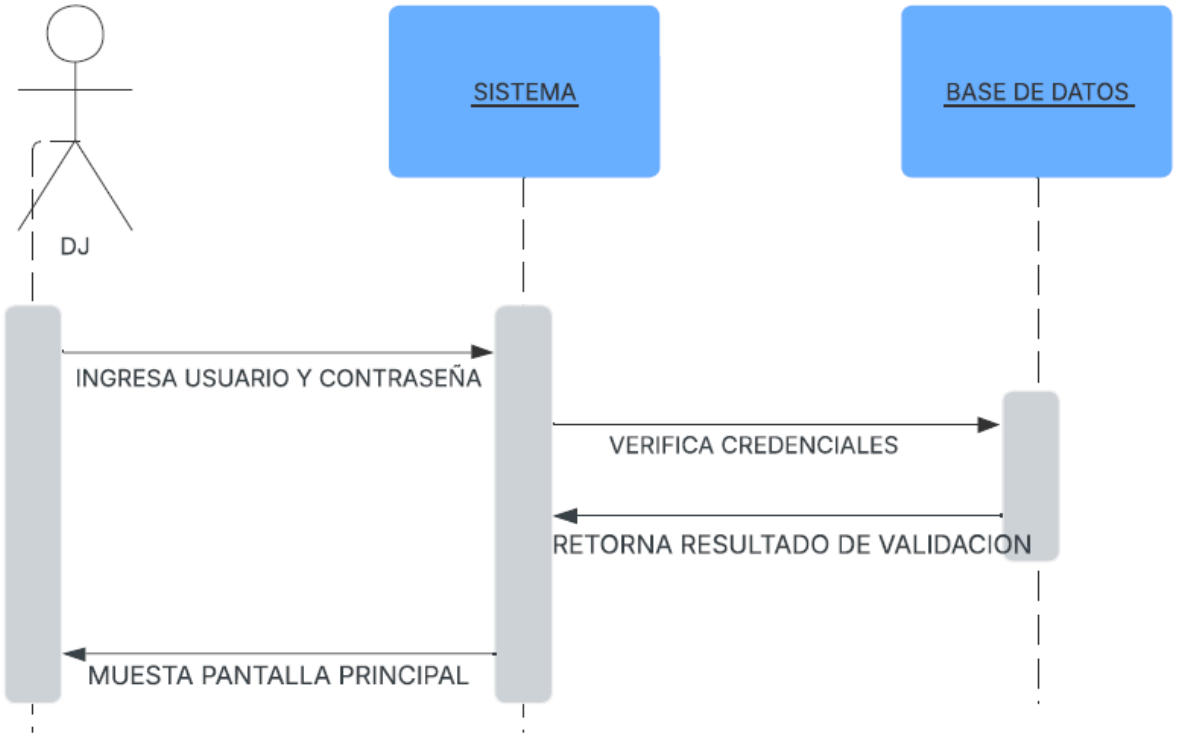

Figura 4. Diagrama de Secuencia – Inicio de Sesión

Diagrama de Actividades

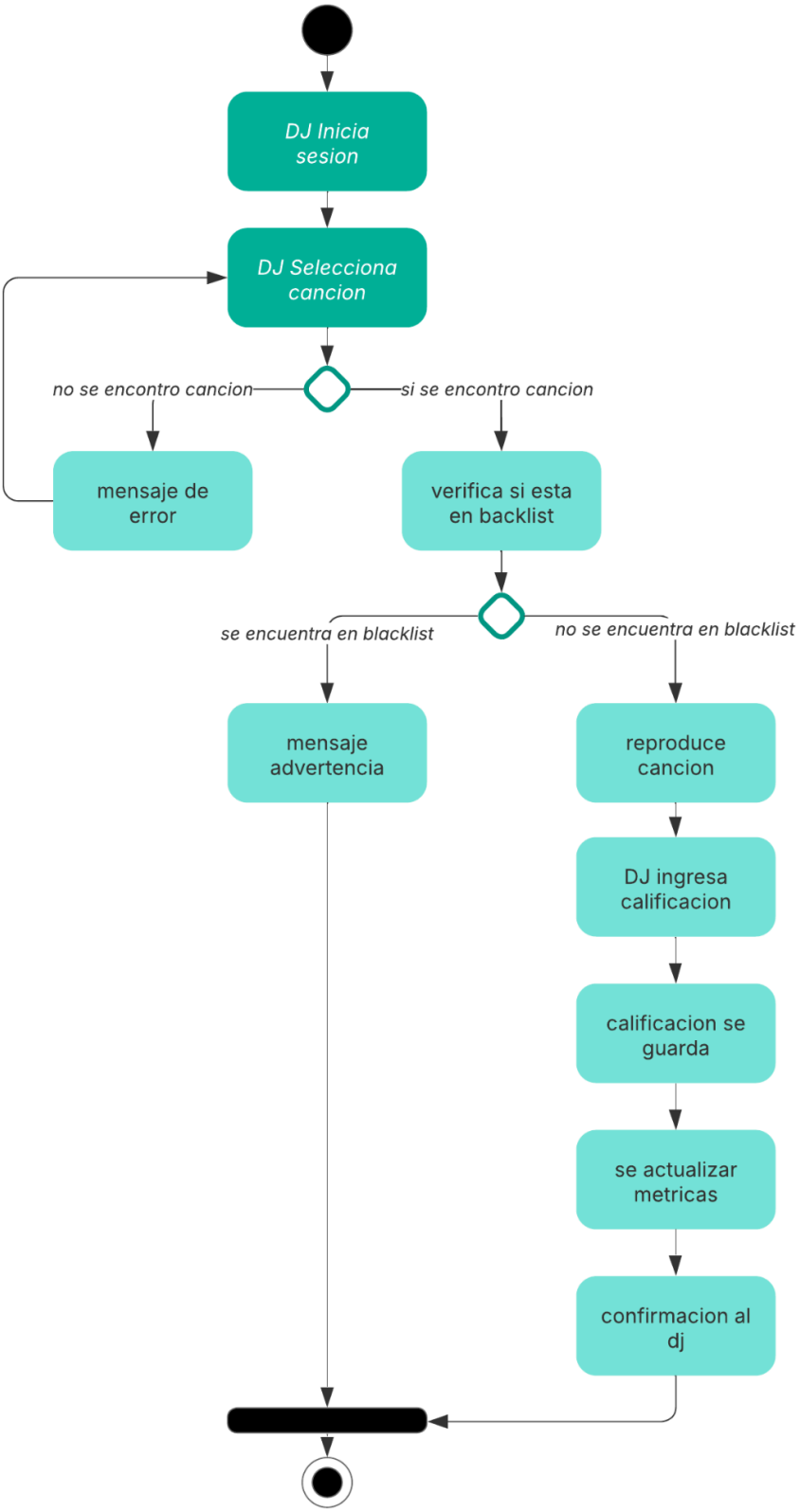

Figura 5. Diagrama de Actividades – Proceso de Calificación con Validación

Diseño de Datos y Arquitectura

Arquitectura del Sistema

Figura 6. Diagrama de arquitectura del sistema por capas.

- Patrón arquitectónico usado

El proyecto utiliza el patrón de arquitectura por capas (Layered Architecture), común en aplicaciones web que siguen el modelo cliente-servidor.

En esta estructura, cada capa cumple una responsabilidad específica:

1. Capa de presentación (frontend)

Muestra la información al usuario mediante HTML, CSS y JavaScript. Envía peticiones al backend para obtener el estado actual de reproducción.

2. Capa de lógica de negocio (backend Flask)

Contiene la lógica principal del sistema: autenticación, comunicación con la API de Spotify, monitoreo del reproductor y almacenamiento de datos.

3. Capa de datos (MongoDB Atlas y Spotify API)

Gestiona el almacenamiento persistente del historial y provee los datos externos del reproductor mediante la API de Spotify.

- Justificación

El uso de una arquitectura en capas se justifica porque:

- Favorece la separación de responsabilidades, permitiendo que la interfaz, la lógica y los datos evolucionen independientemente.
- Mejora la mantenibilidad y escalabilidad, ya que se pueden actualizar módulos (por ejemplo, cambiar MongoDB por otro sistema) sin afectar al resto.
- Facilita la integración con servicios externos como la API de Spotify, ubicándolos en la capa de datos sin alterar la lógica del sistema.
- Compatible con Flask, que naturalmente permite organizar el código en rutas (controladores), plantillas y servicios de datos.
- Permite el despliegue en la nube, manteniendo una comunicación clara entre cliente y servidor.

Tecnologías Utilizadas

Tabla 4. Relación de las tecnologías utilizadas por cada capa del sistema

Componente	**Tecnología**
Backend	Flask (Python)
Frontend	HTML5, CSS3, JavaScript
Base de Datos	MongoDB Atlas
Integración externa	Spotify Web API
Procesamiento de reproducción	Threading (Python)

Modelo de Datos

- Diagrama Entidad Relación (En nuestro caso, el equivalente a NoSQL)

Aunque MongoDB no usa entidades ni relaciones de forma tradicional, se puede representar el modelo lógico equivalente

- Diccionario de datos

Tabla 5. Representación del diccionario de datos para la base de datos.

Colección	Campo	Tipo
Canción	spotify_id	string
	artista	string
	imagen_album	string (URL)
	album	string
	duracion_ms	integer
	cancion	string
	genero	string

- Script SQL de creación (En nuestro caso, el equivalente a NoSQL)

```
db.createCollection("canciones", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["spotify_song_id", "nombre", "artista", "album", "usuario_id"],
      properties: {
        spotify_song_id: { bsonType: "string", description: "ID único de la canción" },
        nombre: { bsonType: "string", description: "Nombre de la canción" },
        artista: { bsonType: "string", description: "Artista principal" },
        album: { bsonType: "string", description: "Nombre del álbum" },
        imagen_album: { bsonType: "string" },
        duracion_ms: { bsonType: "int" },
        fecha_reproduccion: { bsonType: "date" },
      }
    }
  }
});
```

Figura 6. Imagen del script utilizado para la creación de la base de datos.

Prototipo Funcional

Funcionalidades Implementadas

Hasta el momento algunas de las funcionalidades explicadas en el primer entregable fueron totalmente implementadas, el resto de funcionalidades están parcialmente implementadas pero son visibles desde la interfaz gráfica.

Evidencias

Tenemos las siguientes evidencias de las funcionalidades implementadas hasta el momento:

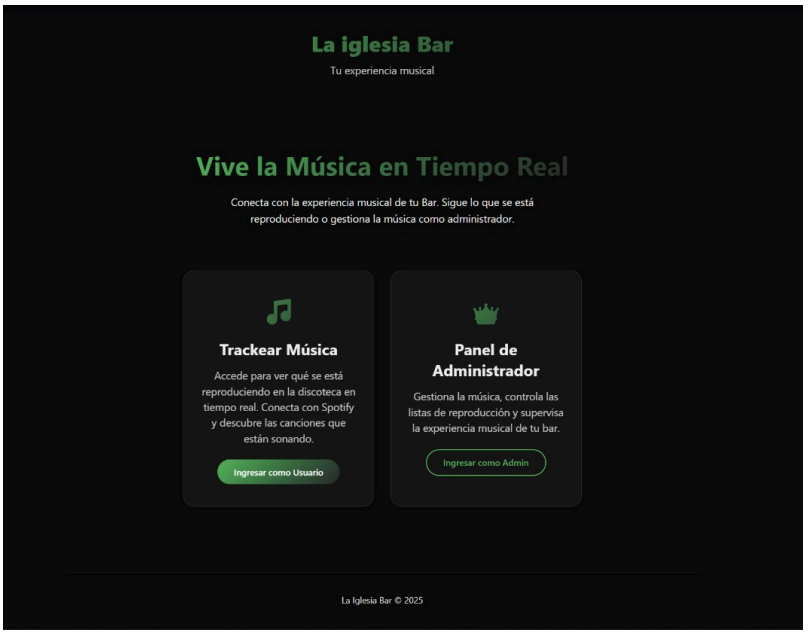

Figura 7. Inicio de sesión para los dos únicos usuarios del sistema.

Figura 8. Evidencia del reproductor desde el programa.

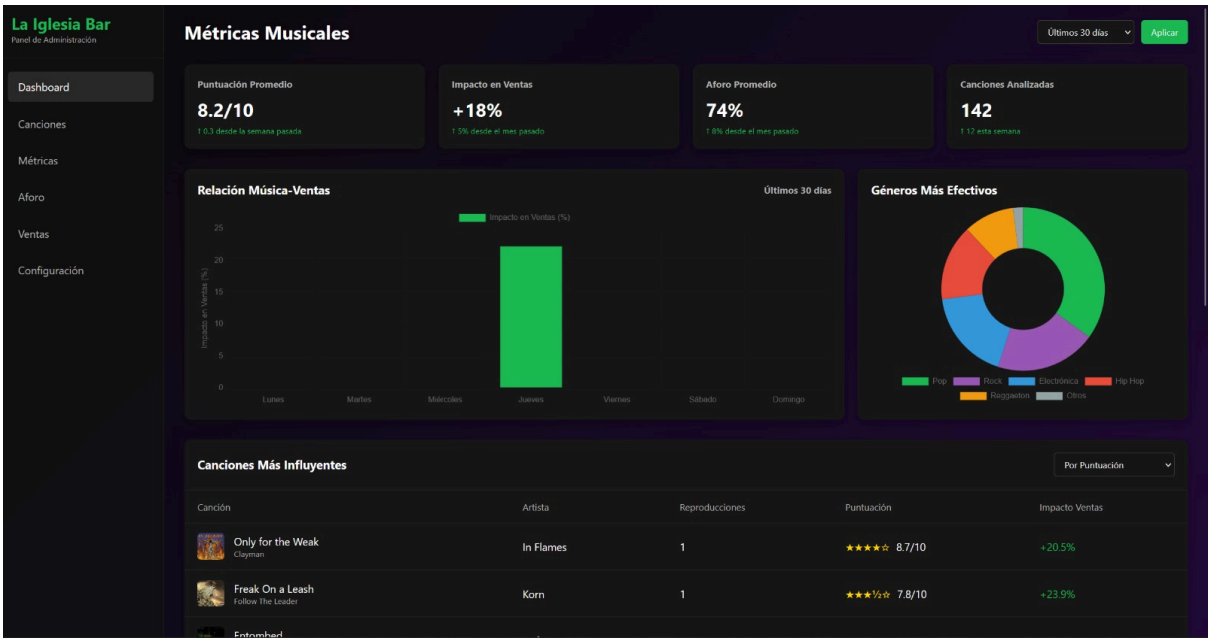

Figura 9. DashBoard general para el administrador del sistema.

Casos de Prueba

Tabla 6. Relación de los casos de prueba con su documentación.

ID	Nombre del Caso de Prueba	Descripción	Precondiciones	Entradas	Pasos	Resultado Esperado
CP01	Registro de calificación válida	Verifica que el DJ pueda calificar una canción correctamente.	DJ autenticado, canción disponible.	Canción = “Track A”, Calificación = 5	1. DJ selecciona canción. 2. Ingresa calificación. 3. Envía calificación.	Calificación guardada y reflejada en el Dashboard.
CP02	Intento de calificación de canción en blacklist	Verifica que el sistema bloquee calificar canciones restringidas.	DJ autenticado, canción en blacklist.	Canción = “Track B”	1. DJ selecciona canción en blacklist.2. Intenta calificar.	Sistema muestra mensaje “No se puede calificar canción en blacklist”.
CP03	Visualización del Dashboard	Verifica que el DJ pueda visualizar el Dashboard con estadísticas actualizadas.	DJ autenticado, existen calificaciones registradas.	—	1. DJ accede al Dashboard. 2. El Sistema genera los gráficos y datos.	Dashboard muestra datos actualizados de calificaciones.
CP04	Administración de blacklist	Verifica que el Administrador pueda agregar una canción a la blacklist.	Administrador autenticado.	Canción = “Track C”	1. Administrador selecciona canción. 2. Agrega a blacklist. 3. Confirma la acción.	Canción agregada exitosamente a la blacklist.
CP05	Generación de reporte automático	Verifica que el Administrador pueda generar y visualizar reportes automáticos.	Administrador autenticado, datos registrados.	—	1. Administrador solicita reporte. 2. Sistema procesa información. 3. Muestra resultados.	Reporte generado y mostrado correctamente.

Acceso a Base de Datos

Como se mencionó anteriormente, estamos utilizando una base de datos NoSQL, por consiguiente tenemos las siguientes evidencias:

Figura 10. Herramienta de gestión con algunos datos cargados y consultados.

Documentación Técnica

Manual de Instalación

Tabla 7. Relación de requisitos para la instalación y correcto funcionamiento del sistema.

Componente	Requisito mínimo
Python	Versión 3.10 o superior
MongoDB atlas	Cuenta activa y una base de datos creada en la nube
Spotify Developer Account	Aplicación registrada para obtener Client ID y Client Secret
Navegador Web	Google Chrome, Edge o Firefox actualizados

Archivo de instalación (Adjunto .txt de dependencias)

Credenciales de Acceso

Tabla 8. Relación de usuarios y claves de acceso al sistema.

Tipo de Usuario	Descripción	Credenciales de acceso	Permisos
Administrador del Sistema	Usuario encargado de la configuración inicial del entorno, la conexión con MongoDB Atlas y el mantenimiento del servidor Flask.	Por definir…	● Configurar variables de entorno. ● Accede al servidor Flask. ● Gestiona credenciales de Spotify y conexión con la BD.
Usuario Operativo	Usuario final que accede mediante su cuenta personal de Spotify para registrar y visualizar su historial musical.	Autenticación mediante Spotify OAuth 2.0	● Inicia sesión con su cuenta de Spotify. ● Visualiza canciones reproducidas en tiempo real. ● No tiene acceso a la configuración del sistema.

Problemas y Limitaciones

Tabla 9. Problemas encontrados con sus respectivas soluciones.

Problema	Descripción	Solución implementada
Errores de autenticación con Spotify (OAuth 2.0)	Durante las primeras pruebas, la API rechazaba el token de acceso porque el redirect URI no coincidía con el configurado en el panel de Spotify Developer.	Se verificó la configuración del redirect_URI tanto en la app Flask como en la consola de Spotify Developer, asegurando coincidencia exacta con http://localhost:5000/callback.
Expiración del token de acceso	Spotify expiraba el token de autenticación cada cierto tiempo (generalmente una hora), lo que provocaba que el monitoreo se detuviera.	Se implementó un mecanismo de refresco automático usando el refresh token provisto por Spotify, garantizando sesiones persistentes sin necesidad de reiniciar manualmente.
Inserción de canciones duplicadas en MongoDB	Cuando una canción se repetía o se cambiaba rápidamente, el sistema guardaba registros duplicados.	Se agregó una verificación por ID de Spotify antes de insertar nuevos documentos, utilizando un índice único en la base de datos.
Desconexión del hilo secundario (thread)	En algunos casos, el proceso loop_player() dejaba de ejecutarse tras reinicios o errores de red.	Se encapsuló el hilo en un bloque try/except y se reinicia automáticamente en caso de falla.
Retrasos en la actualización de la interfaz HTML	La información de la canción no se actualiza inmediatamente sin recargar la página.	Se habilitó una función JavaScript que realiza peticiones periódicas (fetch) al endpoint /estado cada pocos segundos, actualizando dinámicamente el contenido.