Especificación de requerimientos Musicloud

Irving Iván Gumesindo Trujillo

Luis Manuel García Moreno

Contenido

[Introducción 2](#_Toc40308294)

[Proceso 3](#_Toc40308295)

[Artefactos de análisis y diseño 4](#_Toc40308296)

[Clases y caracteristicas de usuarios 4](#_Toc40308297)

[Requerimientos funcionales. 4](#_Toc40308298)

[Requerimientos no funcionales. 5](#_Toc40308299)

[Modelo de dominio preliminar. 5](#_Toc40308300)

[Diagrama de casos de uso 7](#_Toc40308301)

[Diagrama de paquetes 7](#_Toc40308302)

[Diagrama de casos de uso de Cuenta 7](#_Toc40308303)

[Diagrama de casos de uso de Reproductor 8](#_Toc40308304)

[Diagramas de casos de uso de Creador de contenido 9](#_Toc40308305)

[Diagrama de casos de uso de Configuración 9](#_Toc40308306)

# Introducción

En el presente documento, se presentarán de manera general los requerimientos del sistema Musicloud. Servicio de música bajo demanda (streaming) en su versión para escritorio. El propósito de Musicloud es proporcionar al usuario la capacidad de reproducir música atreves de internet o localmente, esta última gracias a la característica de descargar música,

Otra de las características proporcionadas es el poder subir su propria música, esto es para los usuarios creadores de contenido, también se podrán crear lista de reproducción personalizadas por parte del usuario consumidor.

Musicloud surge con la finalidad de aplicar los conocimientos adquiridos en la experiencia educativa de Desarrollo de sistemas en red, donde se busca construir los servicios antes mencionados, para ello se utilizarán los lenguajes de programación Ruby y Python, para generar los entornos de desarrollo se hará uso de Vagrant y así crear y configurar las máquinas virtuales de Linux,

Para la construcción del Fronted se hará uso del lenguaje C# ejecutado en de .NET Framework, se llegó esta elección dado que ya se tiene experiencia en el desarrollo y ayudará a optimizar el tiempo de elaboración.

También se describirán aspectos relacionados con el diseño, la arquitectura y las tecnologías a utilizar para el desarrollo del sistema de streaming.

# Proceso

# Artefactos de análisis y diseño

## Clases y caracteristicas de usuarios

* Creador de contenido
  + Usuario que se registra como creador, el cual tiene la capacidad de subir música personal, modificarla, o eliminarla, también puede promocionar sus canciones.
* Consumidor de música
  + Usuario que puede escuchar la música en línea o de ser necesario descargarla y escucharla sin conexión, puede crear sus propias listas de producción y su biblioteca personalizada. Mantiene la experiencia de manejar la calidad de audio, y escuchar listas de radio personalizadas.

## Requerimientos funcionales.

**General**

**RF01.-** El sistema debe ser una aplicación de escritorio.

**RF02.-** El sistema debe permitir la suscripción de usuarios.

**Usuarios**.

**RF03.-** El sistema debe admitir dos tipos de usuario: Consumidor de música y creador de contenido.

**Biblioteca canciones**.

**RF04.-** El sistema debe contar con una biblioteca de canciones compuesta por: Álbumes, Artistas, Canciones y Géneros.

**Biblioteca Personal**.

**RF05.-** El sistema debe permitir al usuario agregar su propia biblioteca de música.

**RF06.-** El sistema debe permitir subir canciones al usuario para su biblioteca personal.

**Cola de reproducción.**

**RF07.-** El sistema debe permitir agregar una canción al final de la lista de reproducción actual.

**RF08.-** El sistema debe permitir agregar una canción a continuación de la reproducción actual.

**Listas de Reproducción.**

**RF09.-** El sistema debe permitir crear listas de reproducción.

**RF10.-** El sistema debe guardar un historial de reproducción del usuario.

**Estaciones de radio.**

**RF11.-** El sistema debe permitir estaciones de radio personalizadas.

**RF12.-** El sistema debe generar a partir de una canción, una estación de radio con canciones del mismo género.

**Calidad de audio**.

**RF13.-** El sistema debe permitir la entrega de las canciones a velocidades de 48kbps a 320kbps.

**RF14.-** El sistema debe permitir al usuario seleccionar entre las siguientes cuatro configuraciones: Bajo, Medio, Alto, Automático para la transmisión del audio.

**RF15.-** El sistema debe permitir al usuario descargar las canciones para escucharlas sin conexión.

## Requerimientos no funcionales.

**Seguridad**.

El sistema debe permitir la autenticación de usuarios.

El sistema debe mantener integra la información de usuarios.

El sistema debe hacer respaldos de la información cada semana,

**Disponibilidad**.

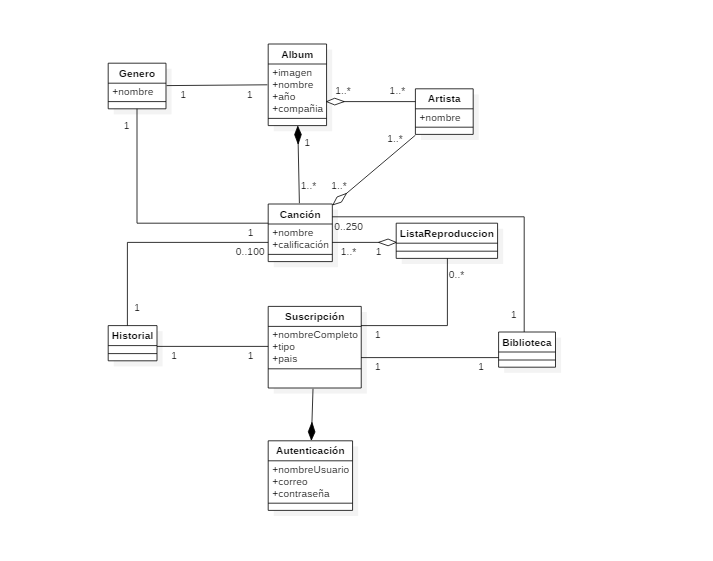
El sistema debe estar disponible el 90% del tiempo en una semana.

El sistema debe ser capaz de estar disponible, aunque una función no esencial esté deshabilitada.

**Escalabilidad.**

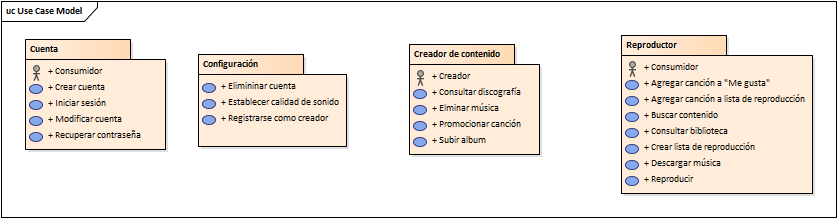
El sistema debe ser capaz de mantener el mismo rendimiento, aunque la demanda de usuarios crezca en al menos un 50%.

## Modelo de dominio preliminar.

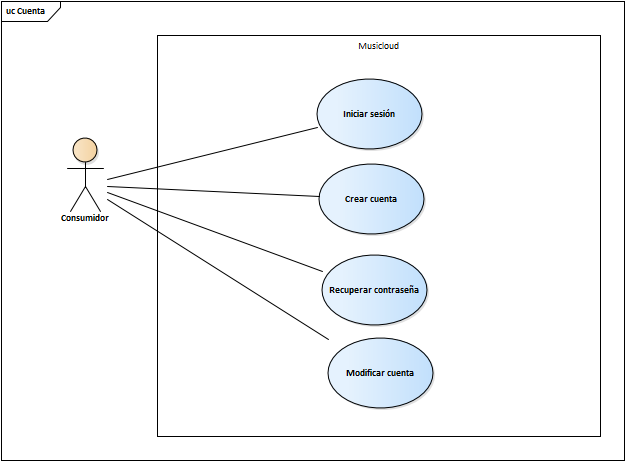


## Diagrama de casos de uso

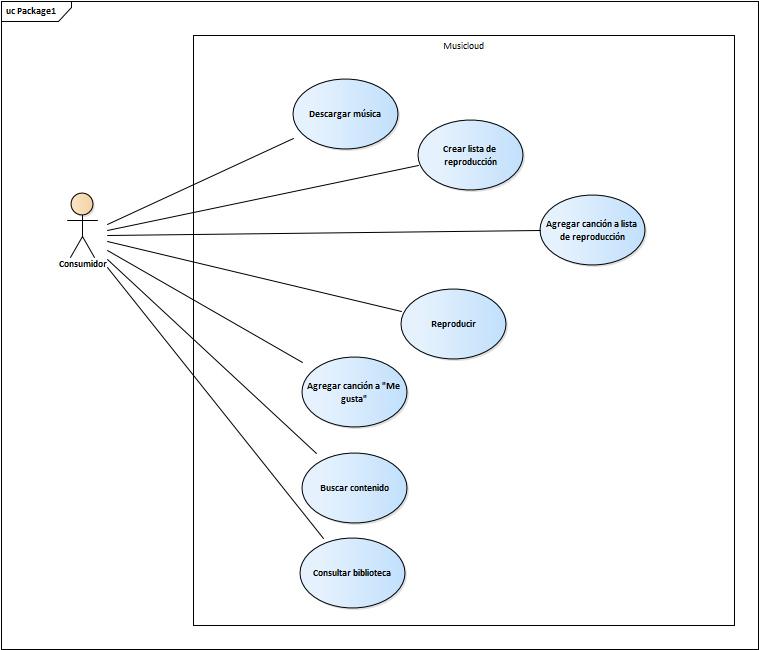
### Diagrama de paquetes



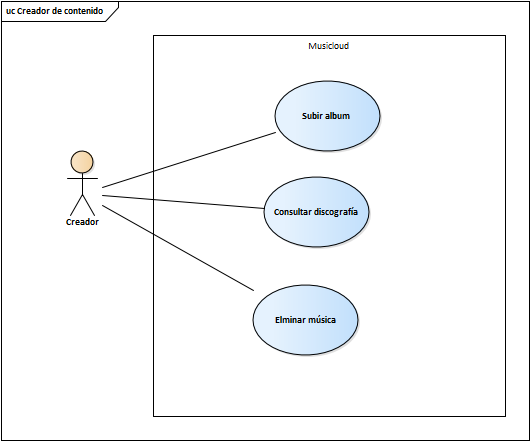
### Diagrama de casos de uso de Cuenta



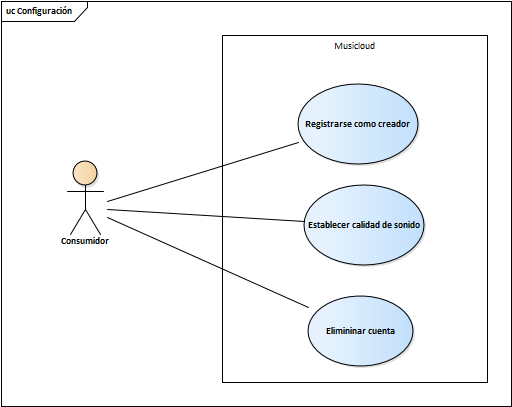
### Diagrama de casos de uso de Reproductor



### Diagramas de casos de uso de Creador de contenido



### Diagrama de casos de uso de Configuración



https://docs.microsoft.com/en-us/dotnet/csharp/getting-started/introduction-to-the-csharp-language-and-the-net-framework