

# El Fogón

Sistema distribuido para músicos con estado compartido del compás, la  
canción y el repertorio

Segundo Cuatrimestre de 2025

Tutor: Diego Montaldo

Luis Waldman

Padrón: 79279

# Índice

<b>Palabras Clave</b>	<b>3</b>
<b>Abstracto</b>	<b>4</b>
Español . . . . .	4
English . . . . .	4
<b>Agradecimientos</b>	<b>6</b>
<b>1. Introducción</b>	<b>7</b>
1.1. Motivación . . . . .	7
1.2. Objetivos . . . . .	7
1.2.1. Objetivo General . . . . .	7
1.2.2. Objetivos Técnicos . . . . .	8
<b>2. Estado del Arte</b>	<b>9</b>
2.1. Sobre el sonido y la música . . . . .	9
2.2. Herramientas . . . . .	9
2.2.1. Pentagramas . . . . .	9
2.2.2. Cancioneros . . . . .	10
2.2.3. Afinadores . . . . .	10
2.2.4. Metrónomos . . . . .	10
2.3. Aplicaciones Web Musicales . . . . .	10
2.3.1. Cancioneros . . . . .	10
2.3.2. Reproductores online de video y acordes . . . . .	11
2.3.3. Editores de partituras online . . . . .	11
2.3.4. Reproductores online de audio y video . . . . .	11
2.4. El futuro llega, hace rato... . . . .	11
2.5. Conclusión . . . . .	11
<b>3. Solución Implementada</b>	<b>13</b>
3.1. WWW.FOGON.AR . . . . .	13
3.1.1. tocar . . . . .	14
3.1.2. Sincronizar . . . . .	15
3.1.3. Buscar y Listar . . . . .	16
3.1.4. Editar . . . . .	17
3.1.5. Configurar . . . . .	19
3.2. Arquitectura del Sistema . . . . .	20
3.2.1. Vista Lógica . . . . .	20
3.2.2. Vista de procesos . . . . .	22
3.2.3. Vista Física . . . . .	24
3.2.4. Vista de Desarrollo . . . . .	25
3.2.5. Vista de Escenarios . . . . .	27
3.3. Tecnologías Utilizadas . . . . .	28
<b>4. Metodología Aplicada</b>	<b>29</b>
4.1. Marco Metodológico . . . . .	29
4.2. Proceso de Desarrollo . . . . .	29
4.3. Herramientas de Desarrollo . . . . .	29
4.4. Gestión del Proyecto . . . . .	29
<b>5. Experimentación y Validación</b>	<b>30</b>
5.1. Diseño de Experimentos . . . . .	30
5.2. Casos de Prueba . . . . .	30
5.3. Resultados Obtenidos . . . . .	30
5.4. Análisis de Resultados . . . . .	30
5.5. Validación con Usuarios . . . . .	30

<b>6. Cronogramas</b>	<b>31</b>
6.1. Cronograma Planificado . . . . .	31
6.2. Cronograma Real . . . . .	31
6.3. Desviaciones y Ajustes . . . . .	31
<b>7. Riesgos y Lecciones Aprendidas</b>	<b>32</b>
7.1. Identificación de Riesgos . . . . .	32
7.2. Gestión de Riesgos . . . . .	32
7.3. Problemas Encontrados . . . . .	32
7.4. Lecciones Aprendidas . . . . .	32
<b>8. Impactos Sociales y Ambientales del Proyecto</b>	<b>33</b>
8.1. Impacto Social . . . . .	33
8.2. Impacto Ambiental . . . . .	33
8.3. Responsabilidad Social y Ética . . . . .	33
8.4. Medidas de Mitigación . . . . .	33
<b>9. Desarrollos Futuros</b>	<b>34</b>
9.1. Mejoras Propuestas . . . . .	34
9.2. Funcionalidades Adicionales . . . . .	34
9.3. Escalabilidad . . . . .	34
9.4. Investigación Futura . . . . .	34
9.5. Conclusiones Finales . . . . .	34
<b>Referencias</b>	<b>35</b>

## Palabras Clave

- Web
- Vue.js
- WebSocket
- WebRTC
- Diseño responsivo
- Golang
- NUnit
- Playwright
- Sincronización
- Compensación de *jitter*
- Música
- Letras
- Acordes
- Partituras

## Abstracto

### Español

Fogón es un sistema distribuido orientado a facilitar la práctica musical y la sincronización entre músicos. A cada uno le ofrece, desde una aplicación web progresiva, vistas para su instrumento: letras, acordes o partituras. Permite crear listas y editar canciones. También, crear sesiones en la que los participantes comparten el estado de la canción, el repertorio y hasta el compás exacto que están tocando.

El sistema ayuda al aprendizaje y la enseñanza musical al mostrar cómo realizar los acordes en cada instrumento y al permitir afinarlos.

La arquitectura soporta numerosas vistas complejas y responsivas para distintos instrumentos, y permite agregar nuevas vistas para otros instrumentos. Para las partituras emplea la librería VexFlow, mientras que para las vistas de letra y acordes se usó un desarrollo propio.

El principal desafío técnico fue el de la reproducción en dispositivos distribuidos: un "delay" de 20 ms empieza a ser perceptible por el oído humano y la latencia en internet es mayor. Se implementó un protocolo que sincroniza los dispositivos usando WebSocket y WebRTC (con compensación de jitter) a través de un servidor Golang.

Para probar y "debuggear" el sistema de sincronización hubo que desarrollar algunas vistas y controles.

Las pruebas de aceptación se desarrollaron de punta a punta con NUnit y Playwright.

**Palabras Clave:** Web, Vue.js, WebSocket, WebRTC, Diseño responsivo, Golang, NUnit, Playwright, Sincronismo, Compensación de \*jitter\*, Música, Letras, Acordes, Partituras.

### English

Fogón is a distributed system designed to facilitate musical practice and synchronization among musicians. It provides each musician with a progressive web application featuring instrument-specific views: lyrics, chords, or sheet music. It allows users to create playlists and edit songs. Additionally, it enables the creation of sessions where participants share the song state, repertoire, and even the exact bar they are playing.

The system aids in musical learning and teaching by showing how to perform chords on each instrument and allowing for tuning.

The architecture supports numerous complex and responsive views for different instruments, and allows for the addition of new views for other instruments. For sheet music, it employs the VexFlow library, while for lyrics and chord views, a custom development was used.

The main technical challenge was distributed device playback: a delay of 20 ms starts to become perceptible to the human ear, and internet latency is typically higher. A protocol was implemented

that synchronizes devices using WebSocket and WebRTC (with jitter compensation) through a Golang server.

To test and debug the synchronization system, several views and controls had to be developed. End-to-end acceptance tests were developed with NUnit and Playwright.

**Keywords:** Web, Vue.js, WebSocket, WebRTC, Responsive design, Golang, NUnit, Playwright, Synchronization, Jitter compensation, Music, Lyrics, Chords, Sheet Music.

## Agradecimientos

A mi madre y a mi padre,

A la Educación Pública y en particular a las cátedras de Arquitectura de Software, Base de Datos, Introducción a Sistemas Distribuidos y Sistemas Distribuidos de la Universidad de Buenos Aires.

A Pitágoras, a Newton, a Turing y a todos los que asumen la heroica tarea de descubrir y transmitir ciencia.

A los artistas que prefieren estructuras rigurosas.

A las Cadenas de Márkov y la proliferación de IAs.

A vos que estás leyendo esto.

# 1 Introducción

Tanenbaum y Van Steen definen: “Un sistema distribuido es una colección de computadoras independientes que aparece ante sus usuarios como un sistema único y coherente” [1]

La definición coincide con la de un grupo musical que combina armonías, melodías y ritmos de modo que se escuche como un tema único y coherente.

Para hacer esto los músicos se nutren de protocolos y mecanismos de coordinación que les permiten sincronizarse y compartir información.

Los avances en la ciencia y la ingeniería fueron incorporados por los músicos: Pitágoras sintetizó la matemática y la armonía; la imprenta permitió la publicación de partituras; la revolución industrial introdujo el metrónomo de Maelzel.

Desde que existe Internet, circulan archivos con páginas de acordes que evolucionaron a páginas multimedia, archivos MIDI, aplicaciones de edición de partituras, etc.

La aplicación .<sup>El</sup> Fogón”pone a disposición de los músicos estos avances y propone un nuevo enfoque: cada músico puede acceder con su dispositivo a un Estado compartido de compás, canción, acordes, partituras, repertorio, etc.

Esto permite que varios músicos toquen juntos y en sincronía, compartiendo y actualizando información en tiempo real, pero cada uno ve la información de su instrumento.

## 1.1 Motivación

Busca hacer un aporte novedoso a la música desde la informática, incorporando soluciones anteriores y agregando un enfoque novedoso: el estado compartido entre músicos.

Además, busca la alta disponibilidad: los músicos pueden acceder a la aplicación en su dispositivo en cualquier momento, sin necesidad de conexión a la red.

## 1.2 Objetivos

Fogón es una solución dirigida tanto a cantantes y guitarristas aficionados como a músicos de orquestas profesionales: una aplicación en donde puedan buscar letras de acordes y canciones de modo intuitivo y también una herramienta que los ayude a ensayar y crear cosas nuevas.

### 1.2.1 Objetivo General

Cada músico podrá ver en su dispositivo la vista de su instrumento: el cantante, la letra; el guitarrista, los acordes; el pianista, sus partituras. Tendrá autoscroll y subrayado automático del compás actual; podrá editar los tamaños de letra y acordes.

El público podrá ver y editar una variedad de canciones publicadas en el mismo sitio. Si se loguea con su usuario, podrá compartir sus canciones con otros usuarios.

Varios usuarios podrán unirse en una sesión para sincronizar la lista de canciones, la canción que se está reproduciendo y el compás actual: de este modo podrán organizar un ensayo, un concierto o una



noche de karaoke entre amigos.

### **1.2.2 Objetivos Técnicos**

La sincronización del compás en una sesión debe ser exacta cuando un grupo de músicos está tocando: un delay de 20 ms empieza a ser perceptible por el oído humano y la latencia en Internet puede ser mayor. Los distintos dispositivos se conectarán con un servidor Golang e implementarán un protocolo que combine timestamps sincronizados (basados en NTP), buffers adaptativos y compensación del jitter (variación en la latencia) para resolver esto.

El mismo servidor, además, por medio de HTTP intercambia los archivos de las canciones con las aplicaciones.

Todas las vistas deberán poder adaptarse a distintos dispositivos, ser configurables y extensibles: será posible incorporar vistas adicionales, como notación numérica para armónica, tablaturas para guitarra y reproductores multimedia como YouTube o MIDI.

Edición de letra y acordes: también podrán editar las canciones mediante una interfaz intuitiva y accesible; la compleja relación entre las letras y los acordes, que además se agrupan en partes que se repiten según una secuencia, podrá modificarse de una manera natural y sencilla.

Sincronización: varios usuarios logueados podrán unirse en una sesión para sincronizar la lista de canciones, la canción que se está reproduciendo y el estado de la reproducción.

Construir algunas herramientas necesarias para la música, como un afinador que permita afinar distintos instrumentos.

Construir herramientas para probar y "debuggear" el sistema de sincronización desarrollado.

## 2 Estado del Arte

Como es parte nuestro "negocio", describiremos algunos conceptos sobre la naturaleza del sonido y de la música.

Luego, comentaremos algunas tecnologías destacadas que tomamos en el Fogón y sobre las novedades que llegaron con internet. También, repasaremos aplicaciones web con IA que ofrecen servicios relacionados con la música.

Finalmente, explicaremos cómo nuestro enfoque es novedoso y aporta valor, comparado con las herramientas antes mencionadas.

### 2.1 Sobre el sonido y la música

.El sonido es una onda mecánica que se propaga a través de un medio elástico, como el aire o el agua.<sup>z</sup> que sus Característica principales sin la frecuencia, la amplitud y el timbre.

La frecuencia determina el tono o la nota musical, la amplitud el volumen y el timbre la calidad del sonido, permite distinguir entre diferentes instrumentos que tocan la misma nota.

La mayoría de los humanos no podemos determinar la frecuencia a la que vibra una cuerda, pero si podemos distinguir si produce un sonido agradable. Pitágoras descubrió, en el siglo VI a.C., que para producir sonidos agradables entre dos cuerdas vibrantes tienen que tener relaciones matemáticas. Ej: (2:1)Una octava, (3:2)una quinta justa. (4:3)una cuarta justa.

Todos conocemos la definición de música acerca de hacer algo con la melodía, la armonía y el ritmo. Pues bien, la melodía es como cambia la frecuencia en el tiempo, la armonía es la combinación de varias frecuencias. El ritmo es la repetición de sonidos (o volúmenes o intenciones) en intervalos regulares. En la música suelen organizarse de 2, 3 o 4 tiempos, formando compases.

### 2.2 Herramientas

#### 2.2.1 Pentagramas

15 Siglos antes que Pitágoras, por el 2000 A.C., en la Mesopotamia se usaban tablillas de arcilla para anotar música, con símbolos que representaban las alturas de las notas.

Recien en el siglo IX, el monje benedictino Guido d'Arezzo desarrollo un sistema de notacion musical basado en cuatro líneas y espacios, que luego evolucionó al pentagrama con cinco lineas que usamos hoy.

solo 50 años años despues de la invencion de la imprenta, Ottaviano Petrucci, en 1501, publica "Harmonice Musices Odhecaton", con las primeras partituras impresas con tipos móviles, permitiendo la difusión masiva de la musica escrita.

### 2.2.2 Cancioneros

En el siglo XIX, se transmitían los acordes folclóricos de forma oral, pero para el siglo XX, en la música popular (tango, rock, jazz) se empieza a usar la notación de acordes sobre la letra de la canción.

Se difunden las revistas de acordes y letras, y luego los cancioneros impresos. Con la llegada de internet, surgen diversas páginas y aplicaciones que revisamos en la próxima sección.

### 2.2.3 Afinadores

En el siglo XVII, Marin Mersenne formuló las leyes que gobiernan la vibración de cuerdas tensas, permitiendo una base física para afinar cuerdas y abrió el camino hacia la acústica moderna.

$$f = \frac{1}{2L} \sqrt{\frac{T}{\mu}}$$

donde  $f$  es la frecuencia,  $L$  es la longitud de la cuerda,  $T$  es la tensión aplicada y  $\mu$  es la densidad lineal de masa.

Se afinaba con un diapason, un metal que siempre sonaba con la misma frecuencia y que se tomaba como nota de referencia.

En 1939 durante la conferencia de Londres científicos y músicos acordaron que la nota La<sub>4</sub> se afinaría a 440 Hz, es decir, 440 vibraciones por segundo.

En 1980 se popularizaron afinadores digitales compactos, portátiles y precisos.

### 2.2.4 Metrónomos

En 1815 Johann Maelzel patentó el metrónomo mecánico, ganándose el reconocimiento de Beethoven y popularizándose en toda Europa.

En el siglo XX llegaron metrónomos electrónicos, que permiten mayor precisión y funcionalidades adicionales como sonidos personalizables, luces intermitentes y conectividad con otros dispositivos.

## 2.3 Aplicaciones Web Musicales

Con dispositivos móviles inteligentes, escuchar un sonido y procesarlo para detectar su frecuencia es muy sencillo, igual que repetir un comportamiento cada un periodo constante de tiempo. Es por eso, que hubo metronomos y afinadores en cada celular desde el comienzo de este siglo.

Las próximas líneas transcurrirán sobre aplicaciones web que son usadas actualmente por músicos, pero además de señalar solamente sus aportes, se hará foco en sus debilidades.

### 2.3.1 Cancioneros

Los cancioneros online Reemplazaron a los folletines ofreciendo la diversidad de la web pero copiaron un defecto.

El músico está tocando su instrumento, ajustado perfecto con el metrónomo, cuando llega al último acorde de la página (o de la pantalla) y lo obliga a soltar su instrumento para manipular el cancionero, perdiendo así el ritmo.

Las paginas suelen en su mayoría estar diagramadas en la pantalla pensado en maximizar el espacio para publicidad y no en la usabilidad del musico, por lo que ofrecen poca personalizacion en la vista. Esto hace que aunque muchas ofrezcan instrucciones sobre como se realizan los acordes en el instrumento, o algunos implementen un rudimentario autoscroll, sea poco visible para el musico.

De las que tienen contenido argentino, las principales son lacuerda.net, cifraclub y acordesweb.

Y claro, muestra solo los acordes. Algunas paginas de acordes, sin embargo, son paginas distintas!

### 2.3.2 Reproductores online de video y acordes

Estas aplicaciones web ofrecen videos con acordes sincronizados, como Chordify y Ultimate Guitar, ajustan la reproduccion a videos de YouTube, apenas permite editar los acordes

### 2.3.3 Editores de partituras online

Los editores de partituras online permiten crear, editar y compartir partituras musicales a través de una interfaz web. musescore.com destaca por su buen balance entre contenido gratuito y de pago y su comunidad activa de usuarios,

### 2.3.4 Reproductores online de audio y video

Como YouTube y Spotify, permiten reproducir audio y video en línea, pero no están diseñados para músicos que tocan juntos. Estas aplicaciones suelen permitir compartir una lista de reproducción.

## 2.4 El futuro llega, hace rato...

Al momento de escribir esto, diciembre de 2025, los resultados del uso de la IA vienen avanzado enorme y exponencialmente.

Logran generar todo tipo de contenido, letras y partituras, audios y videos.

Tambien permiten procesar audio para separar los instrumentos y transcribir partituras. Sobresale en ese sentido la aplicacion Moises.ai y Spleeter.

## 2.5 Conclusión

Mostramos una tabla comparativa de las herramientas presentadas:

Característica	Metronomo	Diapasón	Cancioneros	Cancioneros Web	Reprod. video y acordes	Edit. partituras online	Reprod. audio y video	Herramientas Con IA	Fogon.ar
Marca el ritmo?	Sí	No	No	No	No	No	No	Sí	Sí
Afina?	No	Sí	No	No	No	No	No	Sí	Sí
Muestra acordes?	No	No	Sí	Sí	Sí	Sí	No	Sí	Sí
Se actualiza con la canción?	No	No	No	Algunos	Sí	No	No	Sí	Sí
Muestra partitura?	No	No	No	No	No	Sí	No	Sí	Sí
Funciona sin Internet?	Sí	Sí	Sí	No	No	No	No	No	Sí
Permite estados compartidos?	No	No	No	No	No	No	Algunos	No	Sí

Cuadro 1: Comparativa de herramientas y aplicaciones musicales

Se deduce entonces que el estado compartido es un enfoque novedoso para aplicaciones web dedicadas a músicos.

La gran cantidad de tipos de aplicaciones web musicales existentes, cada una con su solución particular, muestra la necesidad de unificar herramientas para mejorar la experiencia del músico.

El criterio de privilegiar el aspecto educativo y de usabilidad está ausente en los cancioneros online y en la mayoría de las aplicaciones web musicales.

### 3 Solución Implementada

En esta sección está la descripción de la solución implementada y una explicación técnica basada en el modelo de vistas 4+1 de Kruchten [2].

En la última vista, la vista de escenarios, los escribimos de igual modo que a las pruebas de aceptación en Reqroll [3].

Por último, incluimos una revisión de las tecnologías utilizadas.

#### 3.1 WWW.FOGON.AR

Es una aplicación progresiva (PWA) y multiplataforma que permite buscar y tocar canciones, y también crear, editar y compartirlas.

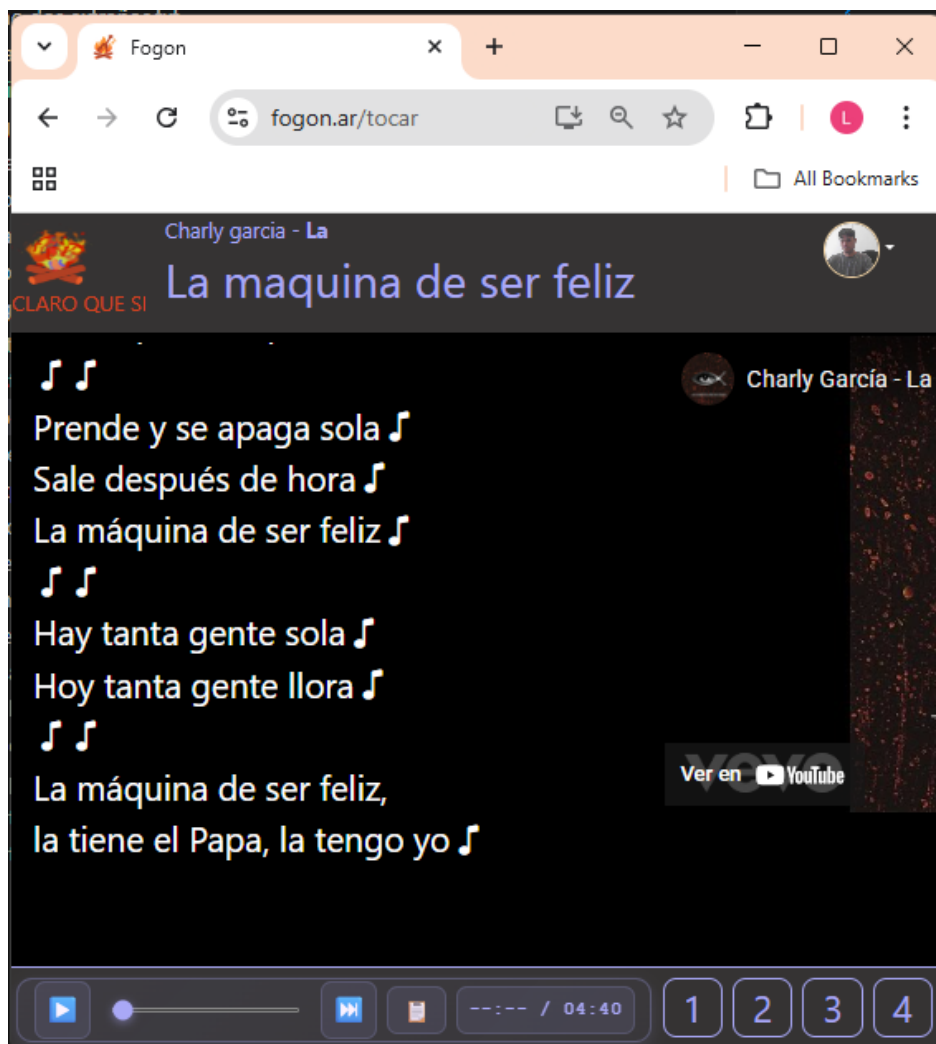


Figura 1: "La maquina de ser feliz", tocando con un video

Además, Permite crear sesiones colaborativas para llevar, en un estado compartido, el compás y la canción, de modo que cada músico vea las instrucciones para su instrumento, en su dispositivo, de manera sincronizada.

Administra la lista de reproducción y listas en general, guardandolas localmente o en el servidor.

Ofrece tambien otras herramientas utiles para los musicos, como un afinador y cambios automáticos de escala.

### 3.1.1 tocar

La pagina tocar muestra la vista segun el musico:

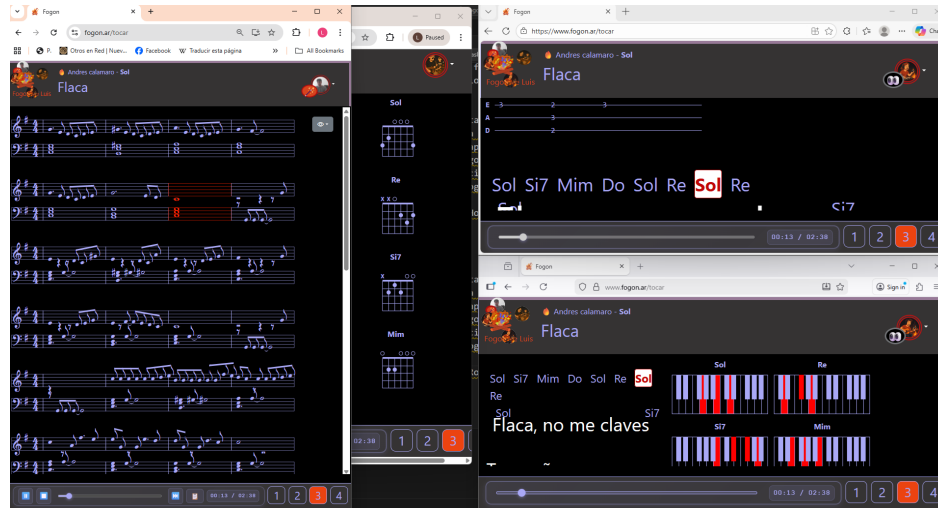


Figura 2: Página tocar con distintos instrumentos en varios exploradores

En la cabecera, muestra los datos de la cancion y permite cambiar la vista y la Configuración.

A lo largo de la pantalla, la vista particular del instrumento.

Debajo, el control de reproduccion y el metrónomo.

**icono-fogon** Arriba a la izquierda, el icono del fogon muestra el estado general de la aplicación.

Marca el pulso cuando estan tocando y muestra los usuarios en la sesión.

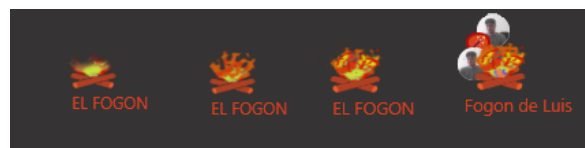


Figura 3: Iconos para los estados: desconectado, conectado, logueado, o en sesion.

**menu** Arriba a la derecha, el icono de menu abre las opciones para configurar la vista y el usuario.

**vistas** Para configurar su vista, cada musico accede desde el menu a "Ver":

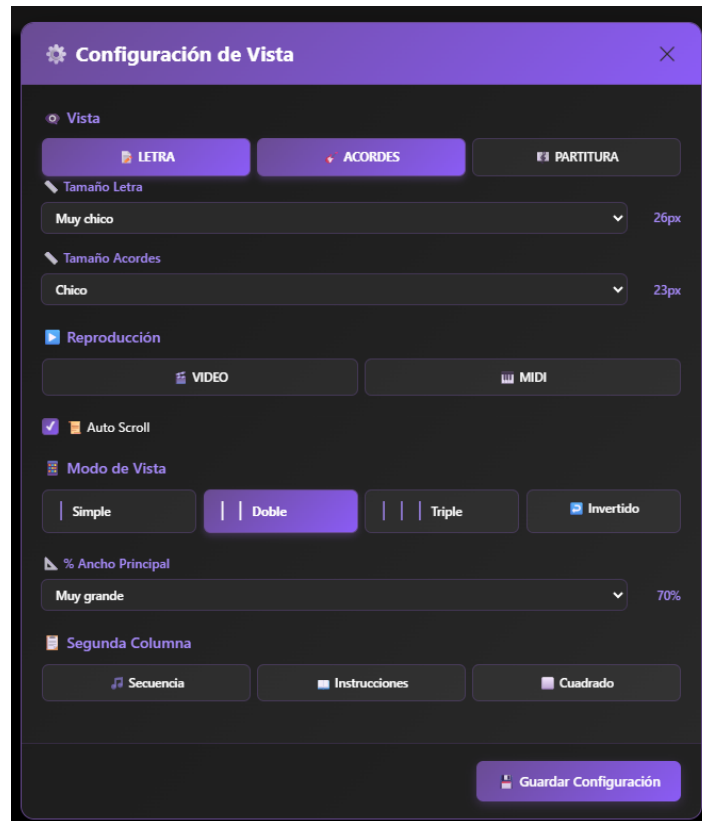


Figura 4: Configuración de la vista

Desde aquí puede elegir entre ver letras y/o acordes o la partitura. Además, puede ajustar el tamaño de la letra, los acordes y la partitura para ajustarla a cada dispositivo.

En la opción de reproducción, puede elegir acompañar la reproducción con un video o con un midi generado a través de las partituras.

En las últimas opciones, se ajuste la cantidad de columnas que se muestra en pantalla y si muestran instrucciones para el músico, la secuencia de acordes o la pantalla para reproducir MIDI.

### 3.1.2 Sincronizar

Descripción de la funcionalidad de sincronización.

Cuando un usuario crea un fogón, puede invitar a otros a unirse a la sesión colaborativa. Desde el menú, también, puede asignar distintos roles a cada usuario:



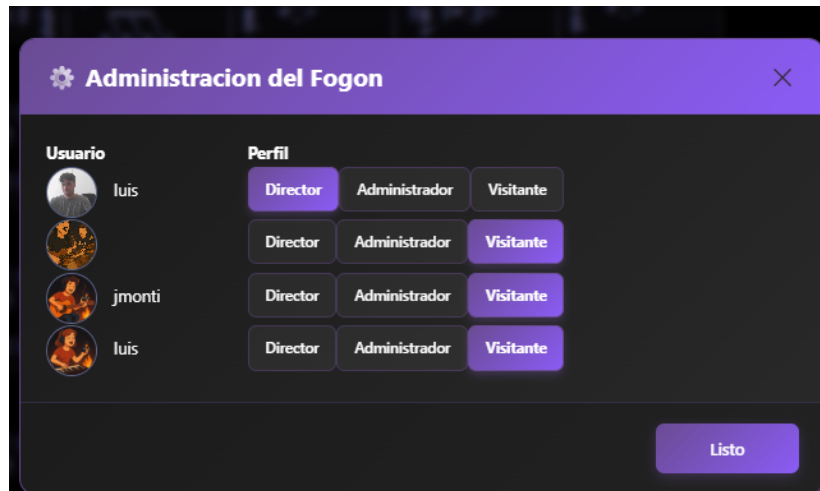


Figura 5: Configuración de roles en la sesión

Todos los usuarios ven la misma canción y el mismo compas, pero cada uno ve las instrucciones para su instrumento, en su dispositivo. Los administradores pueden cambiar la canción y controlar la reproducción. El director, es el único capaz de reproducir algún video o audio.

### 3.1.3 Buscar y Listar

El fogon, además de permitir buscar canciones por banda o por artista, tienen una serie de filtros multiopcion

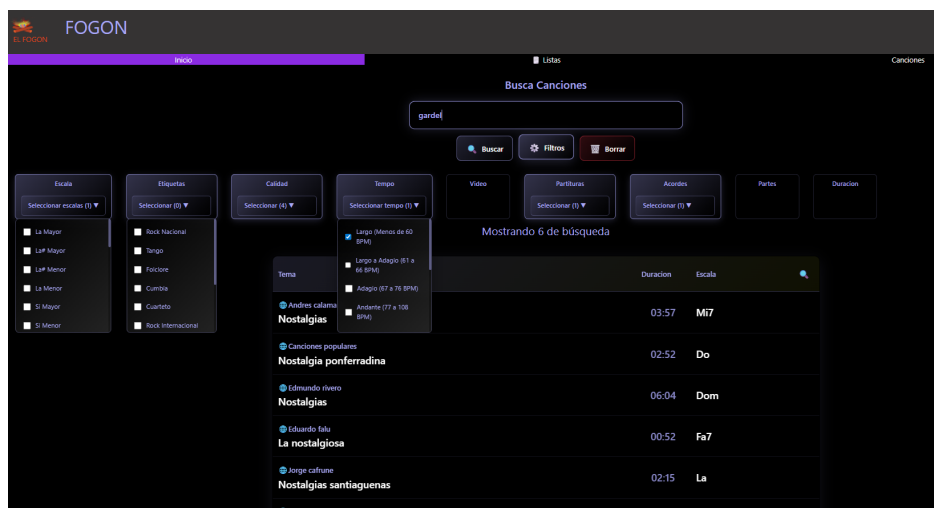


Figura 6: Búsqueda de "garden" con los filtros desplegados de Escala, etiqueta y tempo

De este modo, permite buscar por:

- Escala
- Etiquetas
- Calidad

- Tempo
- Si tiene o no Videos o Partituras
- Cantidad de Acordes
- Cantidad de Partes
- Duracion

Permite armar listas de reproduccion tanto locales como en el servidor.

#### **3.1.4 Editar**

Se puede editar o una cancion, o crear una nueva desde cero, eligiendo sus principales características como escala, secuencia armonica y tipo de letra.

**Nueva Canción** ✕

**Tema**  
tema

**Banda**  
banda

**Tempo (BPM)** **Compás**  
80 4/4

**Modelo de Letra** **Escala**  
Cancion 16 versos Do Mayor

**Funciones Armónicas**

I-IV-V-I I-II-IV-V I-II-V-VI I-III-VI-V  
I-III-V-VI I-VI-IV-V I-V-VI-V I-VI-II-V  
I-V-I-IV

**Estructura**  
Intro Puente Outro

Cancelar + Crear Canción

Figura 7: PopUp para crear una nueva cancion

En la pantalla de edicion, se puede modificar la letra, los acordes y las partes de la cancion.

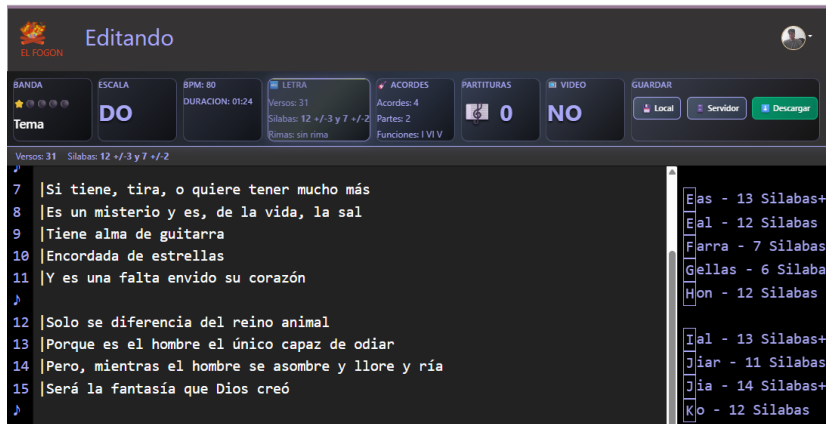


Figura 8: Ej de pantalla de edicion de letra

Editando la escala, es posible transponerla a cualquier otra tonalidad.

Tambien se puede agregar o editar pentagramas e importarlos desde archivos MusicXML.

### 3.1.5 Configurar

En la pantalla de Configuración se administran los datos basicos del usuario.

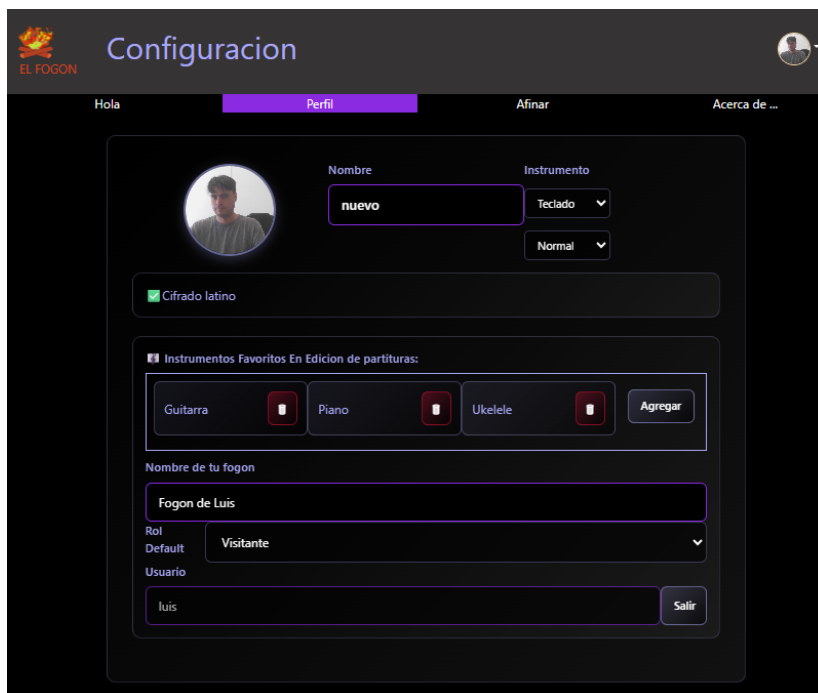


Figura 9: Pantalla de configuración del usuario

Desde aqui el musico puede cambiar su nombre, el instrumento que toca y la imagen que representa su dispositivo.

Configura de que modo se crean sus sesiones: el nombre y el rol default de los nuevos usuarios que se unan a la sesion.

Permite configurar el usuario para conectarse con el servidor.

Ademas, en esta pantalla, desde la barra superior, se puede acceder a las secciones: "Hola", "Afinar",

”Acerca de...”

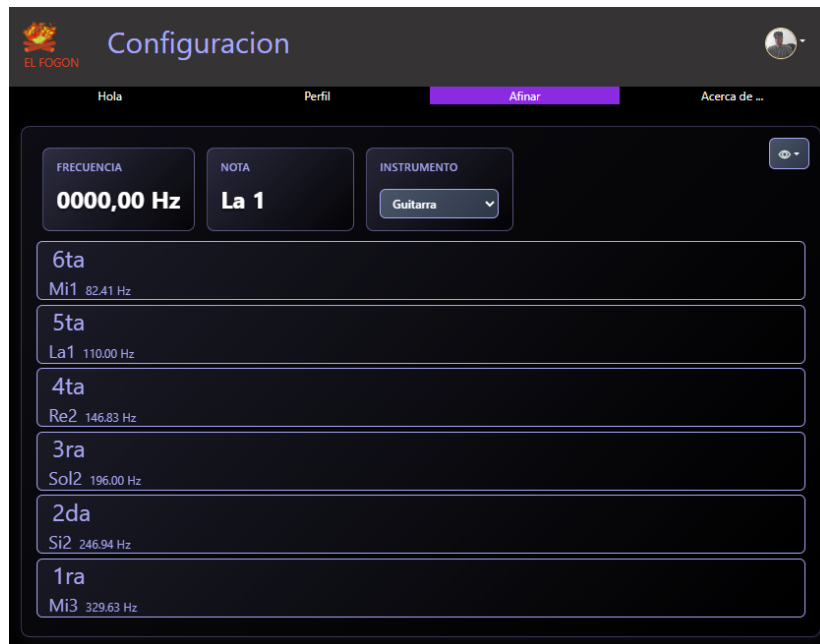


Figura 10: Pantalla de afinar instrumentos

## 3.2 Arquitectura del Sistema

Para describir la arquitectura del sistema, utilizamos el modelo de vistas 4+1 de Kruchten [2].

En la vista lógica detallaremos componentes principales del cliente y las clases que permiten comunicarse con el servidor. Luego, en la vista de procesos, veremos el protocolo para sincronización y mantener el estado compartido entre sesiones. En la vista física veremos el mapeo físico en la red en la que ocurren dichos procesos. El 4 del 4+1 es la vista de desarrollo, donde describimos la organización del código. Y el +1: la vista de escenarios, que serán descritos en formato Reqroll. [3].

### 3.2.1 Vista Lógica

Presentamos el modelo de datos, la clase principal es “Cancion”.

Luego la clase “Aplicacion”, que funciona como orquestador de los controles de la interfaz y la conexión con el backend, y por último la clase “Reproductor”, que maneja la reproducción de audio y la sincronización con los acordes y letras.

**Canción** Tiene dos propiedades de clase Letra y Acordes; además de propiedades título, artista, bpm, compás, etc.

La clase Acorde diseñada como una secuencia de partes, cada parte formada por una serie de acordes por compas. La propiedad “secuencia” es la lista de partes como se tocan en la canción.

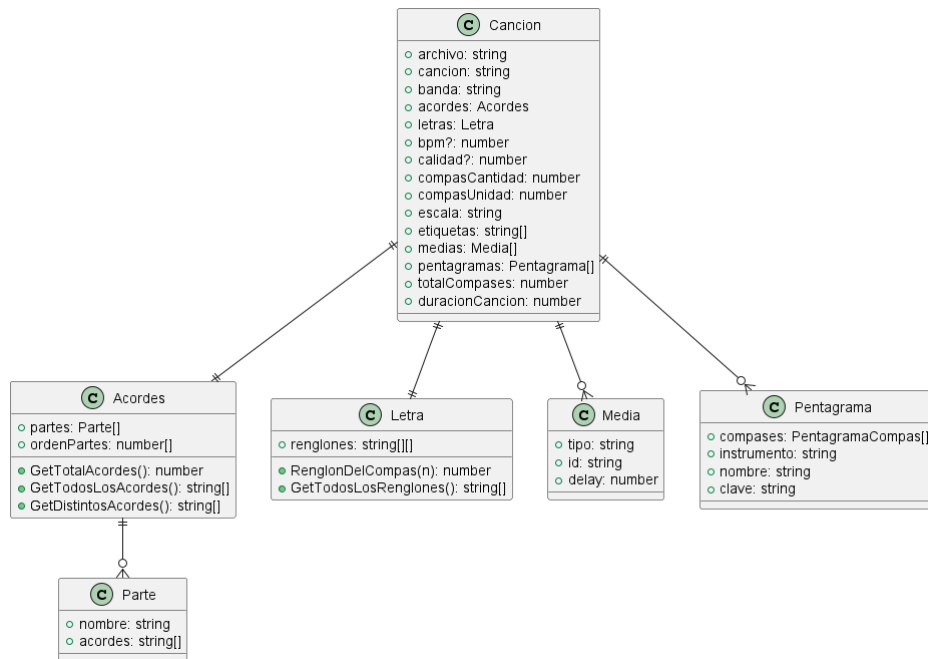


Figura 11: Diagrama de clases del modelo de Canción

**Aplicación** La clase Aplicacion orquesta entre el reproductor, la conexion y la interfaz de usuario. Maneja las clases ConexionManager para la comunicación con el servidor, Reproductor para la reproducción de la cancion, AutenticacionManager para el login/logout, y SesionManager para el manejo de sesiones colaborativas.

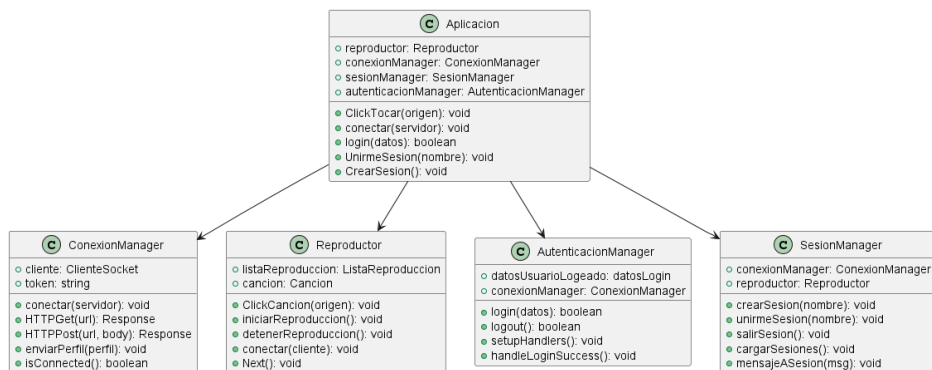


Figura 12: Diagrama de clases del sistema de Aplicación

**Reproductor** La clase reproductor administra la cancion y el momento actual. Implementa el patrón Strategy para reproducir en una sesion o en modo desconectado.

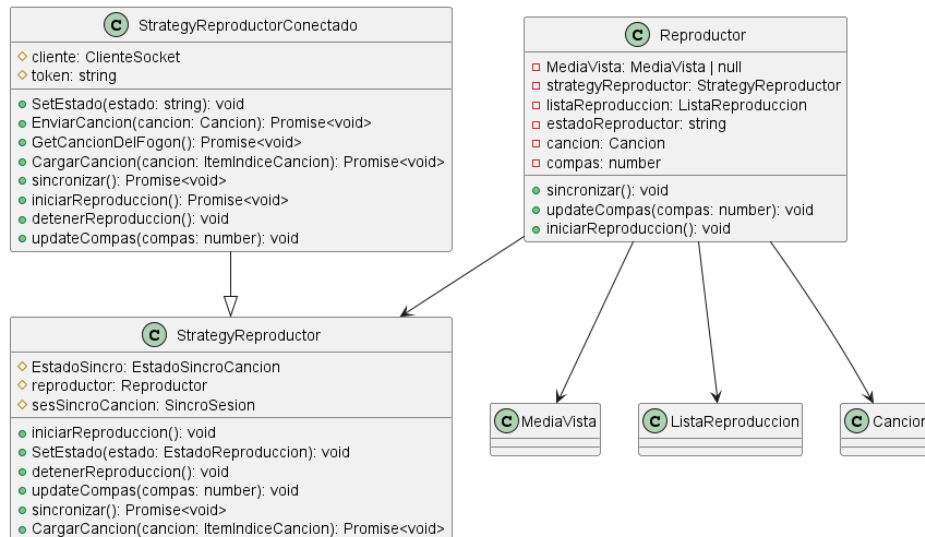


Figura 13: Diagrama de clases del Reproductor

En el siguiente capítulo esta explicado como interactúan estas clases.

### 3.2.2 Vista de procesos

Aquí repasaremos, con diagramas de secuencia, como la clase reproductor utiliza un Strategy para manejar la reproducción en modo desconectado y en modo conectado.

Ejemplificamos con el proceso IniciarReproduccion, pero debe suponerse uno similar para DetenerReproduccion, CambiarCompas, CambiarLaListaDeCanciones, etc.

Luego, mostraremos la secuencia con la que logra sincronizar relojes con el servidor y entre clientes usando WebRTC.

**IniciarReproduccion en modo local** El reproductor delega en la estrategia "StrategyReproductor" desconectada, que prepara la canción en el cliente.

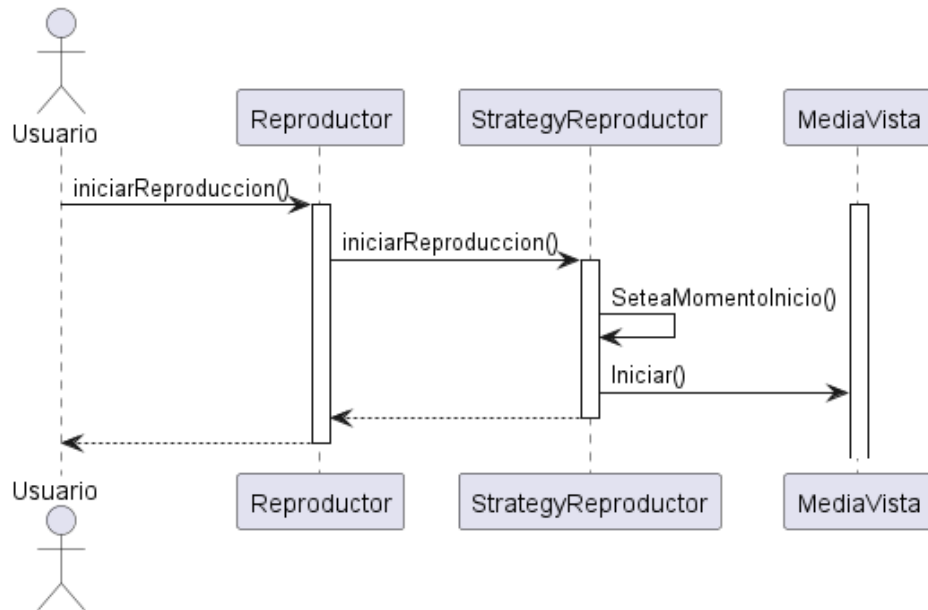


Figura 14: Secuencia de inicio de reproduccion en modo local

**IniciarReproduccion en un Fogon** Cuando se esta en una sesion, se manda al servidor la orden de iniciar reproduccion, y se espera la señal de inicio coordinado.

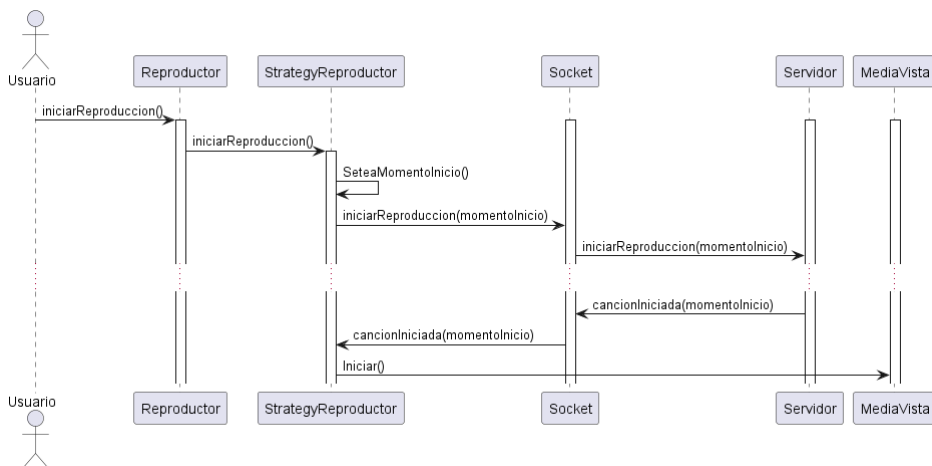


Figura 15: Secuencia de inicio de reproduccion distribuida: esta vez, la estrategia incluye al servidor y luego se inicia coordinadamente.

**Sincronizar relojes** Se muestran dos tipos de sincrozicaciones que se realizan en el sistema: por socket, cuando el usuario se conecta al servidor, sincorniza su reloj con este. Cuando se conecta a un fogon, envia una oferta WebRTC para establecer un canal. La aplicacion puede sincronizarse con un usuario particular a pedido del servidor o del usuario: el primero porque los supone en la misma red WiFi, el usuario desde la configuracion. Vemos el intercambio de mensajes:





Figura 16: Secuencia de sincronizacion de reloj

### 3.2.3 Vista Física

En el diagrama de despliegue vemos que nuestra aplicacion cliente es Web y Progresiva, por lo que corre en el dispositivo del usuario, aunque busca en github pages el html, css y js necesarios; ademas de los indices.

El servidor en Go esta alojado en los servidores de Render y se conectan a una base MongoDB Atlas. Con algunos scripts, se actualizan canciones, por ejemplo para extraer la letra del video de YouTube.

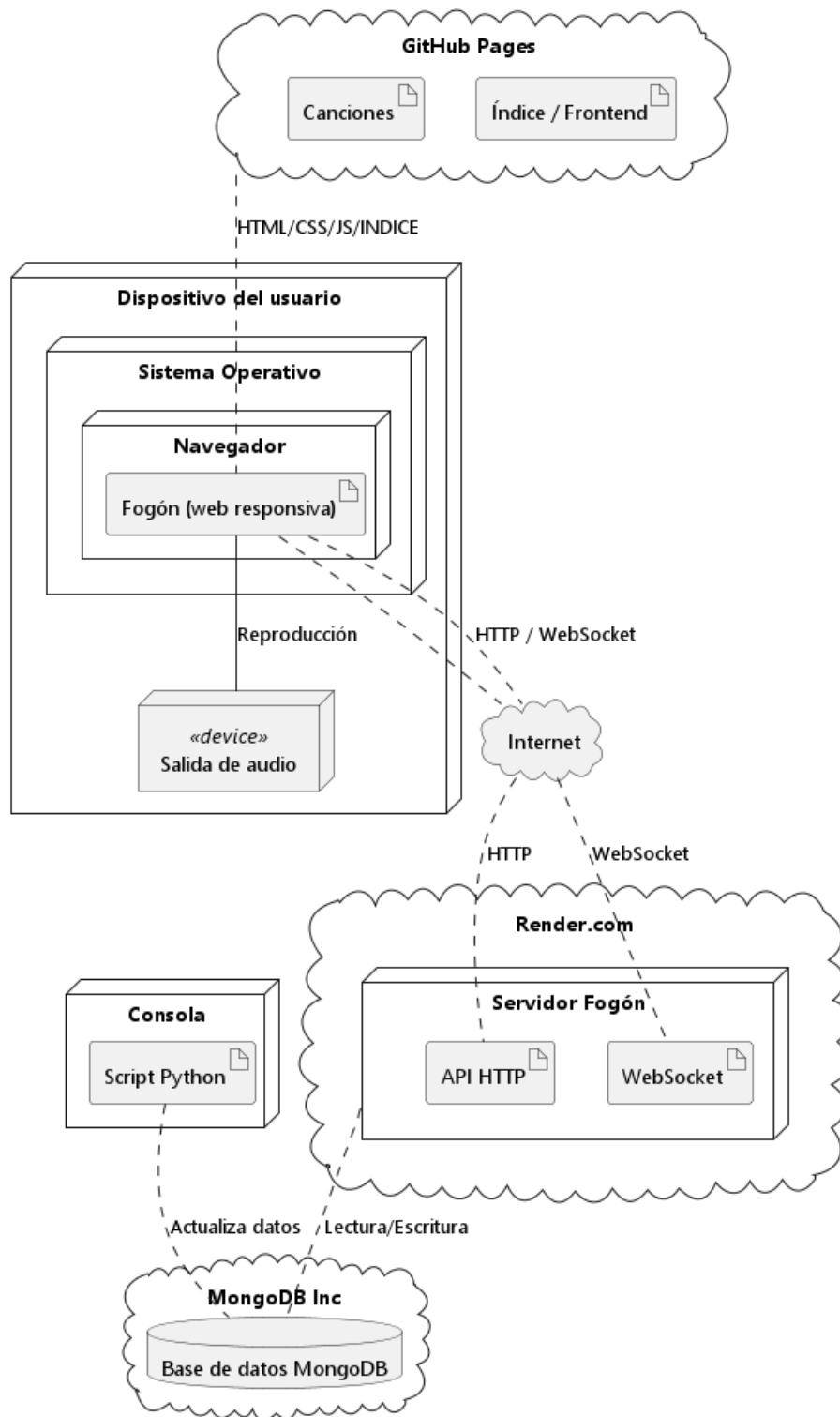


Figura 17: Diagrama de despliegue

### 3.2.4 Vista de Desarrollo

Estos son los principales paquetes del cliente, las vistas, los helpers y la aplicación.

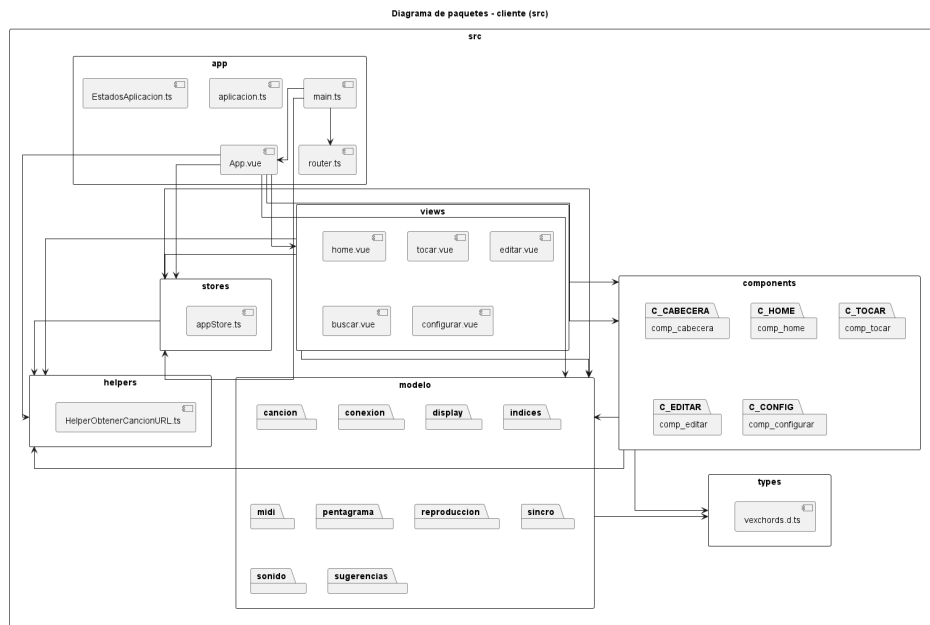


Figura 18: Diagrama de paquetes del cliente

En el servidor, los servicios acceden a los datos, a veces de modo directo o usando algunas reglas de negocios para listas o usuarios. Los controladores usan a los servicios para atender las peticiones y manipular los datos y la aplicacion, es creada desde el main.

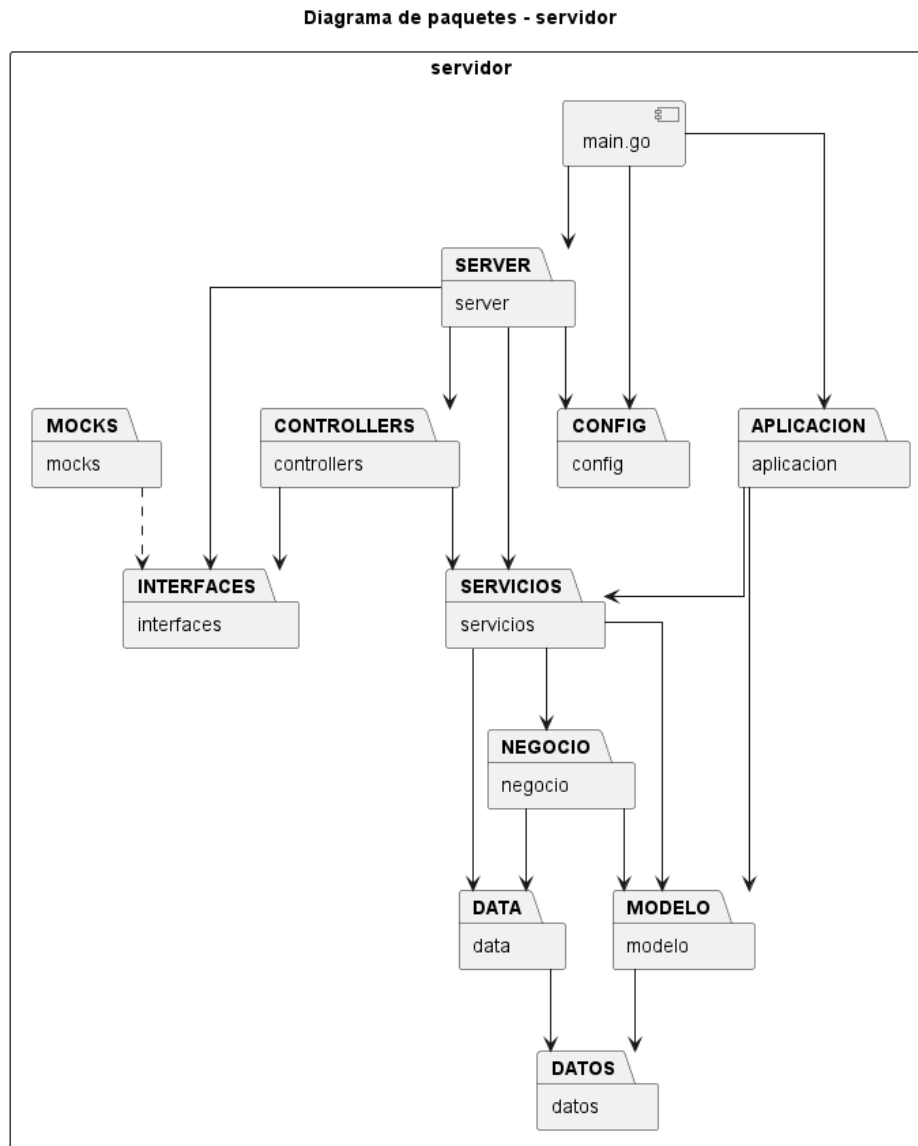


Figura 19: Diagrama de paquetes del servidor

### 3.2.5 Vista de Escenarios

La vista de escenarios (la -1ª del modelo de Kruchten) ilustra cómo las otras cuatro vistas trabajan juntas a través de casos de uso concretos. Documentamos los escenarios principales usando el formato de pruebas de aceptación de Reqnroll (Gherkin).

**Escenario 1: Tocar una canción en solitario** **Contexto:** Un músico quiere buscar y tocar una canción desde su dispositivo.

Listing 1: Escenario: Búsqueda y reproducción de canción

```

1 Feature: Tocar canciones
2   Como musico
3   Quiero buscar y tocar canciones

```

```
4   Para practicar con mi instrumento
5
6   Scenario: Buscar y reproducir una cancion
7       Given que soy un usuario no autenticado
8       When ingreso "la maquina de ser feliz" en el buscador
9       Then veo una lista de resultados con esa cancion
10      When selecciono "La maquina de ser feliz - Charly Garcia"
11      Then veo la vista de mi instrumento (acordes para guitarra)
12      When presiono el boton de reproducir
13      Then comienza el autoscroll sincronizado con el metronomo
14      And el compas actual se resalta automaticamente
```

**Vistas involucradas:**

### **3.3 Tecnologías Utilizadas**

Listar y justificar las tecnologías seleccionadas.

## **4 Metodología Aplicada**

Describir la metodología empleada para el desarrollo del trabajo.

### **4.1 Marco Metodológico**

Explicar el marco metodológico utilizado (ágil, cascada, etc.).

### **4.2 Proceso de Desarrollo**

Detallar las etapas del proceso de desarrollo:

1. Análisis de requisitos
2. Diseño de la solución
3. Implementación
4. Pruebas
5. Despliegue

### **4.3 Herramientas de Desarrollo**

- Herramienta 1
- Herramienta 2
- Herramienta 3

### **4.4 Gestión del Proyecto**

Describir cómo se gestionó el proyecto (sprints, reuniones, etc.).

## 5 Experimentación y Validación

Presentar los experimentos realizados y la validación de la solución implementada.

### 5.1 Diseño de Experimentos

Describir el diseño de los experimentos realizados para validar la solución.

### 5.2 Casos de Prueba

Detallar los casos de prueba ejecutados.

### 5.3 Resultados Obtenidos

Presentar los resultados obtenidos del trabajo realizado.

Caso de Prueba	Resultado Esperado	Resultado Obtenido
Prueba 1	Éxito	Éxito
Prueba 2	Éxito	Éxito

Cuadro 2: Resultados de las pruebas realizadas

### 5.4 Análisis de Resultados

Analizar e interpretar los resultados obtenidos.

### 5.5 Validación con Usuarios

Describir el proceso de validación con usuarios finales.

## 6 Cronogramas

Presentar el cronograma de actividades del proyecto.

### 6.1 Cronograma Planificado

Descripción del cronograma inicial planificado para el proyecto.

Actividad	Duración	Período
Análisis de requisitos	2 semanas	Enero
Diseño	3 semanas	Febrero
Implementación	8 semanas	Marzo-Abril
Pruebas	2 semanas	Mayo
Documentación	1 semana	Mayo

Cuadro 3: Cronograma planificado del proyecto

### 6.2 Cronograma Real

Descripción del cronograma real de ejecución del proyecto.

### 6.3 Desviaciones y Ajustes

Análisis de las desviaciones respecto al plan original y los ajustes realizados.



## 7 Riesgos y Lecciones Aprendidas

Análisis de los riesgos identificados durante el proyecto y las lecciones aprendidas.

### 7.1 Identificación de Riesgos

Listar y describir los riesgos identificados al inicio del proyecto.

Riesgo	Probabilidad	Impacto
Riesgo técnico 1	Alta	Alto
Riesgo de recursos	Media	Medio
Riesgo de tiempo	Baja	Alto

Cuadro 4: Matriz de riesgos del proyecto

### 7.2 Gestión de Riesgos

Describir cómo se gestionaron los riesgos durante el proyecto.

### 7.3 Problemas Encontrados

Detallar los problemas principales enfrentados durante el desarrollo.

### 7.4 Lecciones Aprendidas

Compartir las lecciones aprendidas durante la ejecución del proyecto:

- Lección 1
- Lección 2
- Lección 3

## **8 Impactos Sociales y Ambientales del Proyecto**

Análisis de los impactos sociales y ambientales del proyecto desarrollado.

### **8.1 Impacto Social**

Describir el impacto social esperado o generado por el proyecto:

- Beneficios para la comunidad
- Mejoras en la calidad de vida
- Accesibilidad y inclusión

### **8.2 Impacto Ambiental**

Analizar el impacto ambiental del proyecto:

- Consumo de recursos
- Huella de carbono
- Sostenibilidad de la solución

### **8.3 Responsabilidad Social y Ética**

Consideraciones éticas y de responsabilidad social del proyecto.

### **8.4 Medidas de Mitigación**

Describir las medidas implementadas para minimizar impactos negativos.

## **9 Desarrollos Futuros**

Describir posibles extensiones, mejoras y trabajos futuros relacionados con el proyecto.

### **9.1 Mejoras Propuestas**

Listar las mejoras que podrían implementarse en el futuro:

- Mejora 1
- Mejora 2
- Mejora 3

### **9.2 Funcionalidades Adicionales**

Describir funcionalidades adicionales que podrían agregarse.

### **9.3 Escalabilidad**

Analizar cómo el sistema podría escalarse para soportar mayor carga o alcance.

### **9.4 Investigación Futura**

Proponer líneas de investigación futuras relacionadas con el proyecto.

### **9.5 Conclusiones Finales**

Presentar las conclusiones finales del trabajo, resumiendo los principales hallazgos y logros alcanzados.

## Referencias

### Referencias

- [1] A. S. Tanenbaum and M. Van Steen, *Distributed Systems: Principles and Paradigms*, 3rd ed. Boston, MA: Pearson, 2017.
- [2] P. Kruchten, “The 4+1 view model of architecture,” *IEEE Software*, vol. 12, no. 6, pp. 42-50, Nov. 1995.
- [3] “Reqnroll: Open-source Cucumber-style BDD test automation framework for .NET,” Reqnroll. [Online]. Available: <https://reqnroll.net/>. [Accessed: Jan. 7, 2026].
- [4] “Moises AI Studio: Lyric Writer,” Moises AI Studio. [Online]. Available: <https://studio.moises.ai/lyric-writer/>. [Accessed: Jan. 7, 2026].