Clefchamp

Plataforma de aprendizaje musical mediante gamificación

Music learning platform through gamification

Trabajo Fin de Grado
Curso 2024-2025

Autor
Jesús González Carrillo

DIRECTOR
Ramón González del Campo Rodríguez Barbero

Calificación: 8,5

Grado en Ingeniería Del Software
Facultad de Informática
Universidad Complutense de Madrid

Plataforma de aprendizaje musical mediante gamificación

Music learning platform through gamification

Trabajo de Fin de Grado en Ingeniería Del Software

Autor
Jesús González Carrillo

DIRECTOR
Ramón González del Campo Rodríguez Barbero

Calificación: 8,5

Convocatoria: Septiembre 2025

Grado en Ingeniería Del Software
Facultad de Informática
Universidad Complutense de Madrid

1 de Septiembre de 2025

DEDICATORIA

A Martín, mi gato, que no ha hecho mucho por este proyecto pero aún así le quiero.

AGRADECIMIENTOS

A mis amigos y compañeros por su ayuda durante el desarrollo de esta aplicación. A los voluntarios que participaron en las pruebas y me ofrecieron feedback constructivo.

RESUMEN

Este trabajo presenta "Clefchamp", una plataforma web de aprendizaje musical mediante gamificación. En sus inicios nació de una necesidad personal, aprender piano de forma autodidacta y encontrar dificultad en la lectura de notas. La aplicación busca innovar en la enseñanza del solfeo mediante mecánicas propias de los videojuegos. Desarrollada con Node y Express, ofrece un juego principal donde el usuario identifica notas musicales con el teclado, obteniendo puntos según su precisión y velocidad. Incluye autenticación, historial de partidas, estadísticas, y más funcionalidades que complementan la experiencia. Aunque no compite con grandes juegos de ritmo, tiene un gran potencial como herramienta educativa en colegios y academias. El proyecto ha sido desplegado en un servidor propio con Ubuntu Server y configuración de CI/CD mediante webhook desde GitHub.

Palabras clave

Solfeo, gamificación, educación musical, Node.js, aprendizaje, interfaz, estadísticas, juego.

ABSTRACT

This project introduces "ClefChamp", a web platform for musical learning through gamification. Born from a personal need while learning the piano, the platform addresses the difficulty of reading sheet music. ClefChamp uses game mechanics to enhance music theory learning. Built with Node and Express, it includes a main game where users identify musical notes using keyboard keys, scoring based on accuracy and speed. Features include authentication, match history, statistics and more features that complement the experience. Though it's not pretended to compete with commercial rhythm games, it has great potential as an educational tool in schools and music academies. The app is deployed on a self-hosted Ubuntu Server using a CI/CD pipeline triggered via GitHub webhook.

Keywords

Gamification, music education, Node.js, learning, UI, stats, self-learning, game.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo 1 - Introducción

El aprendizaje musical, y más concretamente el solfeo, es una de las barreras más comunes para la gente a la hora de comenzar con un instrumento musical como el piano. Muchos estudiantes autodidactas que tratan de aprender solfeo con el objetivo de tocar algún instrumento musical, pierden la motivación por el tiempo que tardan en interpretar partituras con sus respectivos instrumentos. Como resultado, se pierde la motivación en el proceso de aprendizaje pudiendo llegar al abandono, no por falta de destreza en el instrumento, sino por la frustración inicial que conlleva aprender solfeo.

En paralelo, el uso de la gamificación ha demostrado ser una herramienta eficaz en el contexto educativo. Plataformas como Duolingo o Chess han logrado integrar mecánicas de videojuegos para facilitar y motivar el aprendizaje, haciendo que este sea más accesible, visual y lúdico para el usuario.

Este proyecto propone el desarrollo de una aplicación web denominada Clefchamp, centrado en el aprendizaje musical mediante dinámicas de juego. Clefchamp dispone de una interfaz intuitiva que permite al usuario practicar la lectura de notas, recibir retroalimentación inmediata, ganar puntos y seguir su progreso en el tiempo.

En las siguientes secciones se detallan la motivación que ha impulsado este trabajo, los objetivos marcados y el plan de desarrollo que se ha seguido. Posteriormente se presenta un análisis del estado del arte, las tecnologías utilizadas, el funcionamiento de la aplicación y, finalmente, una reflexión sobre los resultados obtenidos, así como las posibilidades de mejora y evolución futura.

1.1 Motivación

El origen de este proyecto nace de una necesidad personal: aprender piano de forma autodidacta. A pesar de tener conocimientos básicos de música, la lectura de partituras y su traducción a las teclas del piano suponía una barrera.

Para algunos instrumentos de cuerda existe un sistema de notación musical denominado tablatura. Este consiste en representar cada cuerda como una línea en sustitución a las cinco que forman un pentagrama y el número de traste que se ha de tocar en lugar de la propia nota. Bajo mi experiencia personal, este sistema permite a gente principiante poder leer notas musicales de forma más fluida y dedicar más tiempo a practicar con el propio instrumento.

Personalmente disfruto de los videojuegos y he podido comprobar que los métodos de gamificación han sido eficaces en mi aprendizaje. Esto me llevó a plantear una forma alternativa de enseñar solfeo, y con ello poder reducir las barreras que separan a los estudiantes de un instrumento a leer partituras. Clefchamp nace como una plataforma web donde desarrollar y alojar juegos educativos centrados en la música.

1.2 Objetivos

El objetivo principal del proyecto es desarrollar una aplicación web en la que aplicar los conocimientos que he ido adquiriendo durante la carrera, así como poder investigar y aplicar conocimientos nuevos relacionados. Además, como objetivo personal, se busca aprender sobre interfaz de usuarios, diseño de páginas web y poder desplegar el entorno en un dominio a fin de obtener un resultado lo más parecido a un entorno real.

Llevando esto a mayor desglose estos son los objetivos que se plantean para poder lograrlo:

- Crear una plataforma web funcional que permita aprender música de forma divertida.
- Familiarizarse con la herramienta de desarrollo Chrome DevTools, explorando sus funcionalidades orientadas al diseño web responsive y al análisis de rendimiento mediante Lighthouse, con el fin de optimizar la experiencia de usuario y la calidad del proyecto.
- Diseñar e implementar un juego de solfeo basado en pentagramas y reconocimiento de notas.
- Incluir funcionalidades como registro, perfil de usuario o estadísticas, que acompañen la experiencia de aprendizaje.
- Desarrollar un MVP (Producto Mínimo Viable) completamente funcional y preparar la plataforma como una demo, de forma que sirva de base para su futura ampliación con nuevos juegos y funcionalidades.
- Ampliar y perfeccionar las habilidades técnicas en JavaScript, CSS y el uso de animaciones, aplicando principios de diseño de interfaces centradas en la experiencia de usuario.
- Desplegar la aplicación en un entorno accesible públicamente a través de Internet.
- Implementar un sistema de integración continua que automatice y optimice el proceso de despliegue.
- Crear un juego principal que proporcione puntuaciones y estadísticas con las que poder interactuar con el resto de funcionalidades de la aplicación.

1.3 Plan de trabajo

Antes de definir un plan de proyecto, se llevó a cabo una prueba de concepto con el objetivo de identificar tecnologías clave y anticipar posibles dificultades técnicas. Esta fase incluyó la investigación y experimentación con herramientas para el renderizado de partituras (mediante bibliotecas como VexFlow), la generación de gráficas estadísticas, el cifrado de contraseñas y los distintos métodos de despliegue de aplicaciones web.

A partir de esta exploración, se construyó un pequeño prototipo que permitía probar funcionalidades básicas: una ruta que mostraba una nota musical aleatoria en pantalla, un campo de entrada para cifrar textos con funciones hash y una gráfica generada con datos de ejemplo. Esto permitió elegir las herramientas del proyecto y estimar el alcance realista del mismo.

Una vez superada esta fase, se procedió a configurar un servidor con Ubuntu Server e implementar un sistema de integración continua (CI) mediante webhook de GitHub, lo que facilitó enormemente las tareas de despliegue y pruebas posteriores.

El desarrollo de la aplicación siguió una metodología iterativa e incremental. Esta se define como una metodología que combina la entrega progresiva de partes funcionales del producto (incremental) con la mejora continua de esas partes y del producto en general a través de ciclos repetidos de diseño, desarrollo y prueba (iterativo). Para ello se creó una lista de funcionalidades por orden de prioridad. Se siguió un ciclo en el que cada interacción pasaba por el desarrollo de la funcionalidad, la comprobación de su correcta integración y la adición de nuevas funcionalidades a la lista.

Figura 1-1: Representación gráfica de la metodología iterativa e incremental

Este proceso se repetía mediante iteraciones cortas. Permitiendo detectar errores, mejorar la experiencia de usuario y adaptar el producto según el feedback recibido. Las pruebas se realizaron con usuarios reales desde las primeras etapas, lo que permitió incorporar sugerencias valiosas y asegurar la estabilidad y utilidad del sistema desde el principio.

Capítulo 2 - Introduction

One of the most common barriers people face when starting to learn a musical instrument, such as piano, is learning music theory. Many self-taught students lose motivation when trying to learn music theory with the aim of playing an instrument because of how long it takes them to interpret sheet music. Consequently, they become discouraged and may abandon their studies, not due to a lack of skill on the instrument, but because of the initial frustration that accompanies learning music theory.

Meanwhile, gamification has proven to be an effective educational tool. Platforms such as Duolingo and Chess integrate video game mechanics to facilitate and motivate learning, making it more accessible, visual and fun for users.

This project proposes developing a web application called Clefchamp that focuses on learning music through game dynamics. With an intuitive interface, Clefchamp allows users to practice reading notes, receive immediate feedback, earn points and track their progress over time.

The following sections detail the motivation behind this work, the set objectives and the development plan. Next is an analysis of the state of the art, the technologies used and how the application works. Finally, there is a reflection on the results obtained and the possibilities for improvement and future development.

2.1 Motivation

This project stems from a personal need: to teach myself to play the piano. Despite my basic knowledge of music, I struggled with reading sheet music and translating it to the piano keys.

Some string instruments use a musical notation system called tablature. In tablature, each string is represented by a line instead of the five lines that form a staff and the frets to be played are indicated by numbers instead of notes. In my experience, this system enables beginners to read music more fluently and spend more time practicing with the instrument.

As someone who enjoys video games, I have found that gamification methods have been effective in my learning. This led me to consider an alternative way of teaching music theory that reduces the barriers separating students from instruments and sheet music. I created Clefchamp, a web platform for developing and hosting educational games focused on music.

2.2 Project goals

The main objective of the project is to develop a web application that allows me to apply the knowledge I acquired during my studies and research and apply new, related knowledge. Additionally, as a personal goal, is to learn about user interfaces, web page design and deploying the environment in a domain to obtain a result as close as possible to a real environment.

To achieve this, the following objectives were set:

- Create a functional web platform that allows users to learn music in a fun way.
- Get familiar with the Chrome DevTools development tool and its features geared toward responsive web design and performance analysis using Lighthouse. This will help you optimize the user experience and quality of the project.
- Design and implement a music theory game based on staves and note recognition.

- Include features such as registration, user profiles and statistics to enhance the learning experience.
- Develop a fully functional minimum viable product (MVP) and prepare the platform as a demo so that it can serve as a basis for future expansion with new games and features.
- Expand and refine your technical skills in JavaScript, CSS and animation, applying user-experience-centered interface design principles.
- Deploy the application in an environment accessible to the public via the internet.
- Implement a continuous integration system that automates and optimizes the deployment process.
- Create a main game that provides scores and statistics that interact with the rest of the application's features.

2.3 Workplan

Before creating a project plan, a proof of concept was conducted to identify key technologies and anticipate potential technical challenges. This phase included researching and experimenting with tools for score rendering (using libraries such as VexFlow), generating statistical graphs, encrypting passwords and deploying web applications in different ways.

Based on this exploration, a small prototype to test basic functionalities was built: a path that displayed a random musical note on the screen, an input field for encrypting text with hash functions and a graph generated with sample data. This allowed us to select project tools and estimate the project's scope.

After completing this phase, a server with Ubuntu Server and implemented a continuous integration (CI) system with GitHub webhooks was configured. This greatly facilitated subsequent deployment and testing tasks.

The application was developed using an iterative and incremental methodology. This approach combines the delivery of functional parts of the product in stages (incremental) with the continuous improvement of those parts and the product in general through repeated cycles of design, development and testing (iterative). To this end, it was created a list of features in order of priority. Each interaction went through the development of the feature, verification of its correct integration and the addition of new features to the list.

Figure 2-1: Graphic representation of the iterative and incremental methodology

This process was repeated in short iterations. Allowing to identify errors, enhance the user experience and refine the product based on feedback. It was tested with real users from the beginning, which enabled us to incorporate valuable suggestions and ensure the system's stability and usefulness from the outset.

Capítulo 3 - Estado de la cuestión

El aprendizaje musical mediante herramientas digitales no es un terreno nuevo. Existen múltiples plataformas que abordan esta problemática desde distintas perspectivas como el entretenimiento o el ámbito académico.

Una de las fuentes de inspiración principales ha sido Duolingo, cuya estructura gamificada para la enseñanza de idiomas ha demostrado ser efectiva para mantener la motivación del usuario. Clefchamp busca trasladar este enfoque al contexto del solfeo y la lectura musical, implementando puntos, estadísticas y un sistema de progresión para motivar la práctica continua.

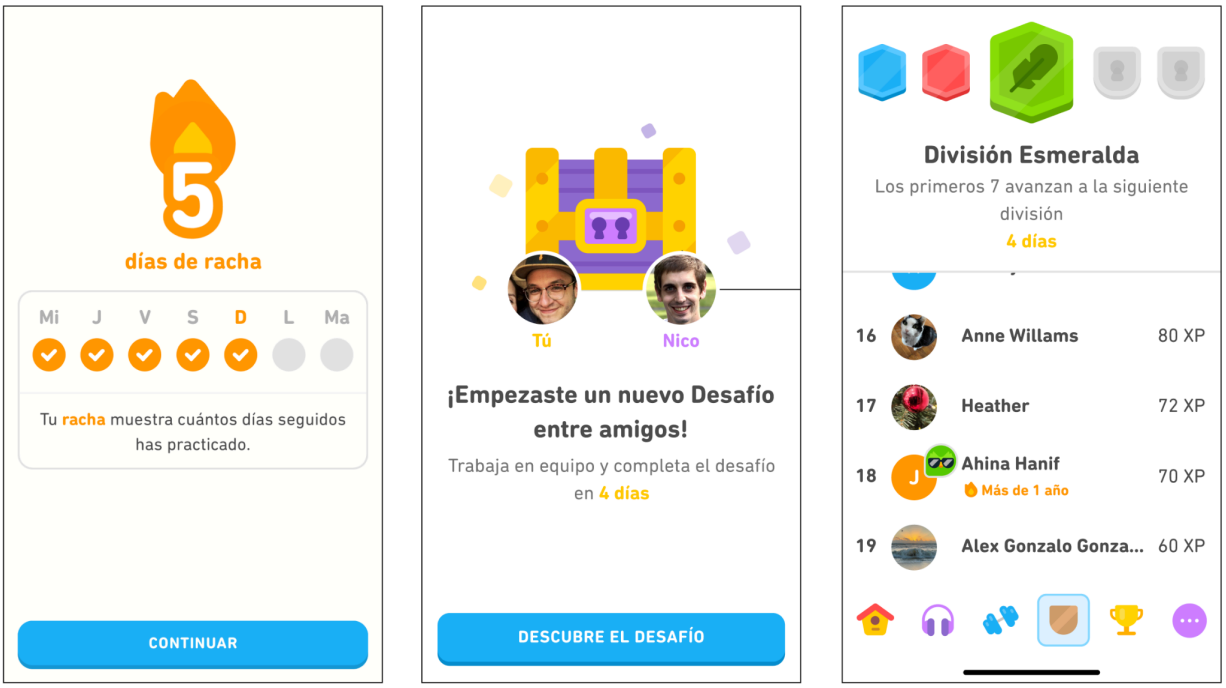

Figura 3-1: Ejemplo de gamificación en Duolingo

También se ha tomado referencia de plataformas como Chess.com, especialmente en el diseño de la zona de usuario y el seguimiento del progreso a lo largo del tiempo. Esta página combina componentes lúdicos con análisis estadísticos,

algo que Clefchamp busca replicar con gráficos de rendimiento diario en cada nivel de dificultad.

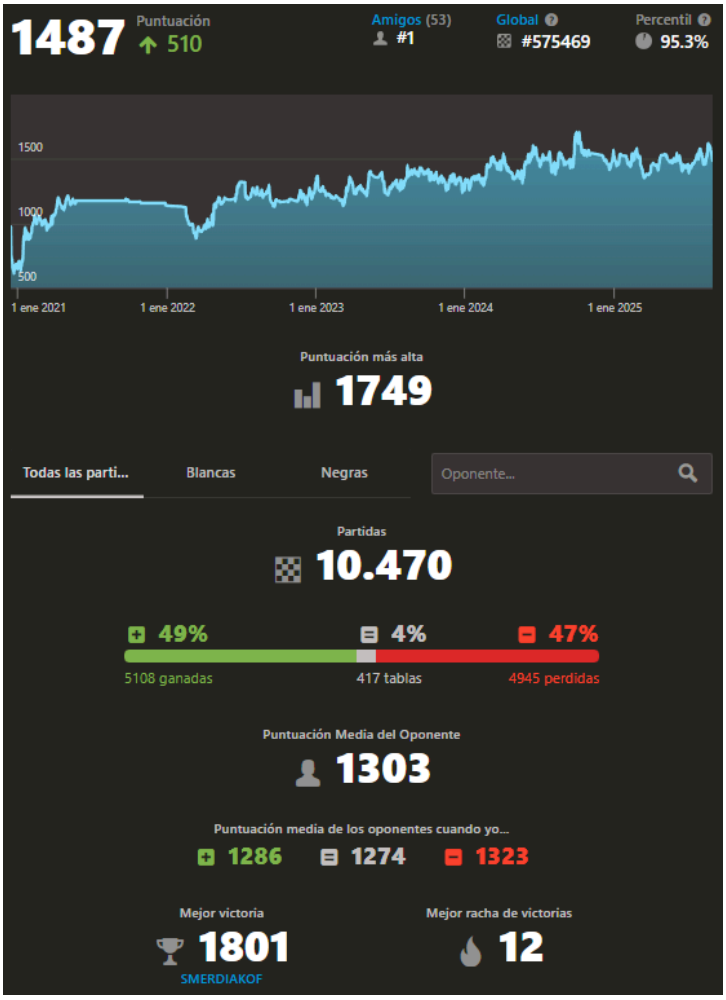

Figura 3-2: Pantalla de estadísticas de Chess.com

En cuanto a aplicaciones estrictamente musicales, existen propuestas como Simply Piano, Yousician o Perfect Ear, que utilizan sonido real y entrada MIDI. Sin embargo, muchas de estas herramientas están cerradas, requieren suscripción y priorizan el entendimiento y manejo de un instrumento sobre el aprendizaje teórico. Clefchamp se distingue por ser una alternativa gratuita y centrada en la lectura de notas, especialmente útil para principiantes, autodidactas o contextos escolares.

A diferencia de los grandes videojuegos de ritmo como Guitar Hero, Clefchamp no busca la espectacularidad ni el desafío audiovisual, sino la mejora progresiva de una habilidad musical concreta. Tampoco pretende competir con herramientas profesionales, sino facilitar una primera toma de contacto con el lenguaje musical en un entorno accesible y motivador.

Capítulo 4 - Fundamentos teóricos y tecnológicos

En este capítulo se presentan los fundamentos teóricos y tecnológicos que han servido de base para el desarrollo de la aplicación. Por un lado, se describen las principales herramientas, librerías y tecnologías empleadas en el proyecto, explicando qué son y cuál es su función. Este recorrido permitirá, en los apartados posteriores, hacer referencia a ellas de forma clara y comprensible.

Por otro lado, se exponen los principios de diseño y desarrollo que han guiado la construcción de la aplicación. Estos incluyen tanto las bases utilizadas para tomar las decisiones relacionadas con la arquitectura y organización del código, además de las buenas prácticas de programación que se han seguido a fin de garantizar claridad, mantenibilidad y eficiencia en la implementación.

4.1 Tecnologías utilizadas

La plataforma está desarrollada con Node.js y el framework Express en el backend. El motor de vistas utilizado es EJS, complementado con jQuery para la lógica en el lado del cliente y la parte visual está hecha mediante CSS y Bootstrap. La representación del pentagrama y las notas musicales se realiza con la biblioteca VexFlow, mientras que las gráficas de estadísticas se construyen con Highcharts.

La base de datos utilizada es MariaDB, alojada en un servidor propio con Ubuntu Server, al que se accede mediante un sistema de autenticación y sesiones almacenadas en la base de datos.

Para el despliegue, se ha configurado un servidor con Nginx como proxy inverso, y un sistema de CI/CD automático mediante un webhook de GitHub que lanza un

script en el servidor tras cada push a la rama principal. Este script actualiza dependencias y reinicia la aplicación usando PM2.

4.1.1 Bcrypt

Bcrypt[1] es una librería para Javascript utilizada para el cifrado de contraseñas. A diferencia de otros métodos de cifrado más simples, Bcrypt incorpora un mecanismo llamado salting, que añade información aleatoria a cada contraseña antes de cifrarla. Esto evita que dos contraseñas idénticas generen el mismo hash como se puede apreciar en las imágenes. Además, el algoritmo de Bcrypt está diseñado para ser deliberadamente costoso computacionalmente, lo que dificulta los intentos de fuerza bruta.

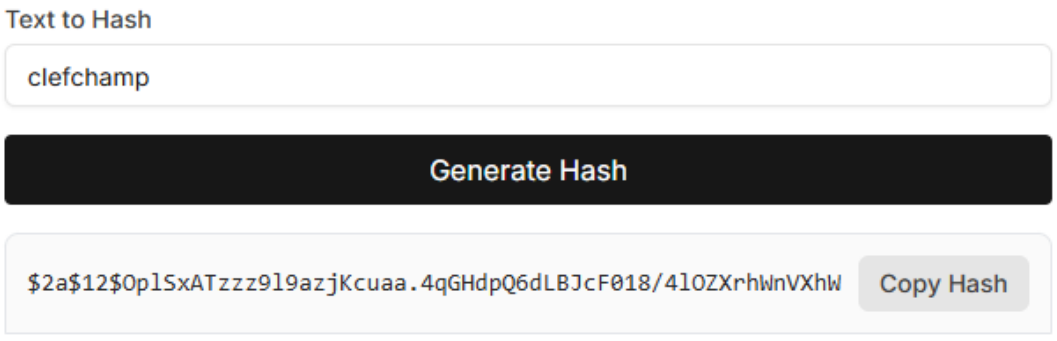

Figura 4-1: Primer hash generado

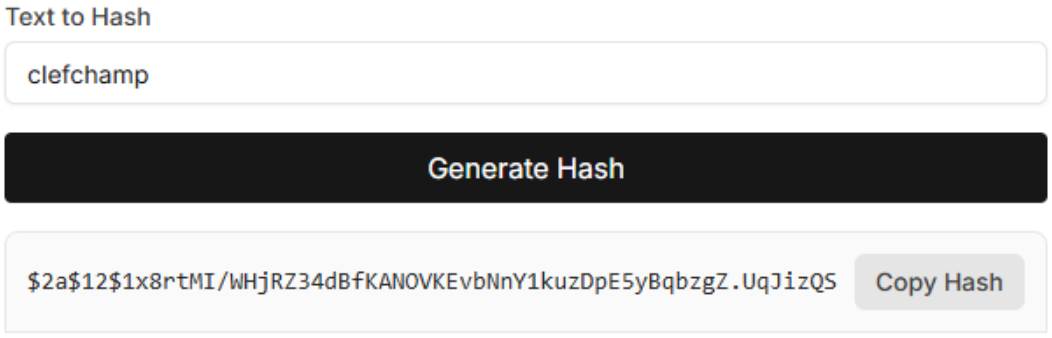

Figura 4–2: Segundo hash generado

4.1.2 Highcharts

Highcharts[2] es una librería de JavaScript especializada en la creación de gráficos interactivos para la web. Permite representar datos en distintos formatos como gráficos de líneas, barras, áreas, dispersión o circulares, con un alto nivel de personalización y compatibilidad entre navegadores. Una de sus principales ventajas es que ofrece interactividad integrada, como el resaltado de puntos de datos, la visualización de información al pasar el cursor o el filtrado de series.

4.1.3 Vexflow

VexFlow[3] es una librería de JavaScript enfocada en el renderizado de notación musical en el navegador, utilizando tecnologías como el Canvas de HTML5 y SVG. Está diseñada para representar partituras, pentagramas, notas y otros elementos musicales con gran precisión tipográfica. A diferencia de librerías gráficas genéricas, VexFlow incluye una API específica para manipular elementos musicales, lo que facilita tareas como mostrar notas generadas dinámicamente o representar ejercicios en tiempo real.

4.1.4 Webhooks de GitHub

Los webhooks de GitHub son un mecanismo que permite que un repositorio envíe notificaciones HTTP a una URL externa cuando ocurre un evento determinado, como un *push* de código. Esto facilita la automatización de procesos, ya que el servidor que recibe la notificación puede ejecutar scripts o actualizar aplicaciones sin intervención manual.

4.1.5 Node

Node[4] es un entorno de ejecución de JavaScript que funciona del lado del servidor, basado en el motor V8 de Google Chrome. Permite construir aplicaciones de alto rendimiento gracias a su arquitectura asíncrona y orientada a eventos. Es especialmente adecuado para aplicaciones que requieren manejar múltiples conexiones simultáneas sin bloquear el flujo de ejecución.

4.1.6 Express

Express[5] es un *framework* minimalista para Node que simplifica la creación de aplicaciones web y APIs. Proporciona un conjunto de herramientas para manejar rutas, peticiones y respuestas HTTP de forma eficiente, manteniendo al mismo tiempo la flexibilidad de Node. Express permite definir endpoints, aplicar *middlewares* para procesar datos o controlar el acceso, y estructurar el código del servidor de forma ordenada.

4.1.7 DevTools

Las *Developer Tools* (DevTools) son un conjunto de herramientas integradas en los navegadores modernos que permiten a desarrolladores inspeccionar, depurar y optimizar aplicaciones web directamente desde el navegador. Entre sus funcionalidades destacan la inspección del DOM y los estilos CSS, el análisis de rendimiento, la monitorización de peticiones de red, la depuración de código JavaScript y la simulación de dispositivos móviles.

4.1.8 MariaDB

MariaDB[6] es un sistema de gestión de bases de datos relacional (RDBMS) derivado de MySQL, totalmente compatible con este y de código abierto. Se caracteriza por su alto rendimiento, escalabilidad y estabilidad, siendo ampliamente utilizado en entornos de producción. Utiliza SQL como lenguaje de consulta y ofrece soporte para transacciones, integridad referencial y múltiples motores de almacenamiento.

4.1.9 Bootstrap

Bootstrap[7] es un framework de código abierto orientado al desarrollo web responsive y mobile-first. Proporciona un conjunto de herramientas basadas en HTML, CSS y JavaScript. Incluye componentes predefinidos como botones, formularios, menús de navegación y más componentes que permiten estructurar de manera flexible y escalable el diseño de una aplicación web.

4.2 Fundamentos de diseño y desarrollo

En este proyecto se utilizan fundamentos, principios y patrones propios de la Ingeniería del Software, además de fundamentos de desarrollo de aplicaciones web.

El diseño de la aplicación trata de ser accesible y usable mediante una interfaz sencilla y prácticas de accesibilidad que se detallarán más adelante al igual que el diseño responsivo, que es otra de las bases del diseño web. El diseño busca ser consistente, tratando de crear una identidad visual: se mantiene la misma paleta de colores en la aplicación, todos los iconos siguen un estilo monócromo y simple y el formato de las páginas sigue un mismo patrón. Se intenta mantener una jerarquía visual a lo largo de las distintas pantallas, usando cabeceras más grandes para guiar la

atención del usuario y separando los apartados en secciones bien diferenciadas para ayudar al entendimiento. Todo el diseño busca facilitar y mejorar la experiencia del usuario a través de una navegación fluida y tratando de reducir fricciones para el usuario a la hora de usar la aplicación.

Por la parte de desarrollo, el código está separado de forma modular buscando la reutilización de código y el bajo acoplamiento. En la aplicación se integran protocolos de seguridad como la validación de los formularios a fin de evitar la inyección de código SQL o el cifrado de contraseñas. Las consultas y el diseño de la base de datos buscan ser eficientes a fin de maximizar el rendimiento de la aplicación. Se logra compatibilidad entre dispositivos permitiendo los eventos táctiles y el funcionamiento en distintos navegadores.

A la hora de programar el código se intentan aplicar criterios como DRY (Don't repeat yourself) o evitar el uso de magic numbers mediante constantes. Además, se procura seguir principios como KISS (mantener las soluciones lo más simples posible) o el principio de responsabilidad única de cada módulo y la modularidad del código, con el objetivo de mejorar la legibilidad, facilitar la reutilización y reducir la complejidad del mantenimiento. Del mismo modo, se fomenta el uso de nombres significativos, lo que contribuye a la calidad del software.

Capítulo 5 - Diseño

En el ámbito del software, el diseño es una parte fundamental, y así sucede en Clefchamp, ya que establece las bases sobre las cuales se construye la aplicación tanto a nivel estructural como visual. En esta sección se definen las decisiones de diseño que guían la arquitectura del sistema, la organización de la base de datos, la interacción entre los distintos componentes y la apariencia de la interfaz de usuario.

El objetivo principal es garantizar que la aplicación no solo cumpla con los requisitos funcionales planteados, sino que también ofrezca una experiencia de uso clara e intuitiva. Para ello, se combinan las tecnologías y bases establecidas anteriormente.

5.1 Diseño del juego

El juego está pensado para ser breve y competitivo, de tal forma que evite la tasa de abandono por tener rondas muy largas y que el usuario pueda jugar múltiples partidas aunque no disponga de mucho tiempo. Para integrar al jugador en esta dinámica, el feedback está acompañado de animaciones rápidas que te incitan a seguir intentándolo con mejores marcas de tiempo.

Al comenzar la partida, el usuario visualiza dos pentagramas (clave de sol y fa), una barra de progreso, un marcador de puntuación y un teclado interactivo con letras y teclas asociadas. Al comenzar el juego, aparece una nota aleatoria. El usuario debe pulsar la tecla (A, S, D, F, J, K, L) que corresponda con la nota que aparece en el pentagrama.

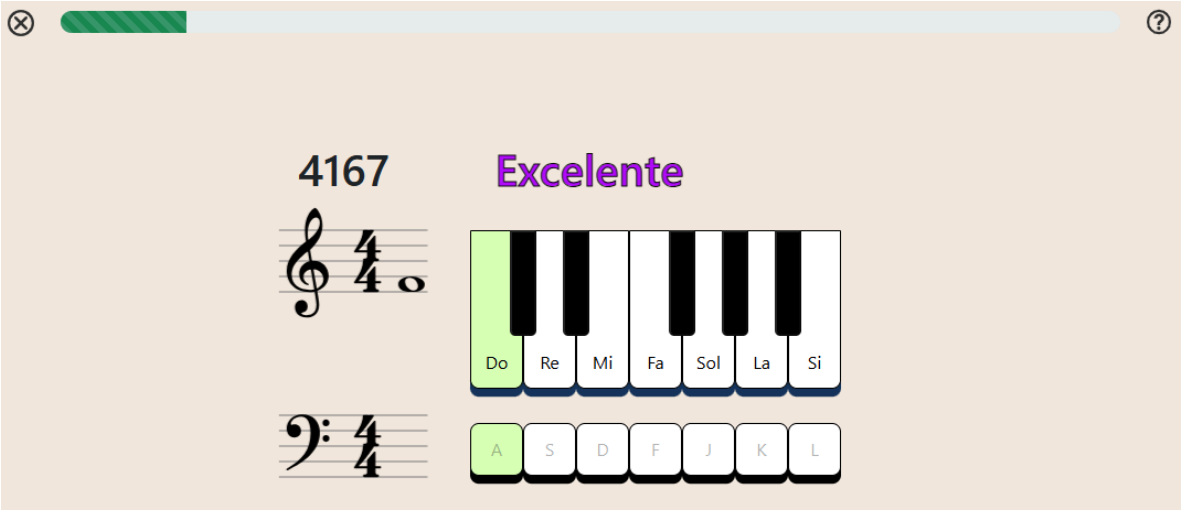

Figura 5-1: Estructura de la sección de juego de Clefchamp

Cada nivel de rendimiento (de más a menos: Perfecto, Excelente, Genial, Bien, Ok) tiene un umbral de tiempo y puntuación base. La fórmula ajusta puntos según la rapidez con que se responde. Esta se calcula como la puntuación base del umbral correspondiente, más el producto de los puntos extra por la diferencia entre los milisegundos que ha obtenido y el límite inferior del umbral correspondiente.

Resultado	Tiempo	Puntos base	Puntos extra
Perfecto	Menos de 1 segundo	5000	2
Excelente	Entre 1 y 2 segundos	4000	1
Genial	Entre 2 y 4 segundos	2500	0,75
Bien	Entre 4 y 8 segundos	1500	0,25
Ok	Más de 8 segundos	1000	0

Tabla 5-1: Rangos de tiempo y puntuaciones obtenidas en el juego principal

Hay tres niveles de dificultad, que modifican la cantidad de rondas y la probabilidad de que las notas aparezcan en la clave de sol o en la clave de fa. Existe además un modo TRIAL accesible sin registro. En el modo fácil solo aparecen notas en la clave de sol, en el normal hay un 25% de probabilidad de que aparezcan notas en la clave de fa y en el difícil hay la misma probabilidad de que aparezca en clave de sol o en clave de fa.

5.1.1 Diseño del piano

Para el diseño del piano se partió de la base que ofrecía el tutorial de Initgrammers[8], sobre el que se añadieron animaciones de pulsado para conseguir un resultado más auténtico. Además, debido a una decisión de diseño que se detalla más adelante, se quitó el sonido del pulsado de las teclas.

Una vez hecho el piano faltaba elegir qué teclas del ordenador estarían vinculadas a las teclas del piano virtual. La idea inicial fue poner teclas de control lo más parecidas posibles a un piano, por ello se eligieron las letras A, S, D, F, G, H, J, por ser la fila central. Esto favorecía una posición inicial de las manos más acorde a la posición inicial en mecanografía, favoreciendo así una postura más cómoda. La importancia de la colocación de las manos reside en que el usuario no pierda tiempo en saber dónde está cada tecla y se pueda centrar solo en el reconocimiento de las notas. Tras oír el consejo de compañeros que han probado el juego, me sugirieron ampliar la separación de las teclas ya que en algunos portátiles de menor tamaño resultaba muy incómodo tenerlas tan juntas. Tras probar varias configuraciones y fijarme en algunos juegos de ritmos se decidió usar A, S, D, F para la mano izquierda y J, K, L para la derecha de tal forma que hubiese más espacio entre ambas manos y fuese acorde a la posición inicial en mecanografía.

5.1.2 Teclado sin sonido

En un inicio, el diseño del juego contemplaba el uso de sonido, con el objetivo de que el usuario pudiera familiarizarse progresivamente con las notas musicales a través de la escucha. El prototipo inicial del piano virtual ya incluía sonidos en su código, lo que reforzó la idea de mantener este enfoque.

Sin embargo, surgió una dificultad al intentar incorporar más de una escala dentro de los juegos. El problema radicaba en que una misma tecla, tanto del teclado del ordenador como del piano virtual, podía corresponder a notas diferentes según la escala seleccionada, lo cual podría resultar confuso para el jugador. Limitar los juegos a una única escala tampoco era una opción viable, ya que reduciría considerablemente el nivel de dificultad y, en consecuencia, el interés para usuarios con mayor experiencia.

Ante esta situación, se optó por dividir la propuesta en dos juegos independientes: uno basado únicamente en el apoyo visual y otro centrado exclusivamente en el apoyo sonoro. Esta decisión no solo buscó evitar la ventaja competitiva de quienes pudieran beneficiarse de ambos estímulos (visual y auditivo), sino también ofrecer experiencias de aprendizaje diferenciadas y adaptadas a las capacidades de cada jugador. Del mismo modo, permitió plantear una alternativa inclusiva para aquellas personas que no pueden hacer uso de la visión, de manera que pudieran disfrutar del juego únicamente a través del reconocimiento auditivo.

Finalmente, la versión sonora se restringió a una sola escala, dado que el reconocimiento auditivo presenta una complejidad mucho mayor que el visual, proporcionando así un reto suficiente incluso para perfiles más avanzados.

5.2 Diseño de la aplicación

La aplicación sigue un enfoque visual jerárquico y minimalista, buscando claridad y facilidad de uso. La mayor parte de las pantallas se organiza con un header que ocupa aproximadamente el 10% superior, mientras que el body ocupa el 90% restante, dejando el footer accesible mediante scroll. Esta estructura asegura que la navegación y los elementos esenciales estén siempre disponibles sin sobrecargar la interfaz.

El estilo se inspira en el concepto de "bento boxes", muy marcado en la página de inicio, donde los contenidos se presentan en bloques claros y equilibrados. El resto de la aplicación mantiene un diseño minimalista, destacando únicamente los elementos funcionales y empleando animaciones simples que refuerzan la interactividad sin distraer al usuario. A fin de contribuir a una experiencia visual uniforme y agradable el diseño busca la coherencia de colores, tipografía y tamaños. Esto, al mismo tiempo, permite al usuario orientarse y comprender la información de manera inmediata.

Figura 5-2: Diseño de la página de inicio

5.2.1 Diseño en Figma

Antes de implementar la parte visual, se elaboró un diseño preliminar en Figma para definir la estructura general de las pantallas y la estética de la aplicación. Sin embargo, dado que el desarrollo se llevó a cabo en ciclos iterativos, el diseño no quedó cerrado desde el inicio: se fueron realizando ajustes y nuevas propuestas gráficas en paralelo a la programación, lo que permitió mejorar la usabilidad y adaptar la interfaz a las necesidades que iban surgiendo.

Cabe señalar que no todo el diseño en Figma llegó a completarse, precisamente con la intención de dejar espacio para futuras iteraciones. En la figura 5-3 puede apreciarse un ejemplo de diseño que quedó en estado de borrador.

Figura 5-3: Diseño planteado en Figma de los apartados del historial

Del mismo modo los diseños iniciales sufrían modificaciones, ya sea por rechazar ideas que no tenían peso para los objetivos del proyecto o para adaptar detalles visuales. Aquí se muestran algunos ejemplos del diseño inicial que se hizo en Figma, y como quedó en desarrollo.

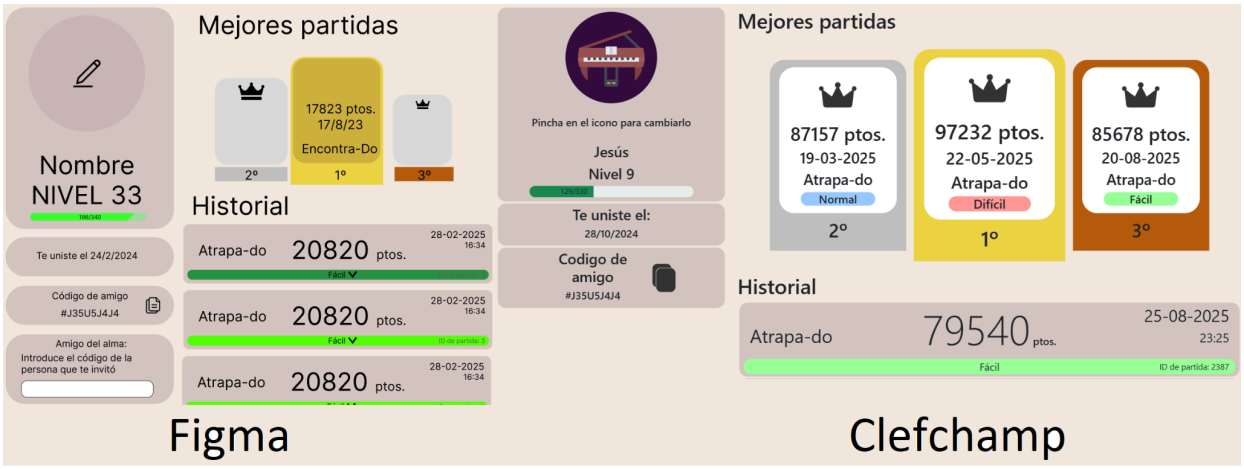

Figura 5-4: Comparación entre el diseño inicial y el resultado del perfil

Figura 5-5: Comparación entre el diseño inicial y el resultado del selector de juego

5.2.2 Diseño prerregistro

En la aplicación, toda la parte previa al inicio de sesión está diseñada con el objetivo de conducir al usuario hacia la creación de una cuenta. Para ello, se le ofrece primero una experiencia introductoria sencilla que no requiere registro, de manera que pueda familiarizarse rápidamente con la dinámica de la plataforma y percibir su utilidad. Esta estrategia busca reducir la fricción inicial y generar interés antes de solicitar datos de registro.

Dentro de esta experiencia preliminar, el usuario puede acceder a un juego de prueba. Al finalizar la partida, el sistema le muestra el porcentaje de aciertos, lo que funciona como refuerzo inmediato y como una primera métrica de rendimiento. Acto seguido, se le redirige a la pantalla de creación de cuenta, incentivando así el paso natural de convertirse en usuario registrado. La idea es que, tras haber probado la aplicación y recibido un feedback, el jugador tenga una mayor motivación para continuar y seguir practicando.

En definitiva, tanto la navegación inicial como el flujo del juego están orientados a lograr una transición progresiva: primero despertar la curiosidad, luego ofrecer un adelanto del funcionamiento y, finalmente, guiar hacia el registro para acceder a todas las funcionalidades de la plataforma.

5.2.3 Diseño posregistro

En la parte posterior al inicio de sesión, la aplicación está diseñada para fomentar de manera constante la interacción con los juegos, convirtiéndolos en el eje central de la experiencia. Desde el propio menú inicial, la opción de jugar se sitúa como la funcionalidad principal y más accesible, asegurando que el usuario perciba rápidamente cuál es el corazón de la aplicación.

Además, el resto de funcionalidades se han concebido también con la intención de incitar a seguir jugando. El historial de partidas permite revisar el rendimiento anterior y compararlo con el progreso reciente, lo que genera una motivación por mejorar. Las estadísticas ofrecen una visión más detallada del desempeño, pudiendo materializarlo en una gráfica que te permite visualizar la sensación de avance y aprendizaje continuo.

El ranking general introduce un componente competitivo. Esto invita al usuario a comparar sus resultados con los de otros jugadores y despertando el deseo de escalar posiciones y de destacar entre otros jugadores. Finalmente, las partidas rápidas actúan como una vía inmediata para retomar la experiencia, reduciendo las barreras de entrada y motivando al usuario a intentar superar sus propios récords.

En conjunto, el diseño posregistro busca generar un ciclo de retroalimentación positiva, en el que cada funcionalidad, directa o indirectamente, incite al usuario a volver a jugar y a mantener una interacción frecuente con la plataforma.

5.2.4 Imágenes e iconos de la aplicación

En cuanto a los recursos gráficos de la aplicación, se ha realizado un esfuerzo por mantener una identidad visual propia y coherente. Para ello, la mayoría de los iconos utilizados, a excepción de los iconos de perfil, han sido diseñados personalmente mediante Figma, aprovechando las herramientas de fusión y extracción de formas que la propia aplicación proporciona. Este proceso permitió generar iconografía adaptada a las necesidades específicas de Clefchamp, en lugar de recurrir únicamente a bibliotecas externas, lo que contribuye a otorgarle un carácter distintivo a la aplicación.

Respecto a las imágenes ilustrativas empleadas, estas pertenecen a Inés Olimpia, una compañera de la universidad, mientras que los iconos de perfil fueron obtenidos a partir del repositorio abierto SVG Repo[9].

Figura 5-6: Muestra de iconos diseñados en Figma

5.2.5 Diseño de la Base de Datos

En el diseño de la base de datos se ha buscado garantizar la consistencia, escalabilidad y facilidad de mantenimiento de la aplicación. La estructura sigue un enfoque relacional, en el que se definen entidades principales como los usuarios, su progreso, las relaciones sociales dentro de la plataforma y la información relativa a los juegos. Cada tabla se ha diseñado teniendo en cuenta tanto la normalización de los datos como la eficiencia en las consultas más frecuentes de la aplicación.

El modelo refleja distintos aspectos de la experiencia de usuario: desde la gestión de la cuenta y su personalización (iconos, niveles, preferencias), hasta la interacción social (amistades y solicitudes) y el seguimiento del rendimiento en las actividades de aprendizaje musical. Además, se han introducido tablas de apoyo que permiten implementar mecánicas de gamificación, como el sistema de niveles y recompensas.

De esta forma, la base de datos no solo funciona como un repositorio de información, sino que también es la pieza central que sostiene la lógica de la aplicación, asegurando que todas las funcionalidades se integren de forma coherente.

En los siguientes subapartados se detallarán las tablas principales que constituyen el núcleo de la aplicación, pero además de estas, se han definido otras tablas auxiliares que permiten implementar funcionalidades de apoyo y enriquecer la experiencia de usuario. Las tablas auxiliares son:

- levelprogression: almacena la experiencia necesaria para alcanzar cada nivel, sirviendo de referencia para el cálculo de progresos y recompensas.
- userlevel: registra el nivel y la experiencia actual de cada usuario, enlazando con la progresión definida en la tabla anterior.
- usericons: gestiona los iconos desbloqueados por cada usuario, permitiendo personalizar su perfil con distintos iconos y colores de fondo.
- icons: contiene el catálogo de iconos disponibles en la aplicación, indicando cuáles están desbloqueados por defecto y cuáles requieren condiciones específicas.
- userpreferences: guardan parámetros relacionados con la personalización de la experiencia. Por el momento solo almacena si el usuario ha seleccionado la opción de no volver a mostrar el tutorial.

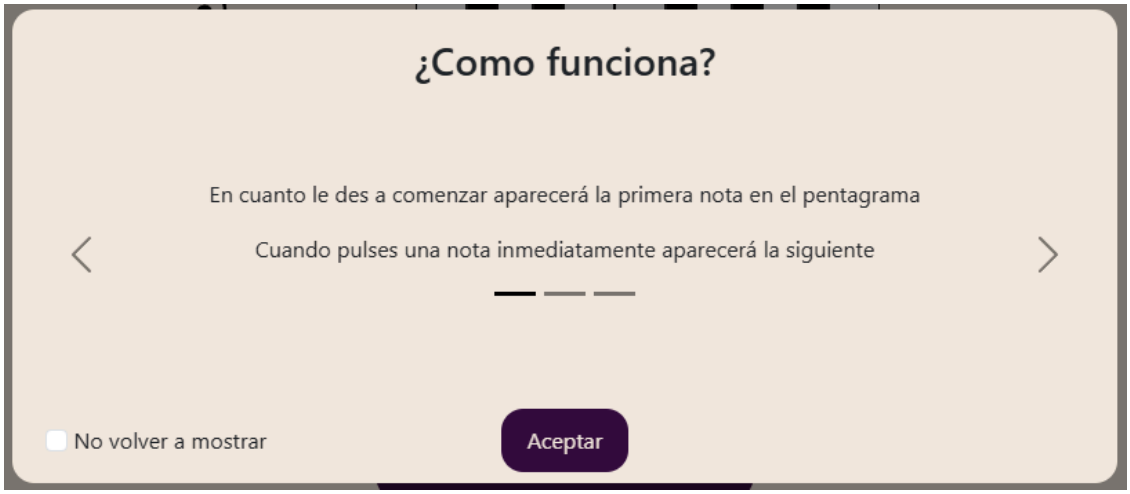

Figura 5-7: Modal del tutorial con la opción de "No volver a mostrar"

5.2.5.1 Tabla sessions

La tabla sessions almacena los datos de las sesiones activas de los usuarios. Cada sesión se identifica mediante un session_id único y un tiempo de expiración (expires). La columna data contiene información serializada del usuario y otros datos asociados a la sesión, como preferencias de visualización o estado temporal del juego.

Esta tabla permite mantener a los usuarios autenticados de forma segura y facilita la gestión de la experiencia mientras navegan por la plataforma, asegurando que su progreso y configuración se mantengan consistentes entre diferentes visitas.

5.2.5.2 Tabla amigos

La tabla amigos gestiona las relaciones sociales entre usuarios. Cada registro representa una solicitud de amistad, indicando quién la envía (userId) y quién la recibe (friendId), así como el estado de la relación (pendiente, aceptado). La columna created_at permite registrar cuándo se produjo la acción, útil para funciones de historial o notificaciones.

Este diseño facilita la implementación de funcionalidades sociales, como aceptar o rechazar solicitudes, mostrar la lista de amigos de un usuario o comparar puntuaciones entre amigos. Su estructura simple pero relacional permite consultar fácilmente tanto los amigos de un usuario como el estado de sus solicitudes.

5.2.5.3 Tabla userrecord

La tabla userrecord guarda el historial de cada juego realizado por los usuarios, incluyendo la fecha y hora (time), la dificultad, los resultados parciales (perfect, excellent, etc.) y los puntos obtenidos. También se registran detalles más finos, como las notas tocadas y los tiempos individuales de cada acción.

Esta información permite analizar el rendimiento del usuario, generar estadísticas de progreso, mostrar rankings y ofrecer feedback personalizado. Al relacionarse con la tabla users, cada registro queda asociado a un jugador concreto, lo que posibilita el seguimiento individual y la comparación con otros usuarios.

5.2.5.4 Tabla usuarios

La tabla usuarios almacena la información principal de cada usuario registrado en la plataforma. Incluye datos identificativos (nombre, correo electrónico, tagname, código de amigo), credenciales de acceso (contraseña cifrada) y estado de la cuenta (activa/inactiva). Además, se registra la fecha de creación de la cuenta, que permite llevar un seguimiento temporal de la actividad del usuario.

Esta tabla es la pieza central del modelo relacional, ya que se relaciona con prácticamente todas las demás: sus iconos (usericons), niveles (userlevel), sesiones activas (sessions), amigos (amigos) y registros de juegos (userrecord). De esta manera, todas las funcionalidades de la aplicación tienen un punto de referencia único para identificar al usuario.

5.3 Integraciones

En el desarrollo de la aplicación, las integraciones con librerías externas han sido un aspecto fundamental para incorporar funcionalidades avanzadas sin necesidad de implementarlas desde cero. Estas integraciones permiten aprovechar soluciones probadas y robustas en ámbitos clave como la seguridad, la visualización de datos y la representación musical. De esta forma, se logra no solo optimizar el tiempo de desarrollo, sino también garantizar una mayor calidad y fiabilidad en la experiencia del usuario.

5.3.1 Bcrypt

El cifrado de contraseñas se realiza mediante Bcrypt. A la hora de registrarse, a la contraseña introducida se le aplica una función de hash que se ejecuta un número determinado de rondas.

Estas rondas actúan como un nivel de dificultad: cuantas más rondas, más tiempo tarda en generarse el hash, lo que complica los ataques por fuerza bruta. Además, Bcrypt añade un valor aleatorio llamado salt, que garantiza que dos contraseñas iguales no generen el mismo hash. Esto significa que incluso si varios usuarios tienen la misma contraseña, los hashes que se almacenan en la base de datos serán distintos.

Una vez aplicada la función de hash con el salt incluido, el resultado se guarda en la base de datos en lugar de la contraseña original. De esta manera, aunque alguien accediera a la base de datos, no podría ver las contraseñas reales de los usuarios.

En el proceso de inicio de sesión, cuando el usuario introduce su contraseña, Bcrypt se encarga de compararla con el hash almacenado. Para ello, toma la

contraseña introducida, le aplica el mismo proceso de hashing usando el salt del hash original, y comprueba si el resultado coincide. Si coincide, significa que la contraseña es correcta.

Este sistema permite una autenticación segura sin necesidad de almacenar nunca la contraseña en texto plano, lo que mejora considerablemente la seguridad general de la aplicación.

5.3.2 Highcharts

La librería Highcharts se integra en la aplicación en el apartado de estadísticas, donde se utilizan tres gráficos, uno para cada dificultad, para representar el progreso de cada jugador. En concreto, se muestra mediante un gráfico de tipo areaspline la mejor puntuación obtenida cada día a lo largo del tiempo, lo que permite visualizar con claridad la evolución del rendimiento. Además, la implementación incluye la posibilidad de desplazar el cursor sobre los puntos del gráfico para consultar de manera exacta la puntuación de un día concreto, lo que aporta un nivel adicional de detalle en el seguimiento del aprendizaje. Esta integración ofrece una representación visual clara e intuitiva de los datos, facilitando al usuario interpretar su progreso y motivándolo a continuar con la práctica.

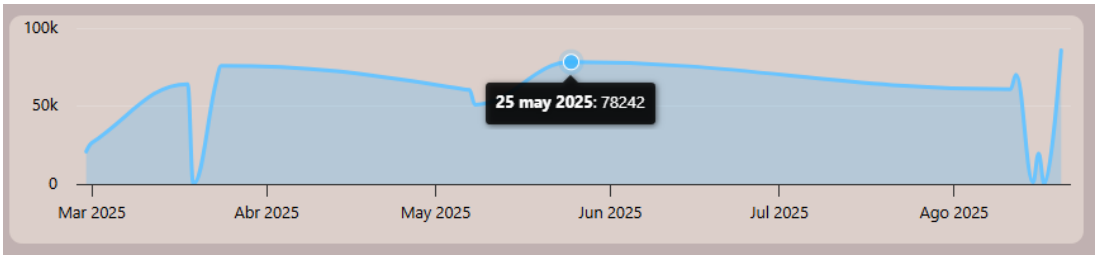

Figura 5-8: Gráfico mostrado en el apartado de estadísticas

5.3.3 VexFlow

En el proyecto, VexFlow se emplea en los distintos juegos para dibujar de forma dinámica las notas musicales sobre el pentagrama, utilizando tanto la clave de sol como la clave de fa según el ejercicio. Para gestionar esta integración de manera más ordenada, se desarrolló un gestor específico (manager) que centraliza la interacción con la librería, facilitando la creación, actualización y renderizado de los elementos gráficos en función de las acciones del usuario. Gracias a este enfoque, los juegos pueden mostrar en tiempo real nuevas notas generadas de forma aleatoria y adaptarse con flexibilidad a distintas dinámicas de práctica, lo que evita recurrir a recursos estáticos como imágenes y aporta una experiencia interactiva y escalable.

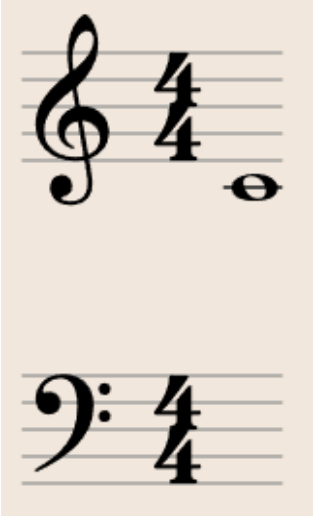

Figura 5-9: VexFlow renderizando una redonda

Capítulo 6 - Desarrollo de la aplicación

El desarrollo de ClefChamp es la fase en la que las decisiones de diseño y los principios de la ingeniería del software se convierten en una aplicación funcional. En este apartado se abordan las estrategias, tecnologías y prácticas de programación que permiten materializar la arquitectura planteada, garantizando que el sistema sea mantenible, escalable en la medida de lo necesario y seguro para los usuarios.

El propósito principal es transformar los requisitos en una solución implementada que respete los principios de calidad del software: modularidad, reutilización, bajo acoplamiento y legibilidad del código. Para lograrlo, se emplean metodologías y buenas prácticas de desarrollo como las ya mencionadas en el apartado de fundamentos. Esto facilita tanto la evolución futura de la aplicación como la corrección de posibles errores.

Asimismo, el desarrollo no solo busca cumplir con las funcionalidades básicas, sino también asegurar un rendimiento adecuado, compatibilidad entre dispositivos y navegadores, y la integración de mecanismos de seguridad que refuercen la confianza del usuario. Todo ello se consigue a través de una programación clara y organizada, que refleja los principios previamente definidos en el diseño.

6.1 Funcionalidades

La aplicación cuenta con una serie de funcionalidades que permiten al usuario interactuar con el sistema de forma sencilla, intuitiva y personalizada. Algunas de ellas son esenciales para garantizar el correcto funcionamiento de la plataforma, como el inicio de sesión, el registro o la visualización del perfil. Otras, como el sistema de amigos, el reproductor de música o la opción de borrar cuenta, complementan la

experiencia del usuario, pero no constituyen el núcleo principal de la aplicación, por lo que se mencionan brevemente en esta introducción.

El sistema de amigos permite a los usuarios conectarse entre sí y, en un futuro, podría habilitar funciones sociales adicionales como la comparación de puntuaciones o el reto entre usuarios. La opción de borrar cuenta garantiza el derecho del usuario a eliminar sus datos si así lo desea, mientras que el reproductor de música ofrece una herramienta básica para escuchar pistas dentro de la plataforma, aunque su uso no es central para el aprendizaje musical en esta primera versión.

A continuación, se describen en detalle las funcionalidades más relevantes de la aplicación:

6.1.1 Registro

En la página de registro se ha optado por un diseño sencillo, limitándose a solicitar los datos mínimos para comenzar a usar la aplicación. Esto es con el fin de eliminar barreras de entrada a nuevos usuarios. Los datos requeridos son:

- Nombre: ha de tener mínimo dos caracteres, siendo estos solo letras. Se usará este campo para referirse al usuario. Es el valor que aparecerá en secciones privadas del usuario como la vista del perfil.
- Alias: sin restricciones mínimas de caracteres, puede contener letras y números. La comprobación de si el alias que elija el usuario ya existe es inmediata a fin de que no tenga que reenviar el formulario. Este valor se usa en secciones públicas como el ranking, es el valor que se muestra a otros usuarios.
- Correo: en el estado actual del proyecto no se hace uso de este valor, sin embargo se almacena con el objeto de incorporarlo en desarrollos futuros.

- Contraseña: ha de contener más de 8 caracteres, un símbolo, una mayúscula y un número. Al no disponer de la funcionalidad de correo no se puede crear un método de recuperación de contraseña. Para evitar crear contraseñas no deseadas hay un campo de confirmación de contraseña y está desactivada la función de portapapeles en la página.

Cada usuario recibe un código de amigo autogenerado con una función aleatoria simulando la generación de hashes pero hecha a mano. Esta función genera valores aleatorios entre número y letras y los concatena hasta llegar a nueve caracteres. Además se almacena el momento del registro y se genera un identificador interno único para el usuario.

6.1.2 Login

El inicio de sesión permite entrar mediante el uso de correo electrónico o alias. A diferencia del registro no se realiza ninguna comprobación, sino que se comprueba en el envío del formulario para evitar ataques de fuerza bruta. Si el usuario es correcto se recuperan otros datos y se almacenan en la sesión para evitar accesos innecesarios a la BBDD, además de almacenarse en la tabla de sesiones. Esta tabla sirve para poder guardar la sesión del usuario si éste cierra temporalmente la pestaña del navegador.

6.1.3 Perfil

Cada usuario puede acceder a su perfil de distintas formas: pulsando sobre su icono, seleccionando la opción "Mi perfil" en la pantalla principal o utilizando el menú de acceso rápido situado en la parte superior derecha del header de la aplicación.

La vista del perfil está dividida en dos secciones. En la parte izquierda se muestra la información principal de la cuenta, como el nivel alcanzado, el icono personalizado,

la fecha de creación y el código de amigo. Este último incluye un botón que permite copiarlo directamente al portapapeles, lo que facilita compartirlo con otros usuarios.

La parte derecha, en cambio, está dedicada al historial de actividad. En primer lugar se presentan las tres mejores partidas del usuario, ordenadas por puntuación sin importar la dificultad. Justo debajo aparece un registro con las cinco partidas más recientes, acompañado de la opción de expandir la vista para consultar el historial completo.

Por último, dentro de esta misma vista, al seleccionar el icono de perfil se despliega un modal que permite modificar el icono de perfil.

6.1.4 Cambiar icono

El usuario tiene la posibilidad de modificar su icono de perfil a través de un modal. Desde esta interfaz se pueden elegir diferentes iconos disponibles, permitiendo así una mayor personalización de la cuenta. Además de poder elegir el color de fondo que podrá modificar mediante un selector de color propio del navegador.

Figura 6-1: Modal de elección de icono de perfil

Aunque en la versión actual únicamente se incluyen iconos básicos, está previsto que en futuras iteraciones aparezcan también iconos especiales que se desbloquearán al cumplir ciertas condiciones dentro de la aplicación o al alcanzar determinados niveles de experiencia. De esta manera, el sistema de personalización se convierte en un elemento adicional de motivación para el progreso del usuario.

6.1.5 Ranking

El sistema de ranking ofrece a los usuarios una forma de comparar su desempeño con el de otros jugadores dentro de la plataforma. Este apartado se encuentra dividido en tres categorías según la dificultad del juego: fácil, normal y difícil.

En cada una de estas categorías se muestran las diez mejores puntuaciones alcanzadas por los usuarios, lo que permite reconocer a los jugadores más destacados en cada nivel de dificultad. De esta manera, el ranking no solo cumple una función informativa, sino que también introduce un componente competitivo que incentiva la superación personal y fomenta la participación activa dentro de la aplicación.

6.1.6 Estadísticas

En la página de estadísticas, accesible desde la pantalla principal o el menú de acceso rápido, el usuario encontrará distintas métricas separadas por dificultad. La primera es una gráfica generada con Highcharts, que recoge la mejor puntuación de cada día que el usuario haya jugado. Para renderizar esta gráfica se ha decidido establecer un mínimo de tres valores, por lo que el usuario ha tenido que jugar al menos tres días distintos a este modo de juego para poder visualizar la gráfica. A continuación se muestra la puntuación más alta obtenida, y la fecha de la última partida, que aunque son datos que se pueden obtener en la gráfica, es conveniente

remarcar su importancia desde la interfaz para que tenga un peso sobre el usuario y le incite a superar esa marca o volver a jugar. Otros indicadores que encontramos son la cantidad de partidas que el usuario ha jugado y el lugar en el podio que este ocupa. Sobre este último valor: el jugador puede conocer este dato en el apartado de "Ranking global" si se encuentra entre los 10 primeros, de lo contrario, aquí es donde le indica su posición.

6.2 Problemas conocidos

En el desarrollo de cualquier sistema software resulta inevitable encontrar una serie de limitaciones y riesgos que, de no ser controlados, pueden comprometer aspectos fundamentales como la seguridad, la estabilidad o la propia experiencia de usuario. Lejos de verlos como fallos insalvables, reconocer estos problemas es un ejercicio de autocrítica y un punto de partida para seguir mejorando el proyecto. Señalar estos problemas o limitaciones permite anticipar posibles escenarios adversos, comprender el impacto que podrían tener en el uso real de la aplicación y establecer las bases para futuras soluciones o decisiones estratégicas. En este apartado se abordan algunos de los problemas más significativos que acompañan al sistema en su estado actual, con la finalidad de ofrecer una visión clara y transparente de los desafíos que todavía quedan por resolver.

6.2.1 Protocolo HTTPS

Durante el desarrollo de la aplicación, se intentó configurar el protocolo HTTPS mediante un certificado autofirmado para asegurar la conexión con el servidor. Para ello, se siguieron todos los pasos necesarios recomendados para Apache2, incluyendo la generación del certificado y su correcta configuración del servidor. Sin embargo, a pesar de realizar el proceso completo en varias ocasiones, nunca se consiguió que la

página web fuese accesible mediante HTTPS. Tras varias pruebas, como esto no era un objetivo principal para el proyecto se descartó seguir dedicando tiempo a este problema. Se llegó a la conclusión de que el problema probablemente se debiese a un bloqueo del puerto 80 por parte del proveedor de servicios de Internet (ISP), lo cual impide que las peticiones HTTP se redirijan correctamente a HTTPS. Esta limitación imposibilita servir la aplicación de forma segura desde el entorno actual sin recurrir a soluciones externas o a una red con menos restricciones.

6.2.2 Ataques de fuerza bruta de rutas

En el servidor también se han registrado intentos de ataque de fuerza bruta y exploración de rutas sensibles. Estos ataques no lograron su objetivo, pero dejan en evidencia el tipo de información que los atacantes suelen buscar de manera automatizada.

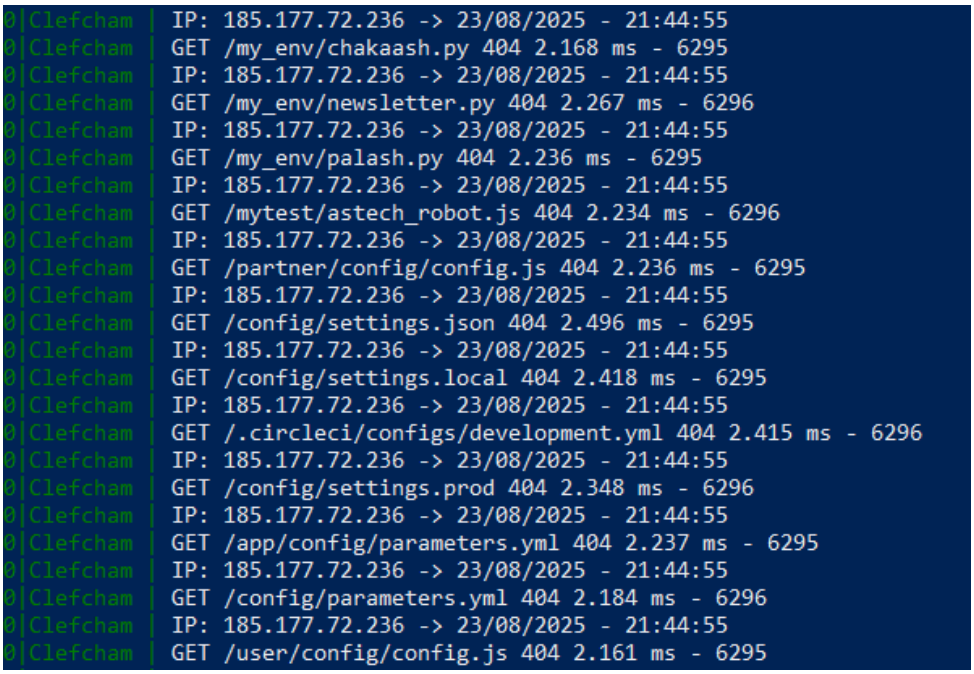

Figura 6-2: Captura de tráfico del servidor mediante la herramienta PM2

Por ejemplo, se recibieron múltiples peticiones dirigidas a rutas como:

- /config/local.yml, /backend/config/settings.yml o /api/config/config.yml. Intentos de obtener archivos de configuración internos que podrían contener credenciales o parámetros críticos de la aplicación.
- /appsettings.json y /.travis.yml. Búsquedas de archivos de configuración comunes en entornos .NET o en integraciones continuas, que suelen almacenar claves y secretos.
- /aws.yml, /aws/credentials o /.aws/config. Intentos de robar credenciales de Amazon Web Services, lo que daría acceso a infraestructura en la nube.
- /sms.py, /helpers/utility.js o /server/s3.js. Intentos de localizar scripts de utilidad o componentes internos que podrían revelar lógica sensible del servidor.
- /login.htm. Prueba básica para comprobar si existe una página de autenticación accesible de forma directa.

Todos estos intentos devolvieron un código 404 (no encontrado), lo que indica que los atacantes no lograron acceder a información crítica. En la práctica, estos escaneos suelen realizarse de forma masiva y automatizada. Su forma de operar es recorrer listas de rutas conocidas en proyectos de software para identificar configuraciones mal expuestas o accesos inseguros.

Este tipo de registros reflejan que, aunque la aplicación no se vio comprometida, es fundamental mantener buenas prácticas de seguridad: no exponer archivos sensibles en el directorio público, configurar correctamente los permisos del servidor y monitorizar los accesos.

6.2.3 Elección de servidor dedicado frente a servicios de hosting

A la hora de desplegar la aplicación, se consideraron distintas alternativas disponibles como Azure, Railway u otras plataformas de hosting en la nube. Estas opciones resultaban atractivas por la facilidad con la que, en teoría, permiten poner en marcha aplicaciones web sin necesidad de preocuparse por la infraestructura subyacente. Sin embargo, en la práctica se encontraron dificultades técnicas que impidieron completar un despliegue totalmente funcional. Este fue el motivo inicial que me llevó a optar por un servidor dedicado, aunque pronto descubrí que esta decisión también conllevaba una serie de ventajas adicionales que merecen ser destacadas.

En primer lugar, disponer de un servidor propio otorga un control total tanto sobre el hardware como sobre el sistema operativo. Esto significa que se ha podido configurar cada aspecto del entorno según necesidades específicas, sin depender de las limitaciones impuestas por un proveedor. A esto se suma la cuestión de la privacidad de los datos, que depende por completo del propietario del servidor, lo que reduce la exposición a terceros y aumenta la confianza en la gestión de la información.

Otro punto relevante ha sido la independencia frente a los modelos comerciales de las plataformas de hosting. Con un servidor dedicado no existen limitaciones derivadas de planes gratuitos, periodos de prueba o renovaciones de suscripción. Esto ha permitido centrar todos los esfuerzos en la aplicación sin estar condicionado por este tipo de restricciones. En cuanto a la seguridad, el control directo sobre la infraestructura permite personalizar las medidas de protección de acuerdo con las características del proyecto, sin tener que ceñirse a configuraciones predefinidas.

Finalmente, uno de los aspectos más valiosos de esta elección ha sido el aprendizaje adquirido. Gestionar un servidor real, desde la instalación de dependencias hasta la configuración de servicios de red y seguridad, ha permitido profundizar en áreas que normalmente quedan ocultas en las soluciones que dan las plataformas de hosting. Esta experiencia práctica ha supuesto un complemento fundamental a los conocimientos adquiridos durante la carrera.

Ahora bien, esta decisión también ha traído consigo ciertos inconvenientes. Al tratarse de un servidor físico en un entorno doméstico, se han producido caídas de corriente que interrumpieron temporalmente el servicio. En otras ocasiones, el propio proveedor de internet reseteó la configuración del router, lo que afectó a la accesibilidad externa de la aplicación y obligó a realizar ajustes adicionales. Asimismo, como ya se ha mencionado, la configuración de HTTPS con certificados válidos ha resultado más compleja de implementar que en plataformas de hosting.

6.2.4 Explotación del sistema de puntuación

Aunque esto no suponga una brecha de seguridad, el juego puede ser explotado. Un compañero desarrolló un script en Python que interactuaba con el navegador para leer el SVG que VexFlow generaba al renderizar las notas musicales. El programa establecía un mapeo entre cada valor de los elementos <path> que representaban las notas y la tecla correspondiente en el teclado del usuario. De esta forma, al ejecutar el script, este quedaba a la espera de la aparición de nuevas notas y pulsaba automáticamente la tecla correcta en apenas 100 ms.

El resultado era un "jugador perfecto", capaz de obtener la máxima puntuación en todos los modos de dificultad sin apenas intervención humana. Esto, a pesar de que no ponía en riesgo los datos ni la infraestructura, sí comprometía la competitividad y la equidad del juego, y derivó en el primer "baneo" oficial en ClefChamp.

6.3 Accesibilidad

La accesibilidad en aplicaciones web es fundamental porque garantiza que personas con diferentes capacidades, ya sean visuales, auditivas, motoras o cognitivas, puedan interactuar con el contenido y las funcionalidades sin barreras. Esto no solo promueve la inclusión y el derecho universal al acceso a la información, sino que también mejora la experiencia de usuario para todos. Además, optimiza el posicionamiento en buscadores, incrementa el alcance potencial de la aplicación y reduce los riesgos legales derivados del incumplimiento de normativas de accesibilidad.

A fin de conseguir los requisitos de accesibilidad en la aplicación, están incluidos los elementos 'alt' en las imágenes o iconos que lo requieran, dejando vacías las imágenes que tienen un fin meramente decorativo, con objeto de que estas sean evitadas por los lectores de pantallas. Se trata de usar una estructura semántica y un orden lógico que facilite su entendimiento a los lectores de pantallas. Se han utilizado colores con suficiente contraste y se han utilizado páginas que comprueban que esto sea así. Del mismo modo se ha evitado usar el color como único método informativo, siendo siempre acompañado de texto o imágenes.

6.4 Responsive

Una página web responsive es aquella capaz de adaptarse automáticamente al tamaño de la pantalla en la que se visualiza, ya sea un ordenador de escritorio, una tablet o un dispositivo móvil. El objetivo es que el contenido se muestre siempre de forma clara y accesible, evitando la necesidad de ampliar manualmente o desplazarse horizontalmente para poder leer o interactuar con los elementos de la página. Como ya se ha mencionado, Bootstrap ofrece un soporte inicial para el diseño

responsive, pero no siempre resulta suficiente, especialmente cuando se incorporan elementos más complejos o se añaden propiedades CSS personalizadas. En estas situaciones pueden surgir desajustes que afectan directamente a la experiencia del usuario, como ocurre en algunas pantallas móviles donde los componentes no encajan correctamente o el diseño pierde la coherencia que se había planteado.

Los problemas derivados de una vista no optimizada pueden manifestarse de diferentes formas. En algunos casos los elementos se renderizan pero no lo hacen respetando la intención original del desarrollador, lo que obliga al usuario a recurrir al zoom o a desplazarse de manera lateral. En otros casos, se omite o se deforma información importante, lo que puede llegar a inutilizar una página entera si el contenido deja de ser legible o queda fuera del área visible. Este tipo de situaciones son particularmente críticas en dispositivos móviles, donde el espacio es limitado y el aprovechamiento del espacio tiene un mayor impacto.

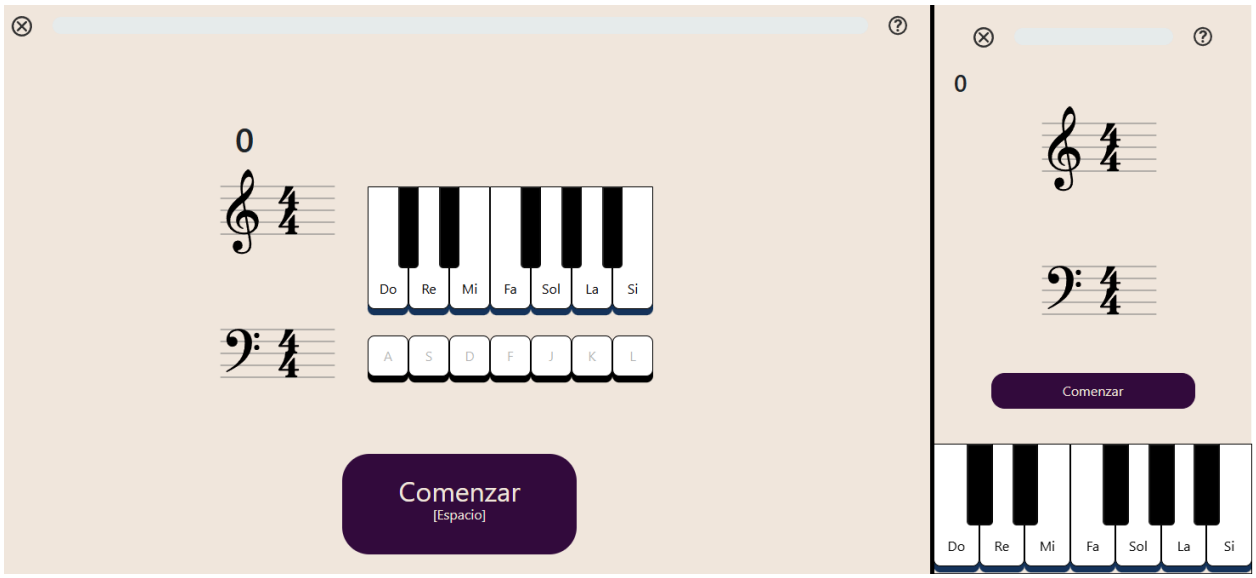

Figura 6-3: Comparativa entre la vista de ordenador y la de móvil

El diseño responsive se organiza normalmente en función de categorías de tamaños de pantalla, que van desde dispositivos pequeños hasta monitores de gran

formato. Esta división suele apoyarse en porcentajes y puntos de corte que permiten definir cómo se redistribuye el contenido en función del espacio disponible.

Para alcanzar este comportamiento se han empleado diferentes técnicas. Una de las principales ha sido el uso de media queries en CSS, que permiten aplicar estilos específicos dependiendo del ancho de la pantalla. En menor medida se han utilizado clases de Bootstrap, aprovechando su sistema de rejilla, y en otros casos ha sido necesario rediseñar secciones concretas de una vista, o incluso la vista completa, con el fin de asegurar una visualización adecuada en todos los dispositivos. Como apoyo durante el desarrollo, se ha recurrido a la herramienta DevTools para la simulación de distintos tamaños de pantalla.

Por último mencionar que por falta de tiempo o prioridad de objetivos, no se ha proporcionado un diseño responsive para todos los tamaños de pantalla. Se ha dado prioridad al tamaño de pantallas grandes, que es donde se ha trabajado y a dispositivos móviles, debido al gran porcentaje de accesos que se hacen a Internet a través de estos dispositivos. La decisión de optimizar estos dispositivos se debe a que, según las estadísticas de StatCounter[10], en Julio de 2025 el 58,39% de accesos a Internet se han hecho a través de un dispositivo móvil, y el 40,04% a través de un ordenador, mientras que el 1,57% restante a través de tablets.

6.5 Despliegue y CI/CD

El término CI/CD hace referencia a dos prácticas fundamentales en el desarrollo de software moderno: la Integración Continua y el Despliegue Continuo. La primera se centra en integrar de manera frecuente los cambios de código en un repositorio común, donde habitualmente se realizan pruebas y validaciones automáticas para asegurar la calidad del proyecto. La segunda, persigue que esos cambios validados

lleguen a producción de forma automatizada, reduciendo la intervención manual y minimizando el riesgo de errores.

En este proyecto, dichas prácticas se han adaptado a una versión más ligera y ajustada a las necesidades reales. No se ha implementado un pipeline completo con pruebas automatizadas, pero sí un mecanismo sencillo y efectivo que mantiene la aplicación en producción siempre sincronizada con la última versión del repositorio. El propósito de esta decisión fue evitar la necesidad de acceder manualmente al servidor para ejecutar comandos, asegurar que la aplicación desplegada refleje en todo momento los cambios más recientes y automatizar la actualización de dependencias junto con la puesta en marcha de la aplicación tras cada modificación.

Aunque esta solución no constituye un sistema de CI/CD completo, sí mantiene la esencia de estas prácticas al reducir el error humano, optimizar el tiempo de despliegue y garantizar la coherencia entre el código del repositorio y la aplicación en producción.

6.5.1 Webhook /deploy

Un webhook de GitHub es un mecanismo que permite que un repositorio notifique de manera automática a un servidor externo cuando ocurre un evento concreto, como puede ser un push. En la práctica, esto significa que cada vez que se sube código nuevo a una rama determinada, GitHub envía una petición HTTP con información sobre ese evento al destino configurado. Esta funcionalidad resulta especialmente útil en procesos de despliegue, ya que permite automatizar tareas que, de otro modo, deberían ejecutarse manualmente en el servidor.

En este proyecto, se configuró un webhook asociado a la rama principal del repositorio. Cada vez que se realiza un push sobre esta rama, GitHub envía una petición POST al servidor, concretamente al endpoint /deploy. Dicho endpoint se encuentra implementado en Express y se diseñó para aceptar únicamente peticiones procedentes de las direcciones IP oficiales de GitHub, lo cual añade una capa de seguridad y evita que de forma externa se pueda invocar el proceso de despliegue.

El endpoint /deploy cumple una función sencilla pero esencial: actuar como punto de entrada al proceso de actualización automática de la aplicación. Al recibir la petición, desencadena la ejecución de un script en el servidor que se encarga de obtener los últimos cambios del repositorio, instalar las dependencias necesarias y reiniciar la aplicación en el puerto correspondiente.

La elección de este enfoque frente a los despliegues manuales responde a la necesidad de hacer el proceso más ágil, confiable y libre de errores humanos. Mientras que un despliegue manual requiere acceder al servidor, descargar cambios, instalar dependencias y reiniciar servicios, el webhook automatiza cada uno de estos pasos y garantiza que la aplicación en producción esté siempre actualizada de manera inmediata tras cada modificación en el código fuente.

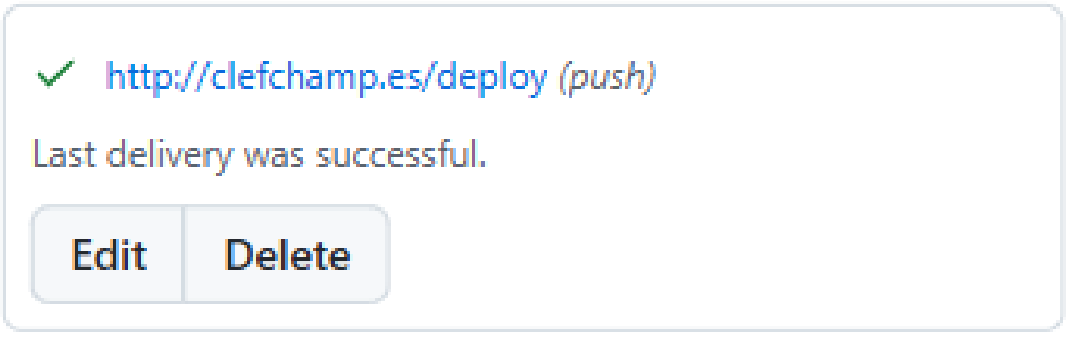

Figura 6-4: Captura de Github que muestra la última llamada a /deploy exitosa

6.5.2 Configuración del servidor y puerto 3000

La aplicación web desarrollada en Node.js se ejecuta de manera interna en el servidor a través del puerto 3000. Sin embargo, este puerto no se expone directamente a Internet, ya que no resultaría seguro ni práctico abrirlo al exterior. En lugar de ello, se optó por utilizar un reverse proxy configurado con Apache, que actúa como intermediario entre el tráfico externo y el servicio interno de Node.

El esquema de funcionamiento es el siguiente: todas las peticiones externas llegan al servidor a través de los puertos 80 (HTTP) o 443 (HTTPS). Apache recibe este tráfico y lo redirige de manera transparente hacia el servicio que corre en el puerto 3000. De esta forma, el usuario accede a la aplicación a través de la dirección habitual del servidor o dominio configurado, sin necesidad de conocer ni interactuar directamente con el puerto en el que se está ejecutando Node.

El endpoint /deploy también se encuentra detrás de esta configuración de Apache, lo que permite que sea accesible de manera segura bajo las mismas reglas que el resto del tráfico. Esto garantiza que el mecanismo de despliegue automático aproveche la infraestructura ya existente de seguridad y enrutamiento, sin exponer directamente el servicio interno al exterior.

Por último, la aplicación en el puerto 3000 se mantiene siempre activa gracias a PM2, un gestor de procesos diseñado para Node. PM2 asegura que la aplicación se reinicie en caso de fallo y que continúe ejecutándose de forma persistente en segundo plano. En conjunto, la combinación de Apache como reverse proxy y PM2 como gestor de procesos proporciona un entorno robusto, seguro y estable para la ejecución de la aplicación.

6.5.3 Flujo de actualización automática

El ciclo de despliegue automático comienza en el momento en que el desarrollador realiza un push al repositorio de GitHub en la rama principal. Este evento activa el webhook previamente configurado, el cual envía una petición al servidor a través del endpoint /deploy. A partir de ahí, se desencadena todo el proceso de actualización de manera autónoma.

El endpoint recibe la petición y ejecuta un script alojado en el propio servidor. Dicho script se encarga de realizar las acciones necesarias para que la aplicación en producción quede alineada con la última versión del código fuente. En primer lugar, se ejecuta un git pull para descargar los cambios más recientes desde el repositorio. A continuación, se lanza un npm install, que garantiza que todas las dependencias estén actualizadas en caso de que se hayan introducido nuevas librerías o versiones diferentes. Finalmente, se utiliza pm2 restart para reiniciar la aplicación en el puerto 3000, asegurando que los cambios entren en vigor de inmediato.

Gracias a este flujo, la nueva versión de la aplicación queda disponible en producción de forma casi instantánea, sin necesidad de intervención manual por parte del desarrollador. Esto supone una serie de ventajas claras: el proceso es más rápido, está completamente automatizado y reduce de manera significativa el riesgo de errores humanos que suelen producirse en despliegues manuales.

No obstante, el sistema implementado también presenta ciertas limitaciones. Se trata de un flujo sencillo, pensado para cubrir las necesidades básicas del proyecto. No incluye pruebas automatizadas previas al despliegue ni contempla la existencia de entornos separados (por ejemplo, de desarrollo, pruebas y producción). Aun así, constituye una solución eficiente y ajustada a los objetivos, que asegura la continuidad del desarrollo y la disponibilidad constante de la aplicación.

Capítulo 7 - Evaluación y conclusiones

Tras el desarrollo del proyecto, en cualquier proyecto, es esencial evaluar los resultados obtenidos y el proceso seguido. Esta sección no se limita a resumir los logros alcanzados, sino que busca analizar con profundidad tanto las fortalezas como las limitaciones del trabajo, identificando oportunidades de mejora y aprendizajes relevantes. Además, permite señalar posibles direcciones futuras, relacionando lo logrado con el potencial de evolución del proyecto y ofreciendo una visión de su alcance.

7.1 Conclusión y reflexiones

El desarrollo de esta aplicación ha permitido explorar cómo la gamificación puede aplicarse al aprendizaje musical como una posible vía de apoyo a los métodos tradicionales. La incorporación de dinámicas propias de los juegos, como la obtención de puntos, la superación de retos o el seguimiento del progreso, se plantea como una estrategia que podría favorecer la motivación y hacer que el aprendizaje se perciba de forma más participativa y activa.

A lo largo del proyecto se ha conseguido desarrollar una herramienta funcional que no solo permite poner en práctica distintos conocimientos técnicos (programación, gestión de bases de datos, diseño de interfaces), sino que también demuestra cómo la tecnología puede adaptarse al aprendizaje musical. En este sentido, la aplicación no se limita a ser un producto de software, sino que se convierte en un puente entre el conocimiento musical y la interacción digital.

Desde una perspectiva personal, el proyecto también ha supuesto un ejercicio de síntesis, ya que ha sido necesario unir competencias adquiridas durante la carrera

con la exploración autónoma de nuevas tecnologías. Esta experiencia invita a reflexionar sobre la importancia de mantener una mentalidad de aprendizaje continuo en la ingeniería de software. Así como sobre el papel del ingeniero no solo como programador, sino también como diseñador de experiencias significativas para los usuarios.

En definitiva, este trabajo reafirma la idea de que la educación musical puede beneficiarse de los enfoques interactivos y tecnológicos, y que el aprendizaje, cuando se convierte en juego, puede estimular tanto la curiosidad como el compromiso de los estudiantes.

7.2 Limitaciones y potencial de mejora

La principal limitación del proyecto se encuentra en la escalabilidad. La aplicación ha sido diseñada para un entorno controlado y con un número reducido de usuarios, por lo que su arquitectura y funcionalidades no están optimizadas para soportar un crecimiento significativo en la cantidad de interacciones o en el volumen de datos. Esto afecta no solo al rendimiento general, sino también a aspectos como la velocidad de respuesta, la gestión simultánea de usuarios y la capacidad de la base de datos para manejar grandes cantidades de información de manera eficiente.

Asimismo, algunas decisiones técnicas tomadas durante el desarrollo, como la elección de tecnologías o la estructura de almacenamiento, están orientadas a facilitar la implementación y el mantenimiento en un contexto de prueba, lo que limita la capacidad de la aplicación para adaptarse a entornos más exigentes o con mayor concurrencia. Aunque estas limitaciones no impiden que la aplicación cumpla su propósito principal, constituyen un factor a tener en cuenta si se considera un despliegue a mayor escala.

7.3 Futuras direcciones

El proyecto ofrece múltiples vías para su desarrollo y ampliación, tanto mejoras planificadas para próximas iteraciones como posibles caminos a explorar, algunas de las cuales podrían potenciar significativamente su utilidad y alcance. Una de las principales direcciones es adaptar la aplicación al contexto educativo formal, creando roles diferenciados de profesores y alumnos. Esto permitiría a los docentes gestionar cursos, monitorizar el progreso del alumnado y diseñar rutas de aprendizaje personalizadas, integrando la herramienta de manera más efectiva en las aulas.

Otra línea de exploración sería la integración de MIDI, lo que abriría la posibilidad de interactuar directamente con instrumentos digitales, ofreciendo una experiencia más completa y cercana a la práctica musical real.

Por otro lado, el proyecto podría evolucionar hacia aplicaciones móviles para Android e iOS, ampliando su accesibilidad y permitiendo que los usuarios aprendan y practiquen en cualquier momento y lugar, lo que también facilitaría la gamificación en contextos más flexibles.

Otra posibilidad interesante es estructurar el aprendizaje de manera progresiva mediante niveles, comenzando desde nociones básicas y creando contenidos introductorios sobre las notas musicales. Esta aproximación permitiría diseñar rutas de aprendizaje más estructuradas y adaptadas al nivel de cada usuario, favoreciendo un entorno accesible para distintos perfiles.

Finalmente, se plantea la opción de convertir la aplicación en un entorno modular, capaz de integrar otros juegos desarrollados en JavaScript. Esto ampliaría su flexibilidad y permitiría añadir nuevos retos y actividades sin necesidad de modificar la estructura central de la aplicación, potenciando su capacidad de evolución y personalización.

Capítulo 8 - Evaluations and conclusions

After completing a project, it is essential to evaluate the results and process. This section does more than summarize the achievements; it seeks to analyze the strengths and limitations of the work in depth, identify opportunities for improvement and highlight relevant lessons learned. Additionally, this evaluation allows us to suggest possible future directions by relating the project's achievements to its potential for evolution and offering a vision of its scope.

8.1 Conclusion and reflections

The development of this application has made it possible to explore the application of gamification to music learning as a way to support traditional methods. Incorporating game dynamics such as earning points, overcoming challenges and tracking progress is proposed as a strategy to promote motivation and make learning more participatory and active.

Throughout the project, a functional tool has been developed that not only enables the practical application of various technical skills (programming, database management, interface design) and demonstrates how technology can be adapted for music learning. In this sense, the application is not just a software product, it becomes a bridge between musical knowledge and digital interaction.

From a personal perspective, the project has been an exercise in synthesis. It required me to combine the skills I acquired during my studies with my independent exploration of new technologies. This experience highlights the importance of maintaining a mindset of continuous learning in software engineering as well as on the

role of the engineer not only as a programmer, but also as a designer of meaningful experiences for users.

In short, this work reaffirms the idea that music education can benefit from interactive, technological approaches and that learning, when transformed into a game, can stimulate curiosity and commitment in students.

8.2 Limitations and improvement potential

The project's main limitation lies in its scalability. The application was designed for a controlled environment with a small user base, so its architecture and functionalities are not optimized to support significant growth in interactions or data volume. This impacts overall performance as well as response speed, simultaneous user management and the database's ability to efficiently handle large amounts of information.

Similarly, certain technical decisions made during development, such as the selection of technologies or storage structure, aim to facilitate implementation and maintenance in a test environment. However, this limits the application's ability to adapt to more demanding environments or those with greater concurrency. While these limitations do not prevent the application from fulfilling its main purpose, they should be considered if larger-scale deployment is planned.

8.3 Future directions

The project offers multiple avenues for development and expansion. Planned improvements for future iterations are one avenue, as are possible paths to explore. Some of these paths could significantly enhance the application's usefulness and scope. One main direction is adapting the application to the formal educational context by creating differentiated roles for teachers and students. Teachers could then

manage courses, monitor student progress and design personalized learning paths, effectively integrating the tool into the classroom.

Another possibility is integrating MIDI, which would enable direct interaction with digital instruments, providing a more comprehensive experience that more closely resembles actual musical practice.

On the other hand, the project could evolve into mobile applications for Android and iOS. This would expand its accessibility, allowing users to learn and practice anytime, anywhere. It would also facilitate gamification in more flexible contexts.

Another interesting possibility is structuring learning progressively through levels, beginning with basic concepts and creating introductory content on musical notes. This approach would enable the design of structured learning paths tailored to each user's level, encouraging an inclusive environment for diverse learners.

Finally, the application could be converted into a modular environment capable of integrating other games developed in JavaScript. This would increase the application's flexibility, allowing new challenges and activities to be added without modifying its central structure, thus enhancing its capacity for evolution and customization.

Bibliografía

[1] BCrypt, «Bcrypt» [En línea]. Available: https://www.npmjs.com/package/bcryptjs

[2] Highcharts, «Highcharts - Interactive Charting Library» [En línea]. Available: https://www.highcharts.com/

[3] VexFlow «VexFlow - HTML5 Music Engraving» [En línea]. Available: https://www.vexflow.com/

[4] Node, «Node.js» [En línea]. Available: https://nodejs.org/es/

[5] Express, «Express - Node.js web application framework» [En línea]. Available: https://expressjs.com/

[6] MariaDB, «MariaDB» [En línea]. Available: https://mariadb.org/https://www.npmjs.com/package/bcryptjs

[7] Bootstrap, «Bootstrap» [En línea]. Available: https://getbootstrap.com/

[8] Initgrammers, «InitGrammers» [En línea]. Available: https://www.initgrammers.com/es

[9] Repositorio SVG, «SVG Repo,Free SVG Vectors and Icons» [En línea]. Available: https://www.svgrepo.com/collection/music-instruments-flat-icons/

[10] StatCounter, «Statcounter: Web Analytics Made Easy» [En línea]. Available: https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet