eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Ingeniería de Software

Desarrollo de una Plataforma Web Interactiva de Datos Musicales Obtenidos de la API de Spotify

Jon Ortega Goikoetxea

Dirección
Miren Bermejo Llopis

19 de octubre de 2024

Agradecimientos

En caso de querer añadir agradecimientos, escribir aquí el texto.

En caso de no querer este apartado, comentalo en el fichero *main.tex*.

Resumen

Escribe aquí el resumen.

Índice de contenidos

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Contexto

El uso de datos personalizados es un componente esencial en el desarrollo de aplicaciones y servicios digitales actuales. Las plataformas buscan ofrecer experiencias ajustadas a las preferencias de los usuarios, utilizando grandes volúmenes de datos para generar contenido adaptado. En este contexto, *Spotify* se ha consolidado como una de las plataformas más destacadas de *streaming* musical, no solo por su extenso catálogo, sino también por su capacidad de ofrecer recomendaciones y estadísticas de uso personalizadas.

Una de las características más valoradas por los usuarios de *Spotify* es la posibilidad de acceder a estadísticas personales que les permiten comprender mejor sus hábitos de escucha. En 2016, como producto de una campaña de márketing viral, *Spotify* lanzó *Spotify Wrapped*, un resumen anual de cada usuario, que generó un gran interés y participación en las redes sociales. Sin embargo, el acceso a estas estadísticas solo está disponible en el mes de diciembre, lo que genera una oportunidad para desarrollar herramientas complementarias que ofrezcan este tipo de análisis de manera más accesible y frecuente.

El acceso a estos datos es posible gracias a la *Web API* pública que *Spotify* pone a disposición de los desarrolladores. De esta manera, es posible obtener una amplia gama de datos sobre el comportamiento del usuario, como sus canciones más escuchadas, artistas favoritos y playlists. Además, la API ofrece acceso a muchos datos adicionales que no son utilizados en *Spotify Wrapped*, pero realizando diferentes combinaciones, se pueden crear nuevas formas enriquecidas de análisis que presenten la información de manera más profunda y personalizada.

También es necesario mencionar, que el crecimiento de las aplicaciones web interactivas ha sido posible gracias a tecnologías y frameworks modernos, que permiten crear interfaces rápidas y eficientes, optimizando el rendimiento y la experiencia de usuario y simplificando el desarrollo. Además, los sistemas de integración y despliegue continuo (CI/CD) facilitan la entrega de aplicaciones de manera automatizada y escalable, garantizando que estén siempre actualizadas y accesibles para los usuarios.

1.2. Motivación

La elección de este Trabajo de Fin de Grado surge de un interés personal que he tenido desde el inicio de mi carrera universitaria. Desde el primer año, he tenido la idea base para implementar un servicio en torno a la *Web API* de *Spotify*. Sin embargo, en ese momento no contaba con los conocimientos necesarios para llevarlo a cabo. Ahora, tras completar la carrera, me siento con las capacidades necesarias para poder realizar dicho proyecto y materializar esta idea.

Como usuario habitual de *Spotify*, siempre me ha fascinado la función de *Spotify Wrapped*. No obstante, me resulta limitante que esta información solo esté disponible durante un breve periodo del año. Existen otras herramientas y páginas web que analizan datos de *Spotify*, pero suelen ofrecer estadísticas genéricas y demasiado básicas que carecen de profundidad. Estoy convencido de que es posible obtener estadísticas mucho más interesantes y, conversando con compañeros y otros usuarios, existe un interés general por acceder a estas estadísticas en cualquier momento, no solo una vez al año. La música que cada individuo escucha es algo personal y, a menudo, forma parte de su identidad. Por ello, considero que este proyecto no solo es interesante y motivador para mí, sino que también puede aportar valor a otros usuarios que comparten esta misma idea.

La posibilidad de crear un proyecto que me interese de principio a fin, y que yo mismo desearía utilizar, es una gran motivación. Además, la selección de tecnologías que he decidido emplear, como se detallará más adelante, no solo son ampliamente demandadas en el mercado actual, sino que también contribuyen a mi crecimiento profesional y enriquecen mis conocimientos.

1.3. Objetivos

Para empezar a caracterizar el proyecto de manera concreta, en este apartado se define el propósito principal del desarrollo de la plataforma, así como los objetivos específicos que desglosan de manera más detallada las acciones necesarias para cumplir dicho objetivo general. Además, también se relaciona este proyecto con los Objetivos de Desarrollo Sostenible (ODS) de las Naciones Unidas, tomando como marco de referencia la *EHUagenda 2030* de la UPV/EHU.

1.3.1. Objetivos del Proyecto

El **objetivo general** de este proyecto es desarrollar una plataforma web interactiva que permita a los usuarios de *Spotify* acceder a las visualizaciones y análisis de sus datos musicales personales cuando lo deseen. Para alcanzar este objetivo general, se plantean los siguientes **objetivos específicos**:

- Analizar la API de *Spotify* para obtener los datos necesarios.
- Desarrollar la lógica necesaria para filtrar, organizar y transformar los datos obtenidos, preparándolos para su representación gráfica y asegurando que las estadísticas sean relevantes y significativas.
- Diseñar una interfaz de usuario intuitiva, interactiva y adaptable a diferentes dispositivos.

- Adoptar tecnologías modernas como *Next.js* y *TypeScript* para beneficiarse de las ventajas que ofrecen.
- Implementar prácticas de CI/CD usando la plataforma de *Vercel*.
- Implementar políticas de seguridad con respecto a los datos de usuarios establecidas por *Spotify* para desarrolladores.
- Documentar el proceso de desarrollo.

1.3.2. Objetivos de Desarrollo Sostenible

En este proyecto se alinean varios Objetivos de Desarrollo Sostenible (ODS). Como ya se ha mencionado anteriormente, se toma el marco de referencia de la *EHUagenda 2030* de la UPV/EHU. Esta agenda, "recoge la contribución de la UPV/EHU a 12 de los 17 ODS de la Agenda 2030, al que ha sumado el su compromiso con la diversidad lingüística y cultural a través del ODS 17+1" [1]. A continuación, se detallan los objetivos específicos relacionados:

Figura 1.1: Los ODS alineados con este trabajo.

- **ODS 3: Salud y Bienestar**. La música juega un papel importante en el bienestar emocional y mental. La capacidad que ofrece esta aplicación de mejorar la experiencia musical y permitir a los usuarios conectar más profundamente con su música contibuye positivamente a su bienestar general.
- **ODS 9: Industria, Innovación e Infraestructura**. Como este proyecto implica el desarrollo utilizando tecnologías modernas como React, Next.js y Vercel, se fomenta la innovación tecnológica y se contribuye al desarrollo de infraestructuras digitales eficientes.
- **ODS 18 (17+1): Garantizar la diversidad lingüística y cultural**. Al permitir que los usuarios exploren y aprecien música de diferentes culturas y en diversos idiomas, se facilita la exposición a estas, fomentando así el entendimiento y la apreciación cultural.

1.4. Estructura de la Memoria

La estructura de esta memoria refleja las diferentes etapas por las que atraviesa un proyecto de desarrollo software. Cada capítulo aborda aspectos clave que contribuyen al logro de los objetivos propuestos.

En la Introducción se ha establecido el contexto del proyecto, la motivación que impulsa su realización y los objetivos tanto del proyecto como los relacionados con los ODS. A

continuación, en la Planificación, se define el alcance del proyecto y se aborda lodo lo relacionado con la gestión de tareas, riesgos y calidad del proyecto. Además, se presentan las tecologías que se han utilizado durante todo el desarrollo.

El Análisis se centra en el estudio de la API de *Spotify*, el cual es **fundamental para el desarrollo del proyecto**. Se especifican los requisitos y se presentan los casos de uso que guiarán el desarrollo de la aplicación. En conjunto con el capítulo anterior, en el Diseño se describe la arquitectura del sistema y la estructura de la interfaz de usuario (UI), tanto a nivel técnico como visual. Además, se abordan aspectos de seguridad y se planifica el diseño de las pruebas.

La Implementación detalla el proceso de desarrollo de la aplicación, incluyendo las decisiones tomadas durante esta fase. Se describen los retos enfrentados y cómo se han resuelto. Seguido, en el capítulo de las Pruebas se exponen las pruebas realizadas y se analizan los resultados obtenidos. Como consecuencia lógica, en Despliegue se explica el proceso de puesta en producción de la aplicación y las estrategias de CI/CD implementadas.

Finalmente, en Conclusiones se hace una reflexión sobre los resultados alcanzados, evaluando el cumplimiento de los objetivos iniciales, el seguimiento en comparación a la planificación inicial y se discuten las lecciones aprendidas y se proponen líneas futuras de trabajo.

Mediante este flujo, se espera que el público lector no tenga problemas para seguir la línea de pensamiento que se ha llevado a cabo y que cada apartado quede aclarado o, en menor medida, justificado por los capítulos anteriores.

Capítulo 2

Estado del Arte

Antes de proceder con la planificación del proyecto, es un buen ejercicio analizar las alternativas similares que actualmente existen en el mercado. Como se menciona en la sección de Contexto, la mayoría de otras webs ofrecen estadísticas muy básicas, como visualizar los *top géneros*, *artistas* y *álbumes* del usuario, además de su *historial de reproducción* más reciente. Cada una de estas funcionalidades corresponde a llamadas directas a la *Web API* de *Spotify*, siendo estadísticas triviales de implementar, con una mínima necesidad de procesamiento de datos.

Estadísticas	stats.fm	Obscurify	Stats for Spotify	Replayify	Zodiac Affinity	Music Scapes
Básicas	Top canciones Top artistas Top géneros Historial reciente	Top canciones Top artistas Top géneros	Top canciones Top artistas Top géneros Historial reciente	Top canciones Top artistas Historial reciente		
Avanzadas	Tiempo escuchado Número de artistas Número de álbumes Porcentaje por décadas					
Creativas	Escuchas por hora del día	Rating de "obscurity" Top artistas "oscuros" Top canciones "oscuras" Tus estados de ánimo			5 canciones según tu signo zodiacal	Imagen basada en las propiedades de las canciones que más escuchas

Tabla 2.1: Comparativa de funcionalidades ofrecidas por otros servicios afines.

Una de las opciones más populares es la aplicación web *stats.fm*[1]. Esta ofrece las funcionalidades básicas mencionadas a todos los usuarios, pero, mediante un plan de pago, se habilitan gráficas más avanzadas. Estas gráficas no se obtienen directamente de la API, sino que requieren que el usuario descargue manualmente los datos históricos guardados por *Spotify* y los suba a la página web como un archivo comprimido. De esta manera, se

[1] stats.fm: https://spotistats.app/

generan estadísticas relativamente más complejas, pero que aún carecen de "creatividad" en su presentación.

Otro servicio notable es *Obscurify*[2], que se centra en la "oscuridad" o "rareza" de la música que el usuario escucha. El concepto principal de *Obscurify* es identificar las canciones y artistas que son menos populares entre otros usuarios de *Spotify*, clasificándolos como más "oscuros". Esta funcionalidad permite que el usuario se sienta especial al escuchar música que no es común. Sin embargo, su enfoque está limitado en su mayoría a este concepto, lo que deja fuera otras formas de visualización o análisis más amplios y variados.

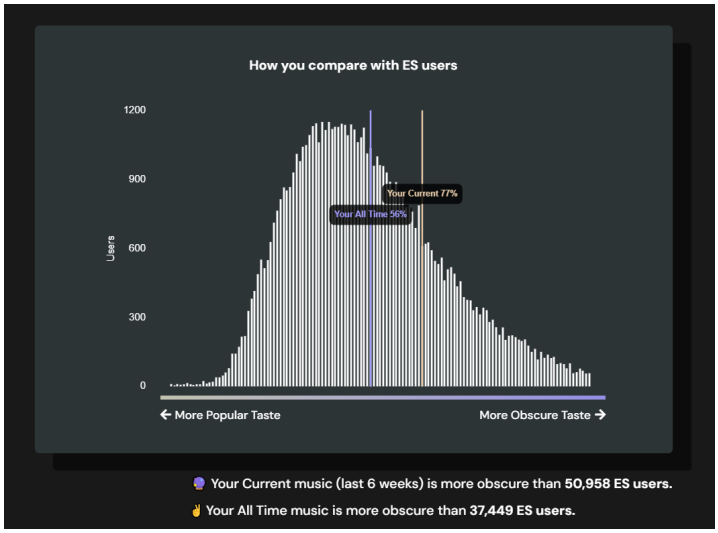

Figura 2.1: Estadística de "oscuridad" de *Obscurify*.

Además de estas páginas principales, existen otras que se enfocan toda su funcionalidad en una única característica original, sin ofrecer mucho más. Dos ejemplos destacados son *Zodiac Affinity*[3] y *MusicScapes*[4]. La primera genera una recomendación de cinco canciones en base a los hábitos de escucha del usuario y su signo zodiacal, mientras que la segunda crea de manera procedural una imagen basada en diferentes propiedades de las canciones que el usuario ha escuchado recientemente. Estas páginas no ofrecen ninguna funcionalidad adicional, limitándose a esa única característica. Aunque hay otros servicios similares, he elegido estos dos ejemplos por su aspecto más "acabado", ya que muchas otras alternativas se presentan más como una prueba de concepto o una demo, en lugar de páginas completamente desarrolladas.

[2]Obscurify: https://www.obscurifymusic.com/
[3]Zodiac Affinity: https://zodiacaffinity.eu/
[4]MusicScapes: https://musicscapes.herokuapp.com/

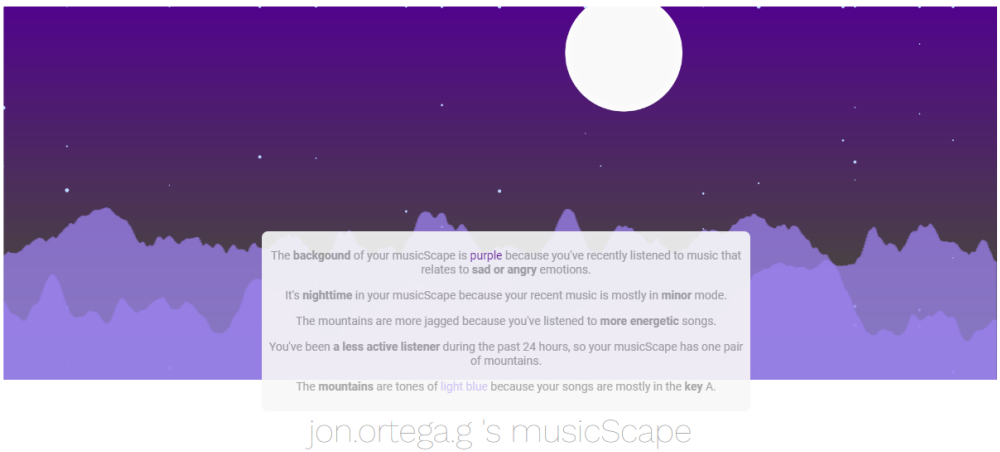

Figura 2.2: Imagen generada por MusicScapes.

Capítulo 3

Planificación

3.1. Alcance

Definir con precisión el alcance es fundamental para asegurar que el desarrollo se ajuste a los objetivos propuestos y se realice dentro de los recursos y tiempos establecidos. Para ello, en esta sección se delimitarán las funcionalidades, exclusiones y limitaciones que se esperan en este proyecto.

3.1.1. Funcionalidades Incluidas

En la plataforma web se ofrecen las siguientes funcionalidades principales:

- Autenticación segura mediante las credenciales de *Spotify*.
- *Home* o Panel Inicial donde se muestran la información básica de la cuenta.
- Análisis detallado y visualizaciones gráficas avanzadas, interactivas y actualizadas de sus datos musicales.
- Interfaz adaptativa, intuitiva y responsiva.
- Cierre de sesión seguro.

3.1.2. Exclusiones

Para establecer expectativas claras sobre el alcance del proyecto, se detallan a continuación las funcionalidades que **no** serán incluidas en la plataforma web:

- No se desarrollarán aplicaciones nativas de otras plataformas como móvil, PC, Mac o Linux; el acceso será exclusivamente a través de la web.
- Aunque se seguirá un diseño intuitivo, no se implementarán funcionalidades específicas de accesibilidad avanzadas como compatibilidad con lectores de pantalla o navegación por teclado.
- La plataforma se enfoca exclusivamente en la integración con *Spotify*; se excluyen todos los otros servicios de streaming como *Apple Music*, *Deezer*, etc.

- No se almacenarán de forma persistente datos personales del usuario en servidores propios más allá de lo necesario para la sesión actual; todos los datos se obtendrán directamente de la API de *Spotify* y se manejarán en tiempo real.
- Se excluye el desarrollo de funcionalidades relacionadas con la interacción social (envío de mensajes, compartir estadísticas, rankings entre usuarios, etc.) dentro o a través de la plataforma, ya que superarían el alcance recogido dentro de un TFG.

3.1.3. Limitaciones

Durante el desarrollo del proyecto, se han identificado las siguientes limitaciones que han podido afectar al alcance y a las funcionalidades de la web:

- Las políticas de seguridad de *Spotify* impiden el almacenamiento persistente de datos personales, limitando funcionalidades que requieran conservar información del usuario entre sesiones.
- El procesamiento de los datos se ve limitada por los recursos computacionales que la nube de *Vercel* ofrece, descartando técnicas avanzadas como el aprendizaje automático.
- El tiempo y los recursos disponibles para el desarrollo del proyecto son finitos, lo que ha obligado a priorizar funcionalidades esenciales y descartar características adicionales.
- Al hacer uso de una API de terceros, todas las funcionalidades necesitan una conexión activa a Internet para poder funcionar.

3.2. Gestión de Tareas

3.2.1. Descripción de Tareas

3.2.2. Dedicaciones

3.2.3. Dependencias entre Tareas

3.2.4. Periodos de Desarrollo

3.2.5. Hitos

3.3. Gestión de Riesgos

A continuación, se detallan los riesgos identificados que pueden afectar el desarrollo y el alcance del proyecto:

- Dependencia de la API de *Spotify*: La API de *Spotify* podría cambiar, tener interrupciones o limitar el acceso a ciertos datos.
- Limitaciones de la API de *Spotify*: Restricciones en los tipos de datos accesibles, datos históricos limitados y permisos que los usuarios pueden no conceder.

- Tasa de peticiones (*rate limiting*) de la API de *Spotify*: Exceder el número máximo de peticiones permitidas en un período de tiempo puede afectar la capacidad de actualización de datos en tiempo real.
- Falta de almacenamiento persistente de datos del usuario: La imposibilidad de guardar datos entre sesiones limita ciertas funcionalidades como la personalización o el historial de preferencias del usuario.
- Dependencia de servicios de terceros (*Vercel*): Posibles interrupciones en los servicios de despliegue continuo o cambios en las políticas de *Vercel*.
- Tiempo limitado para el desarrollo: El tiempo y los recursos disponibles son finitos, lo que obliga a priorizar ciertas funcionalidades esenciales y descartar características adicionales.
- Necesidad de conexión a Internet: La plataforma requiere una conexión a Internet activa para funcionar correctamente, lo que puede afectar a los usuarios en entornos con conectividad limitada.
- Compatibilidad con navegadores y dispositivos antiguos: La plataforma puede no ser completamente funcional en navegadores o dispositivos más antiguos, lo que afecta a un porcentaje pequeño de usuarios.
- Falta de experiencia con las tecnologías utilizadas: La curva de aprendizaje de tecnologías nuevas como *Next.js* y *TypeScript* podría ralentizar el desarrollo.
- Rendimiento en dispositivos móviles o de gama baja: Las visualizaciones avanzadas pueden no funcionar de manera óptima en dispositivos con menor capacidad de procesamiento.
- Seguridad y privacidad de los datos del usuario: El manejo incorrecto de los datos de los usuarios podría infringir las políticas de *Spotify* o el RGPD, con consecuencias legales o de suspensión del servicio.

3.4. Gestión de Calidad

El objetivo de la gestión de calidad es garantizar que el proyecto cumpla con los requisitos funcionales y no funcionales, asegurando un producto robusto, eficiente y alineado con las expectativas de los usuarios.

3.4.1. Línea Base

Los estándares mínimos establecidos para asegurar la calidad del proyecto son:

- Buenas prácticas de desarrollo, aseguradas mediante revisiones de código.
- Uso de metodologías ágiles para una planificación iterativa y control de calidad.
- Diseño de interfaz de usuario intuitiva y conforme a los estándares de accesibilidad.

3.4.2. Criterios de Éxito y Aceptación

Para asegurar que el producto cumple con las expectativas, los criterios de aceptación incluyen:

- Cumplimiento de los requisitos funcionales y no funcionales.
- Ausencia de errores críticos que afecten la experiencia de usuario.
- Cumplimiento de los objetivos de rendimiento (tiempos de carga, capacidad de respuesta).
- Feedback positivo en las pruebas de usabilidad realizadas con usuarios finales.

3.4.3. Plan de Calidad

El plan de calidad incluye:

- Pruebas automatizadas, incluyendo pruebas unitarias, de integración y de aceptación.
- Revisión de código por pares para asegurar la consistencia y calidad del código.
- Implementación de CI/CD usando Vercel para garantizar despliegues automáticos y controlados.
- Pruebas de usabilidad con usuarios reales para validar la experiencia del usuario.

3.4.4. Herramientas y Tecnologías

Para asegurar la calidad del proyecto, se utilizarán las siguientes herramientas:

- **CI/CD**: Vercel y GitHub Actions para control de calidad durante el despliegue.
- **Testing**: Jest para pruebas unitarias y Cypress para pruebas de integración.
- **Control de calidad del código**: ESLint y Prettier para mantener estándares de código.
- **Monitoreo del rendimiento**: Lighthouse para verificar tiempos de carga y rendimiento.

3.4.5. Indicadores de Calidad

Los KPIs definidos para evaluar la calidad del proyecto son:

- Número de bugs críticos detectados.
- Tiempos de carga y respuesta del sistema.
- Porcentaje de cobertura de pruebas.
- Satisfacción del usuario en pruebas de usabilidad.

3.5. Tecnologías y Herramientas Utilizadas

Capítulo 4

Análisis

Capítulo 5

Diseño

5.1. Arquitectura del Sistema

5.2. Diagrama de Componentes de React

5.3. Interfaz de Usuario

5.4. Diagramas de Secuencia

5.5. Seguridad

5.6. Diseño de Pruebas

Capítulo 6

Implementación

Capítulo 7

Pruebas

Capítulo 8

Despliegue

8.1. Vercel

8.2. CD/CI

Capítulo 9

Conclusiones

Apéndice

Apéndices

Bibliografía

[1] "EHUagenda 2030 - Sostenibilidad - UPV/EHU." [Online]. Available: https://www.ehu.eus/es/web/iraunkortasuna/ehuagenda-2030 Ver página 3.