DuocUC

Justificación de la Arquitectura de la Aplicación Featuring

Índice

Contenido

1.-Introducción:

En este proyecto, hemos diseñado una red social para músicos cuyo propósito es ofrecer una plataforma donde los usuarios puedan conectar, colaborar, compartir proyectos, y explorar nuevas oportunidades musicales. La arquitectura de la aplicación se ha planteado de manera que facilite un alto rendimiento, escalabilidad, seguridad y una experiencia de usuario intuitiva. Para lograr estos objetivos, hemos decidido usar una combinación de tecnologías modernas y robustas que se complementan para una aplicación como Featuring.

2.-**Frontend: React Native con Expo, TypeScript y JSX, NativeWind**

React Native con Expo:

★ **Justificación:**

- **React Native** es una de las tecnologías más populares para el desarrollo de aplicaciones móviles, los integrantes del grupo encargado de desarrollar Featuring, estaban familiarizados de antemano con esta tecnología.La creciente comunidad de React Native, permite el acceso a una gran cantidad de documentación, si bien su curva de aprendizaje es mediana, esto compensa ese hecho, otorgando una gran flexibilidad a la hora de desarrollar. Su estructura basada en componentes permite que la aplicación sea más fácil de mantener al usar componentes reutilizables, sin necesidad de duplicar código, otorgando escalabilidad y mantenibilidad, al funcionar mediante estados, esta permite reflejar los cambios constantes que tiene una aplicación como Featuring en cuanto el estado de un componente cambia(nuevos mensajes, notificaciones, cambios en el perfil), permite también acceso a funcionalidades nativas como la cámara para grabar videos , cargar imágenes y notificaciones push, esto es clave para una app con las funcionalidades de **Featuring.**

- **Expo** posee una gran compatibilidad con React Native, Expo proporciona una serie de herramientas que permiten simplificar la configuración inicial y el desarrollo de la aplicación, permitiendo ver los cambios en tiempo real, con su

aplicación móvil Expo Go, la cual permite correr la aplicación en el teléfono celular. Además Expo incluye Apis y componentes nativos que con Featuring son necesarios como:

- **Notificaciones push**
- **Acceso a cámara y micrófono**
- **Acceso a ubicación**

Expo con React Native es una elección estratégica para el desarrollo de Featuring, ya que proporciona un entorno de desarrollo optimizado, herramientas preconfiguradas, acceso a funcionalidades nativas cruciales (como notificaciones push, acceso a la cámara, almacenamiento de archivos, etc.), y la posibilidad de trabajar de manera rápida y eficiente en un proyecto multiplataforma

.

Comparación de Opciones Tecnológicas

Expo React Native vs. Flutter vs. React Native Nativo

Cuando comenzamos a evaluar las opciones para el desarrollo de la app móvil, consideramos varias alternativas:

Flutter: Es un framework prometedor para el desarrollo multiplataforma, pero aunque ofrece un rendimiento muy cercano al nativo, la curva de aprendizaje es un poco más pronunciada si ya estamos familiarizados con JavaScript y React. Además, el ecosistema de **Flutter** aún no tiene la misma madurez y variedad de librerías que **React Native**.

React Native Nativo (sin Expo): Aunque React Native ofrece un control más detallado sobre el proyecto y una mejor optimización del rendimiento, usarlo sin **Expo** significa tener que configurar y mantener más partes del sistema (por ejemplo, gestión de dependencias nativas, configuración de build, etc.), lo que incrementaría el tiempo y esfuerzo de desarrollo.

Elección de Expo React Native:

Finalmente, elegimos **Expo React Native** por su facilidad de uso, integración rápida con herramientas de desarrollo como **Expo Go** (para previsualización en vivo), y su amplio ecosistema de componentes listos para usar. Además, su integración con servicios como **Push Notifications** y su enfoque simplificado para el desarrollo nativo nos permitió acelerar el desarrollo y concentrarnos más en la funcionalidad de la app.

TypeScript, JSX y NativeWind

★ **Justificación:**

- **TypeScript** añade tipado estático a javascript, facilitando la legibilidad del código, al ser Featuring una aplicación con funcionalidades complejas es necesario tener un código ordenado y entendible, TypeScript garantiza que las librerías y herramientas de React Native sean más fáciles de integrar y usar por su tipado estático el cual permite mitigar errores, otorgando facilidad de corrección y orden.
- **JSX** permite escribir estructuras HTML dentro de archivos JavaScript, esto facilita la comprensión visual de las interfaces, al no tener que crear elementos del DOM, en su lugar simplemente se escribe código HTML en el mismo archivo javascript, en Featuring, donde la interfaz de usuario es crucial en la interacción de la aplicación es importante que sean legibles y fáciles de modificar.
- **NativeWind** permite el uso de clases de utilidad para el diseño de interfaces, similar a **Tailwind CSS** en la web. Al igual que Tailwind, NativeWind ofrece un enfoque rápido y flexible para el estilizado de la aplicación, permitiendo la creación de interfaces atractivas de forma eficiente y con código más limpio. NativeWind reduce la complejidad de escribir estilos personalizados, ahorrando tiempo al funcionar mediante clases.

3.-Backend: Supabase, PostgreSQL, Supabase Auth, Supabase Storage, Supabase Realtime

★ **Justificación:**

Supabase:

Es una plataforma de Backend as a Service (BaaS) que proporciona una alternativa de código abierto a Firebase. Este otorga una serie de herramientas las cuales implican el manejo de usuarios, publicaciones, mensajes, y contenido multimedia (como audio o video), este maneja estas cosas con una infraestructura que es eficiente para manejar ese tipo de datos. Supabase permite crear aplicaciones sin gestionar el backend de manera manual, en un contexto como este, eso permite ahorrar tiempo al estar desarrollando

una aplicación con varias funcionalidades complejas como lo es Featuring, las siguientes herramientas ayudan en esto perfectamente al estar estrechamente relacionadas:

- **Supabase Auth:** Esto ofrece autenticación sencilla y segura, está integrado directamente con la base de datos de supabase, facilitando la gestión de los datos, restablecimiento de contraseñas y autenticación social.
- **Supabase Storage:** Permite almacenamiento de contenido multimedia, Featuring tiene una serie de funcionalidades relacionadas como lo es imágenes de perfil y vídeos, entre otros contenidos multimedia, esta herramienta facilita la gestión de este tipo de archivos.
- **Supabase Realtime:** Permite sincronizar datos en tiempo real, en una aplicación como Featuring, donde los usuarios están en constante interacción en funcionalidades como el chat, es fundamental el uso de una herramienta como esta.

PostgreSQL:

Es un sistema de gestión de bases de datos relacional, este es conocido por su alto rendimiento en consultas complejas, siendo esto importante para Featuring ya que en funcionalidades como la de la conexión de músicos que requieren consultas de este tipo, permite flexibilidad en el desarrollo, al ser este también capaz de manejar un gran volumen de datos otorgando rendimiento.

Comparación de Opciones Tecnológicas

Supabase vs. Firebase vs. Backend Tradicional

A la hora de elegir el backend para la aplicación, consideramos varias alternativas:

Firebase: Firebase es una opción popular para aplicaciones en tiempo real, especialmente para proyectos que requieren notificaciones push o funcionalidades en tiempo real. Sin embargo, uno de los principales inconvenientes de Firebase es el costo, que tiende a ser impredecible a medida que la base de usuarios crece. Además, la dependencia de un proveedor propietario puede ser un problema si buscamos mayor flexibilidad o control sobre nuestra base de datos.

Backend Tradicional (Node.js + PostgreSQL): Usar un servidor propio con una base de datos tradicional nos habría dado más control sobre la infraestructura, pero

también habría requerido mucho más tiempo y esfuerzo en la configuración, mantenimiento y escalabilidad del sistema. Esto no era una prioridad para nosotros, ya que preferimos centrarnos en la experiencia de usuario y las funcionalidades del producto.

Elección de Supabase:

Elegimos **Supabase** principalmente por su arquitectura basada en **PostgreSQL**, su facilidad para gestionar autenticación, bases de datos en tiempo real, y almacenamiento de archivos. Además, **Supabase** tiene un coste más predecible y una infraestructura más fácil de manejar en comparación con Firebase. La integración directa con **PostgreSQL** facilita consultas complejas y el crecimiento horizontal a medida que la base de datos crece, algo fundamental para una aplicación de red social que espera manejar grandes cantidades de usuarios y datos.

4.- Panel de Administración: Next.js

★ **Justificación:**

- **Next.js:**

  Para el panel de administración, decidimos usar Next.js, una framework de React que optimiza la creación de aplicaciones web mediante la generación de páginas estáticas y dinámicas. Consideramos otras opciones como Vue.js o Angular, pero Next.js ofreció una integración perfecta con React (lo que ya usábamos en el frontend de la app móvil) y características como SSR (Server-Side Rendering) y SSG (Static Site Generation), que optimizan tanto el rendimiento como la SEO.
- Elección de Next.js para el Panel de Administración: La razón principal para elegir Next.js es que nos permitió construir un panel de administración rápido, eficiente y con buena performance, utilizando el mismo stack tecnológico de React. Además, Next.js facilita la integración con Supabase para gestionar datos en tiempo real, lo que garantiza que el panel de administración y la app móvil compartan la misma infraestructura y lógica de negocio.

Escalabilidad y Rendimiento

La infraestructura basada en Supabase es altamente escalable, permitiendo que la aplicación crezca horizontalmente a medida que aumenta la base de usuarios. Además, la base de datos PostgreSQL utilizada por Supabase es capaz de manejar grandes cantidades de datos sin sacrificar el rendimiento. La interfaz de usuario, construida con Expo React Native y Next.js, está optimizada para ofrecer tiempos de respuesta rápidos tanto en la aplicación móvil como en el panel de administración.

Seguridad

La seguridad es una prioridad en nuestra arquitectura. Supabase Auth garantiza que los usuarios estén autenticados de manera segura, utilizando protocolos como OAuth 2.0 para el inicio de sesión con cuentas de Google. Además, todos los datos sensibles, como contraseñas y perfiles de usuario, se cifran y protegen utilizando las mejores prácticas de seguridad.

5.-Conclusión

La aplicación móvil está construida con Expo React Native, lo que nos permite entregar una experiencia fluida en ambas plataformas (iOS y Android) sin tener que duplicar el trabajo. Expo ofrece una serie de herramientas que han simplificado nuestro proceso de desarrollo, como el hot-reloading, el manejo simplificado de notificaciones push, y el acceso a APIs nativas para la cámara, ubicación y más. Utilizamos Supabase como nuestro servicio de backend para gestionar la base de datos, la autenticación de usuarios, el almacenamiento de archivos multimedia y las operaciones en tiempo real. Supabase se basa en PostgreSQL, lo que garantiza un rendimiento excelente en consultas complejas y la posibilidad de manejar grandes volúmenes de datos a medida que la base de usuarios crece. Además, Supabase proporciona una interfaz intuitiva para gestionar y consultar datos sin necesidad de escribir un servidor completo, finalmente el panel de administración está desarrollado con Next.js, permitiendo a los administradores gestionar el contenido, los usuarios y las interacciones de la plataforma de manera eficiente. Dado que Next.js también soporta la generación de páginas estáticas, el rendimiento de la aplicación web es rápido y optimizado, incluso con un volumen alto de contenido.