

FITCONNET
LET'S THRIVE

ÍNDICE

3	Introducción
4	Motivación
5	Objetivos
6	Historias de usuario
10	Stack Tecnológico
15	Modelo de la base de datos
22	Prototipo de figma
29	Manual de despliegue
31	Enlaces

TÍTULO DE LA APLICACIÓN



DESCRIPCIÓN

Esta app esta diseñada para funcionar como una aplicación dual, es decir, funciona como una red social y una de salud y bien estar.

En ella podrás, mantener un historial de tu actividades física pero además podrás compartirla con tus amigos que tengan la app.

Esta pensada para crear comunidad y hacer planes de entreno conjuntos, además de compartir los entrenos individuales en el feed.

MOTIVACIÓN

Esta idea nace reciclada de un proyecto que se nos mandó hacer en la dual, en el cual se nos daban 3 palabras clave de las cuales saliera una idea para una aplicación, las mías fueron: Humanidad – Naturaleza – Gráfica.

Lo primero que se me ocurrió fue hacer una aplicación simple de fitness, donde registraras tu actividad y progreso, sin embargo pensé ya había demasiadas aplicaciones y quería hacer algo más novedoso, diferente.

Fue ahí cuando se me ocurrió hacer la aplicación como una red social, esto da significado al eslogan, ya que este significa prosperemos, vamos a prosperar o a prosperar. Tiene un mensaje de comunidad que se ayuda a mejorar en el área de la salud, tema que me es especialmente relevante porque antes de programador soy nutricionista.

Además es una idea poco explotada ya que hay menos de 5 apps en el mercado, de las cuáles las más conocidas son: Fitbit, MapMyFitness y Nike Training Club.

Sin embargo, todas tienen partes de pago para funciones premium y están más enfocadas al entrenamiento con planes y programas, mientras que mi app está más enfocada en la parte social así como instagram.

OBJETIVO

Desarrollar una aplicación móvil de red social, FitConnect, que fomente un estilo de vida saludable mediante la creación de una comunidad centrada en el deporte.

La aplicación permitirá a los usuarios mayores de 18 años registrar sus entrenamientos, conectar con amigos y compartir rutinas, proporcionando una plataforma motivadora y enfocada exclusivamente en actividades deportivas.

FitConnect resolverá la falta de una red social dedicada al ejercicio, eliminando el ruido de otras publicaciones no relacionadas y cubriendo la necesidad de un entorno especializado donde los usuarios puedan apoyarse mutuamente en el establecimiento y mantenimiento de sus rutinas de entrenamiento.

HISTORIAS DE USUARIO

01 RF01 - Sistema de inicio de sesión:

Como usuario registrado, quiero poder iniciar sesión en la aplicación utilizando mi correo electrónico para acceder a todas las funcionalidades de la plataforma.

02 RF02 - Registro de usuario:

Como usuario nuevo, deseo poder registrarme en la aplicación utilizando mi nombre, email, edad y una contraseña.

03 RF03 - Gestión de perfil:

Como usuario, quiero tener un perfil personalizado donde pueda ver y editar mi información personal, incluyendo mi nombre, edad, foto de perfil, actividades posteadas y ajustes.

04 RF04 - Agregar amigos:

Como usuario, quiero poder ver una lista de mis amigos en la aplicación y tener la opción de agregar nuevos amigos utilizando su nombre de usuario.

05 RF05 - Visualización de entrenamientos de amigos:

Como usuario, deseo poder ver los últimos entrenamientos realizados por mis amigos en la página principal de la aplicación, para mantenerme motivado e inspirado en mi propio entrenamiento.



HISTORIAS DE USUARIO

06 RF06 - Creción de entrenamientos personalizados

Como usuario registrado, quiero poder compartir mis entrenamientos con mis amigos en la plataforma, incluyendo detalles como la duración, la distancia, las calorías quemadas y cualquier comentario adicional que desee agregar desde cualquier lugar de la aplicación.

02 RF07 - Actualización de información de entrenamientos:

Como usuario registrado, quiero poder gestionar mis actividades, editar el tipo, título, la duración, el lugar, los amigos y las imágenes. Además de eliminarla.

03 RF08 - Visualizar otros perfiles

Como usuario registrado, quiero poder ver el perfil de otros usuarios además de mis amigos.

04 RF09 - Cambiar tema de color

Como usuario, quiero poder cambiar el tema de color de mi aplicación.

HISTORIAS DE USUARIO

05 RF10 - Salir de la aplicación

Como usuario registrado quiero tener un boton de logout.

6 RF11 - Dashboard

Como administrador, quiero poder tener un dashboard con gráficas de la aplicación relevantes, como numero de usuarios por edad y el número de actividades a lo largo de tiempo.

7 RF12 - Administración de usuarios:

Como administrador, quiero poder gestionar a todos los usuario ver su información y poder actualizarla.

8 RF13 - Administración de entrenamientos:

Como administrador, quiero poder gestionar a todas las actividades ver su información y poder actualizarla.

9 RF14 - Moderación de contenidos:

Como administrador, quiero poder enviar notificaciones y comunicaciones a los usuarios de la aplicación, como anuncios importantes, actualizaciones de funciones o recordatorios de eventos, para mantenerlos informados y comprometidos con la plataforma.

HISTORIAS DE USUARIO

9 RF15 - Gestión de notificaciones y comunicaciones:

Como administrador, quiero poder enviar notificaciones y comunicaciones a los usuarios de la aplicación, como anuncios importantes, actualizaciones de funciones o recordatorios de eventos, para mantenerlos informados y comprometidos con la plataforma.

6 RF16 - Gestión de reportes:

Como administrador, deseo poder revisar y responder a los reportes de usuarios sobre contenido inapropiado o comportamiento abusivo en la plataforma, tomando las medidas necesarias para abordar estos problemas de manera oportuna.

7 RF17 - Herramientas de seguridad:

Como administrador, quiero tener acceso a herramientas y funcionalidades de seguridad adicionales para proteger la integridad de la plataforma y prevenir actividades maliciosas, como ataques de phishing o hacking.

8 RF18 - Configuración del sistema:

Como administrador, deseo tener la capacidad de configurar y personalizar diferentes aspectos de la aplicación, como ajustes de privacidad, políticas de uso y opciones de monetización, para adaptar la plataforma a las necesidades específicas de la comunidad y mantenerla actualizada

STACK TECNOLÓGICO

BACKEND

El stack tecnológico del proyecto se divide en dos grandes componentes: el backend construido con Spring Boot y el frontend desarrollado con React. A continuación, se detallan las razones para la elección de estas tecnologías y cómo se complementan para crear una aplicación robusta y eficiente.

Backend: Spring Boot

1. Spring Boot 3.2.3:

- **Simplicidad y Rapidez:** Spring Boot facilita la creación de aplicaciones stand-alone, production-grade con el menor esfuerzo de configuración. Proporciona configuraciones predeterminadas que permiten empezar rápidamente, lo cual es ideal para proyectos que requieren un desarrollo ágil.
- **Ecosistema Extenso:** La compatibilidad con una amplia gama de bibliotecas y proyectos del ecosistema Spring permite integrar fácilmente funcionalidades adicionales sin necesidad de configuraciones complejas.

2. Persistencia de Datos:

- **spring-boot-starter-data-jpa:** Utiliza JPA para manejar la persistencia de datos de una manera orientada a objetos. Simplifica las operaciones CRUD y la gestión de transacciones, lo que mejora la productividad del desarrollador.
- **MySQL con mysql-connector-java:** MySQL es una base de datos relacional robusta y ampliamente utilizada, ideal para manejar grandes volúmenes de datos con alta eficiencia y confiabilidad.

STACK TECNOLÓGICO**BACKEND****3. Seguridad:**

- **spring-boot-starter-security:** Proporciona un marco integral para manejar la seguridad de la aplicación, incluyendo autenticación y autorización. Es altamente configurable y extensible, lo que permite implementar políticas de seguridad detalladas.
- **JSON Web Token (JWT):** JWT es una solución popular para la autenticación basada en tokens, adecuada para aplicaciones web modernas. Permite la transmisión segura de información entre las partes y facilita la implementación de autenticación stateless.

4. Servicios Web RESTful:

- **spring-boot-starter-web:** Facilita la creación de servicios web RESTful. Proporciona una configuración predeterminada para construir y consumir API REST, que son fundamentales para la comunicación entre el frontend y el backend.

5. Herramientas de Desarrollo:

- **spring-boot-devtools:** Mejora la productividad al proporcionar funcionalidades como el reinicio automático y la recarga en caliente durante el desarrollo.

6. Utilidades y Otras Herramientas:

- **Apache Commons Lang:** Proporciona utilidades adicionales que complementan el lenguaje Java, facilitando operaciones comunes y mejorando la eficiencia del código.
- **Lombok:** Reduce el código boilerplate mediante anotaciones, lo que simplifica la escritura y el mantenimiento del código.

STACK TECNOLÓGICO**BACKEND**

- Java Faker: Útil para generar datos falsos en las pruebas, lo que mejora la robustez y la cobertura de las pruebas.
- Hibernate Validator: Implementa la especificación de Bean Validation, asegurando la integridad y validación de datos a nivel de objeto.

7. Documentación de la API:

- SpringDoc OpenAPI y Springfox Swagger: Facilitan la generación automática de la documentación de la API, mejorando la transparencia y la facilidad de uso de las API por parte de otros desarrolladores.

8. Pruebas:

- Spring Boot Test y otras herramientas de prueba: Proporcionan un marco completo para realizar pruebas unitarias e integradas, asegurando la calidad y la funcionalidad del código antes de su despliegue.

9. Empaquetado:

- spring-boot-maven-plugin: Permite empaquetar la aplicación Spring Boot en un JAR ejecutable, facilitando el despliegue y la distribución.

STACK TECNOLÓGICO**FRONTED****Frontend: React****1. React 18.3.1 y ReactDOM 18.3.1:**

- **Interactividad y Reactividad:** React es una biblioteca popular para construir interfaces de usuario dinámicas y reactivas. Permite crear componentes reutilizables que facilitan el desarrollo y el mantenimiento de la UI.
- **Virtual DOM:** Mejora el rendimiento de la aplicación mediante la actualización eficiente del DOM, lo que es esencial para aplicaciones con interfaces de usuario complejas y altamente interactivas.

2. Estilo y Animaciones:

- **@fontsource/roboto, framer-motion y sass:** Estas librerías permiten estilizar la aplicación y agregar animaciones de manera efectiva, mejorando la experiencia del usuario. SASS facilita la escritura de CSS modular y mantenible.

3. Manejo de Rutas:

- **react-router-dom:** Proporciona una solución robusta para manejar la navegación y las rutas dentro de la aplicación React, permitiendo la creación de aplicaciones de una sola página (SPA) con múltiples vistas.

4. Manejo de Estado y Propiedades:

- **prop-types:** Facilita la validación de propiedades en los componentes, mejorando la robustez y la fiabilidad del código.

5. Solicitudes HTTP:

- **axios y react-axios:** Facilitan la realización de solicitudes HTTP, permitiendo la comunicación entre el frontend y el backend. Axios es conocido por su simplicidad y eficiencia.

6. Componentes y Utilidades Adicionales:

- **react-md, react-modal y react-tooltip:** Proporcionan componentes adicionales y utilidades que mejoran la funcionalidad y la experiencia del usuario en la aplicación React.

**STACK TECNOLÓGICO****FRONTED****7. Pruebas:**

- `@testing-library` y `jest`: Proporcionan un conjunto de herramientas para realizar pruebas unitarias y de integración en componentes React, asegurando la calidad y funcionalidad del código.

8. Herramientas de Desarrollo:

- `react-scripts` y `@babel/plugin-transform-private-property-in-object`: Facilitan el desarrollo y la construcción de la aplicación, permitiendo utilizar características modernas de JavaScript y configuraciones predeterminadas optimizadas.

9. Monitoreo de Rendimiento:

- `web-vitals`: Permite medir y monitorear métricas de rendimiento web, asegurando una experiencia de usuario óptima.

10. Configuración y Scripts:

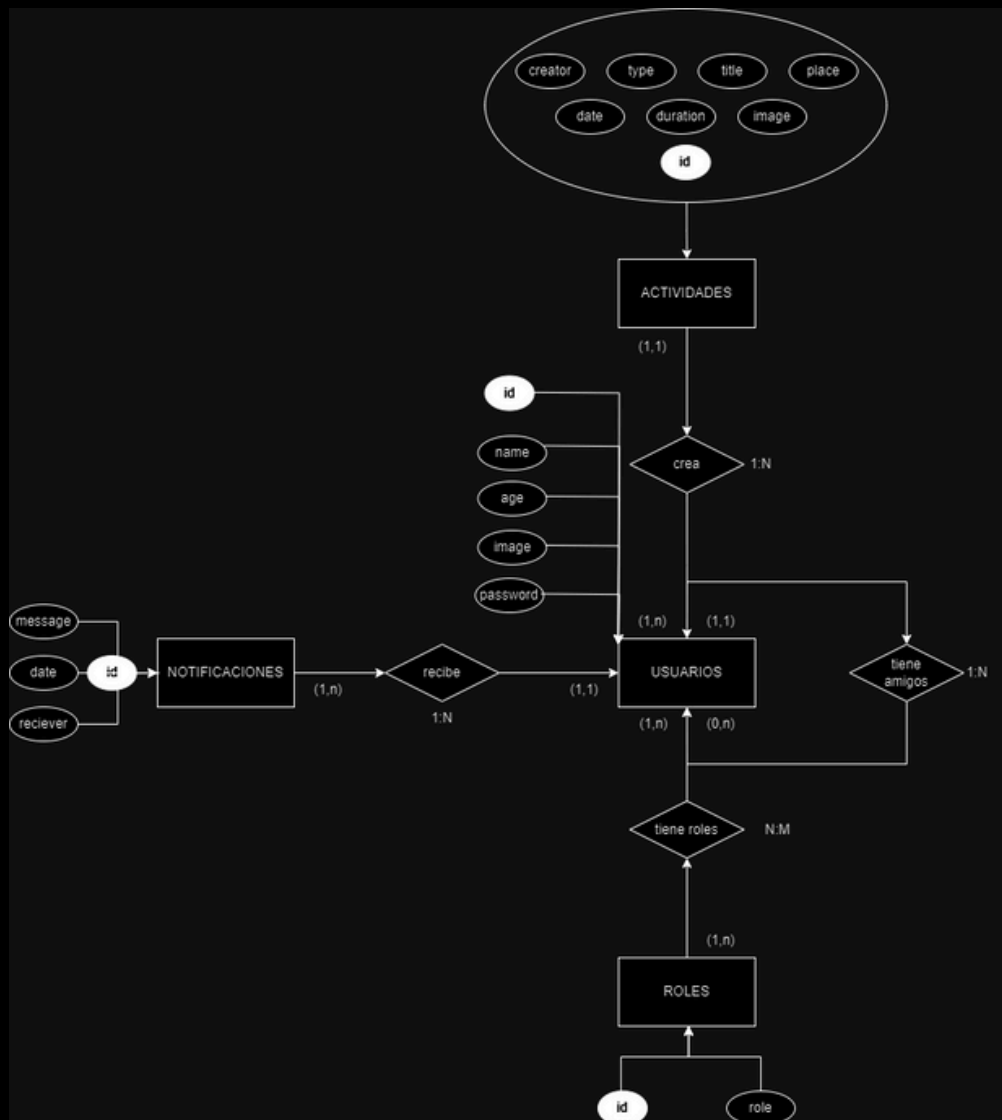
- `Scripts` de NPM y configuración de `ESLint` y `Browserslist`: Proporcionan comandos y configuraciones que facilitan el desarrollo, la prueba y el despliegue de la aplicación, asegurando la compatibilidad y el rendimiento en diferentes entornos y navegadores.

Conclusión

El stack tecnológico seleccionado para este proyecto combina lo mejor de dos mundos: la robustez y escalabilidad del backend de Spring Boot con la interactividad y flexibilidad del frontend de React. Esta combinación permite desarrollar una aplicación web moderna, segura, y eficiente, capaz de manejar tanto las demandas del negocio como las expectativas de los usuarios finales. Cada tecnología y dependencia ha sido elegida cuidadosamente para maximizar la productividad, la calidad del código y la experiencia del usuario.

BASE DE DATOS

DIAGRAMA EER



BASE DE DATOS**DIAGRAMA EER**

Descripción de la entidades:

- **User (Usuario)**
 - **Descripción:** La entidad User representa un usuario en el sistema.
 - **Utilidad:**
 - Almacena información personal del usuario como nombre, edad, dirección de correo electrónico, contraseña, imagen de perfil, etc.
 - Gestiona las relaciones del usuario con otras entidades, como actividades, notificaciones y roles.
 - Permite la autenticación y autorización en el sistema mediante roles y permisos asociados al usuario.
 - Facilita la gestión de relaciones entre usuarios, como la lista de amigos.
 - **Ejemplo de uso:** Registro de usuarios en una aplicación, gestión de perfiles de usuario, autenticación y control de acceso.
- **Activity (Actividad)**
 - **Descripción:** La entidad Activity representa una actividad o evento en el sistema.
 - **Utilidad:**
 - Almacena información sobre una actividad, como título, tipo, duración, lugar, fecha, imagen, etc.
 - Permite a los usuarios crear y participar en diferentes actividades.
 - Facilita la organización y programación de eventos dentro del sistema.
 - Proporciona información relevante sobre actividades creadas y participadas por los usuarios.

BASE DE DATOS

DIAGRAMA EER

- **Ejemplo de uso:** Calendario de eventos, planificación de actividades, gestión de eventos en una comunidad en línea.

- **Notification (Notificación)**

- **Descripción:** La entidad Notification representa una notificación enviada a un usuario.
- - **Utilidad:**
 - Almacena información sobre la notificación, como el mensaje, la fecha de envío, etc.
 - Permite a los usuarios recibir información importante o actualizaciones del sistema.
 - Facilita la comunicación entre el sistema y los usuarios.
 - Proporciona un medio para notificar a los usuarios sobre eventos, mensajes nuevos, cambios en el sistema, etc.
 - **Ejemplo de uso:** Notificaciones de mensajes nuevos, alertas de eventos próximos, notificaciones de cambios en la configuración de la cuenta.

BASE DE DATOS**ATRIBUTOS****User (Usuario)**

- id: Identificador único del usuario.
- name: Nombre del usuario.
- age: Edad del usuario.
- email: Dirección de correo electrónico del usuario.
- password: Contraseña del usuario.
- image: Imagen de perfil del usuario.
- roles: Conjunto de roles asignados al usuario.
- friends: Lista de amigos del usuario (relación muchos a muchos con User).
- notifications: Lista de notificaciones recibidas por el usuario (relación uno a muchos con Notification).

Activity (Actividad)

- id: Identificador único de la actividad.
- title: Título de la actividad.
- type: Tipo de la actividad.
- duration: Duración de la actividad.
- place: Lugar donde se lleva a cabo la actividad.
- date: Fecha de la actividad.
- image: Imagen asociada a la actividad.
- creator: Usuario que creó la actividad (relación muchos a uno con User).
- participants: Lista de usuarios que participan en la actividad (relación muchos a muchos con User).

Notification (Notificación)

- id: Identificador único de la notificación.
 - message: Mensaje contenido en la notificación.
 - date: Fecha de creación de la notificación.
 - receiver: Usuario que recibe la notificación (relación muchos a uno con User).
-



BASE DE DATOS

RELACIONES

User (Usuario) - Activity (Actividad)

- Tipo de relación: Uno a muchos (User crea muchas Activity; Activity tiene un creador User).
- Utilidad:
 - Permite que los usuarios creen y gestionen actividades dentro del sistema.
 - Facilita la organización y planificación de eventos por parte de los usuarios.
 - Proporciona una manera de asociar actividades específicas con los usuarios que las crearon.
- Casos de uso:
 - Varios usuarios participan en una actividad organizada por otro usuario.
 - Visualización de todas las actividades creadas por un usuario específico.

User (Usuario) - Notification (Notificación)

- Tipo de relación: Uno a muchos (User recibe muchas Notification; Notification tiene un receptor User).
 - Utilidad:
 - Permite al sistema enviar notificaciones a usuarios específicos.
 - Facilita la comunicación entre el sistema y los usuarios sobre eventos importantes o cambios relevantes.
 - Proporciona una manera de mantener a los usuarios informados sobre nuevas actividades, mensajes, actualizaciones del sistema, etc.
-

**BASE DE DATOS****RELACIONES**

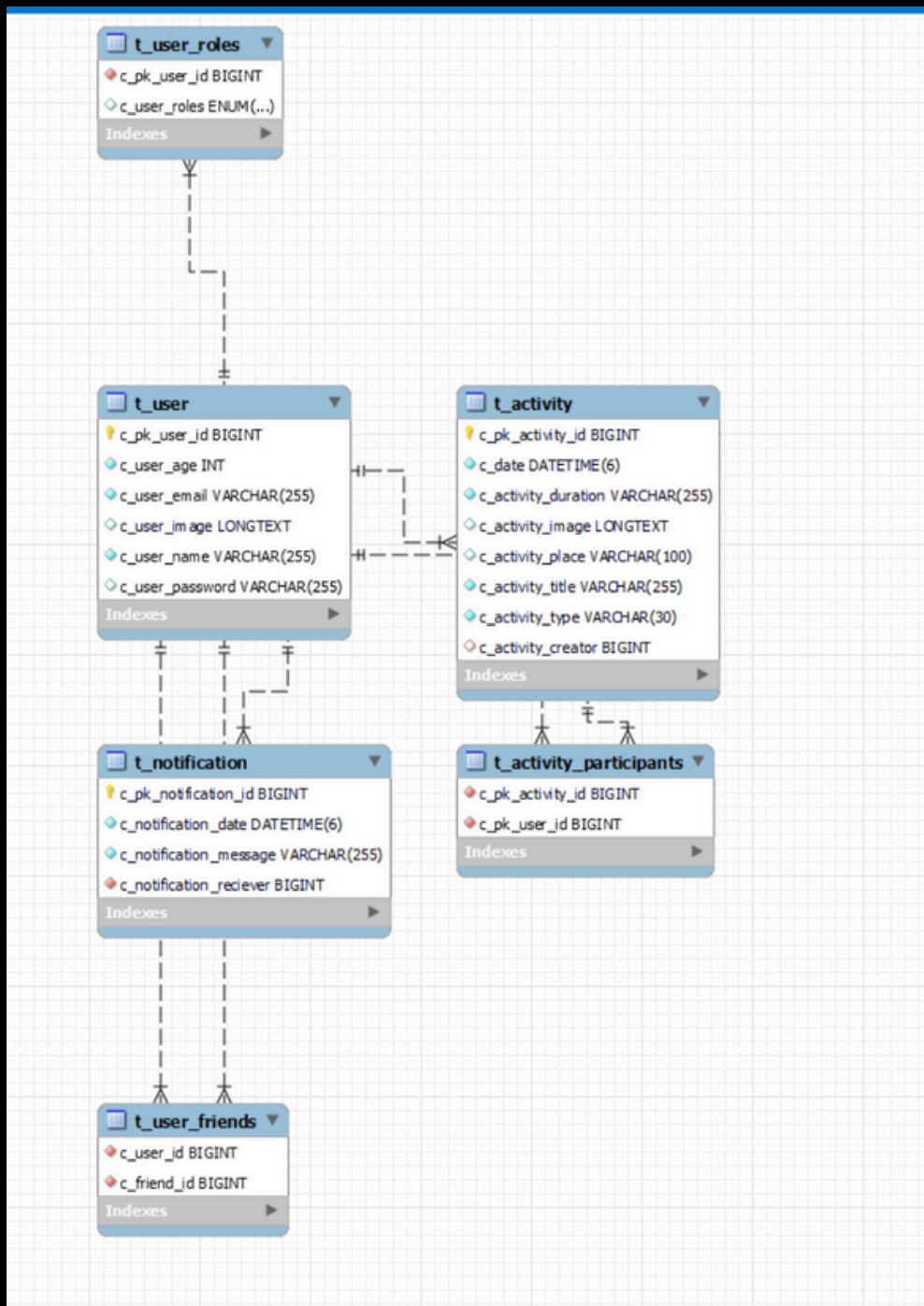
- Casos de uso:
 - Envío de notificaciones a un usuario cuando te han añadido como amigo.
 - Notificar a un usuario sobre cambios en su perfil o configuración de cuenta.

User (Usuario) - User (Usuario) (amistad)

- Tipo de relación: Muchos a muchos (User tiene muchos amigos; cada User puede ser amigo de muchos otros User).
- Utilidad:
 - Facilita la gestión de relaciones sociales entre los usuarios.
 - Permite a los usuarios seguir las actividades y actualizaciones de sus amigos.
 - Proporciona una forma de establecer conexiones entre usuarios dentro del sistema.
- Casos de uso:
 - Un usuario agrega a otro usuario como amigo para seguir sus actividades y actualizaciones.
 - Visualización de la lista de amigos de un usuario y sus actividades recientes.
 - Notificación a un usuario cuando un amigo realiza una nueva actividad o actualiza su perfil.

BASE DE DATOS

MODELO FÍSICO

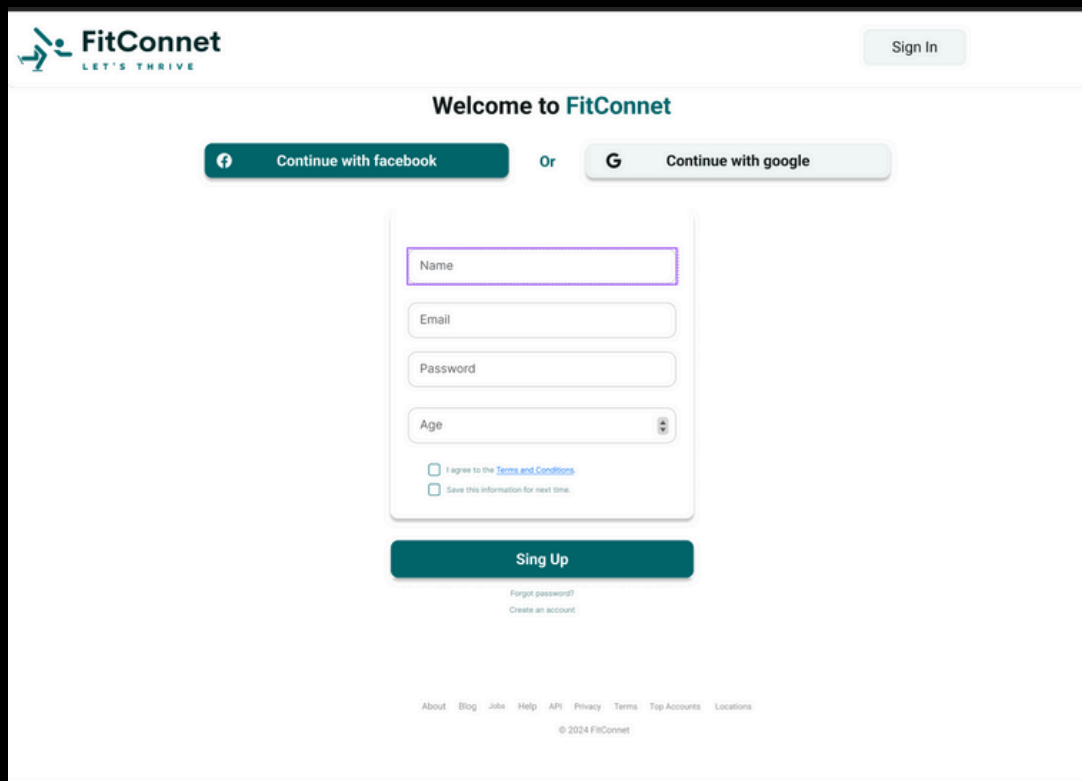


PROTOTIPO DE FITCONNET

1. Autenticación login y registro:

En estas páginas el usuario puede bien iniciar sesión o registrarse.


Ambas pantallas incluyen un formulario de registro para nuevos usuarios y otro de inicio de sesión para usuarios ya existentes.



The image shows a web form for FitConnet. At the top left is the logo "FitConnet LET'S THRIVE". At the top right is a "Sign In" button. Below the logo is the text "Welcome to FitConnet". Underneath are two buttons: "Continue with facebook" and "Continue with google", separated by "Or". Below these is a registration form with fields for "Name", "Email", "Password", and "Age". There are two checkboxes: "I agree to the Terms and Conditions" and "Save this information for next time". Below the form is a "Sing Up" button. At the bottom of the form are links for "Forgot password?" and "Create an account". At the very bottom of the page are links for "About", "Blog", "Jobs", "Help", "API", "Privacy", "Terms", "Top Accounts", and "Locations", followed by the copyright notice "© 2024 FitConnet".



PROTOTIPO DE FITCONNET



Sign Up

Welcome back to FitConnet

@Username

Password

☐ Save this information for next time.

Sign In

Forgot password?

Create an account?

AboutBlogJobsHelpAPIPrivacyTermsTop AccountsLocations

© 2024 FitConnet

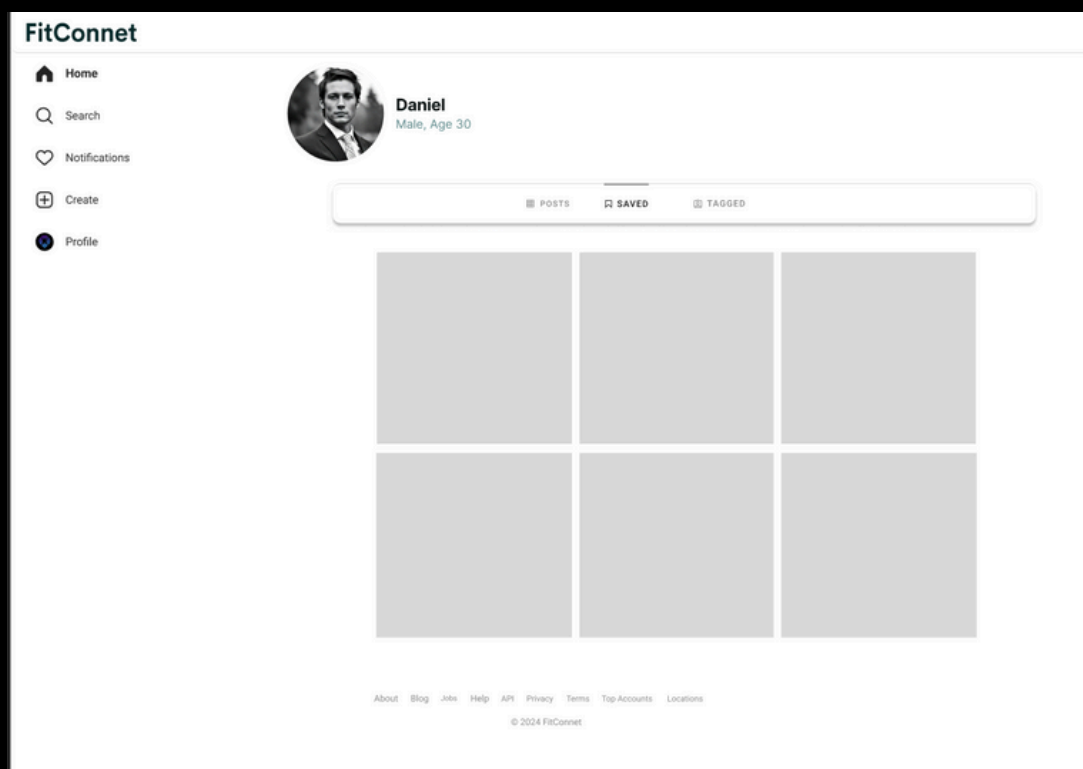
PROTOTIPO DE FITCONNET

1. Perfil de usuario

En estas página el usuario tienen la opción de ver los detalles de su perfil como su edad, actividades creadas, y aquellas en las que ha participado.

Además de esto, se incluye un sidebar que acompaña al usuario en todas las pantallas para poder acceder a ellas que son las siguientes:

- Home: pantalla principal donde se pueden ver las publicaciones de tus amigos.
- Search: modal que se abre, donde puedes buscar otros usuarios y añadirlos como amigo.
- Create: modal que se abre con un formulario para crear una actividad.
- Profile: botón que redirige al perfil del usuario autenticado.

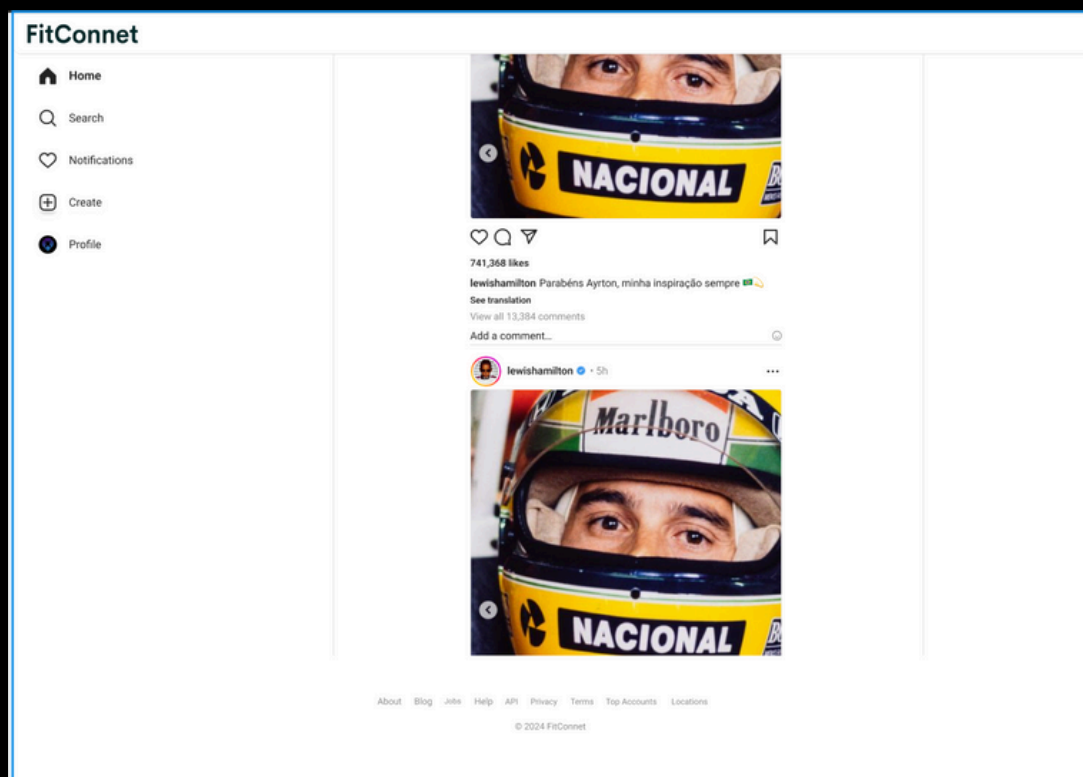


PROTOTIPO DE FITCONNET

1. Página principal (Home)

En esta página el usuario tiene la posibilidad de ver las últimas publicaciones que han subido tanto el usuario que está registrado como de las de los amigos.

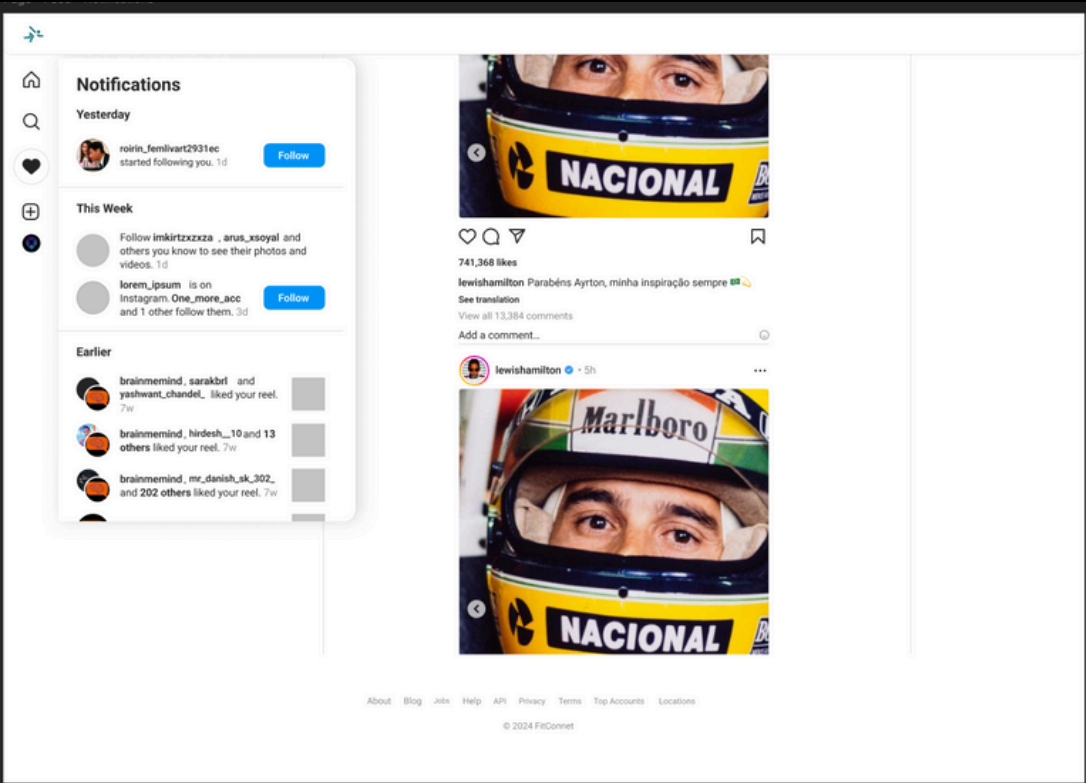
Funcionalidades: Ver publicaciones, buscar y añadir nuevos amigos, crear nuevas actividades e ir al perfil del usuario.



PROTOTIPO DE FITCONNET

1. Panel de notificaciones

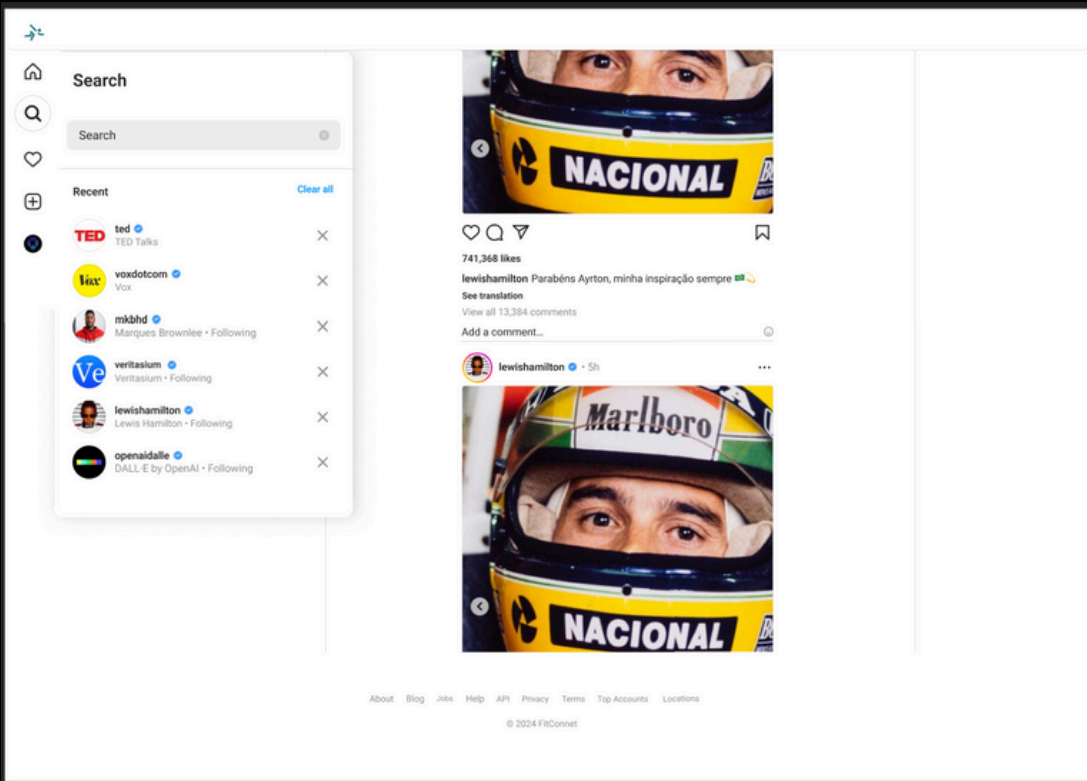
En estas página el usuario poder ver las últimas notificaciones que le han llegado, como que un amigo le ha seguido o ha sido etiquetado en un entrenamiento.



PROTOTIPO DE FITCONNET

1. Panel de búsqueda

En estas página el usuario poder buscar usuario para añadirlos como amigos y/o ver sus perfiles.

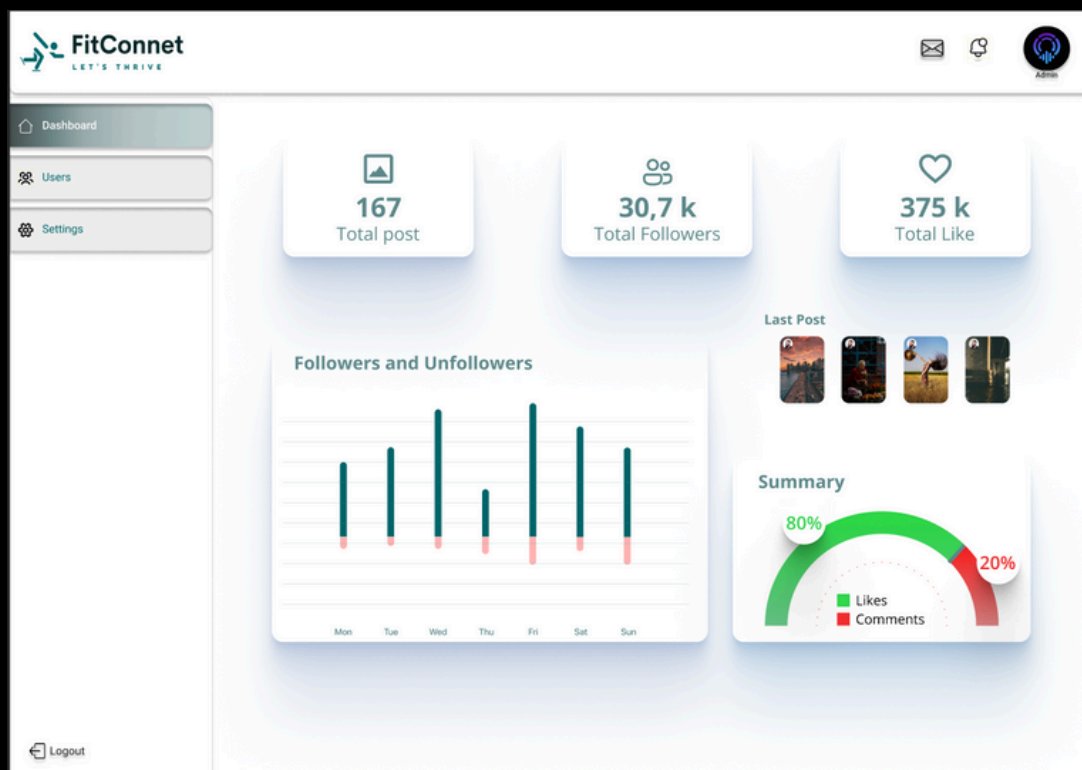


PROTOTIPO DE FITCONNET

1. Página del administrador (Dashboard)

En esta página el administrador puede ver información relevante del total de usuarios que hay registrados, el número de entrenamientos subidos y algunas métricas.

En el panel lateral tendrá un acceso para administrar los datos de usuario y otro para administrar los ajustes de la aplicación, además de un botón logout para salir de la aplicación.



MANUAL DE DESPLIEGUE

Manual de Despliegue de Servicios con Docker Compose

Este manual de despliegue detalla los pasos necesarios para desplegar los servicios especificados en el archivo docker-compose.yml proporcionado. El archivo define tres servicios: backend, frontend y base de datos (db).

Prerrequisitos:

Antes de comenzar, asegúrate de tener instalados los siguientes componentes en tu sistema:

1. Node: Puedes descargarlo e instalarlo desde nodejs.org.
2. Git: Puedes descargarlo e instalarlo desde git-scm.com.
3. Java 21: Puedes descargarlo e instalarlo desde oracle.com.
4. Maven: Puedes descargarlo e instalarlo desde maven.apache.org.
5. Docker: Puedes descargarlo e instalarlo desde docker.com.
6. Docker Compose: Normalmente se incluye con Docker Desktop, pero puedes verificar la instalación ejecutando `docker-compose --version`.

Estructura del Proyecto:

```
/src
├── docker-compose.yml
├── src-api
│   └── Dockerfile
└── src-frontend
    └── Dockerfile
```

Guía para construir y desplegar los servicios

1. Clonar el Repositorio

Clona el repositorio que contiene el archivo docker-compose.yml y los directorios src-api y src-frontend.

```
git clone https://github.com/Dani-Ps/fitconnect_final_project.git
cd fitconnect_final_project
```

MANUAL DE DESPLIEGUE

2. Construir y Desplegar los Servicios

Antes de ejecutar el siguiente comando para construir y desplegar los servicios definidos en el archivo docker-compose.yml, asegúrate de seguir estos pasos:

- Paso 1: Construir el Backend

En el directorio `fitconnect_final_project/src/src-api`, ejecuta en la consola `mvn clean install` para construir el archivo `.jar`:

```
cd fitconnect_final_project/src/src-api
mvn clean install
```

- Paso 2: Instalar Dependencias del Frontend

En el directorio `fitconnect_final_project/src/src-fronted`, ejecuta en la consola `yarn install` para instalar las dependencias del proyecto:

```
cd fitconnect_final_project/src/src-fronted
yarn install
```

- Paso 3: Compilar el Frontend

Una vez instaladas las dependencias, compila el frontend con Yarn en la consola:

```
yarn build
```

- Paso 4: Construir y Desplegar los Servicio

Finalmente, ejecuta el siguiente comando en el directorio raíz del proyecto para construir y desplegar los servicios definidos en el archivo `docker-compose.yml`:

```
docker-compose up --build
```

3. Verificación

- Backend: Accede a `http://localhost:8080` para verificar que el backend está funcionando.
- Frontend: Accede a `http://localhost:3000` para verificar que el frontend está funcionando.
- DB: El servicio MySQL estará disponible en `localhost:3306`.

ENLACE AL REPOSITORIO

- Repositorio: https://github.com/Dani-Ps/proyecto_final_iesalixar
- Figma:
<https://www.figma.com/proto/N6xj7HhUhvxwSGRsxlWCl/UI?node-id=0-1&t=d4QL7MtHIM3FmVMS-1>
- Documentación de la API:
<https://documenter.getpostman.com/view/34870994/2sA3JNa15u>



FIN