MEMORIA FINAL DE PROYECTO

GetFit

GetFit S.A

CICLO FORMATIVO DE GRADO SUPERIOR

DAW

AUTORES

Raúl del Arco Lombardo

David Ricote Simal

TUTOR

Roberto González Álvarez

COORDINADOR

Guillermo Sanz Herranz

Licencia

Esta obra está bajo una licencia Reconocimiento-Compartir bajo la misma licencia 3.0 España de Creative Commons. Para ver una copia de esta licencia, visite http://creativecommons.org/licenses/by-sa/3.0/es/ o envíe una carta a Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA.

ÍNDICE

1. INTRODUCCIÓN 5
2. ALCANCE DEL PROYECTO Y ANÁLISIS PREVIO 5
3. ESTUDIO DE VIABILIDAD 6
   3.1. Estado actual del sistema 7
   3.2. Resumen de requisitos del cliente 7
   3.3. Posibles soluciones 7
   3.4. Solución elegida 8
   3.5. Planificación temporal de las tareas del proyecto "*GetFit*" 8
   3.6. Planificación de los recursos a utilizar 9
4. ANÁLISIS 9
   4.1. Requisitos Funcionales /No Funcionales 9
   4.2. Diagrama de casos de uso 11
   4.3. Modelo de datos 12
5. DISEÑO 13
   5.1. Estructura de la aplicación 13
   5.2. Componentes del sistema / arquitectura de red 13
   5.3. Herramientas y tecnologías utilizadas 14
6. IMPLEMENTACIÓN 16
   6.1. Implementación del modelo de datos 16
   6.2. Carga de datos 17
   6.3. Configuraciones realizadas en el sistema 20
   6.4. Implementaciones de código realizadas 22
7. PRUEBAS 23
   7.1. Casos de pruebas 23
8. EXPLOTACIÓN 25
   8.1. Planificación 25
   8.2. Preparación para el cambio 26
   8.3. Implantación propiamente dicha 27
   8.4. Pruebas de implantación 28
9. DEFINICIÓN DE PROCEDIMIENTOS DE CONTROL Y EVALUACIÓN 30
10. CONCLUSIONES 30
11. FUENTES 31
   11.1. Legislación 31

11.2. Bibliografía 31

12. ANEXOS 32

1. INTRODUCCIÓN

Este documento recoge el trabajo realizado para el Módulo de Proyecto del CFGS en Desarrollo de Aplicaciones Web.

Este módulo profesional complementa la formación establecida para el resto de los módulos profesionales que integran el título en las funciones de análisis del contexto, diseño o desarrollo del proyecto y organización de la ejecución.

Los apartados que figuran en esta plantilla de memoria se adecuarán al tipo de proyecto escogido (investigación bibliográfica, gestión o experimental) y a las características particulares del mismo. El texto se organizará en función de las fases, tareas o actividades previstas en el anteproyecto o en la definición del alcance del proyecto.

2. ALCANCE DEL PROYECTO Y ANÁLISIS PREVIO

El proyecto "*GetFit*" tiene como objetivo desarrollar una aplicación web para gestionar y administrar un gimnasio. Esta aplicación proporcionará a los usuarios una plataforma intuitiva y funcional para acceder a información sobre el gimnasio, rutinas de entrenamiento y ejercicios.

El objetivo general del proyecto es crear un sistema integral que permita a los usuarios tener una experiencia completa en el gimnasio, brindando información detallada sobre las instalaciones y servicios disponibles, así como ofrecer rutinas de entrenamiento personalizadas.

El proyecto se ha llevado a cabo en las siguientes fases:

- **Estudio de viabilidad:** En esta fase, se planteó la creación de una aplicación web de gestión y administración de un gimnasio comercial como proyecto. Se evaluó la viabilidad de esta idea teniendo en cuenta el tiempo disponible y nuestros conocimientos.

- **Análisis:** En el inicio del desarrollo, se recopilaron ideas de otras páginas web de gimnasios en la comunidad de Madrid y de proyectos similares realizados por otras personas en el pasado. Con base en esta investigación, se comenzó el desarrollo del proyecto.

- **Diseño:** Durante esta fase, se llevaron a cabo diversas actividades para definir la arquitectura y estructura de la aplicación. Se diseñó la arquitectura de la base de datos, se seleccionaron las herramientas y tecnologías para el *backend* y se crearon esquemas y bocetos ilustrados del *frontend.*

- **Implementación:** En esta etapa, se desarrolló la aplicación web utilizando las tecnologías y herramientas seleccionadas. Se configuró el entorno de desarrollo utilizando *Spring Tool Suite 4* con *Eclipse* para el *backend* y se utilizó *Visual Studio Code* para desarrollar en Angular en el *frontend*.

  Se realizaron las tareas necesarias para construir el *backend*, incluyendo la importación de dependencias y la creación de la estructura de datos. Se modificaron los controladores para utilizar *@RestControllers* y se implementó la integración con *Angular* para el *frontend.*
  Una vez completados los cambios en el *backend*, se generaron los componentes necesarios para el *frontend* y se estructuró la información. Se realizaron pruebas de vistas en *Angular* y se estableció la conexión con *Spring* mediante una *API RESTful*. Se llevaron a cabo pruebas de integración para garantizar el correcto funcionamiento de los componentes y el intercambio de datos.

- **Pruebas:** Se realizaron pruebas en el *backend* para verificar el correcto funcionamiento de los controladores y la interacción con la base de datos. Se comprobó que las operaciones de creación, lectura, actualización y eliminación (*CRUD*) funcionaran correctamente.

  También se realizaron pruebas de integración para garantizar la comunicación adecuada entre los diferentes componentes del *backend*. Se verificó el manejo correcto de las relaciones entre entidades y el funcionamiento sin errores de las operaciones relacionadas. Se comprobó la correcta integración de la seguridad y la autenticación.

  En el *frontend*, se llevaron a cabo pruebas para verificar la correcta visualización de la interfaz de usuario y la fluidez de las interacciones con los usuarios. Se verificó que la navegación entre pantallas fuera intuitiva y que todas las acciones del usuario se realizarán sin errores. Se aseguró que la interfaz fuera receptiva y se adaptara correctamente a diferentes dispositivos y tamaños de pantalla.

- **Explotación:** Una vez completada la fase de desarrollo y realizadas las pruebas necesarias, se procedió a implementar la aplicación y ponerla en funcionamiento para su uso por parte de los usuarios finales.

3. ESTUDIO DE VIABILIDAD

En esta fase se realiza un estudio de viabilidad para determinar si el proyecto se puede llevar a cabo considerando las circunstancias internas y externas de la empresa, así como los recursos disponibles. Se evalúan diferentes soluciones y se analizan los requisitos del cliente para presentar una propuesta de solución.

3.1. Estado actual del sistema

La versión actual de la aplicación cuenta con tres funciones principales. La primera función es mostrar información detallada del gimnasio y sus servicios a los usuarios nuevos. La segunda función permite a los usuarios registrados consultar sus datos y acceder a contenido exclusivo para clientes. La última función es la gestión de datos por parte del administrador de la aplicación.

3.2. Resumen de requisitos del cliente

El cliente deseaba mejorar la visibilidad de su gimnasio en la web, ya que la aplicación anterior era anticuada y no permitía la gestión de datos ni la interacción por parte de los usuarios. El cliente solicitó una página web con un diseño atractivo, pero no excesivamente colorido, que permitiera gestionar los datos integrados en la página y los datos internos de la empresa. En cuanto a la parte de usuario, se requería que los usuarios pudieran ver sus propios datos y acceder al contenido exclusivo para clientes, como rutinas de entrenamiento.

3.3. Posibles soluciones

Se evaluaron varias alternativas para el desarrollo del proyecto, entre ellas:

- Otras tecnologías de *backend*: Además de *Spring Boot*, existen otros *frameworks* y tecnologías populares para desarrollar aplicaciones web en el lado del servidor, como *Django* (con *Python*), *Ruby on Rails* (con *Ruby*), *Laravel* (con *PHP*) y *ASP.NET* (con *C#*), entre otros.
- Otras tecnologías de frontend: Junto con *Angular*, existen otros *frameworks frontend* populares, como *React.js* y *Vue.js*, que también ofrecen capacidades para desarrollar aplicaciones web interactivas y dinámicas.
- Base de datos alternativa: Aunque *MySQL* es una opción común, también se consideraron otras bases de datos como *PostgreSQL*, *Oracle* y *MongoDB*, entre otras.
- Arquitectura de la aplicación: Además de una *API RESTful*, se consideró la posibilidad de una arquitectura basada en micro servicios, donde cada componente de la aplicación se divide en servicios independientes. Otra opción sería una arquitectura monolítica, donde todos los componentes se implementan en un solo sistema.

3.4. Solución elegida

Se seleccionaron las siguientes tecnologías y soluciones:

- *Spring Boot*: Se eligió *Spring Boot* como *framework* de *backend* debido a su facilidad de configuración, amplia comunidad y soporte, así como su integración con otras herramientas y proyectos del ecosistema *Spring*, como *Spring Security* y *Spring Data JPA*.
- *Angular*: Se optó por *Angular* como *framework* de *frontend* debido a su arquitectura basada en componentes, rendimiento y velocidad, así como la disponibilidad de *Angular Material* y *Bootstrap* para agilizar el desarrollo y proporcionar una apariencia visualmente atractiva y coherente.
- *MySQL*: Se eligió *MySQL* como base de datos debido a su amplia adopción, soporte y rendimiento, así como su integración con *Spring Data JPA* para facilitar las operaciones *CRUD*.

3.5. Planificación temporal de las tareas del proyecto "*GetFit*"

La planificación de las tareas del proyecto se divide en las siguientes etapas:

1. **Análisis y planificación inicial**: Aproximadamente 1 semana, con 1 persona dedicada al análisis y planificación.
2. **Diseño de la base de datos**: Menos de 1 semana, con 1 persona encargada.
3. **Desarrollo del *backend***: Alrededor de 2 semanas, con la participación de 2 personas.
4. **Desarrollo del *frontend***: Aproximadamente 3 semanas, con 2 personas involucradas.
5. **Pruebas y depuración**: Medio semana, con 2 personas encargadas.
6. **Documentación y memoria**: Aproximadamente 2 semanas, con 1 persona dedicada.

3.6. Planificación de los recursos a utilizar

Para el desarrollo eficiente del proyecto, se considera lo siguiente:

- Contratación de personal adicional: Dependiendo del alcance y la complejidad del proyecto, podría ser necesario contratar personal adicional para cubrir tareas específicas como desarrollo, diseño, pruebas, despliegue y documentación.
- Formación en tecnologías y metodologías específicas: Si el equipo actual no tiene experiencia en las tecnologías utilizadas, se puede proporcionar formación para adquirir los conocimientos necesarios.
- Adquisición de equipos y herramientas: Es posible que se requieran equipos y herramientas adicionales, como hardware, licencias de software y servicios en la nube, para llevar a cabo el proyecto de manera eficiente.
- Establecimiento de un proceso de gestión de proyectos: Implementar una metodología de gestión de proyectos, como *Scrum* o *Kanban*, puede mejorar la organización, el seguimiento y la comunicación dentro del equipo. Esto garantiza una asignación adecuada de tareas y una mayor visibilidad del progreso.

4. ANÁLISIS

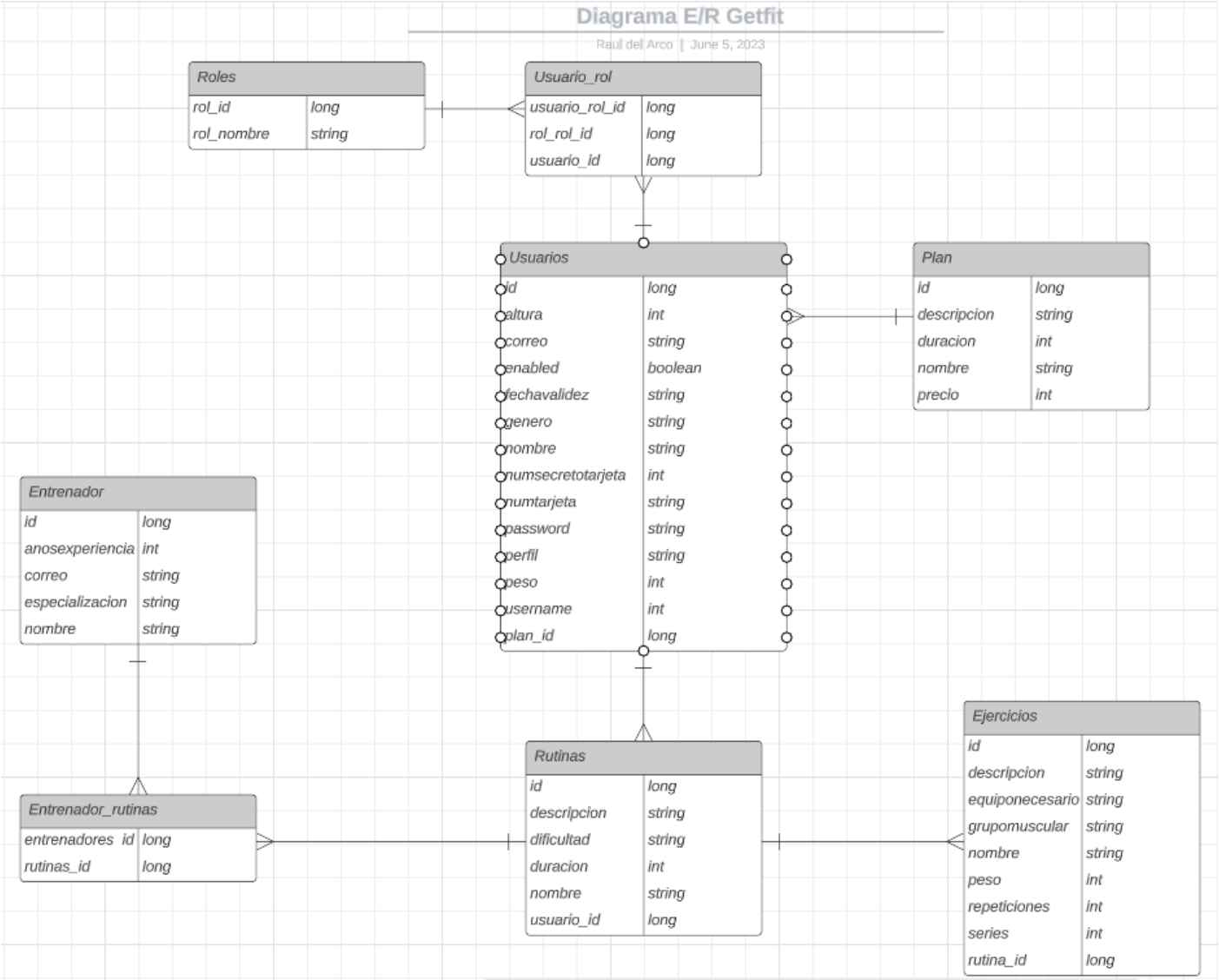
4.1. Requisitos Funcionales /No Funcionales

- **Requisitos No Funcionales**:
    - RNF 1: El sistema debe ser seguro y proteger la información personal de los usuarios.
    - RNF 2: El sistema debe ser fácil de usar y tener una interfaz intuitiva.
    - RNF 3: El sistema debe ser compatible con múltiples navegadores web modernos.
    - RNF 4: El sistema debe ser eficiente en cuanto a rendimiento y capacidad de respuesta.
    - RNF 5: El sistema debe ser escalable y capaz de manejar un número creciente de usuarios.
    - RNF 6: El sistema debe ser compatible con dispositivos móviles, permitiendo a los usuarios acceder a través de sus *smartphones*

- **Requisitos Funcionales**:
  - RF 1: El sistema debe permitir a los usuarios registrarse con su nombre de usuario, correo electrónico y contraseña.
  - RF 2: El sistema debe permitir a los usuarios iniciar sesión utilizando su nombre de usuario y contraseña.
  - RF 3: El sistema debe proporcionar a los usuarios una página de inicio donde puedan acceder a información general sobre el gimnasio.
  - RF 4: El sistema debe mostrar a los usuarios una lista de rutinas disponibles, incluyendo su descripción, dificultad y duración.
  - RF 5: El sistema debe permitir a los usuarios ver detalles de cada rutina, incluyendo los ejercicios asociados con ella.
  - RF 6: El sistema debe permitir a los usuarios ver información sobre los diferentes planes de membresía ofrecidos por el gimnasio.
  - RF 7: El sistema debe proporcionar a los usuarios una sección de "Quiénes somos" donde puedan encontrar información sobre la ubicación del gimnasio y los entrenadores.
  - RF 8: El sistema debe permitir a los usuarios actualizar su perfil, incluyendo altura, peso y género.

4.2. Diagrama de casos de uso

4.3. Modelo de datos



5. DISEÑO

5.1. Estructura de la aplicación

La aplicación "*GetFit*" sigue una arquitectura cliente-servidor y se organiza en tres capas:

1. **Capa de presentación (*frontend*):** Esta capa se encarga de la interfaz de usuario con la que los usuarios interactúan. Se utiliza *Angular* como *framework* de desarrollo del *frontend*. Angular permite crear una interfaz de usuario dinámica, interactiva y receptiva, facilitando la navegación del usuario a través de diferentes secciones y componentes de la aplicación.
2. **Capa de lógica de negocio (*backend*):** Esta capa contiene la lógica de negocio de la aplicación y se encarga de procesar las solicitudes del cliente, interactuar con la base de datos y proporcionar los datos solicitados. Se utiliza *Spring Boot* como *framework* de desarrollo del *backend*. *Spring Boot* permite la implementación de servicios *RESTful*, la integración con la base de datos y la gestión de la seguridad y autenticación de los usuarios.
3. **Capa de persistencia (base de datos):** Esta capa se ocupa de almacenar y gestionar los datos de la aplicación. Se utiliza una base de datos relacional, en este caso, *MySQL*. La base de datos almacena información sobre los usuarios, rutinas, planes de membresía y otros datos relevantes para el funcionamiento de la aplicación.

La comunicación entre estas capas se realiza a través de solicitudes *HTTP*. Los clientes (navegadores web) envían solicitudes al servidor web (*frontend*), que a su vez procesa estas solicitudes y las envía a la capa de lógica de negocio (*backend*) para su procesamiento. La capa de lógica de negocio accede a la base de datos (capa de persistencia) para obtener o almacenar datos necesarios y proporciona las respuestas correspondientes al cliente.

5.2. Componentes del sistema / arquitectura de red

En el proyecto "*GetFit*", los componentes del sistema y la arquitectura de red se pueden describir de la siguiente manera:

- **Base de datos (*BBDD*):** Se utiliza una base de datos relacional, en este caso, *MySQL*, para almacenar y gestionar la información. La base de datos consta de varias tablas, como "Usuarios", "Roles", "Rutinas", "Plan", "Entrenador", "Ejercicios", entre otras, para almacenar los datos relacionados con los usuarios, rutinas, planes de membresía y entrenadores.
- **Servidor web:** El servidor web, implementado con *Spring Boot*, es el componente central del sistema que maneja las solicitudes de los clientes y sirve las respuestas correspondientes.

- **Clientes:** Los clientes son los dispositivos o aplicaciones que interactúan con el servidor web para acceder a los servicios y funcionalidades del sistema. En el caso de "*GetFit*", los clientes pueden ser navegadores web en diferentes dispositivos. Utilizan una interfaz de usuario desarrollada con tecnologías web como *HTML*, *CSS* y *JavaScript*, y se comunican con el servidor web a través de solicitudes *HTTP*.
- **Dispositivos de red:** Los dispositivos de red, como enrutadores, *switches* y puntos de acceso inalámbrico, permiten la conectividad y comunicación entre los clientes y el servidor web. Estos dispositivos establecen la infraestructura de red que permite la transferencia de datos entre los clientes y el servidor.

La implementación del sistema "*GetFit*" se realiza en un entorno de aplicaciones web, donde el servidor web actúa como el punto central para procesar las solicitudes de los clientes y acceder a la base de datos para recuperar y almacenar datos. Los clientes, a través de sus dispositivos y navegadores web, interactúan con el servidor web para acceder a las diferentes funcionalidades y servicios ofrecidos por la aplicación.

5.3. Herramientas y tecnologías utilizadas

- **Plataforma:**
  - Plataforma web: La aplicación se implementa como una aplicación web accesible a través de navegadores web modernos.

- **Lenguajes de programación:**
  - *Java*: Se utiliza para desarrollar el *backend* de la aplicación utilizando el *framework Spring Boot*.

  - *TypeScript*: Se utiliza para desarrollar el *frontend* de la aplicación utilizando el *framework Angular*.

- **Servidor web:**
  - Servidor *Apache Tomcat*: Se utiliza como servidor web para implementar y ejecutar la aplicación.

- ***Frameworks* y bibliotecas:**
  - *Spring Boot*: Se utiliza como *framework* de desarrollo del *backend* para la implementación de servicios *RESTful*, gestión de la seguridad y autenticación, y acceso a la base de datos.

  - *Angular*: Se utiliza como *framework* de desarrollo del *frontend* para crear una interfaz de usuario dinámica y receptiva.

- *Bootstrap*: Se utiliza como biblioteca *CSS* para el diseño y apariencia visual de la aplicación.

- *Angular Material*: Se utiliza como biblioteca de componentes *UI* para proporcionar elementos de interfaz de usuario consistentes y modernos.

- **Base de datos:**
  - *MySQL*: Se utiliza como sistema de gestión de bases de datos relacional para almacenar y administrar los datos de la aplicación.

- **Herramientas de desarrollo y entorno de desarrollo integrado (*IDE*):**
  - *Spring Tool Suite*: Se utiliza como *IDE* principal para el desarrollo de la aplicación en *Java*.

  - *Visual Studio Code*: Se utiliza como *IDE* para el desarrollo del *frontend* utilizando *TypeScript* y *Angular*.

  - *Postman*: Se utiliza para probar y validar las *API RESTful* durante el desarrollo.

  - *Git*: Se utiliza como sistema de control de versiones para el seguimiento y control de los cambios en el código fuente.

  - *Maven*: Se utiliza como herramienta de gestión de dependencias para el proyecto *Java*.

  - *npm* (*Node Package Manager*): Se utiliza como gestor de paquetes para instalar y administrar las dependencias del proyecto *Angular*.

6. IMPLEMENTACIÓN

6.1. Implementación del modelo de datos

Durante el desarrollo de la aplicación "*GetFit*", se implementó un modelo de datos inicial que consistía en las siguientes tablas:

- **Tabla "Usuarios":** Atributos: id, altura, correo, género, nombre, *password*, peso, username.
- **Tabla "Planes": Atributos:** id, descripción, duración, nombre, precio.
- **Tabla "Entrenadores":** Atributos: id, años de experiencia, correo, especialización, nombre.
- **Tabla "Rutinas":** Atributos: id, descripción, dificultad, duración, nombre.
- **Tabla "Ejercicios":** Atributos: id, descripción, equipo necesario, grupo muscular, nombre.

Sin embargo, a medida que avanzamos en el proyecto y se identificaron nuevos requerimientos, se realizaron modificaciones y se agregaron más tablas al modelo de datos existente. Estas tablas adicionales se detallan a continuación:

- **Tabla "Roles":** Atributos: rol_id, rol_nombre.
- **Tabla "Usuario_Rol":** Atributos: usuario_rol_id, rol_rol_id, usuario_id.
- **Tabla "Entrenador_Rutinas":** Atributos: entrenadores_id, rutinas_id.

Estas tablas adicionales se implementaron para proporcionar funcionalidades adicionales en la aplicación, como la gestión de roles de usuarios y la relación entre entrenadores y rutinas.

Además, se añadieron atributos a tablas ya existentes como puede ser:

- **Tabla "Usuarios":** Atributos añadidos: enabled, fechavalidez, numsecretotarjeta, numtarjeta, perfil, plan_id.
- **Tabla "Rutinas":** Atributos añadidos: usuario_id.
- **Tabla "Ejercicios":** Atributos añadidos: peso, repeticiones, series, rutina_id.

Es importante destacar que se utilizaron tecnologías y herramientas específicas, como *Spring Boot* con *JPA* (*Java Persistence API*), para mapear el modelo de datos a la base de datos subyacente y realizar las operaciones de persistencia.

6.2. Carga de datos

Datos iniciales:

- Se utilizó *PhpMyAdmin* para importar la base de datos previamente creada. Esto incluye las tablas "Usuarios", "Planes", "Entrenadores", "Rutinas" y "Ejercicios". Los datos iniciales incluyen dos usuarios: el usuario administrador y un usuario normal llamado "raul", y otro usuario normal llamado "david".

- También se cargaron tres planes de membresía con sus respectivos detalles, como descripción, duración, nombre y precio. Se agregaron entrenadores genéricos al sistema, con información como el nombre, años de experiencia, especialización y correo electrónico. Además, se crearon varias rutinas con sus descripciones, dificultades, duraciones y nombres correspondientes.

- Para cada rutina, se agregaron ejercicios relacionados, que incluyen información como descripción, equipo necesario, grupo muscular, nombre, peso, repeticiones, series y la *ID* de la rutina a la que pertenecen.

Estos datos iniciales proporcionan una base para probar y utilizar la aplicación "*GetFit*" con usuarios, planes, entrenadores, rutinas y ejercicios predefinidos. A partir de estos datos, los usuarios pueden interactuar con la aplicación y personalizar su experiencia según sus necesidades y objetivos de fitness.

Imagen de la carga de Usuarios:

Mostrando filas 0 - 2 (total de 3, La consulta tardó 0,0007 segundos.)

```
SELECT * FROM `usuarios`
```

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]

Mostrar todo | Número de filas: 25 Filtrar filas: Buscar en esta tabla Ordenar según la clave: Ninguna

Opciones extra

	id	altura	correo	enabled	fechavalidez	genero	nombre	numsecretotarjeta	numtarjeta	password
Editar Copiar Borrar	1	0	getfit@getfit.com	1		Hombre	admin	0		$2a$10$d.QPRfSqzumsN8nHMYc9we2t1xgDTlML3dM8Yr
Editar Copiar Borrar	2	177	raul@getfit.com	1	2023-06-01	Hombre	raul	340	2354 6235 6324 5746	$2a$10$eeT9LZ8mc/m7xmWnuFQY/egaJYwQqm8WsK0iD
Editar Copiar Borrar	3	177	david@getfit.com	1	2023-06-07	Hombre	david	340	1565 1615 1151 5151	$2a$10$Bc1rOYD4QBi.coyKsuRb9OsVNqhY7HGugk0ace

Seleccionar todo *Para los elementos que están marcados:* Editar Copiar Borrar Exportar

Imagen de la carga de Rutinas:

Mostrando filas 0 - 2 (total de 3, La consulta tardó 0,0003 segundos.)

```
SELECT * FROM `rutinas`
```

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]

Mostrar todo | Número de filas: 25 Filtrar filas: Buscar en esta tabla Ordenar según la clave: Ninguna

Opciones extra

	id	descripcion	dificultad	duracion	nombre	usuario_id
Editar Copiar Borrar	8	Esta rutina se enfoca en el desarrollo de fuerza y...	Dificil	2	Fuerza y Hipertrofia (Superior)	*NULL*
Editar Copiar Borrar	9	Esta rutina se centra en mejorar la resistencia ca...	Medio	1	Resistencia Cardiovascular	*NULL*
Editar Copiar Borrar	10	Esta rutina se enfoca en mejorar la flexibidad y l...	Facil	1	Flexibilidad y Movilidad	*NULL*

Seleccionar todo *Para los elementos que están marcados:* Editar Copiar Borrar Exportar

Imagen de la carga de Ejercicios:

Mostrando filas 0 - 8 (total de 9, La consulta tardó 0,0004 segundos.)

```
SELECT * FROM `ejercicios`
```

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]

Mostrar todo | Número de filas: 25 Filtrar filas: Buscar en esta tabla Ordenar según la clave: Ninguna

Opciones extra

←T→	id	descripcion	equiponecesario	grupomuscular	nombre	peso	repeticiones	series	rutina_id
Editar Copiar Borrar	3	Acuéstate en el banco, sostén la barra con un agar...	Bancos de ejercicio	Pectorales (pecho)	Press de banca con barra	0	0	0	8
Editar Copiar Borrar	4	Agarra la barra con las palmas hacia adelante y lo...	Equipos de entrenamiento de peso corporal	Dorsales (espalda)	Dominadas	0	10	3	8
Editar Copiar Borrar	5	Sostén una mancuerna en cada mano a la altura de l...	Pesas libres	Deltoides (hombros)	Press de hombros con mancuernas	0	12	3	8
Editar Copiar Borrar	6	Ajusta la velocidad de la cinta según tu nivel de ...	Máquinas de cardio	Otro	Correr en cinta	0	0	1	9
Editar Copiar Borrar	7	Coloca los pies en los pedales y sujeta las asas. ...	Máquinas de cardio	Otro	Máquina elíptica	0	0	1	9
Editar Copiar Borrar	8	Sujeta las asas de la cuerda de saltar y salta con...	Máquinas de cardio	Otro	Saltos de cuerda	0	0	3	9
Editar Copiar Borrar	9	Siéntate en el suelo con las piernas extendidas. I...	Equipos de estiramiento y movilidad	Otro	Estiramientos de piernas	0	3	3	10
Editar Copiar Borrar	10	De pie, extiende los brazos hacia los lados. Gira ...	Equipos de estiramiento y movilidad	Otro	Rotación de hombros	0	10	2	10
Editar Copiar Borrar	11	Acuéstate boca arriba y lleva las rodillas hacia e...	Equipos de estiramiento y movilidad	Otro	Estiramiento de espalda	0	3	3	10

Imagen de la carga de Entrenadores:

Mostrando filas 0 - 2 (total de 3, La consulta tardó 0,0003 segundos.)

```
SELECT * FROM `entrenador`
```

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]

Mostrar todo | Número de filas: 25 Filtrar filas: Buscar en esta tabla Ordenar según la clave: Ninguna

Opciones extra

←T→	id	anosexperiencia	correo	especializacion	nombre
Editar Copiar Borrar	1	1	javi@getfit.com	Entrenamiento de fuerza y acondicionamiento	Javier
Editar Copiar Borrar	2	4	natalia@getfit.com	Entrenamiento deportivo	Natalia
Editar Copiar Borrar	3	6	pedro@getfit.com	Entrenamiento funcional	Pedro

Seleccionar todo *Para los elementos que están marcados:* Editar Copiar Borrar Exportar

Imagen de la carga de Planes:

6.3. Configuraciones realizadas en el sistema

Para garantizar el correcto funcionamiento de la aplicación "*GetFit*" en diferentes entornos, se realizaron configuraciones específicas en el sistema. A continuación, se detallan las configuraciones necesarias en diferentes niveles:

- **Sistema operativo:** La aplicación "*GetFit*" ha sido probada y ejecutada en diferentes sistemas operativos, como *Windows 11* y *Windows 10*. No se requieren configuraciones adicionales en el sistema operativo para el funcionamiento básico de la aplicación.

- **Requisitos de hardware:** La aplicación "*GetFit*" no es exigente en cuanto a recursos y ha sido probada en equipos con diferentes capacidades. Se ha comprobado que la aplicación puede ejecutarse en equipos con procesadores antiguos y 2 *GB* de *RAM* sin problemas significativos. Se recomienda disponer de un entorno con capacidad suficiente para ejecutar una máquina virtual Java y otras aplicaciones necesarias.

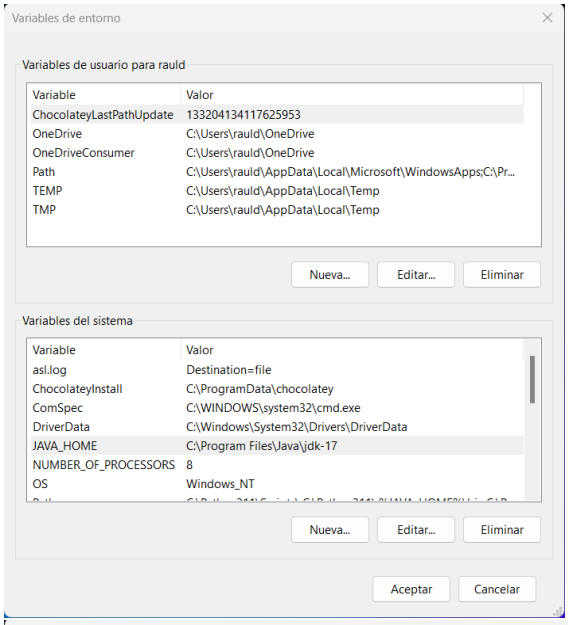
- **Máquinas virtuales:** La aplicación "*GetFit*" se desarrolló utilizando *Spring Boot* con Java, lo que implica que se requiere una máquina virtual *Java* para ejecutarla. Se recomienda tener instalada una versión de Java igual o superior a *Java 17* para asegurar la compatibilidad con la aplicación. Las máquinas virtuales Java deben estar correctamente configuradas en el sistema, con las variables de entorno adecuadas y la versión de *Java* correcta seleccionada.

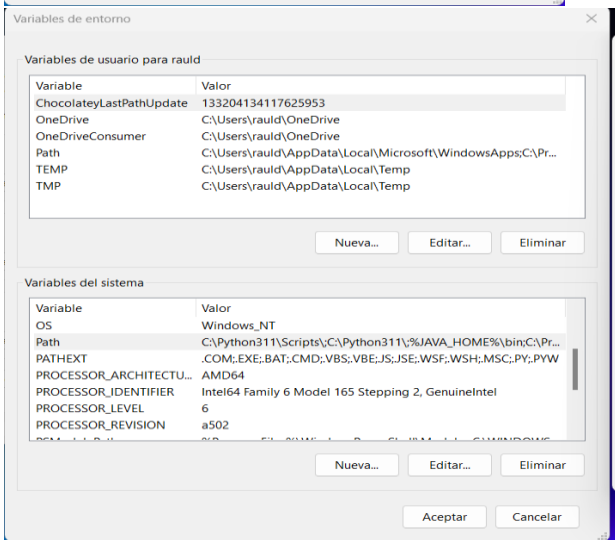
- **Ficheros de configuración:** La aplicación "*GetFit*" utiliza archivos de configuración proporcionados por *Spring Boot* y otras bibliotecas utilizadas en el proyecto. Se deben asegurar las configuraciones adecuadas en estos archivos para establecer parámetros como la conexión a la base de datos, la configuración de seguridad, entre otros. Estos archivos de configuración se encuentran en el proyecto y se deben modificar según los requisitos específicos del entorno de implementación.

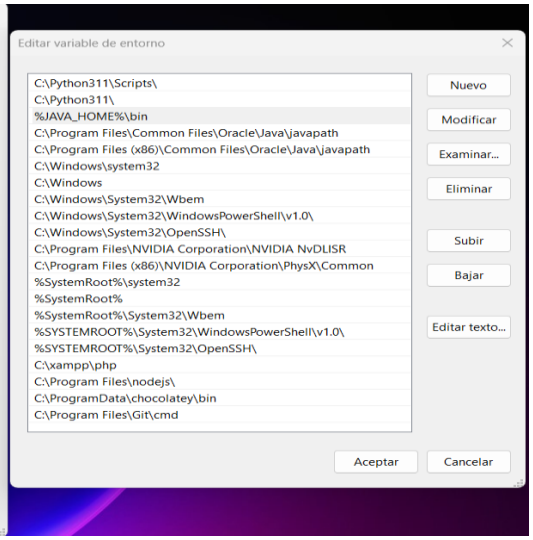
A continuación, se incluyen algunas imágenes que ilustran la configuración necesaria para el funcionamiento de la aplicación.

- **Configuración de *Java*:**

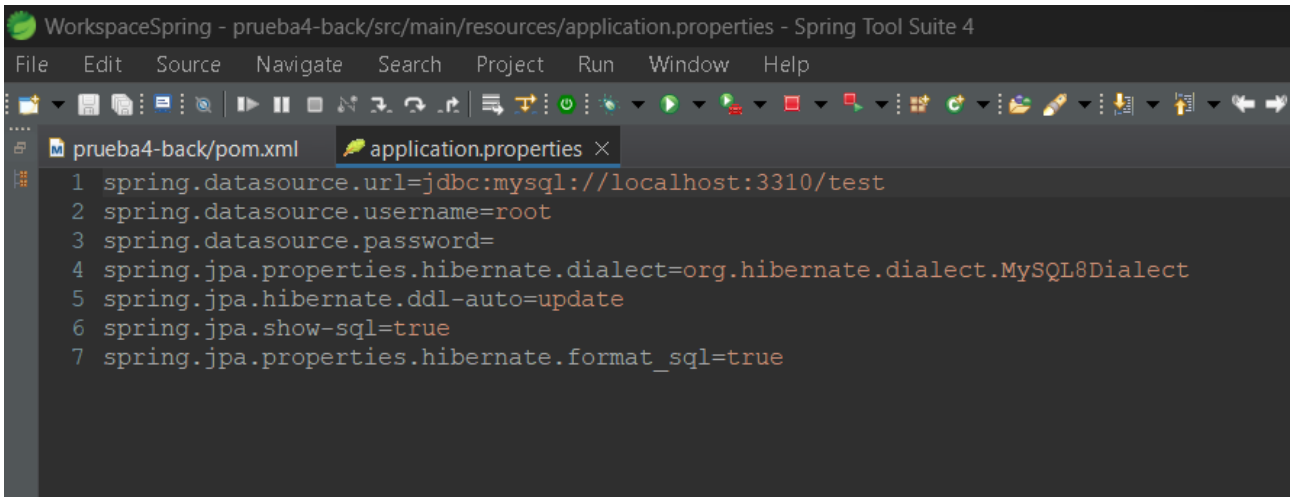
- **Variables de Entorno:**







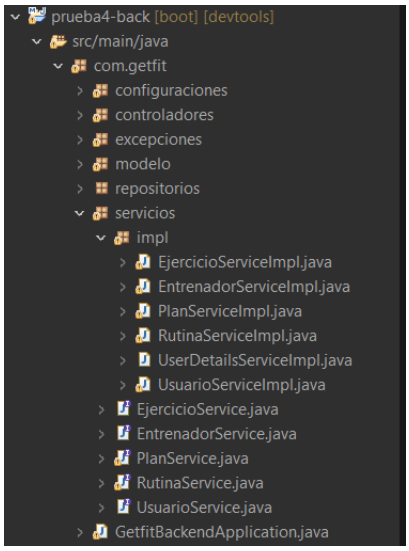
- **Configuración de *Spring*:**



6.4. Implementaciones de código realizadas

En cuanto al código de la aplicación, hay que distinguir los dos grandes bloques de código: el código del *backend* de la aplicación, que utiliza el lenguaje *Java*, y el *frontend* de la aplicación, que utiliza *HTML*, *TypeScript* y *CSS*.

1. ***Backend***: En el código del *backend*, se sigue una estructura de datos inspirada en los "PAP" proporcionados en la asignatura de "Desarrollo en Entorno Servidor", pero con ciertos cambios. Al declarar los servicios, primero se crean una interfaz donde declaramos los métodos y luego se crea la implementación de los servicios. En cuanto a las entidades, no se utiliza *Lombok*, ya que al utilizar sus anotaciones se generaban algunas estructuras de código que nos impedían realizar otras funciones, por lo que se omiten esta dependencia. En cuanto a las relaciones, se utilizan anotaciones como *fetch* para reducir el número de consultas generadas contra la base de datos. También se utilizan anotaciones como *cascade* para permitir la propagación de operaciones en las relaciones entre entidades.



   Otro aspecto destacado del código del *backend* es toda la configuración que hemos utilizado para la autenticación de usuarios y la seguridad de la aplicación. Era importante que la seguridad fuera sólida y restrictiva, por lo que se utilizó *Spring Security*. Con esta dependencia se encriptó las contraseñas de los usuarios y restringió el acceso a determinadas rutas. Además, se implementó *JWT* (*JSON Web Token*) para limitar aún más el acceso a usuarios sin privilegios.

2. ***Frontend***: En cuanto al código del *frontend*, se descartó la generación de modelos para los distintos objetos y se utilizó una estructura de componentes organizados por roles del sistema.
   Se crea una sección llamada "Servicios", donde se encuentra toda la lógica de la *API REST*, además de los servicios propios de cada entidad, para que el controlador del *backend* sea capaz de procesar los datos. Dentro de esta sección, también se encuentra toda la lógica de autenticación mediante el *token.* En cuanto al aspecto visual de los elementos de la página, se decidió implementar las librerías *Bootstrap 5* y *Angular Material,* ya que estas dos librerías proporcionan muchas estructuras y mejoran el aspecto visual de la aplicación.

7. PRUEBAS

Son muchas las pruebas que pueden realizarse en un proyecto para eliminar los posibles errores y garantizar su correcto funcionamiento. Los casos de prueba establecen las condiciones/variables que permitirán determinar si los requisitos establecidos se cumplen o no.

A continuación, se detallan algunos de los casos de prueba que se ejecutarán para comprobar la correcta construcción de este proyecto.

7.1. Casos de pruebas

- **Prueba Nº1:**
  - **Nombre del caso:** Mostrar botón de registro.
  - **Descripción:** El botón de registro en el menú solo debe de aparecer en caso de que ningún usuario haya iniciado sesión previamente. Si hay un usuario que ya ha iniciado sesión debe de aparecer un botón con su nombre en vez del botón de registro, por el cuál puede entrar en su perfil.
  - **Condiciones de ejecución:** Solo hace falta entrar en la página y comprobar el funcionamiento del botón situado en el menú.
  - **Entrada:** No es necesario.
  - **Resultado esperado:** Si hay un usuario con sesión iniciada que no aparezca el botón de registro, y en caso contrario que aparezca.
  - **Resultado obtenido:** Se muestran bien en ambos casos.

- **Prueba Nº2:**
  - **Nombre del caso:** Verificación de campos de registro.
  - **Descripción:** Para que un usuario pueda darse de alta, o a su vez el administrador pueda dar de alta un usuario, se deben de rellenar los campos correctamente. Si no se rellenan todos ellos correctamente, el botón de *submit* del formulario debe estar inhabilitado hasta que todo esté correcto. Una vez que los campos del formulario sean escritos correctamente, el botón se habilitará y se podrá guardar el usuario.
  - **Condiciones de ejecución:** En caso de un usuario, no debe de haber iniciado sesión ningún usuario, para así poder registrarse. Un administrador solo tendría que añadir usuario desde el panel de administrador.
  - **Entrada:** Para poder hacer la prueba, se deben rellenar o no los campos del formulario.
  - **Resultado esperado:** Que el botón funcione correctamente en función de haber rellenado los campos bien o no.
  - **Resultado obtenido:** El botón hace lo que se esperaba.
- **Prueba Nº3:**
  - **Nombre del caso:** Mostrar planes.
  - **Descripción:** Los planes y sus atributos se guardan en la base de datos. En la página de Planes se deben mostrar los planes guardados con un formato específico. En este caso se quiere comprobar que, si el administrador desde su panel borra, edita o crea un plan, que en la vista se actualice.
  - **Condiciones de ejecución:** Haber iniciado sesión como administrador.
  - **Entrada:** Tener ya algún plan guardado para ver las modificaciones que se vayan a hacer.
  - **Resultado esperado:** La página de planes se actualiza según se hagan cambios en la base de datos.
  - **Resultado obtenido:** Se actualizan los datos correctamente según lo esperado.

- **Prueba Nº4:**
  - **Nombre del caso:** Página *responsive*.
  - **Descripción:** Se ha adaptado todo el contenido de la página web para que se muestre en cualquier tamaño de pantalla. Con esta prueba se comprueba cómo se acopla el contenido en distintos tamaños.
  - **Condiciones de ejecución:** Ninguna.

- **Entrada:** Navegar por la página cambiando el tamaño de la pantalla.
- **Resultado esperado:** La página se adapta perfectamente a cualquier resolución.
- **Resultado obtenido:** Se han obtenido los resultados esperados, que todo que vea correctamente.

- **Prueba Nº5:**
  - **Nombre del caso:** Borrar rutinas con ejercicios.
  - **Descripción:** Las rutinas en la base de datos tienen asignada a cada una serie de ejercicios. Cuando se quiere borrar una rutina se debe de poder borrar y que a su vez se borren los ejercicios que hay dentro de ellas.
  - **Condiciones de ejecución:** Haber iniciado sesión como administrador.
  - **Entrada:** Tener rutinas creadas con ejercicios en ellas.
  - **Resultado esperado:** Se borra la rutina y se borran también los ejercicios incluidos en ellas.
  - **Resultado obtenido:** Funciona según lo esperado, borrando todo correctamente sin ningún error.

8. EXPLOTACIÓN

8.1. Planificación

❖ **Tareas a realizar:**

A. Preparación del entorno de producción: Configuración de los servidores y sistemas necesarios para alojar la aplicación web "*GetFit*".

B. Instalación de dependencias: Configuración de las bibliotecas y *frameworks* necesarios, como *Spring Boot*, *Angular Material* y *Bootstrap 5*.

C. Configuración de la base de datos: Creación de la estructura de la base de datos y carga de los datos iniciales y de prueba.

D. Despliegue de la aplicación web: Implementación de la aplicación en el entorno de producción y configuración de la infraestructura de red.

E. Pruebas de funcionamiento: Realización de pruebas exhaustivas para asegurar el correcto funcionamiento de todas las funcionalidades de la aplicación.

- **Recursos necesarios:**
  - Recursos materiales:
    - Servidores o máquinas virtuales con capacidades específicas para alojar la aplicación y la base de datos.
    - Conexión a Internet estable.
    - Licencias de software necesarias.
  - Recursos humanos:
    - Desarrolladores para configurar y desplegar la aplicación.
    - Personal técnico para la configuración y mantenimiento de la infraestructura de red.
    - Usuarios finales para probar y utilizar la aplicación.

- **Logística y tiempos de ejecución:**
  - Asignación de recursos: Planificación de la asignación de recursos materiales y humanos para cada tarea, considerando la disponibilidad y la complejidad de cada una.
  - Establecimiento de plazos: Definición de plazos realistas para cada tarea, teniendo en cuenta la complejidad y los recursos disponibles.
  - Coordinación de actividades: Comunicación efectiva entre los miembros del equipo para asegurar una ejecución fluida y coordinada de las tareas, incluyendo la coordinación con otros equipos o departamentos relevantes.

8.2. Preparación para el cambio

- **Necesidades de permisos y autorizaciones:**
  - Acceso a los sistemas: Garantizar que los usuarios y los responsables del proyecto tengan los permisos adecuados para acceder a los sistemas y recursos necesarios durante la implementación.
  - Permisos de administrador: Identificar qué usuarios necesitan permisos de administrador para llevar a cabo tareas como la configuración de la aplicación, la gestión de usuarios y la carga de datos.
  - Control de acceso: Establecer políticas y medidas de seguridad para garantizar que solo los usuarios autorizados puedan acceder a la aplicación y sus funcionalidades específicas.
- **Reticencias al cambio:**

- Comunicación efectiva: Establecer una estrategia de comunicación clara y transparente para informar a los usuarios sobre los beneficios y objetivos del proyecto "*GetFit*".
- Capacitación y formación: Proporcionar programas de capacitación y formación a los usuarios para familiarizarlos con la aplicación y ayudarles a comprender cómo su trabajo se verá afectado positivamente por el cambio.
- Participación de los usuarios: Involucrar a los usuarios en el proceso de implementación y brindarles la oportunidad de expresar sus preocupaciones y sugerencias.

- **Procedimientos de actuación o ejecución de las actividades:**
  - Documentación de procesos: Elaborar manuales o documentos de referencia que describen los procedimientos específicos a seguir para llevar a cabo actividades como la configuración de la aplicación, la carga de datos y la resolución de problemas comunes.
  - Asistencia y soporte técnico: Establecer un canal de soporte técnico para que los usuarios puedan resolver cualquier duda o problema que puedan encontrar durante el proceso de cambio.
  - Monitorización y evaluación: Establecer mecanismos para monitorear y evaluar continuamente el progreso de la implementación, identificar áreas de mejora y tomar acciones correctivas según sea necesario.

8.3. Implantación propiamente dicha

Se ha llevado a cabo la implantación del sistema de la aplicación web "*GetFit*", demostrando de manera efectiva que está en producción y listo para su uso. A continuación, se detallan las acciones que se han realizado para lograrlo:

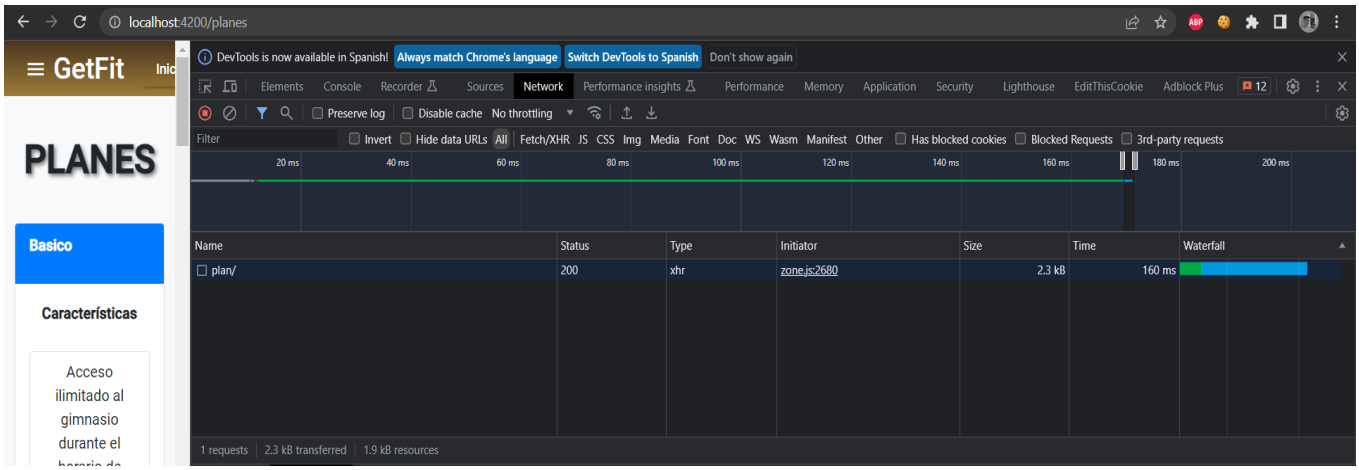
1. Creación de cuentas de usuario: Se ha implementado un formulario de registro en línea que permite a los usuarios registrarse en la aplicación. Como parte de la implantación, se han creado cuentas de usuario de prueba para demostrar la funcionalidad del sistema. Por ejemplo, se han creado cuentas ficticias como “raul" y "david".
2. Acceso a funcionalidades clave: Se han mostrado a los usuarios las funcionalidades más importantes de la aplicación web "*GetFit*" que están disponibles en el entorno de producción. Se les ha permitido crear rutinas de ejercicio personalizadas, entre otras características destacadas.
3. Carga de datos: Se ha asegurado de que los datos iniciales y de prueba se hayan cargado correctamente en la base de datos del sistema. Por ejemplo, se han incluido ejemplos de planes de membresía predefinidos, rutinas de ejercicios y datos de usuarios ficticios para mostrar el funcionamiento de la aplicación.

4. Pruebas en tiempo real: Se han realizado pruebas en vivo con usuarios reales utilizando el sistema implantado. Durante estas pruebas, se ha observado y documentado el comportamiento del sistema, identificando y solucionando posibles problemas o errores que pudieran surgir.
5. Monitoreo de la disponibilidad: Se han implementado herramientas de monitoreo para supervisar la disponibilidad del sistema de manera continua. Esto ha permitido configurar alertas en caso de caídas del servidor o tiempos de respuesta lentos, asegurando así un funcionamiento óptimo del sistema.

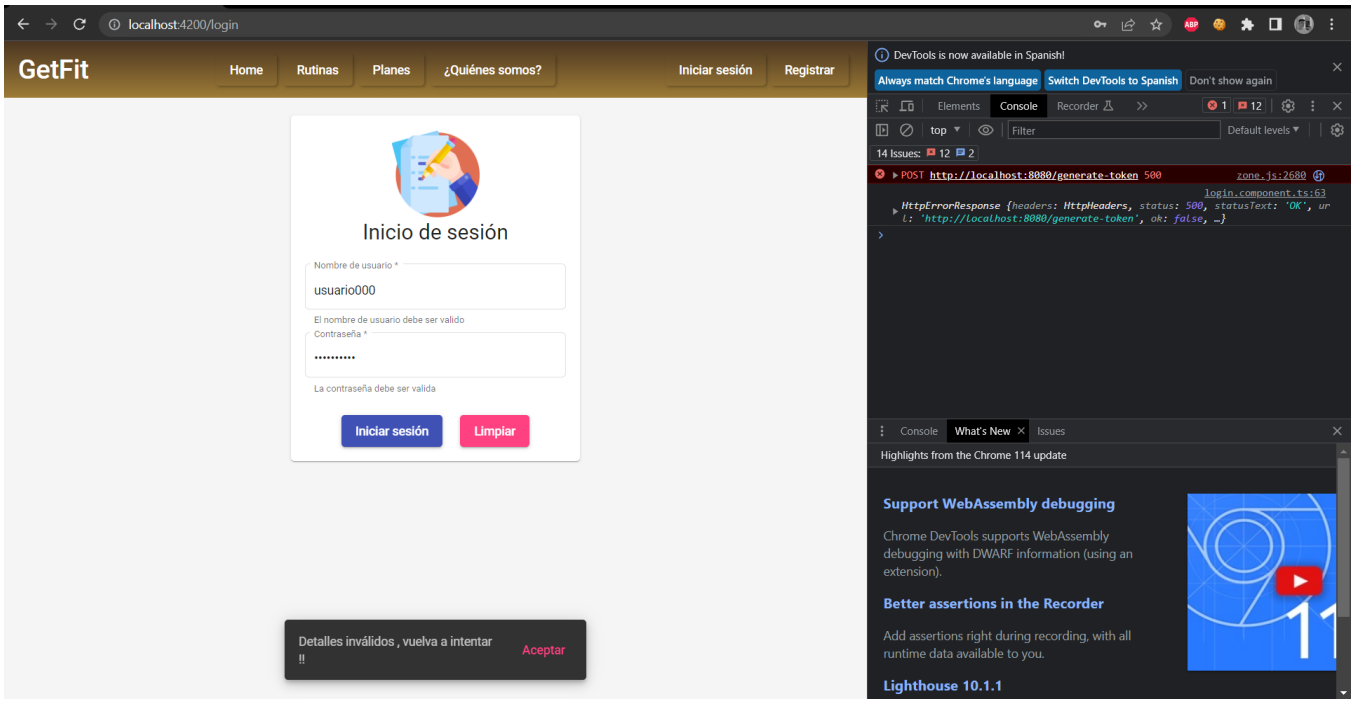
8.4. Pruebas de implantación

Una vez que el sistema ha sido implantado en el entorno del cliente, se han llevado a cabo pruebas exhaustivas para asegurarnos de que todo funcione correctamente. Estas pruebas de implantación han permitido validar el funcionamiento del sistema y verificar su adaptación a las necesidades y expectativas del cliente. A continuación, se detallan las pruebas que se han realizado:

1. **Pruebas de funcionalidad:** Se ha verificado que todas las funcionalidades de la aplicación web "*GetFit*" estén disponibles y funcionando según lo especificado en los requisitos del cliente. Se han realizado pruebas exhaustivas de cada funcionalidad para asegurarnos de que cumpla con los criterios de aceptación previamente establecidos.
2. **Pruebas de rendimiento:** Se han evaluado el rendimiento del sistema en el entorno del cliente, midiendo aspectos como el tiempo de respuesta de las páginas, la carga de datos y la capacidad de manejo de usuarios concurrentes. Estas pruebas han permitido identificar posibles cuellos de botella y optimizar el sistema para mejorar su rendimiento.



3. **Pruebas de seguridad:** Se han realizado pruebas de seguridad para garantizar que la aplicación web "*GetFit*" cumple con los estándares y requisitos de seguridad establecidos. Se ha verificado que los datos de los usuarios estén protegidos de accesos no autorizados y que no existan vulnerabilidades conocidas en la aplicación.



4. **Pruebas de usabilidad:** Se ha solicitado la participación de usuarios reales para evaluar la usabilidad de la aplicación. Mediante encuestas, entrevistas y observación directa, se ha recopilado *feedback* sobre la experiencia de uso y se ha realizado ajustes en la interfaz y la navegación para mejorar la usabilidad general.

5. **Pruebas de recuperación y contingencia:** Se han llevado a cabo pruebas de recuperación ante posibles fallos o situaciones de contingencia. Esto ha incluido simular escenarios de pérdida de conexión a Internet, caídas del servidor o errores en la base de datos para verificar la capacidad del sistema de recuperarse correctamente y mantener la integridad de los datos.

6. **Pruebas de aceptación del cliente:** Se ha involucrado al cliente en la etapa de pruebas, solicitando su participación activa para validar el sistema implantado. Se han presentado los resultados de las pruebas anteriores y hemos recopilado sus comentarios y aprobación final.

9. DEFINICIÓN DE PROCEDIMIENTOS DE CONTROL Y EVALUACIÓN

A lo largo del ciclo de vida del proyecto se producirán cambios e incidencias que deberán controlarse y registrarse.

En estos momentos no hay ninguna herramienta específica para recibir las incidencias que surjan, el cliente se pone en contacto con nosotros mediante el correo electrónico y se le intenta solucionar el problema cuando sea posible.

Durante la realización del proyecto, y a su vez las distintas versiones que se han ido publicando con el producto ya entregado, se ha utilizado la herramienta *GitHub* en el que se pueden ver todas las distintas versiones por las que ha pasado la aplicación.

10. CONCLUSIONES

Este proyecto ha estado repleto de retos y superaciones en cuanto a los conocimientos previos, comenzando con una idea con las bases bastantes sólidas, pero según se fue avanzando con el desarrollo se fueron encontrando problemas e inconvenientes que hacían retroceder más pasos de los que se avanzaban, estos problemas suponían un reto, ya que se desconocía el origen y la solución de los mismos, pero tras largo trabajo de búsqueda de información, se acabó logrando.

Uno de los grandes retos era lograr crear el *backend* de la aplicación en *Spring,* y el *frontend* en *Angular* y que se pudieran comunicar, al final tras mucho esfuerzo y mucha documentación de por medio, se logró conseguir.

Otro gran reto que se tuvo, y el que se empleó más tiempo, era poder lograr un sistema de autenticación potente y robusto, y que angular fuese capaz de tramitar los datos. Esta tarea ocupó mucho tiempo de desarrollo, ya que se trataba de una parte crucial de la aplicación.

Se comenzó el desarrollo con un montón de ideas y de funcionalidades que podrían estar bien dentro del ámbito de la aplicación, pero debido al corto plazo de entrega y los problemas en el desarrollo, se quedaron en el camino.

Habría sido interesante implantar una gestión de reservas, para poder acceder al centro, que los entrenadores tuvieran su agenda de actividades y que los usuarios puedan crear sus rutinas propias, pero no ha sido posible por el tiempo, pero no se descarta implementar estas funcionalidades en un futuro.

11. FUENTES

11.1. Legislación

Enseñanzas mínimas: Real Decreto 686/2010, de 20 de mayo (BOE 12/06/2010)

http://pdf/IFCS03/titulo/RD20100686_TS_Desarrollo_Aplicaciones_Web.pdf

Currículo: Decreto 1/2011, de 13 de enero (BOCM 31/01/2011)

http://pdf/IFCS03/curriculo/D20110001_TS_Desarrollo_Aplicaciones_Web.pdf

Definición de procedimientos de control y evaluación:

- http://www.xperta.es/es/descripcion.asp
- http://www.xperta.es/es/aquienvadirigido.asp
- http://churriwifi.wordpress.com/2010/04/10/gestion-de-incidencias/
- http://es.wikipedia.org/wiki/Control_de_versiones

11.2. Bibliografía

Angular

- https://angular.io/guide/developer-guide-overview
- https://angular.io/guide/http
- https://angular.io/guide/animations
- https://material.angular.io/components/button/overview
- https://material.angular.io/components/card/overview
- https://material.angular.io/components/datepicker/overview
- https://material.angular.io/components/grid-list/examples
- https://material.angular.io/components/icon/examples
- https://material.angular.io/components/input/overview
- https://material.angular.io/components/menu/overview
- https://material.angular.io/components/snack-bar/overview
- https://material.angular.io/components/select/overview
- https://sweetalert2.github.io/#examples

Api Rest

- https://gonzalogarciamr.com/2021/04/28/fullstack-rest-angular-spring/
- https://www.youtube.com/watch?v=siKe5cVWOcs

Spring Boot

- https://spring.io/projects/spring-security
- https://www.toptal.com/spring/spring-security-tutorial
- https://docs.spring.io/spring-security/reference/servlet/oauth2/resource-server/jwt.html
- https://github.com/bbarbs/spring-boot-jwt/blob/master/src/main/java/com/auth/core/security/JwtAuthenticationFilter.java
- https://www.baeldung.com/spring-security-granted-authority-vs-role

12. ANEXOS

En la carpeta donde se han alojado los datos del proyecto, hay también dos documentos que sirven para complementar la información referente al proyecto, dichos documentos son:

- Guia de usuario GetFit
- Guia de Puesta en Marcha

Así como también se emplazan, el código del proyecto y la base de datos utilizada.