TEMPLE BODY

CICLO FORMATIVO DE GRADO SUPERIOR:

Desarrollo de Aplicaciones Multiplataforma

AUTORES:
Javier Ramiro Castellano
Alberto Arcos Moreno

Licencia

DEDICATORIAS Y AGRADECIMIENTOS

Agradecimientos a nuestro profesor y tutor David Mateos, por las enseñanzas y conocimientos que nos ha proporcionado a la hora de poder ejecutar y programar el proyecto. Gracias a los conocimientos obtenidos orientado al hosting de un correo electrónico para poder mandar correos de verificación y autenticación.

También agradecimientos a Jorge Dueñas por la enseñanza en Android orientado a Java, crear de manera rápida las distintas actividades y fragmentos para crear esta aplicación integrando un sistema de paso entre actividades basado en lenguaje Kotlin además de la implementación de librerías.

RESUMEN

Concretar en esta página el resumen ejecutivo del proyecto:

- Empresa/organización que lo realiza
- Necesidades que cubre
- Posible demanda/clientes
- Breve descripción de la solución que propone este proyecto

La empresa TEMPLE BODY es una empresa enfocada a los ejercicios usados en un gimnasio, la suplementación que puede ayudar a la persona a facilitar el crecimiento, volumen y desarrollo de los despectivos músculos, la localización de los gimnasios más cercanos y recomendados por nuestro equipo y el uso de las máquinas y otros útiles para emplear un buen ejercicio y reducir el riesgo de lesiones. Las posibles demandas de los clientes puede ser la implementación de enlaces de compra de las suplementaciones, la descripción al detalle de cada ejercicio y maquina y de todas las maquinas además de contenido multimedia para poder realizar el ejercicio de la mejor manera viendo un caso práctico.

La solución que nosotros hemos empleado es en crear una aplicación completa el cual puedas buscar los mejores gimnasios a tu alcance, un historial con el que, el usuario, puede ver los ejercicios además del peso, repeticiones y series que hizo en un día concreto, un login con autenticación con el cual puedes crear una cuenta y podrás usar todos los apartados de la aplicación como el desarrollo de cada ejercicio y poder crear los distintos históricos de los ejercicios realizados y un apartado especializado para el tema de la suplementación y la compra de los mismos, con una descripción en cada producto y enlaces de compra a las mejores tiendas de venta de suplementos.

ABSTRACT

Specify the executive summary of the project on this page:

- Company/organization that carries it out
- Needs that cover
- Possible demand/customers
- Brief description of the solution proposed by this project.

TEMPLE BODY is a company focused on gym exercises, supplements that can help facilitate muscle growth, volume, and development, locating the nearest and most recommended gyms by our team, and the use of machines and other tools for proper exercise and injury reduction. Potential customer demands may include the implementation of supplement purchase links, detailed descriptions of each exercise and machine, and multimedia content for optimal exercise demonstration.

Our solution is to create a comprehensive application where you can find the best gyms within your reach, a history log for users to track exercises, weight, repetitions, and sets done on a specific day, a login with authentication to create an account and access all sections of the application, such as exercise instructions and the ability to create different exercise logs, and a specialized section for supplements and their purchase, with descriptions for each product and links to the best supplement stores.

Índice de contenidos

1 INTRODUCCIÓN

-El propósito de este proyecto es la formación de una aplicación Android programado en Java y Kotlin con implementación de distintas librerías para crear nuevas funciones con conexión de la Base de Datos en Firebase y Firebase Authentication y el uso de una página de Hosting llamada piensaSolutions con un Hosting, cuentas de correo y un FTP para subir los archivos de contenido multimedia para una de las fases del proyecto.

Las fases de Desarrollo de este proyecto se dividen en cinco fases:

- Fase de Login de usuarios.
- Fase de creación de usuarios.
- Fase de perfil, configuración e información.
- Fase de la actividad de búsqueda de gimnasios
- Fase de la actividad de suplementos
- Fase de la actividad de ejercicios
- Fase de la actividad historial de ejercicios de usuario

2. NECESIDADES DEL SECTOR PRODUCTIVO

-Dentro de las necesidades en el sector es la creación y uso de una aplicación que contenga todas las funcionalidades que tienen distintas aplicaciones del sector y poder unificarlas en una sola app para centralizar y para el cliente, el uso fácil, intuitivo y cómodo de la aplicación a la hora de practicar un ejercicio.

-La aplicación le brindará al usuario una variedad de posibilidades:

- Como por ejemplo poder realizar una búsqueda de gimnasios en su ubicación actual, dependiendo de cual le haya llamado más la atención del listado que le proporciona la aplicación.
- También podrá consultar un listado de suplementos en los que ver las características de cada uno para así escoger los que más crea convenientes según sus necesidades actuales. Además de una recomendación de marcas de calidad donde podrá ser reenviado al sitio web de búsqueda de la suplementación en la que se encuentre actualmente, eligiendo uno de las marcas que se ofertan.

- Junto a la creación de su cuenta, podrá almacenar sus datos en el perfil, además de tener la posibilidad de consultar su historial de entrenamiento, que irá creando desde la parte de los ejercicios, donde obtendrá una gran variedad de ejercicios, obtenidos a partir de la selección del musculo que el usuario elija.
- En cada músculo podrá ver una descripción del ejercicio, además de un video sobre como poder realizarlo.

1.1 Análisis de la situación actual

-La mayoría de aplicaciones actuales de fitness actuales no recogen una tan amplia variedad de funcionalidades como el catálogo de ejercicios con su explicación y video de cómo realizarlo, además de la posibilidad de crear un historial de ese ejercicio y llevar un control de sus entrenamientos. También podrá consultar su historial de cada día que realice su entrenamiento y cree su registro de lo que ha realizado.

-A parte de ello otras aplicaciones no ofrecen a su usuario la posibilidad de ver sus gimnasios cercanos, ni obtener un listado de las mejores suplementaciones recomendadas actualmente.

-Y todo ello solo con un registro en la aplicación, cuando en otras cobran por ello o introducen publicidad innecesaria que perjudica la navegabilidad y la experiencia de usuario.

1.2 Necesidades del cliente y oportunidad de negocio

-Con la aplicación se buscar que el cliente obtenga una buena experiencia y un completo abanico de posibilidades, para poder cubrir todo lo necesario, tanto en gente experta como novata, de poder asistir al gimnasio, sin necesidad de contratar entrenadores personales, y tener un asistente a la hora de realizar sus ejercicios, llevar su suplementación, crear su historial de entrenamientos y poder -consultarlos para así poder cada día ir mejorando a la hora de realizar su actividad física.

-Con ello se espera mejorar las aplicaciones que hay actualmente en el mercado, sin necesidad de cobrar al cliente gran parte de su sueldo.

-Esta aplicación estará disponible en cualquier parte de España, incluso a nivel mundial, ya que todo lo que ofrece, excepto la posibilidad de búsqueda de gimnasios, ya que el listado está orientado a nivel nacional, se podrá utilizar en cualquier parte del mundo.

-Aquí abajo proporcionamos un gráfico del uso de apps de fitness en la actualidad, en el que podemos ver, que la oportunidad de negocio es muy amplia.

1.3 El nuevo proyecto: Temple Body

2.3.1Tipo de proyecto

Visión general de Temple Body

Temple Body nace con el objetivo de satisfacer una necesidad creciente en el sector del fitness y la salud: la unificación de diversas funcionalidades en una sola aplicación móvil para brindar una experiencia integral, intuitiva y cómoda al usuario durante su rutina de ejercicios. Esta solución innovadora se posiciona como una herramienta imprescindible para cualquier entusiasta del fitness, proporcionando un acceso centralizado a gimnasios, suplementos, y rutinas de entrenamiento que podrá personalizarse él mismo.

Necesidades Detectadas

El análisis del mercado del fitness y la salud ha revelado varias áreas clave donde los usuarios demandan mejoras:

- Centralización de funcionalidades: Existen múltiples aplicaciones que ofrecen funcionalidades específicas, pero no hay una que las integre todas de manera efectiva.
- Acceso rápido y fácil a la información de gimnasios: Los usuarios buscan gimnasios en función de su ubicación actual y desean información detallada y relevante.
- Información y recomendaciones de suplementos: Los usuarios necesitan acceso a una base de datos de suplementos con recomendaciones de marcas de calidad.
- Historial y seguimiento de entrenamiento: La capacidad de almacenar y consultar datos de entrenamiento personal es muy importante para tener un seguimiento del progreso.

Soluciones Propuestas

Temple Body propone una solución integral mediante una aplicación móvil que ofrece:

- Búsqueda de gimnasios basada en la ubicación actual del usuario.
- Consulta de un listado detallado de suplementos con características y recomendaciones.
- Creación y almacenamiento de perfiles de usuario con historial de entrenamiento y acceso a una base de datos de ejercicios con descripciones y videos tutoriales.

Contexto Empresarial

Para dar respuesta a las necesidades detectadas, se creará una nueva empresa dedicada al desarrollo y mantenimiento de la aplicación Temple Body. Esta empresa se centrará en la innovación tecnológica en el sector del fitness y la salud, aprovechando las últimas tecnologías móviles y de bases de datos para ofrecer una experiencia de usuario excepcional.

Tipo de Proyecto a Desarrollar

Desarrollo de Software

El proyecto principal involucra el desarrollo de la aplicación Temple Body, utilizando herramientas y tecnologías como Firebase para backend y almacenamiento de datos, Android Studio para el desarrollo de la aplicación móvil, y la integración de APIs REST para obtener datos dinámicos.

Implementación de Bases de Datos (BBDD)

Se requiere la implementación de una base de datos robusta y escalable que almacene información de usuarios, gimnasios, suplementos y rutinas de entrenamiento. Firebase proporcionará la infraestructura necesaria para manejar esta información de manera eficiente y segura.

Implantación de Software

La implantación del software incluye la distribución de la aplicación a los usuarios finales a través de las plataformas de distribución de aplicaciones (Google Play Store), así como la capacitación y el soporte técnico para asegurar una adopción fluida y eficiente.

Seguridad de Redes

Dada la naturaleza sensible de los datos de los usuarios, es crucial implementar medidas de seguridad robustas para proteger la información personal y de entrenamiento. Esto incluye la encriptación de datos, autenticación segura, y monitoreo continuo de amenazas.

Tecnologías Utilizadas

- Firebase
- Android Studio como entorno de desarrollo integrado (IDE) para la creación de aplicaciones Android
- API REST
- Lenguajes de programación Java y Kotlin

Con la creación de Temple Body, buscamos no solo cubrir las necesidades actuales del sector del fitness, sino también anticiparnos a futuras demandas, ofreciendo una plataforma escalable y adaptable que evolucione junto con las necesidades de nuestros usuarios.

2.3.2 Características requeridas al proyecto

Temple Body tiene el objetivo de crear una aplicación móvil que sea el compañero perfecto para cualquier persona interesada en el fitness. Esta aplicación reunirá en un solo lugar todas las herramientas y recursos que un usuario necesita para mejorar su bienestar y forma física.

Tareas Principales

- **Desarrollar la Aplicación Móvil:**
  - Diseñar una interfaz que sea fácil de usar y atractiva.
  - Implementar funcionalidades esenciales: búsqueda de gimnasios, consulta de suplementos, perfiles de usuario personalizados, acceso a ejercicios en base al musculo seleccionado, donde consultar como ejercitarlo y la creación de historiales personalizados.

- **Implementar una Base de Datos:**
  - Crear un sistema robusto para almacenar información importante, como datos de usuarios, e historiales de ejercicios.

- **Integrar API REST:**
  - Conectar la aplicación con servicios externos para obtener información actualizada sobre ejercicios en base a la musculatura deseada.

- **Crear Contenido de Calidad:**
  - Desarrollar descripciones detalladas y videos tutoriales para cada ejercicio.

- **Asegurar la Privacidad y Seguridad:**
  - Implementar medidas de seguridad para proteger la información personal de los usuarios.

Elementos Diferenciadores

Temple Body se destacará en el mercado por:

- Centralizar Funcionalidades: Una única aplicación que ofrece todo lo que los usuarios necesitan, evitando la necesidad de múltiples apps.
- Interfaz Amigable: Un diseño centrado en la facilidad de uso y una experiencia de usuario agradable.
- Contenido de Alta Calidad: Descripciones detalladas y videos de ejercicios, así como recomendaciones de suplementos de marcas confiables.
- Personalización: Perfiles y rutinas de ejercicio personalizados según las necesidades y objetivos individuales.

Entorno Específico

Para desarrollar Temple Body, se utilizarán las siguientes herramientas y tecnologías:

- **Plataforma de Desarrollo:**
  - Android Studio para crear la aplicación móvil, editar el diseño, dar funcionalidad a toda la aplicación e interconectar los diferentes tipos de entornos escogidos, para así poder crear una aplicación con buena utilidad y funcionalidad.

- **Backend y Almacenamiento:**
  - Firebase para manejar la autenticación, almacenamiento de usuarios e historiales de ejercicios
  - Host para almacenar videotutoriales de ejercicios
- **API REST:**
  - Integración con APIs para obtener datos actualizados sobre ejercicios y su descripción.

Justificación de las Herramientas a Utilizar

- Android Studio: Es la herramienta oficial para crear aplicaciones Android, ofreciendo todo lo necesario para desarrollar, depurar y probar la aplicación.

- Firebase: Proporciona una solución completa para el backend, incluyendo autenticación, almacenamiento en tiempo real y notificaciones push. Es ideal para startups por su escalabilidad y facilidad de uso.

- API REST: Permiten integrar datos externos de forma dinámica y segura, esencial para ofrecer información actualizada.

Recursos Necesarios

Recursos Humanos

-Todos los recursos humanos disponibles actualmente se encargarán de:

- Desarrollar la aplicación móvil y el diseño de la misma.
- Responsables del backend y la integración con Firebase y APIs REST.
- También se encargarán de seleccionar los datos de la parte de la suplementación, gimnasios y ejercicios.

Recursos Materiales

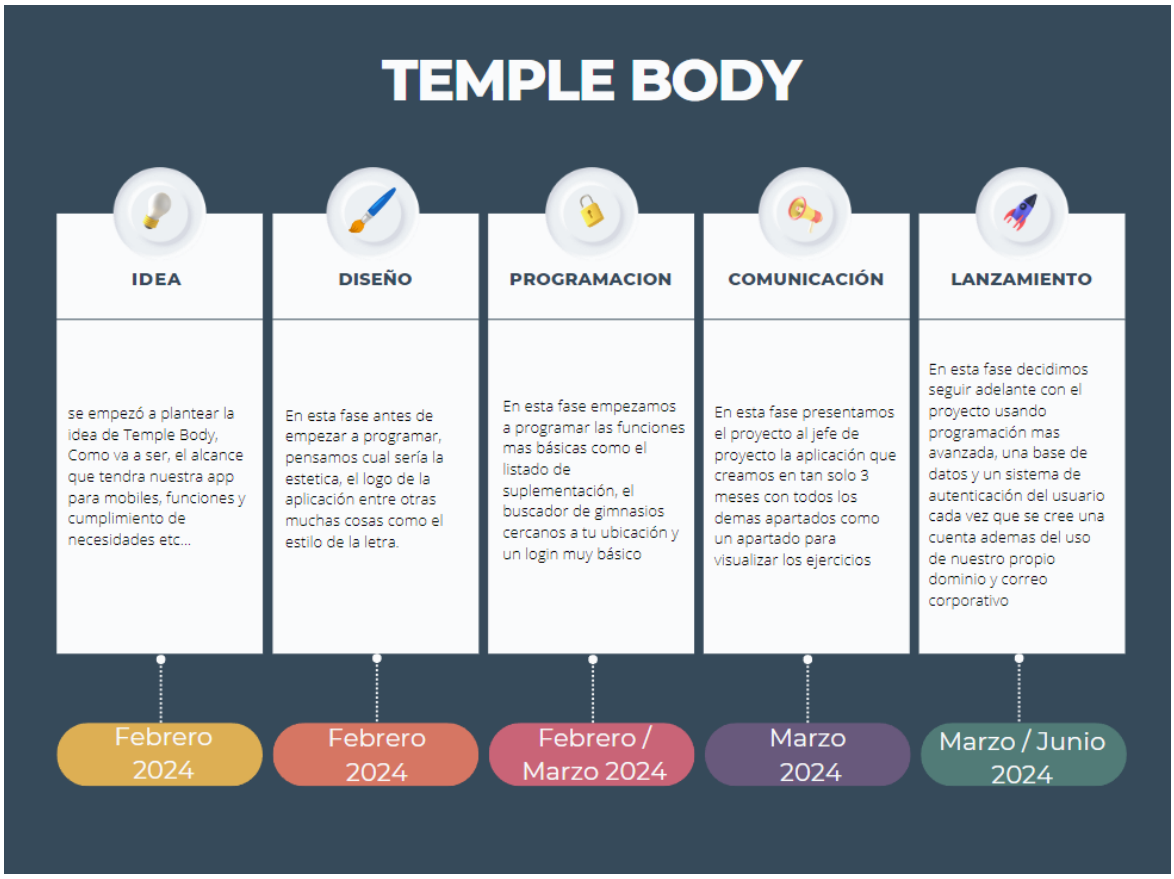
- Equipos de Desarrollo: Ordenadores y software necesarios para desarrollar y probar la aplicación (Android Studio, herramientas de diseño, etc.).
- Infraestructura de Backend: Suscripción a Firebase y otros hosts necesarios para almacenamiento y manejo de datos.
- Acceso a APIs Externas: Mediante el sitio web api-ninjas y consumo de las Apis mediante la biblioteca de Retro-fit.

Resumen

Temple Body quiere ser más que una simple aplicación de fitness. Aspira a ser el aliado perfecto para cualquier persona que quiera mejorar su salud y forma física. Con un equipo dedicado de desarrolladores, diseñadores y expertos en contenido, Temple Body está lista para ofrecer una solución integral y de alta calidad que hará la vida más fácil y saludable para sus usuarios.

Diagramas de Temple Body

- Diagrama de Gantt o Cronograma



Idea

Febrero 2024

En esta fase inicial, se comenzó a planear la idea de Temple Body. Esto incluye definir cómo va a ser la aplicación, el alcance que tendrá nuestra app para móviles, sus funcionalidades y cómo cumplirá con las necesidades de los usuarios. Se realizaron sesiones de lluvia de ideas para explorar diferentes posibilidades y se llevan a cabo estudios de mercado para entender mejor las necesidades y preferencias del público objetivo. Se establecen los objetivos principales y los indicadores clave para el éxito del proyecto. Además, se elaboran perfiles de usuario para identificar los distintos tipos de usuarios que utilizarán la aplicación.

Diseño

Febrero 2024

Antes de comenzar con la programación, en esta fase se pensó en el diseño de la aplicación. Esto incluye cómo será la interfaz de usuario, la experiencia del usuario, la disposición de los elementos, los colores, las tipografías y otros aspectos visuales importantes. Se crean wireframes y prototipos interactivos para visualizar la estructura y el flujo de la aplicación. También se define el estilo visual de la marca, incluyendo el logotipo y los elementos gráficos que se utilizarán en la aplicación.

Programación

Febrero / Marzo 2024

En esta fase, se empieza a programar las funciones más básicas de la aplicación. Esto incluye la creación de una lista de suplementación, un buscador de gimnasios y su ubicación mediante Google Maps y la implementación de un sistema de login muy básico. Se selecciona la tecnología y las herramientas de desarrollo que se utilizarán en el proyecto. El equipo de desarrollo trabaja en la configuración del entorno de desarrollo y en la creación de una arquitectura de software escalable. Se integran servicios de backend y bases de datos para manejar la información de los usuarios y las organizaciones. Se realizan pruebas unitarias y de integración para asegurar la calidad del código.

Comunicación

Marzo 2024

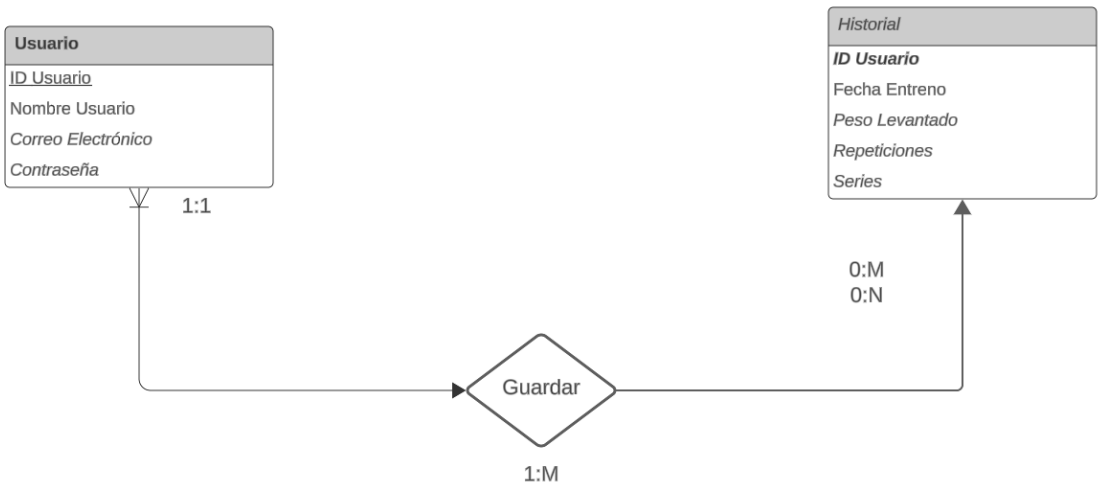
Durante esta fase, se presenta el proyecto al jefe de proyecto y al equipo de trabajo. También se realizan reuniones con todos los integrantes del equipo para asegurarse de que todos estén alineados con la visión y los objetivos del proyecto. Se trabaja en la documentación necesaria para el proyecto y se crean maquetas visuales de los ejercicios que la aplicación incluirá. Además, se desarrollan estrategias de comunicación y marketing para preparar el lanzamiento de la aplicación.

Lanzamiento

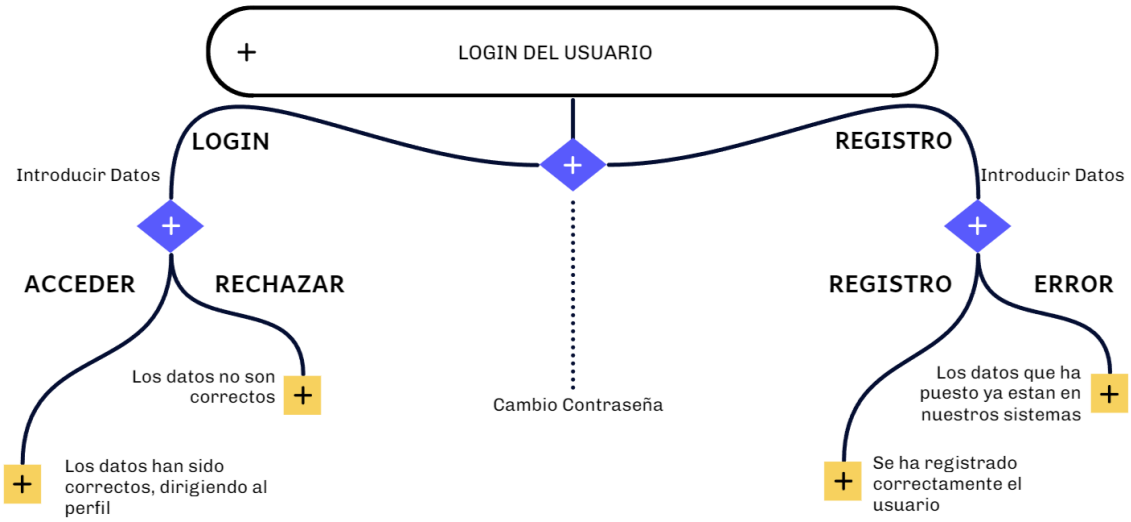
Marzo / Junio 2024

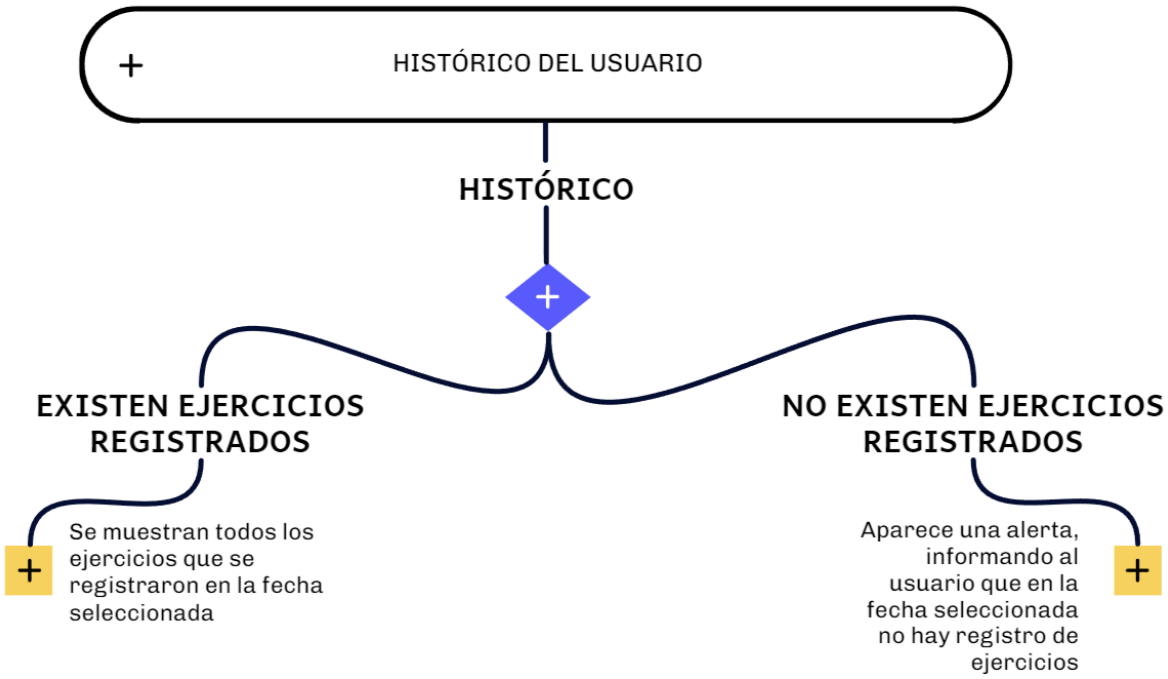
En esta última fase, se decide seguir adelante con el proyecto utilizando la programación más avanzada. Se trabaja en la implementación de nuevas funciones como la autenticación del usuario y se asegura de que cada una de estas nuevas características esté alineada con el propósito corporativo del proyecto. Se realiza una fase de pruebas beta con un grupo seleccionado de usuarios para recoger feedback y realizar ajustes finales. Se prepara toda la infraestructura necesaria para el lanzamiento, incluyendo almacenamiento de datos, uso de APIs, etc. Finalmente, se lanza la aplicación al público, acompañada de una campaña de marketing para atraer a los primeros usuarios. Se monitoriza el rendimiento de la aplicación y se recoge feedback para planificar futuras actualizaciones y mejoras.

- Diagrama de Bases de datos



- Diagrama de bifurcación:





2 DISEÑO DEL PROYECTO

Dando por hecho la viabilidad del proyecto, en este apartado se concretarán las fases necesarias para llevarlo a cabo, y cumplir con los objetivos que se establezcan, teniendo en cuenta los recursos necesarios.

2.1 Fases del proyecto

El desarrollo de este proyecto se llevará a cabo en cuatro fases: análisis, diseño, implementación y pruebas, que pasan a detallarse a continuación.

2.1.1 Análisis

En esta fase se establecerán los requisitos del proyecto, distinguiendo entre los funcionales y no funcionales, así como el alcance y las restricciones del sistema.

Requisitos Funcionales

Búsqueda de gimnasios: Permitir a los usuarios buscar gimnasios en su ubicación actual. Mostrar detalles de cada gimnasio, como dirección, servicios ofrecidos y horarios de apertura.

Consulta de suplementos: Ofrecer un listado de suplementos con descripciones detalladas y características. Incluir recomendaciones de marcas de calidad y enlaces a sitios web donde se pueden comprar.

Perfiles de usuario: Permitir la creación y gestión de perfiles de usuario. Almacenar y mostrar el historial de entrenamiento del usuario. Permitir la personalización de los perfiles.

Consulta de ejercicios: Proporcionar una amplia variedad de ejercicios clasificados por grupo muscular. Incluir descripciones detalladas y videos tutoriales para cada ejercicio. Permitir a los usuarios crear y seguir rutinas de entrenamiento personalizadas en base a los ejercicios proporcionados.

Autenticación de Usuarios: Implementar un sistema de registro y login seguro utilizando Firebase Authentication.

Requisitos No Funcionales

Rendimiento: La aplicación debe cargar en menos de 3 segundos. Las búsquedas y consultas deben responder en menos de 2 segundos.

Usabilidad: Diseño intuitivo y fácil de usar, adaptado a usuarios de todos los niveles de habilidad tecnológica. Interfaz atractiva y moderna.

Escalabilidad: El sistema debe ser capaz de manejar un gran número de usuarios simultáneamente sin degradar el rendimiento.

Seguridad: Protección de datos personales a través de cifrado y medidas de seguridad robustas. Cumplimiento con las normativas de protección de datos vigentes.

Mantenibilidad: El código debe ser modular y bien documentado para facilitar el mantenimiento y la expansión futura.

Fiabilidad: La aplicación debe ser resistente a fallos y capaz de recuperarse rápidamente de errores, sin perjudicar la experiencia del usuario.

Alcance del Sistema

El proyecto Temple Body abarcará las siguientes áreas:

-Desarrollo de una aplicación móvil para Android.

-Implementación de un backend utilizando Firebase.

-Integración con una API externa para obtener datos de ejercicios.

-Implementación de funcionalidades clave como consultas, perfiles de usuario, y rutinas de ejercicio guardadas por fechas.

Restricciones del Sistema

-Plataforma: Inicialmente, la aplicación solo estará disponible para dispositivos Android.

-Recursos: Creación y desarrollo de la aplicación con Android Studio.

Dependencia de una API externa para los datos de ejercicios.

Dependencia de conexión a internet para el uso de bases de datos y autenticación mediante Firebase.

-Regulaciones: Cumplimiento con las leyes y regulaciones de protección de datos.

2.1.2 Diseño

En esta fase se realiza una aproximación al diseño tecnológico de la solución.

Requisitos Funcionales

Búsqueda de Gimnasios

Descripción: Permitir a los usuarios buscar gimnasios en su ubicación actual y mostrar detalles como dirección, servicios ofrecidos y horarios de apertura.

-Implementación:

-Crear una interfaz de búsqueda utilizando Android Studio, integrando una llamada externa a Google Maps para la ubicación y visualización de los gimnasios cercanos en base al gimnasio seleccionado en el listado proporcionado.

-Crear solicitudes para el uso de permisos de búsqueda y geolocalización para permitir a la aplicación el acceso a Google Maps

Consulta de Suplementos

Descripción: Ofrecer un listado de suplementos con descripciones detalladas, características, recomendaciones de marcas y enlaces para compra.

-Implementación:

-Desarrollar una interfaz de listado y detalles en Android Studio, donde consultar los suplementos recomendados en base a las necesidades que se busquen.

-Crear solicitudes para el uso de permisos de búsqueda y navegación web para permitir a la aplicación redirigir al usuario a la web solicitada, y la búsqueda del conjunto de suplementos seleccionado.

Perfiles de Usuario

-Descripción: Permitir la creación y gestión de perfiles de usuario, almacenamiento y visualización del historial de entrenamiento y personalización de perfiles.

-Implementación:

- Crear formularios de registro y edición de perfil, además de vistas para el historial de entrenamiento en Android Studio, políticas de privacidad e información legal, condiciones de uso y reglamentos.
- Crear vistas de configuración del perfil, para edición de datos y reportes de errores.
- Utilizar Firebase Authentication para el registro y login de usuarios. Almacenar los datos del perfil y el historial de entrenamiento en Firebase RealTimeDatabase.

Consulta de Ejercicios

-Descripción: Proporcionar una amplia variedad de ejercicios clasificados por grupo muscular con descripciones y videos tutoriales, y permitir la creación de rutinas personalizadas.

-Implementación:

- Crear interfaces para visualizar los ejercicios y videos, y funcionalidades para la creación de rutinas en Android Studio, registradas por fechas, en base a las repeticiones, series y pesos que se han realizado esa fecha.
- Integrar con una API externa para obtener datos de ejercicios y almacenarlos en Firebase RealTimeDatabase.
- Integración de videotutoriales para ejercicios

Autenticación de Usuarios

-Descripción: Implementar un sistema de registro y login seguro.

-Implementación:

- Formularios de login y registro en Android Studio.
- Utilizar Firebase Authentication para gestionar el registro y login de usuarios.

Requisitos No Funcionales

Rendimiento

- Optimizar las consultas y el acceso a la base de datos en Firebase RealTimeDatabase.

Usabilidad

- Implementar un diseño intuitivo y fácil de usar.
- Realizar pruebas de usabilidad con usuarios de diferentes niveles de habilidad tecnológica para asegurar una experiencia de usuario satisfactoria.

Escalabilidad

- Utilizar Firebase Autentication, que ofrecen escalabilidad automática para manejar un gran número de usuarios simultáneos.

Seguridad

- Implementar el cifrado de datos tanto en tránsito como en reposo utilizando las herramientas de seguridad de Firebase.
- Cumplir con las normativas de protección de datos.

Mantenibilidad

- Seguir prácticas de desarrollo modular y bien documentado.
- Utilizar control de versiones, para gestionar el código fuente.

Fiabilidad

- Realizar pruebas exhaustivas y manejo de excepciones para asegurar la estabilidad de la aplicación.

Definición de la estructura de la aplicación y diseño de componentes

Autenticación:

- Implementada con Firebase Authentication.
- Soporte para autenticación por correo electrónico.

Búsqueda de Gimnasios:

- Utiliza el servicio de ubicación de Google para obtener la posición actual del usuario.

Consulta de Suplementos:

- Uso de navegación web para ir a enlaces de compra y descripciones detalladas de los suplementos.

Gestión de Perfiles de Usuario:

- Almacenamiento de datos de perfil en Firebase.
- Historial de entrenamientos y estadísticas personales.

Consulta de Ejercicios:

- Datos de ejercicios obtenidos de una API externa.
- Almacenamiento en Firebase de rutinas personalizadas.

Componentes de la Infraestructura

API REST:

- Utilizada para obtener los ejercicios y sus datos proporcionados.

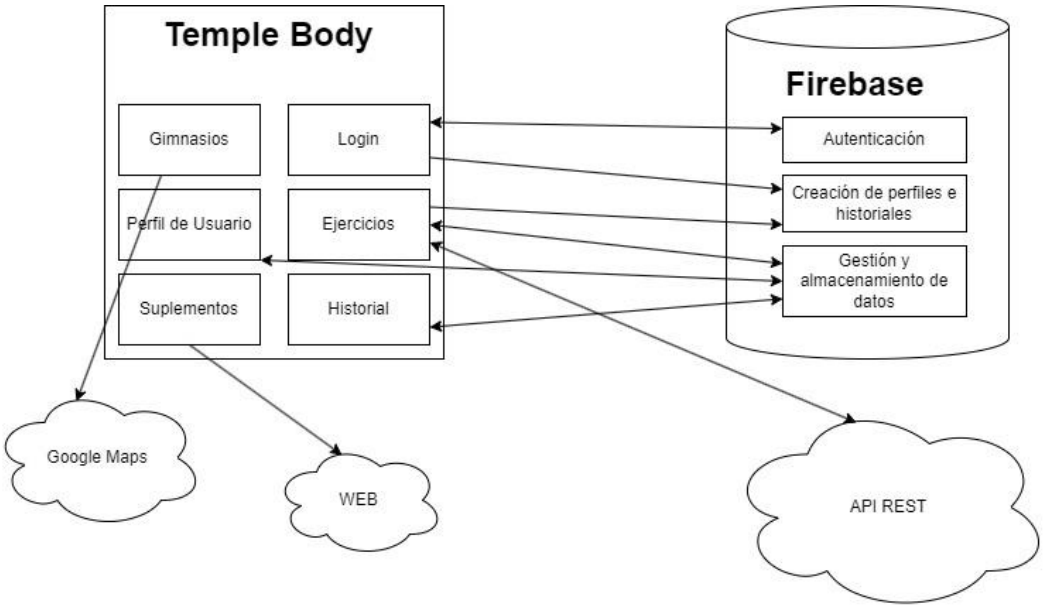
Servidor Web (Firebase):

- Procesamiento de peticiones y consultas de datos para los perfiles de usuarios e historiales creados.

Clientes:

- Aplicación Móvil: Desarrollada en Android Studio para dispositivos móviles Android.

Diagrama de componentes



Aplicación Móvil:

- Home
- Búsqueda de Gimnasios
- Consulta de Suplementos
- Perfil de Usuario
- Configuración del perfil
- Historiales de Ejercicio

Firebase:

- Autenticación
- Creación de perfiles e historiales
- Gestión y almacenamiento de datos

API REST:

- Datos de ejercicios

2.1.3 Implementación

Partiendo del diseño, en esta fase se construye el proyecto, instanciando la configuración del entorno de desarrollo donde se va a realizar la aplicación, la ubicación y almacenamiento de la base de datos, junto a su carga de datos, y como se va a implementar el desarrollo del front-end y back-end de la aplicación.

Configuración del Entorno de Desarrollo

Herramientas

- Android Studio: IDE para desarrollar la aplicación Android.
- Firebase: Plataforma para el almacenamiento de datos de historiales, manejo de creación de usuarios, login y almacenamiento de datos de los mismos.
- Git: Para el control de versiones del código fuente.

Configuración del proyecto Android

- Creación de un nuevo proyecto en Android Studio.
- Configurar los archivos build.gradle y manifest para incluir dependencias como Firebase Authentication, Firestore, Storage y permisos como Google Maps, acceso a internet, etc.

Configuración de Firebase

- Crear un proyecto en Firebase Console.
- Configurar Firebase Authentication, Fire store y Storage.

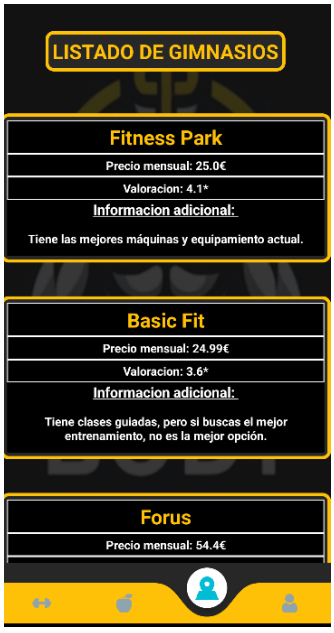
Implementación del Front-end

Diseño de la Interfaz de Usuario (UI)

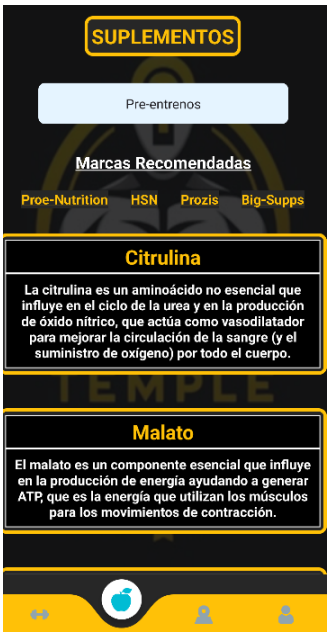
En este apartado se va a detallar la configuración e implementación de lo que incluye la parte gráfica y visual de la aplicación

-Pantalla Principal: Incluye accesos rápidos a las principales funciones, y el login de usuarios, desde la pantalla principal también se podrá acceder al registro de usuarios.

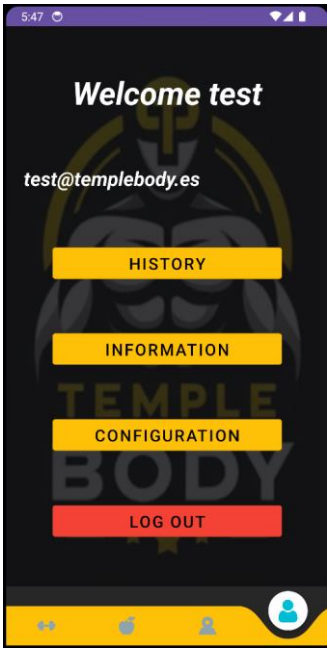
-Búsqueda de Gimnasios: Interfaz para buscar y mostrar gimnasios usando Google Maps, junto a descripciones de sus precios, valoraciones y algunos detalles que incluye cada uno de los gimnasios.



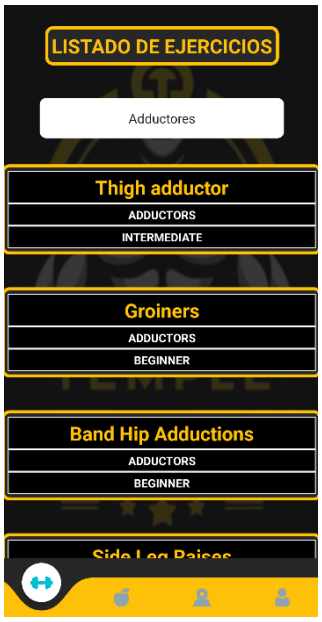
-Consulta de Suplementos: Interfaz para listar y detallar las facetas de los suplementos, que incluye acceso a las páginas de las marcas recomendadas para la compra de los suplementos.

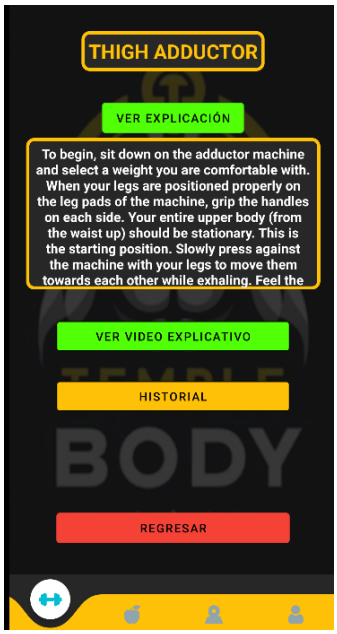


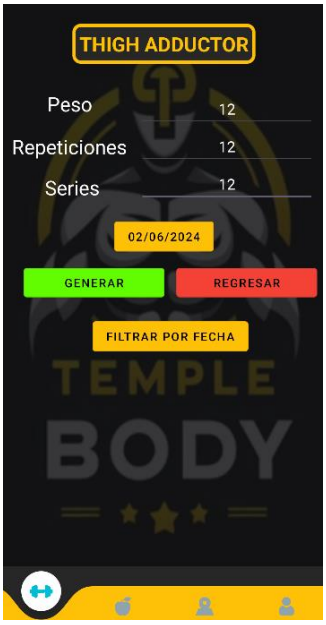
-Perfil de Usuario: Formularios para registro, login, edición de perfil, consulta de historiales de ejercicios y reporte de errores, además de consultas de datos y privacidad.



-Ejercicios: Pantalla para ver ejercicios, descripciones y videos, y crear rutinas personalizadas.







Implementación del Back-end

Firebase Authentication

-Configurar proveedores de autenticación (correo electrónico).

-Implementar lógica de autenticación en la aplicación móvil.

Base de Datos Firebase RealTimeDatabase

Crear colecciones y documentos necesarios:

-Usuarios: Almacena información del perfil.

-Ejercicios: Almacena los datos de los ejercicios con su fecha.

Integración con APIs Externas

-Datos de Ejercicios: Implementar lógica para obtener y actualizar datos de ejercicios desde la API externa.

-Búsqueda de Gimnasios y Suplementos: Implementar lógica para realizar consultas en tiempo real a Google Maps y los sitios web correspondientes. Solicitando permisos al usuario.

2.1.4 Pruebas

Son muchas 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.

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP001 – Movimiento entre Login y registro de usuario y acceso a todos los apartados de la app mediante un menú

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP002 –Funcionamiento correcto del registro del usuario

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP003 – Funcionamiento del envío de correos para registrar la cuenta y cambio de contraseña

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP004 – Funcionamiento hipervínculos en la app

- Autor/Responsable: Alberto Moreno
- Caso de prueba: CP005 – Funcionamiento de la integración de Google Maps para la búsqueda de gimnasios

- Autor/Responsable: Alberto Moreno
- Caso de prueba: CP006 –Funcionamiento de la aparición de los gimnasios cargados en la Base de datos

- Autor/Responsable: Alberto Moreno
- Caso de prueba: CP007 – Funcionamiento de la aparición de todos los suplementos filtrados por tipo y enlace de compra

- Autor/Responsable: Alberto Moreno
- Caso de prueba: CP08 – Funcionamiento de la recogida de datos en una API privada

- Autor/Responsable: Alberto Moreno
- Caso de prueba: CP09 – Funcionamiento de registro de ejercicios en los distintos ejercicios

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP10 – Funcionamiento de la reproducción de contenido multimedia en la aplicación.

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP11 – Funcionamiento del histórico de ejercicios general filtrado por fecha.

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP12 –Funcionamiento del filtrado y la muestra de ejercicios recogidos de API privada

2.2 Objetivos a conseguir

La serie de objetivos que la empresa va a tener planteados conseguir en el menor plazo posible serán los siguientes:

-Expandir la disponibilidad de la aplicación a usuarios en múltiples países.

-Satisfacer los Requisitos del Cliente

-Proveer una experiencia de usuario intuitiva y atractiva que vaya más allá de las expectativas del usuario.

-Generar Ingresos

-Implementar modelos de monetización como suscripciones premium dentro de la aplicación

-Mejorar la Visibilidad y Alcance de la Empresa

-Fomentar una comunidad activa de usuarios que compartan sus experiencias y rutinas de ejercicio.

-Implementar características sociales como foros, comentarios y calificaciones de ejercicios y suplementos.

2.3 Previsión de los recursos materiales y humanos necesarios

En principio los recursos materiales serán:

-Un host donde almacenar la base de datos

-Un host donde almacenar los video tutoriales de los ejercicios

-Ordenadores con hardware preparados para soportar el uso de Android Studio y el desarrollo y uso de la aplicación

-Personal cualificado para llevar a cabo todas las partes del desarrollo del proyecto

2.4 Presupuesto económico.

-El presupuesto económico actual es de 10€, puesto que el personal actual es estudiante y no dispone de un fondo de inversión.

3 PLANIFICACIÓN DE LA EJECUCIÓN DEL PROYECTO

A continuación, se detallan las actividades/tareas/procedimientos por cada una de las fases del proyecto previamente establecidas.

3.1 Fase de Análisis

1. Estudio de las necesidades a cubrir

El proyecto Temple Body se enfoca en proporcionar una solución integral para los usuarios que desean mejorar su condición física. La necesidad principal es crear una aplicación que combine las funcionalidades de varias aplicaciones del sector fitness en una sola, facilitando así su uso y mejorando la experiencia del usuario. Las necesidades incluyen:

-Acceso a Información de Gimnasios: Permitir a los usuarios buscar y encontrar gimnasios en su ubicación actual.

-Consulta de Suplementos: Proporcionar información detallada sobre suplementos y recomendaciones de marcas.

-Gestión de Perfiles de Usuario: Permitir la creación y gestión de perfiles, incluyendo el almacenamiento de datos de entrenamiento.

-Consulta de Ejercicios: Ofrecer una amplia variedad de ejercicios con descripciones y videos tutoriales.

-Posible creación de una rutina de entrenamiento: Ofrecer la posibilidad de que el usuario sea capaz de crear una rutina en base a los ejercicios que va almacenando en las fechas en las que los realiza, pudiendo saber en todo momento los pesos, repeticiones y series que ha realizado en la fecha que desee consultar.

-Facilidad de Uso: Crear una interfaz intuitiva y fácil de usar para usuarios de todos los niveles.

2. Estudio de la situación actual

En la actualidad, el mercado de aplicaciones de fitness está saturado con numerosas aplicaciones que ofrecen funcionalidades específicas, pero carecen de una solución integral que aborde todas las necesidades del usuario en un solo lugar. A continuación, se presenta un análisis detallado del mercado actual:

Aplicaciones de Gimnasios

-Funcionalidades Comunes:

- Búsqueda de gimnasios por ubicación.
- Información detallada sobre servicios ofrecidos.
- Horarios de apertura y cierre.

-Limitaciones:

- No integran funcionalidades relacionadas con suplementos o rutinas de ejercicios.
- Pocas opciones de personalización para el usuario.

Aplicaciones de Suplementos

-Funcionalidades Comunes:

- Listados de suplementos con descripciones y beneficios.
- Recomendaciones de marcas y enlaces a sitios de compra.

-Limitaciones:

- No ofrecen información sobre gimnasios ni rutinas de ejercicios.
- Poca integración con otras aplicaciones o servicios.

Aplicaciones de Ejercicios

Funcionalidades Comunes:

- Listados de ejercicios clasificados por grupo muscular.
- Videos tutoriales y descripciones detalladas.
- Creación de rutinas de entrenamiento personalizadas.

Limitaciones:

- No incluyen funcionalidades para la búsqueda de gimnasios o consulta de suplementos.
- Poca interacción con otros usuarios o aspectos sociales.

3. Establecimiento de los requisitos del proyecto

-En esta fase, se detallan los requisitos específicos que el proyecto Temple Body debe tener.

-Cada parte detallará un poco las funciones que debe abordar y cumplir, para llevar a cabo el desarrollo de la aplicación.

Búsqueda de Gimnasios

- Reunir en una pantalla todos los datos necesarios para implementar la información detallada de los gimnasios, cuyo contenido deberá ser el nombre del gimnasio, una valoración recopilada de la búsqueda de varios gimnasios de la misma cadena, el precio de la mensualidad establecida habitualmente, y una información extra de lo que incluye el gimnasio
- Redirigir a la búsqueda en Google Maps del gimnasio seleccionado

Consulta de Suplementos

- Proporcionar un listado de suplementos con descripciones detalladas
- Incluir información beneficios, recomendaciones de uso, y más detalles de cada suplemento en especifico
- Sugerir marcas de calidad y proporcionar enlaces a sitios web para la compra

Perfiles de Usuario

- Permitir la creación y gestión de perfiles de usuario
- Almacenar información personal, historial de entrenamiento y poder contactar con la empresa y otros usuarios de la aplicación
- Permitir a los usuarios personalizar su perfil y consultar sus rutinas de entrenamiento

Consulta de Ejercicios

- Ofrecer una amplia variedad de ejercicios clasificados por grupo muscular.
- Incluir descripciones detalladas y videos tutoriales
- Permitir a los usuarios crear registros de ejercicios y rutinas de entrenamiento personalizadas según la fecha

Autenticación de Usuarios

- Implementar un sistema de registro y login seguro utilizando Firebase Authentication.
- Permitir autenticación mediante correo electrónico

3.2 Fase de diseño

1. Preparación del entorno de diseño

-Para la preparación del entorno de diseño, se ha decidido utilizar Android Studio para el desarrollo y el diseño de la aplicación. A continuación, se va a detallar las características más relevantes por las que lo hemos elegido como entorno de diseño y desarrollo:

- Este es un entorno de desarrollo integrado (IDE), que se basa en el software IntelliJ IDEA. Como tal es un potente editor, que permite el análisis del código cuando estas desarrollando, además tiene herramientas integradas de Android.
- Permite el aumento de la productividad mediante el soporte de las herramientas para pruebas unitarios Maven y Gradle.
- Cuenta con manejo de control de versiones como las herramientas: Git, GitHub y muchos más.
- Tiene alertas en tiempo real de errores sintácticos, compatibilidad o rendimiento antes de compilar la aplicación.

- Utiliza ProGuard para optimizar y reducir el código del proyecto al exportar a APK (muy útil para dispositivos de gama baja con limitaciones de memoria interna).
- Tiene soporte para programar aplicaciones para Android Wear (sistema operativo para dispositivos corporales como por ejemplo un reloj).
- Posee la herramienta Lint, que detecta código no compatible entre arquitecturas diferentes. Esto permite encontrar problemas de rendimiento, usabilidad y compatibilidad entre versiones de la app.
- Integración de la herramienta Gradle encargada de gestionar y automatizar la construcción de proyectos, como pueden ser las tareas de testing, compilación o empaquetado.
- Vista previa en diferentes dispositivos y resoluciones.

2. Diseño de la arquitectura

La aplicación Temple Body seguirá una arquitectura cliente-servidor, donde la aplicación móvil actuará como cliente y Firebase como backend, alojando la base de datos. Además, se utilizará un hosting externo para alojar los videos tutoriales de los ejercicios además de usar el servidor de correos para los correos automatizados de cambio de contraseña y registro de usuario.

Diagrama de Componentes

Aplicación Móvil (Android Studio) ←→ Firebase (Authentication, Real time Database) ←→ Hosting Externo (Videos de Ejercicios)

Aplicación Móvil:

- Desarrollada con Android Studio utilizando Java y Kotlin.
- Interfaz de usuario intuitiva y fácil de usar, con un diseño moderno y atractivo.
- Pantallas principales:
    - Inicio/Login
    - Búsqueda de Gimnasios
    - Consulta de Suplementos
    - Perfil de Usuario (con historial de ejercicios)
    - Ejercicios (con descripciones y videos)
- Navegación mediante un menú inferior y botones de acción.
- Integración con Google Maps para la búsqueda y visualización de gimnasios.

Backend (Firebase):

- Autenticación de usuarios mediante correo electrónico y contraseña (Firebase Authentication).

- Base de datos en tiempo real (Real time Database) para almacenar:
    - Datos de usuarios (perfil, historial de ejercicios).
    - Información de gimnasios (nombre, ubicación, servicios, horarios).
    - Datos de suplementos (nombre, descripción, características, enlaces de compra).

Hosting Externo:

- Almacenamiento de videos tutoriales de ejercicios.
- La aplicación móvil accederá a los videos a través de URLs.

Flujo de Datos:

1. El usuario inicia sesión o se registra en la aplicación móvil.
2. Firebase Authentication verifica las credenciales y gestiona la sesión del usuario.
3. La aplicación carga los datos del usuario desde Real time Database.
4. El usuario interactúa con las diferentes pantallas de la aplicación:
    - Búsqueda de Gimnasios: La aplicación utiliza la ubicación del usuario y consulta Real time Database para mostrar gimnasios cercanos. Al seleccionar un gimnasio, se muestra su información detallada.
    - Consulta de Suplementos: La aplicación muestra un listado de suplementos desde Real time Database. Al seleccionar un suplemento, se muestra su información detallada y enlaces de compra.
    - Perfil de Usuario: El usuario puede ver y editar su perfil, así como consultar su historial de ejercicios almacenado en Real time Database.
    - Ejercicios: La aplicación muestra un listado de ejercicios por grupo muscular. Al seleccionar un ejercicio, se muestra su descripción, y se reproduce el video tutorial correspondiente desde el hosting externo. El usuario puede registrar sus ejercicios realizados.
5. Los datos generados por el usuario (historial de ejercicios) se almacenan en Real time Database.

3. Diseño de los interfaces

El diseño de las interfaces en Temple Body se centrará en la usabilidad, la estética y la eficiencia para proporcionar una experiencia de usuario óptima. Aquí hay un ejemplo enfocado en el diseño de la interfaz de "Búsqueda de Gimnasios":

Pantalla de Búsqueda de Gimnasios:

- **Mapa**:
  - Integración con Google Maps para mostrar la ubicación del usuario y los gimnasios cercanos.
  - Marcadores personalizados para identificar cada gimnasio en el mapa.
  - Posibilidad de hacer zoom y desplazarse por el mapa.

- **Listado de Gimnasios:**
  - Tarjeta para cada gimnasio con información relevante:
    - Nombre del gimnasio
    - Valoración por estrellas (promedio de opiniones de usuarios)
    - Precio
    - Descripción breve de servicios prestados

- **Detalle del Gimnasio:**
  - Al tocar una tarjeta de gimnasio, se abre una pantalla en Google Maps con información detallada:
    - Nombre completo
    - Dirección completa
    - Número de teléfono
    - Sitio web
    - Horario de apertura
    - Servicios ofrecidos (clases, equipamiento, etc.)
    - Fotos adicionales
    - Opiniones de usuarios
    - Botón para obtener indicaciones de cómo llegar (usando Google Maps).

Elementos de Diseño:

- **Colores**: Utilizar una paleta de colores coherente con la marca Temple Body, que transmita energía, salud y motivación.
- **Tipografía**: Elegir fuentes legibles y atractivas para títulos, subtítulos y texto del cuerpo.
- **Iconografía**: Utilizar iconos claros y reconocibles para representar acciones y elementos de la interfaz (por ejemplo, lupa para búsqueda, estrella para valoración, marcador para ubicación).
- **Espaciado**: Distribuir los elementos de forma equilibrada y dejar suficiente espacio en blanco para facilitar la lectura y evitar la saturación visual.
- **Interacciones**: Diseñar interacciones intuitivas y fluidas, como transiciones suaves entre pantallas y animaciones sutiles para mejorar la experiencia del usuario.

4. Diseño de los datos

El diseño de datos en Temple Body se centrará en estructurar la información de manera eficiente y lógica para facilitar su almacenamiento, recuperación y uso en la aplicación. Se utilizará Firebase Real time Database como base de datos principal, y un hosting externo para almacenar los videos tutoriales de ejercicios.

Estructura de Datos en Firebase Real time Database:

- **Usuarios**:
  - UID (Identificador único de usuario): Clave principal para identificar a cada usuario.
  - Nombre: Nombre completo del usuario.
  - Correo electrónico: Dirección de correo electrónico del usuario.
- **Historial de Ejercicios:**
    - Fecha (Clave): Fecha en formato DD-MM-AAAA (por ejemplo, 04-06-2024).
    - Ejercicios (Lista):
      - Nombre del ejercicio
      - Grupo muscular
      - Series
      - Repeticiones
      - Peso

Estructura de Datos en Hosting Externo:

- Videos de Ejercicios:
    - Los videos se almacenarán en carpetas organizadas por grupo muscular.
    - El nombre de cada archivo de video incluirá el nombre del ejercicio.
    - La aplicación accederá a los videos a través de URLs.

5. Diseño de los procedimientos

Registro de Usuario:

- El usuario accede a la pantalla de registro.
- El usuario ingresa su nombre de usuario, correo electrónico y contraseña.
- La aplicación valida los datos ingresados (formato de correo electrónico, longitud de contraseña, etc.).
- Si los datos son válidos, se crea un nuevo usuario en Firebase Authentication utilizando el correo electrónico y la contraseña.
- Se envía un correo electrónico de verificación al usuario.
- Se muestra un mensaje al usuario indicando que se ha enviado un correo de verificación y que debe confirmar su cuenta.

Verificación de Correo Electrónico:

- El usuario recibe un correo electrónico con un enlace de verificación.
- Al hacer clic en el enlace, Firebase Authentication verifica la cuenta del usuario.
- El usuario es redirigido a la aplicación.
- Se muestra un mensaje al usuario confirmando que su cuenta ha sido verificada.

Inicio de Sesión:

- El usuario accede a la pantalla de inicio de sesión.
- El usuario ingresa su correo electrónico y contraseña.
- Firebase Authentication verifica las credenciales.
- Si las credenciales son válidas, se inicia la sesión del usuario.
- Se carga la información del perfil del usuario desde Real time Database.
- El usuario es redirigido al perfil del usuario

Recuperación de Contraseña:

- El usuario accede a la opción de recuperación de contraseña.
- El usuario ingresa su correo electrónico.
- Firebase Authentication envía un correo electrónico con un enlace para restablecer la contraseña.
- El usuario sigue las instrucciones del correo electrónico para restablecer su contraseña.

Crear un histórico de ejercicios

- El usuario accede a la opción de ejercicios.
- El usuario selecciona por medio de un desplegable el músculo.
- Busca el ejercicio que necesita hacer.
- El usuario accede al botón historial para crear un histórico del ejercicio
- Introduce la fecha, Peso, Series y Repeticiones y se crea un nuevo histórico del ejercicio que se almacena en Firebase Real Data Base por id del usuario, fecha, ejercicio del histórico

3.3 Fase de Implementación

1. Preparación del entorno de implementación

-En esta etapa inicial, se configurarán las herramientas y plataformas necesarias para el desarrollo y despliegue de Temple Body. Los pasos clave incluyen:

Configuración de Android Studio

- Instalar la última versión de Android studio para tener actualizadas todas las funcionalidades que nos puede proporcionar el IDE
-

Configuración Inicial

- Crear un nuevo proyecto en Android Studio. Configurar el nombre del proyecto, la ubicación del proyecto, el tipo de actividad principal, etc.
- Configurar el SDK Manager y seleccionar las versiones de SDK de Android necesarias para el proyecto y descargarlas.

- Configurar el AVD Manager para tener el dispositivo virtual en el que podremos visualizar y manejar nuestra aplicación desde el ordenador, para poder realizar pruebas y comprobar su funcionamiento. Se podrán crear varios dispositivos con diferente hardware y software para poder comprobar el correcto funcionamiento en ellos.

Configuración del Entorno de Desarrollo

- Dependencias: Configurar las dependencias necesarias en el archivo build.gradle del proyecto. Añadir las librerías necesarias para Firebase, Google Maps, y cualquier otra API o SDK que se vaya a utilizar.
- Emuladores o Dispositivos Físicos: Configurar dispositivos físicos conectándolos a través de USB o configurando emuladores para realizar pruebas durante el desarrollo.

Configuración de Firebase

Crear Proyecto en Firebase

- Acceder a Firebase Console para crear un proyecto nuevo y seguir las instrucciones para completarlo, introduciendo un nombre del proyecto y diferentes datos.
- Añadir Firebase al Proyecto Android y seguir las instrucciones para descargar el archivo google-services.json y colocarlo en el directorio app del proyecto de Android Studio.

Habilitar Servicios de Firebase

- Firebase Authentication:
  - En la consola de Firebase, ir a Authentication y habilitar los métodos de autenticación necesarios (por ejemplo, email/password, Google Sign-In, etc.).
  - Configurar las reglas de seguridad para permitir el acceso adecuado.
- Real time Database:
  - Navegar a Database y seleccionar Real time Database.
  - Hacer clic en "Create Database" y seleccionar el modo de inicio (modo de prueba para el desarrollo inicial).
  - Configurar las reglas de seguridad para proteger los datos.
- Cloud Storage:
  - Navegar a Storage y configurar Cloud Storage.
  - Definir las reglas de seguridad para permitir el acceso adecuado a los archivos.

Integración con Android Studio

- Para integrar Firebase habrá que añadir las dependencias necesarias en el archivo build.gradle de la siguiente manera:

```
implementation platform('com.google.firebase:firebase-bom:26.2.0')
implementation 'com.google.firebase:firebase-auth'
implementation 'com.google.firebase:firebase-database'
implementation 'com.google.firebase:firebase-storage'
```

- Una vez lo tengamos ya integrado podremos trabajar con sus diferentes funcionalidades

API Ninjas Fit para Ejercicios

- Configurar la API y obtener la key para poder hacer uso de la misma en la aplicación
- Una vez obtenida la key, se usará la librería RetroFit para hacer uso de la API y crear peticiones

```
public class ServiceEjercicios {
    4 usages
    private static ServiceEjercicios ejercicios;
    3 usages
    private static EjerciciosAPI API;

    2 usages  JavierRamiro8
    private ServiceEjercicios(){
        Retrofit retrofit=new Retrofit.Builder()
                .baseUrl("https://api.api-ninjas.com")
                .addConverterFactory(GsonConverterFactory.create())
                .build();
        API=retrofit.create(EjerciciosAPI.class);
    }
    16 usages  JavierRamiro8
    public static EjerciciosAPI getAPI(){
        if (API == null) {
            ejercicios = new ServiceEjercicios();
        }
        return API;
    }


    JavierRamiro8
    public static ServiceEjercicios getEjercicios(){
        if (ejercicios ==null){
            ejercicios =new ServiceEjercicios();
        }
        return ejercicios;
    }
}
```

- Habrá que controlar los datos recibidos mediante un POJO con el que se trabajará en las demás clases para poder pintar los datos que nos devuelve la consulta a la API.

- Los datos serán utilizados en la aplicación mediante los RecyclerView y los diferentes fragments que contengan información sobre la API

Configuración de videotutoriales

- Crear un elemento de WebView para el uso de videos mediante URL

Estos pasos aseguran que el entorno de implementación esté completamente configurado y listo para el desarrollo y despliegue de Temple Body, facilitando un flujo de trabajo eficiente y una integración sin problemas de las funcionalidades planificadas.

2. Desarrollo de la arquitectura

En esta etapa, se implementará la arquitectura de software definida en el diseño, siguiendo el patrón cliente-servidor. Este proceso se divide en el desarrollo del backend utilizando Firebase y el desarrollo del frontend mediante Android Studio.

Desarrollo del Backend (Firebase):

- Una vez hayamos integrado Firebase en nuestro entorno y proyecto de Android Studio, podremos utilizarlo para el registro de usuarios, login, y diferentes configuraciones de la BBDD, como almacenar los datos de historiales de ejercicios y poder consultarlos más adelante.

- Para inicializar Firebase Authentication en la actividad principal de la aplicación se usará un código parecido a este:

```
FirebaseAuth mAuth = FirebaseAuth.getInstance();
```

- Para el registro de usuarios se usará un código como este:

```
mAuth.createUserWithEmailAndPassword(email, password)
    .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful()) {
                // Registro exitoso
                FirebaseUser user = mAuth.getCurrentUser();
            } else {
                // Registro fallido
            }
        }
    });
```

- Para el inicio de sesión:

```
mAuth.signInWithEmailAndPassword(email, password)
    .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful()) {
                // Inicio de sesión exitoso
                FirebaseUser user = mAuth.getCurrentUser();
            } else {
                // Inicio de sesión fallido
            }
        }
    });
```

Estructuración de la Base de Datos en Realtime Database

- Diseño de la Estructura de la Base de Datos
    - El diseño de la estructura de la base de datos estará formado de la siguiente manera:

    - La tabla de usuarios, contendrá como claves los ID de usuario que se vayan registrando, y dentro de sus atributos estarán el nombre de usuario, el e-mail y la contraseña.

    - La tabla de ejercicios, contendrá como claves los ID de usuario y estos a su vez, crearán diferentes claves de fechas, en las que contendrán las diferentes claves con el nombre de los ejercicios que el usuario vaya registrando.
    - A su vez estos ejercicios, contendrán una serie de atributos, que serán la misma fecha en la que se ha realizado, el peso utilizado, las repeticiones y las series realizadas.

Desarrollo del Frontend (Aplicación Móvil):

- Crear las pantallas de la aplicación (inicio, búsqueda de gimnasios, consulta de suplementos, perfil de usuario, ejercicios) utilizando Android Studio y los lenguajes de programación Java y Kotlin.
- Implementar la lógica para interactuar con el backend de Firebase y las APIs externas.
- Diseñar la interfaz de usuario (UI) siguiendo las pautas de diseño establecidas.

3. Desarrollo de los interfaces, los datos y los procedimientos

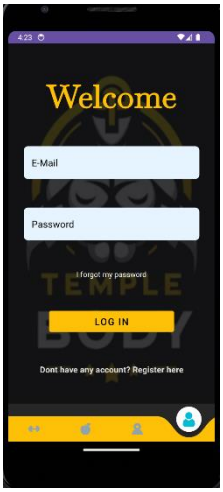
3.1. Interfaces

Desarrollo de las Pantallas de la Aplicación

Para desarrollar las pantallas de la aplicación, utilizaremos Android Studio y sus componentes de interfaz de usuario. A continuación, se detallan las pantallas clave de la aplicación Temple Body y cómo se implementarán.

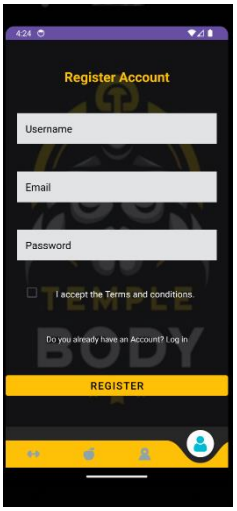
Pantalla de inicio de sesión

- Dispondrá de 2 campos EditText para recoger la información, y un botón para enviar la información a la base de datos y logear si es correcta. Todo ello se enviará mediante la implementación de Firebase.
- También tendrá dos links al registro de usuarios y a la pantalla de reestablecer la contraseña



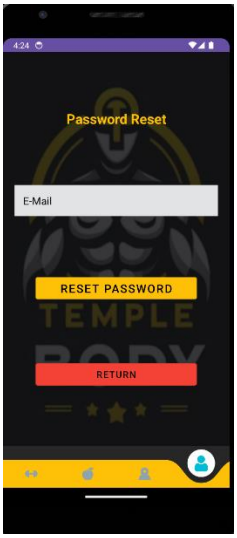
Página de registro

- Formado por 3 EditText para recopilar al información y enviarla a la base de datos si está introducida correctamente, mediante la implementación y uso de Firebase
- Botón de registro para el envió de datos
- Campo para aceptar los términos de uso
- Link a login si ya se dispone de cuenta creada



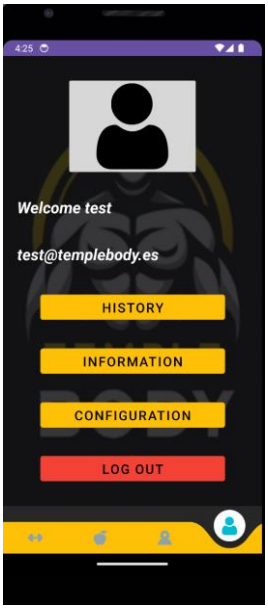
Página de resteo de contraseña

- Botones de reseteo de contraseña para enviar los datos del e-mail y botón de regreso al login o perfil de usuario si ya se ha logueado



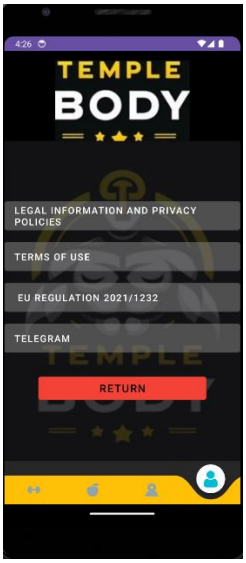
Página de perfil de usuario

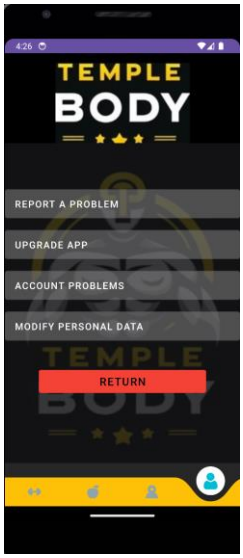
- Botones al acceso del historial, información configuración y log out
- Textos de bienvenida, y correo de la cuenta mediante elementos TextView de Android Studio



Páginas de configuración e información

- Diferentes elementos tipo Button para el acceso a las vistas de políticas, informacion, reporte de problemas, telegram, etc.





Páginas de historial de ejercicios y videotutoriales

- Dispondrá de diferentes campos EditText para introducir los datos de pesos, series y repeticiones
- Elementos DatePicker que recopilaran la informacion de las fechas para la creación de historiales y el filtrado de fechas
- Los historiales registrados del ejercicio se mostrarán mas abajo en un Recycler view
- Todos estos datos serán enviados y consultados mediante Firebase
- Vista de videotutoriales con elemento WebView que mostrará el video que se consume mediante URL

Pantalla de búsqueda de gimnasios

- Elemento RecyclerView para mostrar el listado de gimnasios
- Dentro del RecyclerView contendrá un layout con 4 EditText para mostrar los datos
- Serán clickables para el acceso a Google Maps y la búsqueda posterior

Página de suplementación

- Elemento Spinner para la selección del tipo de suplemento, que posteriormente mostrará mas abajo en el RecyclerView los suplementos recogidos, que tiene la misma estructura que en los gimnasios pero con menos campos TextView para mostrar los datos
- Links mediante textos con URLs a las páginas de marcas recomendadas para la compra de suplementación seleccionada en el Spinner

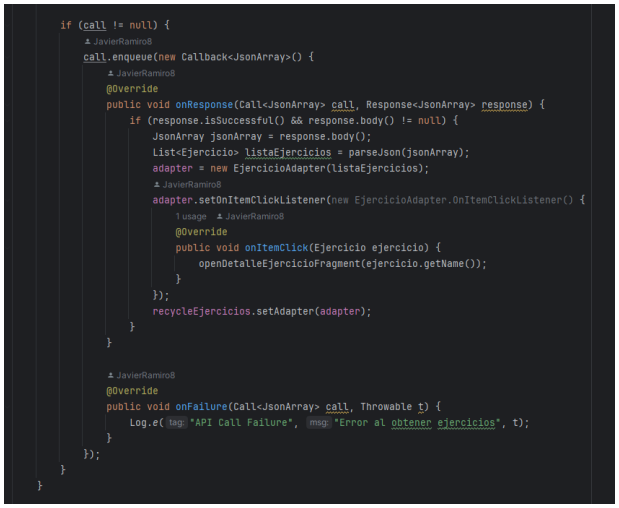
Historial general

- Tendrá un DatePicker para obtener los datos de la BBDD mediante Firebase, y mostrará los datos recopilados en un RecyclerView.
- Botón de regreso al perfil de usuario

3.2. Datos y procedimientos:

-En este apartado se llevara a cabo la implementación de la lógica para obtener y mostrar los datos de los ejercicios desde Firebase Real time Database y las APIs externas.

- Llamadas mediante RetroFit a la API, recogidas por un CallBack, de donde se obtiene el JSON con el que se fragmentará la información obtenida y se pintará en el RecyclerView.
- Todos estos datos serán tratados por un pojo de Ejercicio que manejará todos los atributos



```
if (call != null) {
    JavierRamiro8
    call.enqueue(new Callback<JsonArray>() {
        JavierRamiro8
        @Override
        public void onResponse(Call<JsonArray> call, Response<JsonArray> response) {
            if (response.isSuccessful() && response.body() != null) {
                JsonArray jsonArray = response.body();
                List<Ejercicio> listaEjercicios = parseJson(jsonArray);
                adapter = new EjercicioAdapter(listaEjercicios);
                JavierRamiro8
                adapter.setOnItemClickListener(new EjercicioAdapter.OnItemClickListener() {
                    1 usage   JavierRamiro8
                    @Override
                    public void onItemClick(Ejercicio ejercicio) {
                        openDetalleEjercicioFragment(ejercicio.getName());
                    }
                });
                recycleEjercicios.setAdapter(adapter);
            }
        }

        JavierRamiro8
        @Override
        public void onFailure(Call<JsonArray> call, Throwable t) {
            Log.e( tag: "API Call Failure",  msg: "Error al obtener ejercicios", t);
        }
    });
}
}
```

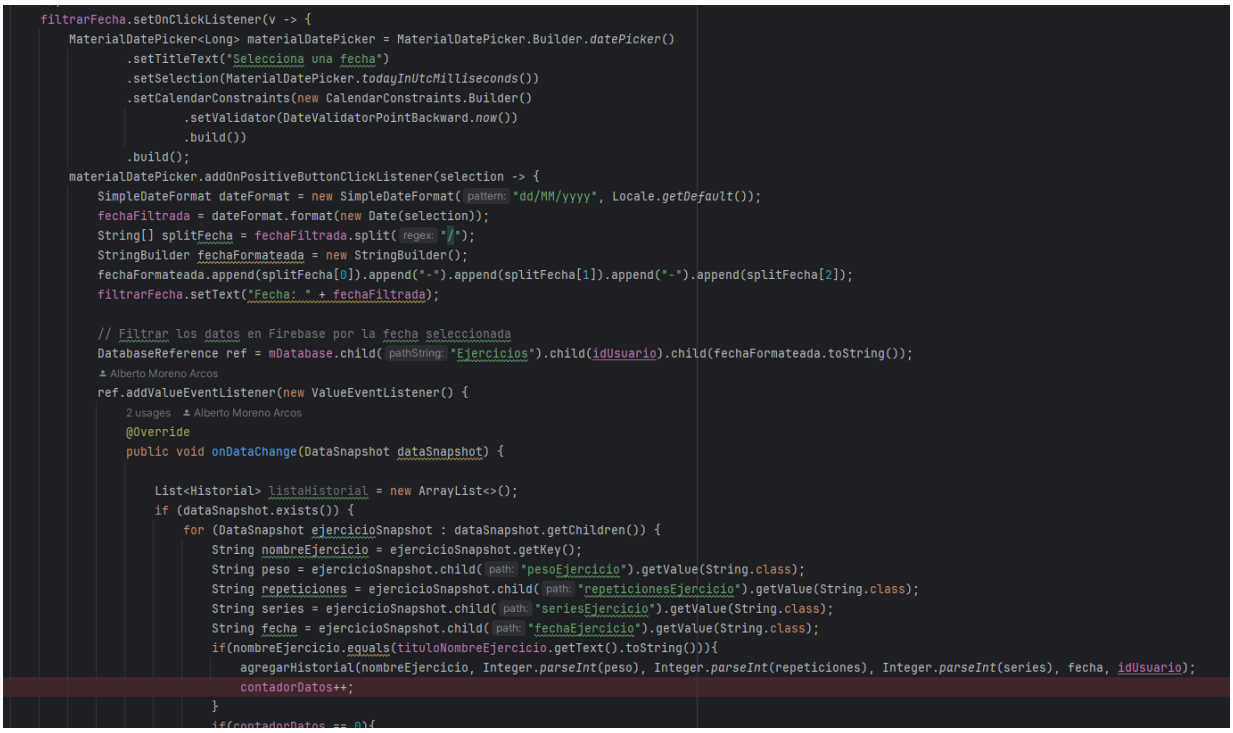
```
1 usage   JavierRamiro8
private void openDetalleEjercicioFragment(String ejercicio) {
    Bundle bundle=new Bundle();
    bundle.putString("nombreEjercicio",ejercicio);
    NavController nav= NavHostFragment.findNavController( fragment: this);
    nav.navigate(R.id.action_ejerciciosFragment_to_detalleEjercicio3,bundle);
}

1 usage   JavierRamiro8
private List<Ejercicio> parseJson(JsonArray jsonArray) {
    List<Ejercicio> listaEjercicios = new ArrayList<>();
    for (JsonElement jsonElement : jsonArray) {
        JsonObject ejercicioObject = jsonElement.getAsJsonObject();
        String name = ejercicioObject.get("name").getAsString();
        String muscle = ejercicioObject.get("muscle").getAsString();
        String difficulty = ejercicioObject.get("difficulty").getAsString();

        Ejercicio ejercicio = new Ejercicio(name, muscle, difficulty);
        listaEjercicios.add(ejercicio);
    }

    return listaEjercicios;
}

}
```

- Implementar la lógica para almacenar los datos generados por el usuario (historial de ejercicios) en Real time Database.

- A continuación, se detalla cómo se realiza el registro de ejercicios en la BBDD de Firebase, una vez se han obtenido todos los datos.

```
private void registrarEjercicio(String idUsuario) {
    if (idUsuario != null) {
        Map<String, Object> mapa = new HashMap<>();
        mapa.put("pesoEjercicio", peso.getText().toString());
        mapa.put("repeticionesEjercicio", repeticiones.getText().toString());
        mapa.put("seriesEjercicio", series.getText().toString());
        mapa.put("fechaEjercicio", fecha.toString());
        String[] splitFecha = fecha.split( regex: "/");
        StringBuilder fechaBD = new StringBuilder("");
        fechaBD.append(splitFecha[0]).append("-").append(splitFecha[1]).append("-").append(splitFecha[2]);

        // Guardar los datos en la base de datos
        mDatabase.child( pathString: "Ejercicios").child(idUsuario).child(fechaBD.toString()).child(tituloNombreEjercicio.getText().toString()).setValue(mapa);
    } else {
        AlertDialog.Builder builder = new AlertDialog.Builder(requireContext());
        builder.setTitle("Error")
                .setMessage("Datos mal introducidos")
                .setPositiveButton( text: "Aceptar",  listener: null);
        AlertDialog dialog = builder.create();
        dialog.show();
    }
}

}
```

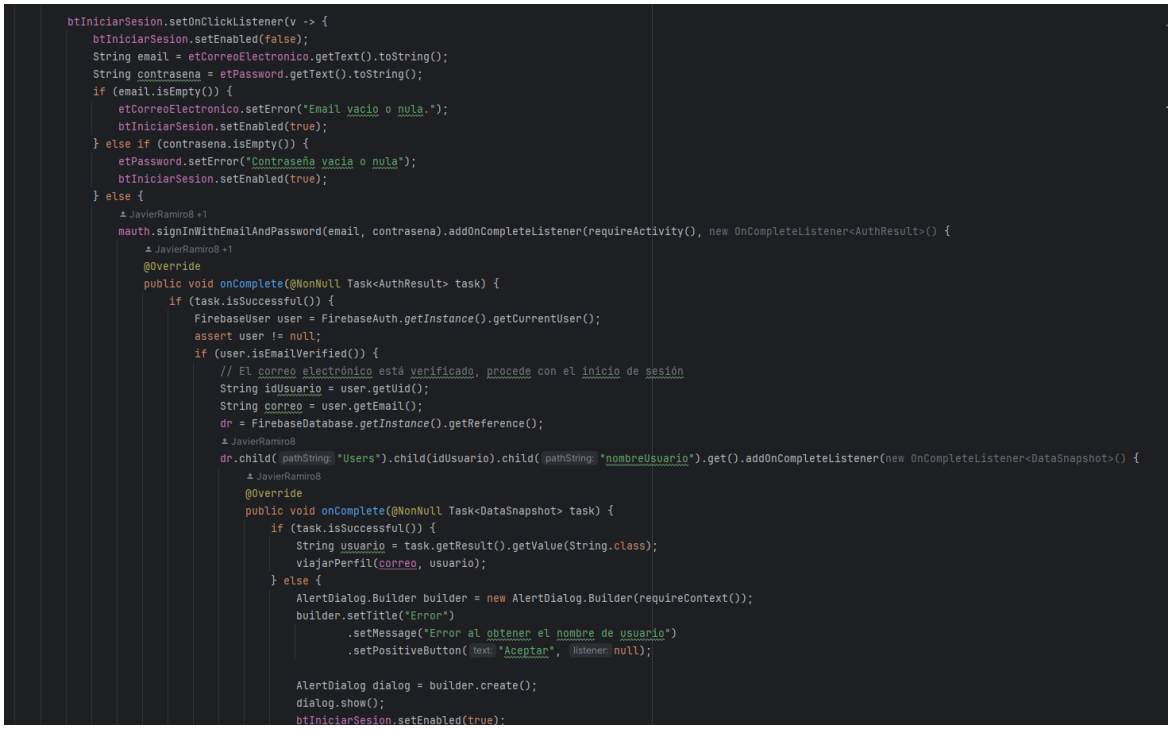
- Una vez creados los historiales se recogerán los datos mediante la fecha de la siguiente manera, para posteriormente mostrarlos en la aplicación



```
filtrarFecha.setOnClickListener(v -> {
    MaterialDatePicker<Long> materialDatePicker = MaterialDatePicker.Builder.datePicker()
            .setTitleText("Selecciona una fecha")
            .setSelection(MaterialDatePicker.todayInUtcMilliseconds())
            .setCalendarConstraints(new CalendarConstraints.Builder()
                    .setValidator(DateValidatorPointBackward.now())
                    .build())
            .build();
    materialDatePicker.addOnPositiveButtonClickListener(selection -> {
        SimpleDateFormat dateFormat = new SimpleDateFormat( pattern: "dd/MM/yyyy", Locale.getDefault());
        fechaFiltrada = dateFormat.format(new Date(selection));
        String[] splitFecha = fechaFiltrada.split( regex: "/");
        StringBuilder fechaFormateada = new StringBuilder();
        fechaFormateada.append(splitFecha[0]).append("-").append(splitFecha[1]).append("-").append(splitFecha[2]);
        filtrarFecha.setText("Fecha: " + fechaFiltrada);

        // Filtrar los datos en Firebase por la fecha seleccionada
        DatabaseReference ref = mDatabase.child( pathString: "Ejercicios").child(idUsuario).child(fechaFormateada.toString());
        Alberto Moreno Arcos
        ref.addValueEventListener(new ValueEventListener() {
            2 usages  Alberto Moreno Arcos
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {

                List<Historial> listaHistorial = new ArrayList<>();
                if (dataSnapshot.exists()) {
                    for (DataSnapshot ejercicioSnapshot : dataSnapshot.getChildren()) {
                        String nombreEjercicio = ejercicioSnapshot.getKey();
                        String peso = ejercicioSnapshot.child( path: "pesoEjercicio").getValue(String.class);
                        String repeticiones = ejercicioSnapshot.child( path: "repeticionesEjercicio").getValue(String.class);
                        String series = ejercicioSnapshot.child( path: "seriesEjercicio").getValue(String.class);
                        String fecha = ejercicioSnapshot.child( path: "fechaEjercicio").getValue(String.class);
                        if(nombreEjercicio.equals(tituloNombreEjercicio.getText().toString())){
                            agregarHistorial(nombreEjercicio, Integer.parseInt(peso), Integer.parseInt(repeticiones), Integer.parseInt(series), fecha, idUsuario);
                            contadorDatos++;
                        }
                        if(contadorDatos == 0){
```

-También se mostrará la implementación de la lógica de registro y login de los usuarios mediante Firebase y Real time Database.

- Para iniciar sesión se creará un elemento de Firebase Authentication con el que se podrán tratar todos los datos necesarios (email, contraseña) para el correcto acceso de usuarios.



```
btIniciarSesion.setOnClickListener(v -> {
    btIniciarSesion.setEnabled(false);
    String email = etCorreoElectronico.getText().toString();
    String contrasena = etPassword.getText().toString();
    if (email.isEmpty()) {
        etCorreoElectronico.setError("Email vacio o nula.");
        btIniciarSesion.setEnabled(true);
    } else if (contrasena.isEmpty()) {
        etPassword.setError("Contraseña vacia o nula");
        btIniciarSesion.setEnabled(true);
    } else {
        JavierRamiro8 +1
        mauth.signInWithEmailAndPassword(email, contrasena).addOnCompleteListener(requireActivity(), new OnCompleteListener<AuthResult>() {
            JavierRamiro8 +1
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
                    assert user != null;
                    if (user.isEmailVerified()) {
                        // El correo electrónico está verificado, procede con el inicio de sesión
                        String idUsuario = user.getUid();
                        String correo = user.getEmail();
                        dr = FirebaseDatabase.getInstance().getReference();
                        JavierRamiro8
                        dr.child( pathString: "Users").child(idUsuario).child( pathString: "nombreUsuario").get().addOnCompleteListener(new OnCompleteListener<DataSnapshot>() {
                            JavierRamiro8
                            @Override
                            public void onComplete(@NonNull Task<DataSnapshot> task) {
                                if (task.isSuccessful()) {
                                    String usuario = task.getResult().getValue(String.class);
                                    viajarPerfil(correo, usuario);
                                } else {
                                    AlertDialog.Builder builder = new AlertDialog.Builder(requireContext());
                                    builder.setTitle("Error")
                                            .setMessage("Error al obtener el nombre de usuario")
                                            .setPositiveButton( text: "Aceptar",  listener: null);

                                    AlertDialog dialog = builder.create();
                                    dialog.show();
                                    btIniciarSesion.setEnabled(true);
```

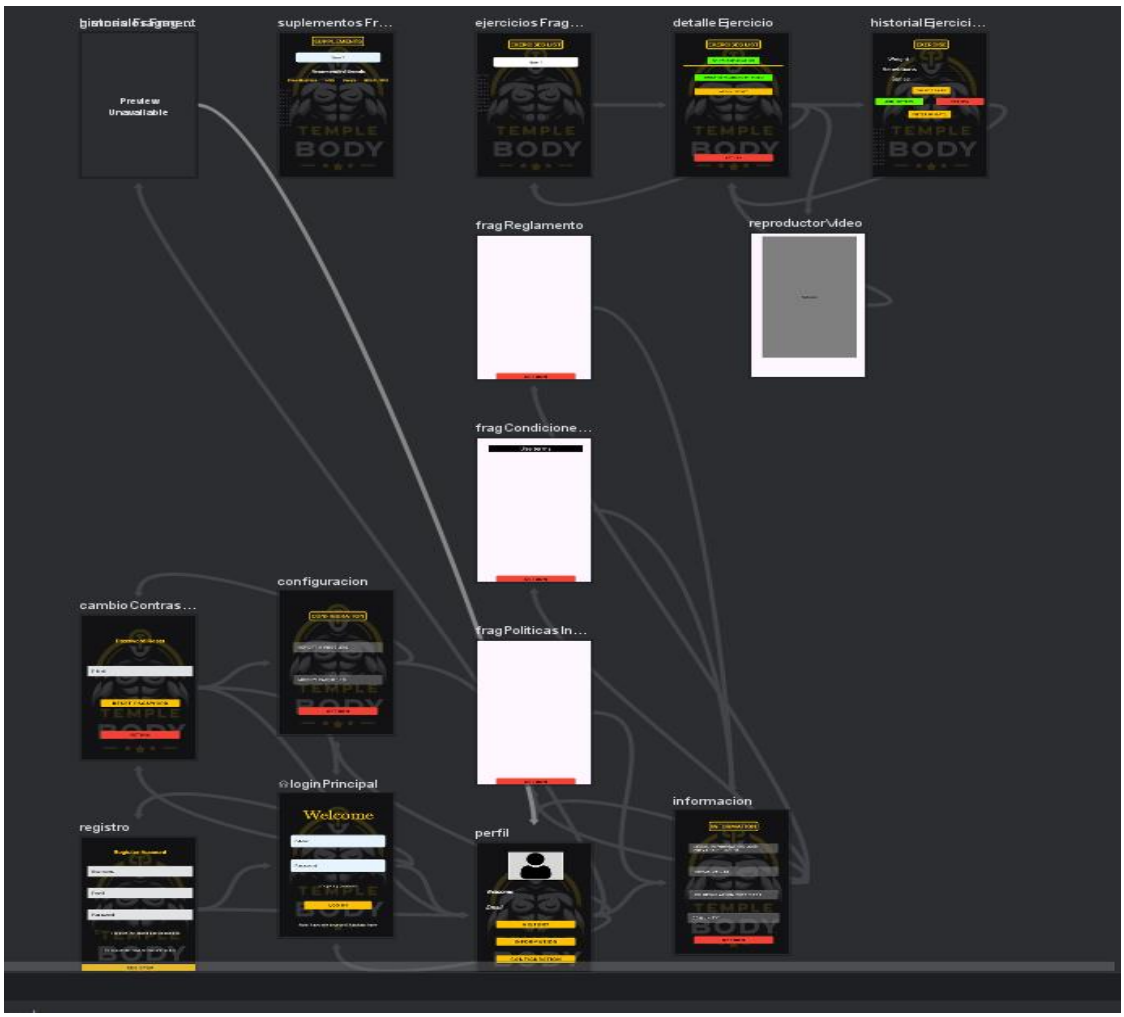
- Para la creación de usuarios se realizará un proceso parecido al registro de historiales donde se recogerán los datos del usuario y se almacenarán en la base de datos mediante un objeto Map.

```
private void registrarUsuario() {
    JavierRamiro8
    mauth.createUserWithEmailAndPassword(email.getText().toString(), contrasena.getText().toString()).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
        JavierRamiro8
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful()) {
                String userId = mauth.getCurrentUser().getUid(); // Obtener el ID de usuario generado por Firebase
                Map<String, Object> mapa = new HashMap<>();
                mapa.put("nombreUsuario", usuario.getText().toString());
                mapa.put("email", email.getText().toString());
                mapa.put("contraseña", contrasena.getText().toString());

                // Guardar los datos en la base de datos con el ID de usuario
                mDatabase.child( pathString: "Users").child(userId).setValue(mapa);
                ventanaEmergenteAvisoEmailVer();
                FirebaseUser user=mauth.getCurrentUser();
                user.sendEmailVerification();
                viajarLogin();
            } else {
                AlertDialog.Builder builder = new AlertDialog.Builder(requireContext());
                builder.setTitle("Error")
                        .setMessage("Error, es posible que ya se esté usando este mismo correo, Intentelo de nuevo!")
                        .setPositiveButton( text: "Aceptar",  listener: null);

                AlertDialog dialog = builder.create();
                dialog.show();
            }
        }
```

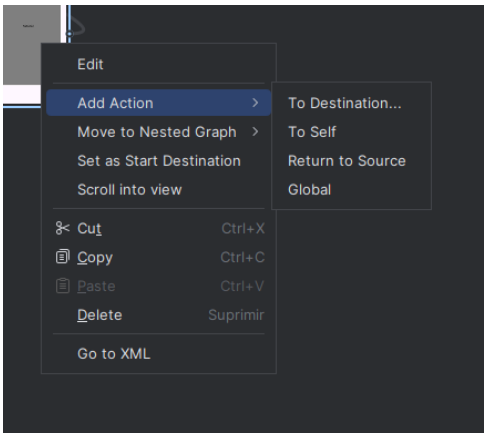
- En este apartado mostraremos un poco la navegación mediante Kotlin y java con el objeto NavController que nos redirigirá a los diferentes fragmentos y vistas de la aplicación, también se mostrara como se guardan las credenciales para que no se cierre la sesión al moverse entre diferentes vistas.

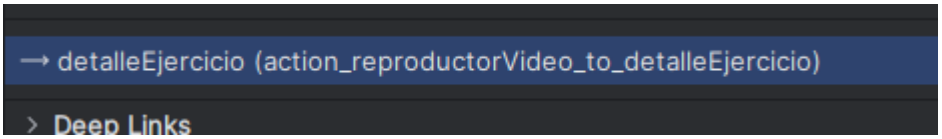
```
// Método para limpiar los datos del usuario de las preferencias compartidas
1 usage  JavierRamiro8
private void guardarCredencialesUsuario(String nombreUsuario, String correo) {
    SharedPreferences sharedPreferences = requireContext().getSharedPreferences( name: "datos_usuario", Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = sharedPreferences.edit();
    editor.putString("nombreUsuario", nombreUsuario);
    editor.putString("correo", correo);
    editor.apply();
}

1 usage  JavierRamiro8
private void guardarLayoutIdaVuelta() {
    SharedPreferences sharedPreferences = requireContext().getSharedPreferences( name: "cargaLayoutCambioContrasena", Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = sharedPreferences.edit();
    editor.putInt("layoutLoginCambio", 1);
    editor.apply();
}

1 usage  JavierRamiro8
private void guardarLayoutLogin() {
    SharedPreferences sharedPreferences = requireContext().getSharedPreferences( name: "cargaLayoutLogin", Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = sharedPreferences.edit();
    editor.putInt("cargaLayoutLogin", 1);
    editor.apply();
}

1 usage  JavierRamiro8
private void viajarRegistro() {
    NavController nav = NavHostFragment.findNavController( fragment: this);
    nav.navigate(R.id.action_loginPrincipal_to_registro);
}

1 usage  JavierRamiro8
private void viajarCambioContrasena() {
    NavController nav = NavHostFragment.findNavController( fragment: this);
    nav.navigate(R.id.action_loginPrincipal_to_cambioContrasena);
}
```

- Aquí se muestra el grafo.xml que diseñará el flujo de navegación entre diferentes vistas



- Para añadir nuevas conexiones tendremos que hacer click derecho en uno de los fragments y crearle una acción de navegación, esto nos creara un método que utilizaremos como previamente se ha descrito.





3.4 Fase de pruebas

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP001 – Movimiento entre Login, registro y perfil del usuario
- Descripción: Probar el correcto funcionamiento de movimiento entre el login, registro y perfil del usuario
- Condiciones de ejecución:
    - Al pulsar los botones de login o registro del usuario, que se remplace correctamente las distintas pantallas y el uso correcto de las funciones, además de mostrar el nombre del usuario y su email al logearse con el usuario en la pantalla del perfil
- Resultado esperado:

  El usuario debe ser redirigido a la página del login, registro del usuario, o a la página del perfil del usuario.
- Resultado obtenido:

  El usuario fue redirigido correctamente a las páginas anteriormente mencionada

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP002 – registro del usuario
- Descripción: Probar el correcto funcionamiento del uso del registro y comprobar el correcto guardado de los datos del usuario en la base de datos
- Condiciones de ejecución:
    - Al introducir el correo electrónico, usuario y contraseña, y haber leído los términos y condiciones, además del uso de un correo electrónico aun no registrado en nuestros sistemas.
- Resultado esperado:

  El usuario ha creado una cuenta en nuestros sistemas y se le ha enviado un correo para verificar que el correo usado es verídico
- Resultado obtenido:

  El usuario ha creado una cuenta con un correo aun no registrado en nuestros sistemas con un correo de verificación, una vez verificado puede el usuario logearse y usar todas las características de la aplicación

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP003 – Cambio de contraseña del usuario
- Descripción: Probar el cambio de contraseña del usuario en el apartado del perfil→ configuración y envío del correo para el cambio de contraseña
- Condiciones de ejecución:
    - Se introduce el correo electrónico
- Resultado esperado:

  Al introducir el correo electrónico del usuario, se le envía al usuario un correo con el cambio de contraseña
- Resultado obtenido:

  El usuario ha introducido su correo y se le ha enviado el correo de cambio de contraseña, el cual contiene un formulario para dicho cambio, se ha cambiado la contraseña y ya puede acceder con la nueva contraseña.

- Autor/Responsable: Alberto Moreno
- Caso de prueba: CP004 – Visualizador de gimnasios
- Descripción: Visualizar los gimnasios en formato lista, pulsar encima de un gimnasio a elección y que aparezca un mapa con los gimnasios con el nombre del seleccionado
- Condiciones de ejecución:
    - Pulsar encima de un gimnasio
- Resultado obtenido:

  Al pulsar encima de un gimnasio, se abre el Google Maps buscando los gimnasios con dicho nombre

- Autor/Responsable: Alberto Moreno
- Caso de prueba: CP005 – listado de suplementación
- Descripción: usar el filtro de suplementación por tipo de suplementos y poder buscar en las paginas web recomendadas dicha suplementación
- Condiciones de ejecución:
    - Filtrar la suplementación y pinchar en cualquiera de los enlaces de las marcas más recomendadas
- Resultado obtenido:

  Al filtrar por cualquier suplementación y pulsar cualquier enlace, este abre el navegador predeterminado del sistema.

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP006 – listado de ejercicios
- Descripción: usar el filtro de ejercicios por musculo a ejercitar, consumir la Api externa para los ejercicios de dicho musculo
- Resultado obtenido:
    - Filtrar los ejercicios por musculo y se mostrara los ejercicios con su dificultad y tipo de musculo

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP007 – comprobar descripción del ejercicio
- Descripción: al entrar a un ejercicio, hace la llamada a la API externa y mostrar la explicación textual del ejercicio.
- Resultado obtenido:
    - Al pulsar el botón de la explicación, funciona y consume de la API, el campo descripción

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP008 – comprobar video del ejercicio
- Descripción: al entrar a un ejercicio pulsar el botón del video y que aparezca una nueva pantalla con el video reproduciéndose, siendo este recogido por un servidor FTP
- Resultado esperado:
    - Al pulsar el botón del video, el video se reproduce para ver la realización del ejercicio
- Resultado obtenido:
    - Al pulsar el botón del video, el video no se reproduce para ver la realización del ejercicio
- Resolución del problema:
    - Al pulsar el botón del video, el video se reproduce gracias a los enlaces embebidos de YouTube, así usando el reproductor del video de YouTube para ver la realización del ejercicio

- Autor/Responsable: Alberto Moreno
- Caso de prueba: CP009 – Uso del histórico
- Descripción: pulsar el botón del histórico y rellenar los campos como la fecha, peso, series y repeticiones
- Resultado obtenido:
    - Se ha creado el ejercicio en la base de datos y se muestra al usuario.

- Autor/Responsable: Alberto Moreno
- Caso de prueba: CP010 – Uso del histórico general del usuario
- Descripción: pulsar el botón del histórico en la pestaña del perfil y rellenar los campos como la fecha, peso, series y repeticiones
- Resultado esperado:
    - Se ha de mostrar el histórico de todos los ejercicios hechos en un día impuesto por el usuario gracias a un selector de fechas.
- Resultado obtenido:
    - Se ha mostrado los ejercicios de todos los usuarios del sistema, haciendo así que la aplicación deje de responder o que tarde demasiado en volver a usarse
- Resolución del problema:
    - Modificar la formación de la base de datos, haciendo que el guardado de los ejercicios del usuario, sea guardado primero por el id del usuario y después por la fecha que se ha realizado ciertos ejercicios.

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP011 – Uso del cambio de contraseña
- Descripción: pulsar el botón del cambio de contraseña en la pestaña del login y poner el email del usuario
- Resultado esperado:
    - Se ha enviado la solicitud del cambio de contraseña y ha devuelto al login
- Resultado obtenido:
    - Se ha enviado el correo para el cambio de contraseña, y ha devuelto a la pestaña de configuración del usuario, en el perfil
- Resolución del problema:
    - Guardamos el estado de la parte donde se habría activado el cambio de contraseña, si es desde el login o desde el perfil del usuario.

- Autor/Responsable: Javier Ramiro
- Caso de prueba: CP012 – Login y cambio de pantalla
- Descripción: logearse en la aplicación y usar las características comunes de la aplicación
- Resultado esperado:
    - El usuario rellena los datos del login del usuario, después usar la función de los ejercicios y mantiene el usuario
- Resultado obtenido:
    - El usuario rellena los datos del login del usuario, después usar la función de los ejercicios y no mantiene el usuario activado, haciendo así que pueda fallar la aplicación ya que al guardar el histórico, este no recoge el id del usuario, haciendo que pueda fallar.
- Resolución del problema:
    - Guardamos el estado de todas las pantallas del login, registro y perfil del usuario, haciendo así que el usuario no tenga la necesidad de estar logeando cada vez que vaya a la opción del perfil del usuario.
    -

4 FUENTES

Páginas web consultadas:

- https://cs.uns.edu.ar/~ldm/mypage/data/oc/info/guia_para_la_documentacion_de_proyectos_de_software.pdf
- https://square.github.io/retrofit/
- https://firebase.google.com/docs?hl=es

5 INTEGRACIONES FUTURAS

Uso de Tarjetas NFC para Almacenar Datos de Usuario

La integración de tarjetas NFC (Near Field Communication) en Temple Body permitirá a los usuarios almacenar y acceder a sus datos de manera rápida y conveniente. Esta tecnología ofrecerá varios beneficios clave:

- Inicio de sesión rápido y seguro: Los usuarios podrán iniciar sesión en diferentes dispositivos sin necesidad de introducir manualmente sus datos. Simplemente acercando la tarjeta NFC al dispositivo, la aplicación leerá la información necesaria para autenticarlos.

- Portabilidad de datos: Los usuarios pueden llevar consigo sus datos de perfil, historial de entrenamiento y preferencias en una tarjeta NFC, lo que facilita el acceso y la actualización de estos datos en cualquier lugar y momento.
- Mejora en la experiencia del usuario: La simplicidad y conveniencia del uso de NFC mejorarán significativamente la experiencia del usuario, haciendo que la gestión de sus datos sea más rápida y sin complicaciones.

Integración con Aplicaciones para Llevar Alimentación y Conteo de Calorías

Integrar Temple Body con aplicaciones populares de alimentación y conteo de calorías permitirá a los usuarios gestionar de manera integral su régimen de ejercicios y su dieta. Esta funcionalidad incluirá:

- API de aplicaciones de alimentación y conteo de calorías: Utilizar las APIs de aplicaciones populares como MyFitnessPal, Lose It, y Fitbit para sincronizar datos de alimentación y actividad.
- Permisos y autenticaciones: Implementar los protocolos de autenticación necesarios (como OAuth) para acceder de manera segura a los datos del usuario en estas aplicaciones.
- Sincronización de datos de salud: Conectar Temple Body con dispositivos de seguimiento de salud como Apple Watch, Garmin, y dispositivos Fitbit para importar datos de frecuencia cardíaca, patrones de sueño y actividad física diaria.
- Visualización y análisis de datos: Proporcionar una vista unificada donde los usuarios pueden ver su ingesta calórica, nutrientes consumidos, y datos de salud junto con su progreso en ejercicios.

Integración con Redes Sociales

La integración con redes sociales permitirá a los usuarios compartir sus logros y progreso, así como interactuar con otros usuarios, fomentando una comunidad activa y motivada:

- Compartir en redes sociales: Los usuarios podrán compartir automáticamente sus logros, como objetivos de ejercicio alcanzados, rutinas completadas y mejoras en su rendimiento, en plataformas como Facebook, Instagram y Twitter.
- Comunidad dentro de Temple Body: Crear una sección comunitaria en la aplicación donde los usuarios pueden interactuar, compartir consejos, experiencias y motivarse mutuamente. Esto puede incluir foros, chats grupales y eventos comunitarios.

Coaching Personalizado Basado en IA

La inteligencia artificial puede llevar la personalización de la experiencia de Temple Body a un nuevo nivel, proporcionando recomendaciones y coaching personalizados:

- Recomendaciones personalizadas de ejercicios y dieta: Utilizar algoritmos de IA para analizar los datos del usuario y proporcionar recomendaciones específicas sobre ejercicios y planes de alimentación que se adapten a sus objetivos y progreso.
- Análisis de datos del usuario: Analizar patrones de uso, historial de entrenamiento, datos de salud y preferencias dietéticas para ofrecer sugerencias precisas y oportunas.
- Planificación y seguimiento adaptativo: Ajustar automáticamente las rutinas de entrenamiento y planes de dieta en función del rendimiento y progreso del usuario, asegurando que siempre estén en el camino correcto para alcanzar sus objetivos.

Integración Futura con Tecnología Wearable

Para continuar mejorando la experiencia del usuario y ofrecer funcionalidades avanzadas, Temple Body puede considerar la integración con tecnología wearable:

- Monitoreo en tiempo real: Integrar con wearables para proporcionar monitoreo en tiempo real durante los ejercicios, ofreciendo feedback instantáneo sobre la técnica y el rendimiento.
- Alertas y notificaciones personalizadas: Enviar alertas y notificaciones personalizadas basadas en los datos recolectados por los wearables, como recordatorios de hidratación, sugerencias de descanso y alertas de sobreesfuerzo.
- Análisis avanzado de rendimiento: Utilizar los datos de wearables para realizar análisis avanzados de rendimiento, identificando áreas de mejora y optimizando los programas de entrenamiento.

Soporte Multiplataforma

Para ampliar el alcance de Temple Body y ofrecer una experiencia consistente en múltiples dispositivos:

- Aplicación para iOS: Desarrollar una versión de Temple Body para dispositivos iOS, asegurando que las funcionalidades principales, como la integración con NFC y dispositivos de seguimiento de salud, estén disponibles para usuarios de Apple.

- Aplicación Web: Crear una versión web de Temple Body para que los usuarios puedan acceder a sus datos y funcionalidades desde cualquier navegador, proporcionando flexibilidad y accesibilidad.
- Compatibilidad con Smart TVs: Permitir a los usuarios seguir sus rutinas de ejercicio y ver videos tutoriales en pantallas grandes, integrando la aplicación con plataformas de Smart TV.

Estas mejoras y expansiones potenciales aseguran que Temple Body no solo se mantenga relevante en un mercado competitivo, sino que también brinde un valor continuo a sus usuarios a través de innovaciones tecnológicas y una experiencia de usuario mejorada.