TrackUp

Trabajo realizado por:
Álvaro Muñoz Panadero, alumno de 2º DAM

ÍNDICE

1. Introducción

1.1 Presentación del proyecto

En el mundo actual, cada vez más personas buscan maneras efectivas de mejorar su salud y bienestar a través de la adopción de hábitos saludables. Sin embargo, muchas veces el desafío radica en la constancia y el seguimiento de estos hábitos a lo largo del tiempo.

Este proyecto tiene como objetivo el desarrollo de TrackUp, una aplicación web diseñada para ayudar a los usuario a establecer, monitorizar y mantener hábitos saludables de forma efectiva.

TrackUp proporciona a los usuarios una plataforma intuitiva que les permite registrar sus hábitos diarios, realizar un seguimiento de sus progresos y obtener estadísticas detalladas sobre su rendimiento a lo largo del tiempo. La aplicación está orientada a personas que buscan mejorar su calidad de vida mediante la implementación de prácticas saludables, como ejercicio físico, alimentación balanceada o descanso adecuado.

La funcionalidad de TrackUp incluye:

- Registro de hábitos: Los usuarios podrán añadir y seguir múltiples hábitos saludables, con la opción de personalizar cada uno según sus necesidades.
- Seguimiento diario: La aplicación permitirá a los usuarios registrar sus actividades diarias, garantizando de esta manera un seguimiento continuo.
- Estadísticas y gráficos: Los usuarios podrán visualizar su progreso mediante estadísticas y gráficos detallados, lo que les permitirá evaluar su evolución con dichos hábitos a lo largo del tiempo.

A través de la aplicación se pretende ofrecer a los usuarios las herramientas necesarias para fomentar la disciplina, motivación y compromiso con su salud. Al hacer el progreso de seguimiento más accesible e interactivo, se intenta facilitar el cambio de hábitos y la adopción de una vida más saludable.

Este proyecto busca no solo mejorar la adopción de hábitos saludables, sino también proporcionar un espacio de reflexión y análisis para que sean los propios usuarios los que puedan evaluar su propio progreso y ajustar sus objetivos dependiendo de sus necesidades y de cómo vayan progresando con los mismos.

1.2 Objetivos del proyecto

El objetivo principal que se busca con el desarrollo de esta aplicación es ayudar a los usuarios a adoptar y mantener hábitos saludables a lo largo del tiempo. A través de TrackUp, lo que se busca es ofrecer una herramienta sencilla e intuitiva que permita a los usuarios registrar, seguir y mejorar sus hábitos diarios. Los objetivos de este proyecto son los siguientes:

- Desarrollar una plataforma intuitiva: Se creará una interfaz fácil e intuitiva que permita a los usuarios registrar los hábitos de una manera rápida, accesible e intuitiva. La aplicación debe ser amigable para todo tipo de público, independientemente de su nivel de familiaridad con la tecnología.
- Implementar un sistema de seguimiento diario: Se permitirá a los usuarios que registren su progreso de una manera eficiente, facilitando la constancia de los hábitos y ofreciendo una forma simple de ver cómo avanzar en su camino hacia una vida más saludable y con unos buenos hábitos establecidos.
- Generar estadísticas y análisis: Desarrollar una funcionalidad que permita a los usuarios poder visualizar estadísticas detalladas y gráficos acerca de su rendimiento y constancia en el desarrollo de sus hábitos. Esto ayudará a los usuarios a identificar patrones, evaluar su propio progreso y poder ver cómo pueden mejorar.
- Fomentar la motivación y el compromiso: Crear una aplicación que inspire a los usuarios a ser constantes y comprometidos con sus objetivos de salud. La aplicación pretende promover y proyectar una actitud de disciplina hacia el usuario de la misma.
- Adaptabilidad a diferentes tipos de hábitos: Ofrecer la posibilidad de personalizar los hábitos que el usuario desea seguir, abarcando una amplia variedad de áreas como ejercicio físico, alimentación, descanso, meditación…
- Desarrollar una experiencia de usuario atractiva: Asegurar que la aplicación no sea solamente funcional, sino también atractiva visualmente (e intuitiva, claro), creando así al usuario una experiencia que los motive a interactuar con ella de forma diaria y así logren mantener constancia a la hora de llevar a cabo sus hábitos saludables.
- Recoger feedback de los usuarios: Implementar una forma sencilla para que los usuarios puedan proporcionar retroalimentación acerca de la aplicación, lo cuál nos podrá permitir saber en qué cosas podemos mejorar y qué necesidades reales tienen nuestros usuarios.

El cumplimiento de estos objetivos permitirá no solo crear una herramienta útil y efectiva, sino también generar un impacto positivo en la vida diaria de los usuarios de la aplicación, ayudándoles a alcanzar sus metas de salud y bienestar.

2. Análisis de requerimientos

2.1 Identificación de necesidades y requerimientos

Para llevar a cabo el desarrollo de TrackUp, es fundamental identificar tanto las necesidades del público objetivo como los requerimientos funcionales que permitirán cumplir con los objetivos del proyecto. Esta sección de la documentación recoge dichas necesidades y las traduce en características concretas que la aplicación implementará.

Necesidades detectadas:

- Seguimiento personalizado de hábitos personales: Los usuarios necesitan una forma sencilla y flexible de registrar sus propios hábitos, ajustados a sus metas personales.
- Motivación y constancia: Es normal que las personas abandonen sus hábitos saludables por fata de seguimiento o por pérdida de motivación. Es por ello que se necesita una herramienta que les permita ser constantes en el desarrollo de los mismos,
- Visualización del progreso: A los usuarios les motiva ver cómo están evolucionando con el tiempo, lo cuál les ayudará a ser más constantes y a estar más motivados por mantener sus hábitos en el tiempo.
- Simplicidad y facilidad de uso: La aplicación debe ser intuitiva y accesible para todo el mundo sin importar su experiencia tecnológica.
- Privacidad y seguridad: Es necesario asegurar la confidencialidad de los datos personales de los usuarios que hacen uso de nuestra aplicación.

Requerimientos funcionales:

- Registro e inicio de sesión para los usuarios
- Registro y edición de hábitos personalizados
- Edición o eliminación de hábitos existentes
- Seguimiento diario con casillas o marcadores de cumplimiento
- Sistema de estadísticas y gráficos (por semana, mes, etc.)
- Posibilidad de marcar hábitos como completados o fallidos (o en proceso)
- Interfaz atractiva
- Perfil para guardar el progreso individual de cada usuario

Requerimientos no funcionales:

- Rendimiento: La aplicación deber ser rápida y responder sin muchas demoras
- Seguridad: Se deben proteger los datos personales del usuario, ya que se almacenarán todos sus registros de hábitos y demás.

- Escalabilidad: El diseño debe permitir agregar nuevas funcionalidades a futuro (como retos, sistema de puntos, etc.)
- Accesibilidad: La aplicación deberá poder ser usada por personas con distintas capacidades y niveles de experiencia con la tecnología

2.2 Identificación de público

Para asegurar el éxito de TrackUp, es fundamental definir claramente a qué tipo de usuario va destinada dicha aplicación. Conocer las características, necesidades y hábitos del público objetivo permitirá adaptar el diseño, funcionalidad y experiencia del usuario de una forma óptima.

La aplicación está pensada para un público general, con especial enfoque en aquellas personas que:

- Desean mejorar su estilo de vida: Usuarios interesados en llevar una vida más saludable a través de la organización y el seguimiento de sus hábitos de forma diaria.
- Tienen metas personales específicas: Personas que quieren adquirir o mantener hábitos como hacer ejercicio, beber más agua, dormir mejor, leer, meditar, reducir el tiempo de uso de las pantallas…
- Buscan herramientas de organización personal: Usuarios aficionado a la productividad personal y al desarrollo personal.
- Valoran la visualización del progreso: Personas que se sienten más gratificadas y motivadas al ver los datos, estadísticas o gráficos acerca de su progresión.
- Tienen entre 14 y 45 años, siendo este el rango más común en el uso de aplicaciones web y móviles de estilo de vida y salud (buscar información que corrobore este dato).
- Poseen conocimientos básicos de tecnología y están algo familiarizados con el uso de smartphones o similares.

Perfil de usuario ideal: cualquier persona que busque mejorar en algún aspecto de su vida y quiera llevar un registro de su progreso.

Ejemplo de usuarios potenciales:

- Estudiantes que quieren organizarse mejor y mantener una rutina saludable durante el curso
- Trabajadores que buscan equilibrar su vida profesional y personal con hábitos saludables
- Personas que inician un cambio de estilo de vida (dieta, deporte, descanso, etc.)
- Usuarios con metas concretas como leer más, dejar de fumar, meditar todos los días…

2.3 Estudio de mercado y competencia

Antes de empezar a desarrollar una aplicación es importante analizar el mercado actual y conocer las alternativas ya existentes. Este estudio permite identificar oportunidades de mejora, detectar necesidades no cubiertas y definir un enfoque único que diferencie TrackUp de la competencia.

En los últimos años, ha habido un aumento significativo en el uso de aplicaciones móviles relacionadas con la salud, el bienestar y la productividad personal. Este crecimiento se debe a:

- Mayor concienciación de la población acerca de la importancia de los hábitos personales.
- Popularización de los retos de 21/30 días y autoayuda.
- Interés en mejorar la organización personal y reducir el estrés diario.
- Uso creciente de herramientas digitales para el desarrollo personal.

Este contexto crea un entorno favorable para lanzar una aplicación enfocada en el seguimiento de hábitos, especialmente si dicha aplicación es intuitiva, atractiva y ofrece un valor real al usuario.

A continuación, se presentan algunas de las alternativas de las aplicaciones más populares actualmente en el mercado:

Nombre	Características destacadas	Puntos fuertes	Puntos débiles
Habitica	Seguimiento de hábitos gamificado, estilo RPG	Muy motivadora y con una comunida activa	Puede resultar un poco compleja para algunos usuarios
Habit	Aplicación minimalista y sencilla	Interfaz clara y sin distracciones	Opciones limitadas de personalización
Fabulous	Enfoque científico y rutinas guiadas	Diseño atractivo y consejos diarios	Muchas opciones vienen solamente en la opción de pago
Loop Habit Tracker	Aplicación gratuita y Open Source	Muy personalizable y sin anuncios	Diseño simple, sin sincronización online
Done	Seguimiento positivo de los hábitos	Ideal para objetivos diarios	Limitada solamente a usuarios de iOS

Oportunidades detectadas:

- Diseño intermedio entre lo visualmente atractivo y lo minimalista para no abrumar al usuario.
- Mayor personalización de los hábitos y rutinas que otras aplicaciones que son más rígidas.
- Aplicación gratuita, evitando así barreras de entrada y haciendo que más usuarios se quieran decantar por nuestra aplicación.
- Estadísticas motivadoras y claras, sin necesidad de una excesiva complejidad.
- Enfoque en hábitos saludables variados (no solamente en dieta o ejercicio).

Por lo tanto, TrackUp busca combinar lo mejor de ambos mundos; una interfaz limpia y funcional con suficientes opciones para adaptarse a diferentes tipos de usuarios, sin caer en la sobrecarga de funciones ni depender de pagos para lo esencial. La clave está en ofrecer simplicidad, motivación y personalización.

3. Diseño y planificación

3.1 Definición de la arquitectura del proyecto

TrackUp está estructurado como una aplicación web con arquitectura cliente-servidor. El backend está desarrollado completamente en Java utilizando el framework Spring Boot, mientras que el frontend se construirá con HTML, CSS y posiblemente algo de JavaScript básico, todo integrado en un mismo proyecto. La base de datos utilizada para el entorno local será H2, una base de datos embebida en memoria ligera ideal para entornos de desarrollo y pruebas.

Componentes principales:

- Backend (Java):
  - Controladores REST para manejar peticiones HTTP
  - Servicios para la lógica de negocio
  - Repositorios (Spring Data JPA) para el acceso a la base de datos
  - Seguridad con Spring Security (autenticación y autorización con JWT)
  - Documentación de la API con Swagger/OpenAPI

- Frontend (HTML/CSS/JavaScript):
  - Vistas estéticas renderizadas desde el mismo servidor backend
  - Diseño limpio enfocado en la usabilidad

- Base de datos (H2):
  - H2 en modo local (modo consola activado para depuración)
  - Esquema definido y limpio enfocado en la usabilidad

- Testing y herramientas de desarrollo:
  - Postman para pruebas manuales de los endpoints
  - Swagger UI para la documentación interactiva de la API

3.2 Diseño de la interfaz de usuario

El diseño de la interfaz está enfocado en ofrecer una experiencia clara, sencilla e intuitiva, manteniendo una estética minimalista que facilite el seguimiento de los hábitos del usuario. Las vistas están organizadas por entidades del sistema, lo que facilita la navegación y gestión de los datos.

Principales vistas de la aplicación web:

- Pantalla de inicio de sesión y registro: Formulario para autenticar o registrar nuevos usuarios, con validación de datos y mensajes claros de error.
- Dashboard principal: Vista general del progreso diario y semanal, mostrando estadísticas relevantes de hábitos completados, próximos objetivos y resumen motivacional.
- Gestión de hábitos: Vista para listar, crear, editar y eliminar hábitos personalizados del usuario.
- Gestión de tipos de hábitos: Pantalla para administrar las categorías o clasificaciones de hábitos, permitiendo una organización más clara (por ejemplo: salud, estudio, ocio).
- Registros diarios: Interfaz para añadir y visualizar registros del cumplimiento de hábitos día a día, con controles intuitivos y navegación por calendario.
- Gestión de metas personales: Formulario y listado para definir metas asociadas a hábitos, con fechas límite y estados de progreso.
- Perfil del usuario: Sección donde el usuario puede ver y editar su información personal, cambiar su contraseña y consultar su configuración.
- Sección “Sobre mí”: Vista estática obligatoria, donde el desarrollador explica los objetivos del proyecto, su propósito personal y profesional, y otros datos relevantes a modo de presentación personal.

Estilo visual y experiencia de usuario:

- Colores suaves y neutros, que transmitan calma y motivación.
- Tipografía legible y jerarquía clara de información.
- Uso de iconos y elementos gráficos para facilitar la comprensión rápida de cada sección.
- Diseño responsive para adaptarse correctamente.
- Integración de gráficos para mostrar el progreso, usando bibliotecas como Chart.js.

3.3 Planificación de tareas y los recursos necesarios

Desde el inicio del proyecto, el enfoque principal ha sido el desarrollo de la API REST con Spring Boot y la definición de los endpoints y entidades básicas. A continuación, se describen los objetivos y entregables previstos para las próximas semanas:

Periodo	Objetivos	Entregables
Desde el inicio del proyecto – 5 de mayo	Desarrollo de la API REST completa con Spring Boot	Proyecto Spring Boot con endpoints CRUD de User, Habit, HabitType, DailyRecord y Goal funcionando perfectamente
5 de mayo – 19 de mayo	Implementación de la seguridad con Spring Security y JWT, desarrollo del frontend	Módulos de autenticación y autorización seguros, vistas HTML/CSS conectadas a la API
19 de mayo – 2 de junio	Últimos retoques del proyecto y de cosas a mejorar	Backend y frontend finalizado al completo, manuales de usuario, documentaciones, pruebas finales...

Recursos utilizados:

- Lenguajes de programación: Java, HTML, CSS y JavaScript.
- Frameworks y librerías: Spring Boot, Spring Security, Spring Data JPA, Swagger UI y Chart.js.
- Herramientas de desarrollo: IntellijIDEA Ultimate, Postman, Git, GitHub, DBeaver, Swagger UI.

4. Implementación y pruebas

4.1 Desarrollo de las funcionalidades del proyecto

Durante la implementación, se ha seguido una estructura modular basada en servicios. Hasta el momento, el trabajo se ha centrado principalmente en el desarrollo del backend con Spring Boot y la construcción de una API REST funcional. A continuación se detallan los avances actuales:

Backend:

- Modelo de datos: Se han definido entidades como User, Habit, HabitType, HabitRecord y Goal, todas gestionadas mediante JPA y mapeadas con relaciones.
- Endpoints REST:
  - /api/users: Para gestionar la información del usuario, incluyendo la actualización de datos personales.
  - /api/habits: Para crear, editar y eliminar los hábitos, permitiendo que los usuarios personalicen completamente su experiencia.
  - /api/habit-types: Para gestionar los tipos de hábitos, lo cual permitirá a los usuarios organizar sus hábitos por categorías (por ejemplo, salud, ejercicio, descanso).
  - /api/daily-records: Para que los usuarios registren su progreso diario en los hábitos establecidos.
  - /api/goals: Para crear y gestionar objetivos específicos asociados a hábitos, como metas de consumo de agua o tiempo de ejercicio.
- Seguridad: Se ha integrado Spring Security con JWT para autenticar y autorizar todas las peticiones al backend.
  - Configuración de filtros JWT para validar tokens en cada solicitud.
  - Definición de roles (ROLE_USER, ROLE_ADMIN) y restricciones de acceso a endpoints según rol.
  - Endpoints de login y refresh-token disponibles en /api/auth.
- Documentación técnica: La API está documentada utilizando Swagger/OpenAPI, lo que facilita la interacción con los endpoints y la prueba manual de los servicios. Swagger se ha integrado en la aplicación para que los desarrolladores y testers puedan probar fácilmente la API.
- Pruebas unitarias: Se han realizado pruebas unitarias mediante JUnit y Mockito para los servicios de todas las entidades.

4.2 Pruebas unitarias y de integración

Dado que la fase de desarrollo ha avanzado significativamente, se han completado todas las pruebas unitarias para garantizar la calidad del código en las funcionalidades clave de la aplicación. Estas pruebas cubren de forma integral las capas más críticas del sistema, asegurando que las operaciones esenciales funcionan según lo esperado.

Las pruebas unitarias se centran en las siguientes áreas:

- Pruebas de los controladores REST: Se han probado las respuestas de los endpoints clave, como la creación de un nuevo usuario, la creación de hábitos y la gestión de registros diarios. Estas pruebas verifican que las respuestas sean correctas, incluyendo códigos de estado HTTP apropiados y los datos correctos en las respuestas. Por lo tanto, actualmente todos y cada uno de los endpoints funcionan correctamente.
- Pruebas de la capa de servicio: Para asegurar la correcta lógica de negocio, se han probado las funcionalidades que gestionan la creación, actualización y eliminación de los hábitos, así como la validación de las metas personales.
- Pruebas de la capa de acceso a datos (repositorios): Se ha verificado que las consultas a la base de datos funcionen correctamente. Las pruebas incluyen la creación y recuperación de hábitos y registros asociados a los usuarios.
- Pruebas de seguridad: Se han implementado y validado las funcionalidades de autenticación y autorización mediante Spring Security y tokens JWT. Estas pruebas incluyen la generación y validación de tokens, la protección de endpoints sensibles y la verificación de roles de usuario.

Gracias a estas pruebas, se ha asegurado la estabilidad de las funcionalidades críticas y la correcta integración entre las distintas capas de la aplicación. En futuras entregas, se enfocarán en pruebas adicionales para mejorar la cobertura, como el manejo de errores no previstos y la validación exhaustiva de entradas de usuario.

4.3 Corrección de errores y optimización del rendimiento

Durante el proceso de desarrollo y pruebas iniciales, se han identificado y corregido varios errores relacionados con la funcionalidad de la aplicación. Estos incluyen:

- Problemas de validación de datos: Se detectaron casos en los que la aplicación no validaba correctamente las entradas del usuario, lo que podía llevar a errores en la base de datos. Estos problemas han sido solucionados y ahora los formularios de entrada de datos incluyen validaciones adecuadas para asegurar la integridad de la información.
- Rendimiento de la base de datos: Durante las pruebas de carga, se observó que algunas consultas a la base de datos podían ser más lentas de lo esperado. Se han optimizado estas consultas para reducir los tiempos de respuesta y mejorar la eficiencia del sistema.
- Implementación y corrección de seguridad : Se ha integrado Spring Security con tokens JWT para garantizar la autenticación y autorización seguras. Se han corregido errores en la gestión de tokens (ej.: expiración incorrecta, generación de claves débiles) y se han reforzado los endpoints sensibles para evitar accesos no autorizados.

Gracias a estas mejoras, la aplicación ahora cuenta con un sistema de seguridad robusto, validaciones de datos completas y una base de datos optimizada. El frontend sigue en fase de desarrollo, enfocado en ofrecer una interfaz atractiva y responsive, pero aún no está integrado con el backend.

5. Documentación

5.1 Documentación técnica

La documentación técnica de TrackUp incluye los siguientes elementos clave:

- Swagger/OpenAPI:
  - Se integra mediante la dependencia ‘springdoc-openapi-ui’ en ‘pom.xml’.
  - Al arrancar la aplicación (‘mvn spring-boot:run), la interfaz interactiva está en ‘http://localhost:8080/swagger-ui.html’.
  - Ahí están descritos todos los endpoints, sus modelos de datos (User, Habit, HabitType, DailyRecord, Goal), ejemplos de petición/respuesta y el esquema de seguridad JWT (botón Authorize).
- Anotaciones en el código:
  - Controladores etiquetados con @Tag para agrupar rutas y @Operation para cada método, incluyendo descripciones y códigos de respuesta con @ApiResponse.
  - Modelos y DTOs con @Schema en campos relevantes para que Swagger genere descripciones y ejemplos automáticos.
- Base de datos:
  - Modelo relacional con tablas para usuarios, hábitos, registros diarios y categorías.
  - Scripts SQL para la creación y migración de tablas (próximamente).
- Seguridad:
  - Configuración detallada de Spring Security con autenticación basada en tokens JWT.
  - Rutas protegidas y roles de usuario (próximamente).
- Estructura del código:
  - Organización por paquetes (controller, dto (request y response), entity, repository, security, services).
  - Uso de DTOs para la transeferencia segura de datos entre las diferentes capas de la API REST.

5.2 Documentación de usuario final

Se ha iniciado la redacción de la documentación para los usuarios finales, que incluirá:

- Manual de uso (en proceso):
  - Guía paso a paso para registrar hábitos, marcar cumplimientos diarios y visualizar estadísticas.
  - Instrucciones para gestionar el perfil del usuario y ajustar las preferencias.
- Preguntas frecuentes (FAQ) (próximamente):
  - Soluciones a errores comunes (ej.: problemas de autenticación, validación de datos).
  - Explicación de funcionalidades clave (ej.: cómo interpretar los gráficos de progreso).

5.3 Manual de instalación y configuración

El manual incluye (en proceso):

- Requisitos técnicos:
  - Backend: Java 21, Maven, H2 Spring Boot X…
  - Frontend (en progreso): HTML, CSS, JavaScript.
- Pasos para configurar el entorno:
  - Variables de entorno para la base de datos y claves JWT.
  - Dependencias necesarias (listadas en el archivo ‘pom.xml’ y ‘package.json’).
- Despliegue:
  - Configuración para poder ejecutar la aplicación en local (archivo ‘application.properties’).

6. Mantenimiento y evolución

6.1 Identificación de posibles mejoras y evolución del proyecto

Con el fin de incrementar la usabilidad y el valor añadido para los usuarios, se han identificado varias líneas de evolución que permitirán enriquecer la experiencia y ampliar las capacidades de TrackUp en futuras versiones:

- Recuperación de contraseña: añadir el endpoint POST ‘/api/auth/forgot-password’ y la lógica de envío de un enlace de restablecimiento mediante Spring Mail, facilitando la recuperación de cuentas.
- Administración de usuarios: crear un panel exclusivo para ‘ROLE_ADMIN’ con funcionalidades para listar usuarios, desactivar cuentas y reasignar roles directamente desde la interfaz web.
- Exportación de datos: diseñar un endpoint que permita descargar registros diarios y metas en CSV o JSON desde la UI, para que el usuario pueda guardar o analizar sus datos de forma externa.

6.2 Actualizaciones y mejoras futuras

Para garantizar la calidad, la seguridad y la facilidad de despliegue en cada nueva entrega, se propone incorporar las siguientes prácticas y herramientas que optimicen el mantenimiento continuo del proyecto:

- Gestión de dependencias: activar Dependabot o GitHub Security Alerts para mantener las librerías (Spring Boot, Spring Security, etc.) siempre al día y seguras.
- Script de arranque simplificado: crear ‘start.sh’ (y ‘start.bat’ opcional) que configure las variables de entorno necesarias y ejecute ‘mvn spring-boot:run’ o el JAR resultante en un solo paso.
- Integración de exportación en UI: añadir un botón en la interfaz que invoque el endpoint de exportación de datos y descargue automáticamente el fichero, mejorando la experiencia de usuario.
- Verificación de la documentación: antes de cada presentación o entrega, acceder a ‘http://localhost:8080/swagger-ui.html’ para validar que todos los endpoints, incluyendo los nuevos, estén correctamente descritos y operativos.

7. Conclusiones

7.1 Evaluación del proyecto

7.2 Cumplimiento de objetivos y requisitos

7.3 Lecciones aprendidas y recomendaciones para futuros proyectos

8. Bibliografía/Webgrafía y referencias

8.1 Fuentes utilizadas para el proyecto

https://www.choosingtherapy.com/best-online-therapy-platforms-that-take-insurance/
https://www.choosingtherapy.com/joon-app-review/
https://www.choosingtherapy.com/best-mindfulness-apps/
https://www.choosingtherapy.com/best-adhd-apps/
https://crm.org/news/habitify-review
https://www.xataka.com/basics/14-mejores-aplicaciones-monitorizar-tus-nuevos-habitos-android-iphone