TITULO DEL PROYECTO

## Propuesta de Proyecto DAM

APLICACIÓN MÓVIL DE BIENESTAR Y SALUD CON CHATBOT INTELIGENTE

## Autor

26587725R

## Desarrollo de aplicaciones Multiplataforma

Curso académico 2023-24

## Tutor de Proyecto

Elena Jiménez Fuentes

Contenido

[Propuesta de Proyecto DAM 1](#_Toc161684689)

[Autor 1](#_Toc161684690)

[Desarrollo de aplicaciones Multiplataforma 1](#_Toc161684691)

[Tutor de Proyecto 1](#_Toc161684692)

[1. Justificación y planificación del proyecto 4](#_Toc161684693)

[1.1.- Descripción del proyecto 4](#_Toc161684694)

[1.2.- Justificación de la tecnología utilizada 4](#_Toc161684695)

[1.3.- Planificación Gantt 5](#_Toc161684696)

[1.4.- Planificación Scrum 7](#_Toc161684697)

[1.5.- Prototipado 8](#_Toc161684698)

[2. Plan de empresa 11](#_Toc161684699)

[2.1.- Oportunidades de negocio y necesidades demandadas 11](#_Toc161684700)

[2.2.- Relación de productos o servicios a prestar 11](#_Toc161684701)

[2.3.- Constitución de la empresa 11](#_Toc161684702)

[2.4.- Recursos 12](#_Toc161684703)

[3. Análisis y diseño de los requisitos de la aplicación 12](#_Toc161684704)

[3.1.- Herramientas escogidas 12](#_Toc161684705)

[3.2.- Diagrama de casos de uso 13](#_Toc161684706)

[3.3.- Diagrama de clases 14](#_Toc161684707)

[3.4.- MER 16](#_Toc161684708)

[3.5.- Diagrama de Base de datos 17](#_Toc161684709)

[4. Desarrollo de la aplicación 18](#_Toc161684710)

[4.1.- Control de usuarios 18](#_Toc161684711)

[4.2.- Desarrollo e implementación de la aplicación 18](#_Toc161684712)

[5. Pruebas e informes de la aplicación 19](#_Toc161684713)

[5.1.- Estrategia de pruebas 19](#_Toc161684714)

[5.2.- Pruebas realizadas 19](#_Toc161684715)

[5.3.- Informes 19](#_Toc161684716)

[6. Implantación y documentación 20](#_Toc161684717)

[6.1.- Exportación y distribución de la aplicación 20](#_Toc161684718)

[6.2.- Manual de instalación 20](#_Toc161684719)

[6.3.- Manual de usuario 20](#_Toc161684720)

[6.4.- Repositorio de la aplicación 20](#_Toc161684721)

[6.5.- JavaDoc 20](#_Toc161684722)

[7.- Bibliografía 21](#_Toc161684723)

[8.- Anexos 22](#_Toc161684724)

# 1. Justificación y planificación del proyecto

### 1.1.- Descripción del proyecto

Mi proyecto se enfoca en crear una aplicación móvil orientada al bienestar y la salud. Lo especial de esta aplicación es que incluye un chatbot avanzado impulsado por inteligencia artificial (IA) y beneficiándose de la API de OpenAI, que ofrece consejos relacionados con la salud. Este chatbot utiliza tecnologías de procesamiento de lenguaje para entender y responder de manera efectiva a las preguntas de los usuarios, proporcionando recomendaciones útiles para llevar un estilo de vida saludable. Gracias a estas tecnologías, el chatbot puede mejorar continuamente su manera de interactuar, ofreciendo así una experiencia muy personalizada que promueve activamente el bienestar de los usuarios.

También voy a agregar funciones como recordatorios personalizados para acciones importantes como tomar medicamentos, beber agua o hacer ejercicio, además de un sistema para seguir metas y progresos en la salud. Estas características buscan motivar a los usuarios a alcanzar sus objetivos de bienestar, promoviendo hábitos saludables y brindando apoyo constante a través del chatbot.

Para finalizar, voy a añadir un sistema de *login* y registro para hacer la aplicación más completa y garantizar la privacidad de los usuarios. Es por ello por lo que usaré una API, creada por mí, para manejar no solo el *login* sino todas o la mayoría de las peticiones de la aplicación.

Para llevar a cabo este proyecto, combinaré conocimientos en desarrollo de aplicaciones móviles, IA y procesamiento de lenguaje natural. Mantendré un enfoque centrado en el usuario para asegurar que la aplicación sea fácil de usar y que brinde una experiencia positiva.

### 1.2.- Justificación de la tecnología utilizada

En la fase inicial de este proyecto, después de evaluar diversas opciones, he decidido utilizar .NET MAUI para el desarrollo de la aplicación móvil y Visual Studio 2022 para el backend y la API. Mi elección se basa en varios factores clave:

* **Experiencia Previa y Curva de Aprendizaje**: Aunque tengo experiencia en el uso de Android Studio, .NET MAUI representa una oportunidad para expandir mis habilidades hacia el desarrollo multiplataforma, abarcando tanto Android como iOS con un único código base. Esta decisión estratégica me permite no solo aprovechar mi base en programación orientada a objetos, sino también explorar el robusto ecosistema de .NET.
* **Rider sobre Visual Studio**: Mi familiaridad con IntelliJ me ha llevado a preferir Rider de JetBrains para el desarrollo. Considero que Rider ofrece una experiencia de usuario más optimizada, especialmente para aquellos que vienen de IntelliJ, sin sacrificar las capacidades y el soporte para .NET MAUI. Aunque Visual Studio es una herramienta poderosa y ampliamente adoptada, mi decisión de comenzar el desarrollo con Rider se basa en la búsqueda de una mayor eficiencia y comodidad personal en el flujo de trabajo.
* **Alcance y Flexibilidad**: Optar por .NET MAUI me permite dirigirme a una audiencia más amplia sin comprometer la calidad ni la funcionalidad de la aplicación. La flexibilidad y la capacidad de adaptación de .NET MAUI son cruciales para este proyecto, que aspira a mejorar el bienestar y la salud de los usuarios a través de una plataforma accesible y confiable.

### 1.3.- Planificación Gantt

Para la planificación Gantt usaré GanttProject cuyo objetivo es la administración de proyectos usando el diagrama de Gantt.

Esta planificación se dividirá en 6 partes fundamentales contando con el Estudio de Viabilidad previo:

Estudio de viablidad (14/03/24 – 18/03/24)

* Evaluar la factibilidad técnica y de mercado 🡪 **14/03/24 – 14/03/24**
* Investigar sobre la API de OpenAI y capacidades de .NET MAUI 🡪 **15/03/24 – 17/03/24**
* Interfaz de usuario gráfica, Aplicación

  Descripción generada automáticamenteAnálisis de la competencia y demanda del mercado 🡪 **18/03/24 – 18/03/24**

Fase 1: Análisis (19/03/24 – 6/04/24)

* Diagrama de casos de uso 🡪 **19/03/24 – 21/03/24**
* Diagrama de clases 🡪 **22/03/24 – 24/03/24**
* Diagrama MER 🡪 **25/03/24 – 27/03/24**
* Interfaz de usuario gráfica, Aplicación

  Descripción generada automáticamenteEspecificar las características del chatbot y su integración 🡪 **27/03/24 – 6/04/24**

Fase 2: Diseño (7/04/24 – 16/04/24)

* Diseñar la arquitectura de la aplicación y del chatbot 🡪 **7/04/24 – 9/04/24**
* Elaborar mockups y prototipos de la interfaz de usuario 🡪 **10/04/24 – 11/04/24**
* Gráfico

  Descripción generada automáticamente con confianza mediaDefinir la estrategia de respuesta del chatbot 🡪 **12/04/24 – 16/04/24**

Fase 3: Implementación (17/04/24 – 18/05/24)

* Codificación de la aplicación y el chatbot 🡪 **17/04/24 – 30/04/24**
* Integrar la API de OpenAI con la aplicación 🡪 **2/05/24 – 8/05/24**
* Interfaz de usuario gráfica, Aplicación

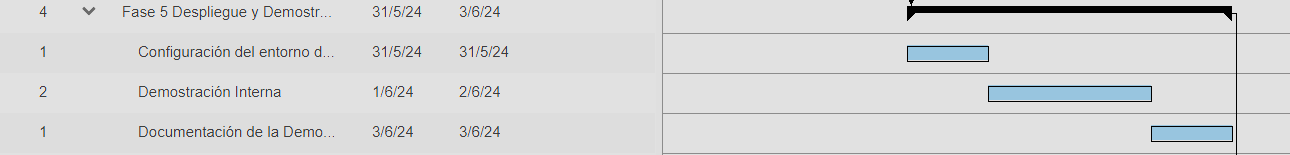
  Descripción generada automáticamenteDesarrollar funciones de recordatorios y seguimiento de metas 🡪 **9/05/24 – 18/05/24**

Fase 4: Pruebas (19/05/24 – 30/05/24)

* Realizar pruebas unitarias y de integración 🡪 **19/05/24 – 23/05/24**
* Pruebas de usabilidad y experiencia de usuario 🡪 **24/05/24 – 26/05/24**
* Imagen que contiene Tabla

  Descripción generada automáticamenteRefinar funcionalidades con base en el feedback 🡪 **27/05/24 – 30/05/24**

Fase 5: Despliegue y Demonstración (31/05/24 – 3/06/24)

* Configuración del entorno de demostración 🡪 **31/05/24 – 31/05/24**
* Demostración Interna 🡪 **1/06/24 – 2/06/24**
* Documentación de la Demostración 🡪 **3/06/24 – 3/06/24**

Fase 6: Documentación y Conclusión (4/06/24 – 12/06/24)

* Compilación de la Documentación del Proyecto 🡪 **4/06/24 – 4/06/24**
* Reflexión sobre el Proyecto y Lecciones Aprendidas 🡪 **5/06/24 – 6/06/24**
* Preparación para la Defensa del TFG 🡪 **7/06/24 – 12/06/24**

En mi proceso de planificación para este proyecto, me enfoqué en estructurar cada fase con tiempos estimados que reflejaran de manera realista lo que podría llevarme completar cada tarea. Utilizando GanttProject, pude distribuir de forma clara y ordenada todas las etapas del proyecto, desde el estudio de viabilidad hasta la documentación y conclusión.

Para ver el Gantt: <https://github.com/xDeew/EnlancesReferenciaTFG>

### 1.4.- Planificación Scrum

Por otro lado, tenemos la planificación Scrum, destaca por su capacidad para manejar la incertidumbre y los cambios rápidos. Scrum, con sus sprints cortos y su enfoque iterativo, me permite adaptar el desarrollo a medida que avanzo, garantizando que el producto final responda efectivamente a las necesidades reales de los usuarios y aproveche las últimas innovaciones tecnológicas.

Scrum general con todas las tareas: <https://trello.com/b/QEJrgnGj/app-bienestar>

Sprint 0 (Estudio Viabilidad): <https://trello.com/b/WoMfx3M9/sprint-0-estudio-de-viabilidad>

Sprint 1 (Análisis): <https://trello.com/b/pxCiZvZT/sprint-1-analisis>

Sprint 2 (Diseño): <https://trello.com/b/4SOly87O/sprint-2-diseno>

Sprint 3 (Implementación): <https://trello.com/b/HoR6V4iR/sprint-3-implementacion>

Sprint 4 (Pruebas): <https://trello.com/b/KtFDBW5g/sprint-4-pruebas>

Sprint 5 (Despliegue y Demonstración): <https://trello.com/b/qSNyufhA/sprint-5-despliegue-y-demonstracion>

Sprint 6 (Documentación y Conclusión): <https://trello.com/b/Nn3MBMn2/sprint-6-documentacion-y-conclusion>

### 1.5.- Prototipado

Para el prototipado me decanto por elaborar diseños usando Marvel app, una herramienta que nos permite agregar **interactividad a los prototipos** ya diseñados o diseñarlos directamente online para luego volverlos interactivos, permitiendo que tengamos una idea de cómo funcionará el proyecto y ayudar al cliente a que visualice y corrija la experiencia de uso y navegación de la web.

En el siguiente enlace podremos ver una aproximación a lo que será la aplicación final pero, lógicamente, sujeta a cambios.

<https://marvelapp.com/prototype/c683fg5/screen/94215242>

1. Página de iniciar sesión

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteEsta es la página por defecto que aparece nada más arrancar la aplicación. Se muestra un logo personalizado, las dos entradas para poner los datos del usuario, y en caso de no estar registrado se le proporciona un texto donde si le das te pasa a la página de registro.

1. Página de Registro

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamenteEn esta página se le permitirá al usuario registrarse proporcionando su nombre o nombre de usuario, email y contraseña.

1. Página de bienvenido

Imagen de la pantalla de un celular con letras

Descripción generada automáticamente con confianza mediaEsta es la página que aparecerá después de que el usuario inciara sesión con éxito. Mensaje de bienvenida con su nombre y diversas opciones, como ir directamente al chatbot o registrar una actividad, además de la posibilidad de desconectarse.

En el panel inferior vemos diferentes elementos que nos permitirán ir a varias páginas.

1. Página del chatbot

Imagen de la pantalla de un celular con letras

Descripción generada automáticamente con confianza mediaEn esta página el usuario podrá comunicarse con el chatbot únicamente con preguntas destinadas a la salud puesto que el comportamiento del chatbot esta únicamente dirigido a este ámbito.

1. Página de recordatorios

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Esta página se enfocará principalmente en que el usuario pueda elegir de las actividades disponibles, como beber agua, hacer deporte un recordatorio a una hora que ellos prefieran para que se les avise de que tienen que realizar la acción que previamente definieron.

# 2. Plan de empresa

### 2.1.- Oportunidades de negocio y necesidades demandadas

El mercado de aplicaciones móviles de bienestar y salud está en pleno auge, impulsado por una creciente conciencia sobre la importancia de mantener un estilo de vida saludable y la facilidad de acceso a tecnologías que facilitan este objetivo. La pandemia de COVID-19 ha acelerado la digitalización de la salud y el bienestar, aumentando la demanda de soluciones tecnológicas que permitan a los usuarios gestionar su salud de manera proactiva desde sus dispositivos móviles. La propuesta de una aplicación móvil de bienestar y salud con un chatbot inteligente se posiciona en este contexto como una solución innovadora, capaz de ofrecer consejos personalizados y promover hábitos saludables de forma interactiva y accesible.

Aplicaciones como **Strava**, **Noom, Headspace** representan una muestra de la diversidad y especialización en el mercado de bienestar y salud. Cada una ofrece funcionalidades únicas que atienden a diferentes aspectos del bienestar, desde el ejercicio físico y la nutrición hasta el soporte para la salud mental y la formación de hábitos

### 2.2.- Relación de productos o servicios a prestar

La aplicación ofrecerá los siguientes productos y servicios:

* **Chatbot Inteligente:** Utilizando la API de OpenAI, proporcionará consejos personalizados sobre salud y bienestar, recomendaciones de ejercicios, nutrición y gestión del estrés.
* **Recordatorios Personalizados:** Configuración de recordatorios para tomar medicación, beber agua, realizar actividad física, etc., fomentando la adopción de hábitos saludables.
* **Seguimiento de Metas y Progresos:** Herramientas para establecer y seguir metas de salud y bienestar, con visualización de progresos a través de estadísticas y gráficos.
* **Sistema de Login y Registro:** Para personalizar la experiencia del usuario y garantizar la privacidad de sus datos.

### 2.3.- Constitución de la empresa

La empresa se constituirá como una Sociedad Limitada (SL) debido a su flexibilidad operativa y la limitación de responsabilidad al capital aportado, ideal para startups tecnológicas en España. Esto facilitará la gestión administrativa y fiscal, permitiendo un crecimiento escalable. Se registrará en el Registro Mercantil y cumplirá con todas las obligaciones legales, fiscales y laborales pertinentes en España, incluyendo la Ley Orgánica de Protección de Datos Personales y garantía de los derechos digitales (LOPDGDD) para asegurar la privacidad de los usuarios.

### 2.4.- Recursos

### **Humanos:** Equipo multidisciplinario compuesto por desarrolladores de software especializados en .NET MAUI y C#, expertos en inteligencia artificial y procesamiento de lenguaje natural, diseñadores UX/UI, especialistas en marketing digital y personal de soporte y mantenimiento.

### **Tecnológicos:** Suscripción a la API de OpenAI, licencias de software de desarrollo (.NET MAUI, Rider de JetBrains, Visual Studio 2022), servicios de cloud para el hosting de la aplicación y la base de datos (preferentemente Azure para una integración óptima con el stack tecnológico), y herramientas de diseño y prototipado (Marvel app).

### **Financieros:** Inversión inicial para cubrir el desarrollo de la aplicación, marketing inicial, infraestructura tecnológica y gastos operativos. Se buscará financiación a través de fondos propios, préstamos bancarios, subvenciones disponibles para startups tecnológicas y, potencialmente, inversores ángeles interesados en el sector de la salud digital.

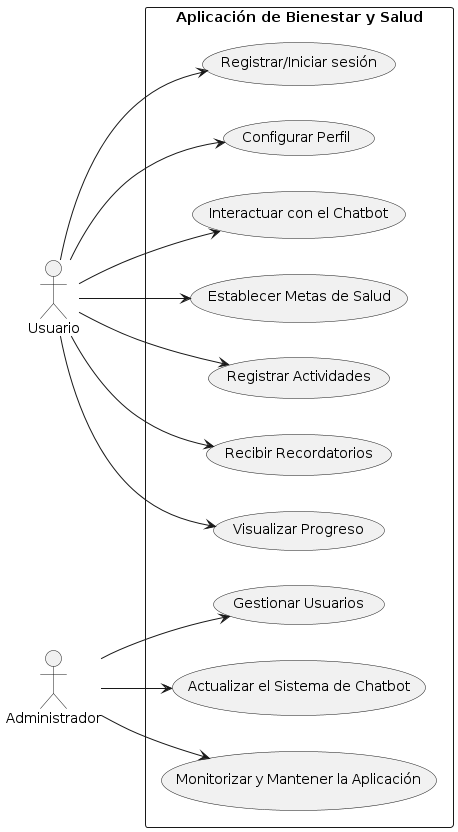
# 3. Análisis y diseño de los requisitos de la aplicación

### 3.1.- Herramientas escogidas

Para el desarrollo efectivo de la aplicación, las herramientas y tecnologías seleccionadas se utilizarán de la siguiente manera:

* **.NET MAUI**: Utilizaré la última versión estable de .NET MAUI para asegurar la compatibilidad y aprovechar las últimas características disponibles. La aplicación se estructurará alrededor del patrón MVVM (Model-View-ViewModel) para facilitar un desarrollo limpio y mantenible. Esto permitirá una fácil expansión o modificación de la aplicación en el futuro.
* **Visual Studio 2022 y ASP.NET Core Web API**: La API se desarrollará utilizando la plantilla ASP.NET Core Web API en Visual Studio 2022. Este enfoque proporciona un marco sólido para construir una API RESTful que será constantemente actualizada y desplegada en Azure. La elección de Azure como plataforma de hosting se alinea con la integración nativa que ofrece con Visual Studio, simplificando el proceso de CI/CD (Integración Continua/Despliegue Continuo).
* **Rider de JetBrains**: A pesar de desarrollar la API en Visual Studio, el resto del desarrollo, incluyendo el frontend de la aplicación, se realizará en Rider. Configuraré Rider para trabajar de forma óptima con .NET MAUI, ajustando el entorno para soportar emuladores tanto de Android como de iOS, y personalizaré mi espacio de trabajo para maximizar la productividad.
* **yEd Graph Editor**: Para el diseño del Modelo Entidad-Relación (MER) de la base de datos, se empleará yEd. Esta herramienta facilita la creación de diagramas complejos de manera intuitiva, permitiendo visualizar de forma clara las entidades, sus atributos y las relaciones entre ellas. Este paso es crucial para asegurar una estructura de datos coherente y eficiente.
* **ArgoUML**: El diseño de la arquitectura de clases se llevará a cabo usando ArgoUML, una herramienta de modelado que soporta todos los diagramas estándar de UML 1.4. Esto es fundamental para estructurar la lógica de la aplicación y sus componentes de manera organizada, facilitando así la comprensión del flujo de datos y las interacciones entre diferentes partes del sistema.
* **Azure para la API y la Base de Datos**: Se utilizará Azure para hospedar tanto la API como la base de datos de la aplicación. Esto no solo facilita la integración y el despliegue continuo gracias a su compatibilidad con Visual Studio y herramientas de desarrollo, sino que también asegura una infraestructura escalable y segura para la operación de la aplicación.
* **SQL Server**: Como sistema de gestión de base de datos, se optará por SQL Server, aprovechando su integración con el entorno de desarrollo de .NET y su robustez para manejar transacciones complejas y grandes volúmenes de datos. La elección de SQL Server complementa el enfoque en la seguridad y el rendimiento, esenciales para la arquitectura de la aplicación.

### 3.2.- Diagrama de casos de uso

****

En este diagrama reflejo cómo los usuarios pueden interactuar con la aplicación para aprovechar al máximo sus funcionalidades:

**Registrar/Iniciar sesión:** Crear cuentas y acceder a sus perfiles.

**Configurar Perfil:** Personalizar preferencias y objetivos de salud.

**Interactuar con el Chatbot:** Obtener recomendaciones y consejos de salud.

**Establecer Metas de Salud:** Definir objetivos personales de bienestar.

**Registrar Actividades:** Llevar un seguimiento de su progreso hacia las metas.

**Recibir Recordatorios:** Mantenerse motivados con alertas personalizadas.

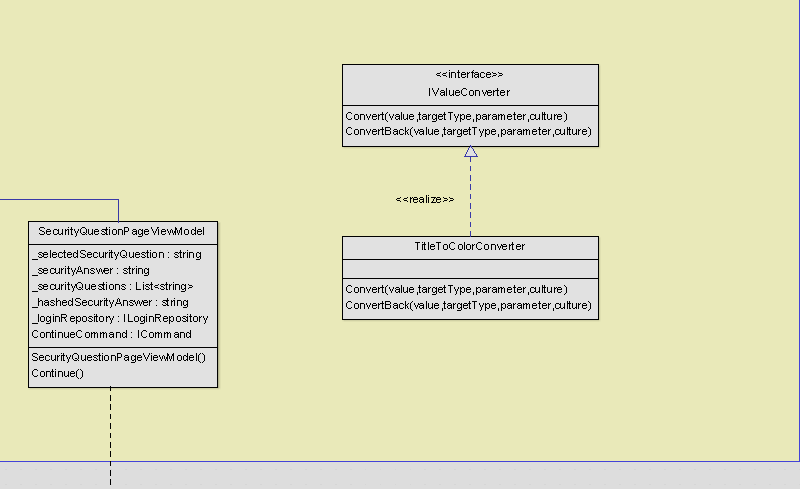
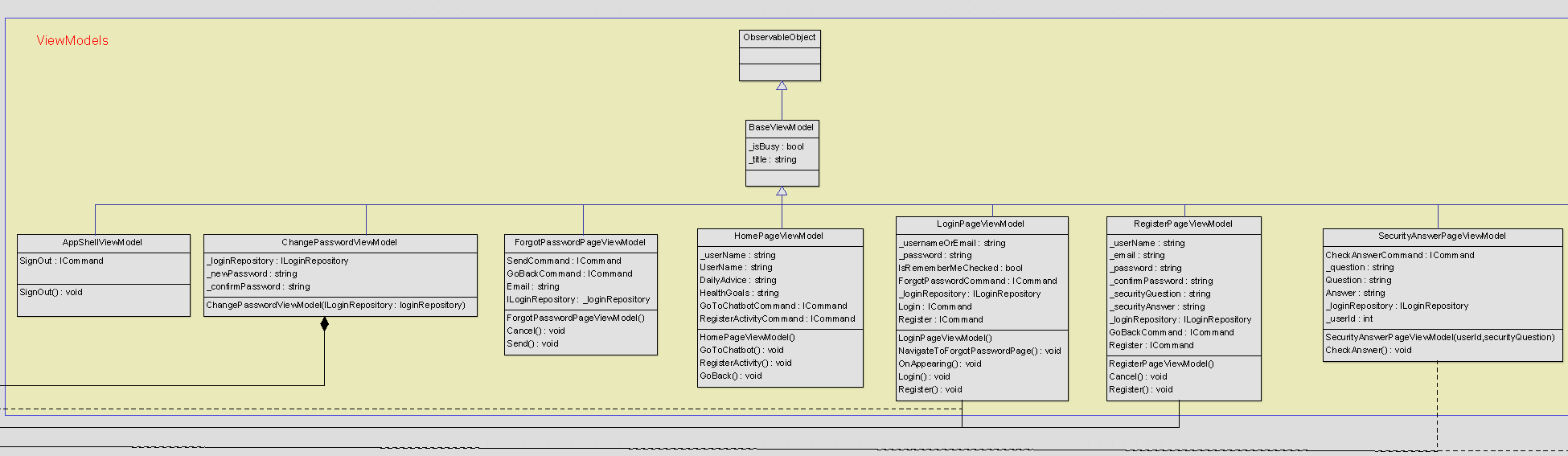
**Visualizar Progreso:** Ver su avance en gráficos y estadísticas.

Además, como administrador, gestiono usuarios, actualizo el chatbot y mantengo la app para asegurar una experiencia fluida y útil para todos.

### 3.3.- Diagrama de clases

Imagen que contiene Diagrama

Descripción generada automáticamente



Diagrama

Descripción generada automáticamente

1. **Dependencias en Models**:
   * UserInfo depende de SecurityQuestion, indicando que cada usuario tiene asociada al menos una pregunta de seguridad.
   * UserInfo también se relaciona con ChangePasswordRequest, lo que sugiere que los usuarios pueden solicitar cambiar su contraseña.
   * UserInfo tiene asociaciones con UserGoals, ActivityRecords, HealthReminders, y ProgressTracking, representando que los datos del usuario se utilizan para rastrear metas, actividades, recordatorios de salud y progresos.
2. **Dependencias en Services**:
   * ILoginRepository es una interfaz que define métodos como Login, Register, y ChangePassword, y es implementada por LoginService, lo que permite abstraer la lógica de acceso a datos para la autenticación del usuario.
   * OpenAIService se conecta con Message, utilizada para enviar y recibir mensajes dentro de una funcionalidad de chatbot.
   * LoadingService proporciona métodos para mostrar u ocultar una interfaz de carga, utilizada en las vistas durante operaciones asíncronas.
3. **Dependencias en ViewModels**:
   * Todos los ViewModels heredan de BaseViewModel, que contiene propiedades comunes como indicadores de actividad (\_isBusy) y títulos (\_title).
   * Cada ViewModel específico interactúa con ILoginRepository, lo que indica que la autenticación y la recuperación de información del usuario son operaciones comunes a varias vistas.
   * ChangePasswordViewModel utiliza ChangePasswordRequest, lo que sugiere que este ViewModel maneja la solicitud de cambio de contraseña.
   * AppShellViewModel maneja la navegación de nivel superior o acciones comunes a toda la aplicación, como el cierre de sesión (SignOut).
   * ForgotPasswordPageViewModel gestiona las operaciones relacionadas con el olvido de contraseña, como enviar instrucciones para restablecerla.
   * HomePageViewModel puede incluir comandos para navegar a otras páginas o secciones de la aplicación, como un chatbot o la funcionalidad de registro de actividades.
   * LoginPageViewModel administra la lógica de inicio de sesión y navegación a la página de olvido de contraseña.
   * RegisterPageViewModel maneja la lógica de registro de nuevos usuarios y la configuración de preguntas de seguridad.
   * SecurityAnswerPageViewModel y SecurityQuestionPageViewModel manejan la lógica relacionada con la verificación de las preguntas de seguridad para la recuperación de la cuenta o el cambio de contraseña.
4. **Dependencias en Views**:
   * Las vistas (como AboutPage, ChangePasswordPage, ChatBotPage, etc.) se relacionan con BaseViewModel a través del patrón MVVM, lo que permite la actualización dinámica de la UI en respuesta a los cambios de estado en los ViewModels.
   * ChatBotPage tiene una relación directa con Message, indicando que esta vista está diseñada para mostrar mensajes dentro de un chatbot

### 3.4.- MER

Diagrama, Esquemático

Descripción generada automáticamente

Este diagrama MER está compuesto por:

* **UserInfos**: Es la entidad central que almacena la información del usuario, como el UserId, UserName, Password y Email.
* **UserGoals**: Está relacionada con UserInfos por la llave UserId. Cada usuario puede tener muchas metas. Los atributos incluyen GoalId, Description, GoalType, TargetDate, y IsAchieved, que representan las metas y su estado (alcanzado o no).
* **ActivityRecords**: También está relacionada con UserInfos por UserId. Cada usuario puede tener múltiples registros de actividad. Los atributos incluyen RecordId, Description, ActivityType, Date, y DurationMinutes.
* **ProgressTracking**: Está vinculada con UserInfos por UserId, indicando que cada registro de seguimiento de progreso está asociado con un usuario específico. Los atributos son ProgressId, Metric, Value, y Date.
* **HealthReminders**: Relacionada con UserInfos a través de UserId, sugiere que cada usuario puede tener varios recordatorios de salud. Incluye ReminderId, ReminderType, ReminderTime, y Message.

### **SecurityQuestions**: Esta entidad está relacionada con UserInfos y guarda las preguntas de seguridad para recuperación de contraseñas o verificación de identidad. Incluye SecurityQuestionsId, QuestionText, y Answer.

### 3.5.- Diagrama de Base de datos

Texto

# 4. Desarrollo de la aplicación

### 4.1.- Control de usuarios

Texto

### 4.2.- Desarrollo e implementación de la aplicación

Texto

# 5. Pruebas e informes de la aplicación

### 5.1.- Estrategia de pruebas

Texto

### 5.2.- Pruebas realizadas

Texto

### 5.3.- Informes

Texto

# 6. Implantación y documentación

### 6.1.- Exportación y distribución de la aplicación

Texto

### 6.2.- Manual de instalación

Texto

### 6.3.- Manual de usuario

Texto

### 6.4.- Repositorio de la aplicación

Texto

### 6.5.- JavaDoc

Texto

# 7.- Bibliografía

Texto

# 8.- Anexos

Texto