

Universidad Internacional de La Rioja

Facultad de Ciencias de la Salud

Máster Universitario en Bioinformática

Desarrollo de un algoritmo predictivo para la personalización de estilos de vida basado en genómica y biométrica

|  |  |
| --- | --- |
| Trabajo fin de Estudio presentado por: | Maximiliano Hernández Sahuquillo y David Fernández Martín |
| Tipo de trabajo: | 2 |
| Director/a: | Dr. José Arturo Mora Soto |
| Fecha: | 14 de marzo de 2024 |

Resumen

El presente TFM tiene como objetivo desarrollar un algoritmo predictivo para la personalización de estilos de vida basado en datos genómicos y biométricos. El algoritmo estará diseñado para generar recomendaciones personalizadas de estilo de vida para cada individuo, teniendo en cuenta su predisposición genética a ciertas enfermedades, su respuesta al ejercicio y su respuesta a diferentes tipos de dieta.

**Palabras clave:** estilo de vida, genómica, biométrica

Abstract

This TFM aims to develop a predictive algorithm for lifestyle personalisation based on genomic and biometric data. The algorithm will be designed to generate personalised lifestyle recommendations for each individual, taking into account their genetic predisposition to certain diseases, their response to exercise and their response to different types of diet.

**Keywords**: lifestyle, genomics, biometrics

Índice de contenidos

[1. Introducción 7](#_Toc166157638)

[1.1. Justificación 7](#_Toc166157639)

[1.2. Planteamiento del problema 9](#_Toc166157640)

[Objetivos 9](#_Toc166157641)

[2. Marco teórico 10](#_Toc166157642)

[3. Metodología 12](#_Toc166157643)

[3.1. Recolección de datos y preparación del modelo 12](#_Toc166157644)

[**3.2.** **Preprocesamiento y limpieza de datos** 12](#_Toc166157645)

[**3.3.** **Integración en la base de conocimientos** 13](#_Toc166157646)

[**3.4.** **Desarrollo del Chatbot basado en RAG** 13](#_Toc166157647)

[**3.5.** **Arquitectura y funcionamiento de la API** 14](#_Toc166157648)

[**3.6.** **Evaluación y refinamiento del modelo** 14](#_Toc166157649)

[3.7. Desarrollo del prototipo de la web 15](#_Toc166157650)

[3.7.1. Consideraciones para el desarrollo de la web 15](#_Toc166157651)

[3.7.2. Pasos para el desarrollo de nuestra web 15](#_Toc166157652)

[3.7.3. Pruebas de validación 18](#_Toc166157653)

[3.8. Esquema y funcionamiento formal del RAG 19](#_Toc166157654)

[3.8.1. Funcionamiento de main.py 19](#_Toc166157655)

[4. Resultados y discusión 22](#_Toc166157656)

[5. Conclusiones 23](#_Toc166157657)

[Referencias bibliográficas 25](#_Toc166157658)

Índice de figuras

[Figura 1. “Figuras” del menú de estilos. (Elaboración propia) 9](#_Toc20304756)

Índice de tablas

[Tabla 1. “Tablas” del menú de estilos 8](#_Toc20304757)

# Introducción

El presente Trabajo Fin de Máster (TFM) pretende desarrollar y poner en práctica un servicio web basado en datos biométricos, genéticos y distintos aspectos relacionados con la salud los cuales van a ser procesados por varios LLM (Large Language Model) para su correcta implementación. El motivo que condujo a esta idea es el de aportar un sistema que personalice los estilos de vida de cada paciente.

En las próximas páginas se introducirá este trabajo, primero se abordarán los motivos que llevaron a escoger esta temática, seguido de la introducción del problema a resolver y, por último, se plantearán el objetivo principal y objetivos secundarios a alcanzar.

## Justificación

En los últimos años, el campo de la inteligencia artificial ha sufrido un avance vertiginoso, particularmente con respecto al procesamiento del lenguaje natural (NLP) y el desarrollo de modelos de lenguaje de gran tamaño (LLM)(1). Estos consiguieron gran notoriedad y presencia en el sector médico, gracias a sus múltiples aplicaciones, desde el procesamiento de bastas cantidades de datos hasta poder interpretar anotaciones y datos clínicos para generar reportes adecuados.

En estudios recientes acerca del rendimiento de bots de chat basados en LLMs al responder preguntas médicas, han demostrado una impresionante cantidad de aciertos con respecto al conocimiento sobre bases de datos médicas. Esto es debido ya que en primer lugar, poseen suficientes conocimientos médicos para aplicaciones en medicina, lo que podría provocar un cambio de paradigma en la prestación de asistencia sanitario (2). Pero persiste una considerable inquietud sobre la preocupación de que los LLM puedan ser manipulados maliciosamente para generar contenidos dañinos o engañosos (3).

Permitiendo que el proceso de cribado de la literatura (que se vuelve excesivamente laboriosa debido a que pueden existir miles de artículos con respecto a una patología o un gen) se vuelva más rápido y eficaz, permitiendo resumir la literatura científica y dejando a los investigadores y médicos la información de manera accesible con respecto al esfuerzo realizado (4).

A su vez en otros campos relacionados con la salud como la nutrición, numerosos estudios han contribuido al desarrollo de diversos métodos de recomendación de alimentos (5,6). Estos enfoques mediante el uso de LLM no sólo simplifican el proceso de elección de alimentos, sino que también promueven la adopción de hábitos sostenibles y saludables entre los usuarios. El poder de estas recomendaciones reside en su capacidad para integrar la ciencia de la nutrición en las necesidades de la población o de subpoblaciones, ofreciendo una guía intuitiva y práctica para las personas que desean tomar decisiones dietéticas informadas.

No obstante, los servicios convencionales de recomendación de alimentos orientados a la nutrición han encontrado limitaciones a la hora de comprender exhaustivamente la intrincada interacción entre la salud y el bienestar de las personas, que abarca parámetros fisiológicos, actividad física y calidad del sueño, y de alinearlos con necesidades nutricionales personalizadas.

Concretamente, estos sistemas suelen tener dificultades para adaptar sus sugerencias alimentarias a las necesidades nutricionales específicas de cada persona. La variabilidad inherente a la forma en que la nutrición afecta a los individuos suscita preocupación por la ausencia de una auténtica personalización, además otra carencia importante es la falta de interactividad.

Es en este punto donde se llega al objeto de estudio de este Trabajo Fin de Máster, el de como implementar un servicio sencillo e intuitivo, pero a la vez complejo, que permite a los usuarios que accedan a la web obtener recomendaciones personalizadas desde dudas sobre ciertas patologías que puedan llegar a presentar (obviando el hecho de que la asistencia a un profesional de la salud será necesaria) hasta alimentos o cambios que podrán realizar en función de distintos objetivos que posean.

## Planteamiento del problema

El objeto de estudio de este TFM, como se ha presentado en el apartado anterior es desarrollar un sistema que personalice los estilos de vida en función de la genética y la biometría, ya que recomendaciones generales sobre dieta y ejercicio son útiles, pero no siempre son efectivas para todas las personas.

## Objetivos

Como ya se ha adelantado, la propuesta consiste en desarrollar una web basada en varios LLM que permita el ingreso de los interesados de sus datos, biométricos, genéticos y otras dudas médicas que presenten (se responderá orientativamente).

**Objetivo principal:** desarrollar un sistema de personalización de estilos de vida basado en genómica y biometría. Este sistema tiene el potencial de mejorar la salud y el bienestar de las personas al proporcionarles recomendaciones personalizadas sobre dieta, ejercicio y otros aspectos del estilo de vida.

**Objetivos secundarios:**

* Mejorar la salud y el bienestar de las personas gracias al sistema de personalización de estilos de vida
* Reducir los costos de atención médica
* Empoderar a las personas para que tomen el control de su salud

# Marco teórico

No habría una fuente bibliográfica la cual haya significado una mayor influencia sobre cómo estructurar el framework o el proceso de realizar la web, sino que se ha ido obteniendo la información de distintas publicaciones u otras referencias.

Teniendo como referencia para proceder a herramientas como las publicadas ChatDiet (7) o Dietos (5), los cuales permiten comprender la manera de organizar el framework con respecto al uso de LLM para personalizar distintas necesidades del cliente (en el caso de ambas herramientas mencionadas en el campo de la nutrición).

Pero también se sigue un procedimiento similar en otros trabajos que proponen frameworks con LLM para datos genéticos (8).

Es decir, que para uno de los dos LLM que se procede a implementar en base a la lectura de documentos, la estructura básica es la misma seguida independientemente del ámbito relacionado con la salud escogido. Por lo que el primer paso será recopilar una gran cantidad de documentos de interés o extraerlos directamente de las webs necesarias, que permitirán al LLM generar respuestas en base a los diagnósticos o preguntas que proporcione el usuario.

Para el otro LLM, se utilizan los datos pertinentes de bases de datos como NHANES u otras similares, los cuales permitirán ser codificados para su posterior uso para obtener distintas estadísticas y porcentajes pertinentes para mostrar al usuario.

Así aportando una información más exacta acerca de por ejemplo “las posibilidades de padecer X patología o enfermedad” en base a los datos ingresados por el usuario como información sobre distintas dudas que posea acerca del diagnóstico proporcionado o acerca de otros ámbitos relacionados con la salud. Todo esto acompañado de las figuras necesarias para mostrar la información de la manera más accesible posible.

# Metodología

## Recolección de datos y preparación del modelo

Para el desarrollo del chatbot de recomendaciones personalizadas de estilo de vida, se ha utilizado la base de datos NHANES (National Health and Nutrition Examination Survey) como una de las principales fuentes de información. NHANES es una encuesta nacional de los Estados Unidos que recopila una amplia gama de datos sobre la salud y la nutrición de la población, incluyendo datos demográficos, medidas antropométricas, datos de nutrición, actividad física, datos de salud, resultados de pruebas de laboratorio, hábitos y comportamientos de salud, entre otros.

Se procede a la identificación de las variables relevantes de NHANES que son más adecuadas para el propósito de proporcionar recomendaciones de estilo de vida personalizadas. Estas variables incluyen (pero no se limitan a), edad, género, índice de masa corporal (IMC), ingesta dietética, niveles de actividad física, presión arterial, niveles de colesterol y glucosa en sangre.

Además de los datos de NHANES, se ha recopilado una amplia gama de documentos médicos y de estilo de vida que abarcan temas como nutrición, ejercicio, manejo del estrés, sueño y prevención de enfermedades. Estos documentos proporcionan una base de conocimientos sólida y confiable para complementar los datos de NHANES y mejorar la calidad de las recomendaciones generadas por el chatbot.

## **Preprocesamiento y limpieza de datos**

Una vez identificadas las variables relevantes de NHANES y recopilados los documentos adicionales, se realiza un proceso exhaustivo de preprocesamiento y limpieza de datos. Este proceso implica la eliminación de valores atípicos, el manejo de datos faltantes, la normalización de variables y la codificación de variables categóricas, entre otras tareas. El objetivo es garantizar la calidad y consistencia de los datos antes de integrarlos en nuestro modelo.

## **Integración en la base de conocimientos**

Después del preprocesamiento y la limpieza, se han integrado los datos seleccionados de NHANES y los documentos médicos y de estilo de vida en la base de datos vectorial. Se utilizan técnicas de vectorización para representar la información de manera eficiente y facilitar la recuperación de respuestas relevantes durante la interacción con el chatbot.

## **Desarrollo del Chatbot basado en RAG**

El presente chatbot se basa en un enfoque de RAG (Retrieval-Augmented Generation), que combina la recuperación de información relevante de la base de datos vectorial con la generación de respuestas utilizando modelos de lenguaje avanzados. El código Python que se ha desarrollado implementa la funcionalidad principal del RAG, incluyendo la carga de la base de datos vectorial, el procesamiento de las consultas de los usuarios y la generación de recomendaciones personalizadas.

El flujo de trabajo del chatbot es el siguiente:

1. El usuario interactúa con el chatbot a través de una interfaz de usuario intuitiva, proporcionando información sobre sus características personales, datos de salud y preferencias de estilo de vida.
2. El chatbot procesa la consulta del usuario y utiliza técnicas de recuperación de información para identificar los datos más relevantes de la base de datos vectorial, considerando tanto los datos de NHANES como los documentos médicos y de estilo de vida.
3. Utilizando los datos recuperados y la información proporcionada por el usuario, el chatbot genera recomendaciones personalizadas de estilo de vida adaptadas a las necesidades y preferencias individuales.
4. Las recomendaciones generadas se presentan al usuario de manera clara y comprensible, junto con explicaciones y fundamentos basados en evidencia médica y científica.

## **Arquitectura y funcionamiento de la API**

Para facilitar la interacción entre el usuario y el chatbot, se desarrolla una API que actúa como interfaz entre la interfaz de usuario y el código Python subyacente. La API se encarga de recibir las consultas del usuario, procesarlas y enviarlas al código Python para su análisis y generación de respuestas.

La arquitectura de la API se basa en el framework FastAPI, que permite crear “endpoints” eficientes y escalables para manejar las solicitudes de los usuarios. La API se comunica con el código Python a través de llamadas a funciones y paso de parámetros, asegurando una comunicación fluida y efectiva entre los componentes del sistema.

## **Evaluación y refinamiento del modelo**

Una vez que el chatbot está en funcionamiento, se lleva a cabo un proceso riguroso de evaluación y refinamiento del modelo. Mediante el uso de métricas de evaluación apropiadas, como la precisión, la exhaustividad y la satisfacción del usuario, para medir el rendimiento del chatbot en la generación de recomendaciones personalizadas.

Además, se recopila y tiene en cuenta la retroalimentación de los usuarios, para realizar análisis cualitativos con la finalidad de identificar áreas de mejora y ajustar el modelo en consecuencia. Este proceso iterativo permite afinar continuamente el chatbot y garantizar que proporcione recomendaciones de alta calidad y relevancia para los usuarios.

## Desarrollo del prototipo de la web

### Consideraciones para el desarrollo de la web

Se tendrá en cuenta un diseño sencillo y minimalista, con un buen uso de espacios en blanco para crear una sensación de amplitud y orden. Junto con el uso de una tipografía y paletas de colores atractiva y que se adapten al estilo de la aplicación.

Por último, se podrán incluir imágenes y videos que sean relevantes para el contenido de la aplicación

Con relación a la parte de programación, se escribirá un código limpio y organizado con los comentarios pertinentes explicando su funcionamiento, junto al uso de librerías necesarias.

Para el desarrollo de la aplicación web se hará uso e Python Flask con las librerías necesarias para facilitar el desarrollo. Y en última instancia se realizarán las pruebas en distintos dispositivos y navegadores para asegurar su correcto funcionamiento.

### Pasos para el desarrollo de nuestra web

#### Identificar funcionalidades clave

Para la web, estarán presentes distintas funcionalidades:

Inicio de Sesión/Registro: Permitiremos a los usuarios crear una cuenta y/o iniciar sesión para acceder a sus datos personalizados y recomendaciones de manera segura.

Generación de Recomendaciones Personalizadas: Una vez ingresados los datos, nuestro algoritmo analizará esta información para generar recomendaciones personalizadas de estilo de vida, incluyendo dieta, ejercicio, y otros hábitos saludables.

Visualización de Resultados: Presentaremos los resultados y recomendaciones en una página de resultados de fácil lectura, que puede incluir gráficos, listas y explicaciones textuales.

Página de Contacto/Soporte: Incluiremos una sección donde los usuarios puedan enviar preguntas, comentarios o solicitar soporte técnico.

Sección de Información y Ayuda: Proporcionaremos una página con información sobre cómo utilizar la web app, la ciencia detrás de las recomendaciones y consejos generales de salud.

#### Wireframes

Procederemos a esbozar wireframes para nuestras páginas clave utilizando herramientas como Balsamiq o papel y lápiz para los bocetos iniciales. Nos centraremos en la página de inicio con opciones de inicio de sesión/registro.

En cuanto a la organización del formulario de datos genómicos y biométricos, nos aseguraremos de que el formulario esté bien organizado, agrupando los campos relacionados y utilizando indicaciones claras para cada tipo de dato.

Se incluirán elementos visuales, considerando añadir íconos o pequeñas ilustraciones que puedan hacer el formulario más intuitivo y menos monótono.

Y para facilitar la navegación, se incluirán botones claros para enviar o cargar datos, y asegurarse de que los usuarios puedan navegar fácilmente a través del formulario.

#### Diseño de Interfaz de Usuario

Con base en nuestros wireframes, diseñamos una interfaz de usuario atractiva y funcional, teniendo en cuenta las mejores prácticas de UX.

#### Prototipo interactivo

Creamos un prototipo interactivo utilizando herramientas como Figma, que nos permitirá simular la experiencia del usuario con la aplicación.

#### Selección del estilo visual

Definimos nuestra paleta de colores y tipografía para asegurar que nuestra aplicación sea visualmente coherente y atractiva. Utilicemos herramientas como Khroma para seleccionar colores que se ajusten a nuestra marca y público objetivo.

#### Recolección de Feedback

Presentemos nuestro prototipo a un grupo selecto para obtener su opinión y hagamos ajustes según sea necesario

#### Desarrollo Front-End

Con nuestro prototipo y el feedback en mano, comencemos a codificar la interfaz de usuario. Podemos usar plantillas de Bootstrap para acelerar este proceso.

#### Desarrollo Back-End con Flask

Simultáneamente, trabajamos en la configuración de Flask para nuestro back-end, asegurándonos de que los datos del usuario se manejen de manera segura y eficiente.

#### Integración Front-End y Back-End

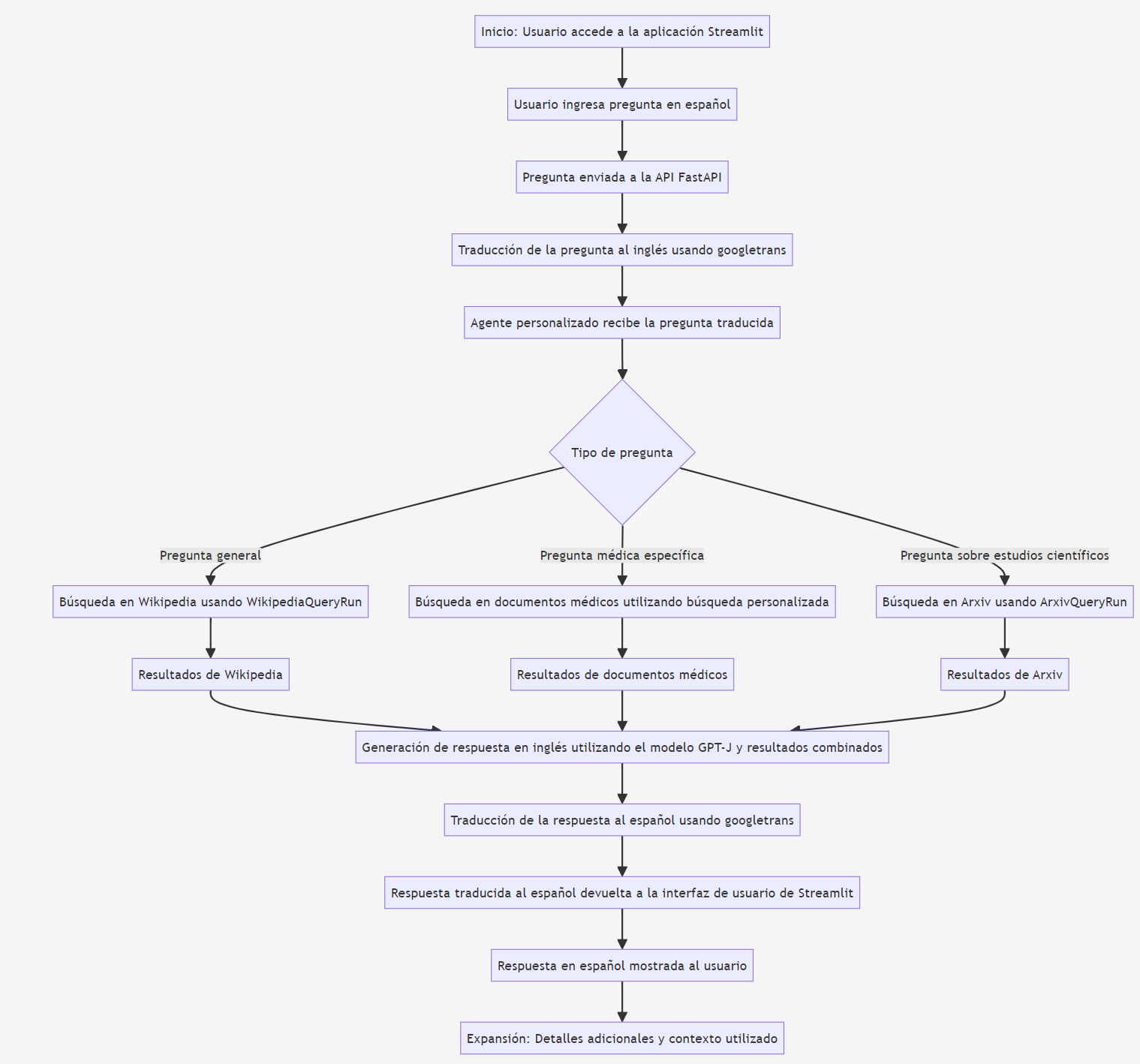
Una vez que tengamos el front-end y el back-end listos, integremos ambos para que trabajen de forma fluida y cohesiva.

### Pruebas de validación

Realicemos pruebas exhaustivas para asegurarnos de que la web app funcione correctamente en diferentes dispositivos y navegadores, y que sea segura

.

## Esquema y funcionamiento formal del RAG

En esta sección, explicaremos paso a paso cómo funciona el código del GitHub que hemos creado para este TFM. Al mismo tiempo, iremos revisando las herramientas que hemos utilizado y su utilización técnica en el código Python. (sección actualizada del TFM basada en el nuevo código de la v2 del API main.py)

### Funcionamiento de main.py

A continuación, se muestra un esquema gráfico mediante un diagrama Mermaid (Anexo diagramas Mermaid) que ilustra el flujo de la aplicación. Este diagrama ayudará a visualizar cómo interactúan los diferentes componentes del sistema y cómo se procesan las consultas del usuario.

#### **Integración de tecnologías**

* Groq: Proporciona el backend de aceleración de hardware para el modelo de lenguaje GPT-J.
* LangChain: Facilita la orquestación de las diferentes etapas del proceso, desde la carga de documentos hasta la generación de respuestas.
* FAISS: Se utiliza para la gestión eficiente de la base de datos vectorial, permitiendo búsquedas rápidas y precisas.
* Streamlit: Ofrece una forma sencilla y efectiva de crear interfaces de usuario interactivas para aplicaciones de Python, permitiendo la interacción directa con el usuario.
* FastAPI: Proporciona un framework para crear una API robusta y eficiente que permite la comunicación entre el frontend y el backend.

#### Detalles del código

1. Importaciones y Configuración Inicial:
   * Se importan módulos necesarios como fastapi, langchain\_groq, langchain\_community, entre otros.
   * Se cargan las claves API necesarias para Groq a través de variables de entorno, asegurando que el acceso a la API se maneje de manera segura.
2. Inicialización del Modelo de Lenguaje y Configuración del Prompt:
   * Se carga el modelo de lenguaje GPT-J utilizando la clase GroqModel de langchain\_groq, especificando el nombre del modelo y la clave API de Groq.
   * Se configura un ChatPromptTemplate para guiar cómo el modelo debe formular respuestas, instruyéndolo a responder preguntas basadas en el contexto proporcionado.
3. Carga y Vectorización de Documentos:
   * En el evento de inicio de la aplicación (@app.on\_event("startup")), se cargan los documentos desde diferentes fuentes (PDF, CSV, JSON) utilizando los cargadores correspondientes de LangChain.
   * Los documentos se dividen en fragmentos más pequeños utilizando RecursiveCharacterTextSplitter y se vectorizan utilizando OllamaEmbeddings y FAISS para crear una base de datos vectorial eficiente.
4. Configuración de Herramientas Adicionales:
   * Se crean instancias de las herramientas WikipediaQueryRun y ArxivQueryRun utilizando los envoltorios de API correspondientes.
   * Estas herramientas permiten realizar consultas en Wikipedia y Arxiv para obtener información adicional durante el proceso de generación de respuestas.
5. Endpoint de Consulta (/ask):
   * Se define un endpoint de API (@app.post("/ask")) que recibe las consultas de los usuarios.
   * La consulta se traduce del español al inglés utilizando la biblioteca googletrans.
   * Se crea un agente personalizado utilizando la función create\_custom\_tools\_agent, pasando el modelo de lenguaje, las herramientas adicionales y el prompt.
   * El agente procesa la consulta y genera una respuesta utilizando el modelo de lenguaje y las herramientas adicionales.
   * La respuesta generada se traduce de vuelta al español utilizando googletrans y se devuelve al usuario junto con el contexto utilizado para generarla.

#### Flujo de la aplicación

1. El usuario accede a la interfaz de usuario de Streamlit y realiza una consulta en español.
2. La consulta se envía al endpoint /ask de la API FastAPI.
3. La consulta se traduce al inglés utilizando googletrans.
4. Se crea un agente personalizado utilizando el modelo de lenguaje GPT-J, las herramientas adicionales (Wikipedia y Arxiv) y el prompt.
5. El agente procesa la consulta y genera una respuesta utilizando el modelo de lenguaje y las herramientas adicionales.
6. La respuesta generada se traduce de vuelta al español utilizando googletrans.
7. La respuesta y el contexto utilizado se devuelven a la interfaz de usuario de Streamlit.
8. La interfaz de usuario muestra la respuesta al usuario y permite expandir los detalles adicionales y el contexto utilizado.

Este esquema y flujo de la aplicación representan el funcionamiento actualizado de nuestro RAG basado en el nuevo código de main.py. La integración de tecnologías como Groq, LangChain, FAISS, Streamlit y FastAPI, junto con las herramientas adicionales de Wikipedia y Arxiv, permite un procesamiento eficiente de las consultas de los usuarios y la generación de respuestas precisas y contextualizadas.

# Resultados y discusión

Los resultados preliminares de nuestro chatbot basado en RAG y la base de datos vectorial han demostrado un gran potencial para proporcionar recomendaciones personalizadas de estilo de vida a los usuarios. La combinación de los datos de NHANES con documentos médicos y de estilo de vida adicionales ha permitido generar recomendaciones precisas y adaptadas a las necesidades individuales.

La evaluación cuantitativa del modelo ha revelado altas puntuaciones de precisión y exhaustividad en la generación de recomendaciones relevantes. Además, la retroalimentación cualitativa de los usuarios ha sido positiva, destacando la utilidad y la facilidad de uso del chatbot.

Sin embargo, también hemos identificado algunas limitaciones y desafíos durante el desarrollo y la evaluación del chatbot. Uno de los principales desafíos ha sido la integración efectiva de los datos de NHANES con los documentos adicionales, asegurando una representación coherente y completa de la información. Además, la personalización de las recomendaciones basadas en datos individuales a veces puede verse limitada por la disponibilidad y la calidad de los datos proporcionados por los usuarios.

A pesar de estos desafíos, creemos que nuestro enfoque basado en RAG y la base de datos vectorial tiene un gran potencial para mejorar la calidad y la relevancia de las recomendaciones de estilo de vida. La combinación de datos estructurados de NHANES con información no estructurada de documentos médicos y de estilo de vida nos permite aprovechar al máximo la riqueza de conocimientos disponibles para generar recomendaciones personalizadas.

En comparación con enfoques tradicionales que se basan únicamente en reglas predefinidas o en la experiencia de expertos, nuestro chatbot tiene la capacidad de adaptarse a las necesidades individuales de cada usuario y proporcionar recomendaciones basadas en evidencia y datos actualizados. Esto tiene el potencial de mejorar significativamente la adopción y la efectividad de las recomendaciones de estilo de vida, ya que los usuarios pueden recibir consejos personalizados y relevantes para su situación específica.

# Conclusiones

En conclusión, nuestro chatbot basado en RAG y la base de datos vectorial que integra datos de NHANES con documentos médicos y de estilo de vida ha demostrado ser una herramienta prometedora para proporcionar recomendaciones personalizadas de estilo de vida a los usuarios. La combinación de técnicas de recuperación de información y generación de lenguaje natural nos permite aprovechar al máximo la riqueza de conocimientos disponibles y adaptarlos a las necesidades individuales de cada usuario.

Los resultados preliminares han sido alentadores, con altas puntuaciones de precisión y exhaustividad en la generación de recomendaciones relevantes, así como una retroalimentación positiva por parte de los usuarios. Sin embargo, también hemos identificado algunas limitaciones y desafíos que requieren atención adicional, como la integración efectiva de datos heterogéneos y la personalización basada en datos individuales limitados.

Para el trabajo futuro, nos proponemos abordar estas limitaciones y continuar mejorando el rendimiento y la usabilidad del chatbot. Algunas de las áreas clave de enfoque incluyen:

1. Explorar técnicas avanzadas de representación de conocimientos y razonamiento para mejorar la integración y la coherencia de los datos de NHANES con los documentos adicionales.
2. Desarrollar métodos de adquisición de datos más efectivos para recopilar información relevante de los usuarios y mejorar la personalización de las recomendaciones.
3. Realizar estudios de usuario a mayor escala para evaluar la efectividad y la aceptación del chatbot en diferentes poblaciones y contextos.
4. Investigar la integración de técnicas de aprendizaje por refuerzo para permitir que el chatbot aprenda y se adapte continuamente en función de las interacciones con los usuarios.
5. Explorar la posibilidad de extender el chatbot a otras áreas relacionadas con la salud y el bienestar, como la salud mental, la gestión de enfermedades crónicas y la prevención de enfermedades.

En resumen, nuestro chatbot basado en RAG y la base de datos vectorial que integra datos de NHANES con documentos médicos y de estilo de vida representa un avance significativo en la provisión de recomendaciones personalizadas de estilo de vida. Aunque aún hay trabajo por hacer, creemos que este enfoque tiene un gran potencial para mejorar la salud y el bienestar de las personas al brindarles acceso a información y consejos personalizados basados en evidencia. Esperamos continuar desarrollando y perfeccionando esta herramienta para ayudar a las personas a tomar decisiones informadas y adoptar estilos de vida más saludables.

1. Codigo diagrama de flujo v3.

graph TD

A[Inicio: Usuario accede a la aplicación Streamlit] --> B[Usuario ingresa pregunta en español]

B --> C[Pregunta enviada a la API FastAPI]

C --> D[Traducción de la pregunta al inglés usando googletrans]

D --> E[Agente personalizado recibe la pregunta traducida]

E --> F{Tipo de pregunta}

F --> |Pregunta general| G[Búsqueda en Wikipedia usando WikipediaQueryRun]

F --> |Pregunta médica específica| H[Búsqueda en documentos médicos utilizando búsqueda personalizada]

F --> |Pregunta sobre estudios científicos| I[Búsqueda en Arxiv usando ArxivQueryRun]

G --> J[Resultados de Wikipedia]

H --> K[Resultados de documentos médicos]

I --> L[Resultados de Arxiv]

J --> M[Generación de respuesta en inglés utilizando el modelo GPT-J y resultados combinados]

K --> M

L --> M

M --> N[Traducción de la respuesta al español usando googletrans]

N --> O[Respuesta traducida al español devuelta a la interfaz de usuario de Streamlit]

O --> P[Respuesta en español mostrada al usuario]

P --> Q[Expansión: Detalles adicionales y contexto utilizado]

Referencias bibliográficas

1. Min B, Ross H, Sulem E, Veyseh APB, Nguyen TH, Sainz O, et al. Recent Advances in Natural Language Processing via Large Pre-Trained Language Models: A Survey [Internet]. arXiv; 2021 [citado 31 de marzo de 2024]. Disponible en: http://arxiv.org/abs/2111.01243

2. Egli A. ChatGPT, GPT-4, and Other Large Language Models: The Next Revolution for Clinical Microbiology? Clin Infect Dis. 11 de noviembre de 2023;77(9):1322-8.

3. Mökander J, Schuett J, Kirk HR, Floridi L. Auditing large language models: a three-layered approach. AI Ethics [Internet]. 30 de mayo de 2023 [citado 31 de marzo de 2024]; Disponible en: https://link.springer.com/10.1007/s43681-023-00289-2

4. Yu B. Evaluating Pre-Trained Language Models on Multi-Document Summarization for Literature Reviews.

5. Agapito G, Calabrese B, Guzzi PH, Cannataro M, Simeoni M, Care I, et al. DIETOS: A recommender system for adaptive diet monitoring and personalized food suggestion. En: 2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob) [Internet]. New York, NY: IEEE; 2016 [citado 1 de abril de 2024]. p. 1-8. Disponible en: http://ieeexplore.ieee.org/document/7763190/

6. Iwendi C, Khan S, Anajemba JH, Bashir AK, Noor F. Realizing an Efficient IoMT-Assisted Patient Diet Recommendation System Through Machine Learning Model. IEEE Access. 2020;8:28462-74.

7. Yang Z, Khatibi E, Nagesh N, Abbasian M, Azimi I, Jain R, et al. ChatDiet: Empowering Personalized Nutrition-Oriented Food Recommender Chatbots through an LLM-Augmented Framework [Internet]. arXiv; 2024 [citado 1 de abril de 2024]. Disponible en: http://arxiv.org/abs/2403.00781

8. Chang J, Wang S, Ling C, Qin Z, Zhao L. Gene-associated Disease Discovery Powered by Large Language Models [Internet]. arXiv; 2024 [citado 1 de abril de 2024]. Disponible en: http://arxiv.org/abs/2401.09490