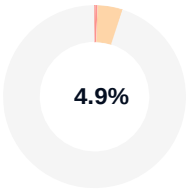
Plagiarism and AI Content Detection Report

Bahamonde Juan_Monge Willam_Tesi...

Scan details

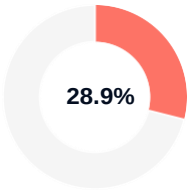
Scan time:	Total Pages:	Total Words:
August 21th, 2023 at 18:0 UTC	36	8797

Plagiarism Detection



Types of plagiarism		Words
Identical	0.2%	16
Minor Changes	0.3%	30
Paraphrased	3.6%	321
Omitted Words	15.6%	1375

AI Content Detection



Text coverage		Words
AI text	28.9%	10382
Human text	71.1%	29730

Learn more

Plagiarism Results: (2)

Copyleaks Internal Database — 4.1%

No introduction available.

Copyleaks Internal Database — 1.2%

No introduction available.

Firmado electrónicamente por:
ALVARO DANILO
UYAGUARI UYAGUARI

ING. Uyaguari Uyaguari, Álvaro Danilo, Msc
C.C: 0103411112

Sistema informático para la normalización de entidades biomédicas basado en búsquedas semánticas multilenguaje, sobre la base de datos médica UMLS.

Bahamonde Tonato, Juan Diego y Monge Aules, William Ariel

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Software

Trabajo de integración curricular, previo a la obtención del título de Ingeniería en Software

Ing. Uyaguari Uyaguari, Alvaro Danilo Msc.

21 de agosto del 2023

Latacunga

ÍNDICE DE CONTENIDO

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

Capítulo I

Introducción

Propósito y contextualización del tema

El presente proyecto de investigación busca desarrollar un sistema informático a través de una búsqueda semántica multilenguaje mediante nuevos métodos para la identificación de entidades biomédicas en español utilizando técnicas de procesamiento de lenguaje natural y el paradigma de Cross-lingual.

Dentro del proceso central para el desarrollo del sistema, es fundamental tener en cuenta la presencia de tres conceptos esenciales que serán utilizados de manera recurrente. Estos conceptos son conocidos como TUI, CUI y AUI.

El TUI es el código de un grupo semántico escogido denominado *"síndromes y enfermedades"* los cuales están compuestos por un código para identificar cada concepto dentro del grupo semántico denominado CUI, para la identificación de cada lexicalización dentro del CUI se asigna un Identificador único del Átomo (AUI). Esta asignación de AUI garantiza que todos los términos se identifiquen, lo que facilita la búsqueda de información de manera precisa.

Es decir, que la búsqueda semántica se refiere a la capacidad del sistema para comprender el contexto y significado de un término médico, no solo la palabra exacta que lo representa. Esto se logra a través de una amplia base de datos UMLS que integra y relaciona terminologías médicas de diferentes fuentes, proporcionando un vocabulario unificado y enriquecido *(Bodenreider, 2004)*.

En este contexto, el desarrollo del sistema dentro del presente proyecto es unificar los conceptos o entidades para que el sistema contribuirá en gran medida a mejorar la calidad, el diagnóstico de síndromes, el desarrollo de tratamientos y la toma de decisiones clínicas.

Justificación del interés de la investigación

Es importante conocer, que los modelos de predicción clínica se aplican comúnmente en la práctica médica para ayudar a los profesionales de la salud a determinar el diagnóstico o pronóstico de un paciente. En la actualidad, existen varios enfoques y métodos para realizar este proceso de predicción, dependiendo de la naturaleza de los datos y de los recursos disponibles en el idioma en el que se redacten las notas médicas.

Al respecto el reconocimiento y la normalización de entidades biomédicas en notas clínicas, es un recurso fundamental para encontrar rasgos en el texto y con estos realizar predicciones de diagnósticos y tratamientos médicos, por lo que encontrar nuevos métodos que no requieran corpus etiquetados para realizar la identificación y normalización de entidades biomédicas es de mucha importancia en el área de la bioinformática y la analítica de datos.

Por tal motivo, es importante realizar esta investigación, a fin de identificar y buscar nuevos métodos y técnicas que permitan realizar la identificación y la normalización de entidades biomédicas.

Objetivo general

Desarrollar un sistema informático para la normalización de entidades biomédicas basado en búsquedas semánticas multilenguaje, sobre la base de datos médica UMLS.

Objetivos específicos

1) Investigar nuevos métodos para la normalización de conceptos biomédicos con un enfoque de cross lingual.
2) Desarrollar bases teórico conceptuales de cómo aplicar nuevos métodos para la normalización de entidades biomédicas mediante el uso de técnicas de procesamiento del lenguaje natural con un enfoque de cross-lingual.
3) Describir el desarrollo del sistema software frameworks y arquitecturas actuales

4) Realizar la validación de los datos, la búsqueda semántica y la similitud que tuvo entre la base de datos y la petición de entrada.

Metodología

El proyecto consiste en desarrollar un sistema capaz de reconocer entidades biomédicas en español a través de búsquedas semánticas multilenguaje aplicando métodos para la detección de entidades médicas.

De esta manera, se cumplirán los siguientes objetivos clave. El proyecto se estructura en tres fases metodológicas principales:

En la primera fase de este proyecto, se llevará a cabo un análisis detallado de la literatura científica actual y de nuevos métodos relacionados con la estandarización de entidades biomédicas. Se emplea un enfoque teórico - lógico para examinar y distinguir entre los diversos métodos que se aplican en este ámbito. Es decir, lo más importante de esta fase investigativa consiste en adquirir una comprensión profunda tanto de las limitaciones presentes como de las posibilidades que se presentan para optimizar el proceso de normalización.

En la segunda fase, utilizaremos nuevos métodos para representar los términos o conceptos médicos en forma de vectores numéricos. El modelo de lenguaje per-entrenado BERTS *(Bidirecionale Encoder Representations from Transformers)* nos permitirá asignar oraciones a un espacio vectorial *(Nils & Iryna, 2019)*, capturando un grupo semántico específico y relacionarlas contextualmente con las entidades médicas, utilizando la base de datos UMLS *(Unified Medical Lenguage System)* como fuente principal para la normalización de las entidades.

Es decir, esto nos permitirá realizar búsquedas semánticas tanto en el idioma del inglés como español, con la finalidad de normalizar o asignar un código para las entidades biomédicas.

En la tercera fase, se implementará un método empírico donde el sistema se someterá a ser evaluado por medio de métricas para verificar su eficiencia y precisión mediante los resultados obtenidos, existirán módulos que serán evaluados mediante un tiempo predeterminado para controlar la precisión de los resultados.

Capítulo II

Marco Teórico

En este capítulo se abordará la exposición de conceptos, así como la presentación de modelos de procesamiento de lenguaje natural. Se realizará una investigación sobre métodos y técnicas que permitan llevar a cabo una búsqueda semántica, implementado herramientas que manejen modelos Transformes. Estos modelos están diseñados para capturar patrones basados en secuencias, los cuales serán transformados en representaciones numéricas conocidas como embeddings vectoriales.

La implementación de las herramientas para la realización de una búsqueda semántica con embeddings facilitará el proceso de transformación, donde son importantes para modelos de lenguaje avanzados, como los basados en la arquitectura Transformer. Estas herramientas facilitan la representación numérica del significado y contexto de palabras, frases o documentos, para mejorar la búsqueda de información relevante en grandes conjuntos de datos textuales (Ashish, et al, 2017).

UMLS

Conceptos

UMLS *(Unified Medical Language System)* sus siglas lo definen como un sistema de lenguaje médico unificado, donde se permite obtener una enciclopedia o biblioteca, en la que se recopila una cantidad sumamente amplia de datos relacionados a vocabularios médicos como síndromes, síntomas, enfermedades, estándares de salud, etc.

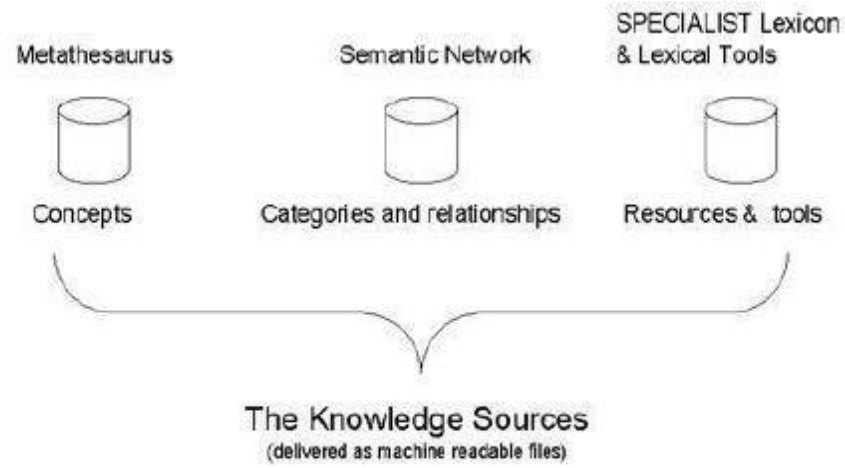
UMLS tiene la característica principal de proporcionar a los desarrolladores información sobre conceptos pertenecientes al campo de la medicina, así como funcionalidades de búsqueda proporcionados para usuarios poco técnicos en el campo *(National Library of Medicine, 2016)*.

Las tres fuentes de conocimiento UMLS las cuales se definen como:

- El Metathesaurus: Es el encargado de proporcionar una biblioteca con más de 5 millones de conceptos, términos o nombres dentro del campo de la biomedicina.
- Red Semántica: Dentro de la Red semántica del UMLS existen 133 categorías y 54 relaciones entre las categorías para etiquetar los dominós en el campo de la biomedicina.
- Herramientas Léxicas: proporciona información léxica que es utilizada para el procesamiento del lenguaje.

Figura 1

UMLS (Unified Medical Language System)



Nota. Tomado de la página web National Library of Medicine *"The Unified Medical Language System (UMLS)"*

Átomos

Dentro del Metathesaurus existe un identificador único conocido como átomo que la National Library of Medicine *(National Library of Medicine, 2016)* define como *"nombres conceptuales o las cadenas de cada uno de los vocabularios de origen"*

AUI es un identificador único capaz de especificar y diferenciar cada concepto del uno con el otro dando así una asignación única a cada concepto, dentro del formato AUI contiene

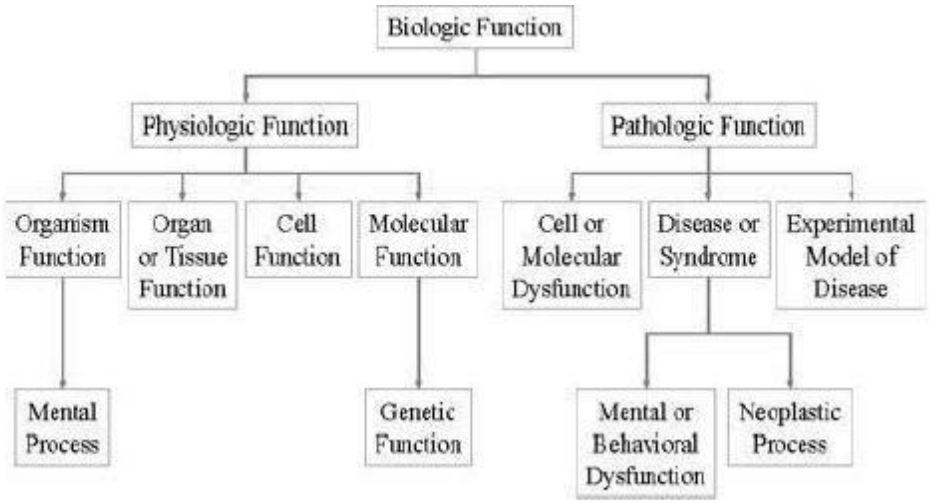
una combinación de letras números alternativos para diferenciación de los conceptos dentro de la base de datos UMLS.

Relaciones entre conceptos

Dentro del UMLS existen relaciones entre los conceptos que establecen conexiones semánticas formando un Red Semántica, dentro existen tipos y relaciones semánticas. Los tipos semánticos están formados por las categorías como síndromes, enfermedades o fármacos, por otro lado, las relaciones semánticas utilizan una relación existente entre los tipos semánticos dando apoyo a la interpretación del significado *(National Library of Medicine, 2016)*.

Figura 2

Red Semántica



Nota. Tomado de la página web National Library of Medicine *"The Semantic Network"*

Reconocimiento de entidades biomédicas con un proceso de Cross-lingual

El reconocimiento de entidades biomédicas mediante un proceso de cross-lingual implica la identificación y etiquetado de términos médicos clave en distintos idiomas. Este proceso se basa en la habilidad para comprender y establecer conexiones entre contextos médicos a lo largo de diversas lenguas. Este método emplea la traducción y vinculación de

términos médicos en varios idiomas, con el propósito de lograr un reconocimiento de entidades biomédicas preciso y coherente en entornos multilingües.

Las representaciones lingüísticas en diferentes idiomas nos brindan la capacidad de comprender el significado de las palabras en contextos multilingües, y desempeñan un papel fundamental como facilitadoras en la transferencia entre lenguajes al construir modelos para el procesamiento de lenguaje natural *(Ruder, Vulić, & Søgaard, 2019)*.

Los modelos basados en Arquitectura Transformer han destacado por su mejor rendimiento comparado con enfoques tradicionales en aplicaciones biomédicas dentro del procesamiento del lenguaje natural. (Cong, et al, 2021).

Normalización de entidades biomédicas

En el campo de la medicina existe una gran variedad de lexicalizaciones principalmente relacionadas a sintomatologías. Gracias a esto se genera la necesidad de normalizar los términos que sean similares relacionados a conceptos específicos, esto se logra a través de la normalización de entidades biomédicas la cual se encarga de agrupar las lexicalizaciones en grupos semánticos con relación a los conceptos médicos. Utilizando identificadores únicos para los dispositivos y así mejorar la calidad y compatibilidad de los datos en la industria de la salud (Pallo, et al, 2023). Ayudando a los investigadores a seguir y relacionar información de forma más efectiva.

Búsqueda semántica

Búsqueda semántica para normalizar entidades biomédicas

La búsqueda semántica es una técnica avanzada para normalizar y categorizar términos biomédicos como genes, proteínas, enfermedades y medicamentos. La normalización es necesaria para asegurar la uniformidad y exactitud al analizar información biomédica.

La búsqueda semántica utiliza modelos de lenguaje avanzados, como las redes neuronales basadas en arquitectura Transformer, eficientes para comprender las complicadas y

contextuales relaciones entre los términos biomédicos. Estas técnicas utilizan un aprendizaje profundo y enfoque para comprender el sentido de las palabras en su contexto. Algunos estudios han empleado exitosamente modelos basados en Transformers para normalizar entidades biomédicas *(Habibi et al., 2023)*.

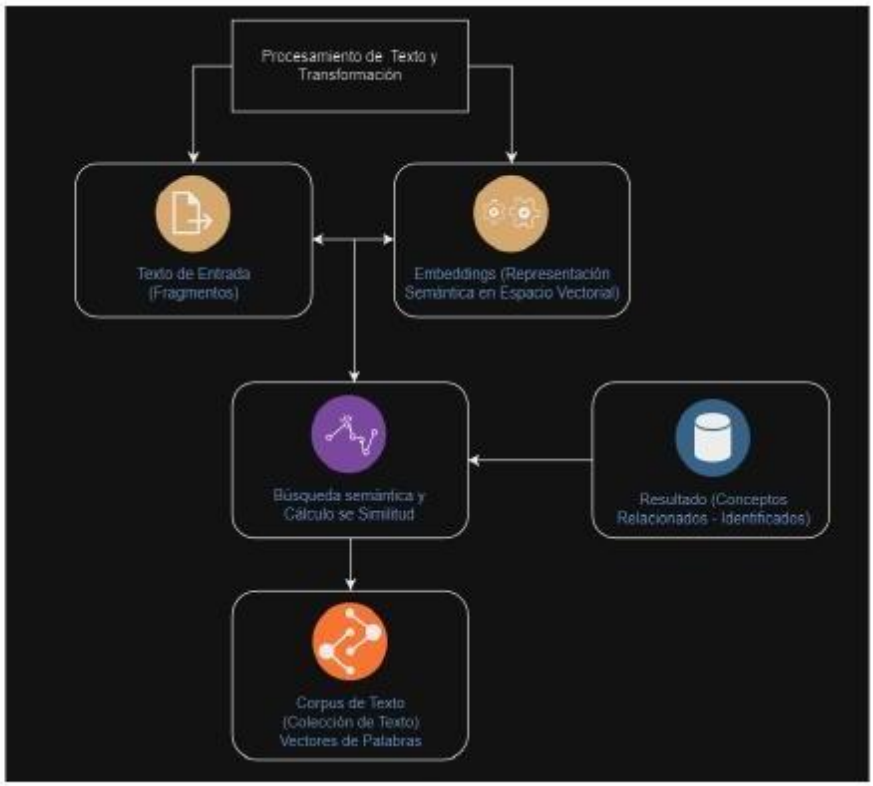
Búsqueda semántica con embeddings

La búsqueda semántica con embedding se define como una técnica dentro del procesamiento del lenguaje natural (NLP), capaz de encontrar y mejorar las relaciones semánticas entre el vocabulario de palabras dentro de un corpus de texto *(Tomas et al., 2013)*. Esta técnica de Lenguaje Natural (NL) utiliza vectores conocidos como embedding para la representación contextual de las palabras.

Embedding permite procesar y transformar fragmentos de texto para expresar su significado con respecto a las relaciones semánticas entre palabras dentro de un espacio vectorial (Avila, s.f.). Así, al realizar una búsqueda semántica, se podrá calcular la similitud entre el vector de consulta y los vectores de otras palabras o expresiones en el corpus y así poder identificar los conceptos relacionados.

Figura 3

Funcionamiento de Embeddings



Nota. Tomado de la página web Medium *"Búsqueda semántica y Embedding"*

Generación de embeddings con modelos Transformer

La generación de embeddings con modelos Transformer se ha utilizado como un método avanzado para el procesamiento de Lenguaje Natural (NLP) que le permita representar el significado de las palabras o frases como vectores numéricos. Existen modelos Transformer, como GTP Y BERT, son un tipo de modelos de lenguaje natural que han demostrado tener mucho éxito gracias a su capacidad de capturar los patrones más complejos y las relaciones semánticas.

En la actualidad se ha centrado el enfoque de los modelos Transformer como atención o atracción propia, esto explica que como técnica permite enfocarse en ciertas partes específicas de un texto durante el proceso de codificación y descodificación *(Rick, 2022)*. Este enfoque contextualizado posibilita crear Embedding existentes y con mayor comprensión del Lenguaje Natural. Un estudio realizado por *(Ashish et al., 2017)* reveló que *"Los modelos*

Transformer son mejores que los enfoques anteriores en muchas tareas de NLP debido a su habilidad para entender las relaciones complicadas entre palabras y su contexto".

Herramientas para la realización de una búsqueda semántica con embeddings

PGVector

PGVector es una extensión de PostgreSQL donde se permite buscar de manera semántica en la base de datos utilizando vectores. Este recurso permite almacenar y manipular vectores de múltiples dimensiones dentro de la base de datos, siendo de esta manera muy útil en aplicaciones que requieren consultas semánticas rápidas *(Greg, 2023)*.

La extensión PGVector permite almacenar embeddings de datos como palabras o frases en forma de vectores dentro de la base de datos para acceder y editar fácilmente sin transferirlos continuamente a la aplicación cliente. Esto es muy útil al trabajar con muchos datos y requiere eficiencia en las consultas semánticas.

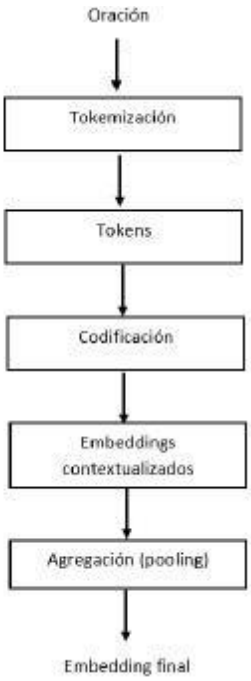
Sentence Transformer

Sentence Transformer es una librería en Python utilizada para generar representaciones vectoriales de alta calidad para oraciones y párrafos completos. A diferencia de los embeddings con palabras tradicionales, los embeddings con Sentence Transformer capturan el significado y la semántica contextual de frases y oraciones completas *(Nils & Iryna, 2019)*.

La librería Sentence Transformer usa modelos basados en la arquitectura Transformer, pre-entrenados para entender el contexto y la estructura de las oraciones. Estos modelos generan embeddings de alta calidad y con comprensión de lenguaje natural.

Figura 4

Arquitectura Sentence Transformer



Nota. Tomado de la documentación Sentence-Transformers que proporciona modelos pre-entrenados para diversas tareas, como búsqueda semántica de texto y recuperación de información.

Capítulo III

Implementación del Sistema

En este capítulo, se llevará a cabo una exposición del proceso de desarrollo llevado a cabo para la creación del sistema actual, el cual se centra en la normalización de entidades biomédicas a través de búsquedas semánticas multilenguaje. El funcionamiento del sistema recibirá un texto candidato con relación a un contexto médico. A partir de este proceso, se generará una salida que comprenderá cuatro aspectos principales: i) El texto que el usuario ha ingresado, ii) La traducción del texto, ofreciendo el resultado del proceso de etiquetado, iii) El resultado obtenido del etiquetador, iiii) La búsqueda del código AUI. Para el proceso de la búsqueda semántica se usará una serie de archivos como el traductor, alineamiento, etiquetado y la similitud encargados de realizar diversos tareas que nos permitirán obtener los resultados mencionados anteriormente, dentro de la Figura *5* se observa el esquema inicial del funcionamiento de cada archivo denominado módulos, desde la traducción del texto a ingles hasta la comprobación de las similitudes de los vectores entre el texto candidato de entrada con los que están almacenados en la base de datos.

Figura 5

Infraestructura de los módulos para el sistema de búsqueda semántica.

De acuerdo a los aspectos principales mencionados, se ha tomado la decisión de adoptar una metodología ágil que garantice la creación de un sistema software de manera organizada, flexible y en un periodo de tiempo determinado. En esta dirección, se ha elegido la metodología Scrum para gestionar el desarrollo del producto, especialmente aquellos con naturaleza intrincada.

Metodología Scrum

La metodología Scrum es una práctica dentro del desarrollo ágil donde nos permitirá realizar actividades y tareas a lo largo el proyecto lo que nos da acceso a implementar un sistema software ágil dentro de un espacio de tiempo determinado.

Las actividades dentro de la Mesología ágil tenemos el Sprint Planning que llevara a cabo una sesión de planificación donde el Team Scrum discute y acuerda las tareas abordar, el Daily Scrum que son reuniones diarias por parte del Development Team con un tiempo de reunión máximo de 10 a 15 minutos donde se detallara el flujo de trabajo de las tareas a desarrollar por el Development Team, el Sprint Review son las revisiones posteriores a cada Sprint donde el Development Team se encarga del análisis respectivo con la finalidad de verificar el cumplimento de las tareas respectivas. Tras esta revisión, se tomarán decisiones respecto a posibles ajustes suplementarios que contribuyan a la optimización continua del sistema, el Sprint retrospective donde se registrarán todos los inconvenientes que surjan durante la ejecución del Sprint. Esto permitirá abordar los obstáculos existentes y aplicar soluciones que conduzcan a la mejora de los procesos para el siguiente sprint.

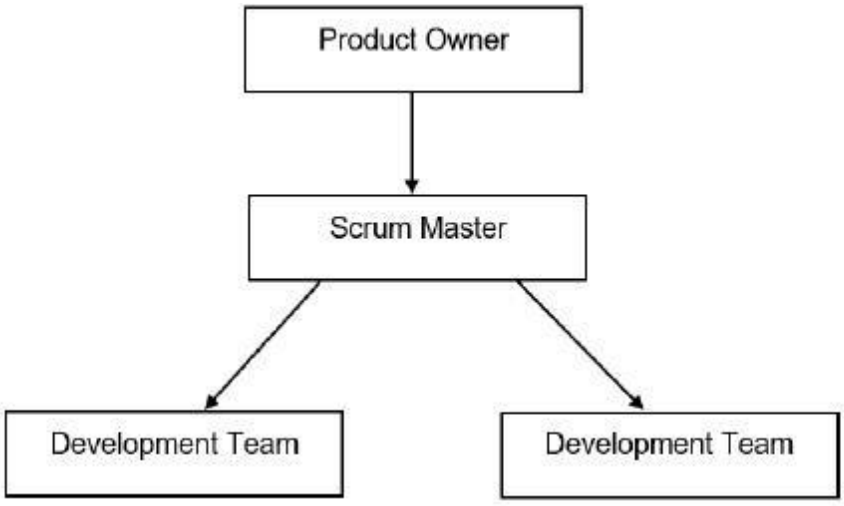
Análisis del Sistema

Antes de adentrarnos en una exploración detallada de la metodología Scrum y sus eventos, se realizará un procedimiento para adquirir los requisitos del sistema. Esto se llevará a cabo mediante la creación de Historias de Usuario (H. U). Estas historias describirán de

manera detallada los roles y las tareas asignadas a cada miembro del equipo de desarrollo dentro del proyecto.

Figura 6

Principales Roles de Scrum dentro del proyecto.



En la Figura 6, se representan los roles asignados a cada miembro del equipo, estableciendo las funciones clave de cada uno. En primera instancia, se encuentra el Product Owner, quien desempeñará un papel central al definir los objetivos a cumplir durante el sprint y determinar los elementos prioritarios del Product Backlog. El Scrum Master es el encargado de asegurarse de seguir el marco de Scrum de manera correcta *(Nader & Frank, 2019)* asignado las tareas y actividades de cada sprint. El equipo de desarrollo *(Development Team)* encargado de desarrollar y ejecutar cada uno de los respectivos Sprint del proyecto.

Team Scrum

Para el desarrollo del sistema informático para la normalización de entidades biomédicas se ha designado roles a cada uno de los miembros del equipo que están dentro del proyecto. En la Tabla 1: Team Scrum se ha identificado y asignado a las personas responsables para realizar cada rol dentro del proyecto.

Tabla 1

Team Scrum

N°	Rol	Integrante	Responsabilidades
1	Product Owner	Uyaguari Uyaguari, Álvaro Danilo Msc.	Tiene la función de guiar el proceso de desarrollo y controlar los tiempos de entrega del producto.
2	Scrum Master	William Ariel Monge Aules	Tiene la función de revisar las actividades y avances de cada Sprint durante el tiempo establecido, es el líder del equipo.
3	Development Team	William Ariel Monge Aules Juan Diego Bahamonde Tonato	Son los responsables del desarrollo de los Sprint e implementación del sistema Software.

Para iniciar con el desarrollo del sistema es importante designar los roles los cuales permitirán identificar a los involucrados que pertenecerán al desarrollo del sistema mediante el Team Scrum, los desarrolladores implementarán y ejecutarán cada sprint, cuyo cumplimiento será revisado por el Scrum Master que decidirá si se mejora el desarrollo para el siguiente sprint. El Scrum Master será el encargado de realizar una reunión preliminar con los desarrolladores del proyecto y el Product Owner. El propósito de esta reunión es brindar información que posibilite la documentación adecuada de las respectivas Historias de Usuario *(Sachdeva, 2016)*.

Historias de Usuario

Para adquirir los requisitos del sistema según se presenta en la Tabla 2, se ha optado por utilizar la técnica de recopilación de información empleada en la metodología ágil denominada *"Historias de Usuario"*. Esto involucra la definición del nombre de cada Historia de Usuario, Rol, características y el resultado final que se lograra obtener en la implementación del sistema.

Tabla 2

Historias de Usuario

ID H. U	Nombre H. U	Rol de la persona	Característica	Resultado final
01	H.U 01	Como Programador	Como desarrollador quiero descargar, instalar y configurar todas las herramientas necesarias para la correcta inicialización del proyecto.	Para inicializar el desarrollo del sistema.
02	H.U 02	Como Cliente Programador	Como programador deseo unir los módulos creados con anterioridad por los anteriores equipos de desarrollo.	Para generar la unión de los módulos de desarrollo.

03	H.U 03	Como Programador	Como programador deseo vectorizar el grupo semántico escogiendo solo los datos en el idioma español e inglés de la base de datos.	Generar el código con modelo Transformer multilenguaje para el Grupo semántico T047 vectorizado.
04	H.U 04	Como Programador	Como programador debe crear un módulo que sea capaza de comprobar la similitud de los vectores tanto como los de la BDD y los de la consulta	Para el desarrollo de la búsqueda semántica.
05	H.U 05	Como Programador	Como programador debo mejorar el rendimiento del sistema y así también corregir los errores del sistema.	Para optimizar los resultados de búsqueda.

Las Historias de Usuario son herramientas diseñadas para agilizar la gestión de requisitos al minimizar la necesidad de documentación formal y el tiempo necesario para su definición (Alexander, et al, 2022).

Product Backlog

En la Tabla 3 se presenta el comienzo del Product Backlog, estructurado en forma de una lista jerarquizada y priorizada. Esta metodología permite al equipo de desarrollo establecer

de manera nítida las tareas o funcionalidades a realizar, considerando el tiempo como un elemento esencial para el proceso de desarrollo. Además, se detallan la fecha de inicio, fecha de finalización y los Sprint correspondientes a cada Historia de Usuario, proporcionando una visión completa del cronograma y la distribución del trabajo a lo largo del proyecto.

Tabla 3

Product Backlog

Historia de Usuario	Nombre	Tiempo (Días)	Fecha de Inicio	Fecha de Fin	N° de Sprint
001	H. U 01	10	08/05/2023	19/05/2023	01
002	H. U 02	20	22/05/2023	16/06/2023	02
003	H. U 03	10	19/06/2023	30/06/2023	03
004	H. U 04	10	03/07/2023	14/07/2023	04
005	H. U 05	10	17/07/2023	28/07/2023	05

Desarrollo del sistema

El sistema permite ingresar un texto el cual pasará por los módulos de traducción, alineamiento y etiquetado, al final de dichos procesos nos dará como resultado el texto relacionado junto a un conjunto de CUI's *(Identificadores Únicos Conceptuales)*.

No obstante, dado que el objetivo principal del sistema es localizar el AUI *(Identificador Único de Átomo)* con la mayor similitud, se procede a la vectorización del texto proporcionado por el etiquetador. Esto implica transformar el texto en un vector numérico.

A través de consultas SQL, se recuperarán los posibles valores de la base de datos, junto con sus respectivos vectores. Luego, se medirá la similitud de los vectores en la base de

datos con el vector del texto ingresado, utilizando la métrica de la distancia del coseno. Cuando esta distancia es igual a 0, indica que los vectores son idénticos, lo que a su vez revela que el valor en la base de datos es el mismo que el introducido por el usuario. Por lo tanto, se puede obtener el AUI correspondiente a ese valor de la base de datos.

Herramientas de Software usadas en el desarrollo.

Tabla 4

Herramientas de desarrollo de Software

Herramienta	Descripción
Docker	Aplicación de código abierto para automatizar el despliegue de aplicaciones dentro de un contenedor de software.
Visual Studio Code	Editor de código fuente para el desarrollo.
Pgadmin	Aplicación para gestionar la gestión de la Bases de Datos PostgreSQL.
Python	Lenguaje de programación utilizado para el desarrollo del sistema (Versión 3.8)

En la Tabla 4 podemos observar las herramientas a usar para el desarrollo del sistema, desde el lenguaje de programación conocido como Python, el gestor de la base de datos (Pgadmin), el editor de código (Visual studio code) y la herramienta Docker que nos ayudará a desplegar la aplicación.

Definición e implementación de módulos

Sprint 01: Descargar, instalar y configurar todas las herramientas de desarrollo.

Los desarrolladores deberán descargar todas las herramientas necesarias para el desarrollo del sistema, dichas herramientas se describen en la Tabla 4, esta tarea se la debe deberá cumplir para dar inicio al desarrollo.

Historia de Usuario 001

Esta historia de usuario describe los roles, tiempos y otras características necesarias, los desarrolladores responsables descritos en la Tabla 5 deberán realizar la actividad en el tiempo determinado para cumplir con los tiempos estimados.

Tabla 5

Historia de Usuario para la Instalación y configuración del software

Historia de Usuario	
Número: H. U 01	**Usuario:** Programador
Nombre de Historia: Instalación y configuración de herramientas	
Prioridad: Alta	**Riesgo de desarrollo:** Bajo
Tiempo de estimación (días): 10	**Interacción Asignada:** 1
Desarrollador responsable: Juan Bahamonde - William Monge	
Descripción: Como desarrollador quiero descargar, instalar y configurar todas las herramientas necesarias para la correcta inicialización del proyecto.	
Valoración: Instalar y configurar las herramientas de desarrollo necesarias para el proyecto.	

Sprint Backlog

En la Tabla 6, se detallan las tareas a llevar a cabo junto con sus respectivos tiempos de ejecución. Cada una de estas tareas tiene una duración de 40 horas y se llevarán a cabo dentro de las fechas límite establecidas.

Tabla 6

Sprint Backlog 1

Sprint 01	**Fecha Inicio**	**08/05/2023**	**Fecha Fin**	**19/05/2023**	**Jornada**	**8**
HU ID	Actividad	Horas	Inicio	Fin	Responsable	Estado
001	Descargar las herramientas	40	08/05/2023	12/05/2023	Juan Bahamonde William Monge	Finalizado
	Configurar las herramientas	40	15/05/2023	19/05/2023		Finalizado

Resultado del Sprint

Los responsables ejecutaron sus actividades de manera precisa y efectiva, cumpliendo con los plazos previstos. Durante el desarrollo de estas tareas, no se registraron errores ni fallos.

Sprint 02: Unión de módulos

Se llevará a cabo la integración de los módulos de traducción, alineamiento y etiquetado en una única secuencia de ejecución. Anteriormente, estos procesos se realizaban de manera individual, lo que generaba complicaciones para el usuario.

Historia de Usuario 002

Tabla 7

Historia de Usuario para la Unión de módulos

Historia de Usuario	
Número: H. U 02	**Usuario:** Cliente Programador
Nombre de Historia: Unión de módulos	
Prioridad: Alta	**Riesgo de desarrollo:** Bajo
Tiempo de estimación (días): 20	**Interacción Asignada:** 2
Desarrollador responsable: Juan Bahamonde - William Monge	
Descripción: Como programador deseo unir los módulos creados con anterioridad por los anteriores equipos de desarrollo.	
Valoración: Como cliente programador deseo ingresar una consulta y me arroje como resultado la consulta vectorizada.	

Sprint Backlog

Tabla 8

Sprint Backlog 2

Sprint 02	**Fecha Inicio**	**22/05/2023**	**Fecha Fin**	**16/06/2023**	**Jornada**	**8**
HU ID	Actividad	Horas	Inicio	Fin	Responsable	Estado
002	Descarga de los módulos	16	22/05/2023	23/05/2023	Juan Bahamonde	Finalizado

				William Monge	
Verificación del funcionamiento de los módulos por separado	24	24/05/2023	26/05/2023	William Monge	Finalizado
Unión de los módulos en una solo línea de producción	64	29/05/2023	07/06/2023	Juan Bahamonde	Finalizado
Verificación del funcionamiento de los módulos unidos	56	08/06/2023	16/06/2023	William Monge	Finalizado

Resultado del Sprint

Las actividades descritas se realizaron de manera exitosa bajo los tiempos establecidos y sin ningún inconveniente por parte de los dos desarrolladores.

Como resultado se obtuvo la correcta unión de los módulos en una sola línea de ejecución facilitando el uso del sistema a los usuarios.

Sprint 03: Vectorización de la base de datos.

Se pretende vectorizar los textos de los CUI relacionados a los conceptos médicos previamente establecidos, dichas vectorizaciones se guardarán en la tabla MRCONSO de la

base de datos UMLS creando una columna llamada Embedding para futuros trabajos relacionados al mismo.

Historia de Usuario 003

Tabla 9

Historia de Usuario para la vectorización de la BDD

Historia de Usuario	
Número: H. U 03	**Usuario:** Programador
Nombre de Historia: Vectorización de la base de datos	
Prioridad: Alta	**Riesgo de desarrollo:** Medio
Tiempo de estimación (días): 10	**Interacción Asignada:** 3
Desarrollador responsable: William Monge	
Descripción: Como programador deseo vectorizar el grupo semántico escogiendo solo los datos con en el idioma de inglés y español de la base de datos.	
Valoración: Como programador al momento de realizar una consulta aparte de los resultados esperados tales como (CUI, Descripción, etc.) también necesito su valor vectorizado.	

Sprint Backlog

Tabla 10

Sprint Backlog 3

Sprint 03	**Fecha Inicio**	**19/06/2023**	**Fecha Fin**	**30/06/2023**	**Jornada**	**8**
HU ID	Actividad	Horas	Inicio	Fin	Responsable	Estado

003	Función encargada de obtener los campos a vectorizar	16	19/06/2023	20/06/2023	William Monge	Finalizado
	Función encargada de vectorizar los campos	32	21/06/2023	26/06/2023	William Monge	Finalizado
	Función de actualización de campo en la BDD con el nuevo campo vectorizado	32	27/06/2023	30/06/2023	William Monge	Finalizado

Resultado del Sprint

Las actividades llevadas a cabo durante el sprint 03 se ejecutaron sin inconvenientes por parte de los desarrolladores y se completaron dentro de los plazos establecidos, cumpliendo así con el cronograma previsto.

Sprint 04: Comparación de los vectores de la BDD y los de la consulta.

En este sprint se desarrollará la comparación de los vectores de la consulta contra los de la base de datos por medio de la distancia del coseno.

El vector de las consultas se obtiene vectorizando las opciones que nos muestra el etiquetador junto a un código CUI, esto nos ayudará en la búsqueda de las posibles posibilidades en la base de datos UMLS.

Historia de Usuario 004

Tabla 11

Historia de Usuario para comparación de vectores

Historia de Usuario	
Número: H. U 04	**Usuario:** Programador
Nombre de Historia: Comprobación de los vectores de la BDD y los de la consulta	
Prioridad: Alta	**Riesgo de desarrollo:** Bajo
Tiempo de estimación (días): 10	**Interacción Asignada:** 4
Desarrollador responsable: Juan Bahamonde	
Descripción: Como programador debo crear un módulo que sea capaz de comprobar la similitud de los vectores tanto como los de la BDD y los de la consulta.	
Valoración: La similitud de los vectores deben tener un error máximo de 20%.	

Sprint Backlog

Tabla 12

Sprint Backlog 4

Sprint 04	**Fecha Inicio**	**03/07/2023**	**Fecha Fin**	**14/07/2023**	**Jornada**	**8**
HU ID	Actividad	Horas	Inicio	Fin	Responsable	Estado

4	Función encargada de obtener los campos vectorizados de la BDD	16	03/07/2023	04/07/2023	Juan Bahamonde	Finalizado
	Función encargada de obtener el valor vectorizado de la consulta	16	05/07/2023	06/07/2023	Juan Bahamonde	Finalizado
	Función encargada de comprobar la similitud de los vectores	48	07/07/2023	14/07/2023	Juan Bahamonde	Finalizado

Resultado del Sprint

Se completaron con éxito las actividades propuestas en el sprint 04, cumpliendo con el cronograma y sin inconvenientes por parte del desarrollador.

Como producto se obtuvo un módulo que es capaz de vectorizar los resultados del etiquetador y por medio de consultas SQL y comprobación de distancias del coseno, el sistema nos arroja el AUI con más similitud al texto ingresado.

Sprint 05: Correcciones de rendimiento.

El propósito central de este sprint consiste en mejorar la eficiencia en el tiempo de ejecución del módulo de similitud. Esto se logrará mediante la optimización de los procedimientos en la base de datos, así como la revisión y optimización de los procesos matemáticos específicos relacionados con dicho módulo.

Historia de Usuario 005

Tabla 13

Historia de Usuario para las correcciones

Historia de Usuario	
Número: H. U 05	**Usuario:** Programador
Nombre de Historia: Correcciones de rendimiento	
Prioridad: Alta	**Riesgo de desarrollo:** Bajo
Tiempo de estimación (días): 10	**Interacción Asignada:** 5
Desarrollador responsable: Juan Bahamonde - William Monge	
Descripción: Como programador debo mejorar el rendimiento del sistema y así también corregir los errores del sistema.	
Valoración: El sistema debe tener un aumento de rendimiento con respecto a la versión anterior.	

Sprint Backlog

Tabla 14

Sprint Backlog 5

Sprint 05	**Fecha Inicio**	**17/07/2023**	**Fecha Fin**	**28/07/2023**	**Jornada**	**8**
HU ID	Actividad	Horas	Inicio	Fin	Responsable	Estado
005	Realización de pruebas	32	17/07/2023	20/07/2023	William Monge	Finalizado
	Informe de las pruebas con los resultados	16	21/07/2023	24/07/2023	William Monge	Finalizado
	Corrección de las funciones con problemas de rendimiento	32	25/07/2023	28/07/2023	William Monge	Finalizado

Resultado del Sprint

Tal como se refleja en la Tabla 14, el desarrollador llevó a cabo las actividades planificadas dentro de los plazos estipulados y sin contratiempos, lo que aseguró el cumplimiento del cronograma establecido. Como resultado directo de estas acciones, el sistema experimentó una mejora en sus tiempos de ejecución, lo cual condujo a un aumento en su eficacia operativa.

Diseño del sistema

Modelo del sistema basado en el C4

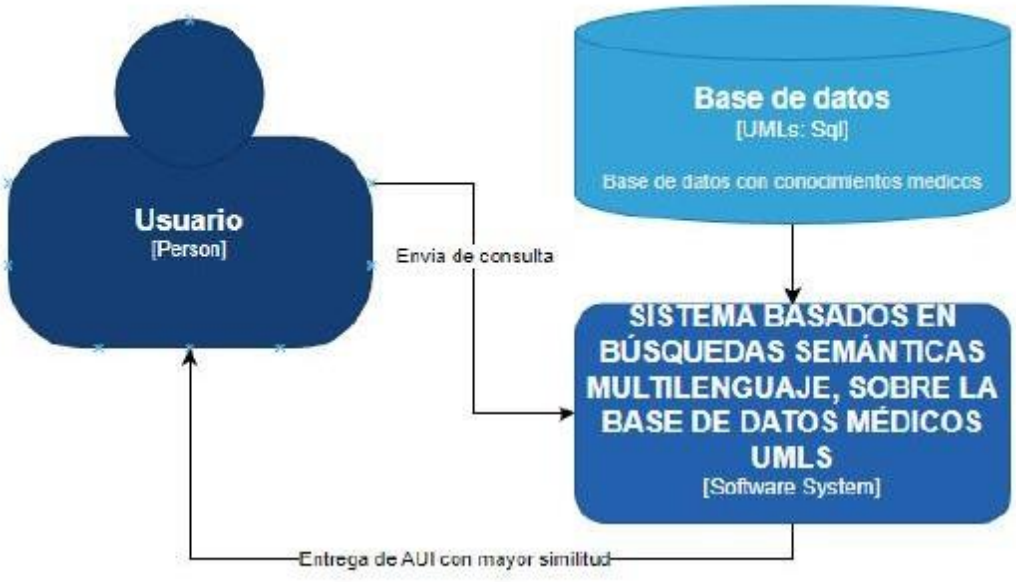
Dentro del diseño para el sistema se ha optado por la utilización de un enfoque C4 que modela las dimensiones estáticas de un sistema de software a través de contenedores (como aplicaciones, almacenes de datos, microservicios, etc.), componentes y el código subyacente que los compone *(Vivanco, 2019)*.

Aunque el modelo C4 originalmente consta de cuatro niveles, en el diseño de este sistema se ha decidido emplear tres niveles, los cuales proporcionan una descripción detallada de la estructura del sistema actual.

Nivel 1: Diagrama de contexto del sistema

Figura 7

Nivel 1 del Modelo C4



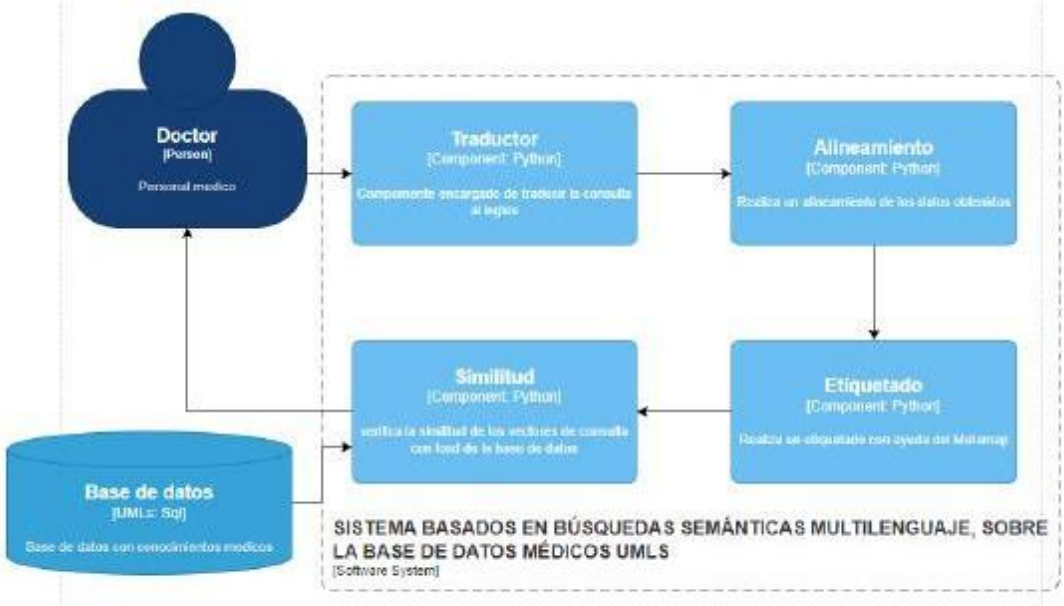
En la Figura 7, se presenta un esquema general del funcionamiento del sistema. En este proceso, el usuario introduce una consulta. Esta consulta es sometida a través del sistema

el cual con la conexión con la base de datos se obtiene un Identificador Único de Átomo (AUI) que corresponde a la similitud vectorial más cercana a la consulta ingresada.

Nivel 2: Diagrama de Contenedores

Figura 8

Nivel 2 del Modelo C4



En la Figura 8 se realiza una exploración más amplia del sistema, proporcionando una comprensión detallada de sus componentes. Entre los componentes que se pueden observar se encuentran:

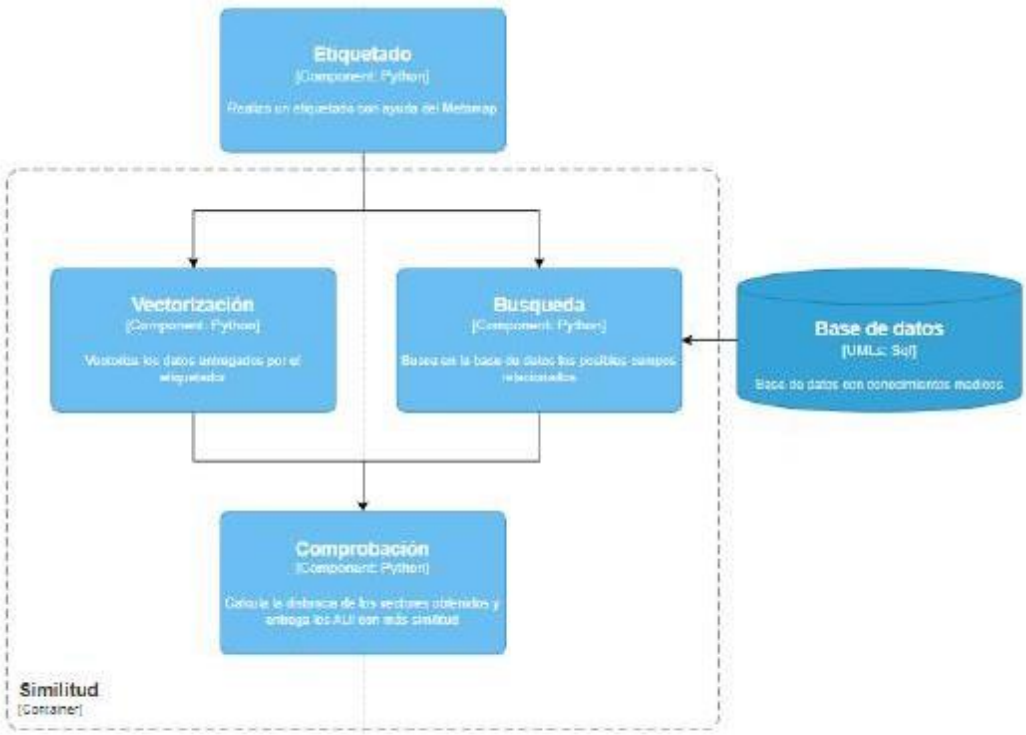
- Traductor: es el encargado de traducir el texto ingresado al idioma inglés por medio de librerías de Python.
- Alineamiento: el trabajo de este es alinear el texto ingresado tanto en español como en inglés.
- Etiquetado: Con los datos obtenidos procede a etiquetar las posibles frases relacionadas a los conceptos médicos establecidos en la base de datos UMLS

- Similitud: genera los vectores de los datos obtenidos por el etiquetador y realiza un proceso de búsqueda en la base de datos para luego comprobar los vectores por medio de la distancia del coseno y así entregar un AUI con mayor similitud.
- Base de datos: la base de datos contiene todos los conceptos médicos normalizados juntos con sus respectivos vectores.

Nivel 3: Diagrama de Componentes

Figura 9

Nivel 3 del Modelo C4



En la Figura 9, se presenta una ampliación del componente de similitud que detalla un nivel adicional de funciones internas. En esta representación, es posible observar cómo el etiquetador transmite sus datos a dos funciones distintas: una función de vectorización y otra función de búsqueda que se conecta con la base de datos. Posteriormente, los resultados obtenidos de estos dos procesos son dirigidos a una función que evalúa la distancia vectorial

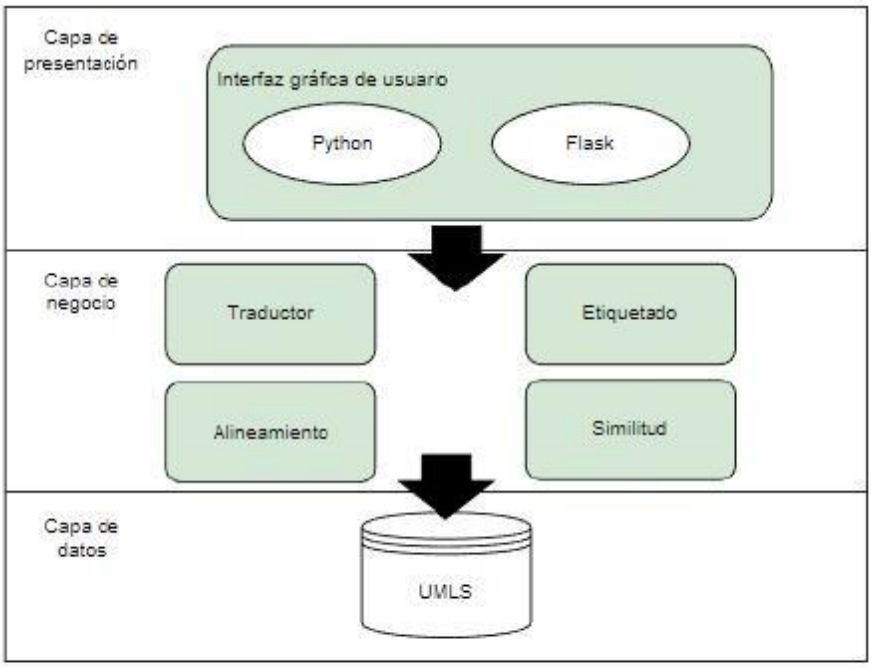
mediante el cálculo del coseno. Esta última función proporciona los Identificadores Únicos de Átomo (AUI) más similares al texto ingresado.

Arquitectura del sistema

El sistema se diseñó siguiendo una arquitectura en capas, como se puede apreciar en la Figura 10. Esta arquitectura está compuesta por tres niveles diferenciados: la capa de presentación, la capa de negocios y la capa de datos. Cada una de estas capas desempeña un rol fundamental en el funcionamiento integral y coordinado del sistema.

Figura 10

Arquitectura en Capas



Nota: La capa de presentación está formada por dos interfaces claramente definidas. La primera interfaz se encarga de la entrada de datos, permitiendo al usuario ingresar la información relevante. Por otro lado, la segunda interfaz tiene como función mostrar los resultados generados por la capa de negocios. La capa de negocios constituye el núcleo del sistema y está compuesta por diversos módulos interconectados. Estos módulos operan en una

única línea de producción, colaborando estrechamente con la capa de datos para generar los resultados requeridos. La capa de datos, por su parte, alberga la base de datos denominada UMLS. Esta base de datos es un depósito de información normalizada relacionada con conceptos médicos. Los datos contenidos son utilizados como recurso esencial por la capa de negocios para llevar a cabo sus operaciones.

Capa de presentación

En esta capa, la interfaz será visible para el usuario. En este espacio, se facilitará la inserción de texto, y a cambio se obtendrán las respuestas generadas por cada uno de los módulos. La construcción de esta interfaz se realizará mediante el uso del lenguaje de programación Python, haciendo uso del framework Flask.

Figura 11

Interfaz gráfica

Nota: En la Figura 11, se puede apreciar cómo el usuario tiene la posibilidad de ingresar texto. Al presionar el botón *"Enviar"*, la capa de negocios toma el control de los procesos lógicos que se llevan a cabo en segundo plano y que no son perceptibles para el usuario.

Figura 12

Interfaz de resultados

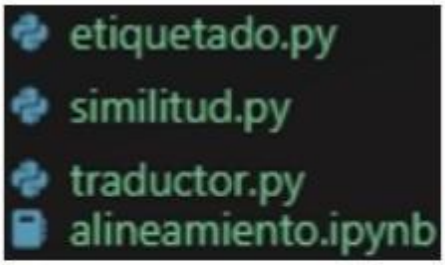
Nota: En la Figura 12, se presentan los datos proporcionados al usuario en cuadros delimitados. En el primero de ellos, se exhibe el texto que el usuario ha ingresado. A continuación, en otro recuadro, se visualiza la traducción del texto, ofreciendo el resultado del proceso de etiquetado. Posteriormente, se presenta el resultado obtenido del etiquetador en un tercer recuadro. Finalmente, en el último recuadro, se muestra el desenlace de la búsqueda del AUI.

Capa de negocios

En esta capa, se llevará a cabo el desarrollo utilizando el lenguaje de programación Python, y se incorporarán las librerías necesarias para la conexión con la base de datos y la vectorización. En este nivel, se alojarán los módulos de traducción, alineamiento, etiquetado y similitud, los cuales serán responsables de abordar la lógica fundamental del sistema. Esta estructura se ilustra en la Figura 13.

Figura 13

Módulos



Enfocándonos en el módulo de similitud, este comienza su proceso estableciendo una conexión con la base de datos a través de la librería psycopg2. Para lograr esta conexión, se deben proporcionar el nombre de usuario, la contraseña, el host, el puerto y el nombre de la base de datos, tal como se representa en la Figura 14.

Figura 14

Conexión a la base de datos UMLS

```
connection = psycopg2.connect(user = "postgres",
                              password = "admin",
                              host = "localhost",
                              port = "5432",
                              database = "umls")
```

Una vez que la conexión ha sido establecida, se da inicio a un primer ciclo iterativo, como se ilustra en la Figura 15. A lo largo de este ciclo, cada dato entregado por el etiquetador es sometido a procesamiento. En este punto, el dato es transformado para adquirir una estructura definida, preparándolo para ser usado en la siguiente fase del proceso, como se muestra en la Figura 16.

Figura 15

Primer ciclo iterativo

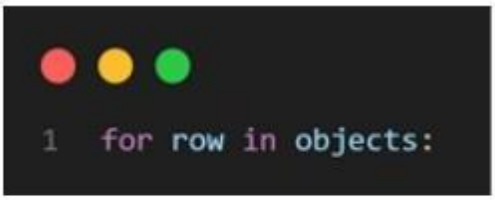
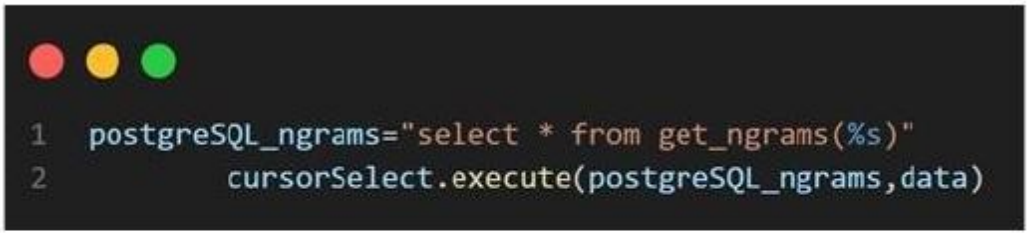


Figura 16

Proceso



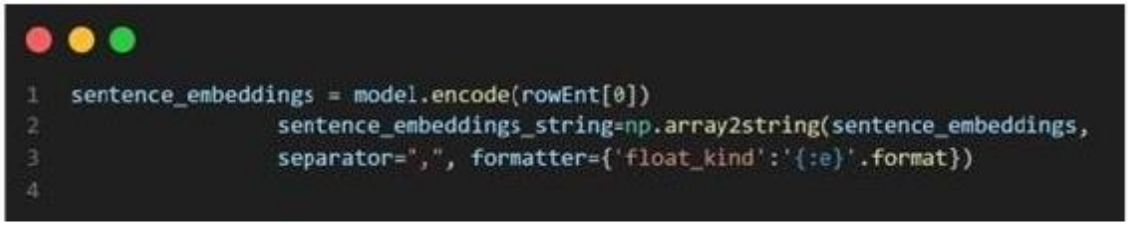
Seguidamente, se ingresa a un segundo ciclo iterativo, como se ilustra en la Figura 17. Durante este ciclo, se recorren los resultados generados para llevar a cabo su vectorización. Posteriormente, los datos se formatean para lograr un manejo estandarizado que sea compatible con la base de datos, como se presenta en la Figura 18. Por último, el vector resultante se incorpora en una consulta SQL.

Figura 17

Segundo Ciclo iterativo

Figura 18

Proceso de vectorización



En el tercer ciclo iterativo *(Figura 19)*, se toman los resultados generados por la consulta SQL y se inicia un proceso iterativo para calcular el coseno de los vectores correspondientes. Este cálculo se realiza mediante el uso de una herramienta proporcionada por pgvector *(Figura 20)*.

Figura 19

Tercer ciclo iterativo

Figura 20

Cálculo del coseno

Luego, se procede a implementar un método de descarte en el cual se evalúan todos los valores de coseno obtenidos. La elección se realiza considerando el AUI asociado al valor del coseno más cercano a cero, tal como se ilustra en la Figura 21.

Figura 21

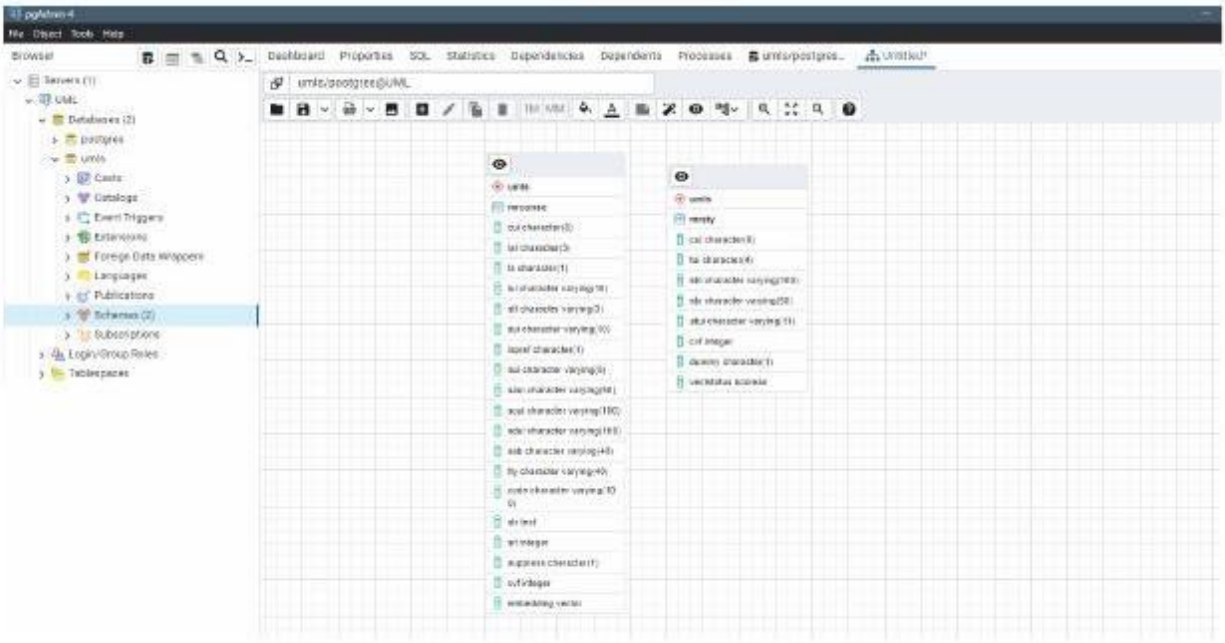
Método de descarte

Capa de datos

En esta capa, la base de datos UMLS desempeña un papel fundamental y se configura como el componente central. En ella se encuentran almacenados tanto los datos normalizados como los vectores esenciales para cada uno de los términos utilizados en el funcionamiento completo del sistema, como se presenta en la Figura 22.

Figura 22

Base de datos UMLS



Capítulo VI

Validación del Sistema

La validación del sistema se apoya en un corpus que se encuentra almacenado en el repositorio de GitHub. Este corpus fue creado de manera manual, lo que significa que cada dato fue revisado y etiquetado por un individuo con conocimientos en el campo médico. Es importante destacar que este corpus desempeñará un papel fundamental en la validación de la capacidad del sistema para obtener los Identificadores Únicos de Átomo (AUI) de manera precisa y adecuada.

Definición y aplicación de métricas de evaluación

Dentro del desarrollo de software, se emplean diversas métricas para evaluar el sistema. En este caso, se han seleccionado las siguientes métricas específicas:

- Fiabilidad: Esta métrica se refiere a la capacidad del sistema para llevar a cabo sus funciones con un nivel de precisión determinado. Su cálculo se basa en la correcta obtención de los Identificadores Únicos de Átomo (AUI).
- Obtención del AUI: Esta métrica consiste en comparar el AUI obtenido por el sistema con el AUI proporcionado por el corpus. Si el AUI coincide con el del corpus, se otorga un punto. En caso contrario, se asigna un valor de 0 puntos.
- Eficiencia: Esta métrica se enfoca en medir el tiempo de respuesta de cada módulo del sistema.

Dentro del sistema, se encuentran los módulos de vectorización, búsqueda y comprobación. Cada uno de estos módulos debe operar dentro de un tiempo predeterminado, que se especifica a continuación:

- 0s – 30s: obtendrá 1 punto.
- 30s – 90s: obtendrá 0,5 puntos.
- +90s: obtendrá 0 puntos.

La elección de estas métricas se fundamenta en la premisa de que el sistema debe ofrecer información precisa en el menor tiempo posible, siendo estos aspectos cruciales para su desempeño óptimo. En esta evaluación, no se han tenido en cuenta elementos como la usabilidad o la seguridad, ya que el enfoque principal radica en la eficacia de la interacción entre el usuario y el sistema. Dado el alcance actual del sistema, estos aspectos no son relevantes para la evaluación.

La asignación de los valores numéricos 1, 0.5 y 0 a las métricas se estableció con el propósito de simplificar el cálculo de los puntajes. En cuanto a los intervalos de tiempo en la sección de eficiencia, estos fueron determinados después de observar el desempeño del sistema en comparación con sistemas similares.

Luego, se procede a llevar a cabo las pruebas del sistema en conjunto con el corpus, tal como se presenta en la Tabla 15.

Tabla 15

Evaluación de las métricas

		Eficiencia		
	Fiabilidad	**Vectorizar**	**Búsqueda**	**Comprobación**
AUI 1	1	1	0	0,5
AUI 2	1	1	0,5	1

AUI 3	1	0,5	0	0,5
AUI 4	1	1	0,5	1
AUI 5	1	1	0	0,5
AUI 6	1	0,5	0,5	1
AUI 7	1	1	0	0,5
AUI 8	1	1	0	1
Total	1	0,875	0,1875	0,8125
	Total		0,625	

Análisis de resultados

Tabla 16

Totales de la evaluación de las métricas

		Eficiencia		
	Fiabilidad	**Vectorizar**	**Búsqueda**	**Comprobación**
Total	1	0,8	0,171875	0,828125
	Total		0,6	

Como se puede apreciar en la Tabla 16, en la sección de fiabilidad, el sistema ha obtenido una puntuación de 1. Esto indica que el sistema logra cumplir su objetivo de brindar un AUI preciso correspondiente a la lexicalización buscada.

En relación a la métrica de eficiencia, hemos obtenido un puntaje total de 0,625. Esto se atribuye principalmente al módulo de búsqueda, el cual presenta tiempos de ejecución excesivos. Esta situación puede deberse a una indexación inadecuada en la base de datos o a la realización de consultas demasiado complejas.

Figura 23

Análisis de la Interfaz de muestra de resultados

La Figura 23 presenta los datos de una manera clara y comprensible, definiendo adecuadamente cada uno de los aspectos requeridos por el usuario, como los resultados de los diferentes módulos utilizados en el sistema.

Es esencial destacar que la interfaz web juega un papel crucial en la facilitación de la interacción con el sistema. Para su construcción, se emplearon lenguajes de programación web como HTML, CSS y el framework Flask, este último basado en Python y destinado al desarrollo web.

Se aplicaron técnicas y prácticas óptimas para el manejo de archivos y componentes dentro del sistema, lo que permite su escalabilidad en el futuro, ya sea mediante la incorporación de nuevos módulos o la optimización de los ya existentes.

Capítulo V

Conclusiones y Recomendaciones

Conclusiones

- Se ha logrado el objetivo de desarrollar un sistema informático para la normalización de entidades biomédicas a través de búsquedas semánticas multilenguaje, utilizando como base de datos médica el UMLS.
- La construcción del marco teórico ha posibilitado una revisión detallada de conceptos sobre búsquedas semánticas y la aplicación de modelos Transformer, así como la identificación de herramientas esenciales para el desarrollo del sistema.
- La implementación de una metodología ágil ha desempeñado un papel clave al fomentar un proceso de desarrollo colaborativo y eficiente, dentro de un marco temporal definido, lo que ha llevado al logro exitoso del sistema.
- El sistema presenta tiempos de ejecución extensos en el módulo de búsqueda que trabaja por medio de consultas enviadas a la base de datos UMLS.
- El diseño de la interfaz de usuario es sencillo y amigable, lo que facilita su uso por parte de los usuarios. Sin embargo, se reconocen áreas de mejora que se detallan en la sección de recomendaciones.
- El enfoque de modelado C4 ha contribuido a una comprensión y visualización claras del funcionamiento del sistema, resultando en una explicación accesible para personas externas al proyecto, permitiendo una comprensión gradual desde los niveles más altos hasta los componentes individuales.

Recomendaciones

- El sistema podría experimentar lentitud en los procesos de búsqueda debido a una posible configuración inadecuada de la base de datos. Por lo tanto, se recomienda implementar índices en las tablas relevantes para mejorar la eficiencia de las consultas. Estas tablas incluyen MRCONSO y MRSTY.
- Aunque el método de vectorización utilizado para el texto es efectivo, existen otros modelos que podrían optimizar este proceso aún más. Considerar la exploración de alternativas podría ser beneficioso en términos de mejora del rendimiento.
- La elección de Flask para la interfaz gráfica ha permitido crear una interfaz visual rápida y sencilla. No obstante, es importante tener en cuenta que, dado que Flask trabaja en conjunto con HTML, CSS y JS, la implementación a nivel de código puede volverse algo compleja. Explorar otras opciones para el diseño de la interfaz podría simplificar esta complejidad.
- Para la vectorización de la base de datos se debería realizar una investigación de nuevos modelos basados en la arquitectura Transformers. Sería valioso considerar esta alternativa para optimizar el proceso de vectorización.
- Se sugiere la revisión y actualización del sistema conforme a la aparición de nuevas *tecnologías.*

Bibliografía

Alexander, M., Gertrudis, L., Juan, P., & Miguel, Á. S. (2022). Historias de Usuario. Scrum Manager®.

Ashish, V., Noam, S. P., Jakob, U., Llion, J., Aidan, N. G., & Łukasz, K. P. (2017). Attention Is All You Need. doi:doi.org/10.48550/arXiv.1706.03762

Avila, D. (s.f.). Búsqueda semántica y Embedding con Cohere. Obtenido de LatinXinAI: https://medium.com/latinxinai/b%C3%BAsqueda-sem%C3%A1ntica-y-embedding-con-cohere-4a14e0209cd9

Bodenreider, O. (2004). The Unified Medical Language System (UMLS):integrating biomedical terminology. Nucleic Acids Research, 32. doi:10.1093/nar/gkh061

Cong, S., Zhihao, Y., Lei, W., Yin, Z., Hongfei, L., & & Jian, W. (2021). Deep learning with language models improves named entity recognition for PharmaCoNER. BMC Bioinformatics, 22. doi:doi.org/10.1186/s12859-021-04260-y

David, E., Miguel, E. R., & Srinivasan, P. (1998). Cross-language information retrieval with the UMLS metathesaurus. Association for Computing Machinery. doi:doi.org/10.1145/290941.290959

Ghajari, A., Fresno, V., & Amigó, E. (2022). Plataforma de exploración de la Composición Semántica a partir de Modelos de Lenguaje pre-entrenados y embeddings estáticos.

Greg, R. (2023). Storing OpenAI embeddings in Postgres with pgvector. Obtenido de supabase: https://supabase.com/blog/openai-embeddings-postgres-vector

Habibi, M., Leon, W., Mariana, N., David, L. W., & Ulf, L. (2017). Deep learning with word embeddings improves biomedical named entity recognition. Bioinformatics, 33, i37–i48. doi:doi.org/10.1093/bioinformatics/btx228

Kuz, A., Mariana, F., & Roxana, S. G. (2018). Comprendiendo la Aplicabilidad de Scrum en el Aula: Herramientas y Ejemplos. Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología, 21, 62-70. doi:10.24215/18509959.21.e07

Lample, G., & Conneau, A. (2019). Cross-lingual language model pretraining. arXiv preprint arXiv:1901.07291.

Liu, Q., Kusner, M. J., & Blunsom, P. (2020). A survey on contextual embeddings. arXiv preprint arXiv:2003.07278.

McCray, A. T., Burgun, A., Bodenreider, O., & Agoncillo, A. V. (2001). Modeling biomedical information for interoperability and decision support. Proceedings of the AMIA Symposium.

Nader, K. R., & Frank, T. (2019). Los Fundamentos de Agile Scrum. (V. Haren, Ed.)

National Library of Medicine. (2016). Obtenido de The Unified Medical Language System (UMLS): https://www.nlm.nih.gov/research/umls/new_users/online_learning/OVR_001.html

Nils, R., & Iryna, G. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. EMNLP. doi:doi.org/10.48550/arXiv.1908.10084

Pallo, T., Brandon, E., Salazar, R., & Adrian, A. (2023). Sistema web para el reconocimiento y la normalización de entidades biomédicas mediante técnicas de cross-lingual. Obtenido de Repositorio Institucional de la Universidad de las Fuerzas Armadas ESPE: http://repositorio.espe.edu.ec/handle/21000/35734

R. Devika, S. Vairavasundaram, C. S. J. Mahenthar, V. Varadarajan and K. Kotecha, "A Deep Learning Model Based on BERT and Sentence Transformer for Semantic Keyphrase Extraction on Big Social Data," in IEEE Access, vol. 9, pp. 165252-165261, 2021, doi: 10.1109/ACCESS.2021.3133651.

Rick, M. (2022). ¿Qué es un Modelo Transformer? Obtenido de NIVIDIA: https://la.blogs.nvidia.com/2022/04/19/que-es-un-modelo-transformer/

Rodríguez, C. &. (2015). ¿Por qué implementar Scrum? Revista ONTARE, 3(1), 125-144. doi:https://doi.org/10.21158/23823399.v3.n1.2015.1253

Ros Pagan, D. (2021). Búsqueda semántica de perfiles en redes sociales (Doctoral dissertation, Universitat Politècnica de València).

Ruder, S., Vulić, I., & Søgaard, A. (2019). A Survey of Cross-lingual Word Embedding Models. Journal of Artificial Intelligence Research, 65, 569-631. doi:https://doi.org/10.1613/jair.1.11640

Sachdeva, S. (2016). Scrum Methodology. International Journal Of Engineering And Computer Science, 5, 16792-16799.

Salyuk Kulinich, M. (2022). Búsqueda de respuestas utilizando redes neuronales (Doctoral dissertation, Universitat Politècnica de València).

Soldaini, L., & Moschitti, A. (2020). The cascade transformer: an application for efficient answer sentence selection. arXiv preprint arXiv:2005.02534.

Tomas, M., Kai, C., Greg, C., & Jeffrey, D. (2013). Efficient Estimation of Word Representations in Vector Space. doi:doi.org/10.48550/arXiv.1301.3781

Valpadasu, H., Sravanthi, T., Naresh, K., Ch, P., C., B. R., & K, M. (2020). Scrum: An Effective Software Development Agile Tool. IOP Conf. Series: Materials Science and Engineering. doi:10.1088/1757-899X/981/2/022060

Vidal-Silva, Cristian L., Sánchez-Ortiz, Aurora, Serrano, Jorge, & Rubio, José M.. (2021). Experiencia académica en desarrollo rápido de sistemas de información web con Python y Django. Formación universitaria, 14(5), 85-94. https://dx.doi.org/10.4067/S0718-50062021000500085

Vivanco, J. (2019). El modelo C4 de documentación para la Arquitectura de Software. Obtenido de Medium: https://medium.com/@javiervivanco/el-modelo-c4-de-documentaci%C3%B3n-para-la-arquitectura-de-software-424704528390

Wang, W., Chen, G., Wang, H., Han, Y., & Chen, Y. (2023). Multilingual Sentence Transformer as A Multilingual Word Aligner. arXiv preprint arXiv:2301.12140.

Anexos

AI content detector report

Sistema informático para la normalización de entidades biomédicas basado en búsquedas semánticas multilenguaje, sobre la base de datos médica UMLS.

Bahamonde Tonato, Juan Diego y Monge Aules, William Ariel

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Software

Trabajo de integración curricular, previo a la obtención del título de Ingeniería en Software

Ing. Uyaguari Uyaguari, Alvaro Danilo Msc.

21 de agosto del 2023
Latacunga

ÍNDICE DE CONTENIDO

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

Capítulo I
Introducción

Propósito y contextualización del tema

El presente proyecto de investigación busca desarrollar un sistema informático a través de una búsqueda semántica multilenguaje mediante nuevos métodos para la identificación de entidades biomédicas en español utilizando técnicas de procesamiento de lenguaje natural y el paradigma de Cross-lingual.

Dentro del proceso central para el desarrollo del sistema, es fundamental tener en cuenta la presencia de tres conceptos esenciales que serán utilizados de manera recurrente. Estos conceptos son conocidos como TUI, CUI y AUI.

El TUI es el código de un grupo semántico escogido denominado “síndromes y

enfermedades" los cuales están compuestos por un código para identificar cada concepto dentro del grupo semántico denominado CUI, para la identificación de cada lexicalización dentro del CUI se asigna un Identificador único del Átomo (AUI). Esta asignación de AUI garantiza que todos los términos se identifiquen, lo que facilita la búsqueda de información de manera precisa.
Es decir, que la búsqueda semántica se refiere a la capacidad del sistema para comprender el contexto y significado de un término médico, no solo la palabra exacta que lo representa. Esto se logra a través de una amplia base de datos UMLS que integra y relaciona terminologías médicas de diferentes fuentes, proporcionando un vocabulario unificado y enriquecido (Bodenreider, 2004).
En este contexto, el desarrollo del sistema dentro del presente proyecto es unificar los conceptos o entidades para que el sistema contribuirá en gran medida a mejorar la calidad, el diagnóstico de síndromes, el desarrollo de tratamientos y la toma de decisiones clínicas.

Justificación del interés de la investigación
Es importante conocer, que los modelos de predicción clínica se aplican comúnmente en la práctica médica para ayudar a los profesionales de la salud a determinar el diagnóstico o pronóstico de un paciente. En la actualidad, existen varios enfoques y métodos para realizar este proceso de predicción, dependiendo de la naturaleza de los datos y de los recursos disponibles en el idioma en el que se redacten las notas médicas.
Al respecto el reconocimiento y la normalización de entidades biomédicas en notas clínicas, es un recurso fundamental para encontrar rasgos en el texto y con estos realizar predicciones de diagnósticos y tratamientos médicos, por lo que encontrar nuevos métodos que no requieran corpus etiquetados para realizar la identificación y normalización de entidades biomédicas es de mucha importancia en el área de la bioinformática y la analítica de datos.
Por tal motivo, es importante realizar esta investigación, a fin de identificar y buscar nuevos métodos y técnicas que permitan realizar la identificación y la normalización de entidades biomédicas.
Objetivo general
Desarrollar un sistema informático para la normalización de entidades biomédicas basado en búsquedas semánticas multilenguaje, sobre la base de datos médica UMLS.
Objetivos específicos
1) Investigar nuevos métodos para la normalización de conceptos biomédicos con un enfoque de cross lingual.
2) Desarrollar bases teórico conceptuales de cómo aplicar nuevos métodos para la normalización de entidades biomédicas mediante el uso de técnicas de procesamiento del lenguaje natural con un enfoque de cross-lingual.
3) Describir el desarrollo del sistema software frameworks y arquitecturas actuales

4) Realizar la validación de los datos, la búsqueda semántica y la similitud que tuvo entre la base de datos y la petición de entrada.
Metodología
El proyecto consiste en desarrollar un sistema capaz de reconocer entidades biomédicas en español a través de búsquedas semánticas multilenguaje aplicando métodos para la detección de entidades médicas.
De esta manera, se cumplirán los siguientes objetivos clave. El proyecto se estructura en tres fases metodológicas principales:
En la primera fase de este proyecto, se llevará a cabo un análisis detallado de la literatura científica actual y de nuevos métodos relacionados con la estandarización de entidades biomédicas. Se emplea un enfoque teórico - lógico para examinar y distinguir entre los diversos métodos que se aplican en este ámbito. Es decir, lo más importante de esta fase investigativa consiste en adquirir una comprensión profunda tanto de las limitaciones presentes como de las posibilidades que se presentan para optimizar el proceso de normalización.
En la segunda fase, utilizaremos nuevos métodos para representar los términos o conceptos médicos en forma de vectores numéricos. El modelo de lenguaje per-entrenado BERTS (Bidirecionale Encoder Representations from Transformers) nos permitirá asignar oraciones a un espacio vectorial (Nils & Iryna, 2019), capturando un grupo semántico específico y relacionarlas contextualmente con las entidades médicas, utilizando la base de datos UMLS (Unified Medical Lenguage System) como fuente principal para la normalización de las entidades.
Es decir, esto nos permitirá realizar búsquedas semánticas tanto en el idioma del inglés como español, con la finalidad de normalizar o asignar un código para las entidades biomédicas.

En la tercera fase, se implementará un método empírico donde el sistema se someterá

a ser evaluado por medio de métricas para verificar su eficiencia y precisión mediante los resultados obtenidos, existirán módulos que serán evaluados mediante un tiempo predeterminado para controlar la precisión de los resultados.

Capítulo II
Marco Teórico

En este capítulo se abordará la exposición de conceptos, así como la presentación de modelos de procesamiento de lenguaje natural. Se realizará una investigación sobre métodos y técnicas que permitan llevar a cabo una búsqueda semántica, implementado herramientas que manejen modelos Transformes. Estos modelos están diseñados para capturar patrones basados en secuencias, los cuales serán transformados en representaciones numéricas conocidas como embeddings vectoriales.

La implementación de las herramientas para la realización de una búsqueda semántica con embeddings facilitará el proceso de transformación, donde son importantes para modelos de lenguaje avanzados, como los basados en la arquitectura Transformer. Estas herramientas facilitan la representación numérica del significado y contexto de palabras, frases o documentos, para mejorar la búsqueda de información relevante en grandes conjuntos de datos textuales (Ashish, et al, 2017).

UMLS

Conceptos

UMLS (Unified Medical Language System) sus siglas lo definen como un sistema de lenguaje médico unificado, donde se permite obtener una enciclopedia o biblioteca, en la que se recopila una cantidad sumamente amplia de datos relacionados a vocabularios médicos como síndromes, síntomas, enfermedades, estándares de salud, etc.

UMLS tiene la característica principal de proporcionar a los desarrolladores información sobre conceptos pertenecientes al campo de la medicina, así como funcionalidades de búsqueda proporcionados para usuarios poco técnicos en el campo (National Library of Medicine, 2016).

Las tres fuentes de conocimiento UMLS las cuales se definen como:

- El Metathesaurus: Es el encargado de proporcionar una biblioteca con más de 5 millones de conceptos, términos o nombres dentro del campo de la biomedicina.
- Red Semántica: Dentro de la Red semántica del UMLS existen 133 categorías y 54 relaciones entre las categorías para etiquetar los dominós en el campo de la biomedicina.
- Herramientas Léxicas: proporciona información léxica que es utilizada para el procesamiento del lenguaje.

Figura 1
UMLS (Unified Medical Language System)

Nota. Tomado de la página web National Library of Medicine “The Unified Medical Language System (UMLS)”

Átomos

Dentro del Metathesaurus existe un identificador único conocido como átomo que la National Library of Medicine (National Library of Medicine, 2016) define como “nombres conceptuales o las cadenas de cada uno de los vocabularios de origen”

AUI es un identificador único capaz de especificar y diferenciar cada concepto del uno con el otro dando así una asignación única a cada concepto, dentro del formato AUI contiene una combinación de letras números alternativos para diferenciación de los conceptos dentro de la base de datos UMLS.

Relaciones entre conceptos

Dentro del UMLS existen relaciones entre los conceptos que establecen conexiones semánticas formando un Red Semántica, dentro existen tipos y relaciones semánticas. Los tipos semánticos están formados por las categorías como síndromes, enfermedades o fármacos, por otro lado, las relaciones semánticas utilizan una relación existente entre los tipos semánticos dando apoyo a la interpretación del significado (National Library of Medicine, 2016).

Figura 2
Red Semántica

Nota. Tomado de la página web National Library of Medicine “The Semantic Network”

Reconocimiento de entidades biomédicas con un proceso de Cross-lingual

El reconocimiento de entidades biomédicas mediante un proceso de cross-lingual implica la identificación y etiquetado de términos médicos clave en distintos idiomas. Este

proceso se basa en la habilidad para comprender y establecer conexiones entre contextos médicos a lo largo de diversas lenguas. Este método emplea la traducción y vinculación de

términos médicos en varios idiomas, con el propósito de lograr un reconocimiento de entidades biomédicas preciso y coherente en entornos multilingües.

Las representaciones lingüísticas en diferentes idiomas nos brindan la capacidad de comprender el significado de las palabras en contextos multilingües, y desempeñan un papel fundamental como facilitadoras en la transferencia entre lenguajes al construir modelos para el procesamiento de lenguaje natural (Ruder, Vulić, & Søgaard, 2019).

Los modelos basados en Arquitectura Transformer han destacado por su mejor rendimiento comparado con enfoques tradicionales en aplicaciones biomédicas dentro del procesamiento del lenguaje natural. (Cong, et al, 2021).

Normalización de entidades biomédicas

En el campo de la medicina existe una gran variedad de lexicalizaciones principalmente relacionadas a sintomatologías. Gracias a esto se genera la necesidad de normalizar los términos que sean similares relacionados a conceptos específicos, esto se logra a través de la normalización de entidades biomédicas la cual se encarga de agrupar las lexicalizaciones en grupos semánticos con relación a los conceptos médicos. Utilizando identificadores únicos para los dispositivos y así mejorar la calidad y compatibilidad de los datos en la industria de la salud (Pallo, et al, 2023). Ayudando a los investigadores a seguir y relacionar información de forma más efectiva.

Búsqueda semántica

Búsqueda semántica para normalizar entidades biomédicas

La búsqueda semántica es una técnica avanzada para normalizar y categorizar términos biomédicos como genes, proteínas, enfermedades y medicamentos. La normalización es necesaria para asegurar la uniformidad y exactitud al analizar información biomédica.

La búsqueda semántica utiliza modelos de lenguaje avanzados, como las redes neuronales basadas en arquitectura Transformer, eficientes para comprender las complicadas y

contextuales relaciones entre los términos biomédicos. Estas técnicas utilizan un aprendizaje profundo y enfoque para comprender el sentido de las palabras en su contexto. Algunos estudios han empleado exitosamente modelos basados en Transformers para normalizar entidades biomédicas (Habibi et al., 2023).

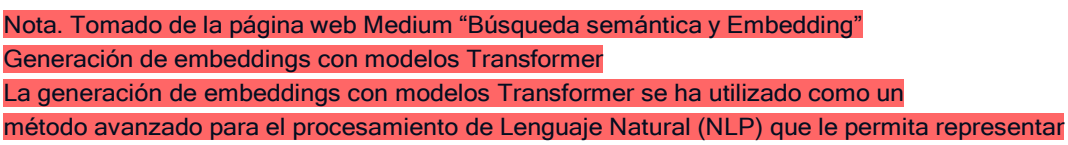
Búsqueda semántica con embeddings

La búsqueda semántica con embedding se define como una técnica dentro del procesamiento del lenguaje natural (NLP), capaz de encontrar y mejorar las relaciones semánticas entre el vocabulario de palabras dentro de un corpus de texto (Tomas et al., 2013). Esta técnica de Lenguaje Natural (NL) utiliza vectores conocidos como embedding para la representación contextual de las palabras.

Embedding permite procesar y transformar fragmentos de texto para expresar su significado con respecto a las relaciones semánticas entre palabras dentro de un espacio vectorial (Avila, s.f.). Así, al realizar una búsqueda semántica, se podrá calcular la similitud entre el vector de consulta y los vectores de otras palabras o expresiones en el corpus y así poder identificar los conceptos relacionados.

Figura 3

Funcionamiento de Embeddings



Nota. Tomado de la página web Medium “Búsqueda semántica y Embedding”

Generación de embeddings con modelos Transformer

La generación de embeddings con modelos Transformer se ha utilizado como un método avanzado para el procesamiento de Lenguaje Natural (NLP) que le permita representar el significado de las palabras o frases como vectores numéricos. Existen modelos Transformer, como GTP Y BERT, son un tipo de modelos de lenguaje natural que han demostrado tener mucho éxito gracias a su capacidad de capturar los patrones más complejos y las relaciones semánticas.

En la actualidad se ha centrado el enfoque de los modelos Transformer como atención o atracción propia, esto explica que como técnica permite enfocarse en ciertas partes específicas de un texto durante el proceso de codificación y descodificación (Rick, 2022). Este enfoque contextualizado posibilita crear Embedding existentes y con mayor comprensión del Lenguaje Natural. Un estudio realizado por (Ashish et al., 2017) reveló que “Los modelos

Transformer son mejores que los enfoques anteriores en muchas tareas de NLP debido a su habilidad para entender las relaciones complicadas entre palabras y su contexto”.

Herramientas para la realización de una búsqueda semántica con embeddings

PGVector

PGVector es una extensión de PostgreSQL donde se permite buscar de manera semántica en la base de datos utilizando vectores. Este recurso permite almacenar y manipular vectores de múltiples dimensiones dentro de la base de datos, siendo de esta manera muy útil en aplicaciones que requieren consultas semánticas rápidas (Greg, 2023).

La extensión PGVector permite almacenar embeddings de datos como palabras o frases en forma de vectores dentro de la base de datos para acceder y editar fácilmente sin transferirlos continuamente a la aplicación cliente. Esto es muy útil al trabajar con muchos datos y requiere eficiencia en las consultas semánticas.

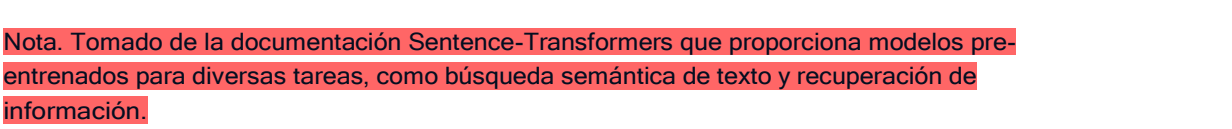
Sentence Transformer

Sentence Transformer es una librería en Python utilizada para generar representaciones vectoriales de alta calidad para oraciones y párrafos completos. A diferencia de los embeddings con palabras tradicionales, los embeddings con Sentence Transformer capturan el significado y la semántica contextual de frases y oraciones completas (Nils & Iryna, 2019).

La librería Sentence Transformer usa modelos basados en la arquitectura Transformer, pre-entrenados para entender el contexto y la estructura de las oraciones. Estos modelos generan embeddings de alta calidad y con comprensión de lenguaje natural.

Figura 4

Arquitectura Sentence Transformer



Nota. Tomado de la documentación Sentence-Transformers que proporciona modelos pre-entrenados para diversas tareas, como búsqueda semántica de texto y recuperación de información.

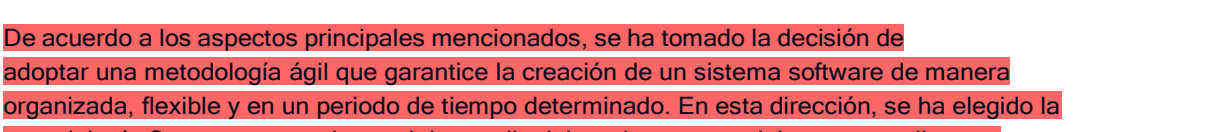
Capítulo III

Implementación del Sistema

En este capítulo, se llevará a cabo una exposición del proceso de desarrollo llevado a cabo para la creación del sistema actual, el cual se centra en la normalización de entidades biomédicas a través de búsquedas semánticas multilenguaje. El funcionamiento del sistema recibirá un texto candidato con relación a un contexto médico. A partir de este proceso, se generará una salida que comprenderá cuatro aspectos principales: i) El texto que el usuario ha ingresado, ii) La traducción del texto, ofreciendo el resultado del proceso de etiquetado, iii) El resultado obtenido del etiquetador, iiii) La búsqueda del código AUI. Para el proceso de la búsqueda semántica se usará una serie de archivos como el traductor, alineamiento, etiquetado y la similitud encargados de realizar diversos tareas que nos permitirán obtener los resultados mencionados anteriormente, dentro de la Figura 5 se observa el esquema inicial del funcionamiento de cada archivo denominado módulos, desde la traducción del texto a ingles hasta la comprobación de las similitudes de los vectores entre el texto candidato de entrada con los que están almacenados en la base de datos.

Figura 5

Infraestructura de los módulos para el sistema de búsqueda semántica.



De acuerdo a los aspectos principales mencionados, se ha tomado la decisión de adoptar una metodología ágil que garantice la creación de un sistema software de manera organizada, flexible y en un periodo de tiempo determinado. En esta dirección, se ha elegido la metodología Scrum para gestionar el desarrollo del producto, especialmente aquellos con naturaleza intrincada.

Metodología Scrum

La metodología Scrum es una práctica dentro del desarrollo ágil donde nos permitirá realizar actividades y tareas a lo largo el proyecto lo que nos da acceso a implementar un sistema software ágil dentro de un espacio de tiempo determinado.

Las actividades dentro de la Mesología ágil tenemos el Sprint Planning que llevara a cabo una sesión de planificación donde el Team Scrum discute y acuerda las tareas abordar, el Daily Scrum que son reuniones diarias por parte del Development Team con un tiempo de reunión máximo de 10 a 15 minutos donde se detallara el flujo de trabajo de las tareas a desarrollar por el Development Team, el Sprint Review son las revisiones posteriores a cada Sprint donde el Development Team se encarga del análisis respectivo con la finalidad de verificar el cumplimento de las tareas respectivas. Tras esta revisión, se tomarán decisiones respecto a posibles ajustes suplementarios que contribuyan a la optimización continua del sistema, el Sprint retrospective donde se registrarán todos los inconvenientes que surjan durante la ejecución del Sprint. Esto permitirá abordar los obstáculos existentes y aplicar soluciones que conduzcan a la mejora de los procesos para el siguiente sprint.

Análisis del Sistema

Antes de adentrarnos en una exploración detallada de la metodología Scrum y sus eventos, se realizará un procedimiento para adquirir los requisitos del sistema. Esto se llevará a cabo mediante la creación de Historias de Usuario (H. U). Estas historias describirán de

manera detallada los roles y las tareas asignadas a cada miembro del equipo de desarrollo dentro del proyecto.

Figura 6
Principales Roles de Scrum dentro del proyecto.

En la Figura 6, se representan los roles asignados a cada miembro del equipo, estableciendo las funciones clave de cada uno. En primera instancia, se encuentra el Product Owner, quien desempeñará un papel central al definir los objetivos a cumplir durante el sprint y determinar los elementos prioritarios del Product Backlog. El Scrum Master es el encargado de asegurarse de seguir el marco de Scrum de manera correcta (Nader & Frank, 2019) asignado las tareas y actividades de cada sprint. El equipo de desarrollo (Development Team) encargado de desarrollar y ejecutar cada uno de los respectivos Sprint del proyecto.

Team Scrum

Para el desarrollo del sistema informático para la normalización de entidades biomédicas se ha designado roles a cada uno de los miembros del equipo que están dentro del proyecto. En la Tabla 1: Team Scrum se ha identificado y asignado a las personas responsables para realizar cada rol dentro del proyecto.

Tabla 1
Team Scrum

N°	Rol	Integrante	Responsabilidades
1	Product Owner	Uyaguari Uyaguari, Álvaro Danilo Msc.	Tiene la función de guiar el proceso de desarrollo y controlar los tiempos de entrega del producto.
2	Scrum Master	William Ariel Monge Aules	Tiene la función de revisar las actividades y avances de cada Sprint durante el tiempo establecido, es el líder del equipo.
3	Development Team	William Ariel Monge Aules Juan Diego Bahamonde Tonato	Son los responsables del desarrollo de los Sprint e implementación del sistema

Software.

Para iniciar con el desarrollo del sistema es importante designar los roles los cuales permitirán identificar a los involucrados que pertenecerán al desarrollo del sistema mediante el Team Scrum, los desarrolladores implementarán y ejecutarán cada sprint, cuyo cumplimiento será revisado por el Scrum Master que decidirá si se mejora el desarrollo para el siguiente sprint. El Scrum Master será el encargado de realizar una reunión preliminar con los desarrolladores del proyecto y el Product Owner. El propósito de esta reunión es brindar información que posibilite la documentación adecuada de las respectivas Historias de Usuario (Sachdeva, 2016).

Historias de Usuario

Para adquirir los requisitos del sistema según se presenta en la Tabla 2, se ha optado por utilizar la técnica de recopilación de información empleada en la metodología ágil denominada "Historias de Usuario". Esto involucra la definición del nombre de cada Historia de Usuario, Rol, características y el resultado final que se lograra obtener en la implementación del sistema.

Tabla 2

Historias de Usuario

ID H. U	Nombre H. U	Rol de la persona	Característica	Resultado final
001	H.U 01	Como Programador	Como desarrollador quiero descargar, instalar y configurar todas las herramientas necesarias para la correcta inicialización del proyecto.	Para inicializar el desarrollo del sistema.
002	H.U 02	Como Cliente Programador	Como programador deseo unir los módulos creados con anterioridad por los anteriores equipos de desarrollo.	Para generar la unión de los

módulos de
desarrollo.

003

H.U 03

Como
Programador

Como programador deseo
vectorizar el grupo
semántico escogiendo solo
los datos en el idioma
español e inglés de la base
de datos.

Generar el código
con modelo
Transformer
multilenguaje
para el Grupo
semántico T047
vectorizado.

004

H.U 04

Como
Programador

Como programador debe
crear un módulo que sea
capaza de comprobar la
similitud de los vectores
tanto como los de la BDD y
los de la consulta

Para el desarrollo
de la búsqueda
semántica.

005

H.U 05

Como
Programador

Como programador debo
mejorar el rendimiento del
sistema y así también
corregir los errores del
sistema.

Para optimizar los
resultados de
búsqueda.

Las Historias de Usuario son herramientas diseñadas para agilizar la gestión de requisitos al minimizar la necesidad de documentación formal y el tiempo necesario para su definición (Alexander, et al, 2022).
Product Backlog

En la Tabla 3 se presenta el comienzo del Product Backlog, estructurado en forma de una lista jerarquizada y priorizada. Esta metodología permite al equipo de desarrollo establecer

de manera nítida las tareas o funcionalidades a realizar, considerando el tiempo como un elemento esencial para el proceso de desarrollo. Además, se detallan la fecha de inicio, fecha de finalización y los Sprint correspondientes a cada Historia de Usuario, proporcionando una visión completa del cronograma y la distribución del trabajo a lo largo del proyecto.

Tabla 3

Product Backlog

Historia de Usuario	Nombre	Tiempo (Días)	Fecha de Inicio	Fecha de Fin	N° de Sprint
001	H. U 01	10	08/05/2023	19/05/2023	01
002	H. U 02	20	22/05/2023	16/06/2023	02
003	H. U 03	10	19/06/2023	30/06/2023	03
004	H. U 04				

10

03/07/2023

14/07/2023

04

005

H. U 05

10

17/07/2023

28/07/2023

05

Desarrollo del sistema
El sistema permite ingresar un texto el cual pasará por los módulos de traducción, alineamiento y etiquetado, al final de dichos procesos nos dará como resultado el texto relacionado junto a un conjunto de CUI's (Identificadores Únicos Conceptuales).
No obstante, dado que el objetivo principal del sistema es localizar el AUI (Identificador Único de Átomo) con la mayor similitud, se procede a la vectorización del texto proporcionado por el etiquetador. Esto implica transformar el texto en un vector numérico.
A través de consultas SQL, se recuperarán los posibles valores de la base de datos, junto con sus respectivos vectores. Luego, se medirá la similitud de los vectores en la base de

datos con el vector del texto ingresado, utilizando la métrica de la distancia del coseno. Cuando esta distancia es igual a 0, indica que los vectores son idénticos, lo que a su vez revela que el valor en la base de datos es el mismo que el introducido por el usuario. Por lo tanto, se puede obtener el AUI correspondiente a ese valor de la base de datos.
Herramientas de Software usadas en el desarrollo.
Tabla 4
Herramientas de desarrollo de Software

Herramienta

Descripción

Docker

Aplicación de código abierto para automatizar el despliegue de aplicaciones dentro de un contenedor de software.

Visual Studio Code

Editor de código fuente para el desarrollo.

Pgadmin

Aplicación para gestionar la gestión de
la Bases de Datos PostgreSQL.

Python

Lenguaje de programación utilizado
para el desarrollo del sistema (Versión
3.8)

En la Tabla 4 podemos observar las herramientas a usar para el desarrollo del sistema, desde el lenguaje de programación conocido como Python, el gestor de la base de datos (Pgadmin), el editor de código (Visual studio code) y la herramienta Docker que nos ayudará a desplegar la aplicación.

Definición e implementación de módulos
Sprint 01: Descargar, instalar y configurar todas las herramientas de desarrollo.
Los desarrolladores deberán descargar todas las herramientas necesarias para el desarrollo del sistema, dichas herramientas se describen en la Tabla 4, esta tarea se la debe deberá cumplir para dar inicio al desarrollo.
Historia de Usuario 001
Esta historia de usuario describe los roles, tiempos y otras características necesarias, los desarrolladores responsables descritos en la Tabla 5 deberán realizar la actividad en el tiempo determinado para cumplir con los tiempos estimados.
Tabla 5
Historia de Usuario para la Instalación y configuración del software

Historia de Usuario

Número: H. U 01

Usuario: Programador

Nombre de Historia: Instalación y configuración de herramientas

Prioridad: Alta

Riesgo de desarrollo: Bajo

Tiempo de estimación (días):
10

Interacción Asignada: 1

Desarrollador responsable: Juan Bahamonde - William Monge

Descripción: Como desarrollador quiero descargar, instalar y configurar todas las herramientas necesarias para la correcta inicialización del proyecto.

Valoración: Instalar y configurar las herramientas de desarrollo necesarias para el proyecto.

Sprint Backlog

En la Tabla 6, se detallan las tareas a llevar a cabo junto con sus respectivos tiempos de ejecución. Cada una de estas tareas tiene una duración de 40 horas y se llevarán a cabo dentro de las fechas límite establecidas.
Tabla 6
Sprint Backlog 1

Sprint

01

Fecha Inicio

08/05/2023

Fecha Fin

19/05/2023

Jornada

8

HU ID

Actividad

Horas

Inicio

Fin

Responsable

Estado

001

Descargar las herramientas

40

08/05/2023

12/05/2023

Juan Bahamonde
William Monge

Finalizado

Configurar las herramientas

40

15/05/2023

19/05/2023

Finalizado

Resultado del Sprint
Los responsables ejecutaron sus actividades de manera precisa y efectiva, cumpliendo con los plazos previstos. Durante el desarrollo de estas tareas, no se registraron errores ni fallos.
Sprint 02: Unión de módulos

Se llevará a cabo la integración de los módulos de traducción, alineamiento y etiquetado en una única secuencia de ejecución. Anteriormente, estos procesos se realizaban de manera individual, lo que generaba complicaciones para el usuario.
Historia de Usuario 002

Tabla 7
Historia de Usuario para la Unión de módulos

Historia de Usuario

Número: H. U 02

Usuario: Cliente Programador

Nombre de Historia: Unión de módulos

Prioridad: Alta

Riesgo de desarrollo: Bajo

Tiempo de estimación (días):
20

Interacción Asignada: 2

Desarrollador responsable: Juan Bahamonde - William Monge

Descripción: Como programador deseo unir los módulos creados con anterioridad por los anteriores equipos de desarrollo.

Valoración: Como cliente programador deseo ingresar una consulta y me arroje como resultado la consulta vectorizada.

Sprint Backlog
Tabla 8
Sprint Backlog 2

Sprint
02

Fecha Inicio

22/05/2023

Fecha Fin

16/06/2023

Jornada

8

HU ID

Actividad

Horas

Inicio

Fin

Responsable

Estado

002

Descarga de los módulos

16

22/05/2023

23/05/2023

Juan Bahamonde

Finalizado

William Monge

Verificación del funcionamiento de los módulos por separado

24

24/05/2023

26/05/2023

William Monge

Finalizado

Unión de los módulos en una solo línea de producción

64

29/05/2023

07/06/2023

Juan Bahamonde

Finalizado

Verificación del funcionamiento de los módulos unidos

56

08/06/2023

16/06/2023

William

Monge

Finalizado

Resultado del Sprint
Las actividades descritas se realizaron de manera exitosa bajo los tiempos establecidos y sin ningún inconveniente por parte de los dos desarrolladores.
Como resultado se obtuvo la correcta unión de los módulos en una sola línea de ejecución facilitando el uso del sistema a los usuarios.
Sprint 03: Vectorización de la base de datos.
Se pretende vectorizar los textos de los CUI relacionados a los conceptos médicos previamente establecidos, dichas vectorizaciones se guardarán en la tabla MRCONSO de la

base de datos UMLS creando una columna llamada Embedding para futuros trabajos relacionados al mismo.
Historia de Usuario 003
Tabla 9
Historia de Usuario para la vectorización de la BDD

Historia de Usuario

Número: H. U 03

Usuario: Programador

Nombre de Historia: Vectorización de la base de datos

Prioridad: Alta

Riesgo de desarrollo: Medio

Tiempo de estimación (días):
10

Interacción Asignada: 3

Desarrollador responsable: William Monge

Descripción: Como programador deseo vectorizar el grupo semántico escogiendo solo los datos con en el idioma de inglés y español de la base de datos.

Valoración: Como programador al momento de realizar una consulta aparte de los resultados esperados tales como (CUI, Descripción, etc.) también necesito su valor vectorizado.

Sprint Backlog
Tabla 10
Sprint Backlog 3

Sprint
03

Fecha Inicio

19/06/2023

Fecha Fin

30/06/2023

Jornada

8

HU ID

Actividad

Horas

Inicio

Fin

Responsable

Estado

003

Función encargada de obtener los campos a vectorizar

16

19/06/2023

20/06/2023

William Monge

Finalizado

Función encargada de vectorizar los campos

32

21/06/2023

26/06/2023

William Monge

Finalizado

Función de actualización de campo en la BDD con el nuevo campo vectorizado

32

27/06/2023

30/06/2023

William Monge

Finalizado

Resultado del Sprint
Las actividades llevadas a cabo durante el sprint 03 se ejecutaron sin inconvenientes por parte de los desarrolladores y se completaron dentro de los plazos establecidos, cumpliendo así con el cronograma previsto.
Sprint 04: Comparación de los vectores de la BDD y los de la consulta.
En este sprint se desarrollará la comparación de los vectores de la consulta contra los de la base de datos por medio de la distancia del coseno.

El vector de las consultas se obtiene vectorizando las opciones que nos muestra el etiquetador junto a un código CUI, esto nos ayudará en la búsqueda de las posibles posibilidades en la base de datos UMLS.
Historia de Usuario 004
Tabla 11
Historia de Usuario para comparación de vectores

Historia de Usuario

Número: H. U 04

Usuario: Programador

Nombre de Historia: Comprobación de los vectores de la BDD y los de la consulta

Prioridad: Alta

Riesgo de desarrollo: Bajo

Tiempo de estimación (días):
10

Interacción Asignada: 4

Desarrollador responsable: Juan Bahamonde

Descripción: Como programador debo crear un módulo que sea capaz de comprobar la similitud de los vectores tanto como los de la BDD y los de la consulta.

Valoración: La similitud de los vectores deben tener un error máximo de 20%.

Sprint Backlog
Tabla 12
Sprint Backlog 4

Sprint 04

Fecha
Inicio

03/07/2023

Fecha Fin

14/07/2023

Jornada

8

HU ID

Actividad

Horas

Inicio

Fin

Responsable

Estado

4

Función encargada de obtener los campos vectorizados de la BDD

16

03/07/2023

04/07/2023

Juan Bahamonde

Finalizado

Función encargada de obtener el valor vectorizado de la consulta

16

05/07/2023

06/07/2023

Juan Bahamonde

Finalizado

Función encargada de comprobar la similitud de los vectores

48

07/07/2023

14/07/2023

Juan
Bahamonde

Finalizado

Resultado del Sprint
Se completaron con éxito las actividades propuestas en el sprint 04, cumpliendo con el cronograma y sin inconvenientes por parte del desarrollador.

Como producto se obtuvo un módulo que es capaz de vectorizar los resultados del etiquetador y por medio de consultas SQL y comprobación de distancias del coseno, el sistema nos arroja el AUI con más similitud al texto ingresado.
Sprint 05: Correcciones de rendimiento.
El propósito central de este sprint consiste en mejorar la eficiencia en el tiempo de ejecución del módulo de similitud. Esto se logrará mediante la optimización de los procedimientos en la base de datos, así como la revisión y optimización de los procesos matemáticos específicos relacionados con dicho módulo.
Historia de Usuario 005
Tabla 13
Historia de Usuario para las correcciones

Historia de Usuario

Número: H. U 05

Usuario: Programador

Nombre de Historia: Correcciones de rendimiento

Prioridad: Alta

Riesgo de desarrollo: Bajo

Tiempo de estimación (días): 10

Interacción Asignada: 5

Desarrollador responsable: Juan Bahamonde - William Monge

Descripción: Como programador debo mejorar el rendimiento del sistema y así también corregir los errores del sistema.

Valoración: El sistema debe tener un aumento de rendimiento con respecto a la versión anterior.

Sprint Backlog

Tabla 14
Sprint Backlog 5

Sprint
05

Fecha Inicio

17/07/2023

Fecha Fin

28/07/2023

Jornada

8	HU ID	Actividad	Horas	Inicio	Fin	Responsable	Estado
	005	Realización de pruebas	32	17/07/2023	20/07/2023	William Monge	Finalizado
		Informe de las pruebas con los resultados	16	21/07/2023	24/07/2023	William Monge	Finalizado
		Corrección de las funciones con problemas de rendimiento	32	25/07/2023	28/07/2023	William Monge	Finalizado

Resultado del Sprint

Tal como se refleja en la Tabla 14, el desarrollador llevó a cabo las actividades planificadas dentro de los plazos estipulados y sin contratiempos, lo que aseguró el cumplimiento del cronograma establecido. Como resultado directo de estas acciones, el sistema experimentó una mejora en sus tiempos de ejecución, lo cual condujo a un aumento en su eficacia operativa.

Diseño del sistema

Modelo del sistema basado en el C4

Dentro del diseño para el sistema se ha optado por la utilización de un enfoque C4 que modela las dimensiones estáticas de un sistema de software a través de contenedores (como aplicaciones, almacenes de datos, microservicios, etc.), componentes y el código subyacente que los compone (Vivanco, 2019).

Aunque el modelo C4 originalmente consta de cuatro niveles, en el diseño de este sistema se ha decidido emplear tres niveles, los cuales proporcionan una descripción detallada de la estructura del sistema actual.

Nivel 1: Diagrama de contexto del sistema

Figura 7

Nivel 1 del Modelo C4

En la Figura 7, se presenta un esquema general del funcionamiento del sistema. En este proceso, el usuario introduce una consulta. Esta consulta es sometida a través del sistema

el cual con la conexión con la base de datos se obtiene un Identificador Único de Átomo (AUI) que corresponde a la similitud vectorial más cercana a la consulta ingresada.

Nivel 2: Diagrama de Contenedores

Figura 8

Nivel 2 del Modelo C4

En la Figura 8 se realiza una exploración más amplia del sistema, proporcionando una comprensión detallada de sus componentes. Entre los componentes que se pueden observar se encuentran:

- Traductor: es el encargado de traducir el texto ingresado al idioma inglés por medio de librerías de Python.
- Alineamiento: el trabajo de este es alinear el texto ingresado tanto en español como en inglés.
- Etiquetado: Con los datos obtenidos procede a etiquetar las posibles frases relacionadas a los conceptos médicos establecidos en la base de datos UMLS

- Similitud: genera los vectores de los datos obtenidos por el etiquetador y realiza un proceso de búsqueda en la base de datos para luego comprobar los vectores por medio de la distancia del coseno y así entregar un AUI con mayor similitud.
- Base de datos: la base de datos contiene todos los conceptos médicos normalizados juntos con sus respectivos vectores.

Nivel 3: Diagrama de Componentes

Figura 9

Nivel 3 del Modelo C4

En la Figura 9, se presenta una ampliación del componente de similitud que detalla un nivel adicional de funciones internas. En esta representación, es posible observar cómo el etiquetador transmite sus datos a dos funciones distintas: una función de vectorización y otra función de búsqueda que se conecta con la base de datos. Posteriormente, los resultados obtenidos de estos dos procesos son dirigidos a una función que evalúa la distancia vectorial

mediante el cálculo del coseno. Esta última función proporciona los Identificadores Únicos de Átomo (AUI) más similares al texto ingresado.

Arquitectura del sistema

El sistema se diseñó siguiendo una arquitectura en capas, como se puede apreciar en la Figura 10. Esta arquitectura está compuesta por tres niveles diferenciados: la capa de presentación, la capa de negocios y la capa de datos. Cada una de estas capas desempeña un rol fundamental en el funcionamiento integral y coordinado del sistema.

Figura 10

Arquitectura en Capas

Nota: La capa de presentación está formada por dos interfaces claramente definidas. La primera interfaz se encarga de la entrada de datos, permitiendo al usuario ingresar la información relevante. Por otro lado, la segunda interfaz tiene como función mostrar los resultados generados por la capa de negocios. La capa de negocios constituye el núcleo del sistema y está compuesta por diversos módulos interconectados. Estos módulos operan en una

única línea de producción, colaborando estrechamente con la capa de datos para generar los resultados requeridos. La capa de datos, por su parte, alberga la base de datos denominada UMLS. Esta base de datos es un depósito de información normalizada relacionada con conceptos médicos. Los datos contenidos son utilizados como recurso esencial por la capa de negocios para llevar a cabo sus operaciones.

Capa de presentación

En esta capa, la interfaz será visible para el usuario. En este espacio, se facilitará la inserción de texto, y a cambio se obtendrán las respuestas generadas por cada uno de los módulos. La construcción de esta interfaz se realizará mediante el uso del lenguaje de programación Python, haciendo uso del framework Flask.

Figura 11

Interfaz gráfica

Nota: En la Figura 11, se puede apreciar cómo el usuario tiene la posibilidad de ingresar texto. Al presionar el botón "Enviar", la capa de negocios toma el control de los procesos lógicos que se llevan a cabo en segundo plano y que no son perceptibles para el usuario.

Figura 12

Interfaz de resultados

Nota: En la Figura 12, se presentan los datos proporcionados al usuario en cuadros delimitados. En el primero de ellos, se exhibe el texto que el usuario ha ingresado. A continuación, en otro recuadro, se visualiza la traducción del texto, ofreciendo el resultado del proceso de etiquetado. Posteriormente, se presenta el resultado obtenido del etiquetador en un tercer recuadro. Finalmente, en el último recuadro, se muestra el desenlace de la búsqueda del AUI.

Capa de negocios

En esta capa, se llevará a cabo el desarrollo utilizando el lenguaje de programación Python, y se incorporarán las librerías necesarias para la conexión con la base de datos y la vectorización. En este nivel, se alojarán los módulos de traducción, alineamiento, etiquetado y similitud, los cuales serán responsables de abordar la lógica fundamental del sistema. Esta estructura se ilustra en la Figura 13.

Figura 13

Módulos

Enfocándonos en el módulo de similitud, este comienza su proceso estableciendo una conexión con la base de datos a través de la librería psycopg2. Para lograr esta conexión, se deben proporcionar el nombre de usuario, la contraseña, el host, el puerto y el nombre de la base de datos, tal como se representa en la Figura 14.

Figura 14

Conexión a la base de datos UMLS

Una vez que la conexión ha sido establecida, se da inicio a un primer ciclo iterativo, como se ilustra en la Figura 15. A lo largo de este ciclo, cada dato entregado por el etiquetador es sometido a procesamiento. En este punto, el dato es transformado para adquirir una estructura definida, preparándolo para ser usado en la siguiente fase del proceso, como se muestra en la Figura 16.

Figura 15

Primer ciclo iterativo

Figura 16

Proceso

Seguidamente, se ingresa a un segundo ciclo iterativo, como se ilustra en la Figura 17. Durante este ciclo, se recorren los resultados generados para llevar a cabo su vectorización. Posteriormente, los datos se formatean para lograr un manejo estandarizado que sea compatible con la base de datos, como se presenta en la Figura 18. Por último, el vector resultante se incorpora en una consulta SQL.

Figura 17

Segundo Ciclo iterativo

Figura 18
Proceso de vectorización

En el tercer ciclo iterativo (Figura 19), se toman los resultados generados por la consulta SQL y se inicia un proceso iterativo para calcular el coseno de los vectores correspondientes. Este cálculo se realiza mediante el uso de una herramienta proporcionada por pgvector (Figura 20).
Figura 19
Tercer ciclo iterativo

Figura 20
Cálculo del coseno

Luego, se procede a implementar un método de descarte en el cual se evalúan todos los valores de coseno obtenidos. La elección se realiza considerando el AUI asociado al valor del coseno más cercano a cero, tal como se ilustra en la Figura 21.
Figura 21
Método de descarte

Capa de datos
En esta capa, la base de datos UMLS desempeña un papel fundamental y se configura como el componente central. En ella se encuentran almacenados tanto los datos normalizados como los vectores esenciales para cada uno de los términos utilizados en el funcionamiento completo del sistema, como se presenta en la Figura 22.
Figura 22
Base de datos UMLS

Capítulo VI
Validación del Sistema
La validación del sistema se apoya en un corpus que se encuentra almacenado en el repositorio de GitHub. Este corpus fue creado de manera manual, lo que significa que cada dato fue revisado y etiquetado por un individuo con conocimientos en el campo médico. Es importante destacar que este corpus desempeñará un papel fundamental en la validación de la capacidad del sistema para obtener los Identificadores Únicos de Átomo (AUI) de manera precisa y adecuada.
Definición y aplicación de métricas de evaluación
Dentro del desarrollo de software, se emplean diversas métricas para evaluar el sistema. En este caso, se han seleccionado las siguientes métricas específicas:
• Fiabilidad: Esta métrica se refiere a la capacidad del sistema para llevar a cabo sus funciones con un nivel de precisión determinado. Su cálculo se basa en la correcta obtención de los Identificadores Únicos de Átomo (AUI).
• Obtención del AUI: Esta métrica consiste en comparar el AUI obtenido por el sistema con el AUI proporcionado por el corpus. Si el AUI coincide con el del corpus, se otorga un punto. En caso contrario, se asigna un valor de 0 puntos.
• Eficiencia: Esta métrica se enfoca en medir el tiempo de respuesta de cada módulo del sistema.

Dentro del sistema, se encuentran los módulos de vectorización, búsqueda y comprobación. Cada uno de estos módulos debe operar dentro de un tiempo predeterminado, que se especifica a continuación:
▪ 0s – 30s: obtendrá 1 punto.
▪ 30s – 90s: obtendrá 0,5 puntos.
▪ +90s: obtendrá 0 puntos.
La elección de estas métricas se fundamenta en la premisa de que el sistema debe ofrecer información precisa en el menor tiempo posible, siendo estos aspectos cruciales para su desempeño óptimo. En esta evaluación, no se han tenido en cuenta elementos como la usabilidad o la seguridad, ya que el enfoque principal radica en la eficacia de la interacción entre el usuario y el sistema. Dado el alcance actual del sistema, estos aspectos no son relevantes para la evaluación.
La asignación de los valores numéricos 1, 0.5 y 0 a las métricas se estableció con el propósito de simplificar el cálculo de los puntajes. En cuanto a los intervalos de tiempo en la sección de eficiencia, estos fueron determinados después de observar el desempeño del sistema en comparación con sistemas similares.
Luego, se procede a llevar a cabo las pruebas del sistema en conjunto con el corpus, tal

como se presenta en la Tabla 15.

Tabla 15

Evaluación de las métricas

Eficiencia

Fiabilidad

Vectorizar

Búsqueda

Comprobación

AUI 1

1

1

0

0,5

AUI 2

1

1

0,5

1

AUI 3

1

0,5

0

0,5

AUI 4

1

1

0,5

1

AUI 5

1

1

0

0,5

AUI 6

1

0,5

0,5

1

AUI 7

1

1

0

0,5

AUI 8

1

1

0

1

Total

1

0,875

0,1875

0,8125

Total

0,625

Análisis de resultados

Tabla 16

Totales de la evaluación de las métricas

Eficiencia

Fiabilidad

Vectorizar

Búsqueda

Comprobación

Total

1

0,8

0,171875

0,828125

Total

0,6

Como se puede apreciar en la Tabla 16, en la sección de fiabilidad, el sistema ha obtenido una puntuación de 1. Esto indica que el sistema logra cumplir su objetivo de brindar un AUI preciso correspondiente a la lexicalización buscada.
En relación a la métrica de eficiencia, hemos obtenido un puntaje total de 0,625. Esto se atribuye principalmente al módulo de búsqueda, el cual presenta tiempos de ejecución excesivos. Esta situación puede deberse a una indexación inadecuada en la base de datos o a la realización de consultas demasiado complejas.
Figura 23

Análisis de la Interfaz de muestra de resultados

La Figura 23 presenta los datos de una manera clara y comprensible, definiendo adecuadamente cada uno de los aspectos requeridos por el usuario, como los resultados de los diferentes módulos utilizados en el sistema.
Es esencial destacar que la interfaz web juega un papel crucial en la facilitación de la interacción con el sistema. Para su construcción, se emplearon lenguajes de programación web como HTML, CSS y el framework Flask, este último basado en Python y destinado al desarrollo web.
Se aplicaron técnicas y prácticas óptimas para el manejo de archivos y componentes dentro del sistema, lo que permite su escalabilidad en el futuro, ya sea mediante la incorporación de nuevos módulos o la optimización de los ya existentes.

Capítulo V
Conclusiones y Recomendaciones
Conclusiones
- Se ha logrado el objetivo de desarrollar un sistema informático para la normalización de entidades biomédicas a través de búsquedas semánticas multilenguaje, utilizando como base de datos médica el UMLS.
- La construcción del marco teórico ha posibilitado una revisión detallada de conceptos sobre búsquedas semánticas y la aplicación de modelos Transformer, así como la identificación de herramientas esenciales para el desarrollo del sistema.
- La implementación de una metodología ágil ha desempeñado un papel clave al fomentar un proceso de desarrollo colaborativo y eficiente, dentro de un marco temporal definido, lo que ha llevado al logro exitoso del sistema.
- El sistema presenta tiempos de ejecución extensos en el módulo de búsqueda que trabaja por medio de consultas enviadas a la base de datos UMLS.
- El diseño de la interfaz de usuario es sencillo y amigable, lo que facilita su uso por parte de los usuarios. Sin embargo, se reconocen áreas de mejora que se detallan en la sección de recomendaciones.
- El enfoque de modelado C4 ha contribuido a una comprensión y visualización claras del funcionamiento del sistema, resultando en una explicación accesible

para personas externas al proyecto, permitiendo una comprensión gradual desde los niveles más altos hasta los componentes individuales.

Recomendaciones

• El sistema podría experimentar lentitud en los procesos de búsqueda debido a una posible configuración inadecuada de la base de datos. Por lo tanto, se recomienda implementar índices en las tablas relevantes para mejorar la eficiencia de las consultas. Estas tablas incluyen MRCONSO y MRSTY.
• Aunque el método de vectorización utilizado para el texto es efectivo, existen otros modelos que podrían optimizar este proceso aún más. Considerar la exploración de alternativas podría ser beneficioso en términos de mejora del rendimiento.
• La elección de Flask para la interfaz gráfica ha permitido crear una interfaz visual rápida y sencilla. No obstante, es importante tener en cuenta que, dado que Flask trabaja en conjunto con HTML, CSS y JS, la implementación a nivel de código puede volverse algo compleja. Explorar otras opciones para el diseño de la interfaz podría simplificar esta complejidad.
• Para la vectorización de la base de datos se debería realizar una investigación de nuevos modelos basados en la arquitectura Transformers. Sería valioso considerar esta alternativa para optimizar el proceso de vectorización.
• Se sugiere la revisión y actualización del sistema conforme a la aparición de nuevas tecnologías.

Bibliografía

Alexander, M., Gertrudis, L., Juan, P., & Miguel, Á. S. (2022). Historias de Usuario. Scrum Manager®.
Ashish, V., Noam, S. P., Jakob, U., Llion, J., Aidan, N. G., & Łukasz, K. P. (2017). Attention Is All You Need. doi:doi.org/10.48550/arXiv.1706.03762
Avila, D. (s.f.). Búsqueda semántica y Embedding con Cohere. Obtenido de LatinXinAI: https://medium.com/latinxinai/b%C3%BAsqueda-sem%C3%A1ntica-y-embedding-con-cohere-4a14e0209cd9
Bodenreider, O. (2004). The Unified Medical Language System (UMLS):integrating biomedical terminology. Nucleic Acids Research, 32. doi:10.1093/nar/gkh061
Cong, S., Zhihao, Y., Lei, W., Yin, Z., Hongfei, L., & & Jian, W. (2021). Deep learning with language models improves named entity recognition for PharmaCoNER. BMC Bioinformatics, 22. doi:doi.org/10.1186/s12859-021-04260-y
David, E., Miguel, E. R., & Srinivasan, P. (1998). Cross-language information retrieval with the UMLS metathesaurus. Association for Computing Machinery. doi:doi.org/10.1145/290941.290959
Ghajari, A., Fresno, V., & Amigó, E. (2022). Plataforma de exploración de la Composición Semántica a partir de Modelos de Lenguaje pre-entrenados y embeddings estáticos.
Greg, R. (2023). Storing OpenAI embeddings in Postgres with pgvector. Obtenido de supabase: https://supabase.com/blog/openai-embeddings-postgres-vector
Habibi, M., Leon, W., Mariana, N., David, L. W., & Ulf, L. (2017). Deep learning with word embeddings improves biomedical named entity recognition. Bioinformatics, 33, i37-i48. doi:doi.org/10.1093/bioinformatics/btx228

Kuz, A., Mariana, F., & Roxana, S. G. (2018). Comprendiendo la Aplicabilidad de Scrum en el Aula: Herramientas y Ejemplos. Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología, 21, 62-70. doi:10.24215/18509959.21.e07
Lample, G., & Conneau, A. (2019). Cross-lingual language model pretraining. arXiv preprint arXiv:1901.07291.
Liu, Q., Kusner, M. J., & Blunsom, P. (2020). A survey on contextual embeddings. arXiv preprint arXiv:2003.07278.
McCray, A. T., Burgun, A., Bodenreider, O., & Agoncillo, A. V. (2001). Modeling biomedical information for interoperability and decision support. Proceedings of the AMIA Symposium.
Nader, K. R., & Frank, T. (2019). Los Fundamentos de Agile Scrum. (V. Haren, Ed.)
National Library of Medicine. (2016). Obtenido de The Unified Medical Language System (UMLS): https://www.nlm.nih.gov/research/umls/new_users/online_learning/OVR_001.html
Nils, R., & Iryna, G. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. EMNLP. doi:doi.org/10.48550/arXiv.1908.10084
Pallo, T., Brandon, E., Salazar, R., & Adrian, A. (2023). Sistema web para el reconocimiento y la normalización de entidades biomédicas mediante técnicas de cross-lingual. Obtenido

de Repositorio Institucional de la Universidad de las Fuerzas Armadas ESPE: http://repositorio.espe.edu.ec/handle/21000/35734
R. Devika, S. Vairavasundaram, C. S. J. Mahenthar, V. Varadarajan and K. Kotecha, "A Deep Learning Model Based on BERT and Sentence Transformer for Semantic Keyphrase Extraction on Big Social Data," in IEEE Access, vol. 9, pp. 165252-165261, 2021, doi: 10.1109/ACCESS.2021.3133651.

Rick, M. (2022). ¿Qué es un Modelo Transformer? Obtenido de NIVIDIA: https://la.blogs.nvidia.com/2022/04/19/que-es-un-modelo-transformer/
Rodríguez, C. &. (2015). ¿Por qué implementar Scrum? Revista ONTARE, 3(1), 125-144. doi:https://doi.org/10.21158/23823399.v3.n1.2015.1253
Ros Pagan, D. (2021). Búsqueda semántica de perfiles en redes sociales (Doctoral dissertation, Universitat Politècnica de València).
Ruder, S., Vulić, I., & Søgaard, A. (2019). A Survey of Cross-lingual Word Embedding Models. Journal of Artificial Intelligence Research, 65, 569-631. doi:https://doi.org/10.1613/jair.1.11640
Sachdeva, S. (2016). Scrum Methodology. International Journal Of Engineering And Computer Science, 5, 16792-16799.
Salyuk Kulinich, M. (2022). Búsqueda de respuestas utilizando redes neuronales (Doctoral dissertation, Universitat Politècnica de València).
Soldaini, L., & Moschitti, A. (2020). The cascade transformer: an application for efficient answer sentence selection. arXiv preprint arXiv:2005.02534.
Tomas, M., Kai, C., Greg, C., & Jeffrey, D. (2013). Efficient Estimation of Word Representations in Vector Space. doi:doi.org/10.48550/arXiv.1301.3781
Valpadasu, H., Sravanthi, T., Naresh, K., Ch, P., C., B. R., & K, M. (2020). Scrum: An Effective Software Development Agile Tool. IOP Conf. Series: Materials Science and Engineering. doi:10.1088/1757-899X/981/2/022060
Vidal-Silva, Cristian L., Sánchez-Ortiz, Aurora, Serrano, Jorge, & Rubio, José M.. (2021). Experiencia académica en desarrollo rápido de sistemas de información web con Python y Django. Formación universitaria, 14(5), 85-94. https://dx.doi.org/10.4067/S0718-50062021000500085

Vivanco, J. (2019). El modelo C4 de documentación para la Arquitectura de Software. Obtenido de Medium: https://medium.com/@javiervivanco/el-modelo-c4-de-documentaci%C3%B3n-para-la-arquitectura-de-software-424704528390
Wang, W., Chen, G., Wang, H., Han, Y., & Chen, Y. (2023). Multilingual Sentence Transformer as A Multilingual Word Aligner. arXiv preprint arXiv:2301.12140.

Anexos

Firmado electrónicamente por:
ALVARO DANILO
UYAGUARI UYAGUARI

ING. Uyaguari Uyaguari, Álvaro Danilo, Msc
C.C: 0103411112