

Universidad de Oriente

Sede “Julio Antonio Mella”

Facultad de Ingeniería en Telecomunicaciones, Informática y Biomédica

Trabajo de Diploma

En opción al título de Ingeniera en Informática

**Título:** “Herramienta de Procesamiento de Lenguaje Natural para el reconocimiento de entidades”.

**Autor:** Luis Andrés Licea Berenguer.

**Tutor:** Dr. C. Dionis López Ramos.

Santiago de Cuba,

“Año de la Revolución”

Resumen

En la actualidad, el reconocimiento de entidades nombradas desempeña un papel fundamental en diversos contextos, desafiando la capacidad de las aplicaciones para identificar y comprender información clave en grandes volúmenes de datos no estructurados. La habilidad de discernir entidades, como personas, lugares, organizaciones y otros elementos específicos en el texto, es un componente crítico en la extracción de conocimiento y toma de decisiones.

Este estudio se enfoca en la creación de un modelo de reconocimiento de entidades mediante la implementación de la biblioteca SpaCy y el lenguaje de programación Python. El modelo no solo tiene la capacidad de identificar entidades, sino que también incorpora una función de reentrenamiento supervisado para abordar la aparición de nuevas y desconocidas entidades.

Para lograr este propósito, se aplican técnicas avanzadas de procesamiento de lenguaje natural y aprendizaje automático. Se utiliza un conjunto de datos etiquetados para entrenar el modelo, cuyo desempeño se evalúa mediante métricas de precisión, exhaustividad y F1-score. Los resultados obtenidos demuestran que el modelo es altamente preciso y exhaustivo en su capacidad para reconocer entidades.

La característica distintiva de este modelo radica en su funcionalidad de reentrenamiento supervisado, lo que permite una adaptación dinámica a nuevas entidades, garantizando un rendimiento óptimo en tiempo real. En resumen, este trabajo aporta significativamente al campo del procesamiento de lenguaje natural al proporcionar un modelo preciso y adaptable que satisface la creciente importancia del reconocimiento de entidades en el entorno contemporáneo.

**Palabras clave:** Reconocimiento de entidades, SpaCy, Python, reentrenamiento supervisado

*ABSTRACT*

In the present day, named entity recognition plays a fundamental role across various contexts, challenging applications to identify and comprehend critical information within extensive volumes of unstructured data. The ability to discern entities, such as people, places, organizations, and other specific elements within text, is a critical component in knowledge extraction and decision-making.

This study focuses on the development of a named entity recognition model through the implementation of the SpaCy library and the Python programming language. The model not only possesses the capability to identify entities but also incorporates a supervised retraining function to address the emergence of new and unfamiliar entities.

To achieve this objective, advanced techniques in natural language processing and machine learning are employed. A labeled dataset is used to train the model, and its performance is evaluated using precision, recall, and F1-score metrics. The results demonstrate that the model exhibits a high degree of accuracy and comprehensiveness in its entity recognition capabilities.

The distinctive feature of this model lies in its supervised retraining functionality, allowing dynamic adaptation to novel entities, ensuring optimal real-time performance. In summary, this work significantly contributes to the field of natural language processing by providing a precise and adaptable model that meets the growing importance of named entity recognition in the contemporary environment.

**Keywords*:***Named Entity Recognition, SpaCy, Python, Supervised Retraining

Tabla de contenido

[**INTRODUCCIÓN** 1](#_Toc159488804)

[**CAPITULO 1 .** **MARCO TEÓRICO** 5](#_Toc159488805)

[1.1 Breve historia de los modelos de procesamiento del lenguaje natural para la detección de entidades 5](#_Toc159488806)

[1.2 Estado del Arte de los marcos de procesamiento del lenguaje natural para la detección de entidades 6](#_Toc159488807)

[1.2.1 Google NLP 6](#_Toc159488808)

[1.2.2 ChatGPT 7](#_Toc159488809)

[1.2.3 IBM Watson Discovery 7](#_Toc159488810)

[1.2.4 Amazon Comprehend 8](#_Toc159488811)

[1.2.5 Dandelion 9](#_Toc159488812)

[1.3 Herramientas, Lenguaje de programación y Tecnologías 9](#_Toc159488813)

[1.3.1 Python 9](#_Toc159488814)

[1.3.2 Visual Studio Code 10](#_Toc159488815)

[1.3.3 Elasticsearch 11](#_Toc159488816)

[1.3.4 Git 12](#_Toc159488817)

[1.3.5 Modelos para el reconocimiento de entidades nombradas 12](#_Toc159488818)

[1.3.6 Modelos generadores de oraciones 14](#_Toc159488819)

[1.3.7 React 15](#_Toc159488820)

[1.3.8 FastAPI 15](#_Toc159488821)

[1.3.9 XP 16](#_Toc159488822)

[**CAPITULO 2 .** **ORGANIZACIÓN Y DISEÑO** 19](#_Toc159488823)

[2.1 Propuesta del sistema 19](#_Toc159488824)

[2.2 Arquitectura del sistema creado 19](#_Toc159488825)

[2.3 Requisitos Funcionales 20](#_Toc159488826)

[2.3.1 Frontend 20](#_Toc159488827)

[2.3.2 Backend 20](#_Toc159488828)

[2.4 Requisitos No Funcionales 21](#_Toc159488829)

[2.4.1 Software 21](#_Toc159488830)

[2.4.2 Hardware 21](#_Toc159488831)

[2.5 Modelo Entidad Relación 21](#_Toc159488832)

[2.6 Historias de Usuario 21](#_Toc159488833)

[2.7 Usuarios del Sistema 23](#_Toc159488834)

[2.8 Diagrama de Secuencia 23](#_Toc159488835)

[2.9 Diagrama de Actividades 24](#_Toc159488836)

[2.10 Diseño de la Interfaz 25](#_Toc159488837)

[**Primer epígrafe del tercer capítulo** 25](#_Toc159488838)

[**Primera división (o sub-epígrafe) dentro del primer epígrafe** 27](#_Toc159488839)

[**Ponga aquí el segundo epígrafe** 27](#_Toc159488840)

[**Primera división (o sub-epígrafe) dentro de este epígrafe** 27](#_Toc159488841)

[**1.** **CONCLUSIONES Y RECOMENDACIONES** 28](#_Toc159488842)

[**1.** **REFERENCIAS BIBLIOGRÁFICAS** 29](#_Toc159488843)

[**2.** **GLOSARIO DE TÉRMINOS** 32](#_Toc159488844)

[**4.** **ANEXOS** 33](#_Toc159488845)

**INTRODUCCIÓN**

El término “entidad nombrada” fue usado por primera vez en la Message Understanding Conference(MUC-6), organizada por Grishman y Sundheim (1996), y se refirió a la tarea de identificar nombres de organizaciones, personas y ubicaciones geográficas en textos, así como expresiones monetarias de tiempo y porcentajes. Desde la conferencia MUC-6, el interés en NER fue incrementándose hasta hacerse presente en varios eventos científicos.[1]

Ya en la actualidad elReconocimiento de Entidades Nombradases una técnica basada en el procesamiento de lenguaje natural (NLP) que se utiliza para extraer, identificar y clasificar información en documentos de texto. Detecta entidades (es decir, partes de la oración) y las clasifica en una categoría predeterminada, como nombre o código de país.[2]

Las categorías NER pueden ser genéricas e indicar, por ejemplo, palabras que significan una organización, una persona o una época. Sin embargo, también pueden personalizarse en función de un caso de uso específico. Por poner un ejemplo, el modelo NER puede crearse para reconocer categorías como “nombre del paciente” y “fecha de nacimiento” en documentos médicos o “nombre del comerciante” y “fecha de compra” en facturas. Las posibilidades son infinitas.[2]

La primera fase del reconocimiento de entidades generalmente se reduce a un problema de segmentación: los nombres son una secuencia contigua de tokens, sin solapamiento ni anidamiento, de modo que Banco de la Nación Cuba es un nombre único, a pesar del hecho de que dentro de este nombre aparezca la subcadena Cuba que es a su vez el nombre de un país. La segunda fase se trata de asignar una categoría, de entre un conjunto predeterminado, a cada una de las entidades previamente reconocidas en la fase uno.[3]

Desde la aparición de distintas tecnologías, como la Internet, la cantidad de información disponible en distintos formatos y fuentes ha crecido a pasos agigantados. El tamaño de las colecciones almacenadas dificulta su manejo y organización, así como la posibilidad de encontrar información específica, ya sea, en un solo documento o en un conjunto de documentos. Además, la información puede no tener una estructura definida, como es el caso de la información textual, donde se encuentran principalmente secuencias de palabras. [4].

Dadas estas dificultades el desarrollo de herramientas que permitan administrar y permitir la búsqueda de ciertos elementos en un documento se ha vuelto necesario y de suma importancia. Por lo que herramientas como Elasticsearch un motor de búsqueda y análisis de datos altamente escalable, Spacy un modelo de procesamiento de lenguaje natural y Llama2 un modelo generador de oraciones más rápido que los modelos de GPT-3.5 y GPT-4, aprovechando al máximo el uso de estas se construiría una poderosa y robusta herramienta de procesamiento de lenguaje natural para la detección de entidades de Elasticsearch, mejorando aún más su búsqueda y siendo capaz de reentrenarse de manera supervisada cuando surja una nueva entidad.

**Problema de investigación:**

El avance en el reconocimiento de entidades se enfrenta a desafíos significativos debido a la complejidad inherente en el entrenamiento de modelos de lenguaje natural por parte de los humanos. Además, la mayoría de los modelos actuales carecen de la capacidad de reentrenamiento automatizado, lo que complica aún más el proceso y resulta en costos elevados. Estas limitaciones hacen que el reconocimiento de entidades sea tedioso y menos efectivo en comparación con las capacidades humanas. Este problema es especialmente relevante para instituciones nacionales como CERPAMID-DATYS, que manejan volúmenes masivos de datos diariamente. Sin un sistema de reconocimiento de entidades eficiente, el procesamiento de estos datos se vuelve altamente ineficiente. Además, la necesidad de identificar nuevas entidades a diario añade otra capa de complejidad a este proceso.

**Objeto de Estudio:**

Herramientas de Procesamiento del Lenguaje Natural.

**Campo de Estudio:**

Reconocimiento de Entidades Nombradas.

**Objetivo General:**

Desarrollar un programa que reciba los datos de Elasticsearch y conjunto con el modelo de procesamiento de lenguaje natural SPACY hacer el reconocimiento de las entidades nombradas, en caso de que el modelo no reconozca una entidad, reentrenarlo de manera supervisada, para que reconozca dicha entidad.

**Objetivo Específicos:**

* **Extracción de Datos de Elasticsearch:** El programa se encargará de establecer la conexión con Elasticsearch y extraer los datos almacenados en la base de datos, asegurando la integridad y calidad de la información recuperada.
* **Reconocimiento de Entidades (NER):** Se desarrollará una funcionalidad de NER utilizando el modelo de Spacy para la identificación y clasificación automática de entidades presentes en los datos extraídos. Esto incluye la detección de nombres de personas, organizaciones, ubicaciones, fechas, entre otros.
* **Aprendizaje Supervisado:** En caso de que el modelo Spacy no reconozca una entidad específica en los datos, se implementará un proceso de aprendizaje supervisado. El programa permitirá a los usuarios etiquetar manualmente las nuevas entidades, reentrenando así el modelo para reconocer estas entidades en el futuro.
* **Almacenamiento de Resultados en Elasticsearch:** Una vez identificadas las entidades, el programa guardará los resultados de manera estructurada en Elasticsearch, permitiendo un acceso eficiente y consultas posteriores.
* **Evaluación de Rendimiento:** Se realizará una evaluación del rendimiento del programa, incluyendo la precisión y la eficiencia del reconocimiento de entidades, así como la efectividad del proceso de aprendizaje supervisado.
* **Optimización Continua:** El objeto de investigación incluye la posibilidad de optimizar y mejorar el programa a medida que se recopilen más datos y se identifiquen nuevas entidades.

**Campo de Acción:**

Aplicación de modelos de procesamiento del lenguaje natural, como Spacy, para detectar entidades, proponiendo el reentrenamiento del modelo cuando no se reconocen correctamente las entidades. Además, se integra una base de datos en Elasticsearch para almacenar y analizar los resultados del reconocimiento de entidades.

**Hipótesis:**

La implementación conjunta de Spacy, Elasticsearch y Python en un sistema de procesamiento de lenguaje natural permitirá la identificación y clasificación eficiente de entidades en grandes volúmenes de datos, facilitando el estudio lingüístico. Esta hipótesis plantea que la combinación de estas tecnologías tendrá un impacto positivo en la capacidad de la empresa para gestionar y aprovechar datos masivos con fines de estudio y análisis lingüístico, y se buscará probar esta hipótesis a través de la implementación y evaluación de la solución propuesta.

**Métodos de investigación:**

* Método histórico-lógico: se aplicó al realizar el análisis de la existencia de otros sistemas que den solución al problema en cuestión.
* Método de análisis y síntesis: se aplicó al realizar el análisis de todo el proceso llevado a cabo en el proyecto y sintetizar las ideas que fueron surgiendo; extrayendo los elementos comunes al objeto de estudio.

**Interrogantes Científicas:**

* ¿Cómo se puede mejorar la precisión y exhaustividad del modelo de reconocimiento de entidades utilizando técnicas de procesamiento de lenguaje natural y aprendizaje automático?
* ¿Cómo se puede implementar una funcionalidad de reentrenamiento supervisado para adaptar el modelo a nuevas entidades en tiempo real?
* ¿Cuál es el impacto de utilizar la biblioteca Spacy y el lenguaje de programación Python en el desarrollo y desempeño del modelo de reconocimiento de entidades?

**Aportes de la Investigación:**

* Un sistema que reconozca las entidades de los textos de la base de datos de elasticsearch.
* Capacidad de reentrenamiento de forma supervisada por parte del usuario o la entidad en caso de surgir o no detectar una nueva entidad.
* Generación de datos de entrenamiento y testing de la entidad que no reconozca el modelo.
* Capacidad de retroceder en caso de el modelo perder conocimiento al realizar el reentrenamiento.

1. **MARCO TEÓRICO**

## Breve historia de los modelos de procesamiento del lenguaje natural para la detección de entidades

El término “entidad nombrada” fue usado por primera vez en la Message Understanding Conference (MUC-6), organizada por Grishman y Sundheim (1996), y se refirió a la tarea de identificar nombres de organizaciones, personas y ubicaciones geográficas en textos, así como expresiones monetarias de tiempo y porcentajes. Desde la conferencia MUC-6, el interés en NER fue incrementándose hasta hacerse presente en varios eventos científicos.

Con respecto a la definición de entidades nombradas, G. Petasis la expresa como: “Una entidad nombrada es un nombre propio, sirviendo como un nombre para algo o alguien”, dicha definición se justifica debido al porcentaje tan significativo de nombres propios que encontramos en un corpus. Nadeau y Sekine afirman que la palabra “nombrada” restringe la tarea a sólo aquellas entidades para las cuales uno o muchos designadores rígidos, representan el referente. A pesar de las múltiples definiciones de las entidades nombradas, investigadores han logrado un acuerdo común de los tipos de entidades nombradas a reconocer.[5]

**1990**:

Los primeros enfoques para el NER se basaban en reglas y patrones en el texto. Estos enfoques utilizaban conjuntos de reglas predefinidas para identificar y extraer entidades nombradas [6]

**2000**:

A medida que los sistemas de procesamiento de lenguaje natural (NLP) se volvieron más sofisticados, se desarrollaron enfoques basados en diccionarios. Estos enfoques utilizaban diccionarios de palabras y frases para identificar y extraer entidades nombradas [6]

**2010**:

Con la aparición del machine learning, los enfoques basados en diccionarios dieron paso a enfoques basados en machine learning. Estos enfoques utilizaban algoritmos de machine learning para aprender de los datos y extraer entidades nombradas [6]

**2018**:

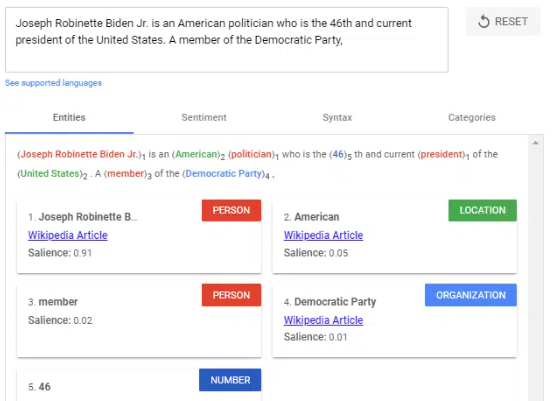
En los últimos años, los enfoques basados en deep learning han ganado popularidad. Estos enfoques utilizan redes neuronales profundas para aprender representaciones jerárquicas de las palabras y extraer entidades nombradas [6]

**2020**:

Los modelos de lenguaje preentrenados como BERT han revolucionado el NER al permitir que los modelos aprendan representaciones jerárquicas de las palabras a partir de grandes cantidades de texto. Estos modelos pueden ser ajustados para tareas específicas de NER, lo que los hace muy eficaces[6]

## Estado del Arte de los marcos de procesamiento del lenguaje natural para la detección de entidades

### Google NLP



Google Cloud NLP es una API de Inteligencia Artificial (IA) que se centra en el procesamiento del lenguaje natural (NLP). Esta API es parte de la suite de productos digitales de Google, que están en constante crecimiento y amplían las capacidades de la tecnología para las empresas[7]

La API de Google Cloud NLP se enfoca en la creación de contenido que se asemeje lo más posible a lo que un humano podría crear, proporcionando un análisis de texto a un nivel superior que fluye de manera más clara que lo que muchos programas de IA actuales son capaces de proporcionar. Esta API es la opción más compleja de Google, lo que significa que se dirige a empresas más grandes que su producto básico AutoML[7]

Entre las ventajas de Google Cloud NLP, destaca su reconocimiento de entidades nombradas multilingüe y su eficiencia en comparación con la contratación de humanos con habilidades de idioma para manejar proyectos similares. Esta API ofrece servicios en 10 idiomas diferentes y proporciona características como la resumación de texto, la generación de texto, la traducción, la detección de idioma, la tokenización, la lematización y el análisis de sentimientos

Sin embargo, Google Cloud NLP tiene algunas limitaciones. Aunque es una opción fuerte en las áreas en las que proporciona, la API actualmente es más limitada en los tipos de información que es capaz de manejar. Muchos usuarios han encontrado que esta API necesita una formación significativamente mayor para ser capaz de realizar menos tareas que muchas de las otras opciones en el mercado. Además, Google Cloud NLP es una opción más costosa que muchas de sus competidoras, lo que significa que puede ser más difícil de experimentar que otras opciones. Aunque esta API proporciona créditos gratuitos a nuevos clientes, el costo más alto de este programa puede no valer la pena una vez que su negocio utiliza estos créditos y necesita pasar a una suscripción continua.

### ChatGPT

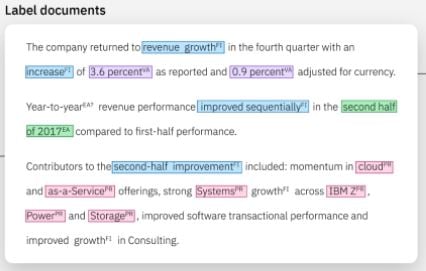


ChatGPT es, probablemente, el referente inmediato para millones de personas sobre lo que es un sistema de procesamiento natural de lenguaje. Esta tecnología está equipada con un potente motor de análisis de lenguaje conocido como GPT-4.

Esta herramienta ganó popularidad a principios del 2023 ya que es una de las pioneras en poner al servicio de las personas una herramienta capaz de responder a demandas concretas con un alto nivel de éxito. Además, esta plataforma lanzó su propia API (interfaz de programación de aplicaciones, por sus siglas en inglés) que puede ser integrada en programas para la creación de chatbots mucho más potentes, útiles y precisos con las respuestas que ofrecen a sus usuarios.

Hoy en día, esta herramienta está detrás de miles de programas y aplicaciones, como son tiendas online, buscadores, aplicaciones de aprendizaje y hasta plataformas de redes sociales. Además, es un gran apoyo si quieres investigar sobre un tema altamente especializado, si deseas reescribir textos o si necesitas ideas creativas para redactar tus propios contenidos. [8]

### IBM Watson Discovery

****

Watson es la plataforma de inteligencia artificial creada por IBM que tiene como objetivo acercar el uso de datos a los negocios de hoy. De entre sus herramientas, Discovery representa el mayor avance en aplicación del procesamiento de lenguaje natural.

Esta herramienta es, esencialmente, un procesador de documentos que evalúa, interpreta y extrae información de archivos de texto con el fin de hacer más simple la consulta y lectura de materiales escritos. Mediante la creación de etiquetas, el sistema aprende a detectar información de relevancia, así como a distinguir entre tablas, títulos y anotaciones.

Gracias a su potente motor de IA, esta tecnología es incluso capaz de traducir imágenes en texto mediante el reconocimiento de caracteres, así como crear mapas o constelaciones de conceptos para que no dejes fuera ninguna conexión entre elementos de tus documentaciones. [8]

### Amazon Comprehend

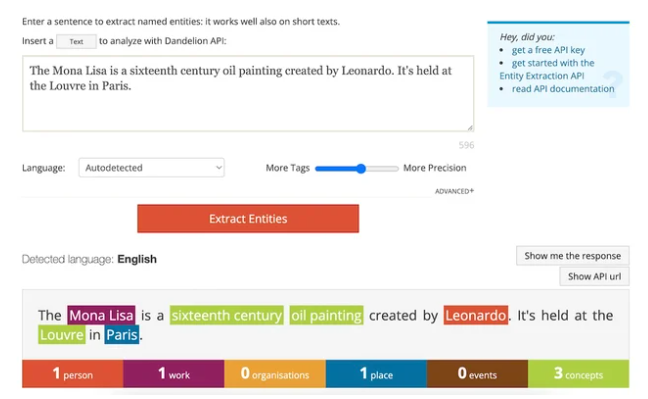
****

Otra excelente herramienta para el procesamiento de texto en documentos es Amazon Comprehend, que forma parte del universo de Amazon Web Services. Este sistema es capaz de extraer información tanto de documentos como de tickets de atención al cliente, reseñas, correos electrónicos y redes sociales.

Por estas funciones es que Comprehend resulta tan útil para tiendas en línea o empresas con gran actividad en medios de comunicación masiva. Con algunos simples pasos, puedes saber qué opinión tienen los clientes de ti, qué piensan de tus productos y hasta el tono de tus interacciones con ellos a través de tus canales de atención.

También, hay empresas que usan este sistema para la administración de enfoques legales y procesamiento de documentos financieros, ya que pueden detectar detalles que hasta tu equipo legal y de ventas puede pasar por alto en los términos y condiciones de alguna operación.[8]

### Dandelion



Dandelion es una API que se centra en el análisis semántico y de texto para ayudar a los usuarios a entender si los documentos que se están visualizando son principalmente positivos, negativos o neutrales. Esto permite a los usuarios aprender más sobre las opiniones generales sobre ciertos temas, lo que puede ayudarles a determinar qué ángulo sería el mejor para trabajar con la información que aprenden.[7]

Esta API se dirige principalmente a las empresas que están interesadas en investigar lo que su público objetivo está diciendo sobre varios temas que están relacionados con su industria, así como cómo sus competidores están manejándolos. Aunque no se adentra tanto en lo que los usuarios pueden hacer con esta información, Dandelion proporciona una visión general sólida de qué información relevante está actualmente disponible[7]

## Herramientas, Lenguaje de programación y Tecnologías

Las herramientas son objetos elaborados a fin de facilitar la realización de una tarea. Se diseñan y fabrican para cumplir uno o más propósitos específicos, por lo que son generalmente artefactos con una función técnica. Un lenguaje de programación es un lenguaje formal que especifica una serie de instrucciones para que una computadora produzca diversas clases de datos. Los lenguajes de programación pueden usarse para crear programas que pongan en práctica algoritmos específicos los cuales controlan el comportamiento físico y lógico de una computadora. La tecnología es la ciencia aplicada a la resolución de problemas concretos. Constituye un conjunto de conocimientos científicamente ordenados, que permiten diseñar y crear bienes o servicios que facilitan la adaptación al medio ambiente y la satisfacción de las necesidades humana.

### Python

Python en su versión 3.10.8 es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML). Los desarrolladores utilizan Python porque es eficiente y fácil de aprender, además de que se puede ejecutar en muchas plataformas diferentes. El software Python se puede descargar gratis, se integra bien a todos los tipos de sistemas y aumenta la velocidad del desarrollo [9].

Se utilizó Python porque ofrece los siguientes beneficios:

* Los desarrolladores pueden leer y comprender fácilmente los programas de Python debido a su sintaxis básica similar a la del inglés.
* Python permite que los desarrolladores sean más productivos, ya que pueden escribir un programa de Python con menos líneas de código en comparación con muchos otros lenguajes.
* Python cuenta con una gran biblioteca estándar que contiene códigos reutilizables para casi cualquier tarea. De esta manera, los desarrolladores no tienen que escribir el código desde cero.
* Los desarrolladores pueden utilizar Python fácilmente con otros lenguajes de programación conocidos, como Java, C y C++.
* La comunidad activa de Python incluye millones de desarrolladores alrededor del mundo que prestan su apoyo. Si se presenta un problema, puede obtener soporte rápido de la comunidad.
* Hay muchos recursos útiles disponibles en Internet si desea aprender Python. Por ejemplo, puede encontrar con facilidad videos, tutoriales, documentación y guías para desarrolladores.
* Python se puede trasladar a través de diferentes sistemas operativos de computadora, como Windows, macOS, Linux y Unix.

### Visual Studio Code

Visual Studio Code (VS Code) es un editor de código fuente ligero pero potente desarrollado por Microsoft. Está disponible para Windows, macOS y Linux y es gratuito para descargar y utilizar. VS Code combina la simplicidad de un editor de código fuente con potentes herramientas para desarrolladores, como la finalización de código IntelliSense y la depuración [10]

Algunas de las características clave de VS Code:

1. Soporte integrado para varios lenguajes de programación: VS Code viene con soporte integrado para JavaScript, TypeScript y Node.JS, y tiene un rico ecosistema de extensiones para otros lenguajes como C++, C#, Java, Python, PHP, Go, y para tiempo de ejecución como .NET y Unity [10].
2. Interfaz de línea de comandos potente: VS Code tiene una potente interfaz de línea de comandos que te permite controlar cómo lanzas el editor. Puedes abrir diferentes archivos, instalar extensiones, e incluso cambiar el idioma de visualización al inicio [10].
3. Integración con Git: VS Code viene con una integración con Git que te permite hacer commit, pull y push de tus cambios de código a un repositorio Git remoto [10].
4. Extensibilidad: Puedes elegir entre miles de extensiones para personalizar tu IDE. Esto te permite agregar funcionalidades adicionales a VS Code para adaptarlo a tus necesidades específicas [10].

### Elasticsearch

Esta base de datos es un motor de analítica y búsqueda de RESTful distribuido que almacena de forma centralizada tus datos para que puedas buscar, indexar y analizar datos de todas formas y tamaños[11].

Algunas características de Elasticsearch son:

* Desarrollado en Java
* Open Source
* Distribuido
* Escalable
* Basado en Lucene

Lucene es una librería de búsqueda de texto[12]:

* Desarrollado en java
* Open Source
* Escalable
* Con alto rendimiento
* Basada en índices invertidos



Fig 1.0

Como se muestra en la figura 1.0 un índice directo en una base de datos es que a cada texto o indexación a la base de datos se guarda con un id, mientras que en índices invertidos se le asigna un id a cada palabra y lo que se guarda en la base de datos es el conjunto de ubicaciones donde se encuentra esta palabra, haciendo así una búsqueda más rápida.

### Git

Git es un sistema de control de versiones distribuido que se utiliza para rastrear cambios en archivos y coordinar el trabajo entre programadores. Es el sistema de control de versiones más popular en el mundo y se utiliza para una amplia gama de proyectos, desde software hasta documentación [13].

Git permite a los usuarios realizar un seguimiento de los cambios en los archivos de un proyecto, lo que facilita la colaboración y la resolución de conflictos. Además, Git permite a los usuarios revertir a una versión anterior de un proyecto si se produce un error o un bug. Esto es especialmente útil en un entorno de desarrollo de software, donde los errores son comunes y es necesario poder volver atrás rápidamente [13].

Git también es compatible con todos los sistemas operativos disponibles y puede acceder directamente a otros repositorios de control de versiones remotos. Esto significa que los usuarios pueden cambiar fácilmente a Git sin tener que mover sus archivos de esos repositorios al repositorio de Git. Además, Git es altamente escalable y más rápido que otros sistemas de control de versiones, lo que permite manejar eficientemente proyectos grande [13].

### Modelos para el reconocimiento de entidades nombradas

**Spacy**

SpaCy es una biblioteca de procesamiento de lenguaje natural (NLP) para Python que proporciona una serie de herramientas para el procesamiento de texto, incluyendo tokenización, etiquetado de partes del discurso, reconocimiento de entidades nombradas y más. spaCy es conocido por su velocidad y eficiencia, y es especialmente útil para tareas de NLP que requieren un análisis rápido y preciso del texto [14].

**Bert**

BERT (Bidirectional Encoder Representations from Transformers) es un modelo de lenguaje preentrenado desarrollado por Google que utiliza la arquitectura Transformer. BERT fue diseñado para comprender el contexto completo de una palabra en una frase considerando todas las palabras antes y después de ella. Este enfoque bidireccional permite que BERT genere representaciones de palabras muy ricas y útiles para una variedad de tareas de NLP.[15]

**Flair**

Flair es un marco de trabajo para el NLP que se centra en la simplicidad y la flexibilidad. Proporciona una interfaz fácil de usar para trabajar con diferentes tipos de incrustaciones de palabras, incluyendo BERT y otros modelos Transformer. Flair también ofrece clases específicas para incrustar palabras y documentos completos. Esto permite a los usuarios incorporar fácilmente estas representaciones avanzadas en sus propias aplicaciones de NLP. Además, Flair permite personalizar el modelo mediante el ajuste fino (fine-tuning) para adaptarlo a tareas específicas. [16]

**NLTK**

NLTK, o Natural Language Toolkit, es una biblioteca gratuita y de código abierto para el procesamiento avanzado del lenguaje natural (NLP) en Python [17]. Se utiliza para simplificar los datos textuales y obtener información profunda de los mensajes de entrada. NLTK proporciona interfaces fáciles de usar para más de 50 corpus y recursos léxicos, como WordNet, junto con una suite de bibliotecas de procesamiento de texto para clasificación, tokenización, stemming, etiquetado, análisis y razonamiento semántico. También incluye envoltorios para librerías de NLP de nivel industrial y un foro de discusión activo. NLTK es reconocido por su amplitud en algoritmos y la calidad de sus recursos, lo que lo convierte en una herramienta poderosa para la educación y la investigación en NLP. Sin embargo, su curva de aprendizaje puede ser pronunciada y, en ocasiones, puede no coincidir con las demandas de uso en producción del mundo real debido a su ritmo de desarrollo más lento [18].

**Stanford**

Stanford NLP (Natural Language Processing) es un conjunto de herramientas de software desarrolladas por el grupo de Stanford NLP para resolver problemas computacionales de lenguaje natural. Estas herramientas incluyen procesamiento estadístico de NLP, aprendizaje profundo de NLP y herramientas basadas en reglas para tareas importantes de lingüística computacional [19].

**Comparación de los modelos de reconocimiento de entidades**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Modelo** | **Dataset** | **Peso del modelo** | **Componentes** | **Precisión** |
| **Spacy** | OntoNotes 5.0 | 541MB | [tok2vec](https://spacy.io/api/tok2vec), [morphologizer](https://spacy.io/api/morphologizer), [parser](https://spacy.io/api/dependencyparser), [senter](https://spacy.io/api/sentencerecognizer), [attribute\_ruler](https://spacy.io/api/attributeruler), [lemmatizer](https://spacy.io/api/lemmatizer), [ner](https://spacy.io/api/entityrecognizer) | 90% |
| **BERT** | CoNLL-2002 | 420MB | Transformer architecture, attention mechanism, fine-tuning capability | 89.86% |
| **Flair** | Conll-03, WikiNER, AIDA-CoNLL-YAGO, | 1.72GB | Transformer architecture, attention mechanism, fine-tuning capability | 86.65% |
| **NLTK** | Personalizable | N/A | Reglas basadas, etiquetado de POS, tokenización | N/A |
| **Standford** | OntoNotes 5.0 | 635MB | Tokenización, etiquetado de POS, análisis de dependencia, reconocimiento de entidades | N/A |

Para este proyecto se elige la biblioteca de Spacy debido a su precisión del 90% lo que es relativamente alto en comparación con los otros modelos mencionados. El tamaño del modelo Spacy es de 541MB, lo que es menor que el de Flair (1.72GB). Esto puede resultar en tiempos de inferencia más rápidos y un menor consumo de memoria, lo cual es importante para aplicaciones en tiempo real o con recursos limitados. Spacy permite personalizar y entrenar modelos en nuevos dominios o con nuevos tipos de entidades, lo cual vamos a utilizar para el reentrenamiento.

### Modelos generadores de oraciones

**GPT – 3.5**

Es la versión mejorada de GPT-3 lanzada en marzo de 2022 por OpenAI. Se conoce como GPT-3.5 y es parte de la familia GPT-3.5 que incluye varias variantes optimizadas, siendo la más avanzada gpt-3.5-turbo.

**GPT-4**

Sucesor de GPT-3.5, lanzado en marzo de 2023, GPT-4 es el modelo de lenguaje más reciente y capaz de OpenAI. Destaca por su capacidad para procesar texto e imágenes, lo que le permite utilizarse para una variedad de propósitos más amplios. GPT-4 está disponible en dos variantes: gpt-4-8K y gpt-4-32K.

**Llama2**

Llama2 es una versión de un modelo de lenguaje grande desarrollado por Facebook. Este modelo ha sido entrenado para generar texto en un estilo similar al de un humano, y es especialmente útil para tareas de generación de texto y procesamiento del lenguaje natural. Llama2 puede generar respuestas a las preguntas, continuar las historias y realizar una variedad de tareas de generación de texto[20].

**Comparación entre Llama 2, GPT-3.5 y GPT-4 para generar oraciones**

|  |  |  |  |
| --- | --- | --- | --- |
| **Modelos** | **Llama2** | **GPT-3.5** | **GPT-4** |
| **Parámetros** | 70 billones | 154 – 175 billones | 1 – 1.76 trillones |
| **Contenido Máximo** | 4096 | 4096  8001  16384 | 8192  32768 |
| **Modalidades** | Texto solamente | Texto solamente | Texto e imagen |
| **Precisión** | 68.9% | 70% | 86.4% |
| **Complejidad** | Baja | Alta | Alta |
| **Velocidad** | Rápido | Lento | Lento |
| **Eficiencia** | Más eficiente | Menos eficiente | Menos eficiente |

* **Tamaño y Parámetros**: Llama 2 es significativamente más pequeño que GPT-3.5 y GPT-4, con el modelo más grande de Llama 2 alrededor del 40-45% menor que GPT-3.5 y aproximadamente el 96% menor que GPT-4. Sin embargo, el tamaño no determina necesariamente la calidad o rendimiento del modelo.
* **Precisión y Complejidad de Tareas**: En pruebas de rendimiento, Llama 2 muestra resultados similares a GPT-3.5, pero GPT-4 supera a ambos en complejidad y creatividad. Llama 2 utiliza una técnica llamada Ghost Attention que mejora su habilidad para controlar el diálogo a lo largo de múltiples turnos.
* **Creatividad**: GPT-4 tiene el mejor nivel de creatividad entre los tres modelos, capaz de generar contenido poético y metáforas sofisticadas. Llama 2 y GPT-3.5 son menos creativos, siendo adecuados para tareas menos complejas y menos artísticas.
* **Token Limit**: Llama 2 tiene el mismo límite de tokens que la variante base de GPT-3.5-turbo, mientras que la variante base de GPT-4 tiene el doble. Si se requieren entradas y salidas más largas, GPT-4 sería la opción preferida.
* **Modalidades**: GPT-4 es el único capaz de procesar entradas visuales estáticas, mientras que Llama 2 y GPT-3.5 solo manejan texto.
* **Velocidad y Eficiencia**: Debido a su tamaño reducido, Llama 2 es más rápido y eficiente que GPT-3.5 y GPT-4, lo que puede ser crítico para proyectos donde la velocidad es importante.
* **Caso de Uso Ideal para Llama 2**: Para construir un chatbot para pequeñas empresas, donde se requiere un manejo de solicitudes relativamente pequeñas y de complejidad moderada. También es una buena opción para herramientas que generen contenido de redes sociales de bajo riesgo debido a su nivel de creatividad limitado.[21]

### React

React es una biblioteca de JavaScript utilizada para construir interfaces de usuario. Permite a los desarrolladores crear aplicaciones web con componentes reutilizables que conforman partes de la interfaz de usuario, lo cual facilita el desarrollo evitando la repetición de código. React introduce JSX, una extensión de la sintaxis de JavaScript que permite combinar la lógica de JavaScript con la lógica de la interfaz de usuario, proporcionando una sintaxis similar a XML que es fácil de leer y escribir [22].

Algunas características clave de React incluyen:

* **Componentes Reutilizables**: Los componentes en React tienen su propia lógica y pueden ser reutilizados en toda la aplicación, lo que reduce la necesidad de repetir código.
* **Renderizado del lado del cliente**: React se utiliza principalmente para construir la interfaz de usuario en el lado del cliente, y se puede utilizar para construir cualquier tipo de aplicación web que se ejecute en la web.
* **Rendimiento**: Gracias al DOM virtual de React, la renderización de las páginas puede hacerse de manera más eficiente, lo que resulta en tiempos de carga más rápidos y una mejor experiencia para el usuario.

### FastAPI

FastAPI es un marco moderno y de alto rendimiento para construir APIs web con Python basado en estándares abiertos. Fue diseñado para ser rápido tanto en términos de velocidad de ejecución como de codificación, facilitar el aprendizaje y reducir errores humanos. Su rendimiento puede compararse con NodeJS y Go y se considera uno de los marcos de trabajo más rápidos disponibles.[23]

Algunas características clave de FastAPI incluyen:

* **Alto rendimiento**: Es comparable en rendimiento con NodeJS y Go, gracias a su uso de Starlette para el manejo HTTP y Pydantic para la validación de datos.
* **Rápido para codificar**: Permite aumentar significativamente la velocidad de desarrollo de las características.
* **Reducción de errores**: Ayuda a reducir aproximadamente el 40% de los errores inducidos por el humano (desarrollador).
* **Intuitivo**: Ofrece excelente soporte de editor, autocompletado y verificación de tipos.
* **Fácil de usar y aprender**: Está diseñado para ser sencillo y requiere menos tiempo leyendo documentación.
* **Robusto**: Proporciona código listo para producción con documentación interactiva automática.
* **Basado en estándares**: Compatible con los estándares abiertos para APIs, OpenAPI y JSON Schema.[24]

### XP

La metodología de software XP se centra en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre. Kent Beck, el padre de XP, describe la filosofía de XP en sin cubrir los detalles técnicos y de implantación de las prácticas. Posteriormente, otras publicaciones de experiencias se han encargado de dicha tarea. A continuación, presentaremos las características esenciales de XP organizadas en los tres apartados siguientes: historias de usuario, roles, proceso y prácticas.[25]

El ciclo de vida de XP consiste en seis fases:

* **Exploración**: En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.
* **Planificación de la Entrega:** En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días. Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos. Por otra parte, el equipo de desarrollo mantiene un registro de la "velocidad" de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración. La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar. Al planificar según alcance del sistema, se divide la suma de puntos de las historias de usuario seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación.
* **Iteraciones:** Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de Entrega está compuesto por iteraciones de no más de tres semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra escogiendo las historias que fuercen la creación de esta arquitectura, sin embargo, esto no siempre es posible ya que es el cliente quien decide qué historias se implementarán en cada iteración (para maximizar el valor de negocio). Al final de la última iteración el sistema estará listo para entrar en producción.
* Los elementos que deben tomarse en cuenta durante la elaboración del Plan de la Iteración son: historias de usuario no abordadas, velocidad del proyecto, pruebas de aceptación no superadas en la iteración anterior y tareas no terminadas en la iteración anterior. Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores. Wake en [18] proporciona algunas guías útiles para realizar la planificación de la entrega y de cada iteración.
* **Producción:** La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase. Es posible que se rebaje el tiempo que toma cada iteración, de tres a una semana. Las ideas que han sido propuestas y las sugerencias son documentadas para su posterior implementación (por ejemplo, durante la fase de mantenimiento).
* **Mantenimiento:** Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura.
* **Muerte del Proyecto:** Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo.[25]

1. **ORGANIZACIÓN Y DISEÑO**

En el siguiente capítulo, se describirá el proceso de planificación y diseño del sistema, aplicando diversas técnicas y herramientas para construir un modelo detallado y preciso de la aplicación. Se exhibirán los diagramas utilizados para la modelación de la aplicación y la estrategia de desarrollo. Se analizarán los criterios de selección de las herramientas y técnicas empleadas, explicando las ventajas de su aplicación durante la fase de desarrollo del sistema. Este capítulo tiene como propósito ofrecer una visión integral y detallada del proceso de planificación y diseño del sistema, actuando como una guía para futuros proyectos similares.

## Propuesta del sistema

Para darle solución al problema de investigación se propone la creación de un sistema de gestión del modelo de reconocimiento de entidades spacy, el modelo generador de oraciones de Llama2. Esto permitiría a los usuarios del centro, realizar el reconocimiento de entidades a los índices de ElasticSearch que cumplan con las condiciones correspondientes para su procesamiento, además en caso de no reconocer una entidad, el usuario podrá especificar el nombre de la entidad, el tipo y una descripción de la misma, para así generar datos de entrenamiento y reentrenar el modelo de Spacy para que reconozca dicha entidad.

## Arquitectura del sistema creado

El diagrama a continuación se puede apreciar los pasos para el reconocimiento de entidades. El paso 1 es seleccionar el índice de Elasticsearch para el reconocimiento de entidades, el paso 2 es una condicional en caso de que el modelo no reconozca una entidad, se procede a los pasos 2.1 para generar nuevos datos de entrenamiento de dicha entidad, y en el paso 2.2 reentrenaremos nuestro modelo de spacy. En caso de que se reconozca todas las entidades pasaría al paso 3 para reincorporar dichas entidades en los índices de Elasticsearch para su posterior estudio

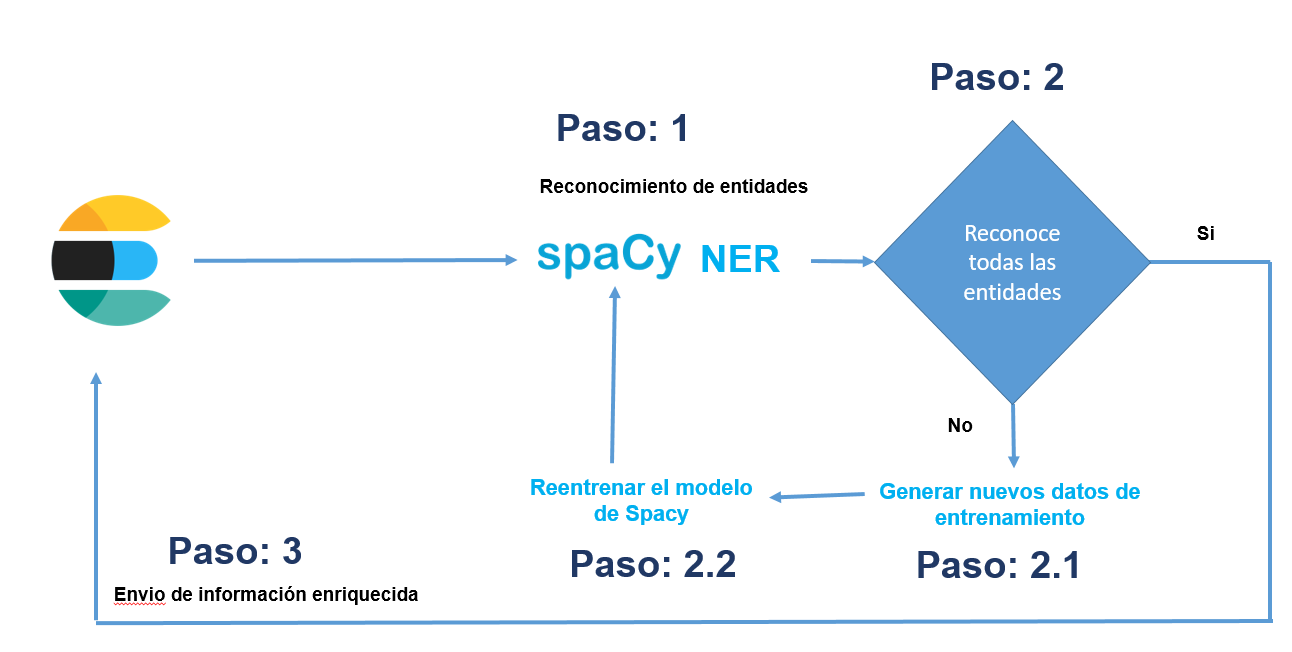


Diagrama de Arquitectura

## Requisitos Funcionales

### Frontend

* RF 1 – Registro del usuario
* RF 2 – Login del usuario
* RF 3 – Mostrar los índices de Elasticsearch
* RF 4 – Mostrar los resultados del reconocimiento de entidades al seleccionar un índice en formato json
* RF 5 – Generar datos de reentrenamiento a partir de un contexto dado
* RF 6 – Reentrenar el modelo de Spacy
* RF 7 – Guardar el índice con las entidades encontradas

### Backend

* RF 1 – Registro del usuario
* RF 2 – Login del usuario
* RF 3 – Mostrar índices de Elasticsearch
* RF 4 – Realizar el reconocimiento de entidades al índice seleccionado y enviar la respuesta en formato json
* RF 5 – Generar datos de reentrenamineto con el modelo de Llama2 dado un contexto dado por el usuario
* RF 6- Reentrenar el modelo Spacy
* RF 7- Comprobación que no hubo pérdida de conocimiento por parte del modelo
* RF 8 – Guardar el índice con las entidades encontradas

## Requisitos No Funcionales

### Software

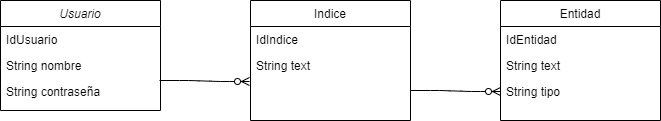
* Todos los sistemas operativos
* Node versión 20.11.0 o superiores
* Python versión 3.10.8
* Elasticsearch versión 8.3.3
* Spacy versión 3.6.1

### Hardware

* Procesador Intel Core i5 12ma gen
* RAM: 16GB aproximadamente
* Almacenamiento: 100GB aproximadamente

## Modelo Entidad Relación

Pese a ser Elasticsearch una base de datos NOSQL, es una buena práctica la creación del modelo de entidad relación el cual refleje las tablas y campos que se van a manejar por el sistema en la base de datos.



## Historias de Usuario

|  |  |
| --- | --- |
| **Historia de usuario** | |
| **Número**: 1 | **Nombre de Historia de Usuario**: Login del usuario en el sistema |
| **Usuario**: Los trabajadores del centro | |
| **Prioridad** **en negocio**: Alta | **Riesgo de desarrollo**: Bajo |
| **Descripción**: Por temas de seguridad es necesario la autenticación al sistema por parte de los usuarios del centro. En caso de no tener cuenta deberá de registrarse | |
|  | |

|  |  |
| --- | --- |
| **Historia de usuario** | |
| **Número**: 2 | **Nombre de Historia de Usuario**: Reconocimiento de entidades de los índices de Elasticsearch |
| **Usuario**: Los trabajadores del centro | |
| **Prioridad** **en negocio**: Alta | **Riesgo de desarrollo**: Alto |
| **Descripción**: Los usuarios serán capaces de seleccionar el índice al cual se le desea hacer un reconocimiento de entidades y en caso de estar correcto almacenarlo en la base de datos de Elasticsearch | |
|  | |

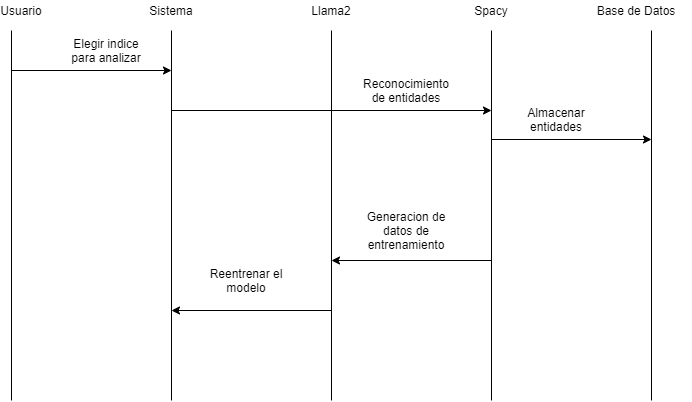
|  |  |
| --- | --- |
| **Historia de usuario** | |
| **Número**: 3 | **Nombre de Historia de Usuario**: Reentrenamiento del modelo |
| **Usuario**: Administradores del centro | |
| **Prioridad** **en negocio**: Alta | **Riesgo de desarrollo**: Alto |
| **Descripción**: Los usuarios serán capaces de reentrenar el modelo pasando los parámetros solicitados para la generación de dichos datos de entrenamiento | |
|  | |

## Usuarios del Sistema

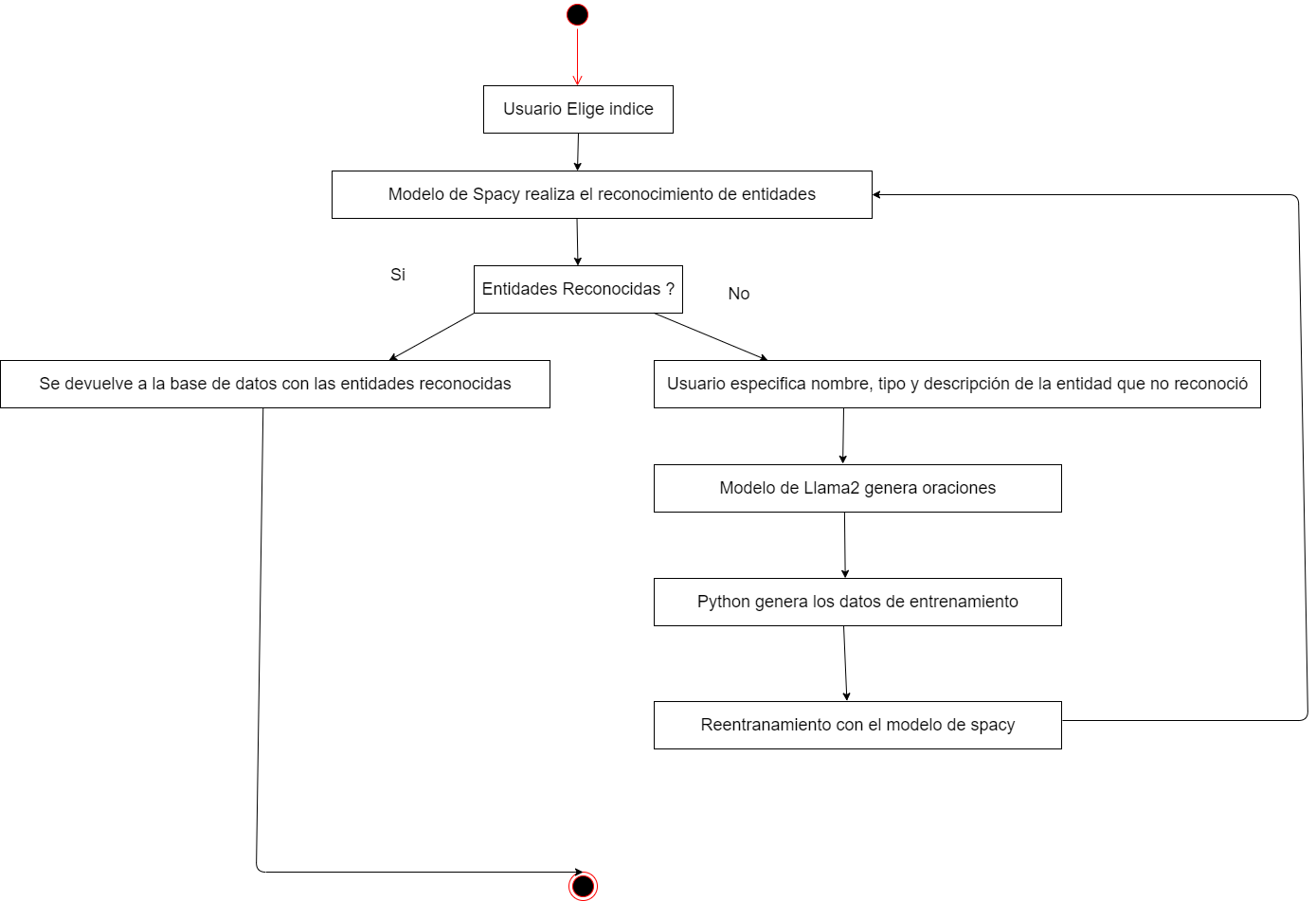
|  |  |
| --- | --- |
| **Actor** | **Tareas** |
| **Usuario** | Visualizar, realizar reconocimiento de entidades, agregar a base de datos |
| **Administrador** | Administra la base de datos del sistema, visualizar, realiza reconocimiento de entidades, agrega a la base de datos, reentrena el modelo de Spacy |

## Diagrama de Secuencia

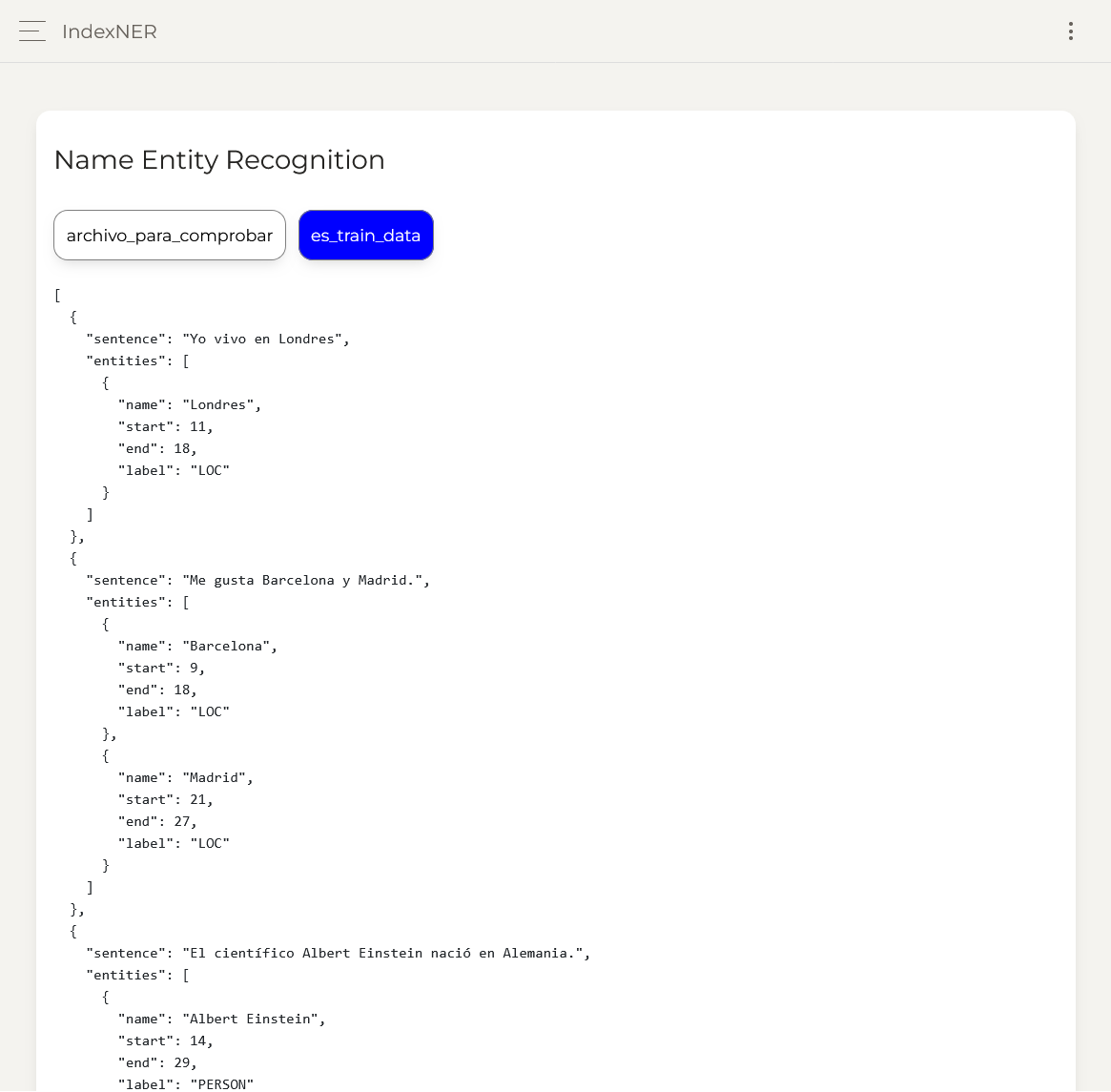
En el siguiente diagrama se observa los pasos para la realización del reconocimiento y el reentrenamiento.

****

## Diagrama de Actividades

****

## Diseño de la Interfaz

****

**Primer epígrafe del tercer capítulo**

A continuación se muestra ejemplo de una tabla. Las principales normas a seguir son: el título se colocará en la parte superior de la tabla a continuación del término Tabla, solamente seguido del número del capítulo, punto y el número de orden de la tabla en ese capítulo. No se debe emplear la abreviatura No. ni el signo #; el espaciado del título de la tabla debe ser sencillo; cada columna de la tabla llevará su título, procurando no hacer abreviaturas; los títulos de las columnas se encerrarán entre dos líneas horizontales sencillas. La tabla se cerrará también con una línea horizontal sencilla. En caso que la tabla no sea de su autoría, debe ser agregada la Fuente de referencia de donde fue tomada la misma.

*Tabla 3.1. Escribir aquí el título de la tabla (Fuente: sitio de la UIT, 2014)*

|  |  |  |  |
| --- | --- | --- | --- |
| Título de la columna uno\* | Título de la columna dos | Título de la columna tres | Título de la columna cuatro |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

\* Las llamadas para explicar algo en la tabla se deberán hacer con asteriscos u otro símbolo, y las notas se colocarán al pie de la tabla y no al pie de la página.

Las ecuaciones deben ser enumeradas siguiendo el mismo esquema de las tablas, y la etiqueta numérica debe ser colocada a la derecha tratando de uniformizar esa ubicación para todas las ecuaciones que se utilicen, Por ejemplo en (3.1) se calcula el área de un círculo.

|  |  |
| --- | --- |
|  | **(3.1)** |

Ahora veamos un ejemplo de figura, en este caso una gráfica experimental, por ser de las más utilizadas. Las principales características a respetar son: el número y título deberán aparecer en la parte inferior de la figura, a continuación del término Figura; colocar nombre, símbolo y/o unidades para las variables sobre los ejes correspondientes y se usarán líneas y/o símbolos diferentes para distinguir las diferentes curvas que se puedan dibujar en una misma figura. No se admiten las palabras "foto" ni "gráfica", debe utilizarse la palabra Figura o Fig. seguida por un espacio, el número del capítulo, punto y el número de orden dentro del capítulo al cual pertenece, Debe utilizarse Arial 10 en itálica. En caso que la figura no sea de su autoría, debe colocarse la Fuente de referencia de los datos en el pie de la misma.

2000

2002

2004

80

100

120

140

160

180

200

220

240

Eje x

Eje Y

Curva 1

Curva 2

*Figura 3.1. Escribir aquí el título de la figura (Fuente: [3]).*

Continúe con el texto.

**Primera división (o sub-epígrafe) dentro del primer epígrafe**

Inicie a escribir aquí. Si alguno de los subtítulos hubiera quedado en la última línea de una página, sería necesario trasladarlo a la primera línea de la página siguiente

**Ponga aquí el segundo epígrafe**

Inicie a escribir aquí.

**Primera división (o sub-epígrafe) dentro de este epígrafe**

1. **CONCLUSIONES Y RECOMENDACIONES**

**Conclusiones**

Las conclusiones deben estar en correspondencia con los objetivos planteados. Ellos deben referirse al problema de investigación planteado, de hecho la primera conclusión de un estudio es evaluar qué ocurrió con el planteamiento. Deben ser objetivas, breves, precisas y convincentes. Es importante destacar que las conclusiones no son un recuento de lo que se realizó en el trabajo. La conclusión constituye una generalización o comentario aplicado,.relativa a los resultados obtenidos

Para una mejor comprensión pueden ser numeradas y si algún caso lo requiere, pueden ser comentadas. Ejemplos

Se comprobó que el efecto del jitter sobre la BER no es significativo cuando la velocidad de transmisión no excede los 4 Mbps

Se verificó que …

Se demostró que …

Conclusión 3

**Recomendaciones**

Escriba lo que usted recomienda a otros investigadores para mejorar su trabajo o para determinar otros datos que usted no pudo lograr, sugerir qué hacer con sus resultados y aportes.

Recomendación 1

Recomendación 2

Recomendación 3

No debería haber muchas recomendaciones pues si uno recomienda muchas acciones la lectura sería que no se avanzó mucho en la investigación.

La extensión de estos aspectos no debe exceder de cinco páginas.

1. **REFERENCIAS BIBLIOGRÁFICAS**

Este capítulo no se enumera. Incluyen los materiales que fueron utilizados durante el desarrollo del trabajo.

Deberá usarse el estilo de citación de la IEEE (orden numérico. ubicando Al citar usando este estilo, las referencias deben ser ordenadas en el orden que fueron citadas.

Recuerde que utilizando un gestor bibliográfico como el que viene embebido en Microsoft Word o en *Endnote*, el ordenamiento de las referencias se hará automáticamente. Debe utilizar las normas Harvard.

Las referencias se señalan cuando se hace alusión directa o indirecta a un autor o a su obra, sobre todo si se le cita textualmente. Recuerde que la utilización de palabras o conceptos de otro autor sin señalar la fuente constituye un delito (PLAGIO) sancionado. Denota además cierta falta de ética.

|  |  |
| --- | --- |
| [1] | R. H. Sampieri, C. F. Collado y P. B. Lucio, Metodología de la Investigación, Istapalapa: Mc Graw Hill, 2006.  Autor(es) (Año). Título del Artículo, Título de la Revista, Número del Volumen, (Número de Edición): Páginas.  Autor(es) (Año). Título del libro en itálica. Número de la edición, Editorial, Lugar de Publicación.  Autor(es) (Año). Título del Artículo o del Capítulo en el Libro, En: Nombre del Libro, Número de la Edición, Número del Capítulo, Editores, Editorial, Lugar de la Publicación. |

Se sugiere utilizar los gestores bibliográficos[[1]](#footnote-1) para la gestión de las referencias bibliográficas, de esta forma ellos harán todo el proceso de organización y formateo por el autor.. El uso del gestor posibilita la exportación de un archivo (base de datos) con una lista detallada de las referencias utilizadas, la cual debe ser entregada al/los tutor/es. Esta base de datos es de utilidad para la continuidad de las investigaciones en la institución académica )

**BIBLIOGRAFIA (**opcional**)**

[1] R. Viana, «Reconocimiento de entidades nombradas: Una breve introducción», SoldAI. Accedido: 8 de noviembre de 2023. [En línea]. Disponible en: https://medium.com/soldai/reconocimiento-de-entidades-nombradas-una-breve-introducci%C3%B3n-ad6e2a3f911a

[2] V. Osorio, «¿Qué es el Reconocimiento de Entidades Nombradas?», Klippa. Accedido: 25 de octubre de 2023. [En línea]. Disponible en: https://www.klippa.com/es/blog/informativo/reconocimento-entidades-nombradas/

[3] «Haag, K. Y. Reconocimiento de entidades nombradas en texto de dominio legal.pdf».

[4] «SanchezPCR.pdf».

[5] R. Viana, «Reconocimiento de entidades nombradas: Una breve introducción», SoldAI. Accedido: 9 de noviembre de 2023. [En línea]. Disponible en: https://medium.com/soldai/reconocimiento-de-entidades-nombradas-una-breve-introducci%C3%B3n-ad6e2a3f911a

[6] «Qué es el reconocimiento de entidad nombrada (NER): definición, ejemplos, tipos y aplicaciones». Accedido: 9 de noviembre de 2023. [En línea]. Disponible en: https://es.shaip.com/blog/named-entity-recognition-and-its-types/

[7] «Top 5 named entity recognition APIs to use in your app». Accedido: 17 de diciembre de 2023. [En línea]. Disponible en: https://datavid.com/blog/named-entity-recognition-apis

[8] D. Santos, «Procesamiento de lenguaje natural: qué es, ejemplos y herramientas». Accedido: 9 de noviembre de 2023. [En línea]. Disponible en: https://blog.hubspot.es/marketing/procesamiento-de-lenguaje-natural

[9] «¿Qué es Python? - Explicación del lenguaje Python - AWS», Amazon Web Services, Inc. Accedido: 9 de noviembre de 2023. [En línea]. Disponible en: https://aws.amazon.com/es/what-is/python/

[10] «Why Visual Studio Code?» Accedido: 9 de noviembre de 2023. [En línea]. Disponible en: https://code.visualstudio.com/docs/editor/whyvscode

[11] «Lista de características de Elasticsearch | Elastic». Accedido: 7 de octubre de 2023. [En línea]. Disponible en: https://www.elastic.co/es/elasticsearch/features

[12] «Qué es Lucene. Qué es Lucene y qué relación tiene con Solr.» Accedido: 7 de octubre de 2023. [En línea]. Disponible en: https://solrtutorial.es/que-es-lucene.html

[13] B. Marijan, «What Is Git and What Is It Used For? {+ Main Features}», Knowledge Base by phoenixNAP. Accedido: 9 de noviembre de 2023. [En línea]. Disponible en: https://phoenixnap.com/kb/what-is-git

[14] «Probamos SpaCy: mucho más que una librería para crear proyectos reales de procesamiento del lenguaje natural | datos.gob.es». Accedido: 7 de octubre de 2023. [En línea]. Disponible en: https://datos.gob.es/es/blog/probamos-spacy-mucho-mas-que-una-libreria-para-crear-proyectos-reales-de-procesamiento-del

[15] «BERT 101 - State Of The Art NLP Model Explained». Accedido: 6 de febrero de 2024. [En línea]. Disponible en: https://huggingface.co/blog/bert-101

[16] «flair | flair». Accedido: 6 de febrero de 2024. [En línea]. Disponible en: https://flairnlp.github.io/

[17] J. Karajgikar, «Guides: Text Analysis: NLTK Package». Accedido: 6 de febrero de 2024. [En línea]. Disponible en: https://guides.library.upenn.edu/penntdm/python/nltk

[18] «NLTK :: Natural Language Toolkit». Accedido: 6 de febrero de 2024. [En línea]. Disponible en: https://www.nltk.org/

[19] «Stanford NLP | Stanford NLP Python | Stanford NLP Tutorial». Accedido: 6 de febrero de 2024. [En línea]. Disponible en: https://www.analyticsvidhya.com/blog/2019/02/stanfordnlp-nlp-library-python/

[20] K. Luzniak, «6 main differences between Llama 2, GPT-3.5 & GPT-4», Neoteric. Accedido: 25 de octubre de 2023. [En línea]. Disponible en: https://neoteric.eu/blog/6-main-differences-between-llama2-gpt35-and-gpt4/

[21] K. Luzniak, «6 main differences between Llama 2, GPT-3.5 & GPT-4», Neoteric. Accedido: 10 de febrero de 2024. [En línea]. Disponible en: https://neoteric.eu/blog/6-main-differences-between-llama2-gpt35-and-gpt4/

[22] «Qué es React: definición, características y funcionamiento». Accedido: 6 de febrero de 2024. [En línea]. Disponible en: https://www.hostinger.es/tutoriales/que-es-react

[23] «Features - FastAPI». Accedido: 6 de febrero de 2024. [En línea]. Disponible en: https://fastapi.tiangolo.com/features/

[24] S. Ramírez, «tiangolo/full-stack-fastapi-postgresql». 6 de febrero de 2024. Accedido: 6 de febrero de 2024. [En línea]. Disponible en: https://github.com/tiangolo/full-stack-fastapi-postgresql

[25] P. Letelier y P. Letelier, «Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP)», www.cyta.com.ar/ta0502/v5n2a1.htm. Accedido: 20 de febrero de 2024. [En línea]. Disponible en: http://www.cyta.com.ar/ta0502/b\_v5n2a1.htm

Es opcional, puede incluir aquí los libros que fueron leídos, pero que no fueron referidos en el trabajo.

1. **GLOSARIO DE TÉRMINOS**

Se utiliza cuando durante el desarrollo se han utilizado palabras, conceptos, siglas, etc., que no expresan una idea clara, precisa y concisa, o que puedan tener múltiples interpretaciones

Cuando aparece una sigla por primera vez, ella se escribe normalmente y entre paréntesis se pone el significado de sus términos en el idioma original y a su vez se pone en el Glosario de términos, cuando aparece nuevamente no es necesario poner entre paréntesis el significado de la sigla.

El glosario de términos se anexa después de la bibliografía y antes de los anexos y se escribe en orden alfabético, Se recomienda no abusar de los anglicismos y si se utilizan deben escribirse en itálicas

1. **ANEXOS**

Anexos: se colocan al final del trabajo, no se numeran como capítulos, pero sí se les coloca un número de orden luego de la palabra ANEXO; en ellos se incluye información que no es necesaria para la presentación del trabajo, pero sí útil al lector. Exigen que se hagan referencia de ellos en el contenido o cuerpo de la obra y deben aparecer en el mismo orden en que han sido citados y deben tener un pie de grabado o título que enuncie lo que ilustra el mismo.

Los anexos deben recoger aquellos aspectos del trabajo que, por su longitud, complejidad o que no era adecuado ponerlos, no se incluyen en el texto de la tesis pero que coadyuvan a una mejor comprensión de lo que se expone en ella. Sin embargo, es necesario dejar bien claro que los anexos no pueden usarse para extender la tesis más allá del límite establecido.

Es opcional su uso

**Anexo I** Inserte título del primer anexo

Inserte aquí el primer anexo.

**Anexo II** Inserte título del segundo anexo

Puede añadir tantos anexos como le sean necesarios.

1. [↑](#footnote-ref-1)