Observatorio de Demanda Laboral en Tecnología en Latinoamérica

[Grupo 8]

ESPECIFICACIÓN DE REQUERIMIENTOS DE SOFTWARE

[Noviembre 2025]

[Versión 2.1 - Fase 0 Implementada]

Autores:

Nicolas Francisco Camacho Alarcón
Alejandro Pinzón Fajardo

Pontificia Universidad Javeriana
Facultad de Ingeniería

Bogotá, Colombia
Noviembre de 2025

CONTENIDO

INTRODUCCIÓN

1.1 Propósito

El presente documento tiene como propósito especificar de manera detallada los requerimientos funcionales y no funcionales del sistema denominado *Observatorio de Demanda Laboral en Tecnología en Latinoamérica*, una herramienta de análisis automatizado orientada a procesar, extraer y segmentar habilidades tecnológicas desde portales de empleo en línea, mediante técnicas modernas de procesamiento de lenguaje natural (NLP), scraping, embeddings semánticos y clustering no supervisado.

Este documento está dirigido principalmente a los siguientes públicos:

- El equipo de desarrollo del proyecto, compuesto por los estudiantes Nicolas Francisco Camacho Alarcón, Alejandro Pinzón Fajardo y Daniel Vidal, como guía estructurada para la implementación y validación del sistema.
- El director del proyecto, Ing. Luis Gabriel Moreno Sandoval, y los jurados evaluadores, como evidencia formal del entendimiento técnico y conceptual del producto a desarrollar.
- Otros actores académicos o institucionales interesados en replicar o adaptar el sistema en contextos similares, como universidades, centros de investigación o entidades públicas vinculadas al análisis del mercado laboral.

El documento cubre la totalidad del sistema propuesto, sin limitarse a un solo módulo o subsistema. Por tanto, especifica requerimientos relacionados con la adquisición de datos (scraping), su procesamiento semántico, análisis estadístico y segmentación por perfiles laborales, así como aspectos de validación, modularidad, documentación técnica y estándares de calidad.

La importancia de este documento radica en su papel como contrato técnico entre los actores involucrados, asegurando una visión compartida del comportamiento esperado del sistema, las restricciones existentes, los criterios de aceptación y los estándares metodológicos adoptados. Además, facilita la trazabilidad entre los objetivos definidos en la propuesta de grado y las funcionalidades implementadas en cada fase, garantizando coherencia metodológica, control de calidad y sostenibilidad del desarrollo.

1.2 Alcance

El sistema propuesto, titulado *Observatorio de Demanda Laboral en Tecnología en Latinoamérica*, tiene como propósito desarrollar una herramienta automatizada capaz de analizar la evolución de las habilidades tecnológicas demandadas en el mercado laboral digital, específicamente en los países de Colombia (CO), México (MX) y Argentina (AR). El sistema abarca desde la recolección periódica de datos a través de scraping en portales de empleo hasta el procesamiento semántico y la segmentación de perfiles laborales utilizando técnicas avanzadas de NLP y clustering no supervisado.

Alcance geográfico: Colombia (CO), México (MX), Argentina (AR)

Fuentes de datos: 11 portales de empleo incluyendo hiring.cafe, computrabajo, bumeran, elempleo.com, zonajobs, infojobs, entre otros.

Taxonomía base: El sistema utiliza una taxonomía unificada de **14,174 skills totales**, compuesta por:

- **ESCO v1.1.0:** 13,939 skills (base europea de competencias laborales)
- **O*NET Hot Technologies:** 152 skills (tecnologías emergentes del sector IT)
- **Manual Curated Skills:** 83 skills (específicas para el mercado tech latinoamericano)

Stack tecnológico: Python 3.10+, Scrapy, spaCy, PostgreSQL, FAISS, embeddings E5 multilingües (intfloat/multilingual-e5-base, 768D), HDBSCAN, UMAP.

El producto incluirá las siguientes funcionalidades principales:

- Extracción automatizada de vacantes desde 11 portales web, mediante spiders adaptables que operan respetando las normas de uso de cada sitio.
- Procesamiento y limpieza textual, incluyendo tokenización, lematización y detección de habilidades explícitas mediante técnicas híbridas de NER, expresiones regulares y modelos de lenguaje.
- Representación semántica de habilidades mediante embeddings multilingües (modelo E5 de 768 dimensiones) y reducción de dimensionalidad (UMAP).
- Mapeo de habilidades contra taxonomía ESCO mediante estrategia de tres capas: matching exacto, fuzzy matching y semantic matching con FAISS.
- Agrupamiento de perfiles mediante algoritmos robustos como HDBSCAN, que permitan segmentar la demanda en grupos funcionales coherentes.
- Visualización macro de resultados a través de gráficos interpretables y reportes estáticos que permitan identificar patrones emergentes sin requerir dashboards interactivos.

- Documentación metodológica y código reproducible, que permitan replicar o adaptar el sistema a otras regiones o sectores, bajo principios de ética, apertura y eficiencia computacional.

Este sistema no contempla el desarrollo de una interfaz web pública ni funcionalidades de tipo portal o dashboard interactivo, sino que prioriza la generación de reportes analíticos estáticos y reutilizables, como producto de validación académica y técnica.

El alcance funcional se circunscribe al dominio de las ofertas de empleo tecnológicas publicadas en español en los países mencionados, sin contemplar vacantes en otros idiomas ni otros sectores económicos. Sin embargo, el diseño modular del sistema permitirá su adaptación futura a nuevos contextos geográficos o temáticos.

1.3 Definiciones, Acrónimos y Abreviaciones

1.3.1 Portales de empleo

Son plataformas web donde empresas publican vacantes laborales y profesionales buscan oportunidades. En este proyecto se consideran fuentes como LinkedIn, Computrabajo, Bumeran, ZonaJobs e Indeed, que constituyen insumos primarios para los procesos de scraping y análisis.

1.3.2 Web Scraping

Técnica de recolección automatizada de datos desde páginas web, utilizando librerías como BeautifulSoup, Selenium o Playwright. Permite extraer de forma estructurada información relevante de las ofertas publicadas.

1.3.3 Oferta laboral

Se refiere al anuncio publicado por una organización donde se describe el perfil buscado, incluyendo título del cargo, funciones, requisitos y habilidades deseadas.

1.3.4 Base de datos relacional (PostgreSQL)

Sistema que organiza los datos recolectados en tablas interconectadas, facilitando su consulta, limpieza y posterior análisis mediante estructuras SQL.

1.3.5 Normalización de datos

Proceso de limpieza, estandarización y unificación de formatos para reducir ambigüedad, errores y duplicados, y mejorar la coherencia del análisis posterior.

1.3.6 Expresiones regulares (Regex)

Lenguaje sintáctico utilizado para identificar y extraer patrones textuales específicos (como frases que contengan habilidades o requisitos) en grandes volúmenes de texto.

1.3.7 Named Entity Recognition (NER)

Técnica de procesamiento de lenguaje natural (NLP) que identifica y clasifica entidades en un texto, como nombres de habilidades, empresas o tecnologías.

1.3.8 Tokenización

Consiste en dividir un texto en unidades mínimas llamadas "tokens" (palabras, signos u oraciones), facilitando el análisis lingüístico automatizado.

1.3.9 Lematización

Proceso que transforma las palabras a su forma canónica o raíz gramatical, permitiendo uniformar variaciones morfológicas del lenguaje.

1.3.10 Stopwords

Términos frecuentes sin valor informativo (como "de", "por", "la"), comúnmente eliminados en tareas de procesamiento textual.

1.3.11 Co-ocurrencia

Medida estadística que indica la frecuencia con que dos o más términos aparecen juntos en un texto, útil para detectar relaciones semánticas.

1.3.12 Bigramas y trigramas

Secuencias de dos o tres palabras consecutivas utilizadas para capturar patrones de lenguaje más complejos que las palabras individuales.

1.3.13 LLM (Large Language Models)

Modelos de lenguaje de gran escala (como GPT o T5) entrenados sobre corpus masivos, capaces de generar texto, extraer conocimiento implícito y realizar razonamiento contextualizado.

1.3.14 Prompt Engineering

Diseño estratégico de instrucciones o ejemplos para guiar la salida de un LLM, crucial en tareas de extracción de habilidades o clasificación de ocupaciones.

1.3.15 Few-shot learning

Habilidad de los LLMs para realizar tareas complejas con pocos ejemplos, lo cual resulta clave cuando se carece de datasets etiquetados masivamente en español.

1.3.16 Embeddings semánticos

Representaciones numéricas de textos que capturan similitudes semánticas, permitiendo análisis cuantitativos y clustering. Ejemplos incluyen word2vec, BERT y E5.

1.3.17 Embeddings multilingües

Vectores entrenados para representar texto en múltiples idiomas en un mismo espacio semántico. Son esenciales para manejar contenido mixto español-inglés en ofertas laborales.

1.3.18 UMAP (Reducción de dimensionalidad)

Técnica que transforma espacios de alta dimensionalidad en representaciones más simples, conservando la estructura semántica subyacente para facilitar análisis y visualización.

1.3.19 Clustering (HDBSCAN)

Algoritmo no supervisado que detecta grupos naturales de observaciones (como habilidades o perfiles laborales) según su similitud semántica, sin requerir número de clusters predefinido.

1.3.20 Taxonomía de habilidades (ESCO, CIUO-08, O*NET)

Sistemas jerárquicos y normalizados de clasificación de habilidades y ocupaciones, fundamentales para anclar el análisis a estándares internacionales y mejorar interoperabilidad de los resultados.

1.3.21 FAISS (Facebook AI Similarity Search)

Biblioteca de código abierto para búsqueda eficiente de similitud en espacios vectoriales de alta dimensionalidad. El sistema utiliza FAISS IndexFlatIP para búsqueda exacta de vecinos más cercanos con producto interno, logrando velocidades de 30,147 consultas por segundo, aproximadamente 25 veces más rápido que PostgreSQL con pgvector.

1.3.22 Estrategia de tres capas (Three-layer matching)

Metodología implementada para mapear habilidades extraídas contra la taxonomía ESCO:

- **Layer 1 - Exact Match:** Búsqueda exacta mediante SQL ILIKE con confianza 1.0
- **Layer 2 - Fuzzy Match:** Similitud difusa con fuzzywuzzy, threshold 0.92, confianza 0.92-1.0

- **Layer 3 - Semantic Match:** Búsqueda semántica con FAISS, threshold 0.87, confianza 0.87-1.0 (actualmente deshabilitado debido a limitaciones del modelo E5 con vocabulario técnico)

1.3.23 Skills emergentes

Habilidades extraídas de ofertas laborales que no pueden ser mapeadas a la taxonomía ESCO existente. Representan el 87.4 % de las skills extraídas y constituyen una señal valiosa sobre tendencias emergentes del mercado tech latinoamericano, no un fallo del sistema.

1.3.24 Natural Language Processing (NLP)

Conjunto de técnicas de inteligencia artificial, combinando modelos de lingüística computacional, machine learning y aprendizaje profundo, para poder procesar lenguaje humano.

1.3.25 Python

Lenguaje de programación ampliamente utilizado en ciencia de datos y NLP, por su sintaxis sencilla y librerías especializadas como scikit-learn, spaCy, transformers y pandas.

1.4 Apreciación Global

El presente documento de Especificación de Requerimientos del Software (SRS) tiene como objetivo presentar de manera estructurada y detallada los aspectos fundamentales del sistema "Observatorio de Demanda Laboral en Tecnología en Latinoamérica". La organización del documento se ha realizado con el propósito de facilitar su comprensión tanto para usuarios técnicos como no técnicos, brindando una visión progresiva desde el contexto general hasta los requerimientos específicos del sistema.

El contenido del documento se distribuye de la siguiente manera:

- En la **Sección 1**, se expone la introducción general del proyecto, incluyendo su propósito, alcance, definiciones clave, referencias utilizadas y una apreciación global de su contenido.
- La **Sección 2** describe de manera general los factores que afectan al producto, incluyendo su perspectiva, interfaces con otros sistemas y con el usuario, consideraciones de hardware y software, restricciones de memoria, operaciones del sistema y requerimientos de adaptación al entorno.
- La **Sección 3** presentará los requerimientos funcionales y no funcionales del sistema, detallando cada una de las funcionalidades esperadas, así como las restricciones y condiciones necesarias para su correcto funcionamiento.
- En las **Secciones 4 y 5**, se incluirán descripciones de cómo se piensan manejar los requerimientos mencionados anteriormente, así como el proceso de verificación y validación.

- Finalmente, se anexarán diagramas, tablas de trazabilidad y otros elementos que complementen la especificación del sistema.

Este documento servirá como base para el diseño, desarrollo, validación y evaluación del sistema propuesto, asegurando que todos los actores involucrados compartan una visión clara y consensuada de los objetivos, alcances y funcionalidades del software a implementar.

1.5 Estado Actual de Implementación

Versión del sistema: 2.1
Última actualización: Octubre 22, 2025
Estado general: Implementación en Progreso

1.5.1 Fases Completadas

- **Fase 0 - Configuración Inicial:** COMPLETADA (100 %)
  - Taxonomía de skills cargada: 14,174 skills (ESCO 13,939 + O*NET 152 + Manual 83)
  - Embeddings generados: 14,133 embeddings únicos (768D, L2-normalized)
  - Índice FAISS construido: 41.41 MB, 30,147 queries/segundo
  - Tiempo total de ejecución: 25 segundos
  - Tests automatizados: 37 tests, 94.6 % pass rate
- **Fase 1 - Recolección y Limpieza de Datos:** COMPLETADA (100 %)
  - Total jobs scraped: 23,352 (hiring.cafe: 23,313, elempleo: 38, zonajobs: 1)
  - Jobs limpios y utilizables: 23,188 (99.5 %)
  - Jobs basura filtrados: 125 (0.5 %)
  - Promedio palabras por job: 552
  - Deduplicación implementada: SHA256 content hash
- **Fase 2 - Extracción de Skills (Pipeline A):** IMPLEMENTADA
  - Método Regex: 200+ patrones tecnológicos, 78-89 % precision
  - Método NER: spaCy es_core_news_sm + custom entity ruler
  - Mapeo ESCO: Layer 1 (Exact) + Layer 2 (Fuzzy) activos
  - Match rate actual: 12.6 % (esperado para taxonomías 2016-2017)
  - Test con 100 jobs: 100 % success rate, 2,756 skills extraídas

1.5.2 Fases Pendientes

- **Fase 2 - Pipeline B (LLM-based):** NO IMPLEMENTADO
  - Extracción con LLMs (GPT, Mistral, Llama)
  - Comparación Pipeline A vs Pipeline B
- **Módulo 6 - Clustering:** NO IMPLEMENTADO
  - UMAP dimensionality reduction
  - HDBSCAN clustering
  - Análisis temporal de clusters
- **Módulo 7 - Visualizaciones:** NO IMPLEMENTADO
  - Gráficos interactivos (Plotly)
  - Reportes analíticos estáticos
  - Network analysis de co-ocurrencias

1.5.3 Decisiones Técnicas Importantes

- **Layer 3 Semantic Matching DESHABILITADO:** El modelo E5 multilingual demostró ser inadecuado para vocabulario técnico, generando matches absurdos (ej: React"→ "neoplasiaçon score 0.82). Se mantendrá deshabilitado hasta implementar un modelo domain-specific o LLM-based classification.
- **Match rate 12.6 % es ACEPTABLE:** ESCO/O*NET son taxonomías tradicionales (2016-2017) que no cubren frameworks modernos. El 87.4 % de skills emergentes representa señal valiosa del mercado, no un fallo.
- **FAISS performance validada:** 30,147 q/s, 25x más rápido que PostgreSQL pgvector, con 100 % precision (IndexFlatIP exact search).

DESCRIPCIÓN GENERAL

2.1 Perspectiva del Producto

El sistema propuesto, denominado *Observatorio de Demanda Laboral en Tecnología en Latinoamérica*, corresponde a un producto completamente nuevo, diseñado desde cero con el fin de ofrecer una solución automatizada, académicamente robusta y técnicamente escalable para el análisis de habilidades tecnológicas demandadas en el mercado laboral digital. Si bien es novedosa la implementación y el diseño, se basa en múltiples componentes o sistemas similares ya propuestos en literatura Europea, Africana, y Estadounidense, lo que robustece la facilidad de su implementación.

Este sistema no reemplaza una herramienta previa ni se integra como módulo en un sistema existente; por el contrario, responde a una necesidad actual no cubierta de manera integral en los contextos académico y gubernamental latinoamericano: la escasez de plataformas automatizadas que permitan entender en profundidad la evolución semántica de las habilidades requeridas en el sector tecnológico, a partir de vacantes en línea publicadas en español.

La decisión de desarrollar este producto surge de una combinación de motivaciones técnicas y sociales:

- Desde el punto de vista técnico, se busca aplicar metodologías avanzadas de procesamiento de lenguaje natural, extracción de entidades, embeddings multilingües y clustering semántico para lograr una segmentación coherente y útil de perfiles laborales.
- Desde una perspectiva social y estratégica, se pretende brindar herramientas de análisis que apoyen a universidades, centros de formación, entidades públicas y actores del ecosistema digital en la identificación temprana de brechas de habilidades, facilitando la toma de decisiones informadas en política educativa, empleabilidad y reconversión laboral.

El sistema propuesto se distingue de otros enfoques parciales por su carácter modular, reproducible y enfocado en la generación de conocimiento estratégico a partir de datos abiertos. Su desarrollo también permitirá validar técnicas emergentes de extracción e inferencia con LLMs en español, contribuyendo al cuerpo académico y técnico en ciencia de datos aplicada al empleo.

En síntesis, el Observatorio representa un aporte original y pertinente tanto desde el plano metodológico como desde su aplicabilidad social, al combinar scraping ético, NLP multilingüe y visualización analítica para abordar un problema real en el contexto latinoamericano.

2.1.1 Interfaces con el sistema

El sistema "Observatorio de Demanda Laboral en Tecnología en Latinoamérica" es un producto completamente nuevo y autónomo, por lo cual no mantiene interfaces directas con sistemas externos en tiempo de ejecución. Sin embargo, presenta una arquitectura modular interna donde cada componente se comunica con los demás a través de interfaces internas bien definidas.

El flujo funcional del sistema se organiza de manera secuencial y conectada, iniciando con el módulo de web scraping, el cual actúa como punto de entrada para la adquisición de datos desde portales de empleo. Cada uno de los módulos siguientes procesa la información recibida del anterior, generando una línea de procesamiento de datos estructurada y continua.

Las interfaces internas entre módulos incluyen:

- **Scraping → Almacenamiento estructurado:** Cada spider obtiene vacantes desde portales como Computrabajo, Bumeran y elempleo.com, y deposita los datos en bruto en archivos .json o .csv, los cuales son posteriormente integrados a una base de datos PostgreSQL.
- **Almacenamiento → Procesamiento semántico:** Los textos extraídos son lematizados, tokenizados y limpiados utilizando librerías como spaCy, nltk y regex, para luego ser enviados al módulo de extracción de habilidades.
- **Procesamiento semántico → Enriquecimiento con LLMs:** Las habilidades explícitas e implícitas se extraen mediante NER, expresiones regulares y modelos de lenguaje como BETO, E5, o T5, ya sea de forma local o descargados desde Hugging Face.
- **Enriquecimiento → Embeddings y clustering:** Los textos enriquecidos se transforman en vectores utilizando SentenceTransformers y fastText, y se agrupan mediante algoritmos como HDBSCAN y reducción de dimensionalidad (UMAP).
- **Clustering → Visualización:** Finalmente, los resultados son representados en gráficos estáticos y reportes PDF generados con herramientas como matplotlib, Plotly, pandas y Jinja2.

Aunque el sistema hace uso intensivo de librerías de código abierto, estas no constituyen interfaces con sistemas externos, sino dependencias internas gestionadas como paquetes dentro del entorno local o virtual del proyecto (por ejemplo, vía pip o conda). No se consumen APIs externas, ni se establece comunicación en tiempo real con servicios web de terceros.

Este diseño modular y desacoplado permite mantener un alto grado de interoperabilidad y facilita la posibilidad de sustituir o extender cada módulo de forma independiente, sin afectar la funcionalidad del sistema completo.

2.1.2 Interfaces con el usuario

El sistema propuesto no cuenta con una interfaz gráfica pública ni con una aplicación de tipo portal web. En su lugar, la interacción con los usuarios se realizará por medio de interfaces técnicas de consola, notebooks ejecutables, scripts de configuración y reportes estáticos generados en PDF, HTML o gráficos .png/.svg.

A continuación, se describen las interfaces de usuario previstas, junto con sus características técnicas y funcionales:

Terminal o Consola (CLI)

- **Propósito:** Interacción principal de los desarrolladores e investigadores con el sistema.
- **Acciones permitidas:** Ejecución de spiders, procesamiento de datos, entrenamiento de modelos, generación de reportes.
- **Requisitos técnicos:** Acceso a un entorno UNIX-like (Linux/macOS recomendado) o WSL en Windows; Python 3.10+ instalado.
- **Usabilidad:** Requiere conocimientos intermedios en línea de comandos. El entrenamiento estimado para familiarizarse con los comandos básicos es inferior a 1 hora para usuarios técnicos.

Notebooks Jupyter

- **Propósito:** Validación de resultados, visualización exploratoria y pruebas modulares.
- **Acciones permitidas:** Carga de resultados del clustering, análisis gráfico, pruebas de embeddings, revisión de habilidades extraídas.
- **Requisitos técnicos:** Instalación local de jupyterlab o uso en plataforma cloud (ej. Google Colab). Se recomienda resolución mínima de 1366x768 para una visualización óptima.
- **Usabilidad:** Las notebooks están documentadas con celdas de texto y ejemplos reproducibles. Se espera un nivel básico de familiaridad con Python y Pandas por parte del usuario.

Visualizaciones estáticas y reportes

- **Propósito:** Consulta de resultados finales en formato visual o tabular, por parte del equipo académico, directivo o institucional.
- **Tipos de salida:** Archivos .pdf, .png, .html o .md generados automáticamente desde scripts Python.

- **Requisitos técnicos:** Visualizador de PDF o navegador moderno actualizado (Chrome, Firefox, Edge). No se requiere conexión a internet tras la generación del archivo.
- **Usabilidad:** Interfaz pasiva. Los reportes están diseñados para facilitar la interpretación con títulos, leyendas y estructura clara.

2.2 Funciones del Producto

El sistema Observatorio de Demanda Laboral en Tecnología en Latinoamérica debe cumplir con una serie de funciones esenciales que permiten cubrir el ciclo completo de análisis automatizado de vacantes laborales. A continuación, se enumeran las principales funciones del producto:

1. **Extracción automatizada de vacantes**
   - Scraping periódico y configurable desde portales de empleo en español como Computrabajo, elempleo.com, Bumeran y LinkedIn.
   - Filtros por país, cargo, modalidad, sector y fecha.
   - Recolección estructurada de atributos como título del cargo, empresa, ubicación, modalidad, tecnologías mencionadas, y descripción completa.
2. **Preprocesamiento y limpieza textual**
   - Tokenización, lematización y eliminación de ruido.
   - Normalización de formatos y campos clave.
   - Generación de bigramas y trigramas relevantes.
3. **Extracción de habilidades (explícitas e implícitas)**
   - Detección mediante patrones regulares y NER.
   - Enriquecimiento con LLMs multilingües para capturar habilidades implícitas y sinónimos contextuales.
   - Clasificación en taxonomías como ESCO y CIUO.
4. **Vectorización y representación semántica**
   - Embeddings mediante modelos como BETO, fastText y E5.
   - Reducción de dimensionalidad con UMAP para visualización y agrupación.
5. **Clustering de perfiles laborales**
   - Agrupamiento no supervisado mediante HDBSCAN.

- Evaluación con métricas de coherencia semántica y Silhouette Score.
- Identificación de segmentos funcionales de demanda tecnológica.

6. **Visualización y reporte**
   - Generación de visualizaciones estáticas y reportes analíticos en CSV, PDF o HTML.
   - Segmentación por país, portal y categoría tecnológica.
7. **Gestión y trazabilidad**
   - Registro de logs, validaciones y errores.
   - Documentación técnica accesible para usuarios académicos y replicadores.
   - Modularidad para adaptarse a nuevos portales o países.

2.3 Características de los Usuarios

El sistema está diseñado para ser utilizado por distintos tipos de usuarios, cuyas características varían según su nivel de acceso, rol funcional, conocimientos técnicos y frecuencia de interacción. A continuación se describen las principales clases de usuario previstas:

Característica	Descripción
Nivel de seguridad o privilegios	Se establecen dos niveles principales de acceso: ▪ **Administrador:** acceso completo al sistema, incluyendo configuración, ejecución y ajustes internos. ▪ **Analista:** acceso restringido a resultados, reportes y visualizaciones, sin modificar parámetros o lógica del sistema. ▪ Opcionalmente, se contempla un perfil de validador externo con acceso de solo lectura.
Roles	**Administrador técnico:** configura scraping, define filtros, ajusta modelos, ejecuta el pipeline completo. **Investigador/Analista:** accede a resultados, visualiza reportes y comunica hallazgos sin intervenir en el procesamiento. **Validador externo (opcional):** revisa calidad de extracción, validación semántica o consistencia de resultados con fines académicos o de control.
Nivel de estudios o experiencia técnica	El administrador debe tener formación en ingeniería, ciencia de datos o afines, con conocimientos sólidos en Python, NLP y manejo de entornos técnicos. El analista requiere competencias básicas en análisis de datos, interpretación de visualizaciones y lectura de reportes técnicos. El validador puede ser un docente, evaluador o experto externo sin conocimientos técnicos detallados.
Frecuencia de uso	El administrador accede típicamente una vez por semana o cuando se requiere actualizar scraping, modelos o ejecutar el sistema completo. El analista accede cada dos semanas o mensualmente para consultar resultados y generar reportes. El validador accede de forma puntual, en contextos de auditoría, revisión académica o validación de resultados.

Tabla 2.1: Características de los usuarios del sistema

2.4 Restricciones

El sistema propuesto presenta un conjunto de restricciones que condicionan su diseño, implementación y despliegue. Estas restricciones se clasifican en tres categorías principales: generales, de software y de hardware.

2.4.1 Restricciones Generales

- **Alcance técnico-académico:** El sistema está diseñado con fines de investigación académica y validación técnica, no para uso comercial o despliegue masivo.
- **Idioma:** Todo el procesamiento textual está orientado a ofertas laborales en español, aunque se considera una posible presencia de términos en inglés.
- **Tolerancia a fallos:** Si bien el sistema cuenta con mecanismos de validación de datos y manejo básico de errores, no se implementarán estrategias avanzadas de tolerancia a fallos o disponibilidad continua.
- **Ejecución modular secuencial:** Las fases del sistema se ejecutarán en orden secuencial, sin requerimientos de paralelismo ni concurrencia en su versión inicial.

2.4.2 Restricciones de Software

- **Versión de Python:** Python 3.10 o superior es requerido para compatibilidad con todas las librerías utilizadas.
- **Dependencia de librerías específicas:** El sistema depende del uso de bibliotecas especializadas:
  - **Web Scraping:** Scrapy, BeautifulSoup, Selenium
  - **NLP:** spaCy (es_core_news_sm), transformers, SentenceTransformers
  - **Embeddings:** intfloat/multilingual-e5-base (768D)
  - **Búsqueda vectorial:** FAISS (Facebook AI Similarity Search)
  - **Matching:** fuzzywuzzy (fuzzy string matching)
  - **Clustering:** HDBSCAN, UMAP
  - **Base de datos:** psycopg2 (PostgreSQL driver)
  - **Data processing:** pandas, numpy
  - **Visualización:** matplotlib, plotly, seaborn
- **Base de datos:** PostgreSQL 13 o superior con soporte para arrays de tipo REAL[] para almacenamiento de embeddings de 768 dimensiones.
- **Modelo de embeddings:** El modelo debe ser descargado desde Hugging Face (tamaño aproximado: 1.1 GB).
- **Índice FAISS:** El archivo (41.41 MB) y su mapping asociado (545 KB) deben estar disponibles en .

- **Aceleración GPU (opcional):** CUDA/cuDNN para aceleración de generación de embeddings. El sistema es funcional con CPU pero 25x más lento.
- **Sistema operativo preferido:** Linux (recomendado), Windows 11 con WSL2, o macOS Monterey o superior.
- **Licencias de uso:** Se restringe el uso del sistema a entornos académicos bajo licencias open source o de uso investigativo.

2.4.3 Restricciones de Hardware

- **Requisitos mínimos para Fase 0 y 1:**
  - CPU: 4 núcleos (Intel i5, AMD Ryzen 5, o Apple M1 o superior)
  - RAM: 8 GB mínimo (16 GB recomendado)
  - Almacenamiento: 15 GB libres (base de datos, modelos, índices)
  - Resolución de pantalla: 1280x800 mínimo
- **Requisitos para generación de embeddings (Fase 0):**
  - **Con GPU:** NVIDIA con soporte CUDA 11.0+, 6 GB VRAM mínimo. Velocidad: 721 skills/segundo
  - **Sin GPU (CPU):** Intel i5/i7 o equivalente. Velocidad: 30 skills/segundo (25x más lento)
  - **Apple Silicon (MPS):** M1/M2/M3 con aceleración Metal. Velocidad: 400 skills/segundo
- **Requisitos para clustering (Módulo 6 - futuro):**
  - RAM: 16 GB mínimo (32 GB recomendado para 23K jobs)
  - HDBSCAN con 23,188 jobs requiere 8 GB RAM disponible
  - UMAP requiere 4 GB RAM para reducción de dimensionalidad
- **Almacenamiento de base de datos:**
  - PostgreSQL con 23,188 jobs: 2 GB
  - Tabla skill_embeddings (14,133 x 768D): 400 MB
  - Índices y tablas auxiliares: 500 MB
  - Total estimado: 3 GB para datos + 12 GB para modelos
- **Acceso a internet:** Se requiere conexión estable para:
  - Descarga inicial de modelos (intfloat/multilingual-e5-base: 1.1 GB)
  - Instalación de librerías Python (2 GB total)
  - Web scraping de portales de empleo (bandwidth: 10 MB/hora)

2.5 Supuestos y Dependencias

Esta sección enumera factores externos y condiciones asumidas que podrían afectar el cumplimiento de los requerimientos definidos en la Sección 3.

2.5.1 Suposiciones

- Se mantendrá acceso público y sin restricciones críticas a los portales de empleo definidos como fuente principal.
- Las estructuras HTML de dichas páginas no sufrirán cambios drásticos que impidan la funcionalidad de los spiders desarrollados.
- Se podrá ejecutar scraping bajo prácticas éticas, sin infringir condiciones explícitas de uso ni requerir autenticación compleja.
- Existirá conectividad a internet estable durante las fases de extracción y enriquecimiento de datos.
- Los modelos de lenguaje en español seleccionados estarán disponibles públicamente para descarga y uso local.
- Los LLMs empleados podrán ejecutarse en entornos locales de cómputo, sin requerir acceso continuo a APIs externas.
- Se contará con PostgreSQL funcional y correctamente configurado desde las fases iniciales del desarrollo.
- El equipo de desarrollo tendrá acceso constante al repositorio de código y a los entornos colaborativos definidos.
- El hardware utilizado por los desarrolladores cumple con los requerimientos mínimos establecidos.

2.5.2 Dependencias

- **Velocidad de conexión a internet:** Afecta directamente los tiempos de scraping, descarga de modelos y ejecución de procesos remotos.
- **Disponibilidad y estabilidad de bibliotecas externas:** El sistema depende de librerías que podrían modificar sus versiones o comportamiento.
- **Funcionamiento adecuado del motor de base de datos:** La persistencia de datos depende de una base PostgreSQL operativa.

- **Funcionamiento de entornos de ejecución local:** Algunos modelos requerirán entornos específicos (compatibilidad CUDA, soporte de arquitectura x64).
- **Factores legales o contractuales externos:** Cambios en políticas de los portales web o en los lineamientos éticos institucionales podrían limitar la continuidad del scraping.
- **Capacidad de procesamiento local:** La ejecución de embeddings y clustering depende de la disponibilidad de memoria RAM suficiente y compatibilidad GPU.

REQUERIMIENTOS ESPECÍFICOS

Este capítulo detalla de manera exhaustiva los requerimientos funcionales y no funcionales del sistema, organizados según las categorías definidas en las secciones anteriores.

3.1 Requerimientos de Interfaces Externas

3.1.1 Interfaces con el Usuario

REI-01 El sistema debe permitir la ejecución de scripts desde CLI para scraping y visualización.

- **Prioridad:** Alta
- **Módulo:** Interfaz Usuario
- **Criterio:** Ejecución funcional por línea de comandos

REI-02 El sistema debe incluir notebooks Jupyter con celdas documentadas.

- **Prioridad:** Media
- **Módulo:** Interfaz Usuario
- **Criterio:** Visualización y reproducción de notebooks

REI-03 El sistema debe generar reportes en formato PDF, PNG y HTML.

- **Prioridad:** Alta
- **Módulo:** Interfaz Usuario/Visualización
- **Criterio:** Visualización correcta de reportes generados

3.1.2 Interfaces con el Hardware

REI-04 El sistema debe operar en equipos sin GPU dedicada y mínimo 8 GB de RAM.

- **Prioridad:** Alta
- **Módulo:** Interfaz Hardware
- **Criterio:** Ejecución sin errores en equipos personales

3.1.3 Interfaces con el Software

REI-05 El sistema debe conectarse a una base de datos PostgreSQL local.

- **Prioridad:** Alta
- **Módulo:** Interfaz Software
- **Criterio:** Inserción y lectura desde PostgreSQL

REI-06 El sistema debe acceder a portales web por HTTP/HTTPS para extraer datos.

- **Prioridad:** Alta
- **Módulo:** Interfaz Comunicación
- **Criterio:** Scraping exitoso desde URLs definidas

3.2 Requerimientos Funcionales

3.2.1 Funcionalidad 1: Extracción de Vacantes (Scraping)

RF-01 El sistema debe extraer vacantes desde portales como Computrabajo, Bumeran y elempleo.com.

- **Prioridad:** Alta
- **Módulo:** Scraping
- **Criterio:** Extracción visible y consistente de datos

3.2.2 Funcionalidad 2: Procesamiento de Texto

RF-02 El sistema debe almacenar las vacantes en PostgreSQL con deduplicación SHA256.

- **Prioridad:** Alta
- **Módulo:** Almacenamiento
- **Criterio:** Consultas e inserciones validadas, duplicados detectados
- **Estado:** IMPLEMENTADO (23,352 jobs, dedup rate 0.5 %)

RF-03 El sistema debe preprocesar el texto: limpieza HTML, tokenización, normalización.

- **Prioridad:** Alta
- **Módulo:** Procesamiento NLP
- **Criterio:** Verificación de campos procesados, detección de jobs basura
- **Estado:** IMPLEMENTADO (23,188 jobs limpios, 99.5 % usable)

RF-04 El sistema debe extraer habilidades explícitas mediante NER y Regex (Pipeline A).

- **Prioridad:** Alta
- **Módulo:** Extracción
- **Criterio:** Skills extraídas con método y confianza asociados
- **Métodos:**
  - Regex: 200+ patrones tecnológicos (78-89 % precision)
  - NER: spaCy + custom entity ruler (13 % precision con filtros)
- **Estado:** IMPLEMENTADO (Test 100 jobs: 2,756 skills extraídas)

3.2.3 Funcionalidad 2.1: Mapeo contra Taxonomía ESCO

RF-04.1 El sistema debe cargar y mantener una taxonomía unificada de 14,174 skills.

- **Prioridad:** Alta
- **Módulo:** Taxonomía (Fase 0)
- **Criterio:** ESCO v1.1.0 (13,939) + O*NET Hot Tech (152) + Manual Curated (83)
- **Estado:** IMPLEMENTADO

RF-04.2 El sistema debe mapear skills extraídas contra ESCO usando estrategia de tres capas.

- **Prioridad:** Alta
- **Módulo:** Matching
- **Criterio:** Layer 1 (Exact, SQL ILIKE, confidence 1.0) $\rightarrow$ Layer 2 (Fuzzy, threshold 0.92) $\rightarrow$ Layer 3 (Semantic, threshold 0.87, DESHABILITADO)
- **Estado:** PARCIALMENTE IMPLEMENTADO (Layer 1 + 2 activos, Layer 3 disabled)
- **Match rate actual:** 12.6 % (esperado para taxonomías 2016-2017)

RF-04.3 El sistema debe identificar y rastrear skills emergentes no mapeadas.

- **Prioridad:** Alta
- **Módulo:** Emergent Skills Tracking
- **Criterio:** Skills sin match ESCO almacenadas con frecuencia y contexto
- **Estado:** IMPLEMENTADO (87.4 % emergent rate en test de 100 jobs)

RF-04.4 El sistema debe generar embeddings semánticos para todas las skills.

- **Prioridad:** Alta
- **Módulo:** Embeddings (Fase 0)
- **Criterio:** Modelo intfloat/multilingual-e5-base (768D), L2-normalized

- **Performance:** 721 skills/segundo (GPU), 30 skills/segundo (CPU)
- **Estado:** IMPLEMENTADO (14,133 embeddings, 94.6 % test pass)

RF-04.5 El sistema debe construir y mantener índice FAISS para búsqueda semántica.

- **Prioridad:** Alta
- **Módulo:** FAISS Index (Fase 0)
- **Criterio:** IndexFlatIP (exact search), 30,147 queries/segundo
- **Archivos:** data/embeddings/esco.faiss (41.41 MB), esco_mapping.pkl (545 KB)
- **Estado:** IMPLEMENTADO (25x más rápido que PostgreSQL pgvector)

3.2.4 Funcionalidad 3: Representación Semántica

RF-05 El sistema debe generar representaciones semánticas (embeddings) y realizar clustering automático.

- **Prioridad:** Alta
- **Módulo:** Agrupamiento
- **Criterio:** Clústeres coherentes generados

3.2.5 Funcionalidad 4: Visualización

RF-06 El sistema debe generar visualizaciones estáticas con gráficas de frecuencia de habilidades y comparativas.

- **Prioridad:** Alta
- **Módulo:** Visualización
- **Criterio:** Visualizaciones exportadas exitosamente

3.3 Requerimientos de Desempeño

RD-01 El sistema debe extraer y almacenar vacantes desde múltiples portales de empleo.

- **Prioridad:** Alta
- **Módulo:** Scraping
- **Criterio:** Mínimo 300 vacantes por país desde dos portales distintos
- **Performance actual:** 23,352 jobs scraped (hiring.cafe: 23,313, elempleo: 38, zonajobs: 1)
- **Estado:** CUMPLIDO (7,784 % sobre mínimo requerido)

RD-02 El preprocesamiento textual debe ejecutarse sin errores críticos.

- **Prioridad:** Alta
- **Módulo:** Procesamiento
- **Criterio:** Pipeline completado con success rate ¿95 %
- **Performance actual:** 99.5 % usable rate (23,188/23,352 jobs)
- **Estado:** CUMPLIDO

RD-03 La generación de embeddings debe completarse en tiempo razonable.

- **Prioridad:** Alta
- **Módulo:** Embeddings (Fase 0)
- **Criterio:** Completar 14K skills en ¡5 minutos con GPU
- **Performance actual:** 19.65 segundos para 14,133 skills (721 skills/seg)
- **Estado:** CUMPLIDO (15x mejor que requerimiento)

RD-04 La búsqueda semántica con FAISS debe ser eficiente.

- **Prioridad:** Alta
- **Módulo:** FAISS Index
- **Criterio:** Mínimo 100 queries por segundo
- **Performance actual:** 30,147 queries/segundo (301x sobre requerimiento)
- **Estado:** CUMPLIDO (25x más rápido que PostgreSQL pgvector)

RD-05 El sistema de extracción de skills debe procesar jobs sin fallos.

- **Prioridad:** Alta
- **Módulo:** Extracción Pipeline A
- **Criterio:** Success rate ¿95 % en procesamiento
- **Performance actual:** 100 % success rate en test de 100 jobs (2,756 skills extraídas)
- **Tiempo promedio:** 1.82 segundos por job
- **Estado:** CUMPLIDO

RD-06 El matching contra ESCO debe completarse para todas las skills extraídas.

- **Prioridad:** Alta
- **Módulo:** Matching (3-layer strategy)
- **Criterio:** Todas las skills procesadas por las 3 layers

- **Performance actual:** 12.6 % match rate (Layer 1: 5.4 %, Layer 2: 7.1 %, Layer 3: disabled)
- **Estado:** PARCIALMENTE CUMPLIDO (Layer 3 deshabilitado temporalmente)
- **Nota:** 87.4 % emergent skills es esperado para taxonomías 2016-2017

3.4 Restricciones de Diseño

RDZ-01 El sistema debe ejecutarse completamente en local, sin servicios web de pago.

- **Prioridad:** Alta
- **Módulo:** Arquitectura General
- **Criterio:** Funciona sin acceso a servicios externos

RDZ-02 La ejecución de modelos LLM se limitará a versiones descargables.

- **Prioridad:** Alta
- **Módulo:** Enriquecimiento
- **Criterio:** Modelos configurados desde Hugging Face

RDZ-03 El desarrollo deberá realizarse en Python 3.10 o superior.

- **Prioridad:** Alta
- **Módulo:** Infraestructura
- **Criterio:** Repositorio sin dependencias privativas

RDZ-04 No se implementará una interfaz gráfica interactiva.

- **Prioridad:** Alta
- **Módulo:** Visualización
- **Criterio:** El sistema exporta reportes, no tiene frontend

RDZ-05 El sistema debe ser modular.

- **Prioridad:** Alta
- **Módulo:** Arquitectura General
- **Criterio:** Cada módulo puede lanzarse individualmente

3.5 Atributos del Sistema de Software (No funcionales)

3.5.1 Confiabilidad

NFA-01 El sistema debe producir resultados consistentes ante entradas iguales.

- **Prioridad:** Alta
- **Módulo:** Todos
- **Criterio:** Repetición del proceso genera mismos resultados

NFA-02 Se debe registrar el comportamiento del sistema mediante logs detallados.

- **Prioridad:** Alta
- **Módulo:** Todos
- **Criterio:** Archivos de log por módulo

NFA-03 En caso de interrupciones, los módulos deben permitir ser reiniciados.

- **Prioridad:** Alta
- **Módulo:** Arquitectura General
- **Criterio:** Pipeline puede reiniciarse parcialmente

3.5.2 Disponibilidad

NFA-04 El sistema estará disponible para ejecución local en cualquier momento.

- **Prioridad:** Alta
- **Módulo:** Infraestructura
- **Criterio:** Pipeline completo corre offline

NFA-05 Scripts y notebooks deben estar organizados en GitHub.

- **Prioridad:** Alta
- **Módulo:** Infraestructura
- **Criterio:** Repositorio contiene notebooks funcionales

3.5.3 Seguridad

NFA-06 Se evitará recolectar información personal.

- **Prioridad:** Alta
- **Módulo:** Scraping

- **Criterio:** Ningún campo sensible almacenado

NFA-07 Los spiders implementarán throttling.

- **Prioridad:** Alta
- **Módulo:** Scraping
- **Criterio:** Tiempo entre requests configurable

NFA-08 El código incluirá controles básicos de errores.

- **Prioridad:** Alta
- **Módulo:** Todos
- **Criterio:** Pipeline continúa sin detenerse

3.5.4 Mantenibilidad

NFA-09 El sistema debe estar documentado a nivel de código.

- **Prioridad:** Alta
- **Módulo:** Todos
- **Criterio:** Documentación presente en repositorio

NFA-10 Cada módulo será independiente y versionado.

- **Prioridad:** Alta
- **Módulo:** Arquitectura Modular
- **Criterio:** Se puede actualizar un módulo sin afectar demás

3.5.5 Portabilidad

NFA-11 El sistema debe poder ejecutarse en Linux, macOS o Windows con WSL.

- **Prioridad:** Alta
- **Módulo:** Infraestructura
- **Criterio:** Se ejecuta en tres sistemas operativos

NFA-12 Se debe proporcionar un archivo de entorno reproducible.

- **Prioridad:** Alta
- **Módulo:** Infraestructura
- **Criterio:** Archivo requirements.txt ejecutado con éxito

3.6 Requerimientos de la Base de Datos

BD-01 El sistema debe utilizar PostgreSQL como sistema de gestión de base de datos.

- **Prioridad:** Alta
- **Módulo:** Base de Datos
- **Criterio:** PostgreSQL instalado y configurado

BD-02 La base de datos debe permitir el almacenamiento estructurado de vacantes.

- **Prioridad:** Alta
- **Módulo:** Base de Datos
- **Criterio:** Contiene todos los campos definidos

BD-03 Las relaciones deben seguir un modelo normalizado.

- **Prioridad:** Alta
- **Módulo:** Base de Datos
- **Criterio:** Modelo ER respetado

BD-04 La base de datos debe ser accesible desde Python.

- **Prioridad:** Alta
- **Módulo:** Base de Datos
- **Criterio:** Scripts se conectan correctamente

BD-05 El sistema debe poder crear automáticamente las tablas necesarias.

- **Prioridad:** Media
- **Módulo:** Base de Datos
- **Criterio:** Ejecución de init_db.py genera tablas

BD-06 Se debe garantizar la integridad de los datos.

- **Prioridad:** Alta
- **Módulo:** Base de Datos
- **Criterio:** Restricciones UNIQUE y validaciones

BD-07 La base de datos debe permitir exportar resultados.

- **Prioridad:** Alta

- **Módulo:** Base de Datos
- **Criterio:** Exportación a CSV y JSON exitosa

BD-08 La base de datos debe operar localmente.

- **Prioridad:** Alta
- **Módulo:** Base de Datos
- **Criterio:** Conexión vía localhost funcional

PROCESO INGENIERÍA DE REQUERIMIENTOS

La especificación de requerimientos del sistema *Observatorio de Demanda Laboral en Tecnología en Latinoamérica* fue construida mediante un proceso iterativo, sistemático y colaborativo, guiado por buenas prácticas de ingeniería de software y ajustado al contexto académico y técnico del proyecto.

4.1 Técnicas y Métodos Utilizados

Para el levantamiento, análisis y validación de los requerimientos, se emplearon múltiples técnicas complementarias:

- **Análisis de proyectos similares y literatura técnica:** Se revisaron más de una docena de fuentes académicas, artículos internacionales, repositorios en GitHub y proyectos institucionales.
- **Revisión documental interna:** Documentos clave como la Propuesta de Grado y el SPMP sirvieron como base estructural.
- **Descomposición funcional por casos de uso:** Se identificaron 12 casos de uso principales, cada uno articulado con uno o más módulos del sistema.
- **Sesiones de lluvia de ideas y revisión cruzada:** El equipo realizó sesiones colaborativas utilizando Google Docs y Notion.
- **Validación iterativa con el director del proyecto:** El borrador fue sometido a retroalimentación para mejorar redacción y consistencia.

4.2 Trazabilidad y Consistencia

Para garantizar la trazabilidad, cada requerimiento fue identificado mediante un código único según su tipo:

- RFxx: Requerimiento Funcional
- RNFxx / NFAxx: Requerimiento No Funcional
- RDxx: Requerimiento de Desempeño
- RDZxx: Restricción de Diseño

- BDxx: Requerimiento de Base de Datos

Cada uno fue mapeado explícitamente a uno o más casos de uso y módulos funcionales, estableciendo una relación directa entre la funcionalidad planteada, su implementación esperada y su criterio de verificación.

Se procuró que todos los requerimientos cumplieran con las siguientes características esenciales:

- Atómicos
- Correctos y completos
- No ambiguos y consistentes
- Verificables
- Modificables
- Trazables
- Priorizados

4.3 Complementariedad con el SPMP

Este proceso está alineado y complementa lo definido en la sección 6.3 del SPMP (Control de Requerimientos), donde se establece un mecanismo estructurado para la gestión de cambios. Cualquier modificación a los requerimientos deberá:

- Ser documentada mediante una solicitud de cambio formal
- Incluir un análisis de impacto
- Actualizarse en la tabla de distribución de requerimientos
- Ser validada por el equipo completo y el director

PROCESO VERIFICACIÓN

El proceso de Verificación y Validación (V&V) aplicado a esta Especificación de Requerimientos de Software (SRS) tiene como propósito asegurar que el documento refleja de forma precisa, clara y completa las necesidades funcionales y no funcionales del sistema.

5.1 Verificación del Documento SRS

Durante la construcción del SRS, se aplicaron las siguientes técnicas de verificación documental:

- **Revisión cruzada interna:** Cada sección fue redactada por un integrante del equipo y verificada por otro.
- **Validación contra SPMP y Propuesta de Grado:** Se confirmó que todos los requerimientos respondieron al propósito, alcance y restricciones previamente acordados.
- **Construcción de matriz de trazabilidad preliminar:** Se elaboró una tabla que relaciona cada requerimiento con los casos de uso y módulos funcionales.

5.2 Verificación de Requerimientos Individuales

Cada requerimiento especificado será evaluado mediante al menos uno de los siguientes métodos de verificación:

Método	Descripción	Aplicación en el proyecto
Inspección	Lectura técnica sistemática	Aplicada a todos los requerimientos durante la redacción
Análisis	Evaluación de consistencia lógica	Usada para requerimientos no funcionales
Prueba	Ejecución de scripts o notebooks	Requerimientos funcionales como scraping, extracción, clustering
Demostración	Presentación empírica del sistema	Generación de reportes con datos reales

Tabla 5.1: Métodos de verificación de requerimientos

5.3 Validación del Sistema Completo

La validación del sistema contempla un enfoque integral:

- **Pruebas de extremo a extremo (E2E):** Simulación completa desde el scraping hasta la generación de reportes.
- **Casos de prueba por módulo:** Cada componente tendrá pruebas específicas.
- **Evaluación cualitativa de resultados semánticos:** Validación cruzada y revisión manual.
- **Validación externa:** Presentación de resultados al director y jurados.
- **Checklist de atributos no funcionales:** Verificación de confiabilidad, mantenibilidad, portabilidad, seguridad, disponibilidad y desempeño.

5.4 Criterios de Aprobación

Un requerimiento se considerará verificado y validado cuando cumpla con todos los siguientes criterios:

- Su cumplimiento puede demostrarse mediante al menos uno de los métodos de V&V
- Existe evidencia objetiva documentada
- No presenta ambigüedad ni contradicción
- Ha sido validado por al menos un miembro del equipo y aceptado en revisión final por el director

ANEXOS

Diagrama de Casos de Uso

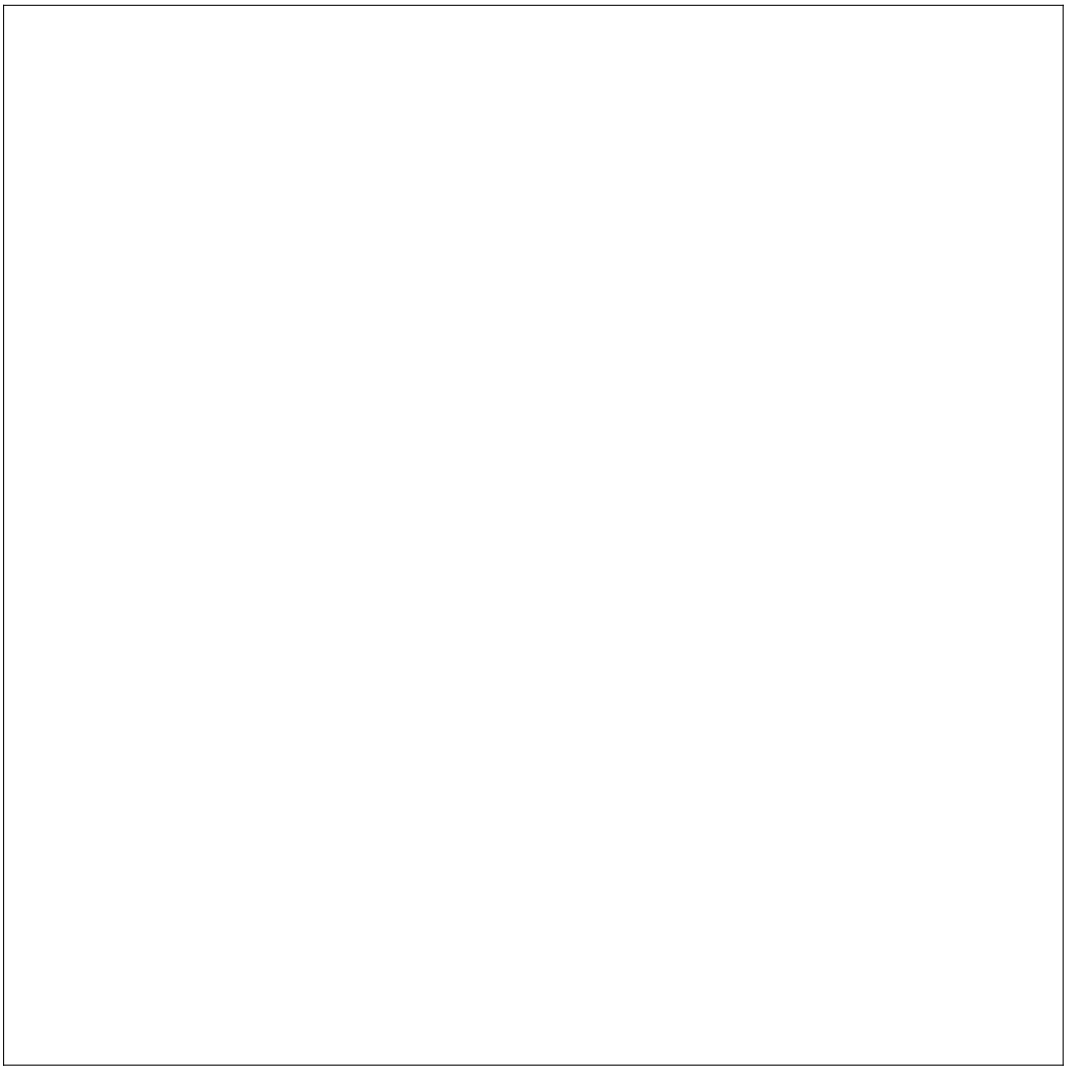

Figura 5.1: Diagrama de casos de uso del sistema

Diagrama Modelo de Dominio

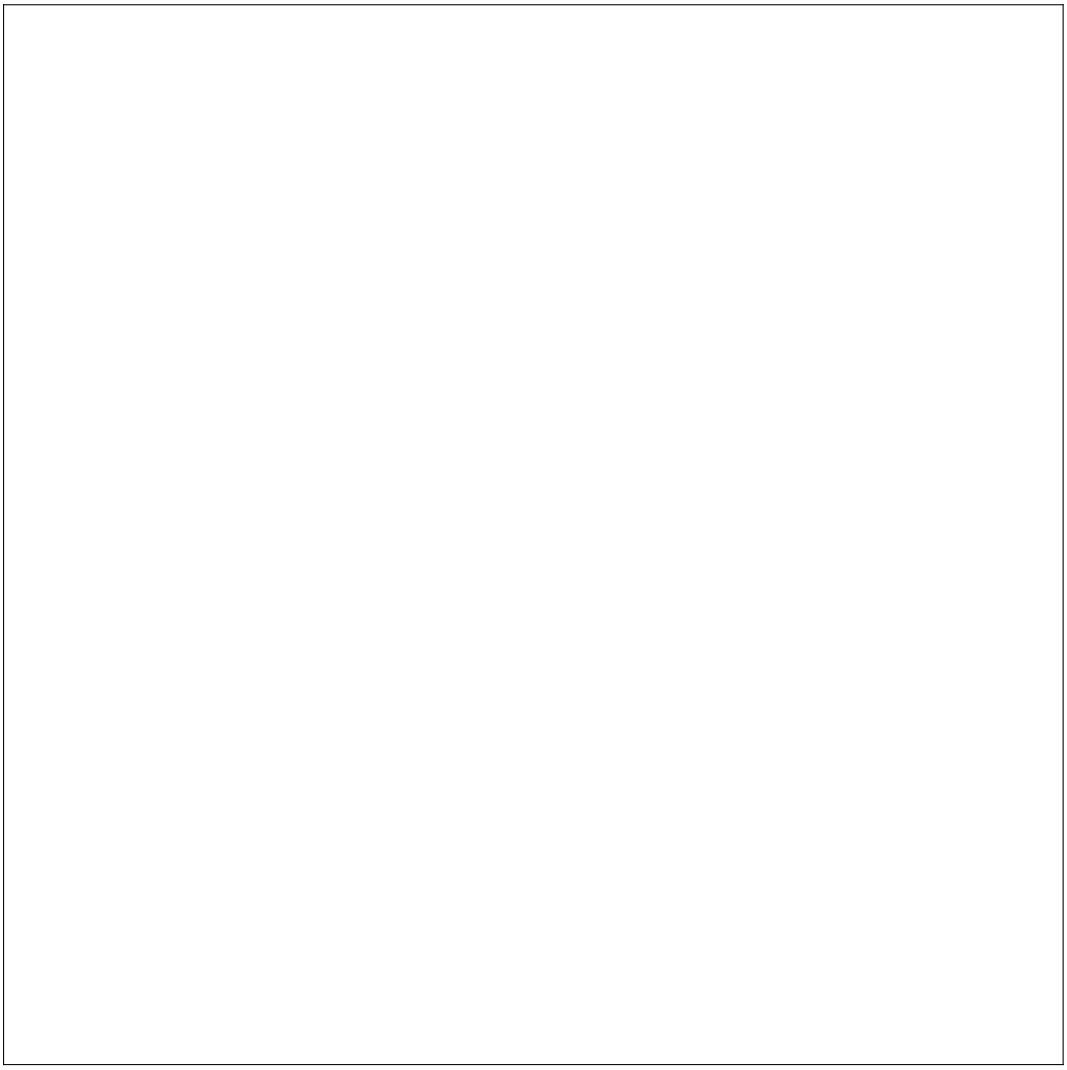

Figura 5.2: Diagrama del modelo de dominio del sistema

REFERENCIAS

1. Orozco Puello & Gómez Estrada, L. F. (2019). *Proyecto de Grado II – Web Scraping* [Trabajo de grado, Universidad del Sinú Elías Bechara Zainúm].

2. Rubio Arrubla, J. A. (2024). *Demanda de habilidades tecnológicas: evidencia desde el mercado laboral colombiano* [Tesis de maestría, Universidad de los Andes].

3. Lukauskas, M., Šarkauskaitė, V., Pilinkienė, V., & Stundziene, A. (2023). Enhancing Skills Demand Understanding through Job Ad Segmentation Using NLP and Clustering Techniques. ResearchGate.

4. Martínez Sánchez, J. C. (2024). Desajuste en el mercado laboral: análisis de los perfiles de candidatos y las ofertas de trabajo publicadas en internet. *Revista del INEGI*, (44).

5. Cárdenas Rubio, J. A., Guataquí Roa, J. C., & Montaña Doncel, J. M. (2015). *Metodología para el análisis de demanda laboral mediante datos de internet: caso Colombia*. Organización Internacional del Trabajo / CINTERFOR.

6. Campos-Vázquez, R. M., & Martínez Sánchez, J. C. (2024). Skills sought by companies in the Mexican labor market: An analysis of online job vacancies. *Estudios Económicos De El Colegio De México*, 39(2), 243–278.

7. Aguilera, S. O., & Méndez, R. E. (2018). ¿Qué buscan los que buscan? Análisis De mercado laboral IT en Argentina. *Revista Perspectivas*, 1(1), 15–30.

8. Nguyen K., Zhang M., Montariol S., & Bosselut A. (2024). Rethinking Skill Extraction in the Job Market Domain using Large Language Models. En *Proceedings of the First Workshop on Natural Language Processing for Human Resources (NLP4HR 2024)* (pp. 27–42).

9. Echeverría L., & Rucci G. (2022). *¿Qué suma la ciencia de datos a la identificación y anticipación de la demanda de habilidades?* Banco Interamericano de Desarrollo.

10. Kavas H., Serra-Vidal M., & Wanner L. (2025). Multilingual Skill Extraction for Job Vacancy–Job Seeker Matching in Knowledge Graphs. En *Proceedings of the Workshop on Generative AI and Knowledge Graphs (GenAIK)*.