Propuesta de Trabajo Fin de Máster
Diseño, Desarrollo e Implementación de un Sistema Serverless de Microservicios de Inteligencia Artificial

josé Javier Gutiérrez Gil jogugil@gmail.com

24 de octubre de 2025

Resumen

Esta propuesta describe el desarrollo de un sistema que integra tecnologías modernas para el despliegue de servicios inteligentes con alta flexibilidad y escalabilidad. El objetivo principal del proyecto TFM es diseñar e implementar un sistema basado en microservicios de inteligencia artificial (IA) utilizando un enfoque serverless. El backend se implementará en Rust, mientras que el frontend será desarrollado con TypeScript, React y Vite, y se integrará una API REST para la comunicación entre los componentes.

Los microservicios iniciales estarán enfocados en tareas como procesamiento de lenguaje natural (PLN) y análisis de imágenes, proporcionando servicios sencillos pero extensibles que demostrarán la viabilidad de un sistema serverless como base para la explotación de IA. Sin embargo, se reconoce que el enfoque serverless, aunque muy útil en ciertos escenarios, presenta limitaciones cuando se requieren tareas más complejas o entrenamientos intensivos.

En este sentido, el proyecto evaluará la integración de un enfoque híbrido que combine las ventajas del modelo serverless con la necesidad de microservicios más convencionales para tareas de mayor complejidad, como entrenamientos de modelos avanzados o procesos que no se ajustan bien a las restricciones del entorno serverless.

Además, se integrará el concepto de Model-as-a-Service (MAAS), permitiendo no solo acceder a modelos preentrenados, sino también personalizarlos y evolucionarlos a medida que el usuario interactúa con ellos. Este enfoque busca ofrecer una experiencia adaptada a los datos, contextos e interacciones de los usuarios, mejorando así la relevancia y precisión de los modelos con el tiempo.

Finalmente, se llevará a cabo un estudio sobre la viabilidad de sistemas serverless para usuarios finales de aplicaciones y servicios de inteligencia artificial, evaluando la eficiencia y escalabilidad del sistema, y cómo el modelo serverless, combinado con MAAS, puede democratizar el acceso a servicios avanzados de IA sin la necesidad de gestionar infraestructura. Introducción

1. Introducción

La expansión de las aplicaciones basadas en inteligencia artificial (IA) ha generado una creciente demanda de soluciones flexibles y escalables para el despliegue eficiente de

servicios inteligentes. En este contexto, el enfoque serverless surge como una alternativa eficaz, ya que abstrae la gestión de infraestructura, facilitando el desarrollo y garantizando la escalabilidad automática de los servicios. Este proyecto tiene como objetivo diseñar e implementar un sistema serverless que pueda operar tanto en aplicaciones móviles como web, proporcionando microservicios de IA accesibles, personalizables y adaptados a las necesidades específicas de los usuarios.

El sistema será diseñado para crear un entorno virtual seguro y personalizado, donde los usuarios autenticados solo podrán interactuar con los servicios de IA a los que se hayan registrado. Este enfoque no solo mejora la experiencia del usuario, sino que también refuerza la seguridad al restringir el acceso no autorizado. Los modelos de IA se ejecutarán dentro de contenedores Docker, lo que permitirá a los usuarios aprovechar las capacidades avanzadas de IA sin preocuparse por la gestión de la infraestructura subyacente. Además, la integración con Kubernetes garantizará una escalabilidad automática, facilitando el manejo de picos de demanda, la distribución eficiente de cargas de trabajo y el balanceo de tráfico entre las distintas instancias de los modelos.

Aunque el enfoque serverless presenta ventajas significativas, como la capacidad para gestionar picos de demanda, realizar inferencias ligeras y paralelizar tareas, también tiene limitaciones en escenarios que requieren entrenamiento intensivo, procesamiento de modelos complejos o tareas prolongadas. Por ejemplo, para tareas de inferencia ligera basadas en modelos preentrenados, el sistema serverless puede procesar datos rápidamente y proporcionar resultados inmediatos. Sin embargo, cuando se trata de entrenar modelos complejos, como WaveNet o Tacotron 2, que requieren recursos intensivos y GPUs, el enfoque serverless no es ideal debido a las restricciones en tiempo de ejecución y capacidad de procesamiento.

Este proyecto también integra un enfoque innovador basado en MAAS (Model-as-a-Service), permitiendo que los usuarios no solo accedan a modelos de IA preentrenados, sino que también puedan evolucionar y personalizar estos modelos según sus datos, contextos e interacciones. Inicialmente, el sistema ofrecerá modelos compartidos para todos los usuarios, permitiendo un punto de partida común. Sin embargo, a medida que los usuarios interactúan con los modelos y realizan entrenamientos adicionales, el sistema generará versiones personalizadas que se ajusten a sus necesidades específicas, mejorando así la precisión y relevancia de las funcionalidades ofrecidas. Además, el sistema se encargará de gestionar el ciclo de vida y las versiones de los modelos, asegurando una evolución controlada y consistente.

Para asegurar el buen funcionamiento de esta arquitectura, los modelos se ejecutarán en contenedores Docker y se orquestarán con Kubernetes, garantizando la flexibilidad y la escalabilidad automática en entornos de alta demanda. La integración de MAAS con estas tecnologías permitirá a los usuarios beneficiarse de capacidades avanzadas sin tener que gestionar la infraestructura subyacente.

Dado el vasto campo de la inteligencia artificial, el proyecto incluye un trabajo preliminar clave para enmarcar y clasificar los tipos de modelos y funcionalidades que se implementarán. Esto incluye modelos de aprendizaje supervisado y no supervisado, algoritmos de optimización de hiperparámetros, métodos de resolución de sistemas lineales en contextos algebraicos, entre otros. Este análisis inicial es fundamental para asegurar que la herramienta sea capaz de satisfacer una amplia gama de necesidades, desde tareas generales de inferencia hasta procesos especializados de entrenamiento y optimización.

Por ello, este proyecto propone un entorno serverless combinado con MAAS, con el objetivo de democratizar el acceso y la personalización de servicios de IA. La integración

de tecnologías modernas como Docker y Kubernetes con un enfoque centrado en el usuario permitirá ofrecer servicios eficientes y escalables, además de facilitar la evolución continua de los modelos de IA, adaptándolos a las necesidades de cada usuario. Este enfoque sienta las bases para una herramienta robusta y versátil, capaz de abordar diversos casos de uso en el dinámico campo de la inteligencia artificial.

2. Contexto y Motivación

La inteligencia artificial (IA) ha emergido como una de las tecnologías más disruptivas y prometedoras en la última década, transformando una amplia gama de industrias, desde la salud y la educación hasta el comercio y la automatización de procesos. A medida que el uso de la IA se expande, surge la necesidad de herramientas y plataformas que permitan a los usuarios acceder, entrenar y personalizar modelos de IA de manera eficiente y sin la complejidad de gestionar infraestructuras costosas y complejas. En este contexto, el enfoque serverless ha ganado popularidad, ya que permite a los desarrolladores centrarse en la lógica del negocio sin preocuparse por la gestión de servidores, lo que facilita la escalabilidad y el mantenimiento de los servicios.

Además, la tendencia hacia la personalización de los modelos de IA ha adquirido una gran relevancia, ya que los usuarios buscan soluciones que se adapten a sus contextos específicos, mejorando la precisión y la efectividad de las aplicaciones. A pesar de la disponibilidad de modelos preentrenados, estos no siempre cumplen con los requerimientos específicos de cada usuario. Por lo tanto, surge la necesidad de un entorno flexible que permita la evolución y personalización de estos modelos de acuerdo con las interacciones y los datos generados por los usuarios. Este enfoque, conocido como Model-as-a-Service (MAAS), no solo facilita el acceso a modelos de IA, sino que también permite a los usuarios modificar y ajustar estos modelos, mejorando así su experiencia y utilidad.

El objetivo de este proyecto es integrar el enfoque serverless con MAAS, proporcionando una solución escalable, accesible y adaptable, que no solo permita la ejecución de modelos de IA preentrenados, sino también la personalización y evolución de estos a lo largo del tiempo. Al combinar tecnologías modernas como Docker y Kubernetes, se busca ofrecer un entorno robusto que permita a los usuarios concentrarse en el diseño de soluciones innovadoras sin preocuparse por la infraestructura subyacente. De esta manera, se abre la posibilidad de democratizar el acceso a la inteligencia artificial, facilitando su adopción en una amplia gama de aplicaciones y usuarios, desde desarrolladores experimentados hasta aquellos con menos conocimiento técnico.

El enfoque adoptado en este proyecto responde a una serie de desafíos comunes en la implementación de soluciones basadas en IA, como la gestión de recursos, la escalabilidad y la personalización. Este trabajo busca, por lo tanto, proporcionar una respuesta innovadora a las necesidades actuales, con el fin de potenciar el uso de la inteligencia artificial en aplicaciones móviles y web, mejorando la experiencia del usuario y facilitando el acceso a tecnologías avanzadas de manera más eficiente y accesible.

3. Objetivos

El objetivo principal de este trabajo es demostrar la viabilidad de una arquitectura `serverless` para ofrecer servicios de inteligencia artificial (IA) accesibles a usuarios finales. Para ello, se diseñará, desarrollará e implementará una plataforma que permita a

los usuarios no solo acceder a los servicios a los cuales se hayan registrado, sino también evolucionar sus modelos según los datos proporcionados y las acciones realizadas dentro del sistema.

El proyecto evaluará la mejor manera de permitir a los usuarios realizar un *fine-tuning* sencillo de sus modelos, adaptándolos al contexto y las necesidades específicas de su trabajo. De esta forma, los usuarios podrán ajustar el modelo para tareas como la extracción de información o la generación de resúmenes, y tendrán la posibilidad de mantener versiones personalizadas de sus modelos en el servicio, utilizándolos en cualquier momento según lo necesiten.

El diseño incluirá las siguientes características:

1. Implementar un backend robusto en `Rust` para gestionar la lógica central y los microservicios de la plataforma.

2. Desarrollar un frontend moderno y eficiente utilizando `TypeScript`, `React` y `Vite`, accesible desde dispositivos móviles y web.

3. Crear una API `REST` para conectar el backend con el frontend, permitiendo que los usuarios interactúen con los microservicios de manera sencilla y eficiente.

4. Desplegar microservicios iniciales para tareas de **Procesamiento de Lenguaje Natural (PLN)**, como traducción, resumen de texto y extracción de información, así como para **Procesamiento de Imágenes**, como clasificación básica y detección de objetos.

5. Integrar características de **autoescalado** y **flexibilidad** para adaptar la infraestructura a las necesidades cambiantes de los usuarios.

6. Crear un entorno virtual donde los usuarios registrados solo puedan acceder a los servicios en los que se han registrado, asegurando una experiencia personalizada.

7. Evaluar la viabilidad de los sistemas `serverless` como infraestructura para ofrecer servicios de IA de manera escalable y accesible para los usuarios finales.

8. Si el estudio es positivo, proveer una base sólida para la creación de una futura plataforma basada en esta arquitectura, permitiendo la explotación y evolución de la idea inicial.

Este enfoque no solo permite la personalización de los servicios, sino que también asegura que los modelos de IA puedan adaptarse de forma dinámica a las necesidades y datos específicos de cada usuario, demostrando la capacidad de la arquitectura `serverless` para ofrecer una solución flexible, accesible y de alto rendimiento bajo el enfoque de `MAAS`.

- Desarrollar un sistema capaz de generar resúmenes de documentos legales.

- Implementar el sistema utilizando tecnologías de IA y `serverless`.

- Evaluar el rendimiento del sistema frente a soluciones existentes.

Marco Teórico

4. Serverless IA

Serverless se refiere a un modelo en el que el proveedor de la nube gestiona automáticamente la infraestructura, el escalado y los servidores, permitiendo que los desarrolladores se enfoquen únicamente en el código y la lógica.

4.1. Ventajas

- **Escalabilidad automática:** La infraestructura escala automáticamente según la demanda, ideal para IA con cargas de trabajo impredecibles.
- **Simplicidad:** No es necesario gestionar servidores, actualizaciones ni configuraciones complejas.
- **Costo basado en uso:** Se paga solo por el tiempo de ejecución de las funciones (ideal para inferencias de IA poco frecuentes o irregulares).
- **Despliegue rápido:** Permite implementar modelos de IA como funciones individuales, sin necesidad de configurar un entorno complejo.
- **Integración con servicios en la nube:** Se conecta fácilmente con herramientas de almacenamiento, bases de datos y análisis en plataformas como AWS Lambda, Google Cloud Functions o Azure Functions.

4.2. Casos de uso ideales

- Inferencia de IA a demanda (chatbots, clasificación de imágenes).
- Automatización de tareas con IA (procesamiento de documentos).
- Sistemas con baja latencia y eventos intermitentes.

4.3. Desafíos

- **Límites de ejecución:** No es adecuado para modelos de IA muy grandes o procesos prolongados.
- **Dependencia del proveedor:** Riesgo de bloqueo con un proveedor de nube específico.
- **Personalización limitada:** Menor control sobre la infraestructura subyacente.

5. Microservicios IA

Los microservicios son una arquitectura modular donde cada componente del sistema es una unidad independiente que realiza una función específica. Los microservicios IA dividen funciones de IA en servicios pequeños y autónomos (entrenamiento, inferencia, preprocesamiento).

5.1. Ventajas

- **Control total:** Puedes personalizar la infraestructura, la escalabilidad y las configuraciones según tus necesidades.
- **Independencia funcional:** Cada microservicio se puede actualizar, escalar y desplegar de forma independiente, permitiendo mayor flexibilidad.
- **Desempeño optimizado:** Adecuado para sistemas de IA complejos o de gran escala que necesitan procesamiento intensivo o tiempo prolongado de ejecución.
- **Portabilidad:** Puedes usar contenedores como Docker y orquestadores como Kubernetes para ejecutar microservicios en cualquier entorno (nube, on-premise, híbrido).

5.2. Casos de uso ideales

- Sistemas complejos de IA, como plataformas de recomendación, visión por computadora o procesamiento de lenguaje natural a gran escala.
- Implementaciones que requieren control sobre el hardware (GPUs, TPUs).
- Arquitecturas híbridas que combinan servicios en la nube y locales.

5.3. Desafíos

- **Complejidad:** Requiere un equipo con experiencia en gestión de microservicios y orquestación.
- **Costos iniciales:** Configurar y mantener la infraestructura puede ser costoso en términos de tiempo y recursos.
- **Mantenimiento continuo:** El monitoreo, la seguridad y las actualizaciones son responsabilidad del equipo de desarrollo.

6. Extracción de información y resúmenes de documentos

6.1. Serverless es adecuado si:

- Los documentos son pequeños o medianos, y puedes dividirlos en partes procesables.
- Usas modelos ligeros o externalizas la inferencia a servicios gestionados como AWS Comprehend, Google Cloud Natural Language o Azure Cognitive Services.
- Necesitas procesar solicitudes de forma esporádica o en picos impredecibles.
- Los resultados pueden obtenerse en un tiempo breve, ya que las funciones serverless suelen tener límites de tiempo (por ejemplo, 15 minutos en AWS Lambda).

6.2. Serverless no es adecuado si:

- Los documentos son muy grandes o complejos (por ejemplo, legislación completa o contratos extensos).
- El procesamiento implica modelos pesados que requieren GPUs o TPUs para funcionar eficientemente.
- Se necesitan procesos que duren mucho tiempo, como análisis iterativos o entrenamiento de modelos personalizados.

6.3. Solución híbrida:

Dividir los documentos grandes en secciones y procesarlos en paralelo mediante Serverless, almacenando resultados parciales en un servicio como S3 o una base de datos. Para modelos grandes, podrías usar un microservicio especializado que funcione en contenedores con hardware dedicado.

7. Comparación clave

Característica	**Serverless IA**	**Microservicios IA**
Gestión de infraestructura	Totalmente gestionada	Gestionada por el equipo
Escalabilidad	Automática	Personalizable
Costo	Pago por uso	Costos iniciales más altos
Control	Limitado	Total
Desempeño	Ideal para tareas ligeras	Mejor para tareas intensivas
Flexibilidad	Baja	Alta

Cuadro 1: Comparación de Serverless IA y Microservicios IA

8. ¿Cuál elegir?

8.1. Elige Serverless IA si:

- Quieres un enfoque ágil y de bajo mantenimiento.
- Tus cargas de trabajo IA son ligeras y eventuales.
- Necesitas rapidez para probar ideas o lanzar productos mínimos viables (MVPs).

8.2. Elige Microservicios IA si:

- Estás construyendo un sistema complejo y escalable.
- Necesitas ejecutar modelos IA grandes o procesos intensivos.
- Quieres control completo sobre la infraestructura y el hardware.

8.3. Plataformas y Herramientas Serverless para Despliegue de IA y ML

Las soluciones serverless están revolucionando el panorama de la inteligencia artificial (IA) y el aprendizaje automático (ML) al ofrecer infraestructuras escalables sin la necesidad de gestión directa del hardware. A continuación se destacan algunas de las plataformas serverless más populares que ofrecen soluciones para tareas intensivas de IA y ML:

- **OpenAI API (Serverless IA)**
  - **Plataforma**: OpenAI proporciona una API serverless para acceder a modelos avanzados como GPT y DALL·E.
  - **Características**:
    - Escalabilidad automática que adapta la capacidad de procesamiento según la demanda.
    - No necesitas preocuparte por la infraestructura ni el hardware.
    - API fácil de integrar en aplicaciones de IA.
  - **Casos de Uso**:
    - Automatización de la generación de contenido (artículos, resúmenes, etc.).
    - Creación de chatbots conversacionales con modelos de lenguaje natural.
  - **Limitaciones**:
    - Costo asociado con un alto volumen de solicitudes.
    - Poca flexibilidad en cuanto a personalización de hardware y configuraciones avanzadas.
- **Google Cloud Vertex AI (AI & ML Serverless)**
  - **Plataforma**: Google Cloud ofrece Vertex AI, una solución serverless para el desarrollo de modelos de IA y su implementación a escala.
  - **Características**:
    - AutoML para entrenar modelos personalizados sin necesidad de escribir código.
    - Uso de GPUs serverless para entrenamiento de modelos complejos.
    - Integración con otros servicios de Google Cloud como BigQuery y Pub/-Sub.
  - **Casos de Uso**:
    - Análisis de texto, video e imágenes.
    - Desarrollo de modelos personalizados de ML.
  - **Ventajas**:
    - Flexibilidad para tareas específicas como procesamiento de lenguaje natural (NLP) y visión por computadora.
  - **Limitaciones**:
    - Complejidad en la configuración inicial.

- Necesidad de conocimientos avanzados de infraestructura de Google Cloud.

- **AWS Lambda y SageMaker (AI & ML en Serverless)**
  - **Plataforma**: Amazon Web Services (AWS) ofrece Lambda para la ejecución de funciones serverless junto con SageMaker para el entrenamiento y la inferencia de modelos de IA.
  - **Características**:
    - SageMaker para inferencia serverless de modelos de IA.
    - Lambda para tareas de preprocesamiento, orquestación y procesamiento de datos.
  - **Casos de Uso**:
    - Análisis de voz y transcripción con Amazon Transcribe.
    - Clasificación de imágenes con Amazon Rekognition.
  - **Ventajas**:
    - Integración nativa con otros servicios de AWS.
    - Escalabilidad automática para manejar cargas variables.
  - **Limitaciones**:
    - Costos asociados con grandes volúmenes de datos y procesamiento de GPU.

- **Microsoft Azure Cognitive Services (Serverless AI)**
  - **Plataforma**: Azure ofrece Cognitive Services, un conjunto de APIs serverless para tareas de IA como visión por computadora, análisis de texto y procesamiento de voz.
  - **Características**:
    - APIs preentrenadas para la conversión de texto a voz, análisis de sentimientos, y visión por computadora.
    - Azure Functions para integrar APIs de IA en aplicaciones sin gestionar servidores.
  - **Casos de Uso**:
    - Procesamiento de texto y voz para chatbots y asistentes virtuales.
    - Análisis de imágenes y videos para sistemas de seguridad.
  - **Ventajas**:
    - Ofrece herramientas especializadas para aplicaciones empresariales.
    - Fácil integración con otros servicios de Microsoft.
  - **Limitaciones**:
    - Costo alto para personalizaciones avanzadas o uso extensivo de recursos.

- **Hugging Face Inference Endpoints**
  - **Plataforma**: Hugging Face permite implementar modelos de IA preentrenados (como GPT-2/3, BERT) en plataformas serverless.

- **Características**:
  - Implementación sencilla de modelos grandes con un solo clic.
  - Opciones de GPUs serverless para modelos intensivos en computación.
- **Casos de Uso**:
  - Modelos de lenguaje natural (NLP) para tareas de texto.
  - Visión por computadora para clasificación de imágenes y objetos.
- **Ventajas**:
  - Flexibilidad para personalizar modelos para aplicaciones específicas.
  - Acceso a una amplia comunidad de desarrolladores.
- **Limitaciones**:
  - Los modelos avanzados requieren suscripción de pago para uso intensivo de recursos.

9. Arquitectura del Sistema

9.1. Diseño e Implementación del Backend

El backend será desarrollado en **Rust**, un lenguaje conocido por su seguridad, eficiencia y capacidad para manejar concurrencia. Este componente gestionará la lógica de negocio, el autoescalado de los microservicios y la conexión con la base de datos.

9.2. Desarrollo del Frontend

El frontend será construido con **TypeScript**, **React** y **Vite**, garantizando una interfaz de usuario moderna, interactiva y responsiva. El diseño se enfocará en la experiencia del usuario y facilitará la interacción con los microservicios disponibles.

9.3. API REST

Se desarrollará una API RESTful para garantizar una comunicación eficiente entre el backend y el frontend. La API también permitirá la integración con otras aplicaciones o servicios externos en el futuro.

9.4. Despliegue de Microservicios

Inicialmente, se implementarán microservicios básicos en PLN y procesamiento de imágenes. Estos servicios estarán diseñados para ejecutarse en contenedores Docker, lo que garantiza una alta portabilidad y consistencia del entorno de ejecución. Además, se permitirá la integración de Kubernetes para gestionar el escalado de los contenedores en función de la demanda de los usuarios.

9.4.1. Docker para la Ejecución de Modelos de IA

Cada modelo de IA (ya sea de clasificación de imágenes, extracción de información, o cualquier otro) se implementará dentro de un contenedor Docker. Estos contenedores

incluirán todo lo necesario para ejecutar los modelos de IA: bibliotecas, dependencias y código necesario para realizar el procesamiento. Los pasos básicos serían:

- Desarrollo de imagen Docker: Crear una imagen que contenga el entorno necesario para ejecutar cada modelo de IA, como TensorFlow, PyTorch, o cualquier otra biblioteca relevante.
- API REST: Utilizar herramientas como Flask o FastAPI dentro del contenedor para crear una API REST que permita la interacción con el modelo de IA, recibiendo las solicitudes de los usuarios, ejecutando el modelo y devolviendo los resultados.
- Aislamiento: Cada modelo será ejecutado en su propio contenedor para asegurar el aislamiento y facilitar el mantenimiento.

9.4.2. Orquestación con Kubernetes (Opcional)

Aunque el uso de Docker por sí solo es adecuado para proyectos pequeños, si el número de usuarios y la demanda de procesamiento aumentan, será necesario utilizar Kubernetes para orquestar los contenedores y garantizar que la plataforma sea escalable. La integración de Kubernetes permite:

- Escalado automático: Kubernetes puede aumentar o disminuir automáticamente el número de instancias de cada modelo en función de la demanda, garantizando que siempre haya suficientes recursos disponibles sin desperdiciar capacidad.
- Balanceo de carga: El tráfico de las solicitudes de los usuarios puede ser distribuido entre varias instancias del contenedor del modelo, asegurando tiempos de respuesta rápidos y eficiente uso de los recursos.
- Despliegue continuo: Con Kubernetes, el sistema puede ser actualizado o modificado sin interrumpir los servicios existentes, garantizando una alta disponibilidad.

9.4.3. Uso de Kubernetes para la Gestión de la Infraestructura Serverless

Kubernetes facilita la implementación de un entorno serverless al permitir que los contenedores se gestionen de manera eficiente y flexible. Aunque Kubernetes no es estrictamente una plataforma serverless, su integración con servicios como `Kubernetes FaaS` o `Knative` proporciona una solución similar. De esta manera, los usuarios pueden ejecutar sus modelos de IA de manera eficiente sin preocuparse por la infraestructura subyacente.

10. A resolver este tema

Un aspecto clave que requiere resolución en el marco de este TFM es el diseño y gestión del almacenamiento de datos y personalización de modelos de IA. Algunos de los retos incluyen:

- **Almacenamiento eficiente**: Diseñar un sistema que permita el almacenamiento eficiente de datos generados y utilizados por los microservicios, asegurando escalabilidad y disponibilidad.

- **Seguridad de datos**: Implementar mecanismos que garanticen la privacidad y la seguridad de los datos de los usuarios.
- **Personalización de modelos**: Permitir a los usuarios entrenar y personalizar modelos de IA para sus necesidades específicas sin comprometer la integridad del sistema.
- **Gestión del entorno virtual**: Diseñar un mecanismo eficiente para la administración del entorno virtual que ofrezca una experiencia fluida y personalizada para los usuarios. Este aspecto incluye evaluar la viabilidad de integrarse con proveedores de terceros que gestionen y almacenen los grandes volúmenes de datos que los modelos de inteligencia artificial (IA) requieren para cada usuario. Es importante considerar que los modelos de IA suelen necesitar una cantidad significativa de datos para entrenarse y generar predicciones precisas. Mantener, gestionar y almacenar estos datos de manera eficiente en el sistema puede representar una gran dificultad. Debido a los desafíos asociados con la gestión de grandes volúmenes de datos, se sugiere que, en este TFM, el enfoque no debería ser mantener estos datos internamente, sino más bien integrarse con proveedores especializados en almacenamiento de datos. De este modo, se aliviaría la carga sobre la infraestructura del sistema y se garantizaría una mayor escalabilidad y eficiencia en la gestión de los datos necesarios para los modelos IA.

11. Conclusiones y Futuro

Este proyecto tiene como objetivo proporcionar una plataforma serverless que permita a los usuarios acceder a microservicios de IA de forma sencilla y eficiente. La implementación de modelos de IA dentro de contenedores Docker, con la posibilidad de orquestarlos utilizando Kubernetes, garantiza una solución escalable y flexible que puede evolucionar para adaptarse a las crecientes demandas de los usuarios. La viabilidad del sistema será evaluada y, si es positiva, el proyecto podrá convertirse en la base para una plataforma más amplia de servicios de IA accesibles para todos.