

Ejercicio 2



Enero 2023

Carta de presentación



Estimad@s

El presente documento contiene la propuesta y enfoque para la solución de *Model Serving & Monitoring* de un sistema que predice el tipo de cambio

Persona de contacto:	Omar Santa Cruz (aka Santa_Cruz)
Correo electrónico:	mcs.santacruz.ia@gmail.com
Teléfono:	953 116 1963

Espero que esta propuesta este alineada a su cultura de datos, al tiempo que les reitero mi agradecimiento por la oportunidad en este proceso y mi intención de seguir colaborando con ustedes en éste y otros proyectos que surjan en el futuro.

Secciones



- 1** Resumen ejecutivo
- 2** Respuesta a las preguntas
- 3** Arquitectura

01 Resumen ejecutivo



En resumen



Objetivo	<ul style="list-style-type: none">• Dar respuesta a las preguntas planteadas y proponer una arquitectura tecnológica para servir y monitorear un modelo de machine learning enfocado en predecir el tipo de cambio actual.	Tecnologías
Descripción		Python Cloud
Objetivos Estratégicos		Fechas
Alcance	<ul style="list-style-type: none">• Respuesta a las preguntas planteadas• Arquitectura de alto nivel	<ul style="list-style-type: none">• Fecha de inicio: 15 de Enero• Fecha de entrega: 20 de Enero

02 Entendimiento



Entendimiento

Imagina que el equipo de ciencia de datos tiene un modelo estadístico predictivo (algoritmo, input y parámetros) para el tipo de cambio. Para el modelo se usaron datos de fuentes públicas (Banxico, Inegi) y datos de bases de datos internas.

Se requiere desplegar este modelo para que el área de crédito pueda consultarlo, a continuación, explica a detalle cómo lo harías tomando en cuenta los siguientes aspectos:

Preguntas planteadas

¿Cómo medirías la calidad de los datos de entrada? (1/4)

Debido a que el tema de calidad de datos pertenece al tópico *data quality* que es una de las actividades fundamentales de una estrategia de gestión y gobierno de datos, se dará por hecho que se a definido los tres pilares:

- Marco de referencia
- Modelo de madurez
- Metodología

Tomando a DAMA como el marco de referencia, daremos por hecho que ya se han defino estándares y especificaciones para controles de calidad de datos.

Como primer paso nos centraremos en el perfilamiento de los datos, así como una posterior limpieza, formato y estandarización.

Preguntas planteadas

¿Cómo medirías la calidad de los datos de entrada? (2/4)

Utilizando un motor de perfilamiento (como Great Expectations o Apache Griffin) produciríamos estadísticas que podemos usar para identificar patrones en el contenido y la estructura de los datos. Por ejemplo:

- Conteos de nulos:** Identifica los nulos existentes y permite la inspección de si son permitidos.
- Valor máx./mín.:** Identifica valores atípicos, como números negativos y fuera de rango.
- Longitud máx./mín.:** Identifica valores atípicos o inválidos para campos con longitudes específicas en datos categóricos
- Distribución de frecuencia de valores para columnas individuales:** Permite la evaluación de la razonabilidad

por ejemplo: distribución de códigos de país para transacciones, inspección de valores frecuentes o poco frecuentes, así como el porcentaje de los registros poblados con valores predeterminados

- Tipo y formato de datos:** Identificar el nivel de incumplimiento de los requisitos de formato, así como la identificación de formatos inesperados (por ejemplo: número de decimales, espacios incrustados, valores de muestra)

Preguntas planteadas

¿Cómo medirías la calidad de los datos de entrada? (3/4)

Ya que tenemos perfilados nuestros datos y conocemos la estructura de estos podemos definir métricas que midan la calidad de datos de los patrones antes mencionados. A través de estas características medibles debemos dar cumplimiento a las 12 dimensiones de calidad de datos .

Por lo general se hace en iteraciones y en cada iteración se toman como objetivo dos dimensiones según la prioridad asignada por los stakeholders. La primera dimensión puede ser , por ejemplo la completitud.

Dimensión de calidad	Descripción
Completitud	<p>Debemos de medir si todos los datos están presentes. La completitud puede medirse al nivel de conjunto de datos, registro o columna.</p> <p>¿Qué preguntas podemos hacernos ?</p> <ul style="list-style-type: none">a) ¿Contienen los datos todos los registros esperados?b) ¿Están los registros poblados correctamente?c) ¿Están las columnas/atributos poblados al nivel esperado?

Preguntas planteadas

¿Cómo medirías la calidad de los datos de entrada? (4/4)

Dimensión de calidad	Descripción
Compleitud	<p>Si le damos solución a la pregunta del inciso a) podemos obtener un planteamiento como este :</p> <ol style="list-style-type: none">1. Regla de negocio de la que se desprende la métrica : El llenado de campos es mandatorio2. Dividir el número obtenido de registros donde los datos estan llenos, por el total de registros de la tabla o de la base de datos. Posteriormente multiplicar por cien para obtener un porcentaje completo3. Mapearlo al software de perfilamiento4. Definir umbral de éxito : Inaceptable 20% de valores ausentes

Preguntas planteadas

2.- ¿De donde obtendrías la información? ¿Cómo?¿Con qué herramienta? ¿Cómo orquestarías? (1/1)

De acuerdo a la especificaciones, el modelo consume datos de fuentes públicas (Banxico, Inegi) y de bases de datos internas. Lo cual plantea diversos escenarios, en los cuales se tiene dos tipos de datos : (1) Datos que se obtendrán en tiempo real y (2) Datos estáticos para ciertos intervalos de tiempos como semanas, meses y/o años.

Data en tiempo real : Se obtiene a través de APIs o Hook/Web Hooks o tecnologías de paso de mensajes.

Datos en batch : Se puede mantener en memoria, cache á través de consultas a una base de datos o consumo de APIs.

Las soluciones y herramientas más populares para la obtención de datos son (1) generar peticiones **HTTP get** utilizando un cron/timer o bajo demanda (2) Suscribirse a hooks (eventos) o topic (kafka).

Cuando la cantidad de fuentes de datos (batch) genera complejidad es necesario utilizar un orquestador de flujos de datos como Airflow.

Preguntas planteadas

¿Qué framework utilizarías? (1/4)

Los frameworks end to end más populares para servir y monitorear una solución basada en datos son:











Preguntas planteadas

¿Qué frameworks utilizarías? (2/4)



Comparación de las herramientas más populares que existen en el mercado

	Extended								 argo
Pipelines	X	X	X	X		X			X
Despliegue	X	X	X		X		X	X	
Entrenamiento	X	X	X	X		X			
Registro	X	X	X	X		X			
Evaluación	X	X	X	X		X			

Preguntas planteadas

¿Qué frameworks utilizarías? (3/4)



1ra propuesta : Esta propuesta se basa en tecnologías *open source* e infraestructura *On-premise/Cloud*

Componetes	Tools
Data model	Scikit learn
Source control	Git
Test & build services	Pytest & Make
Model & data registry	DVC
Feature store	Feast
ML metadata store	MLFlow
ML pipeline orquestación	MLFlow
Human in the loop	Rubrix
Deploy	Terraform
Endpoint	Flask / Fast API
Cloud Server	Digital Ocean

La lista de frameworks propuesta abarcan la mayor parte del el ciclo de vida del dato y clico de vida del modelo.

Preguntas planteadas

¿Qué frameworks utilizarías?



2da propuesta: Esta propuesta se basa en SageMaker Studio como componentes principal

Componetes	Tools
Data model	Scikit learn
Source control	Git
Test & build services	Pytest & Make
Model & data registry	SageMaker Studio
Feature store	SageMaker Feature Store
ML metadata store	SageMaker Studio
ML pipeline orquestación	SageMaker Studio
Human in the loop	Rubrix
Deploy	SageMaker Studio
Endpoint	SageMaker Studio
Cloud Server	AWS EC2

La lista de frameworks propuesta abarcan la mayor parte del el ciclo de vida del dato y clico de vida del modelo.

Preguntas planteadas

¿Qué consideraciones tendrías para el consumo del modelo? (1/3)

1.- Reproducibilidad: El enfoque MLOps exige que se cumpla con el paradigma de infraestructura inmutable e infraestructura como código (IaC) para la creación de sistemas con ambientes reproducibles. El uso de Docker cumple con estos patrones de arquitectura para la mayoría de escenarios, y se complementa según el caso de uso:

(1) Con un software de gestión de infraestructura (como Terraform) y gestión de la configuración, orquestación y aprovisionamiento (como Ansible)

(2) Una plataforma para automatizar la implementación, el escalado y la administración de aplicaciones (cluster) en contenedores (como Kubernetes).

Ambas soluciones permiten explotar a Docker e implementar aplicaciones con mayor rapidez.

Preguntas planteadas

¿Qué consideraciones tendrías para el consumo del modelo? (2/3)

2.- **Escalado:** El equilibrio de carga evita que un sitio web se paralice cuando hay un desbordamiento de solicitudes. Un balanceador de carga envía solicitudes a servidores que pueden manejarlas de manera eficiente para maximizar la velocidad y el rendimiento. Al hablar un balanceador introducimos el concepto de cluster de servidores, el cual pretende mejorar los siguientes parámetros de la arquitectura:

- Alto rendimiento
- Alta disponibilidad
- Equilibrio de carga
- Escalabilidad

La distribución del tráfico entre las instancias en un entorno puede integrarse de manera propia a los web servers, delegarse; si utilizamos un proveedor en la nube (como Elastic Load Balancing de AWS) o especificar un balanceador de carga compartido.

Preguntas planteadas

¿Qué consideraciones tendrías para el consumo del modelo? (3/3)

3.- Seguridad: La seguridad de sistemas es un tema amplio y complejo, sin embargo para la vertical de datos debos considerar almenos tres tópicos para conserva las características y la integridad de los datos de producción.

- Web Tokens y autenticación
- Ficrado de información en los diferentes canales
- Enmascaramiento de datos personales y sensibles

Las normas de seguridad de datos se han vuelto muy estrictas y son punto clave para la confianza del usuario y temas de cumplimiento regulatorio.

Preguntas planteadas

¿Qué monitorearías y medirías del despliegue del modelo? ¿Qué metadata obtendrías de los logs? (1/2)

Datos de entrada (input data drift): Datos atípicos (crean un detecto de anomalías), rango de los datos, máximos y mínimos y distribución estadística de los datos.

Datos de salida (predict data drift): Distribución estadística de los datos predictivos del modelo para detectar una degradación general de modelo. Distribución estadística de los datos por segmentos (genero, ubicación geográfica o edad) para detectar degradaciones de sesgos. Estadísticas de predicción y umbral de desición.

Tiempos: Tiempo que tarda el modelo en arrojar una predicción, número de peticios por minuto que atiende el modelo, tiempo que el usuario tarda en nuestro sistema. Tiempo que tarda los usuarios en ingresa la información (parámetros).

Infraestructura : Uso de CPU, uso de memoria ram, número de instancias desplegadas del modelo

Métricas comerciales y de producto: número de clientes que optaron por la tasa de cambio de nuestro banco en un periodo determiando.

Preguntas planteadas

¿Qué monitorearías y medirías del despliegue del modelo? ¿Qué metadata obtendrías de los logs? (2/2)

Logs : Si no se utiliza una estrategia de recopilación de datos en tiempo real todos los datos de entrada y salida de modelo se almacenan en el log de sistema para su extracción periódica. Dichos datos se deben enviar a la FeatureStore para la estrategia de re-entrenamiento continuo y al datalake para que sean consumidos por el área de analítica o el data warehouse de reporte.

Otro tipo de información que almacena los logs del sistema (si no se cuenta con una base de datos de errores) son los diversos tipos de errores y excepciones del código en tiempo de ejecución, falla de conexiones a bases de datos, etc.

Preguntas planteadas

¿Cada cuánto recomendarías actualizar el modelo?, ¿Cómo lo harías? ¿Qué aspectos y criterios utilizarías? (1/2)

Siguiendo nuestra cultura Data Driven, todas nuestras decisiones son tomadas con evidencia basa en datos, por lo que nuestra, nuestra estrategia de entrenamiento continua se basa en los resultados del monitoreo continuo de desviación de datos, **degradación** del rendimiento, sesgo inesperado o problema de integridad de datos que perjudique los resultados, así como de los KPIs de negocio. Además para re-entrenamiento se debe tomando en cuenta el feedback de los usuarios y **Stakeholders**.

La mejor solución es tratar convinar las estrategias de entrenamiento online y batch, tomando elementos según el caso de uso del modelo :

Entrenamiento on-line: Se utiliza en aplicaciones donde se requieren constantemente nuevos patrones de datos (p. ej., transacciones financieras)

- En necesario un enfoque **human in the loop** para validar datos con mucha incertidumbre y una plataformas de labeling especializadas en este tipo de casos como lo es Rubrix o Prodigy, también es necesario una FeatureStore bien definida.
- Dependecía con algoritmo de ML: la mayoría de los algoritmos estadísticos debe plantearse como un enfoque on-line, ya que al ser parametrizados necesitan ver todos los datos.
- Madurez de datos a nivel pro-activa o data driven (cloud , hiperautomatización y interoperabilidad)

Preguntas planteadas

¿Cada cuánto recomendarías actualizar el modelo?, ¿Cómo lo harías? ¿Qué aspectos y criterios utilizarías? (2/2)

Entrenamiento batch: Se usa en aplicaciones donde los patrones de datos permanecen constantes y no tienen desviaciones repentinas de conceptos (p. ej., detección de objetos en imágenes)

- Se puede implementar casi en cualquier contexto de madurez y bajo cualquier enfoque o metodología de datos
- Más fácil de implementar porque el aprendizaje fuera de línea proporciona a los ingenieros y científicos más tiempo para perfeccionar el modelo antes de la implementación.

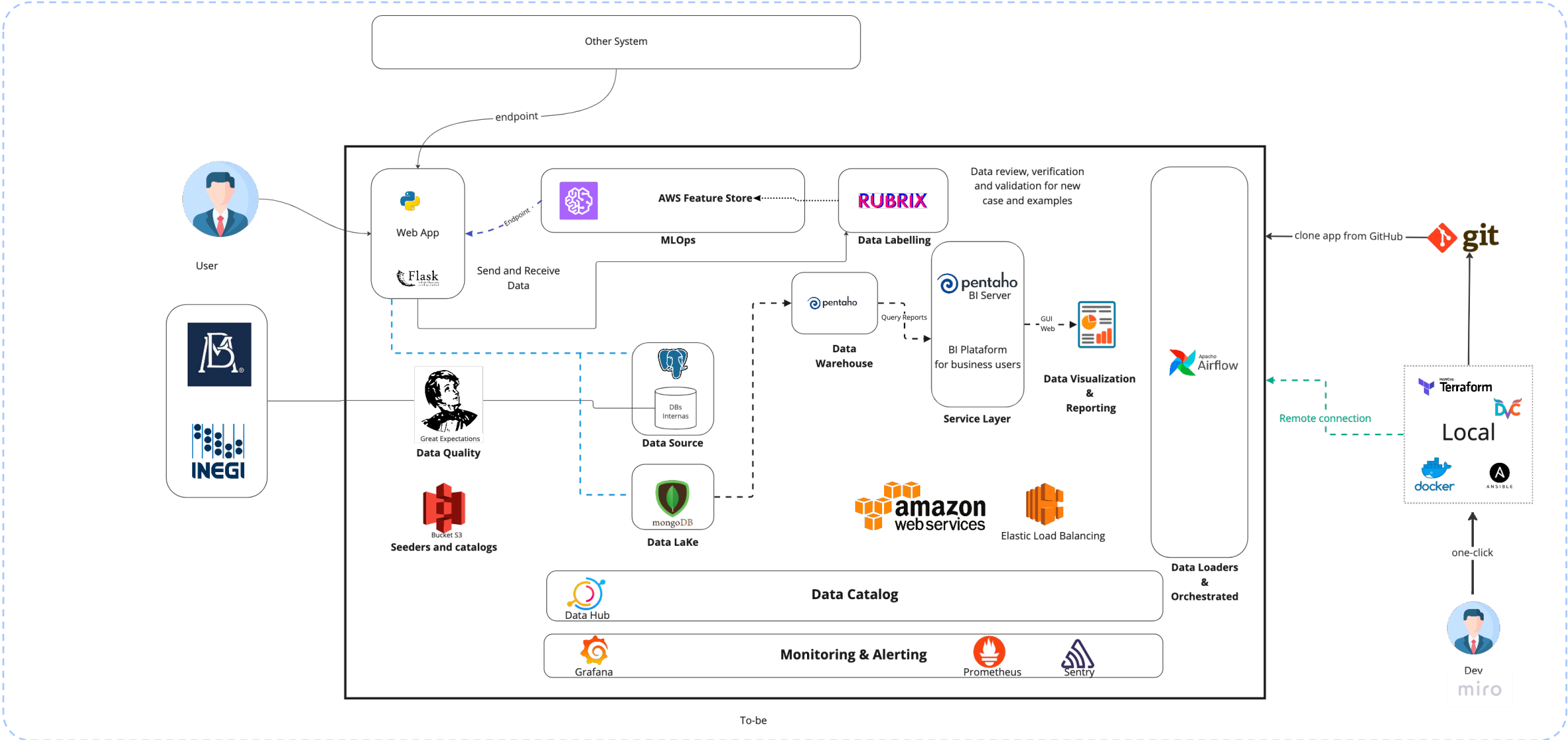
Es posible convinar ambos enfoques para la mayoría de casos de uso con éxito si se tiene una estrategia de versionado de modelos y de datasets para poder realizar un rollback de los despliegues de forma manual o automática en caso de ser necesario.

03 Arquitectura



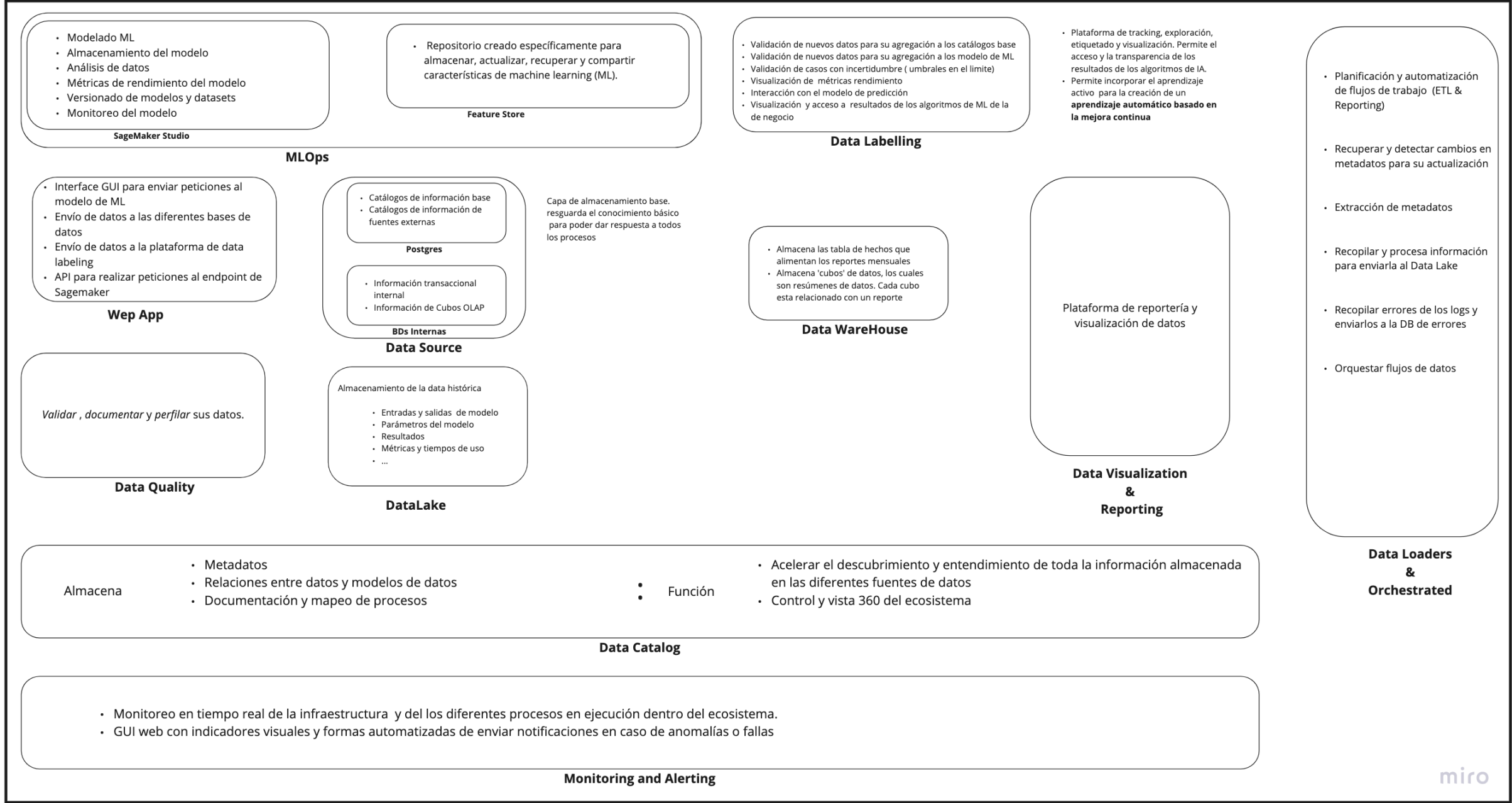
Architectura

Propuesta de arquitectura end to end para un modelo de machine learning



Arquitectura

Descripción de los módulos de la arquitectura propuesta





Gracias por tu tiempo