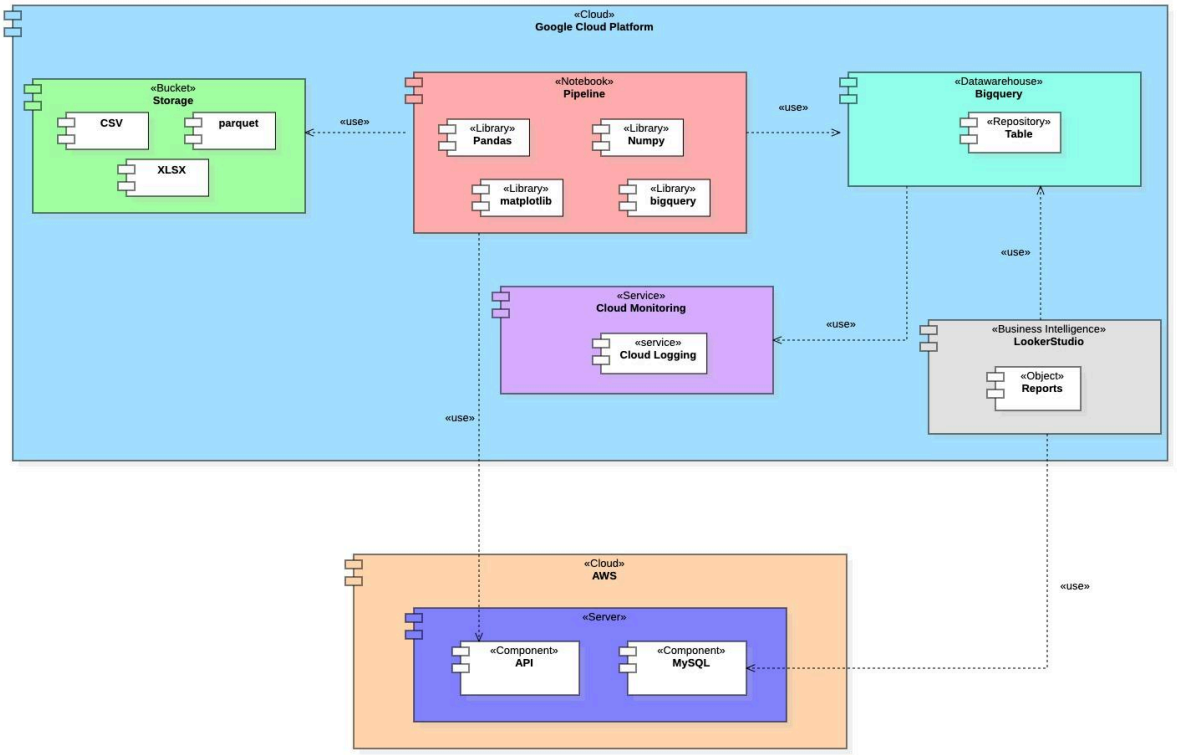
Arquitectura de Software Pipeline ETL
Stack Tecnológico

Diagrama de la Arquitectura



Especificación de los componentes

Esta arquitectura de ETL está estructurada con varios módulos que garantizan la adquisición, procesamiento, almacenamiento y análisis de datos.

a. **Pipeline Central con Colab en GCP**

Este módulo es responsable de ejecutar los flujos de procesamiento de datos. Colab permite la automatización del pipeline de datos y facilita la integración con fuentes externas, como APIs y sistemas de almacenamiento.

b. **Repositorio de Archivos (Cloud Storage)**

Los datos en distintos formatos (CSV, JSON, JSONL, PDF) se almacenan en un bucket, desde donde el pipeline de ETL los procesa y los transforma antes de cargarlos en BigQuery.

c. **Data Warehouse en BigQuery**

BigQuery es el sistema de almacenamiento analítico donde se organizan los datos en tablas y vistas optimizadas para consultas rápidas. Soporta SQL estándar y permite procesar grandes volúmenes de datos sin gestionar infraestructura física.

d. **Monitoreo con Cloud Monitoring**

Permite la supervisión de los pipelines y servicios, proporcionando métricas y alertas en caso de errores o anomalías en la ejecución del ETL.

e. **Módulo BI con Looker Studio**

Looker Studio permite la creación de dashboards interactivos y reportes basados en los datos almacenados en BigQuery.

Atributos de Calidad

- **Rendimiento**: Tener tiempos cortos de respuesta mejora la experiencia del usuario y mejora los tiempos de análisis y toma de decisiones.
  - Cuando llegan datos nuevos de las fuentes detectadas se dispara el pipeline ejecutando varios componentes de extracción, transformación y carga en un ambiente donde se ejecuta una sola instancia el pipeline debe ejecutar todos los componentes en menos de una hora.
    Metrica: 1 Hora
  - Al llegar datos nuevos a las fuentes de datos se dispara el pipeline ETL ejecutando los componentes en un ambiente donde se ejecuta una sola instancia se usará los mínimos recursos para llevar a cabo el correcto funcionamiento y cumplimiento de los tiempos.
    Metrica: Mínimo de recursos de infraestructura (memoria, CPU y nodos) para procesar la info en menos de 1 Hora
- **Escalabilidad**: Se espera que el pipeline ETL soporte la cantidad de datos estimada a procesar y funcionar correctamente si crece dos veces los datos.
  - Cuando se sube al almacenamiento origen los datos con un tamaño de 200 MB el pipeline ETL ejecutará todos los componentes de forma exitosa.
    Metrica: Tamaño máximo de los datos origen 200MB.
- **Seguridad**: Se espera confidencialidad e integridad de los datos.
  - Cuando los usuarios obtienen los datos se subira a un almacenamiento seguro basado en credenciales.
  - El pipeline ETL debe extraer la información de fuentes seguras basado en credenciales.
    Metrica:
    - Cantidad de usuarios con acceso basado en credenciales
    - Cantidad de extracción de datos con acceso basado en credenciales
- **Disponibilidad y Fiabilidad:** El ETL debe tener logging y trazabilizad. Además si llega a fallar debe poder enviar alertas y estar disponible para una nueva ejecución
  - Cuando se suba al storage datos se dispara el pipeline ETL este debe estar disponible y llevar a cabo la ejecución de sus componentes
    Metrica: Disponibilidad 24 horas por 7 días.

- ➢ Cuando los datos se suben al storage se dispara el pipeline ETL y ocurre errores en algun componente se debe registrar los errores y enviar alertas a los stakeholders.
  Metrica: Cantidad de errores
- ➢ Cuando se suba el storage al storage datos se dispara el pipeline ETL y lleva a cabo la ejecución de todos sus componentes de forma exitosa, se debe hacer un logging y trazabilidad de todos los componentes
  Metrica: Cantidad de ejecuciones exitosas

❖ **Interoperabilidad:** El ETL de poder extraer y cargar datos de fuentes como CSVs, JSON, JSONL, datos no estructurados, APIs y bases de datos.

- ➢ Cuando se suba al storage datos CSVs, JSON, JSONL, datos no estructurados se dispara el pipeline ETL y ejecutará exitosamente todos sus componentes.
  Metrica: Número de integraciones con fuentes distintas
- ➢ Una tarea planificada se dispara y ejecutará el pipeline ETL el cual extrae info de APIs y carga los datos a un datawharehouse de forma exitosa.

❖ **Mantenibilidad:** El ETL debe ser modular ya que incrementa la reutilización de sus componentes, la fácil detección de errores, mejores metodos de pruebas y fácil modificación.

- ➢ Cuando se suba al storage los datos se ejecutará el pipeline ETL y sus componentes deben hacer una unica tarea además de poder ser reutilizados para llevar a cabo exitosamente todo el proceso.
  Metrica: Número de componentes reutilizados, Número total de componentes

Decisiones de Arquitectura:

❖ Se implementará el pipeline con Ray[1] que permite el procesamiento distribuido de los componentes lo cual incrementa el performance y la escalabilidad. Para esto se usará cluster de Kubernetes o serverless en la nube de Google

❖ Se cargarán los archivos orígenes de datos en el storage[2] de la nube de Google

❖ Se usará credenciales de acceso para subir archivos al storage y para obtener archivos.

- ❖ Se cargarán los datos luego de las transformaciones en Bigquery[3].
- ❖ El acceso a bigquery para lectura y escritura se realizará con credenciales y dentro de la nube de Google.
- ❖ Se usará las librerias de logging de la nube de Google para la trazabilidad en la ejecución del pipeline
- ❖ Se usará Google Monitoring para revisar, crear alertas y filtrar logs.
- ❖ Se usará python, pandas, numpy, unstructured para procesar diferentes formatos de datos como CSV, JSON, JSONL, APIs .

✅ **Ventajas de la Arquitectura**

- Escalabilidad: Todos los módulos están en la nube, lo que permite adaptarse a la demanda sin gestionar infraestructura. Además el procesamiento distribuido permite tener alta escalabilidad y rendimiento
- Automatización: Vertex AI permite crear pipelines eficientes
- Visualización avanzada: Looker Studio facilita la interpretación de datos a través de dashboards interactivos.
- Optimización de consultas: BigQuery permite manejar grandes volúmenes de datos de forma eficiente.

❌ **Desventajas de la Arquitectura**

- Costos: Si no se optimiza el uso de BigQuery y Vertex AI, para considerables volúmenes de datos los costos pueden ser altos.
- Curva de aprendizaje: Requiere conocimientos intermedios en Google Cloud y SQL.

Referencias:

[1]https://cloud.google.com/vertex-ai/docs/open-source/ray-on-vertex-ai/overview?hl=es-419
[1] https://docs.ray.io/en/latest/data/quickstart.html#data-quickstart
[2] https://cloud.google.com/storage/docs
[3] https://cloud.google.com/bigquery/docs