Msc. Einar Rocha

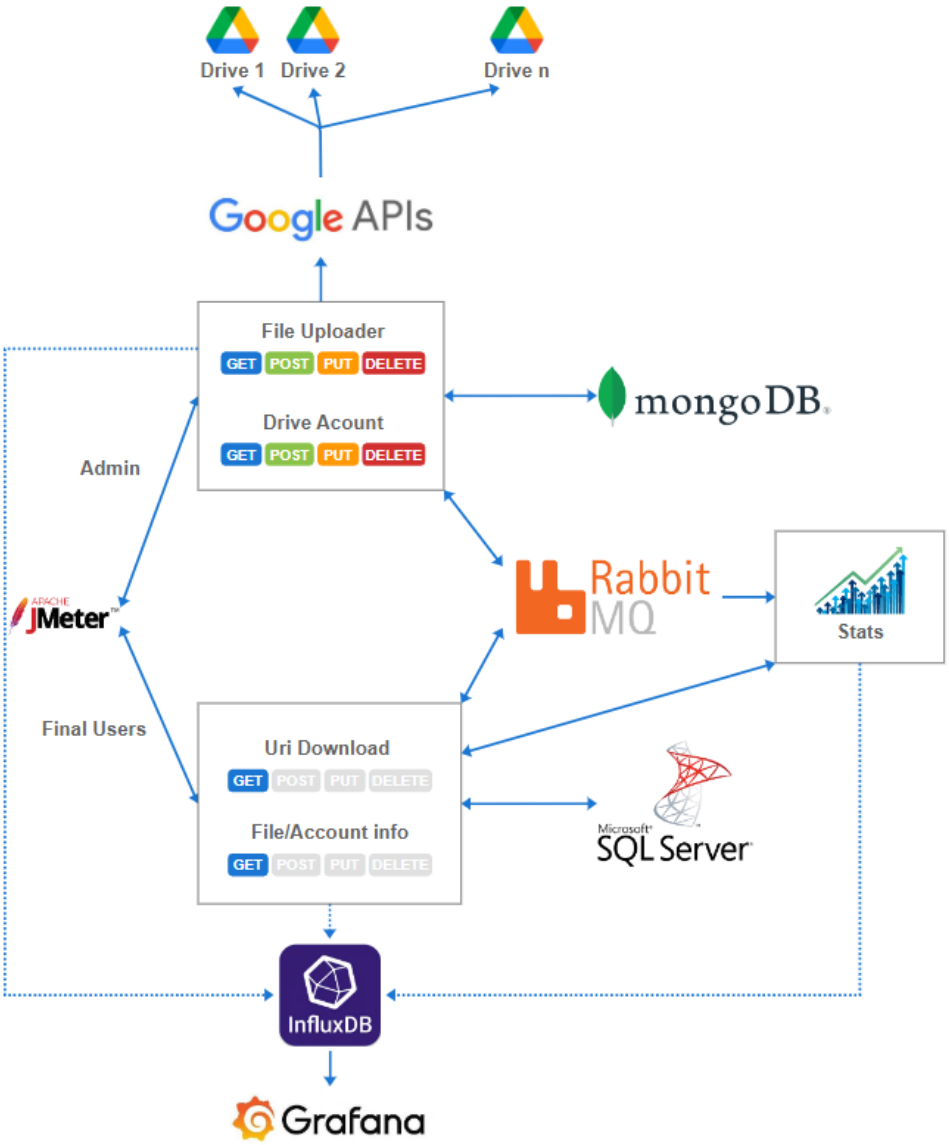
File Server

Descripción:

Se desea implementar un POC de un servidor de archivos. De alguna forma debe soportar un millón de usuarios activos mensualmente, y cada usuario podrá descargar 500 Megabytes mensuales. Solo se cuenta con un servidor VPS que tiene una transferencia mensual de 5 TB.

Para alcanzar la transferencia requerida se utilizarán múltiples instancias o cuentas de Google Drive, sabiendo que cada cuenta permite 100 GB de transferencia diaria.

Diagrama:



Msc. Einar Rocha

Requerimientos:

1. Se tiene los microservicios: Uploader, Downloader y Stats
2. Uploader maneja una base de datos documental con cualquier “data base accesss pattern”. Ej Repository, Mapper, DTO, Active Record, etc.
3. Downloader maneja una base de datos relacional con cualquier “data base accesss pattern”. Ej Repository, Mapper, DTO, Active Record, etc.
4. Stats no tiene una base de datos, solo realizara los cálculos.
5. Los servicios pueden comunicarse entre si pero no pueden acceder al repositorio (DB) de otro servicio.
6. Uploader tiene un crud completo para almacenar las cuentas de Google Drive Ej: id, email, GoogleDriveKey, etc.
7. Uploader tiene un crud completo para subir archivos. Ej: id, name, size, driveId, status, etc.
8. Cuando Uploader recibe un archivo, inmediatamente se guarda en un mongo GridFS temporalmente y se devuelve una respuesta al cliente con un status file: “replicating”.
9. Cuando se sube un archivo a todas las cuentas se cambia el status a “uploaded”.
10. Downloader tiene un enpoint (GET) para devolver las urls de descarga
11. Downloader tiene la lógica para que se utilice todas las cuentas de manera uniforme y secuencial para balancear la descarga diaria.
12. Downloader expone la información pre-calculada de los archivos: numero de descargas, MB descargados el dia actual, MB descargados del archivo en total y un listado con el reporte de todos los archivos
13. Downloader expone la información pre-calculada de las cuentas: número de descargas con la cuenta, MB descargados el dia actual con la cuenta, MB descargados con la cuenta en total y un listado con el reporte de todas las cuentas.
14. Cuando Uploader recibe un archivo, se manejará colas para ir procesando (subiendo) los archivos.
15. **Cuando la cola termina de procesar una operación CRUD (con la api de Google Drive) debe mantener la consistencia con el servicio downloader.**
16. Cuando Downloader devuelve una url de descarga, debe encolar la solicitud para que Stats ejecute un cálculo de reportes.
17. El sistema debe ser **“eventualmente consistente”** en todos los escenarios posibles.
18. Se quiere monitorear la actividad de Uploader, Downloader y Stats usando una TSDB (metricas).
19. Debe probarse todos los puntos anteriores automáticamente utilizando JMeter y poder ejecutarse “n” veces para probar la carga de nuestros servicios.

NOTAS:

Se recomienda usar servicios REST sencillos (microchasis backend).
Pueden usar cualquier patrón de abstracción (ORM, Mapper, DAO, etc.)
En caso de no poder conectarse a las APIS de Google (**después de haber intentado**) se podrá usar algún servicio Mock que simule la API de Google.