Problem Statement

You and your colleagues are often faced with the need to solve graph traversal problems. One typical use-case involves taking a graph as input and querying whether 2 nodes are connected or not. These graphs can range in size from millions to single-digit billions nodes. Setting aside the more complex scenarios where parallel computing techniques are used, the following steps describe how this is currently done:

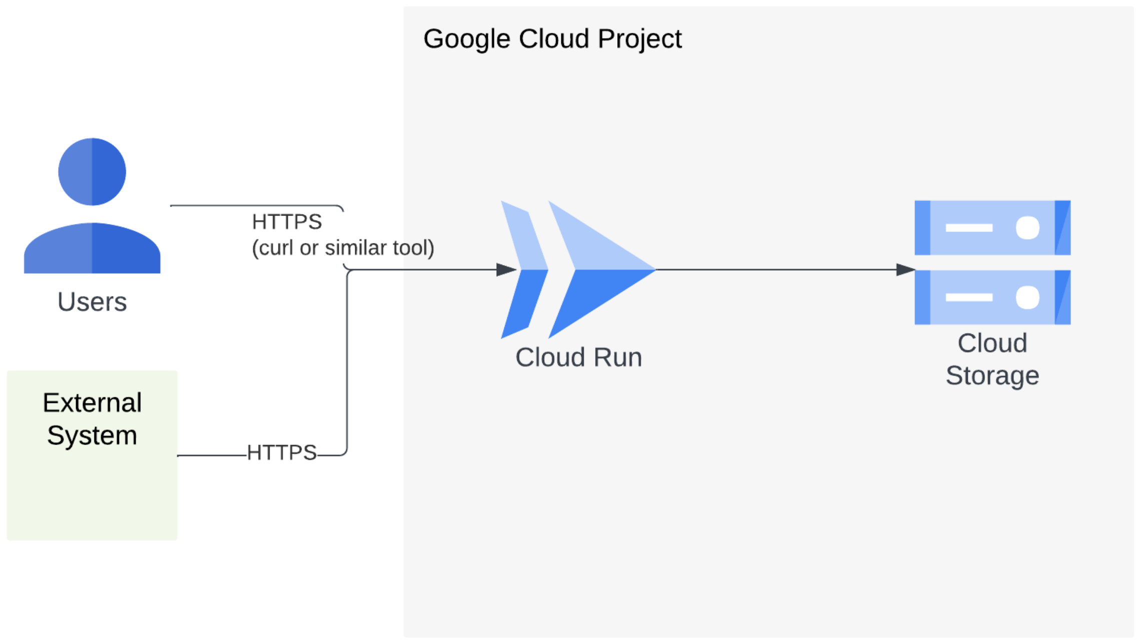
- Schedule time on a shared computer or VM
- Load a file with the adjacency list of the graph
- Load a file with the queries to be executed
- Execute the program (here is a sample)
- Download the results file

There is an opportunity to remove friction in the current process and enable additional use cases, notably the ability to expose these queries programmatically so that they can be consumed by higher-level services. The team has evaluated Google Cloud and decided to use it for a Proof-of-Concept (POC) that, if successful, can be made available to internal stakeholders.

Your task is to implement said POC and submit it and your findings to your peers and to your team's leadership.

POC Definition, Architecture & Design

The POC will implement a simple design:



The flow is simple::

- Users and external systems can query the service via HTTPS
- The Cloud Run service will read the graph definition from Cloud Storage

The query will be formatted in JSON as such (values are examples):
{"start": 789,"end": 1024}

The response will be either something like:
{"end":1024,"length":18,"result":"CONNECTED","start":789}
Or:
{"end":1024,"length":-1,"result":"DISCONNECTED","start":789}

To keep the POC simple, the following can be assumed:

- Unauthenticated access can be allowed to the Cloud Run service
- The graph definition file will be available in a public Cloud Storage location

Suggested steps:

- Using this sample as the basis, create a web application that implements the service described above. Use Cloud Shell and the hosted editor or your preferred code editor on your workstation. A sample graph with 1M nodes can be downloaded from here. Cheat link here; **use only if stuck!** Make sure your service is working properly locally before continuing to the next step.
- Create a Dockerfile that containerizes your web application. This Quickstart may help.

- Deploy your application to Cloud Run. Here is the documentation to the relevant command. Note: this application requires ~1 GiB of RAM; just to be on the safe side, give it 2 GiB. Allow for unauthenticated access to allow for easy testing (note: you would **NOT** do this outside a learning environment).
- Test your application with the curl command. Look at the generated logs.

Beyond the POC

The POC has been deemed wildly successful and leadership has instructed you to move forward with taking it to production. This section proposes some topics for discussion.

Security
- Networking constructs: VPN, IP whitelists, other?
- Identity: IAM for users, Service Accounts for systems, other?
- Securing Cloud Storage files: Service Account & IAM

Scalability
- How to handle multiple graphs?
- Scale horizontally, vertically or both?
- How can graphs be partitioned?

Performance
- Save computed answers? Where? What data structure should you use?
- Pre-compute some or all answers? How?