**REST API creation**

Current hello\_http **function** provides a simple REST API using **Flask** and **Google Cloud Firestore**. It supports: POST to create a new item and GET with or without an id query param to fetch one or all items.

But before that, I need to create a **database** first, and I choose GCP Firestore for the following main reasons:

1) **Budget-friendly** for small projects and scalable for enterprise workloads;

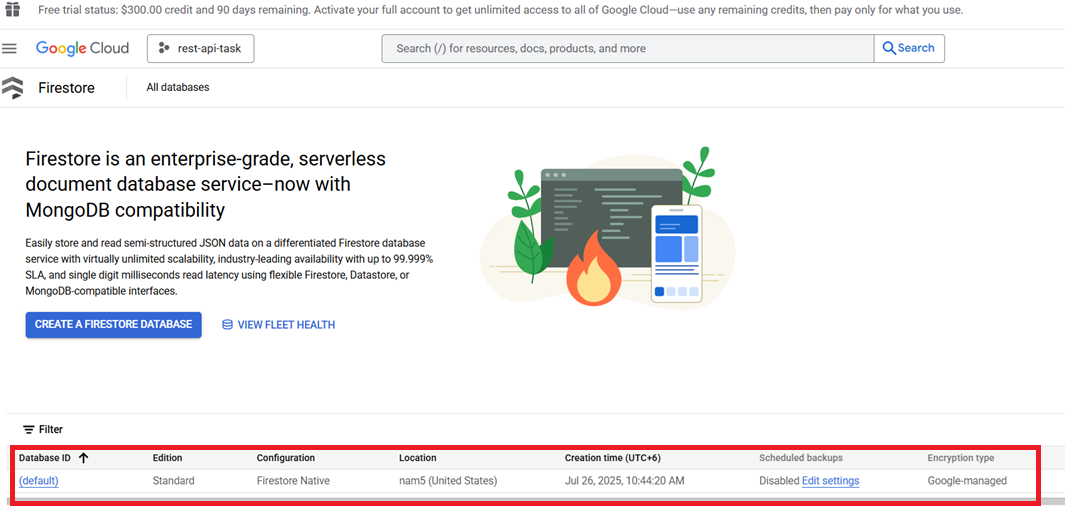
2) Stores data as collections and documents (**JSON**-like structure);

3) Easily **integrates with other service**s;

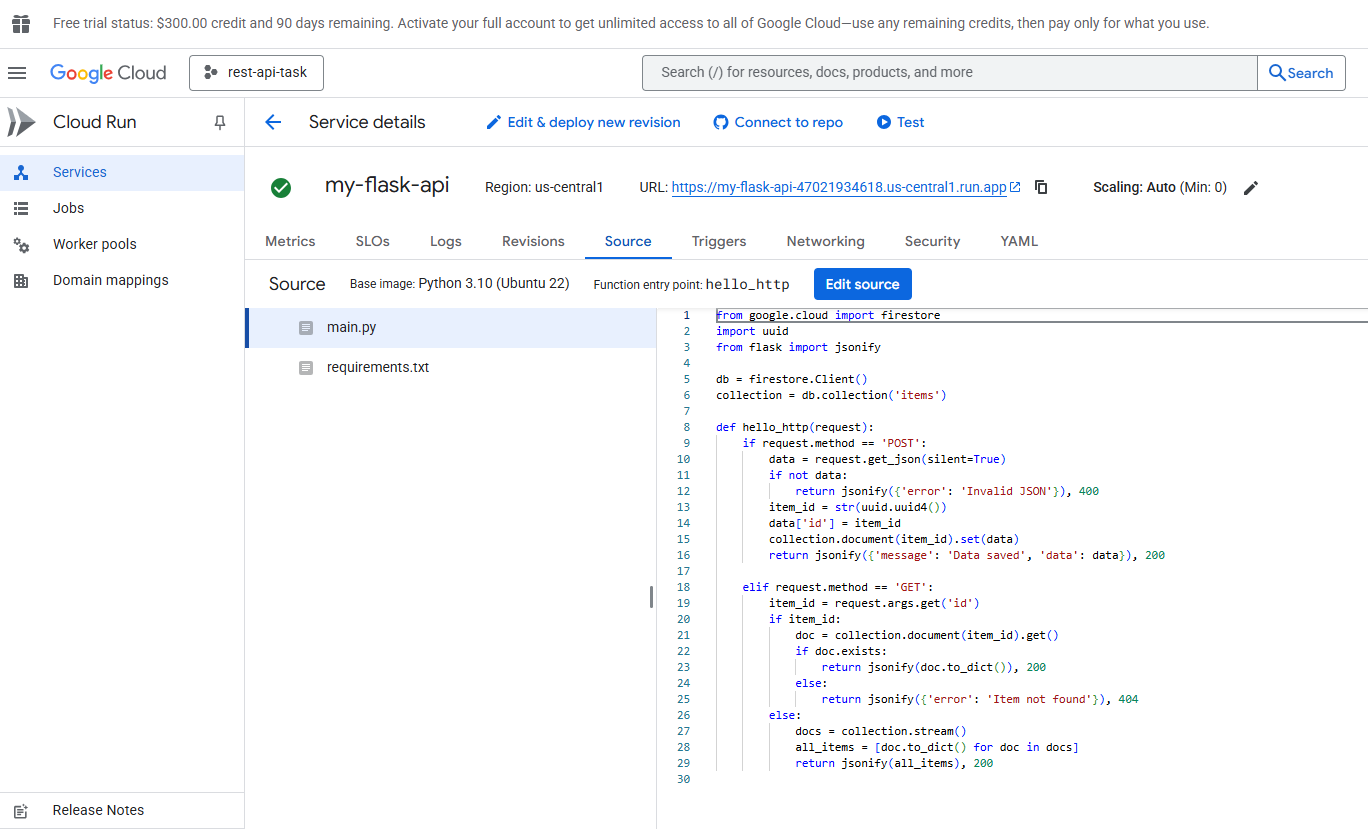
4) Supports **multi-region replication** for high availability and low latency;

5) **No need to provision** or manage servers, clusters, or scaling logic;

6) Scales **automatically** based on traffic and data.



**The function itself:**

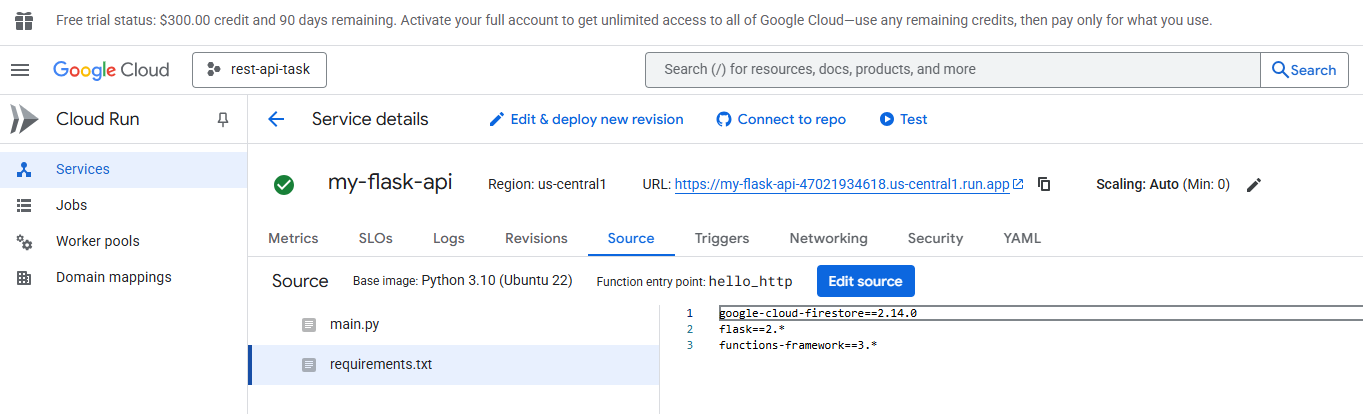


**Requirements (**Stack Summary**):**

**google-cloud-firestore==2.14.0 (**Real-time, NoSQL document database)

**flask==2.\* (**Lightweight web framework, Handles HTTP requests in Cloud Functions, Modern features and async support)

**functions-framework==3.\* (**Google’s tool to run functions locally and deploy to Cloud, Compatible with Flask and Firestore, Enables local testing with functions-framework)



**Results**

**Cloud shell commands:**

POST new data:

**curl -X POST "https://my-flask-api-47021934618.us-central1.run.app" \**

**-H "Authorization: Bearer $(gcloud auth print-identity-token)" \**

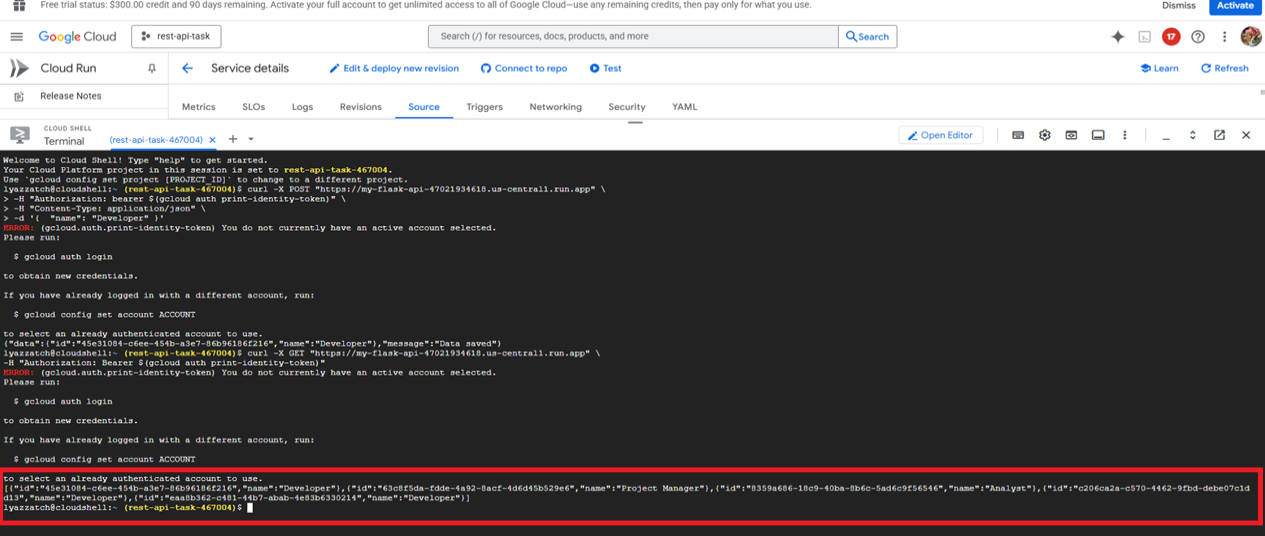
**-H "Content-Type: application/json" \**

**-d '{"name": "Analyst"}'**

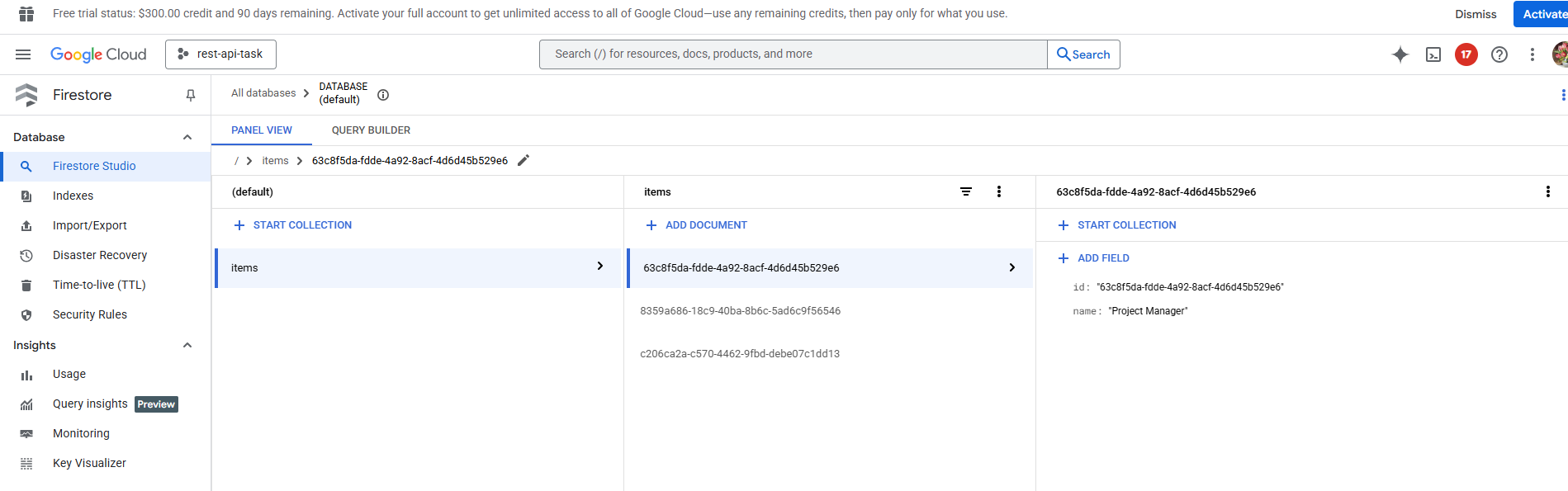
GET all data:

**curl -X GET "https://my-flask-api-47021934618.us-central1.run.app" \**

**-H "Authorization: Bearer $(gcloud auth print-identity-token)"**



**In database:**



**Future enhancements**

It's pretty much feasible to add functionality (e.g., PUT, DELETE, or PATCH) and deployment automatization.