Getting Started with API Gateway: Challenge Lab

Objective

To develop a backend system using **Cloud Run functions**, deploy and manage an **API using API Gateway** to expose the backend, and **publish messages to a Pub/Sub topic** using the API backend in Google Cloud.

Tools & Technologies Used

- **Google Cloud Platform (GCP)**
- Cloud Run (2nd Gen)
- API Gateway
- Pub/Sub
- Node.js 22
- OpenAPI Specification (swagger 2.0)
- Cloud Shell / gcloud CLI

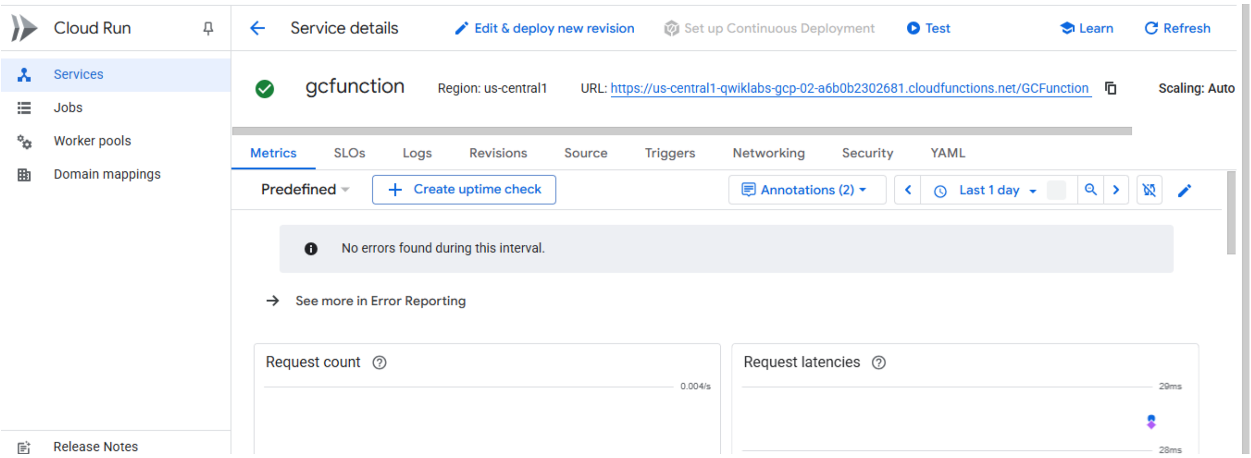
Challenge Scenario

As a **junior data analyst**, you are assisting a newly formed development team to:

1. Expose backend services as APIs using **API Gateway**.
2. React to events using **Pub/Sub**.
3. Extend learned skills by debugging and managing API resources in a live cloud environment.

Task 1: Create a Cloud Run Function

- **Created a Cloud Run function (2nd Gen)** named gcfunction in the **us-central1** region using **Node.js 22**.
- Allowed **unauthenticated invocation** for public API testing.



gcfunction Region: us-central1 URL: https://us-central1-qwiklabs-gcp-02-a6b0b2302681.cloudfunctions.net/GCFunction Scaling: Auto

No errors found during this interval.

Task 2. Create an API Gateway

Once the Cloud Run function is deployed, configure an API Gateway to proxy requests to the backend.

Create a file named `openapispec.yaml` (using the code below), which references the Cloud Run function deployed in Task 1.

Use `openapispec.yaml` when deploying the API Gateway with the following properties:

Name	Value
Display Name	gcfunction API (wherever requested)
API ID	gcfunction-api
Select a service account	Compute Engine default service account
Location	`us-central1`

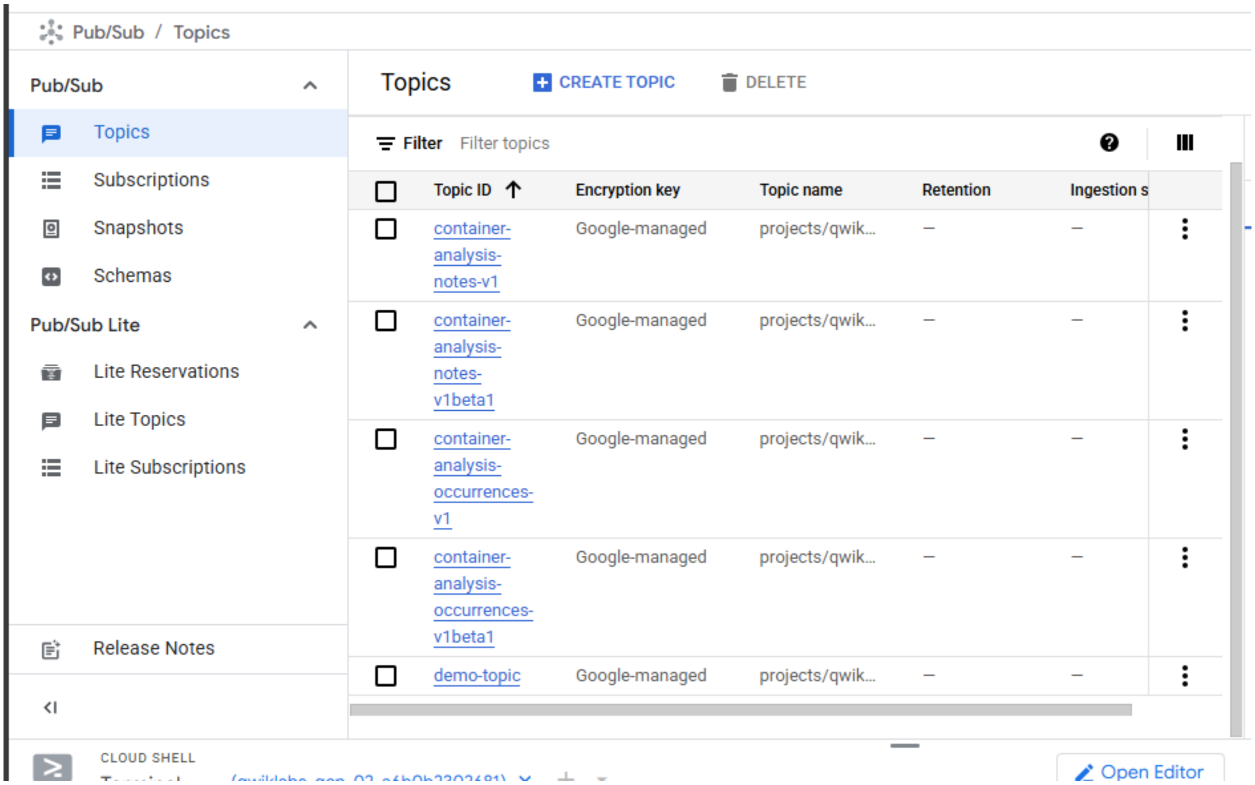
Config Name	gcfunction-api

Deployed the **API Gateway** with:

- Display Name: gcfunction API
- API ID: gcfunction-api
- Config Name: gcfunction-api
- Location: us-central1
- Service Account: Compute Engine default

Task 3: Create a Pub/Sub Topic and Publish Messages

- Created a **Pub/Sub topic** named demo-topic with the default subscription enabled.



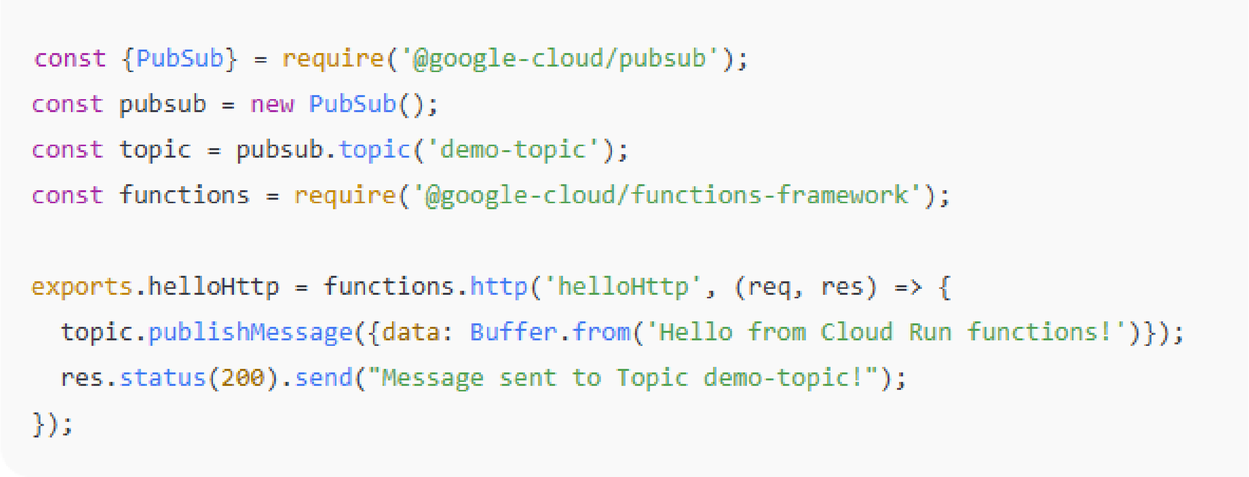
Topics

□	Topic ID ↑	Encryption key	Topic name	Retention	Ingestion s
□	container-analysis-notes-v1	Google-managed	projects/qwik...	–	–
□	container-analysis-notes-v1beta1	Google-managed	projects/qwik...	–	–
□	container-analysis-occurrences-v1	Google-managed	projects/qwik...	–	–
□	container-analysis-occurrences-v1beta1	Google-managed	projects/qwik...	–	–
□	demo-topic	Google-managed	projects/qwik...	–	–

- Updated **package.json**:

```
{
  "dependencies": {
   "@google-cloud/functions-framework": "^3.0.0",
   "@google-cloud/pubsub": "^3.4.1"
  }
}
```

Updated **index.js**:



```
const {PubSub} = require('@google-cloud/pubsub');
const pubsub = new PubSub();
const topic = pubsub.topic('demo-topic');
const functions = require('@google-cloud/functions-framework');

exports.helloHttp = functions.http('helloHttp', (req, res) => {
  topic.publishMessage({data: Buffer.from('Hello from Cloud Run functions!')});
  res.status(200).send("Message sent to Topic demo-topic!");
});
```

- **Redeployed the Cloud Run function** with these changes.
- Invoked the Cloud Run function **via API Gateway endpoint**.
- Verified message delivery under **Pub/Sub > Subscriptions > demo-topic > Messages** after ~5 minutes.
- **Assessment completed and passed.**

Outcome

✅ Successfully **exposed backend services using API Gateway**.
✅ Validated **Cloud Run function deployment and invocation**.
✅ Successfully **published messages to Pub/Sub via API backend**.

All objectives were completed within the lab timeframe, demonstrating applied skills in:

- Serverless backend management.
- API exposure and management using API Gateway.
- Event-driven architecture using Pub/Sub.