Multi-Cloud Interoperability Report: AWS & GCP

Executive Summary

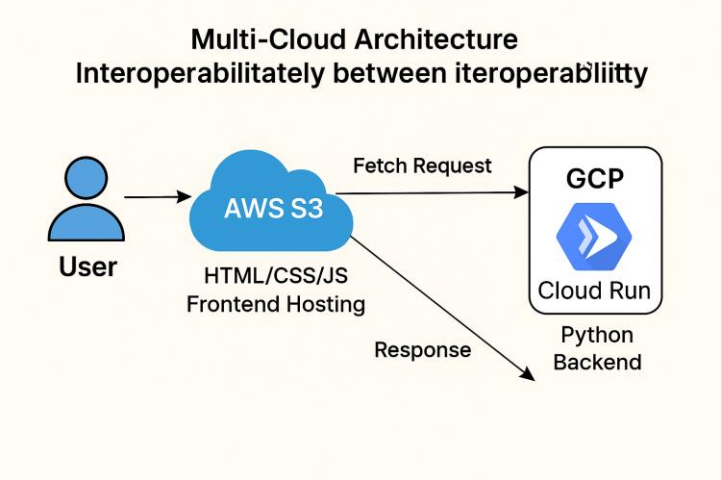
This report documents and demonstrates a practical implementation of multi-cloud interoperability using Amazon Web Services (AWS) and Google Cloud Platform (GCP). The project showcases a seamless interaction between a static frontend hosted on AWS S3 and a backend API deployed using GCP Cloud Run. This integration highlights the ability to leverage the strengths of different cloud platforms within a single application architecture.

Project Objective

The primary objective of this project is to showcase cross-platform cloud integration, allowing components of a web application to run on different cloud service providers. Specifically, the frontend is hosted on AWS S3, while the backend API is powered by GCP Cloud Run. The goal is to demonstrate how to achieve secure and functional communication between the two services.

Architecture Overview

The following architecture represents the layout of the project. A static frontend is served from an AWS S3 bucket with static website hosting enabled. This frontend includes JavaScript that makes a network request to an API endpoint hosted on GCP Cloud Run. The backend processes the request and returns a response, which is displayed on the frontend.



Technologies Used

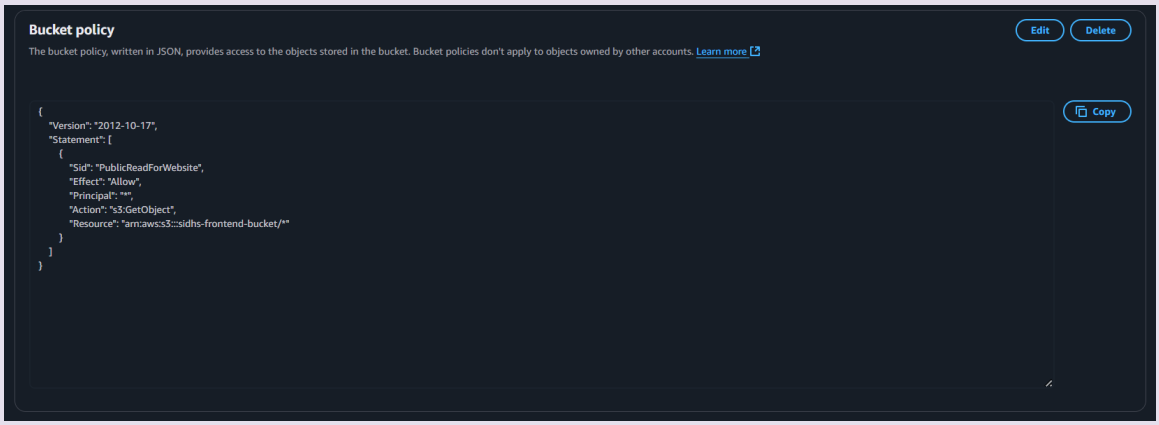
- AWS S3 - Static website hosting
- GCP Cloud Run - Python backend API
- HTML, CSS, JS - Frontend technologies
- CORS - For cross-domain communication
- Python Functions Framework

Implementation Details

1. Frontend (AWS S3)

The frontend is a static HTML page hosted in an S3 bucket with static website hosting enabled. It contains a button that, when clicked, makes a fetch call to the backend hosted on GCP.

AWS S3 Bucket Settings:

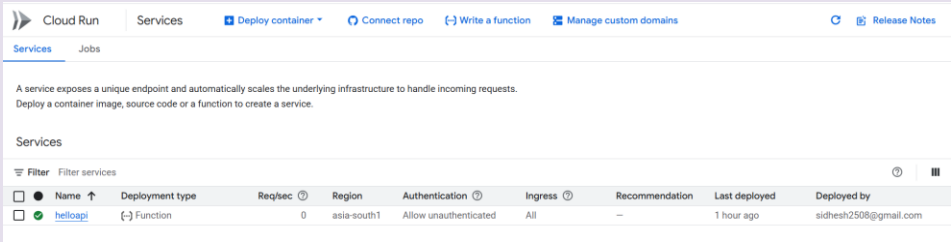


index.html Website in Browser:

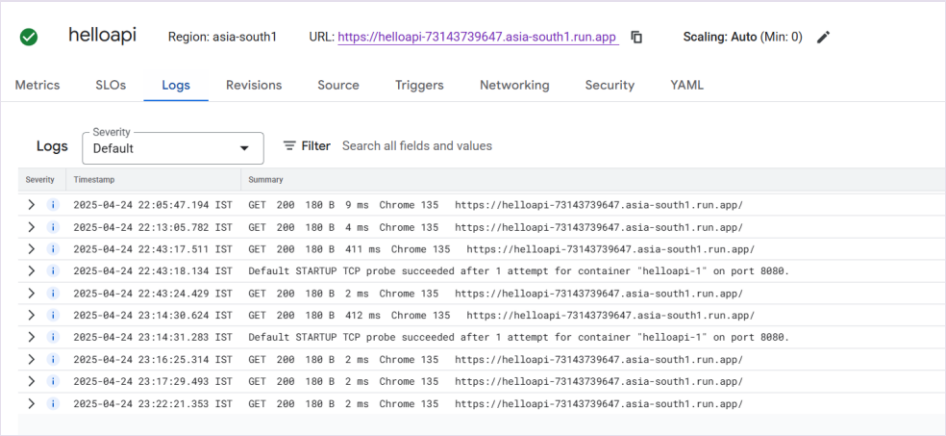
2. Backend (GCP Cloud Run)

The backend is a Python function deployed on GCP Cloud Run. It includes CORS headers to handle requests from other origins. The function returns a simple message which the frontend displays to the user.

GCP Cloud Run Service Page:



Logs Showing API Trigger:



Key Code Snippets

Backend: main.py

```
def hello_world(request):
  if request.method == 'OPTIONS':
    headers = {
      'Access-Control-Allow-Origin': '*',
      'Access-Control-Allow-Methods': 'GET, OPTIONS',
      'Access-Control-Allow-Headers': 'Content-Type',
    }
    return ('', 204, headers)

  headers = {
    'Access-Control-Allow-Origin': '*',
  return ('Hello from GCP backend!', 200, headers)
```

Frontend: index.html (JavaScript Snippet)

```
function callBackend() {
  const output = document.getElementById("output");
  output.innerText = "Calling backend...";
  fetch("https://helloapi-73143739647.asia-south1.run.app")
    .then(response => {
      if (!response.ok) throw new Error("Network response was not ok");
      return response.text();
    })
    .then(data => {
      output.innerText = "Response from GCP: " + data;
    })
    .catch(error => {
      output.innerText = "Error: " + error.message;
    });
}
```

Live Demo Instructions

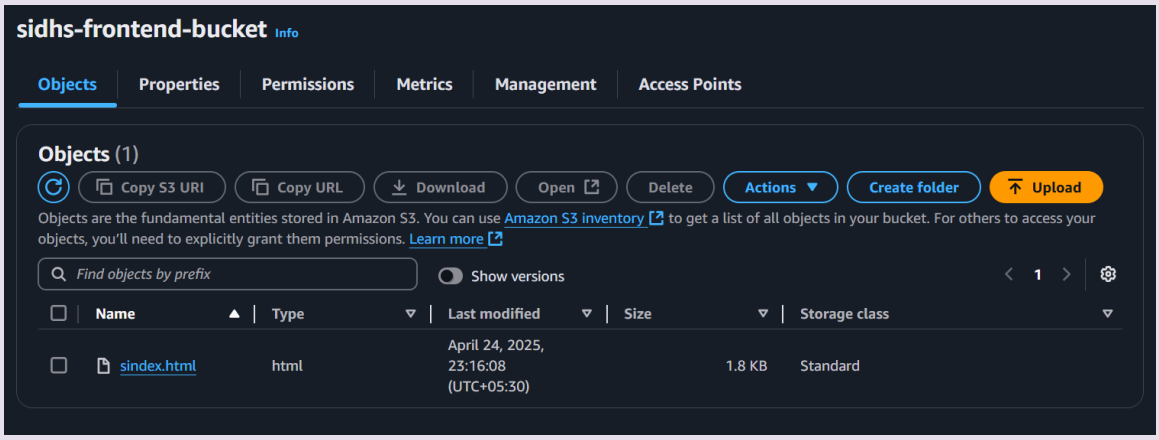
1. Open the hosted frontend URL from AWS S3 in a browser.
2. Click the 'Call GCP Backend' button.
3. The browser sends a fetch request to the GCP Cloud Run backend.
4. The backend responds with a message which is shown on the webpage.

Conclusion

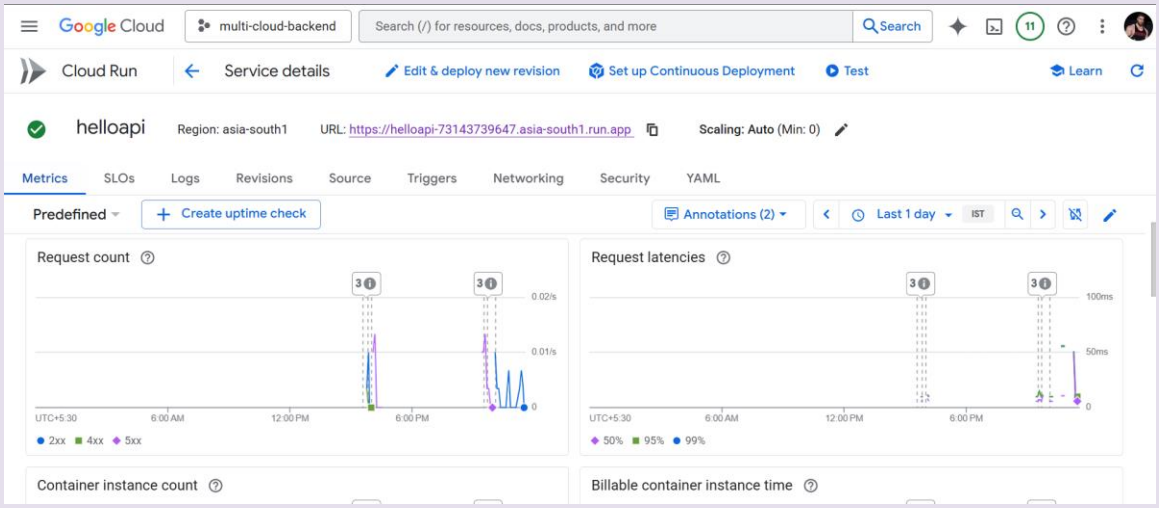
This project successfully demonstrates how two major cloud platforms can interoperate to build a functioning web application. By combining AWS for frontend hosting and GCP for backend services.

Appendix: Screenshots

AWS S3 BUCKET:



GCP Cloud Run Deployment View:



BROWSER SHOWING FRONTEND CALL RESULT: