Report for Assignment: Counting Lines in a File using Google Cloud Functions

Due Date: 2024-06-23, 23:59 IST

Objective: The goal of this assignment is to create a Google Cloud Function that counts the number of lines in a text file stored in Google Cloud Storage (GCS). The function should be triggered by any change in the GCS bucket, read the content of the file, and print the line count.

Solution Overview: The solution involves creating a Google Cloud Function named `ga` that is triggered by changes in a specified GCS bucket. The function reads the content of the file, counts the number of lines, and prints the count.

Components

1. **Google Cloud Storage (GCS):** A service for storing and accessing data on Google Cloud Platform.
2. **Google Cloud Functions:** A serverless execution environment for building and connecting cloud services.
3. **Python:** The programming language used for the Cloud Function.

Steps to Implement the Solution

1. **Setup Google Cloud Function:**
    - Ensure you have a Google Cloud project with billing enabled.
    - Enable the Google Cloud Functions and Cloud Storage APIs.
2. **Grant Necessary Permissions:**
    - Grant the `Storage Object Viewer` role to the service account used by the Cloud Function to ensure it can read files from the GCS bucket.
3. **Write the Cloud Function Code:**
    - Create a Python function that handles the Cloud Event triggered by changes in the GCS bucket.
    - The function reads the file from the bucket, counts the lines, and prints the result.
4. **Deploy the Cloud Function:**
    - Use the `gcloud` command-line tool to deploy the Cloud Function with the appropriate trigger and service account.

Python Code for Cloud Function

python
Copy code

```
import functions_framework
from google.cloud import storage
```

```
# Triggered by a change in a storage bucket
@functions_framework.cloud_event
def ga(cloud_event):
    data = cloud_event.data
    bucket = data["bucket"]
    name = data["name"]

    storage_client = storage.Client()
    bucket_obj = storage_client.bucket(bucket)
    txt_file = bucket_obj.blob(name)

    if not txt_file.exists():
        print(f"There is no file named {name} in the bucket {bucket}")
        return

    txt_file_content = txt_file.download_as_text()
    lines = txt_file_content.split('\n')
    count = len(lines)

    print(f"The number of lines in {name} is {count}")
```

Deployment Instructions

Authenticate and Set Up GCP:

bash

Copy code

```
gcloud auth login
gcloud config set project [YOUR_PROJECT_ID]
```

1.

Deploy the Cloud Function:

bash

Copy code

```
gcloud functions deploy toji_zenin \
    --runtime python39 \
    --trigger-resource [YOUR_BUCKET_NAME] \
    --trigger-event google.storage.object.finalize \
    --service-account [PROJECT_ID]@appspot.gserviceaccount.com
```

2.

Testing the Cloud Function

1. **Upload a File to GCS:**
    - Upload a text file to the specified GCS bucket.
    - This should trigger the Cloud Function.
2. **Check Logs:**
    - Go to the Google Cloud Console.
    - Navigate to Cloud Functions and view the logs for the `toji_zenin` function.
    - Verify that the function prints the correct line count for the uploaded file.

Creating the Report and Submission

1. **Report Content:**
    - Describe the objective, solution overview, and implementation steps.
    - Include the Python code for the Cloud Function.
    - Provide deployment and testing instructions.
2. **Create the ZIP File:**
    - Include the report (PDF) and the Python source code in a ZIP file named `21f3002806assignment_week_3.zip`.
3. **Upload to Google Form:**
    - Fill in the required details in the Google Form.

Conclusion

This assignment demonstrates the use of Google Cloud Functions to process files stored in Google Cloud Storage. By following the steps outlined in this report, you can successfully deploy a Cloud Function that counts the lines in a text file and prints the result. This solution leverages the serverless capabilities of Google Cloud Functions to automate file processing tasks efficiently.