

Introduction to Big Data

Graded Assignment 1

Name - Avijeet Palit

Roll - 21f1005675

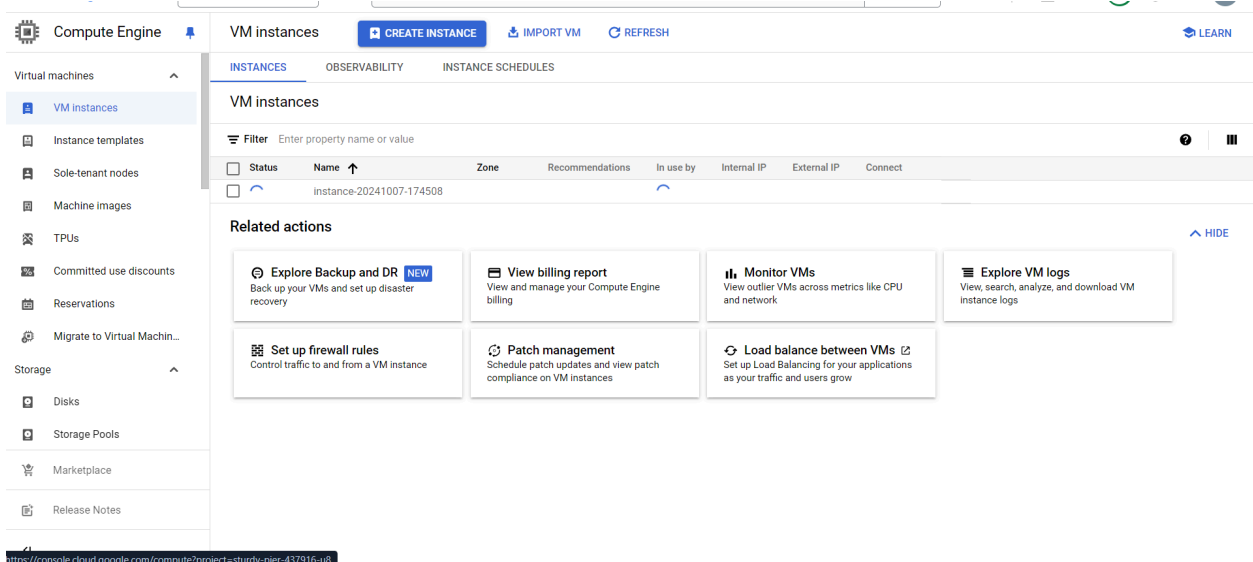
Date - 07/10/2024

This report outlines the workflow for creating a Virtual Machine (VM) in Google Cloud Platform (GCP) and setting up a Cloud Storage bucket for uploading a text file. Additionally, it details the steps to upload a Python script that counts the number of lines in the text file.

To accomplish the tasks in the graded assignments, follow these steps:

Step 1: Create a VM Instance on GCP

1. **Access Google Cloud Console:** Go to the Google Cloud Console.
2. **Navigate to VM Instances:**
 - Click on **Compute Engine** in the sidebar.
 - Select **VM instances**.
3. **Create a New Instance:**
 - Click on **Create Instance**.
 - **Configure the Instance:**
 - Choose the desired machine type.
 - Select the region and zone.
 - Configure any additional settings as needed.
 - Click **Create** to launch the VM.



Step 2: Connect to the VM

1. Open SSH Terminal:

- Once the VM is running, locate your instance in the VM instances list.
- Click on the **SSH** button next to your instance.
- A new window will open with a terminal connected to your VM.

Step 3: Set Up the VM Environment

Install Python and the necessary libraries to interact with Google Cloud Storage (GCS).

Update Packages and Install Python:

```
sudo apt-get update
```

```
sudo apt-get install -y python3-pip
```

Install Google Cloud Storage Library:

```
pip3 install google-cloud-storage
```

Set Up Google Cloud SDK and Authenticate:

```
curl -O
```

```
https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/googl  
e-cloud-sdk-XXX-linux-x86_64.tar.gz
```

```
tar -xf google-cloud-sdk-XXX-linux-x86_64.tar.gz
```

The image shows a terminal window with a dark background. At the top, the terminal's title bar displays the URL: 'ssh.cloud.google.com/v2/ssh/projects/study-pier-437916-u8/zones/us-central1-c/instances/instance-20241007-174508?authuser=081...'. Below the title bar, there's a navigation bar with 'SSH-in-browser', 'UPLOAD FILE', and 'DOWNLOAD FILE' buttons. The main area of the terminal shows the output of a script. The script starts with 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/contrib/_init_.py' and continues with a list of various test scripts and components being downloaded or installed. The list includes files like 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/contrib/duplicate_san.pem', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/contrib/test_pyopenssl.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/contrib/test_securetransport.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_collections.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_compatibility.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_connection.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_connectionpool.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_exceptions.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_fields.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_filepost.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_no_ssl.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_poolmanager.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_proxymanager.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_queue_monkeypatch.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_response.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_retry.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_retry_deprecated.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_ssl.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_ssltransport.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_util.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/test_wait.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/tz_stub.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/with_dummyserver/_init_.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/with_dummyserver/test_chunked_transfer.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/with_dummyserver/test_connectionpool.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/with_dummyserver/test_https.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/with_dummyserver/test_no_ssl.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/with_dummyserver/test_no_proxy.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/with_dummyserver/test_proxy_poolmanager.py', 'google-cloud-sdk/platform/gutil/third_party/urllib3/test/with_dummyserver/test_socketlevel.py', 'google-cloud-sdk/properties', 'google-cloud-sdk/rpm/mapping/command_mapping.yaml', 'google-cloud-sdk/rpm/mapping/component_mapping.yaml', and 'avi/jetclouds/instance-20241007-174508-5'. The terminal output is in a light blue font on a dark background. The bottom of the terminal window shows a 'Release Notes' button. The overall interface is clean and modern, typical of a web-based terminal environment.

1. Create the Python Script:

- On your VM, Upload a new file named `ga_1.py`.

```
ssh.cloud.google.com/v2/ssh/projects/sturdy-pier-437916-u8/zones/us-central1-c/instances/instance-20241007-174508?authuser=0&hl=en_US&projectNumber=584144288540&useAdminProxy=true&pageViewId=E192A100-795A-...
SSH-in-browser
Linux instance-20241007-174508 6.1.0-25-cloud-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.106-3 (2024-08-26) x86_64
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Oct 7 18:04:51 2024 from 35.235.244.33
avi@cloud:~$ ls
google-cloud-sdk  google-cloud-sdk-xxx-linux-x86_64.tar.gz
google-cloud-sdk-494.0.0-linux-x86_64.tar.gz
avi@cloud:~$ cat ga_1.py
from google.cloud import storage

def download_file_from_gcs(bucket_name, source_blob_name, destination_file_name):
    """Downloads a file from GCS."""
    storage_client = storage.Client()
    bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob(source_blob_name)
    blob.download_to_filename(destination_file_name)
    print(f"File {source_blob_name} downloaded to {destination_file_name}.")

def count_lines_in_file(file_path):
    """Counts the number of lines in a file."""
    with open(file_path, 'r') as file:
        line_count = sum(1 for line in file)
    return line_count

if __name__ == "__main__":
    bucket_name = 'ga_1'
    source_blob_name = 'ga_1.txt'
    destination_file_name = '/tmp/result'

    # Download the file from GCS
    download_file_from_gcs(bucket_name, source_blob_name, destination_file_name)

    # Count lines in the downloaded file
    line_count = count_lines_in_file(destination_file_name)
    print(f"The file has {line_count} lines.")
avi@cloud:~$
```

Transferred 1 item
ga_1.py

2. Create a Cloud Storage Bucket:

- In the Google Cloud Console, navigate to **Cloud Storage > Buckets**.
- Click on **Create Bucket**.
- **Name the Bucket:** Enter **ga_21** as the bucket name.
- Configure any additional settings as needed.
- Click **Create** to finalize the bucket creation.

3. Upload the Text File:

- Within the bucket **ga_12**, click on **Upload Files**.
- Select the **ga_1.txt** file from your local machine.
- Upload the file, which contains the content to be read by your Python script.

Google Cloud | My First Project | buck

Cloud Storage | Bucket details

Location: us (multiple regions in United States) | Storage class: Standard | Public access: Not public | Protection: Soft Delete

OBJECTS | CONFIGURATION | PERMISSIONS | PROTECTION | LIFECYCLE | OBSERVABILITY | INVENTORY REPORTS | OPERATIONS

Folder browser

Buckets > ga_12

CREATE FOLDER | UPLOAD | TRANSFER DATA | OTHER SERVICES

Filter by name prefix only | Filter objects and folders | Show Live objects only

Name	Size	Type	Created	Storage class	Last modified
ga_1.txt	52 B	text/plain	Oct 7, 2024, 11:31:59 PM	Standard	Oct 7, 2024, 11:3

1 file successfully uploaded

Uploads and My First Project operations

ga_1.txt Complete

Step 5: Run the Script

```
SSH-in-browser
UPLOAD FILE | DOWNLOAD FILE | [Icons]

avijeetcloud@instance-20241007-174508:~$ cat ga_1.py
from google.cloud import storage
def download_file_from_gcs(bucket_name, source_blob_name, destination_file_name):
    """Downloads a file from GCS."""
    storage_client = storage.Client()
    bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob(source_blob_name)
    blob.download_to_filename(destination_file_name)
    print(f"File {source_blob_name} downloaded to {destination_file_name}.")

def count_lines_in_file(file_path):
    """Counts the number of lines in a file."""
    with open(file_path, 'r') as file:
        line_count = sum(1 for line in file)
    return line_count

if __name__ == "__main__":
    bucket_name = 'ga_12'
    source_blob_name = 'ga_1.txt'
    destination_file_name = '/tmp/result'

    # Download the file from GCS
    download_file_from_gcs(bucket_name, source_blob_name, destination_file_name)

    # Count lines in the downloaded file
    line_count = count_lines_in_file(destination_file_name)
    print(f"The file has {line_count} lines.")
avijeetcloud@instance-20241007-174508:~$
```

Execute the Python Script:

`python3 ga_1.py`

Script Functionality:

- The script will download the specified file from the GCS bucket to the VM's local filesystem (e.g., `/tmp/results`).

- It will then count the number of lines in the file and output the result.

Result:

```
avijeetcloud@instance-20241007-174508:~$ python3 ga_1.py
File ga_1.txt downloaded to /tmp/result.
The file has 9 lines.
avijeetcloud@instance-20241007-174508:~$
```