**4.IAM Hands-on**

|  |  |
| --- | --- |
| Concept of GCP Resource Availability and access | Concept of IAM |
| hierarchy |
| Organization, folders, projects |
| Roles |
| Members |
| Service Accounts |
| Privileges |
| primitive roles, predefined roles, custom roles |
| Understanding of Regions and Zones |

**5.Cloud Storage Hands-on**

|  |  |
| --- | --- |
| Working with Buckets - GUI | Create a bucket |
| Upload an object into the bucket |
| Download the object |
| Share the object |
| Create folders |
| Delete the objects |
| Object Lifecycle |
| Clean up |
| Working with Buckets - Console(GSUTIL) | Create a bucket |
| Upload an object into your bucket |
| Download the object from your bucket |
| Copy the object to a folder in the bucket |
| List contents of a bucket or folder |
| List details for an object |
| Delete an object |
| Clean up |
|  |  |
| Working with Buckets - Python API Calls | Create a bucket |
| Upload an object into your bucket |
| Download the object from your bucket |
| Copy/Move the object to/from a folder in/from the bucket |
| List contents of a bucket or folder |
| List details for an object |
| Delete an object |
| Clean up |

Object versioning (For version control)

For restoring deleted or overwritten objects. To minimize the cost of storing versions, we recommend limiting the number of noncurrent versions per object and scheduling them to expire after a number of

**Lifecycle Management**: GCS allows you to define lifecycle policies to automatically manage objects based on their age or other criteria. For example, you can set a policy to automatically transition objects to a lower-cost storage class after a certain period of time, or to delete objects after they have been stored for a specified duration.

**Working with Buckets - Console(GSUTIL)**

Certainly! Let’s break down the **gsutil** commands for each of the tasks you mentioned:

**Create a Bucket**: To create a new bucket, use the following command:

gsutil mb -c STANDARD -l asia-south1 gs://demobuck\_2

1. **Upload an Object into Your Bucket**: To upload a local file to your bucket, use the following command:

gsutil cp /home/sdrizwan2113/storage1/number.txt gs://demobuck\_2/

Replace local-file with the path to your local file and your-bucket-name with the target bucket.

1. **Download an Object from Your Bucket**: To download an object from your bucket, use the following command:

gsutil cp gs://demobuck\_2/number.txt /home/sdrizwan2113/storage1/newnumber1.txt

Replace object-name with the name of the object in your bucket and local-file with the desired local file path.

1. **Copy an Object to a Folder in the Bucket**: To copy an object within the same bucket or to a different folder, use:
2. gsutil cp gs://source-bucket/source-object gs://destination-bucket/destination-folder/

Replace source-bucket, source-object, destination-bucket, and destination-folder accordingly.

1. **List Contents of a Bucket or Folder**: To list the objects in a bucket or a specific folder, use:
2. gsutil ls gs://your-bucket-name/

Replace your-bucket-name with the bucket you want to list.

1. **List Details for an Object**: To get detailed information about an object, use:
2. gsutil stat gs://demobuck\_2/number.txt

Replace object-name with the name of the object.

1. **Delete an Object**: To delete an object from your bucket, use:
2. gsutil rm gs://demobuck\_2/number.txt

Again, replace object-name with the actual object name.

1. **Clean Up**: If you want to remove a bucket entirely, including all its objects, use:
2. gsutil rm -r gs://your-bucket-name

Be cautious with this command, as it permanently deletes the bucket and its contents.

Remember to replace placeholders like your-bucket-name, object-name, and local-file with your specific values.

**Working GUI STORAGE**

Certainly! Let’s dive into the detailed steps for each task when working with **Google Cloud Storage (GCS)** using the **Google Cloud Console** (GUI). I’ll provide clear instructions to help you understand the process:

1. **Create a Bucket**:
   * **Log in** to your Google Cloud Console.
   * Click on the **Navigation Menu** (☰) and select **Storage > Browser**.
   * Click the **Create Bucket** button.
   * Provide a **unique Bucket Name** (e.g., “my-awesome-bucket”).
   * Choose a **Location** (region) for your bucket.
   * Configure any other settings (such as storage class, access control, etc.) as needed.
   * Click **Create** to create your bucket.
2. **Upload an Object into Your Bucket**:
   * Navigate to your bucket in the Cloud Console.
   * Click the **Upload Files** button.
   * Select the file you want to upload from your local machine.
   * Click **Open** to upload the file to your bucket.
3. **Download an Object**:
   * Navigate to your bucket.
   * Find the object you want to download.
   * Click the three vertical dots (⋮) next to the object’s name.
   * Choose **Download** to save the object to your local machine.
4. **Share an Object**:
   * To share an object publicly:
     + Select the object in your bucket.
     + Click the **Permissions** tab.
     + Add a new permission with the **Entity** set to **AllUsers** and the **Role** set to **Reader**.
     + Save the changes.
   * Now anyone with the object’s URL can access it.
5. **Create Folders**:
   * GCS doesn’t have traditional folders, but you can create a pseudo-folder structure using object names with slashes (e.g., folder/subfolder/object.txt).
   * When uploading an object, include the desired folder structure in the object name.
6. **Delete Objects**:
   * Navigate to your bucket.
   * Select the object(s) you want to delete.
   * Click the trash can icon (🗑️) to delete the object(s).
7. **Object Lifecycle**:
   * You can set lifecycle policies to automatically delete or transition objects based on rules (e.g., delete objects older than a certain age).
8. **Clean Up**:
   * To delete the entire bucket:
     + Navigate to the bucket.
     + Click the trash can icon (🗑️) next to the bucket name.
     + Confirm the deletion.

Remember to replace placeholders like **Bucket Name** and **Object Names** with your specific values.

**Using Python API Calls**

Here are the Python API calls for working with Google Cloud Storage (GCS) buckets:

**Create a Bucket**:

from google.cloud import storage

# Initialize a storage client

storage\_client = storage.Client()

# Create a new bucket

bucket\_name = "ali\_34"

bucket = storage\_client.bucket(bucket\_name)

bucket.storage\_class = "STANDARD"

new\_bucket = storage\_client.create\_bucket(bucket, location="asia-south1")

# Print a success message

print("Bucket created successfully.")

1. **Upload an Object into Your Bucket**:
2. from google.cloud import storage
3. storage\_client = storage.Client()
4. bucket = storage\_client.bucket("al\_34")
5. blob = bucket.blob("number.txt")
6. blob.upload\_from\_filename("number.txt")
7. print("file uploaded sucessfully")
8. **Download the Object from Your Bucket**:

from google.cloud import storage

storage\_client = storage.Client()

bucket = storage\_client.bucket("al\_34")

blob = bucket.blob("number.txt")

blob.download\_to\_filename("down.txt")

print("Blob number downloaded to downFolder.")

1. **Copy/Move the Object to/from a Folder in/from the Bucket**:
2. def copy\_blob(bucket\_name, blob\_name, destination\_bucket\_name, destination\_blob\_name):
3. storage\_client = storage.Client()
4. source\_bucket = storage\_client.bucket(bucket\_name)
5. source\_blob = source\_bucket.blob(blob\_name)
6. destination\_bucket = storage\_client.bucket(destination\_bucket\_name)
7. new\_blob = source\_bucket.copy\_blob(source\_blob, destination\_bucket, destination\_blob\_name)
8. print(f"Blob {blob\_name} in bucket {bucket\_name} copied to blob {destination\_blob\_name} in bucket {destination\_bucket\_name}.")

**List Contents of a Bucket or Folder**:

from google.cloud import storage

storage\_client = storage.Client()

blobs = storage\_client.list\_blobs("al\_34")

for blob in blobs:

    print(blob.name)

**List Details for an Object**:

from google.cloud import storage

storage\_client = storage.Client()

bucket = storage\_client.bucket("al\_34")

blob = bucket.blob("number.txt")

print(f"Blob: {blob.name}")

print(f"Bucket: {blob.bucket.name}")

print(f"Storage class: {blob.storage\_class}")

print(f"ID: {blob.id}")

print(f"Size: {blob.size} bytes")

print(f"Updated: {blob.updated}")

print(f"Generation: {blob.generation}")

**Delete an Object**:

from google.cloud import storage

storage\_client = storage.Client()

bucket = storage\_client.bucket("al\_34")

blob = bucket.blob("number.txt")

blob.delete()

print("Blob {number.txt} deleted.")

**Clean Up**:

1. def delete\_bucket(bucket\_name):
2. storage\_client = storage.Client()
3. bucket = storage\_client.bucket(bucket\_name)
4. bucket.delete()
5. print(f"Bucket {bucket\_name} deleted.")