Accessing the Google Cloud Console and Cloud Shell

Overview

In this lab, you become familiar with the Google Cloud web-based interface. Two integrated environments are available:

- A GUI environment called the Cloud Console
- A command-line interface called Cloud Shell, which has the commands from the Cloud SDK pre-installed

In this course, you use both environments.

You need to know a few things about the Cloud Console:

- The Cloud Console is under continuous development, so the graphical layout occasionally changes. Often, these changes are made to accommodate new Google Cloud features or changes in the technology, resulting in a slightly different workflow.
- You can perform most common Google Cloud actions in the Cloud Console. Sometimes new features are implemented in the Cloud SDK before they are made available in the Cloud Console.
- The Cloud Console is extremely fast for some activities. The Cloud Console can perform multiple actions on your behalf that might require many command-line actions.
- The commands in the Cloud SDK are valuable tools for automation.

Objectives

In this lab, you learn how to perform the following tasks:

- Learn how to access the Cloud Console and Cloud Shell
- Become familiar with the Cloud Console
- Become familiar with Cloud Shell features, including the Cloud Shell code editor
- Use the Cloud Console and Cloud Shell to create buckets and VMs and service accounts
- Perform other commands in Cloud Shell

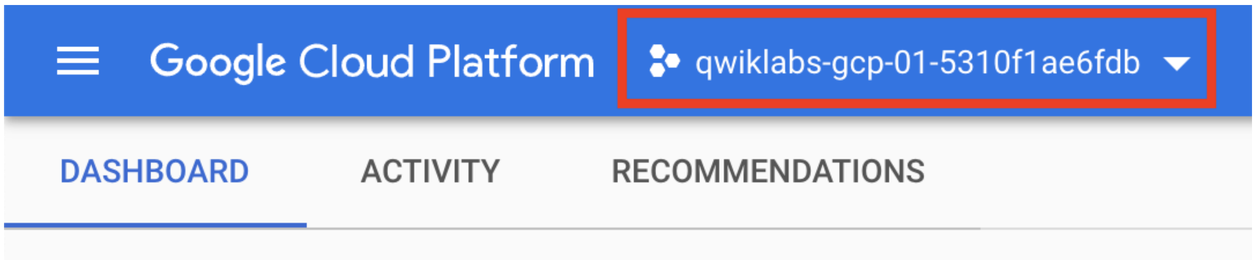
Task 1. Explore the Google Cloud Console

In this task, you explore the Cloud Console and create resources.

Verify that your project is selected

1. In the **Select a project** drop-down list in the title bar select the project ID that Qwiklabs provided with your authentication credentials.
2. The project ID will resemble **qwiklabs-gcp-** followed by a long hexadecimal number.
3. Click **Cancel** to close the dialog.

Your title bar should indicate the project ID as shown in the screenshot. Each lab in the Qwiklabs environment has a unique project ID, as well as unique authentication credentials.



Navigate to Google Cloud Storage and create a bucket

Cloud Storage allows world-wide storage and retrieval of any amount of data at any time. You can use Cloud Storage for a range of scenarios including serving website content, storing data for archival and disaster recovery, or distributing large data objects to users via direct download.

Cloud Storage buckets must have a globally unique name. In your organization, you should follow Google Cloud's Best practices for Cloud Storage Guide. For this lab, we can easily get a unique name for our bucket by using the ID of the Google Cloud project that Qwiklabs created for us, because Google Cloud project IDs are also globally unique.

1. In the Cloud Console, on the **Navigation menu** (≡), click **Cloud overview > Dashboard** .
2. In the **Dashboard** tab of the resulting screen, the **Project info** section shows your Google Cloud project ID.
3. Select and copy the project ID. Because this project ID was created for you by Qwiklabs, it will resemble **qwiklabs-gcp-** followed by a long hexadecimal number.
4. In the Cloud Console, on the **Navigation menu** (≡), click **Cloud Storage** > **Buckets**.
5. Click **Create bucket**.
6. For **Name**, paste in the Google Cloud project ID string you copied in an earlier step. These lab instructions will later refer to the name that you typed as [BUCKET_NAME].
7. Click **Continue**.
8. Click on **Choose how to control access to objects** and uncheck **Enforce public access prevention on this bucket**, now select **Fine-grained**.
9. Click **Continue**.
10. Leave all other values as their defaults.
11. Click **Create**.

Note: The Cloud Console has a **Notifications** (🔔) icon. Feedback from the underlying commands is sometimes provided there. You can click the icon to check the notifications for additional information and history.

Create a virtual machine (VM) instance

Google Compute Engine offers virtual machines running in Google's datacenters and on its network as a service. Google Kubernetes Engine makes use of Compute Engine as a component of its architecture. For this reason, it's helpful to learn a bit about Compute Engine before learning about Kubernetes Engine.

1. On the **Navigation menu** (☰), click **Compute Engine** > **VM instances**.
2. Click **Create Instance**.
3. For **Name**, type `first-vm` as the name for your instance.
4. For **Region**, select **us-central1**.
5. For **Zone**, select **us-central1-c**.
6. For **Machine type**, examine the options.

Note: The machine type menu lists the number of virtual CPUs, the amount of memory, and a symbolic name such as *e1-standard-1*. The symbolic name is the parameter you use to select the machine type when using the `gcloud` command to create a VM. To the right of the region, zone, and machine type is a per-month estimated cost.

7. To see the breakdown of estimated costs, click **Details** to the right of the **Machine type** list underneath the estimated costs.
8. For **Machine type**, click **2 vCPUs (e2-standard-2)**.

How did the cost change?

9. For **Machine type,** click **e2-micro (2 shared vCPU)**.

The micro type is a shared-core VM that is inexpensive.

10. For **Firewall**, click **Allow HTTP traffic**.
11. Leave the remaining settings as their defaults, and click **Create**.

Wait until the new VM is created.

Explore the VM details

1. On the **VM instances** page, click the name of your VM, `first-vm`.
2. In **Machine configuration**, notice the value of **CPU platform**, notice the value, and click **Edit**.
3. Click on the pencil icon on the top to edit the **first-vm** instance.

Note: You can't change the machine type, the CPU platform, or the zone of a running Google Cloud VM. You can add network tags and allow specific network traffic from the internet through firewalls.
Some properties of a VM are integral to the VM and are established when the VM is created. They cannot be changed. Other properties can be edited. For example, you can add disks, and you can determine whether the boot disk is deleted when the instance is deleted.

4. Scroll down to **Management** section and examine **Availability policies**.

Note: Compute Engine offers preemptible VM instances, which cost less per hour but can be terminated by Google Cloud at any time. These preemptible instances can save you a lot of money, but you must make sure that your workloads are suitable to be interrupted.
You can't convert a non-preemptible instance into a preemptible one. This choice must be made at VM creation.
If a VM is stopped for any reason (for example, an outage or a hardware failure), the automatic restart feature starts it back up. Is this the behavior you want? Are your applications idempotent (written to handle a second startup properly)?

During host maintenance, the VM is set for live migration. However, you can have the VM terminated instead of migrated.
If you make changes, they can sometimes take several minutes to be implemented, especially if they involve networking changes, like adding firewalls or changing the external IP.

5. Click **Cancel**.

Create an IAM service account

An IAM service account is a special type of Google account that belongs to an application or a virtual machine, instead of to an individual end user.

1. On the **Navigation menu**, click **IAM & admin** > **Service accounts**.
2. Click **+ Create service account**.
3. On the **Service account details** page, specify the **Service account name** as `test-service-account`.
4. Click **Create and Continue**.
5. On the **Grant this service account access to project** page, specify the role as **Basic** > **Editor**.
6. Click **Continue**.
7. Click **Done**.
8. On the **Service accounts** page, move to the extreme right of the `test-service-account` and click on the three dots.
9. Click **Manage keys**.
10. Click **ADD KEY**
11. Select **Create new key**
12. Select **JSON** as the key type.
13. Click **Create**.

A JSON key file is downloaded. In a later step, you find this key file and upload it to the VM.

14. Click **Close**.

Task 2. Explore Cloud Shell

Cloud Shell provides you with command-line access to your cloud resources directly from your browser. With Cloud Shell, Cloud SDK command-line tools such as gcloud are always available, up to date, and fully authenticated.

Cloud Shell provides the following features and capabilities:

- Temporary Compute Engine VM
- Command-line access to the instance through a browser
- 5 GB of persistent disk storage (`$HOME dir`)
- Preinstalled Cloud SDK and other tools
- `gcloud`: for working with Compute Engine, Google Kubernetes Engine (GKE) and many Google Cloud services
- `gsutil`: for working with Cloud Storage
- `kubectl`: for working with GKE and Kubernetes
- `bq`: for working with BigQuery
- Language support for Java, Go, Python, Node.js, PHP, and Ruby
- Web preview functionality

- Built-in authorization for access to resources and instances

After 1 hour of inactivity, the Cloud Shell instance is recycled. Only the `/home` directory persists. Any changes made to the system configuration, including environment variables, are lost between sessions.

In this task, you use Cloud Shell to create and examine some resources.

Open Cloud Shell and explore its features

1. On the Cloud Console title bar, click **Activate Cloud Shell** ( ).
2. When prompted, click **Continue**.

Cloud Shell opens at the bottom of the Cloud Console window.

The following icons are on the far right of Cloud Shell toolbar:

- **Hide/Restore:** This icon hides and restores the window, giving you full access to the Cloud Console without closing Cloud Shell.
- **Open in new window:** Having Cloud Shell at the bottom of the Cloud Console is useful when you are issuing individual commands. But when you edit files or want to see the full output of a command, clicking this icon displays Cloud Shell in a full-sized terminal window.
- **Close all tabs:** This icon closes Cloud Shell. Everytime you close Cloud Shell, the virtual machine is recycled and all machine context is lost. However, data that you stored in your home directory is still available to you the next time you start Cloud Shell.

Use Cloud Shell to set up the environment variables for this task

In Cloud Shell, use the following commands to define the environment variables used in this task.

1. Replace **[BUCKET_NAME]** with the name of the first bucket from task 1.
2. Replace **[BUCKET_NAME_2]** with a globally unique name of your choice.
3. In Cloud Shell, execute the following command to create environment variables:

```
MY_BUCKET_NAME_1=[BUCKET_NAME]
```

```
MY_BUCKET_NAME_2=[BUCKET_NAME_2]
```

```
MY_REGION=us-central1
```

Note: When you are working in the Cloud Shell or writing scripts, creating environment variables is a good practice. You can easily and consistently re-use these environment variables, which makes your work less error-prone.

Note: Make sure you replace the full placeholder string, such as `[BUCKET_NAME]` with the unique name that you choose, for example `MY_BUCKET_NAME_1=unique_bucket_name.`

Move the credentials file you created earlier into Cloud Shell

You downloaded a JSON-encoded credentials file in an earlier task when you created your first Cloud IAM service account.

1. On your local workstation, locate the JSON key that you just downloaded and rename the file to `credentials.json`
2. In Cloud Shell, click the three dots (⋮) icon in the Cloud Shell toolbar to display further options.
3. Click **Upload** and choose `credentials.json` file from your local machine.
4. Click **Open**.
5. Click **Upload**, to transfer `credentials.json` to Cloud Shell VM.
6. Click the **X** icon to close the file upload pop-up window.
7. In Cloud Shell, type **ls** and press ENTER to confirm that the file was uploaded.

Create a second Cloud Storage bucket and verify it in the Cloud Console

The `gsutil` command, which is supplied by the Cloud SDK, lets you work with Cloud Storage from the command line. In this task, you use the `gsutil` command in Cloud Shell.

1. In Cloud Shell, use the `gsutil` command to create a bucket:

```
gsutil mb gs://$MY_BUCKET_NAME_2
```

Click **Authorize** if prompted.

2. In the Cloud Console, on the **Navigation menu** (☰), click **Cloud Storage** > **Buckets**, or click **Refresh** if you are already in the Cloud Storage page.

The second bucket should appear in the **Buckets** list.

Use the gcloud command line to create a second virtual machine

1. In Cloud Shell, execute the following command to list all the zones in a given region:

```
gcloud compute zones list | grep $MY_REGION
```

2. Select a zone from the first column of the list. Notice that Google Cloud zones' names consist of their region name, followed by a hyphen and a letter.

You may choose a zone that is the same as or different from the zone that you used for the first VM in task 1.

3. Execute the following command to store your chosen zone in an environment variable.

You replace `[ZONE]` with your selected zone:

```
MY_ZONE=[ZONE]
```

4. Set this zone to be your default zone by executing the following command:

```
gcloud config set compute/zone $MY_ZONE
```

5. Execute the following command to store a name in an environment variable you will use to create a VM. You will call your second VM `second-vm`:

```
MY_VMNAME=second-vm
```

6. Create a VM in the default zone that you set earlier in this task using the new environment variable to assign the VM name:

```
gcloud compute instances create $MY_VMNAME \
--machine-type "e2-standard-2" \
--image-project "debian-cloud" \
--image-family "debian-11" \
--subnet "default"
```

7. List the virtual machine instances in your project:

```
gcloud compute instances list
```

You will see both your newly created and your first virtual machine in the list.

8. In the Cloud Console, on the **Navigation menu** (☰), click **Compute Engine** > **VM Instances**. Just as in the output of `gcloud compute instances list`, you will see both of the virtual machines you created.
9. Look at the `External IP` column. Notice that the external IP address of the first VM you created is shown as a link. (If necessary, click the `HIDE INFO PANEL`

button to reveal the `External IP` column.) The Google Cloud Console offers the link because you configured this VM's firewall to allow HTTP traffic.

10. Click the link you found in your first VM's `External IP` column. Your browser will present a `Connection refused` message in a new browser tab. This message occurs because, although there is a firewall port open for HTTP traffic to your VM, no Web server is running there. Close the browser tab you just created.

Use the gcloud command line to create a second service account

1. In Cloud Shell, execute the following command to create a new service account:

```
gcloud iam service-accounts create test-service-account2 --display-name
"test-service-account2"
```

Note: If you see the following output, type **y** and press **ENTER**:

```
API [iam.googleapis.com] not enabled on project [560255523887]. Would
you like to enable and retry (this will take a few minutes)? (y/N)?
```

2. In the Cloud Console, on the **Navigation menu** (☰), click **IAM & admin** > **Service accounts**.

Note: Refresh the page till you see **test-service-account2**.

3. In Cloud Shell, execute the following command to grant the second service account the Project viewer role:

```
gcloud projects add-iam-policy-binding $GOOGLE_CLOUD_PROJECT --member
serviceAccount:test-service-account2@${GOOGLE_CLOUD_PROJECT}.iam.gservicea
ccount.com --role roles/viewer
```

Copied!

content_copy

Note: `GOOGLE_CLOUD_PROJECT` is an environment variable that is automatically populated in Cloud Shell and is set to the project ID of the current context.

4. In the Cloud Console, on the **Navigation menu** (☰), click **IAM & admin** > **IAM**.
5. You will see the new service account called **test-service-account2** listed as a member of the Viewer role.

Task 3. Work with Cloud Storage in Cloud Shell

Download a file to Cloud Shell and copy it to Cloud Storage

1. Copy a picture of a cat from a Google-provided Cloud Storage bucket to your Cloud Shell:

```
gsutil cp gs://cloud-training/ak8s/cat.jpg cat.jpg
```

2. Copy the file into the first buckets that you created earlier:

```
gsutil cp cat.jpg gs://$MY_BUCKET_NAME_1
```

3. Copy the file from the first bucket into the second bucket:

```
gsutil cp gs://$MY_BUCKET_NAME_1/cat.jpg gs://$MY_BUCKET_NAME_2/cat.jpg
```

4. In the Cloud Console, on the **Navigation menu**(≡), click **Cloud Storage** > **Bucket**, select both the buckets that you created, and verify that both contain the `cat.jpg` file.

Set the access control list for a Cloud Storage object

1. To get the default access list that's been assigned to `cat.jpg` (when you uploaded it to your Cloud Storage bucket), execute the following two commands:
2. Execute the following command in Cloud Shell:

```
gsutil acl get gs://$MY_BUCKET_NAME_1/cat.jpg  > acl.txt
cat acl.txt
```

The output should look like the following example, but with different numbers. This output shows that anyone with a Project Owner, Editor, or Viewer role for the project has access (Owner access for Owners/Editors and Reader access for Viewers).

```
[
  {
    "entity": "project-owners-560255523887",
    "projectTeam": {
      "projectNumber": "560255523887",
      "team": "owners"
    },
    "role": "OWNER"
  },
  {
    "entity": "project-editors-560255523887",
    "projectTeam": {
      "projectNumber": "560255523887",
      "team": "editors"
    },
    "role": "OWNER"
  },
  {
    "entity": "project-viewers-560255523887",
    "projectTeam": {
      "projectNumber": "560255523887",
      "team": "viewers"
    },
    "role": "READER"
  },
  {
    "email": "google12345678_student@qwiklabs.net",
    "entity": "user-google12345678_student@qwiklabs.net",
    "role": "OWNER"
  }
]
```

3. To change the object to have private access, execute the following command:

```
gsutil acl set private gs://$MY_BUCKET_NAME_1/cat.jpg
```

4. To verify the new ACL that's been assigned to `cat.jpg`, execute the following two commands:

```
gsutil acl get gs://$MY_BUCKET_NAME_1/cat.jpg  > acl-2.txt
cat acl-2.txt
```

The output should look similar to the following example.

```
[
  {
    "email": "google12345678_student@qwiklabs.net",
    "entity": "user-google12345678_student@qwiklabs.net",
    "role": "OWNER"
  }
]
```

Now only the original creator of the object (your lab account) has Owner access.

Authenticate as a service account in Cloud Shell

1. In Cloud Shell, execute the following command to view the current configuration:

```
gcloud config list
```

You should see output that looks like the following example. In your output, the zone should be equal to the zone that you set when you created your second VM in task 2. The account and project should match your Qwiklabs lab credentials.

```
[component_manager]
disable_update_check = True
[compute]
gce_metadata_read_timeout_sec = 30
zone = us-central1-a
[core]
account = google12345678_student@qwiklabs.net
disable_usage_reporting = False
project = qwiklabs-Google Cloud-1aeffbc5d0acb416
[metrics]
environment = devshell
Your active configuration is: [cloudshell-16441]
```

2. In Cloud Shell, execute the following command to change the authenticated user to the first service account (which you created in an earlier task) through the

credentials that you downloaded to your local machine and then uploaded into Cloud Shell (`credentials.json`):

```
gcloud auth activate-service-account --key-file credentials.json
```

Cloud Shell is now authenticated as `test-service-account`.

3. To verify the active account, execute the following command:

```
gcloud config list
```

You should see output that looks like the following example. The account is now set to the `test-service-account` service account.

```
[component_manager]
disable_update_check = True
[compute]
gce_metadata_read_timeout_sec = 30
zone = us-central1-a
[core]
account = test-service-account@qwiklabs-Google
Cloud-1aeffbc5d0acb416.iam.gserviceaccount.com
disable_usage_reporting = False
project = qwiklabs-Google Cloud-1aeffbc5d0acb416
[metrics]
environment = devshell
Your active configuration is: [cloudshell-16441]
```

4. To verify the list of authorized accounts in Cloud Shell, execute the following command:

```
gcloud auth list
```

You should see output that looks like the following example.

```
Credentialed Accounts
ACTIVE:
ACCOUNT: student-03-5165fd82c14b@qwiklabs.net
To set the active account, run:
    $ gcloud config set account `ACCOUNT`
```

5. To verify that the current account (`test-service-account`) cannot access the `cat.jpg` file in the first bucket that you created, execute the following command:

```
gsutil cp gs://$MY_BUCKET_NAME_1/cat.jpg ./cat-copy.jpg
```

Because you restricted access to this file to the owner earlier in this task you should see output that looks like the following example.

Output

```
Copying gs://test-bucket-123/cat.jpg...
AccessDeniedException: 403  KiB]
```

6. Verify that the current account (`test-service-account`) can access the `cat.jpg` file in the second bucket that you created:

```
gsutil cp gs://$MY_BUCKET_NAME_2/cat.jpg ./cat-copy.jpg
```

Because access has not been restricted to this file you should see output that looks like the following example.

```
Copying gs://test-bucket-123/cat.jpg...
- [1 files][ 81.7 KiB/ 81.7 KiB]
Operation completed over 1 objects/81.7 KiB.
```

7. To switch to the lab account, execute the following command, replacing `[USERNAME]` with the username provided in the Qwiklabs Connection Details pane on the left of the lab instructions page:

```
gcloud config set account [USERNAME]
```

8. To verify that you can access the `cat.jpg` file in the [BUCKET_NAME] bucket (the first bucket that you created), execute the following command.

```
gsutil cp gs://$MY_BUCKET_NAME_1/cat.jpg ./copy2-of-cat.jpg
```

You should see output that looks like the following example. The lab account created the bucket and object and remained an Owner when the object access control list (ACL) was converted to private, so the lab account can still access the object.

```
Copying gs://test-bucket-123/cat.jpg...
- [1 files][ 81.7 KiB/ 81.7 KiB]
Operation completed over 1 objects/81.7 KiB.
```

9. Make the first Cloud Storage bucket readable by everyone, including unauthenticated users:

```
gsutil iam ch allUsers:objectViewer gs://$MY_BUCKET_NAME_1
```

Note: This is an appropriate setting for hosting public website content in Cloud Storage.

10. In the Cloud Console, on the **Navigation menu** (≡), click **Cloud Storage** > **Buckets**.
11. Select the first storage bucket with project ID that you created. Notice that the `cat.jpg` file has a `Public access`.

12. Click **Copy URL** to copy the link.
13. Open a new incognito browser tab and paste the link into its address bar. You will see a picture of a cat. Leave this browser tab open.

Task 4. Explore the Cloud Shell code editor

In this task, you explore using the Cloud Shell code editor.

Open the Cloud Shell code editor

1. In Cloud Shell, click the **Open Editor** icon (✎) and then click on the **Open in a new window** link.

A new tab opens with the Cloud editor. The Cloud console and cloud shell remains on the original tab. You can switch between the Cloud shell and Code editor by clicking the tab.

2. On the Cloud console tab, click **Open Terminal** and in Cloud Shell, execute the following command to clone a `git` repository:

```
git clone
https://github.com/googlecodelabs/orchestrate-with-kubernetes.git
```

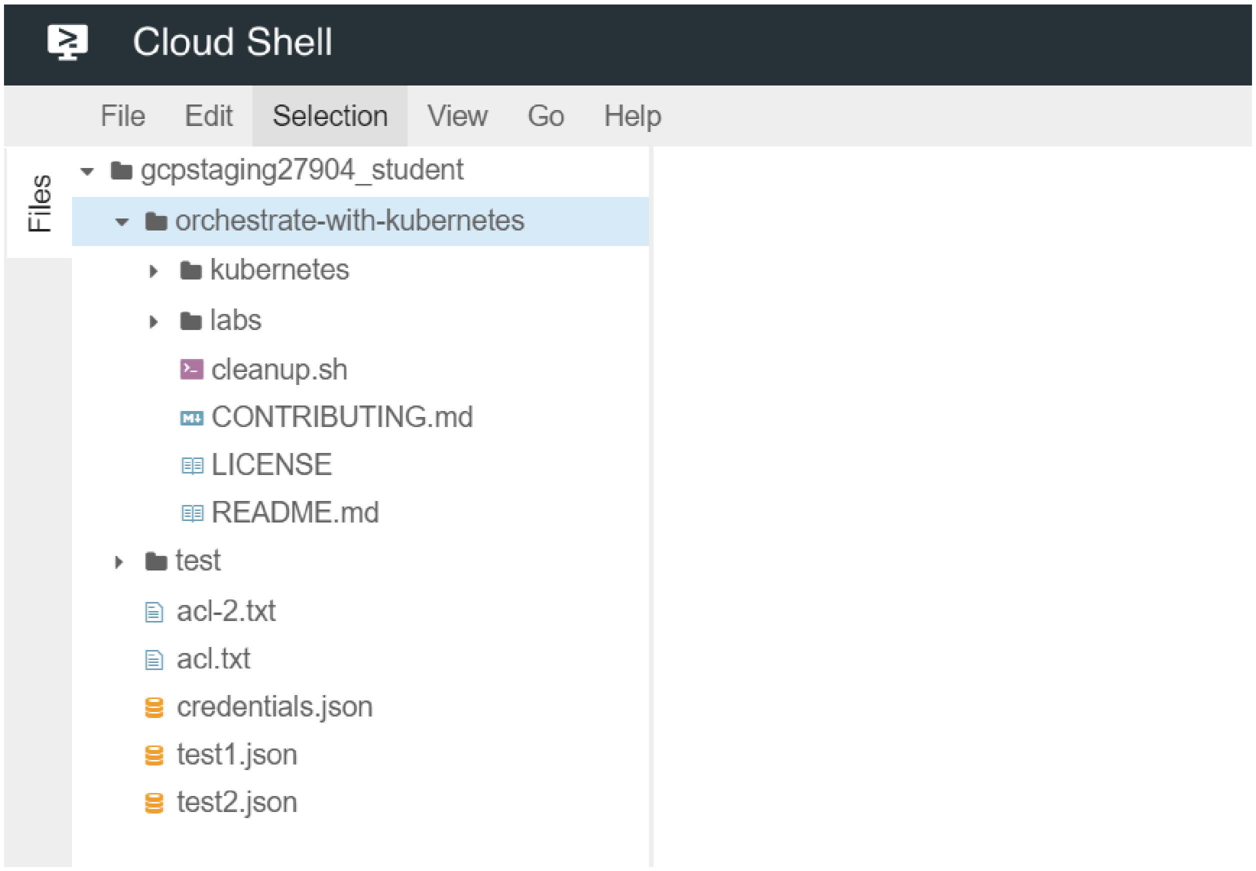
The `orchestrate-with-kubernetes` folder appears in the left pane of the Cloud Shell code editor window.

3. In Cloud Shell, execute the following command to create a test directory:

```
mkdir test
```

The `test` folder now appears in the left pane of the Cloud Shell code editor window.

4. In the Cloud Shell code editor, click the arrow to the left of `orchestrate-with-kubernetes` to expand the folder.



5. In the left pane, click the `cleanup.sh` file to open it in the right pane of the Cloud Shell code editor window.
6. Add the following text as the last line of the `cleanup.sh` file:

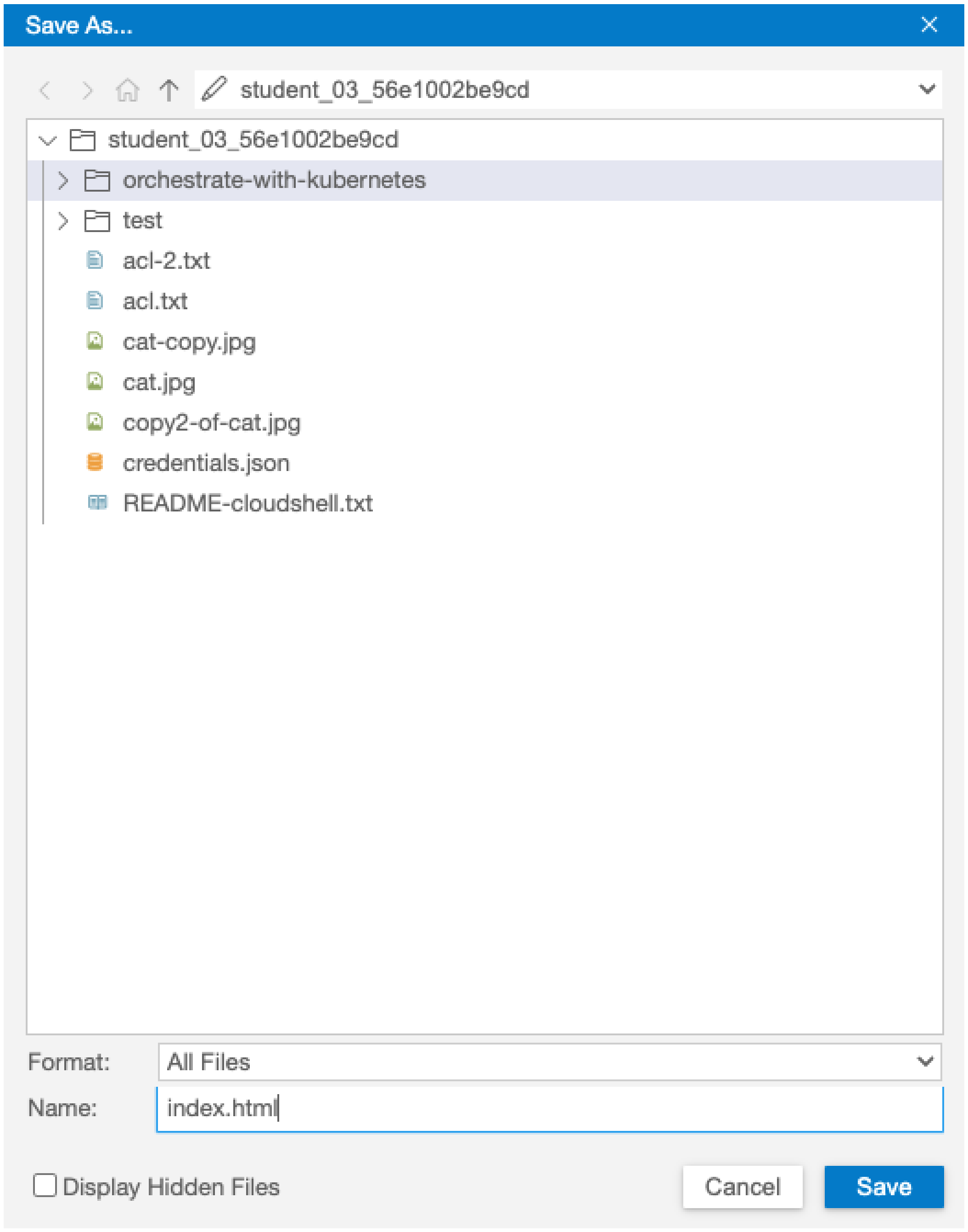
```
echo Finished cleanup!
```

Note: No action is necessary to save your work.

7. In Cloud Shell, execute the following commands to change directory and display the contents of `cleanup.sh`:

```
cd orchestrate-with-kubernetes
cat cleanup.sh
```

8. Verify that the output of `cat cleanup.sh` includes the line of text that you added.
9. In the Cloud Shell code editor, click to open the `File` menu and choose `New File`.
10. Save the file in `orchestrate-with-kubernetes` folder and name the file `index.html`
11. Click **Save**.


Save As...

student_03_56e1002be9cd

student_03_56e1002be9cd
orchestrate-with-kubernetes
test
acl-2.txt
acl.txt
cat-copy.jpg
cat.jpg
copy2-of-cat.jpg
credentials.json
README-cloudshell.txt

Format: All Files

Name: index.html

12. In the right hand pane, paste in this HTML text:

```
<html><head><title>Cat</title></head>
<body>
<h1>Cat</h1>
<img src="REPLACE_WITH_CAT_URL">
```

```
</body></html>
```

Note: Use your local computer's keyboard shortcut to paste: `Cmd-V` for a Mac, `Ctrl-V` for a Windows or Linux machine.

13. Replace the string `REPLACE_WITH_CAT_URL` with the URL of the cat image from an earlier task. The URL will look like this:

```
https://storage.googleapis.com/qwiklabs-Google
Cloud-1aeffbc5d0acb416/cat.jpg
```

14. On the **Navigation menu** (☰) , click **Compute Engine** > **VM instances**.
15. In the row for your **first-vm**, click the `SSH` button.
16. In the SSH login window that opens on your VM, install the `nginx` Web server:

```
sudo apt-get remove -y --purge man-db
sudo touch /var/lib/man-db/auto-update
sudo apt-get update
sudo apt-get install nginx
```

Note: It may take few minutes to complete the process. If prompted, click **Y** to continue.

17. In your Cloud Shell window, copy the HTML file you created using the Code Editor to your virtual machine:

```
gcloud compute scp index.html first-vm:index.nginx-debian.html
--zone=us-central1-c
```

Note: If you are prompted whether to add a host key to your list of known hosts, answer **y**.
Note: If you are prompted to enter a passphrase, press the **Enter** key to respond with an empty passphrase. Press the **Enter** key again when prompted to confirm the empty passphrase.

18. In the **SSH** login window for your VM, copy the HTML file from your home directory to the document root of the `nginx` Web server:

```
sudo cp index.nginx-debian.html /var/www/html
```

19. On the **Navigation menu** (☰), click **Compute Engine** > **VM instances**.
20. Click the link in the `External IP` column for your **first-vm**. A new browser tab opens, containing a Web page that contains the cat image.