Lab 1. Setting up the Learning Environment

First, sign up for Docker Hub at https://hub.docker.com/.

By the end of this lab exercise, you should be able to:

- Create a Google Cloud Platform (GCP) account
- Configure a VM on GCP
- Connect to a VM on GCP
- Install Docker on Linux VM
- Verify that git is installed

Create a Google Cloud Account

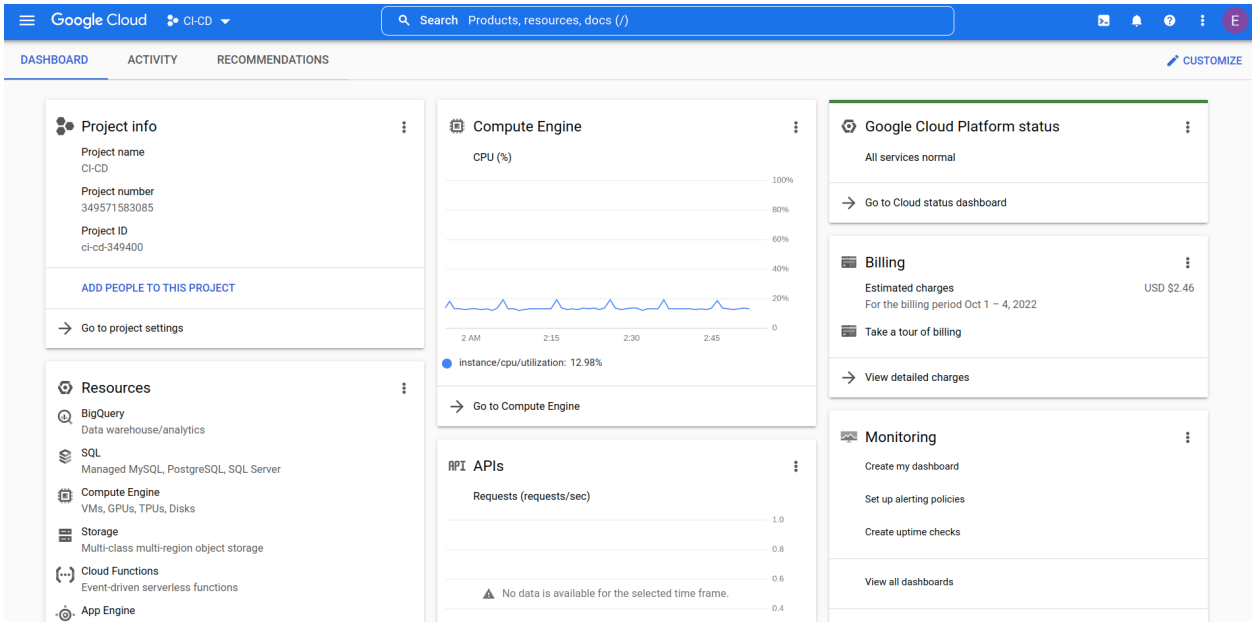
The only prerequisite for this is that you should have a Google account. If you do not have a Google account, go to google.com, click **Sign in** and follow the directions.

Create a Google Cloud Platform Account

Visit cloud.google.com to create a Google Cloud account, get started with a free account and sign up to get free credits which are valid for a limited amount of time.

While creating an account, choose the account type **individual** and provide your address and payment details to complete the signup.

Once you create a Google Cloud Platform account, you will get Console access. You can revisit your account by going to cloud.google.com at any time.



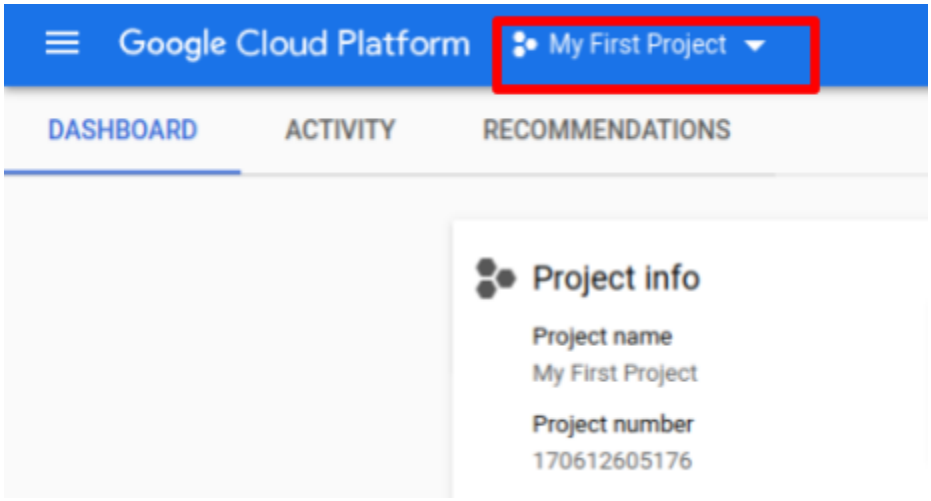
Once logged in, you should see your Google Cloud Console as pictured above.

You can always get to your Google Cloud Console via https://cloud.google.com/ and then click on **Console** in the top left corner.

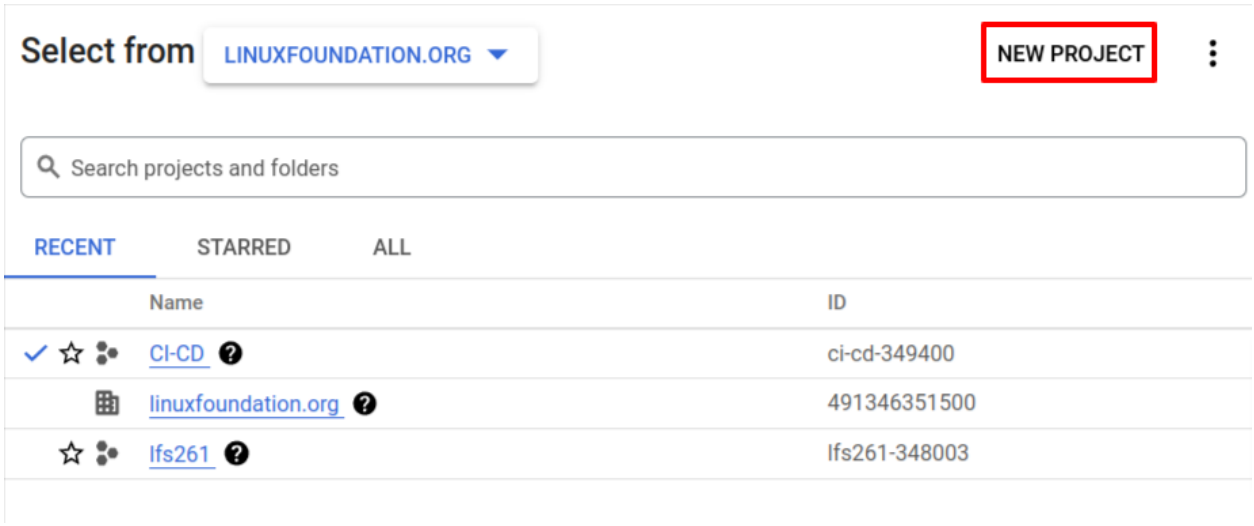
Alternatively you can visit here - https://console.cloud.google.com

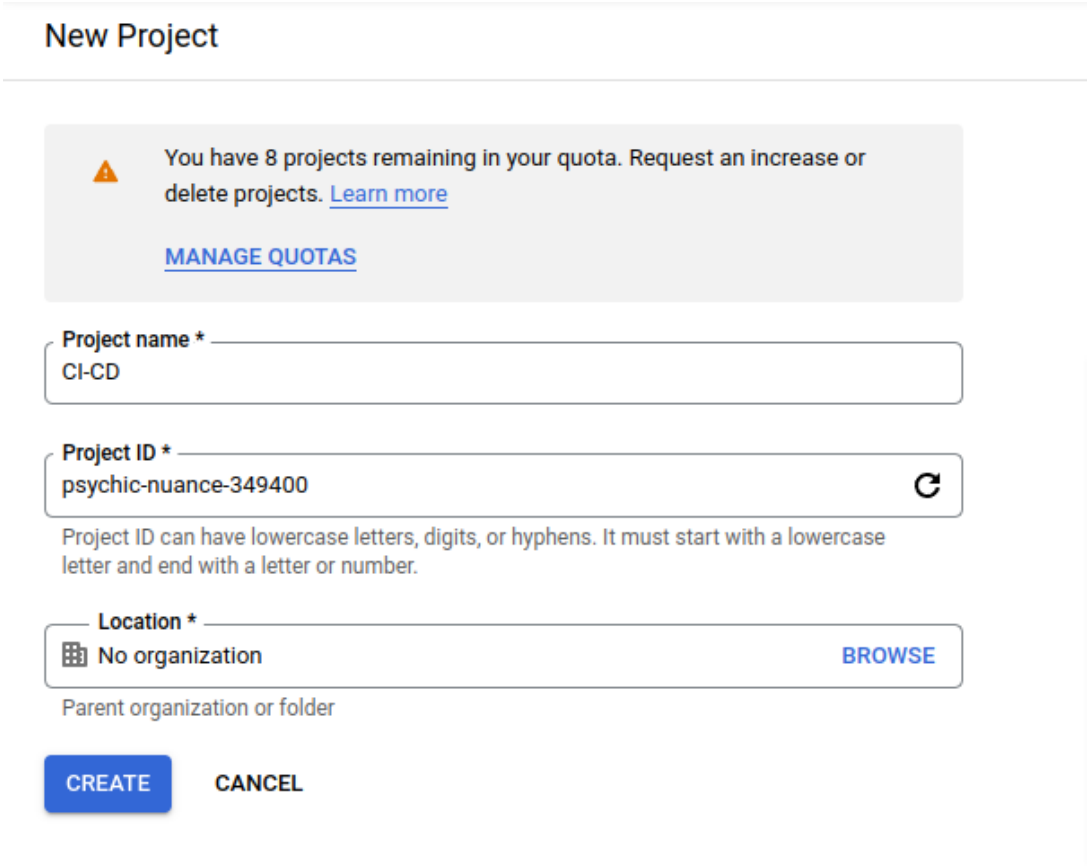
Configuring a VM on Google Cloud

A project provides a namespace to isolate resources for that project. To set up a new project, click on the project dropdown menu. If it is your first time, you will see **My First Project** in the box. Otherwise, you will see the last project that you were in. To create a new project, click inside the box as illustrated below.



Click **NEW PROJECT**

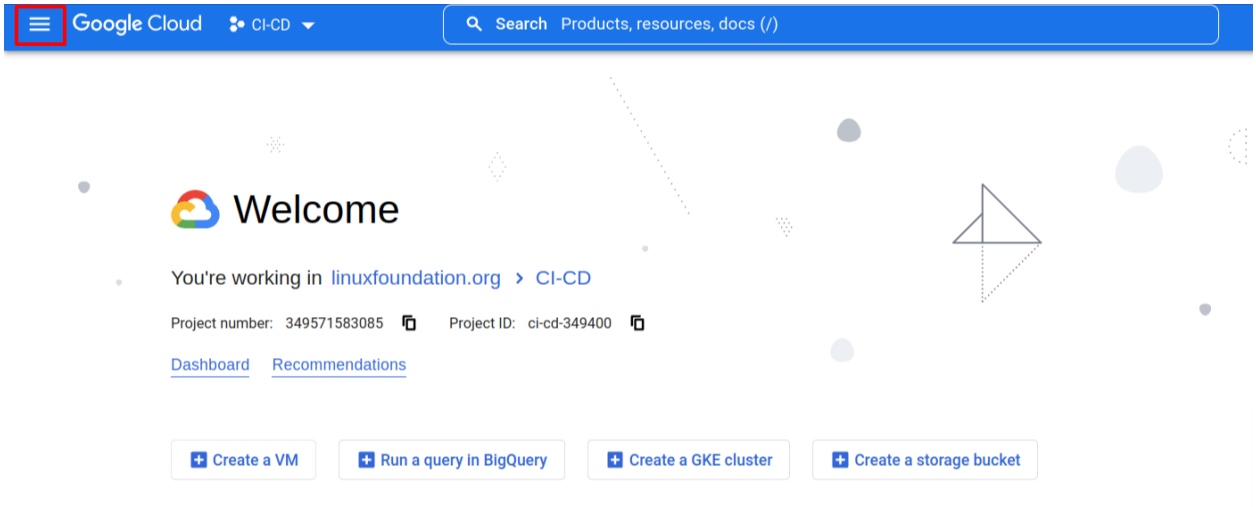




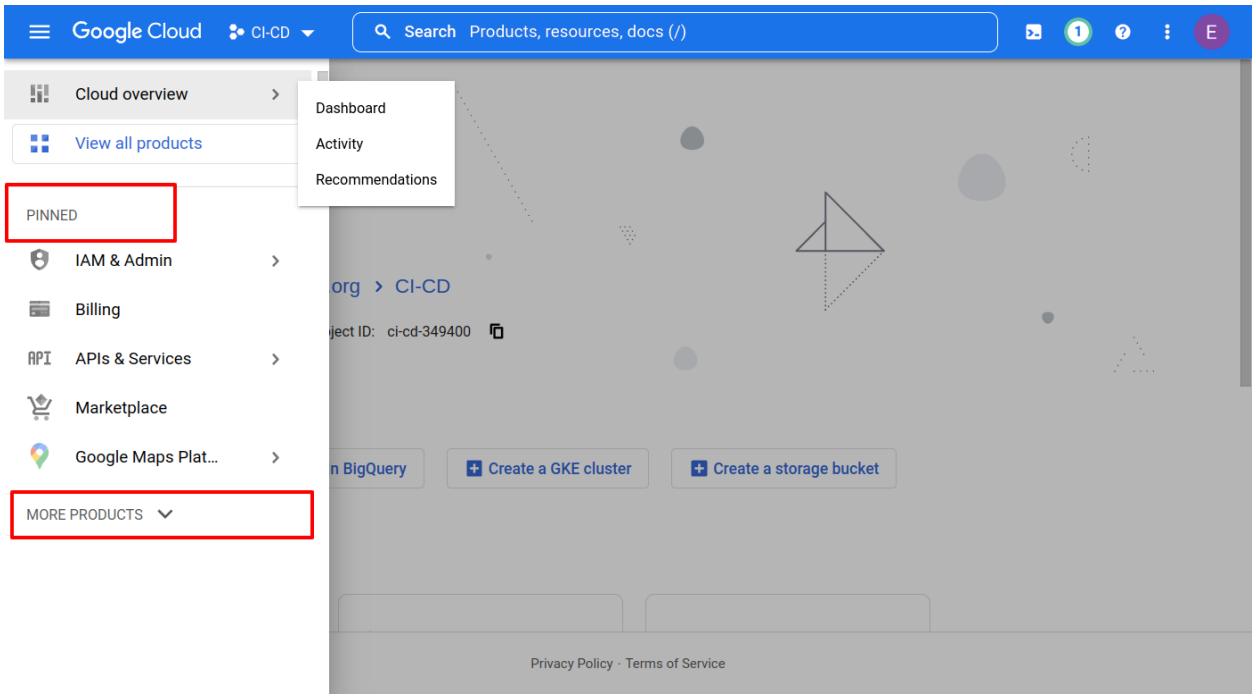
Name the project **CI-CD** and click **CREATE**.

Creating a VM Instance on Google Cloud

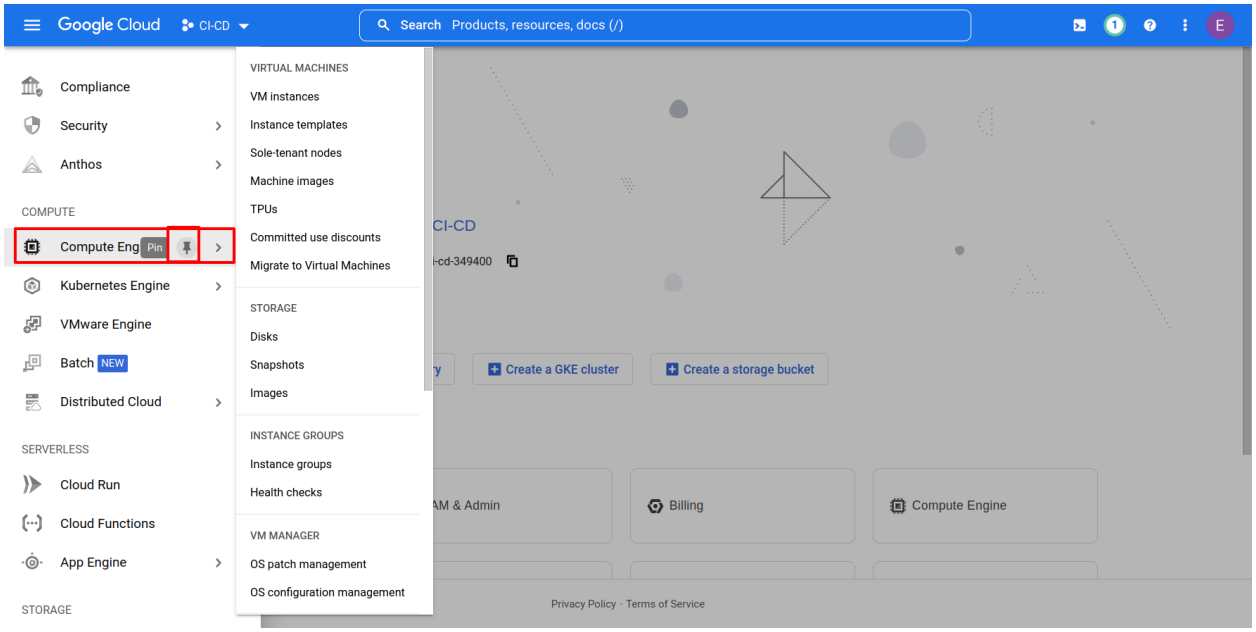
Open the menu by clicking the menu in the top left:



The fly out menu contains two sections. Look for **Compute Engine** either in the **PINNED** section or in the **MORE PRODUCTS** section.



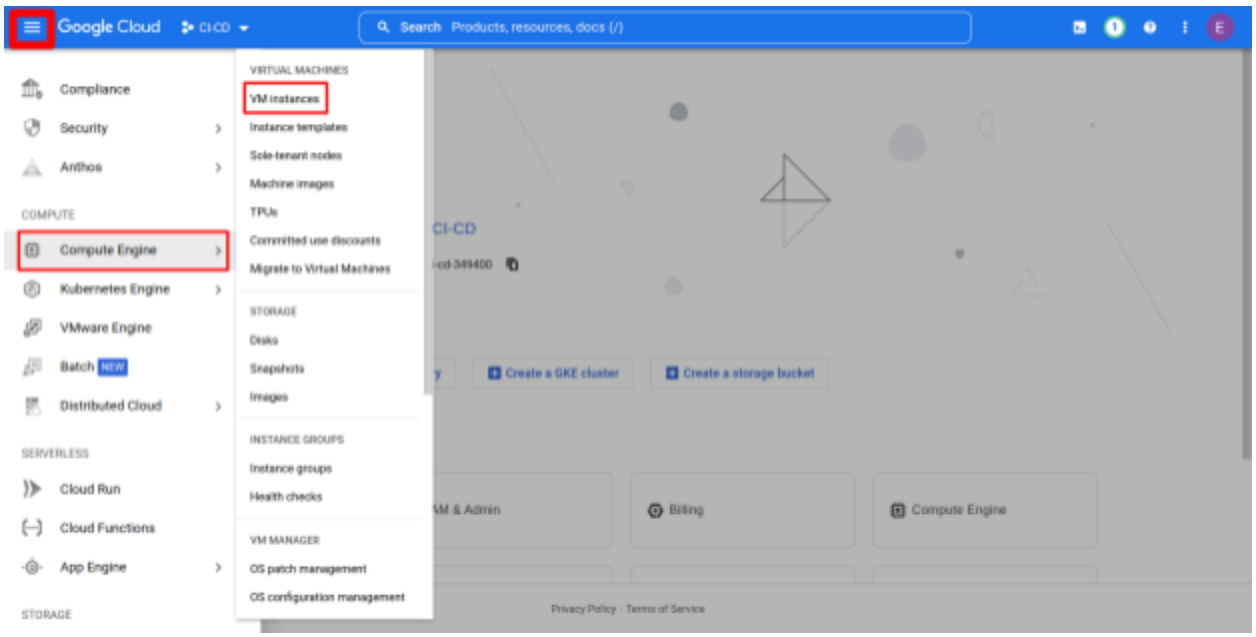
If **Compute Engine** is not in the **PINNED** section, click **MORE PRODUCTS** and scroll down to find **Compute Engine:**



Hover between **Compute Engine** and the **>** to reveal a **pin icon**. Be sure to click the **pin icon** to pin **Compute Engine** to the **PINNED** section for convenience.

VM Instances

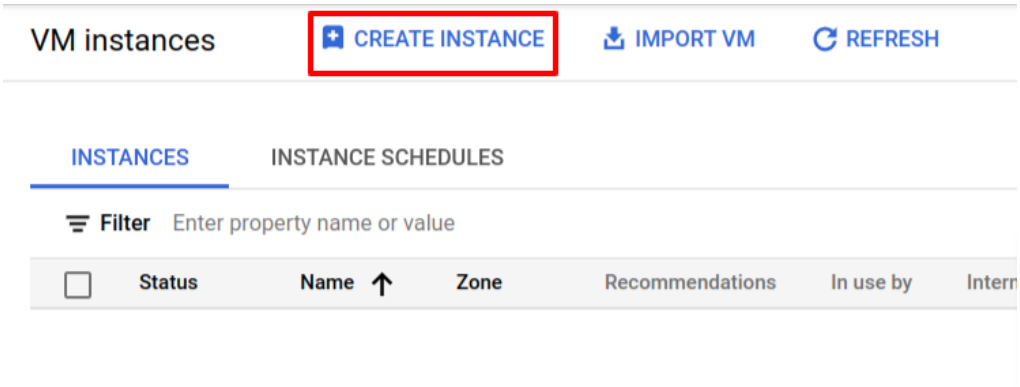
Navigate to **Menu > Compute Engine > VM instances:**



This will bring you to the VM instances page where you can create your VMs and see any VMs that have already been created.

Create a VM

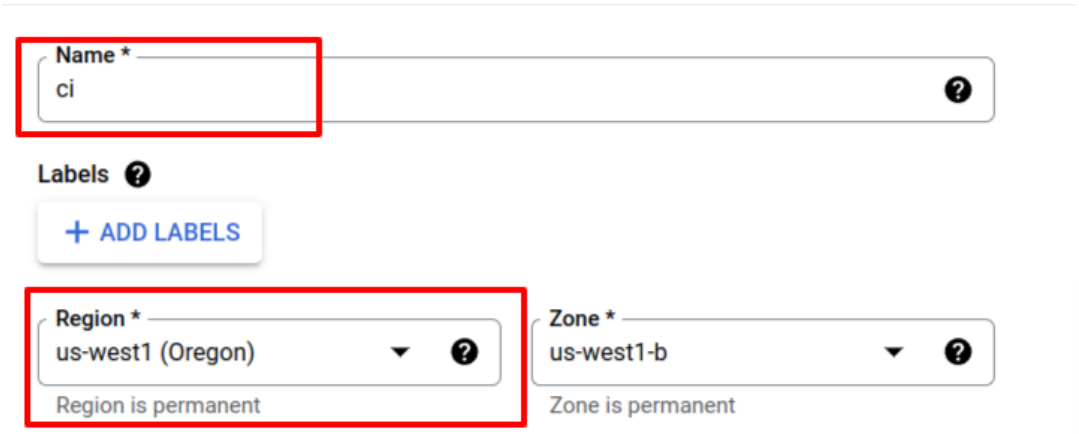
Click **CREATE INSTANCE** at the top of the page:



This will bring up the UI for creating a new VM instance.

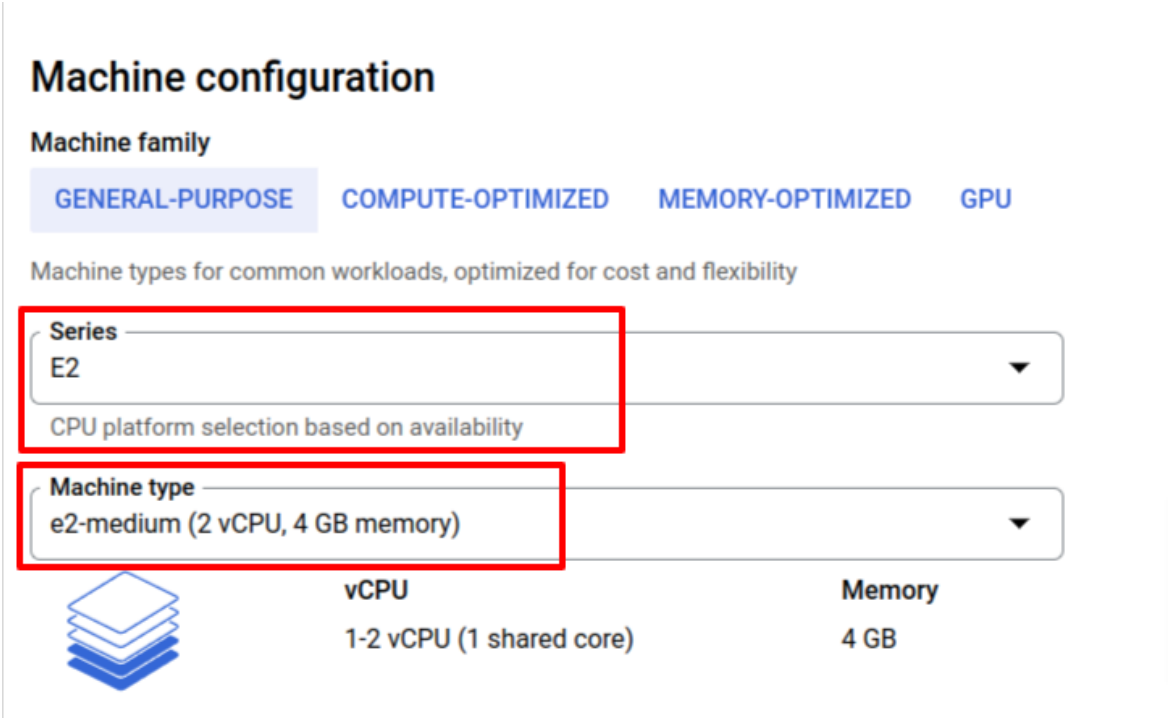
You can name the instance whatever is meaningful to you. In this example, the VM instance is named **ci.**

Choose the **Region** that is closest to you. In this example, we have chosen **us-west1 (Oregon)**.

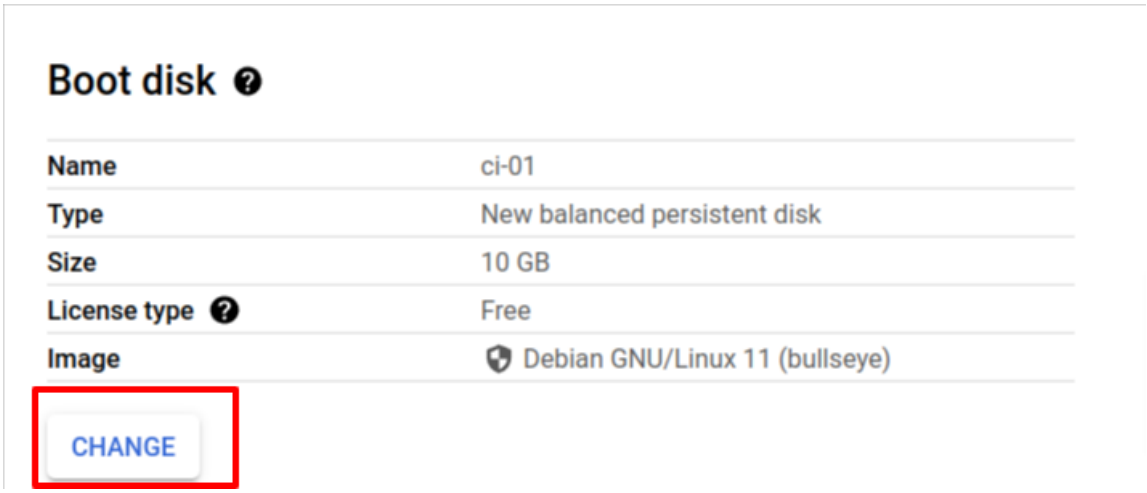


Scroll to **Machine Configuration**.

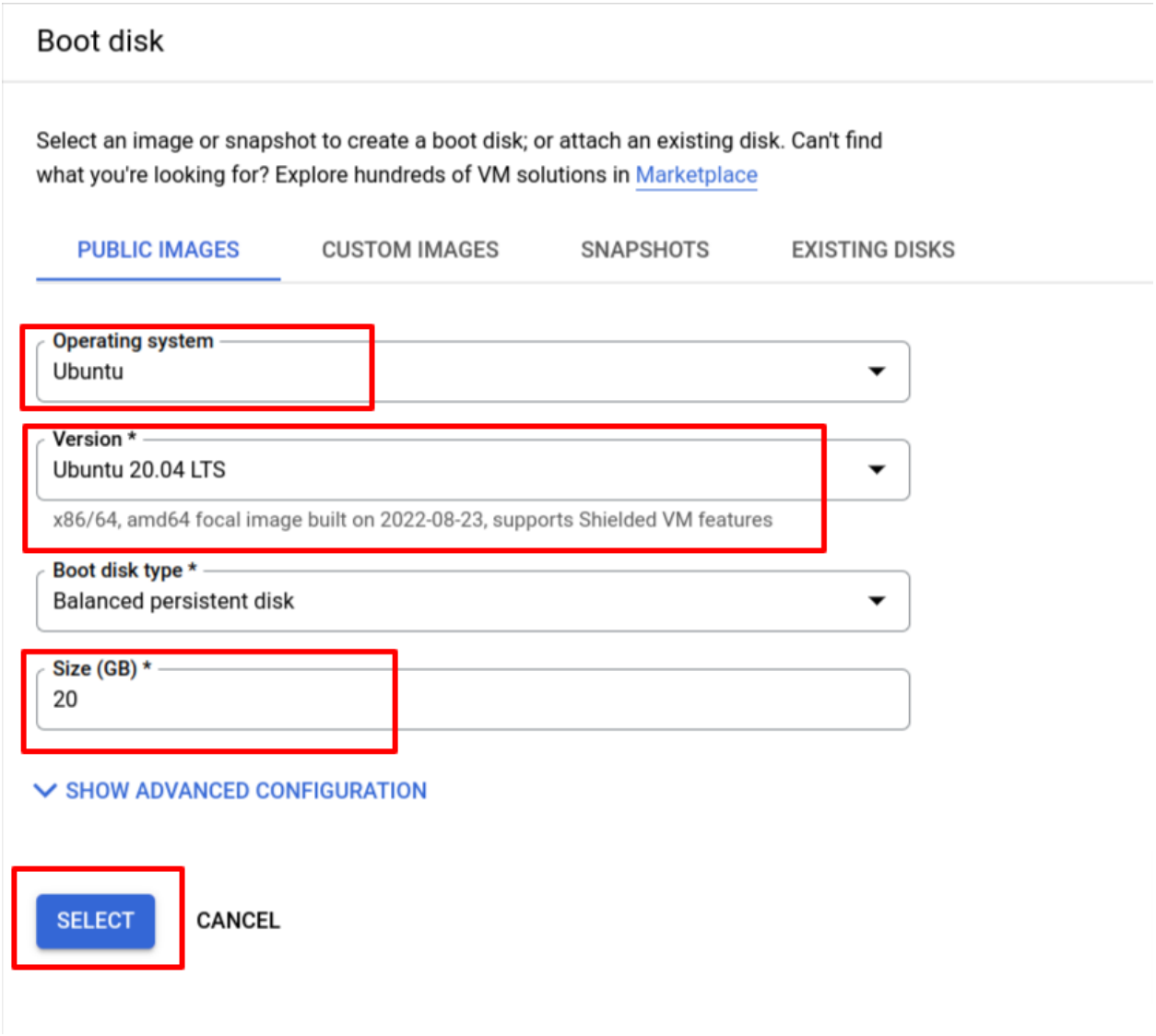
Choose **E2** in the **Series** drop down menu. For machine type, choose **e2-medium (2 vCPU, 4 GB memory)**.



Scroll to **Boot disk** and click **CHANGE**:

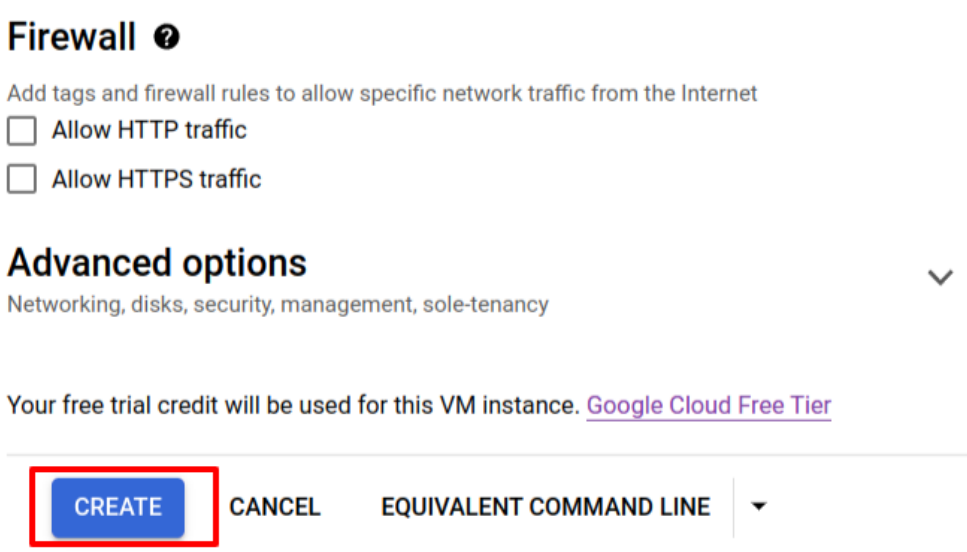


This will bring up a form that allows you to choose the operating system that we want to run. Choose **Operating system > Ubuntu.** Choose the **x86** version of **Ubuntu 20.04**. Enter **20** for the **Size**:



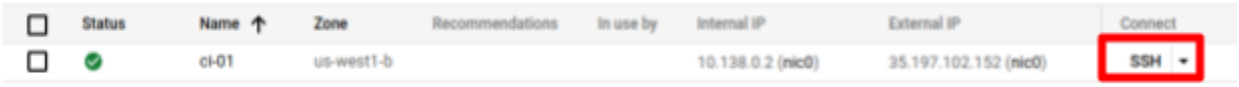
Finally, click **Select**.

Scroll to the bottom of the VM creation form and click **CREATE**.

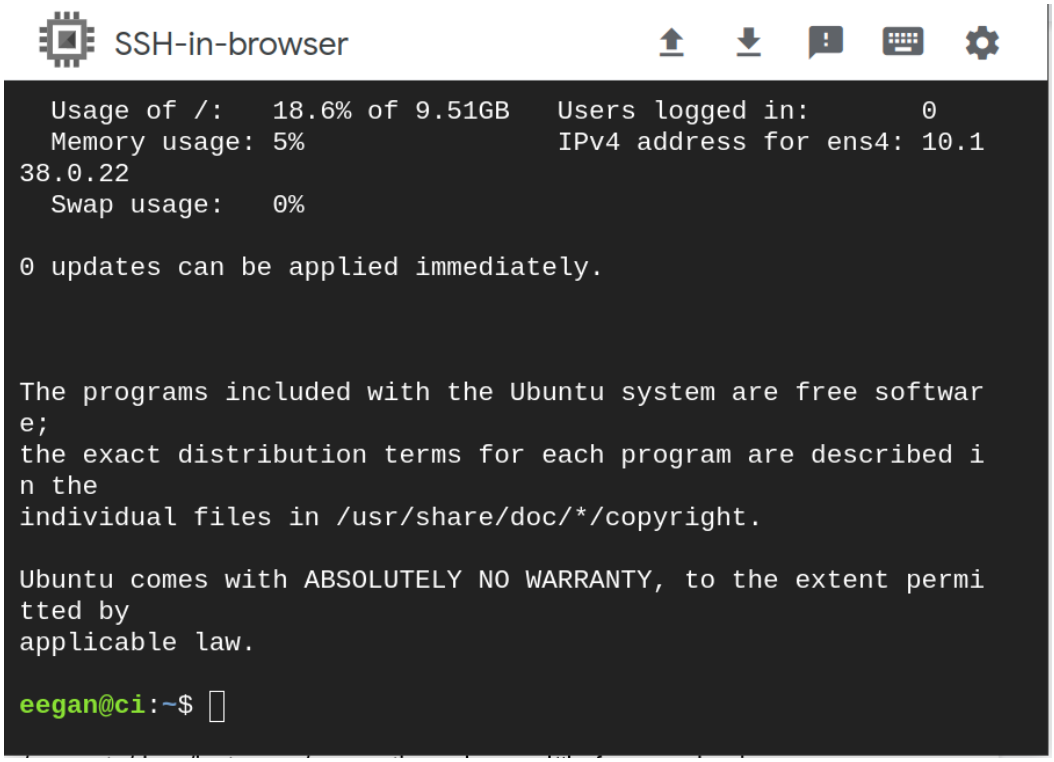


Connecting to Your Instance

You will be taken back to the **VM instances** page and will see your newly created VM. To connect to your VM, click **SSH**.



A terminal window will pop up.



You are inside a Linux Ubuntu machine. This is a terminal that you can use to interact with your VM on GCP.

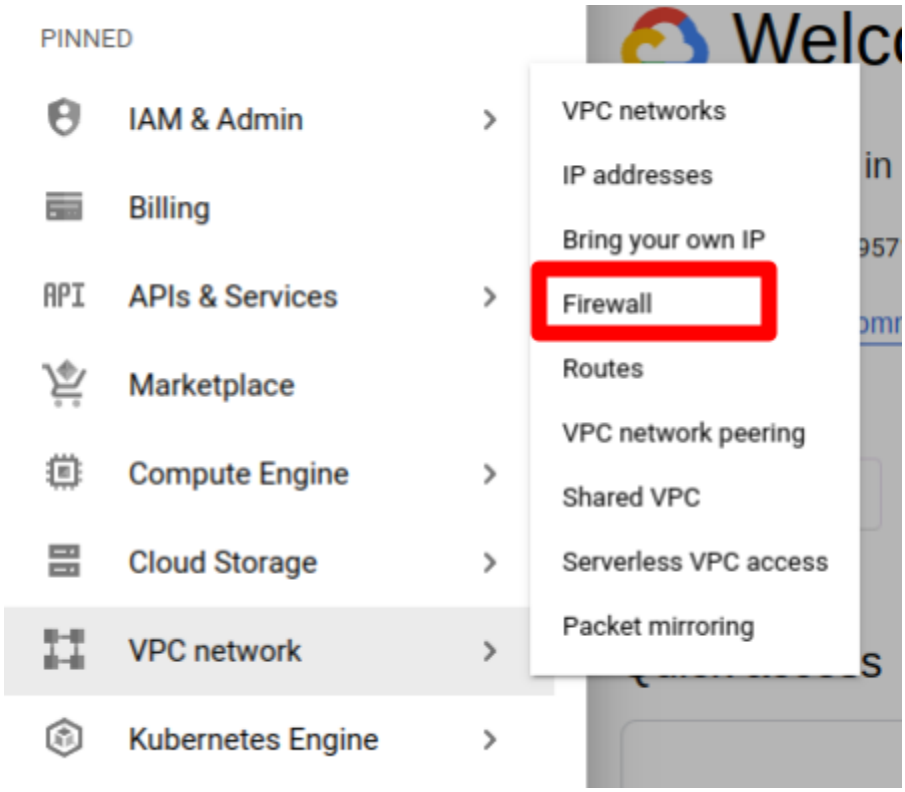
If you would like to connect to a Google Cloud VM instance from a terminal on your own computer, you will need to follow the directions here - https://cloud.google.com/compute/docs/instances/connecting-advanced#before-you-begin

Opening the Firewall for Our Google Cloud VM

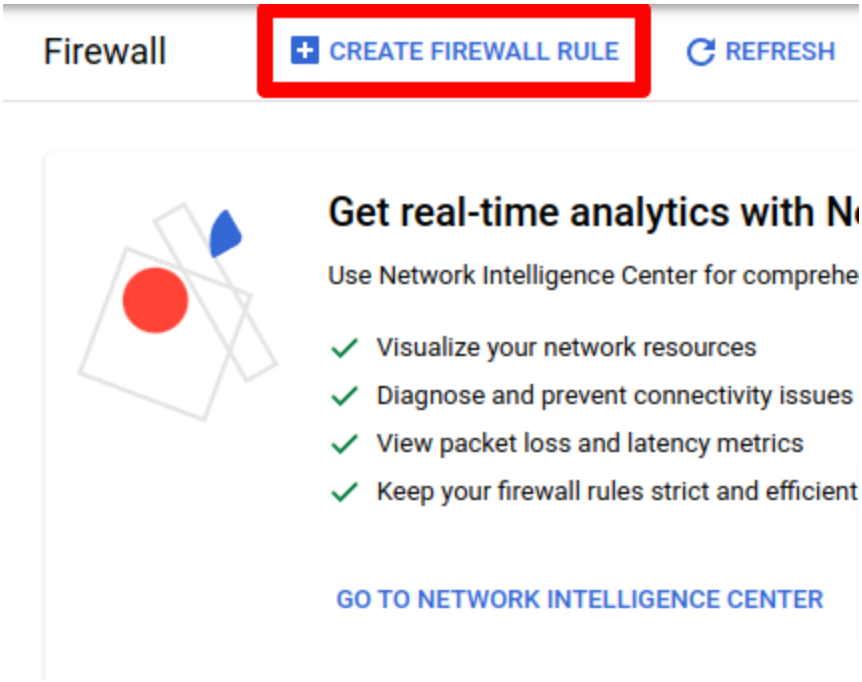
When learning new concepts, it is helpful to configure our learning environments in a way that allows us to focus on the topic at hand. In our case, we don't want to worry about whether issues we are running into are network related or due to the tools we are learning. This is not a networking course. Because of this, we will open our firewall completely to rule out any firewall issues. This is only for learning purposes. **This is *NOT* something you would do in production as it is *NOT* secure.** It will help you with troubleshooting a CI/CD pipeline, however.

Creating a Firewall Rule

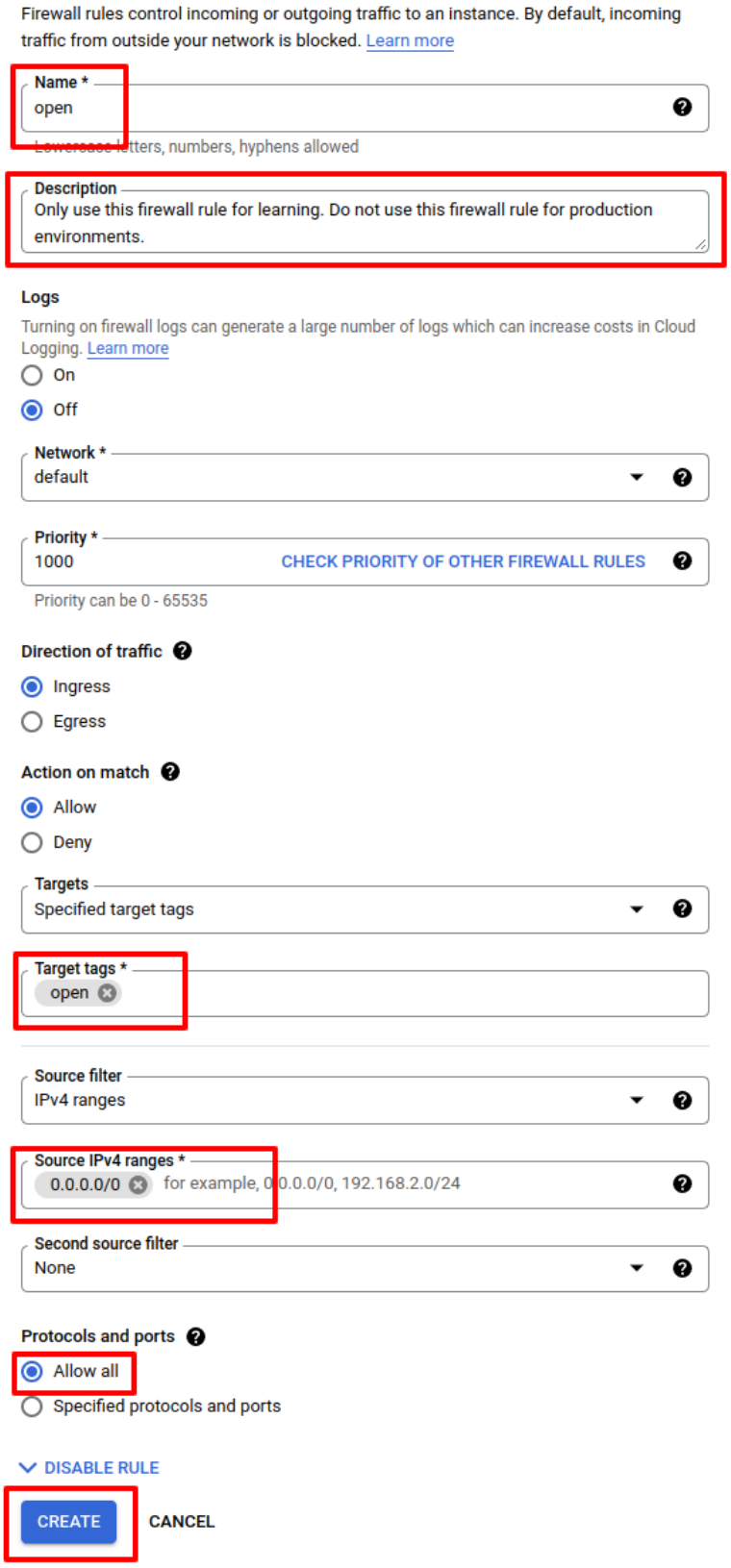
Under **VPC network** go to **Firewall**.



In networking it is common to create **rules** that are then **applied** to specific computers or VMs. We will create a **Firewall rule** and then apply that rule to our VM instance.



Refer to the below screenshot:



Name the firewall rule **open** and give it a description. In our case, we have reminded ourselves NOT to use this rule in production.

Then, under **Target tag,** enter **open**. This is how you will add this rule to your VM instance. You have to remember this or look it up in order to add it to your VM. To make it easy, give this firewall rule the same tag as the name of the firewall rule.

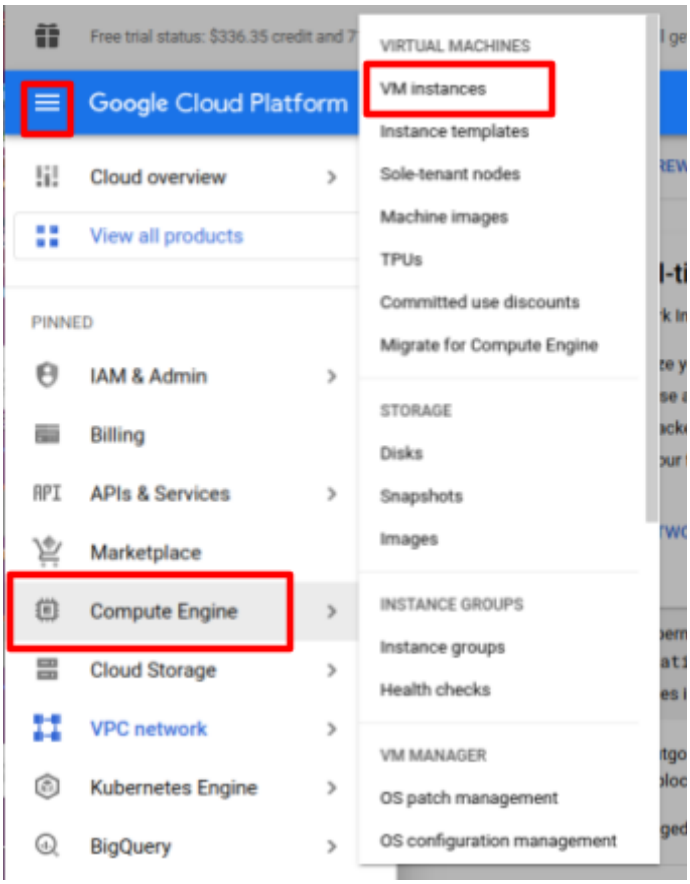
Under **Source IPv4 Ranges** we will enter `0.0.0.0/0`. `0.0.0.0/0` is networking for *all IP addresses*. This means that your VM can be connected to from *any computer.* As noted, this is highly insecure, but it is good for learning.

Under **Protocols and ports** select **Allow all** and then click **CREATE**.

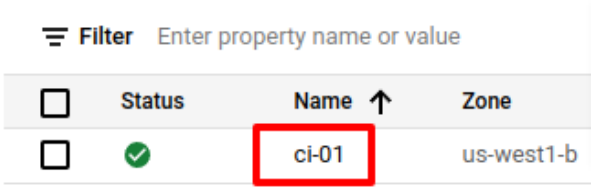
Adding the Firewall Rule to Our VM

Now we have a firewall rule that will allow *all* traffic through. A rule is of no use unless it is applied to something. In our case, we want to apply it to our VM.

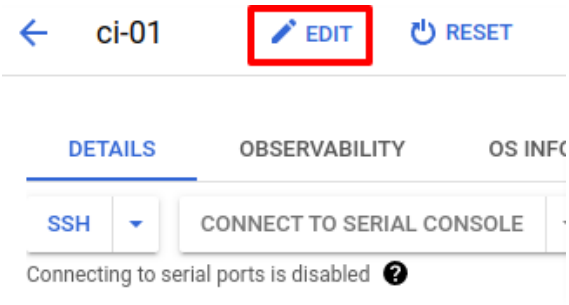
First go to **Compute Engine > VM instances**.



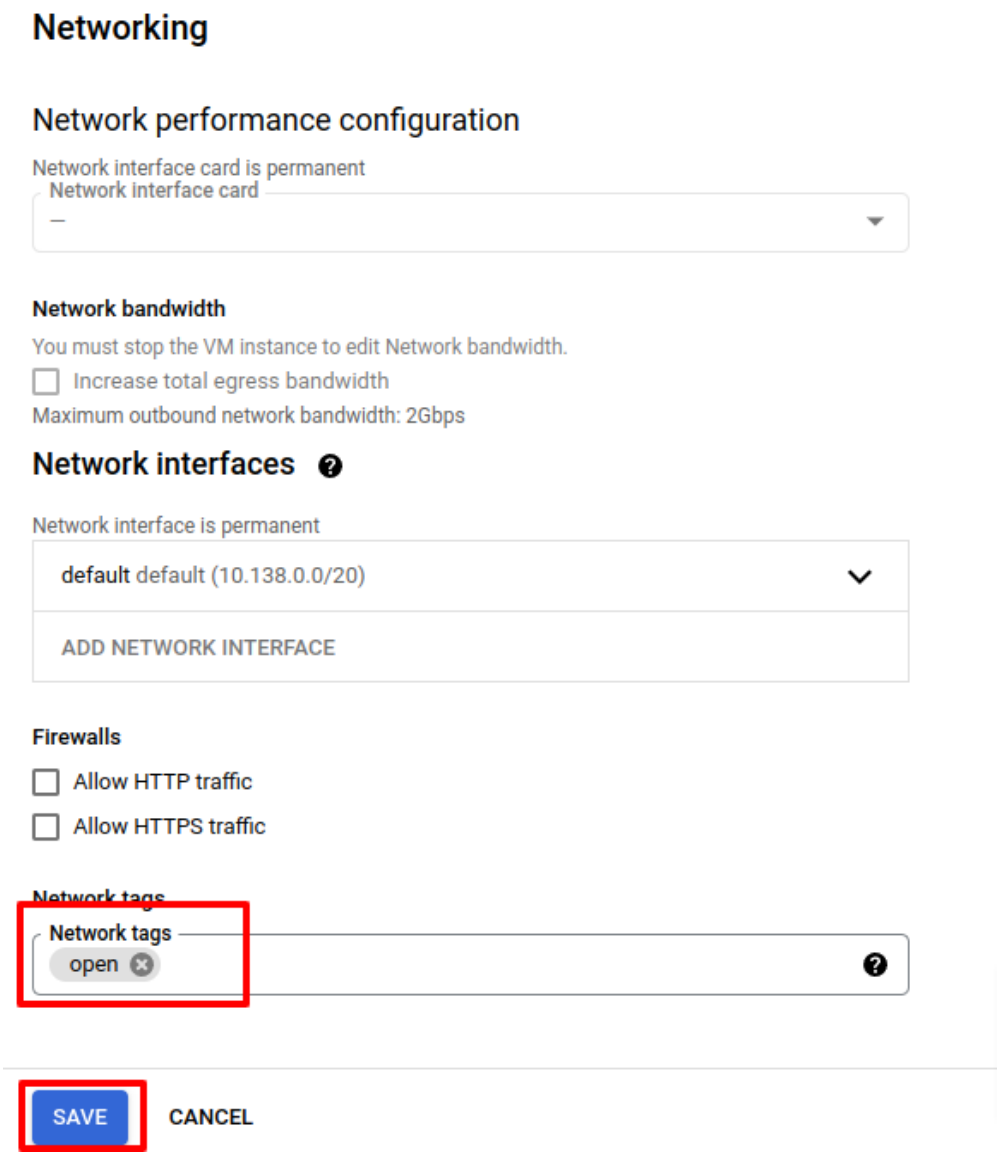
Click on your VM's name.



Click **EDIT** at the top of the screen.



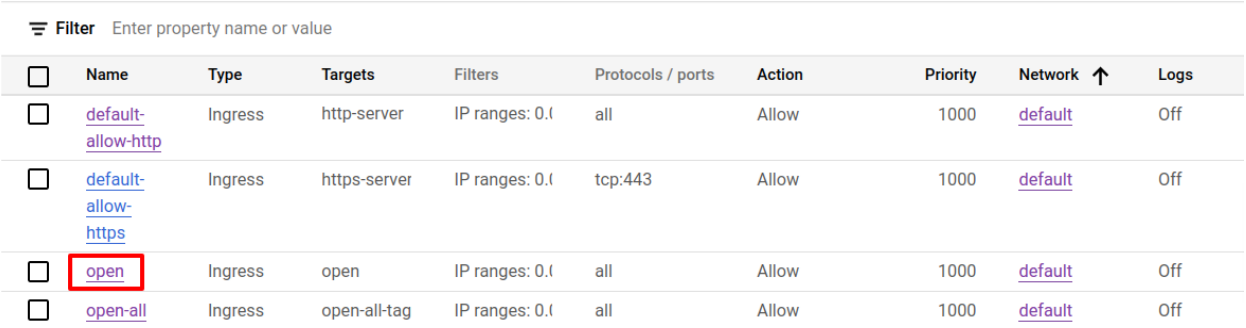
Scroll down to **Networking** if it isn't visible.



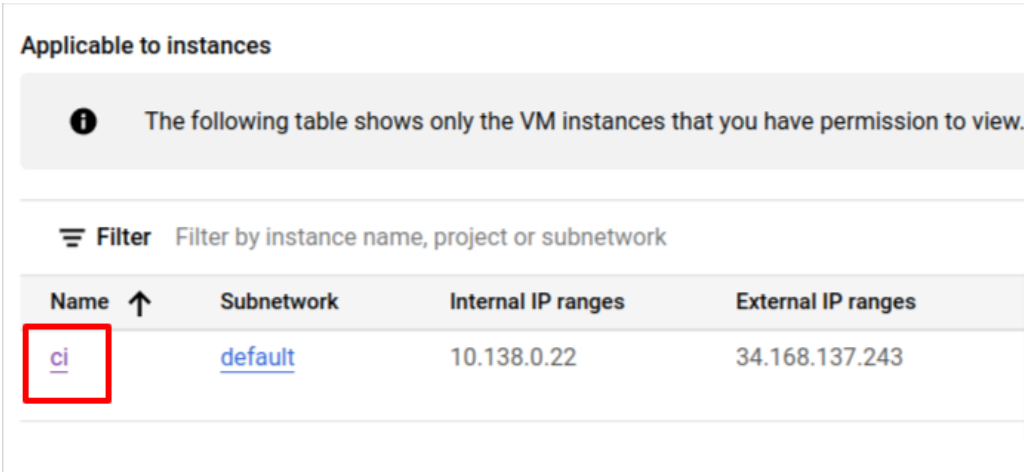
Enter **open** in the **Network tags** box. Remember, that is what we tagged our firewall rule earlier.

Verify Our Rule Was Applied

To verify that the instance is associated with the **open** firewall rule, return to **Google Cloud > VPC Network > Firewall**. Click on the **open** firewall rule.



Then scroll down to **Applicable to Instances**.



You should see your VM listed.

Installing and Verifying Docker on Your Linux VM

The following directions have been pulled directly from Docker's documentation. Run each command in the order they appear. The first command is to ensure that any old Docker versions are removed so that there is no conflict with the new version we will install. If Docker is not installed, the output will inform you that Docker was not found. Either outcome is fine.

```
$ sudo apt-get remove docker docker-engine docker.io containerd runc
```

Now that you have removed or ensured that no Docker is present on the machine, we can proceed with the install.

```
$ sudo apt-get update

$ sudo apt-get install \
```

```
    ca-certificates \
    curl \
    gnupg \
    lsb-release
$ sudo mkdir -m 0755 -p /etc/apt/keyrings
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg


$ echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list \
> /dev/null
$ sudo apt-get update
$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin
```

Type **Y** when prompted.

The directions are found here: https://docs.docker.com/engine/install/ubuntu/.

After installation, validate Docker by running the following commands:

```
$ sudo docker version

Client:             Docker Engine - Community
Version:            20.10.17
……..
Server: Docker Engine - Community
Engine:
Version:           20.10.17
……….
```

Do a smoke test with:

```
$ sudo docker run hello-world

……

Hello from Docker!

This message shows that your installation appears to be working
```

```
correctly.

…..
```

This validates that Docker has successfully been installed.

Verifying and Installing Git

If you are on Ubuntu 20.04 or most other Linux distributions, git comes pre-installed. To verify, run:

```
$ git version
```

If you are not on Ubuntu 20.04, reference the official download and installation guides below to set up Git:

- Git - Downloads
- Git - Installing Git

Summary

You are now set up to get started with the hands-on labs in the rest of the course.