Lab 1. Setting up the Learning Environment

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

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

Create a Google Cloud Platform Account

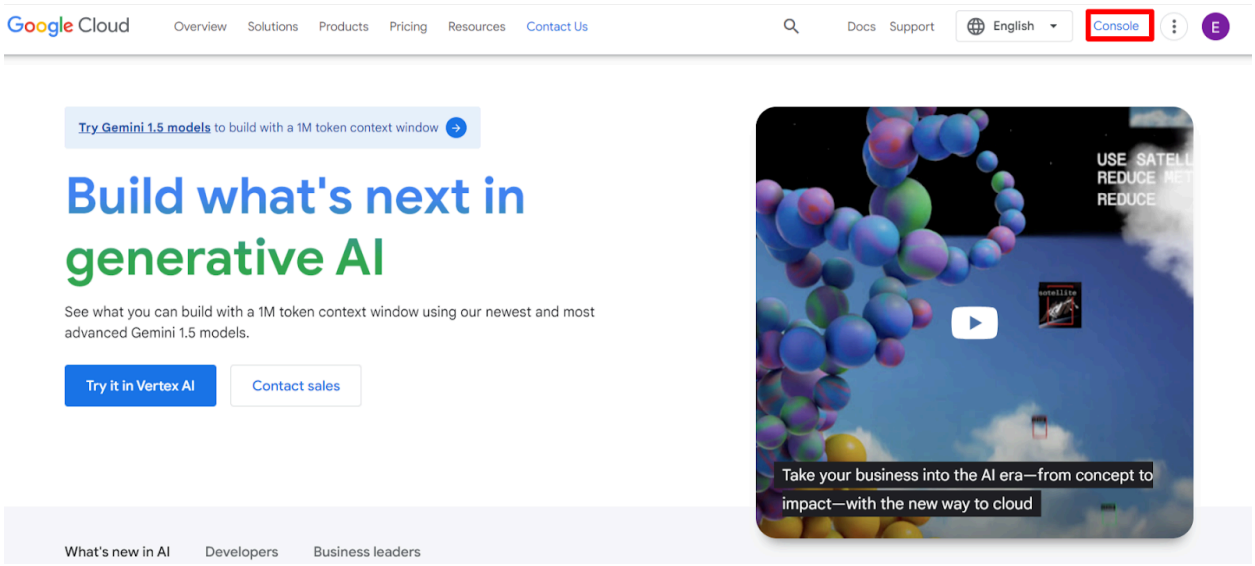
The only prerequisite for this task is a Google account. If you do not have one, go to google.com, select **Sign in** and follow the directions.

To get started, visit cloud.google.com and follow the simple steps to create a Google Cloud account. You can begin with a free account and sign up for free credits, which are valid for a limited 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 below:

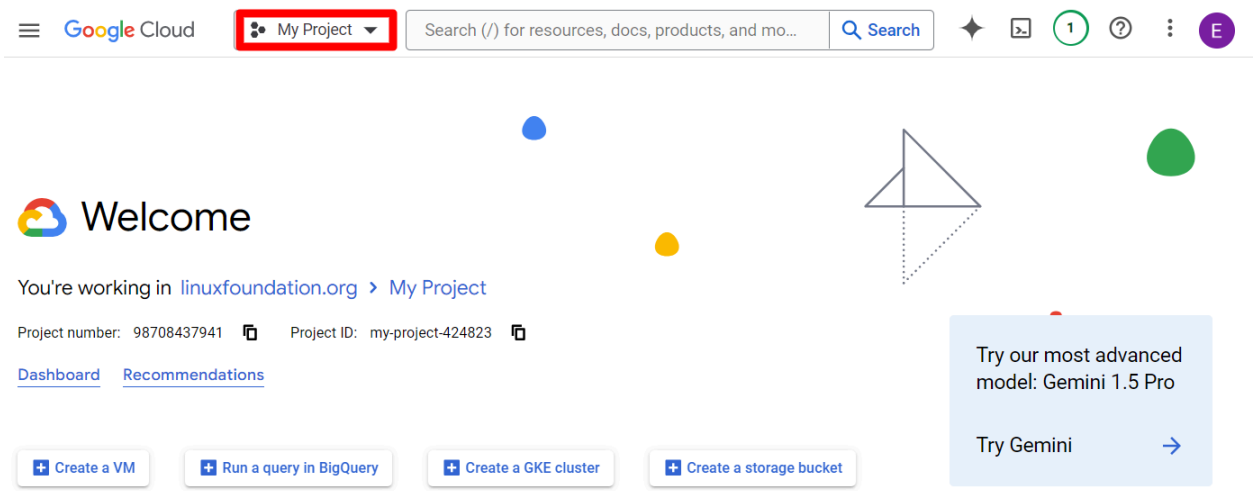


Alternatively, you can visit the console directly with this link: https://console.cloud.google.com.

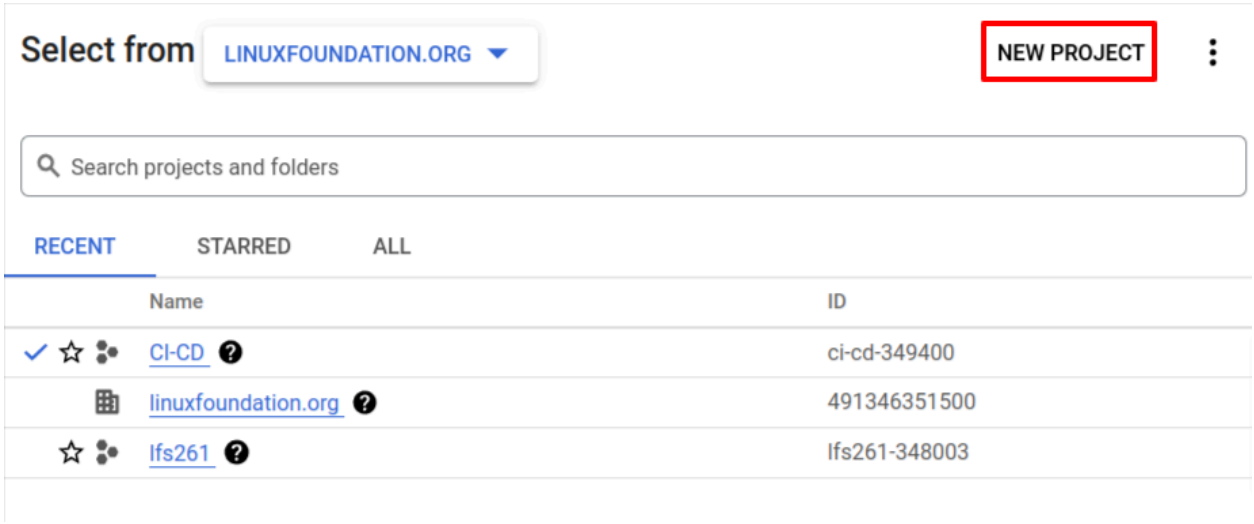
Creating a Project to Contain Your VM

According to Google Cloud documentation, a *"project in Google Cloud Platform (GCP) organizes all of your Google Cloud resources, including Cloud Storage data, Compute instances, monitoring and logging data, and App Engine instances. Projects also include associated permissions for these resources".*

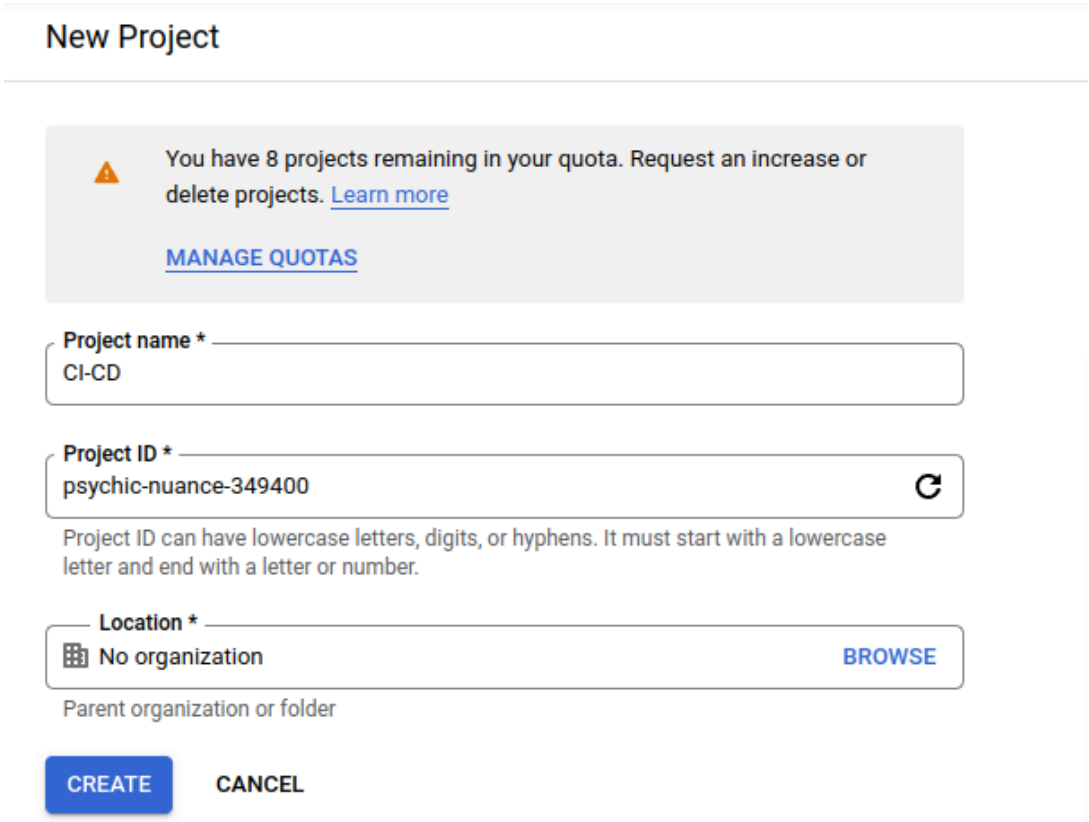
A project provides a namespace to isolate resources for that project. To set up a new project, select the project dropdown menu. If it is your first time, you will see **My First Project** in the box. Otherwise, you will see the last project that you were in. To create a new project, click inside the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



Select **NEW PROJECT**.

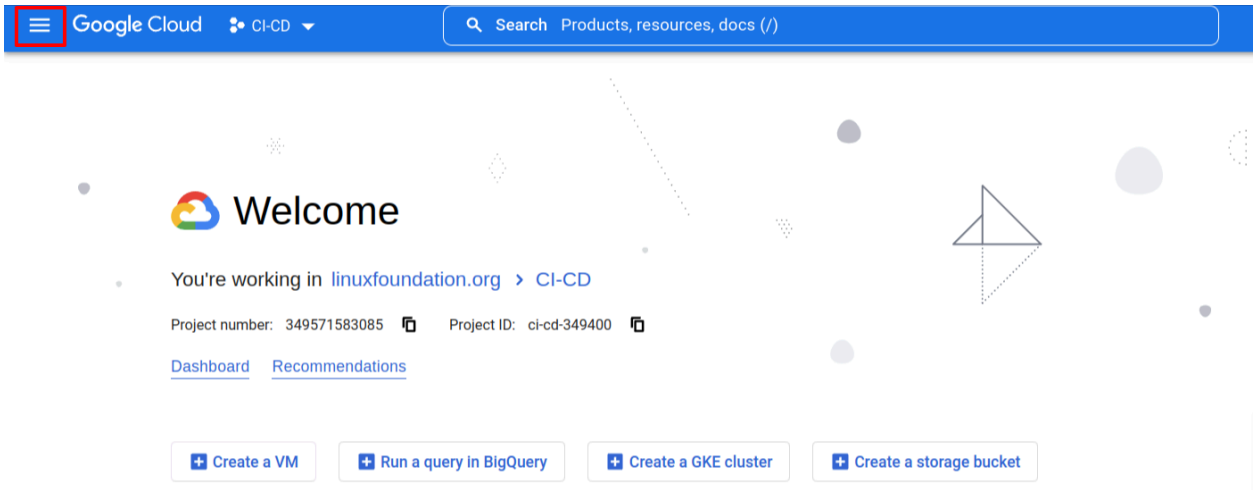


Name the project **CI-CD** and click **CREATE**.

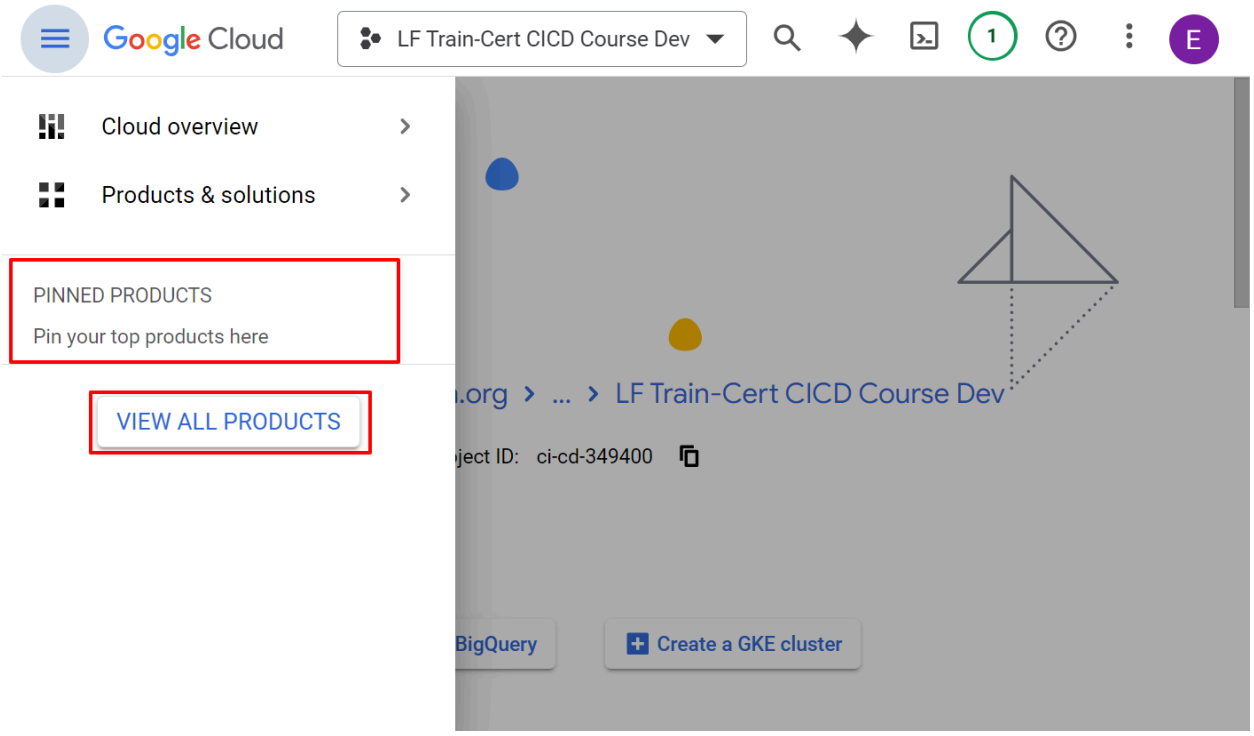


Navigating to VM Instances on Google Cloud

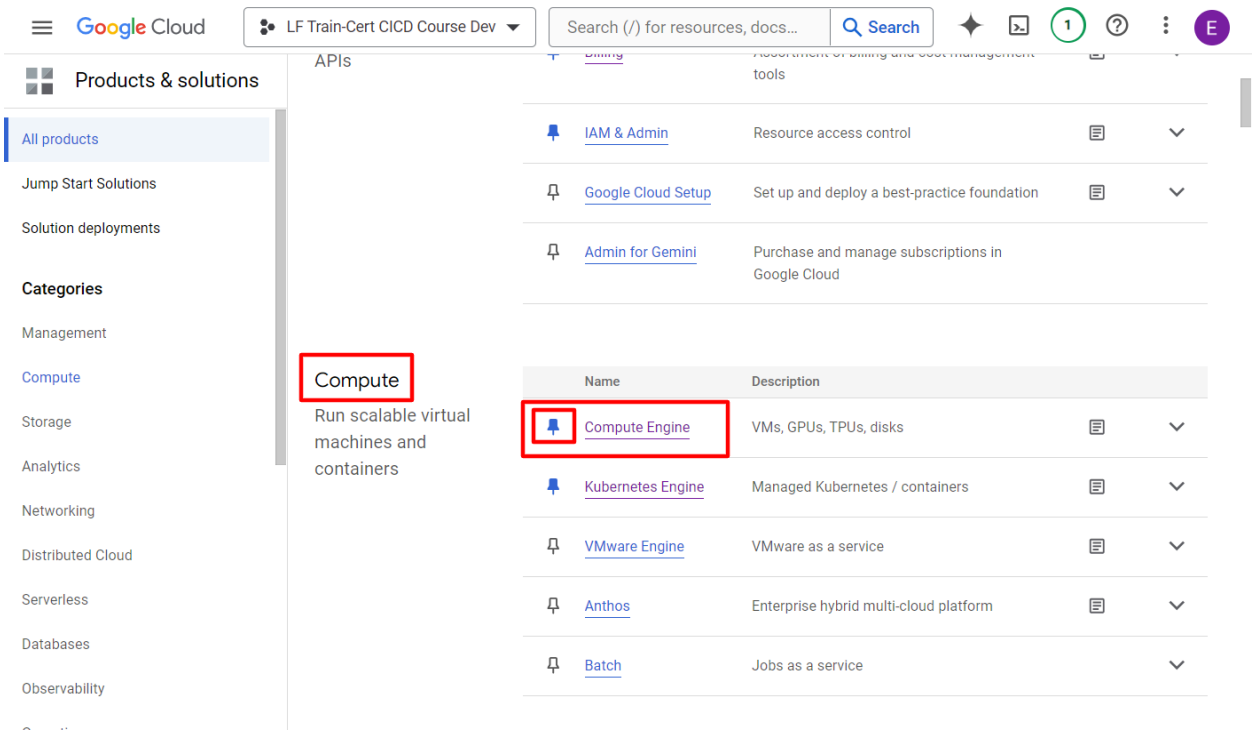
Open the menu in your Google Cloud account by clicking the menu in the top left:



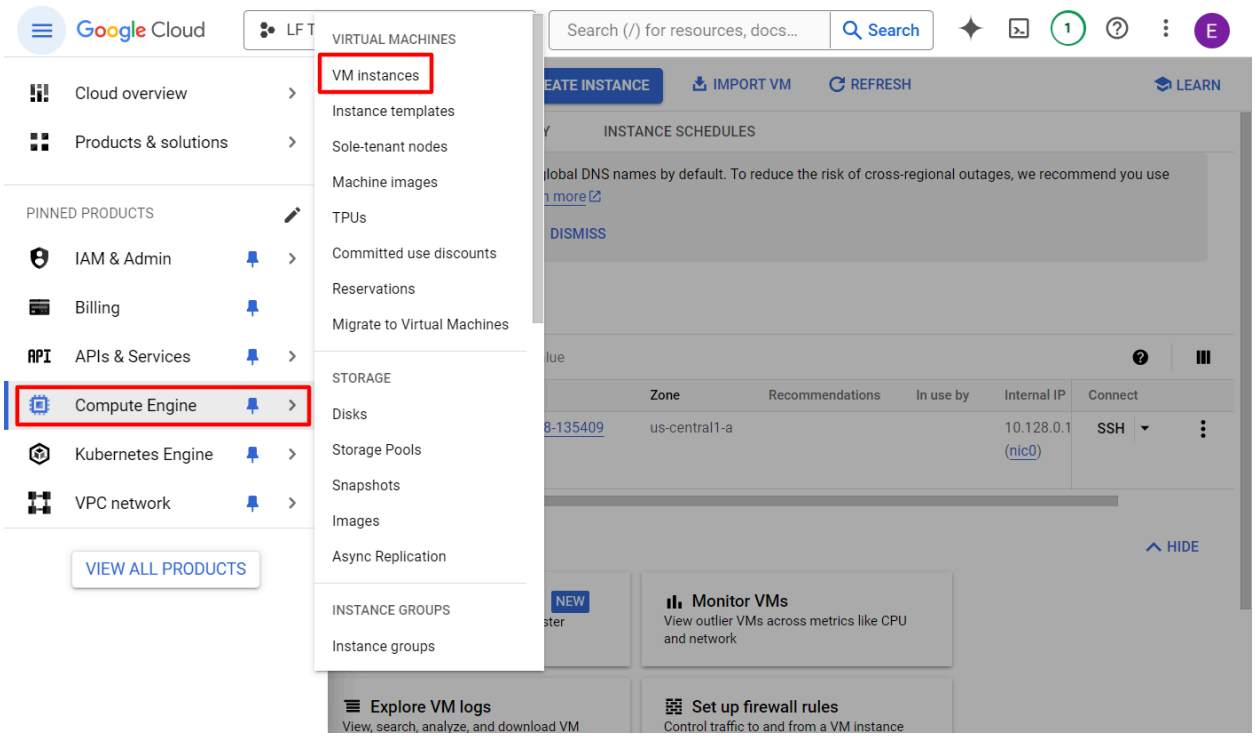
The flyout menu has two sections. Look for **Compute Engine** in the **PINNED** or **VIEW ALL PRODUCTS** sections. If **Compute Engine** is not in the **PINNED** section, click **VIEW ALL PRODUCTS**:



Find **Compute Engine** and then click the **pin icon**.



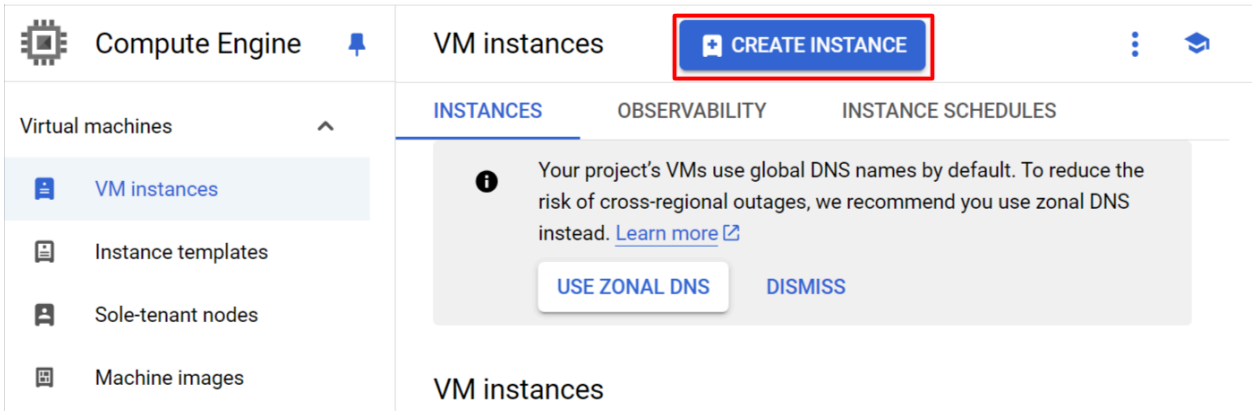
Navigate to **Menu > Compute Engine > VM instances:**



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

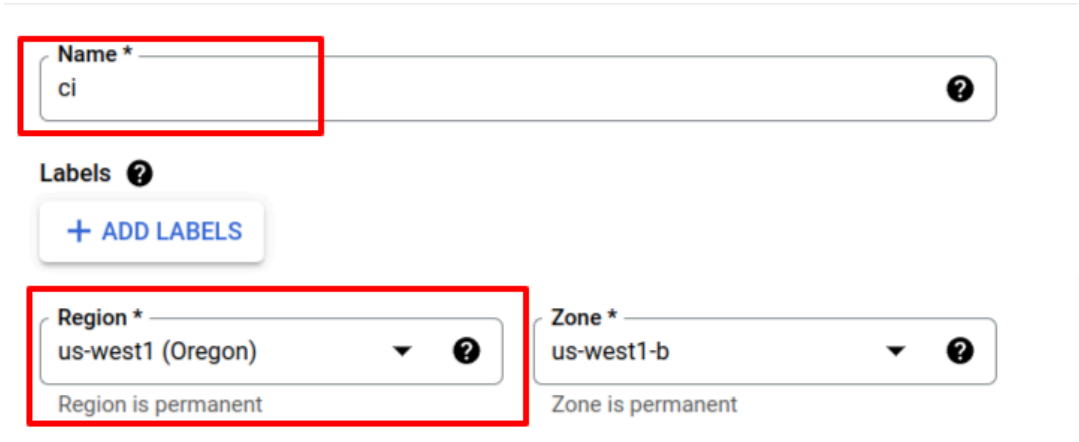
Creating a VM Instance

To bring up the UI for creating a new VM instance, select **CREATE INSTANCE** at the top of the page:

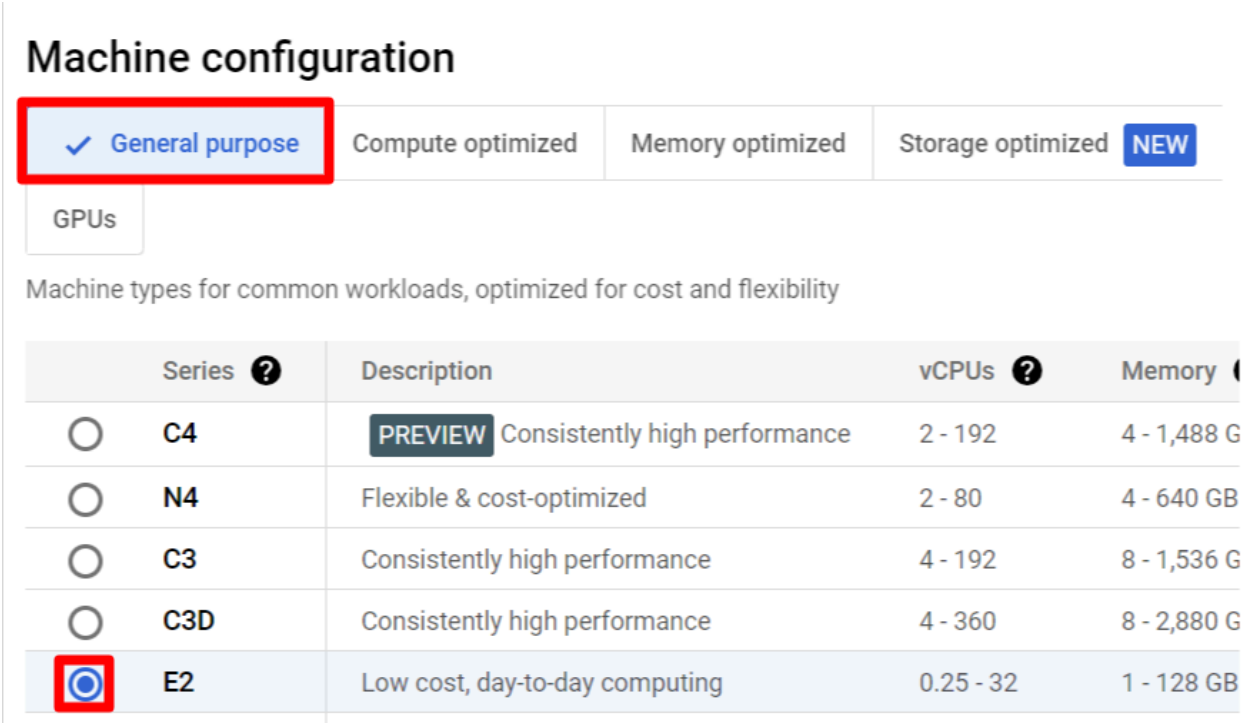


You can give the instance a name that is meaningful to you. In this example, the VM instance is named **ci.**

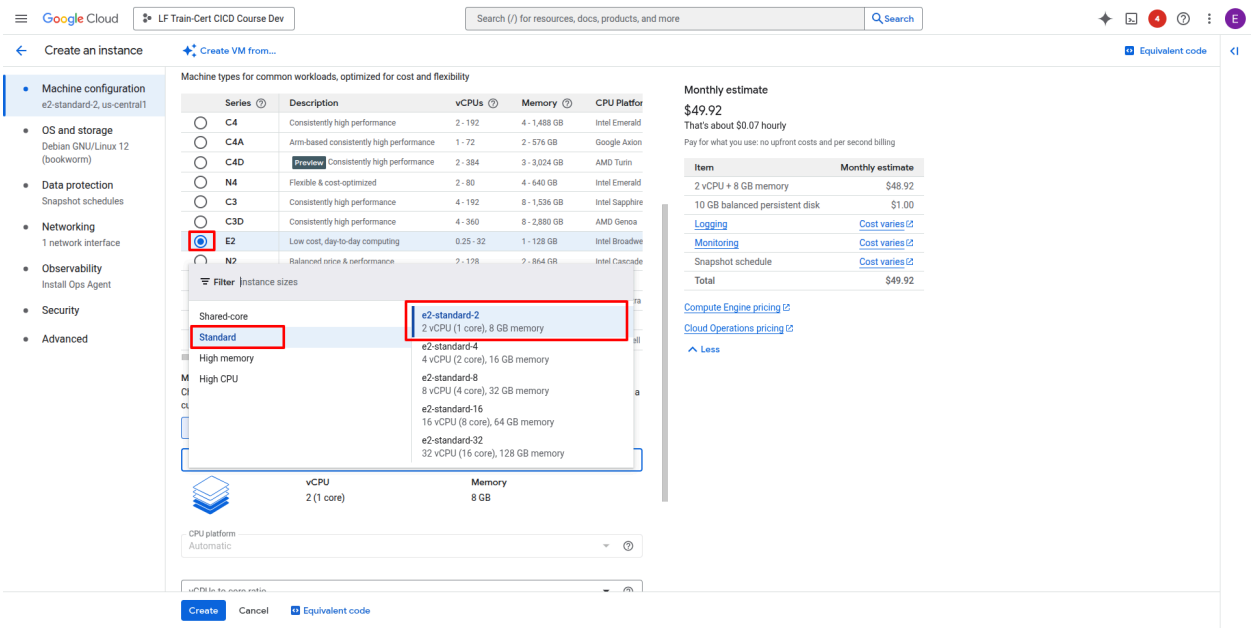
Choose the **Region** that is closest to you. In this example, we have chosen **us-west1 (Oregon)**.



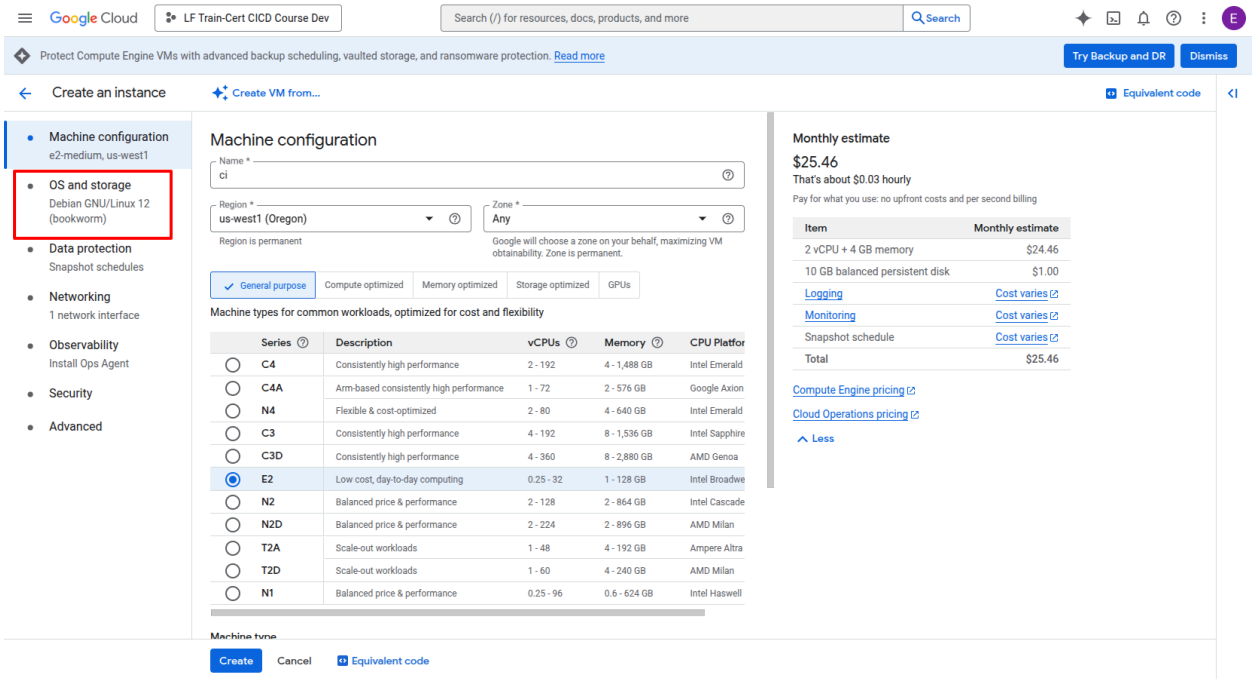
Scroll to **Machine Configuration**. Select **E2** under the **General purpose** tab.



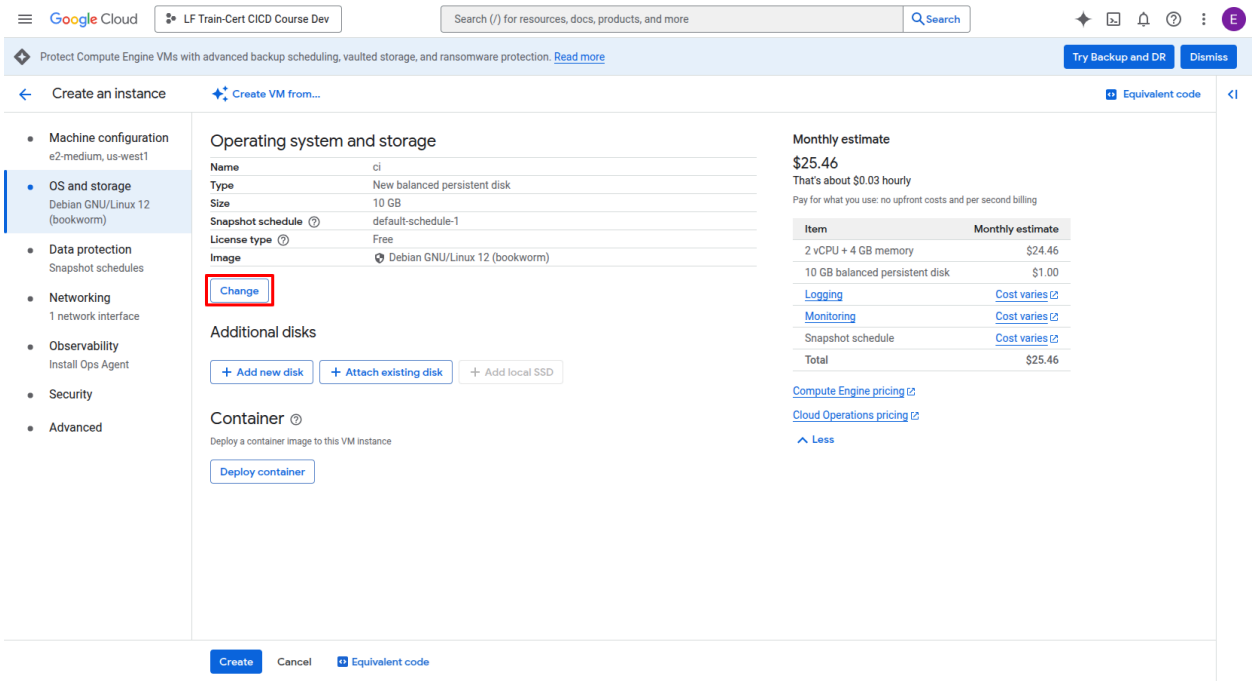
Scroll to **Machine type**. For machine type, choose **e2-standard-2 (2 vCPU, 8 GB memory)**.



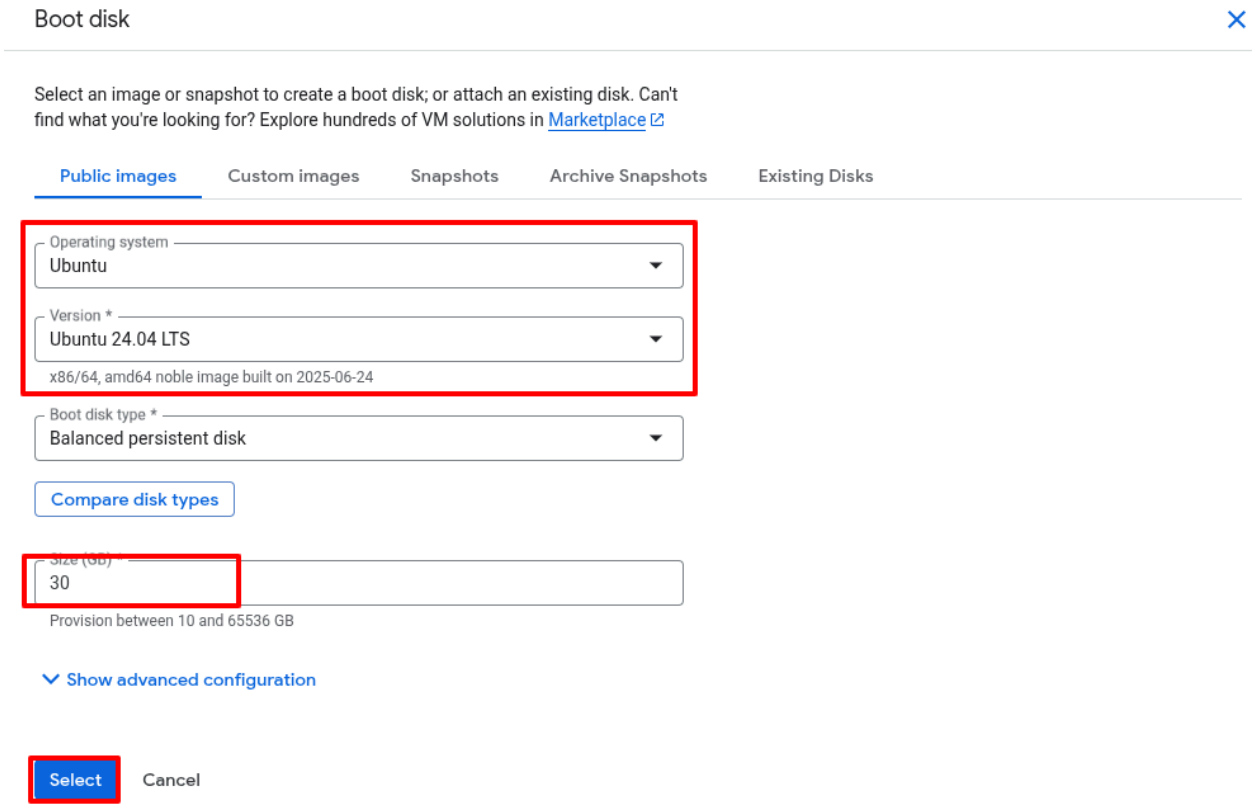
Click **OS and storage** on the right.



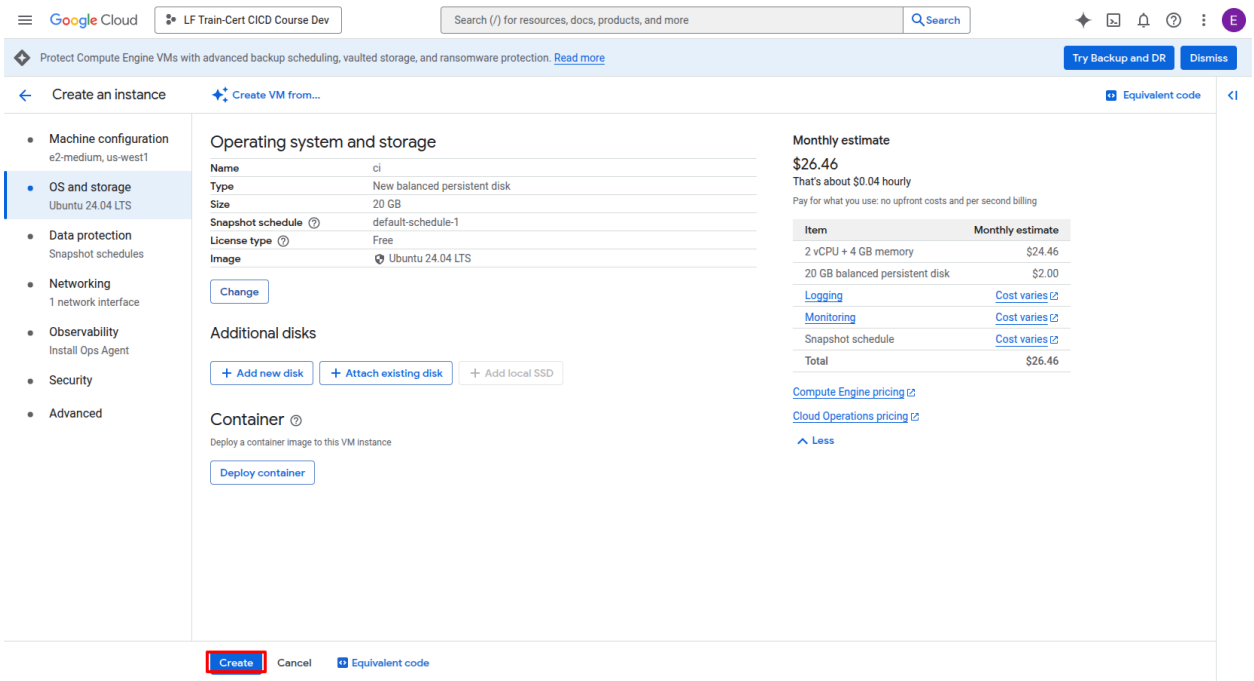
Scroll to **Boot disk** and select **CHANGE**:



This will bring up a form that allows you to choose the operating system you want to run. As we decided to use Ubuntu 24.04 for lab exercises, choose **Operating system > Ubuntu.** Choose the **x86** version of **Ubuntu 24.04**. Enter **30** for the **Size**. Then, click **Select**.

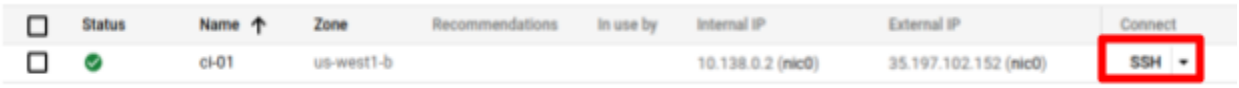


Scroll to the bottom of the VM creation form and click **CREATE**.

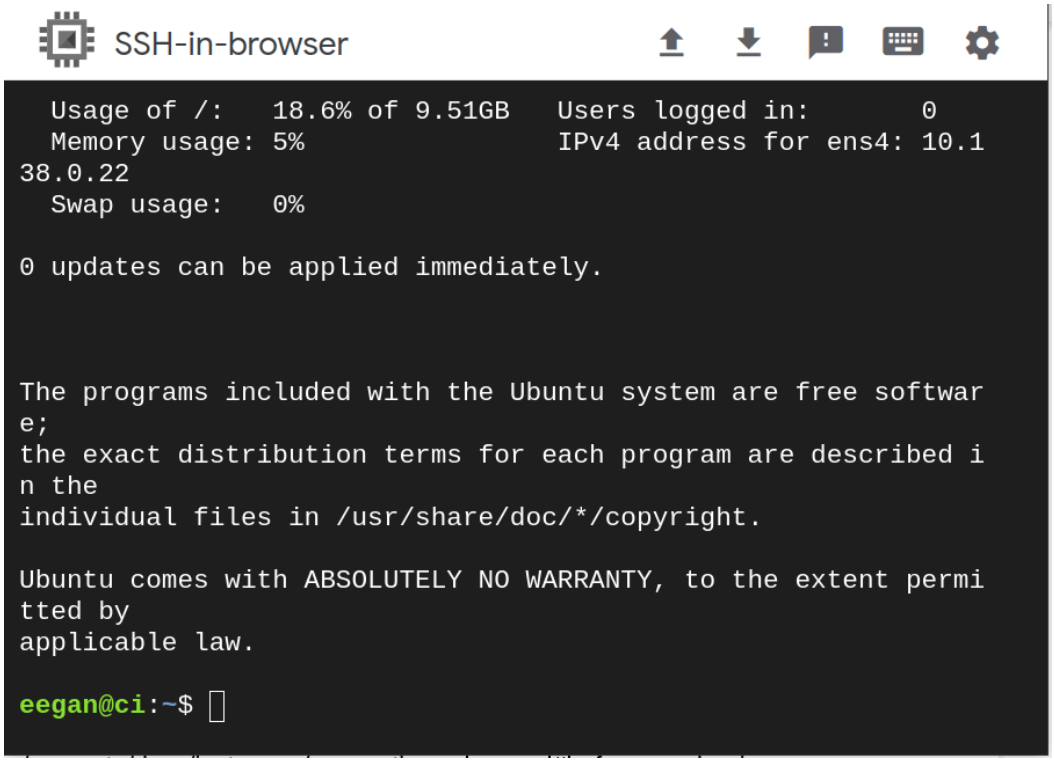


Connecting to Your VM Instance

You will be taken back to the **VM instances** page where you 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.

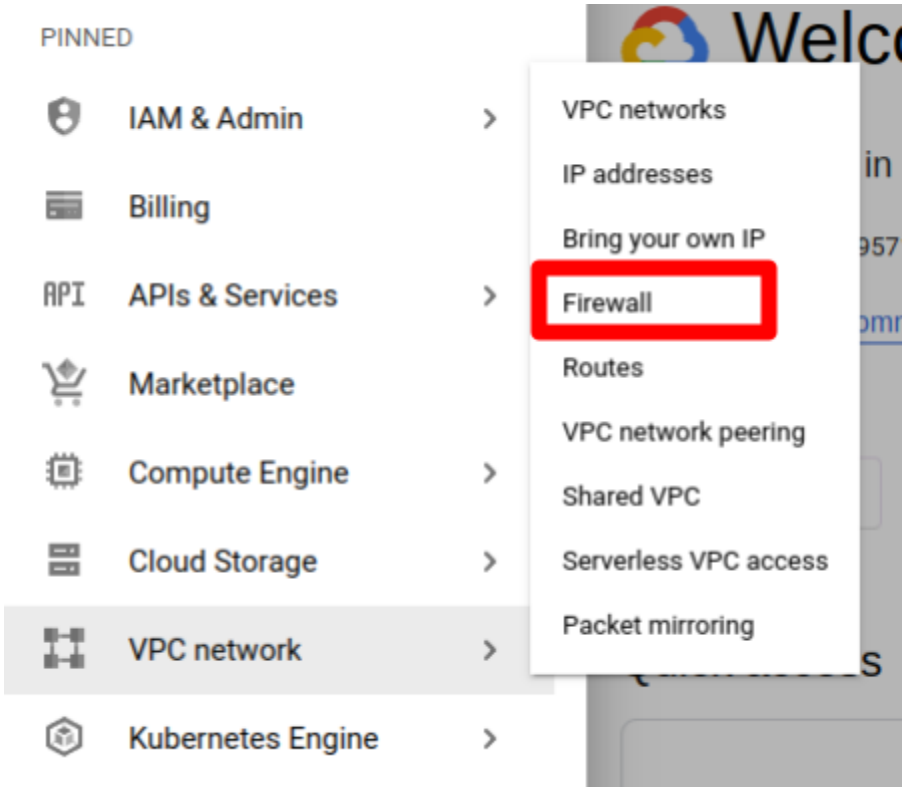
If you would like to connect to a Google Cloud VM instance from a terminal on your own computer, you will need to follow the directions here: https://cloud.google.com/compute/docs/instances/connecting-advanced#before-you-begin

Opening the Firewall for Our Google Cloud VM

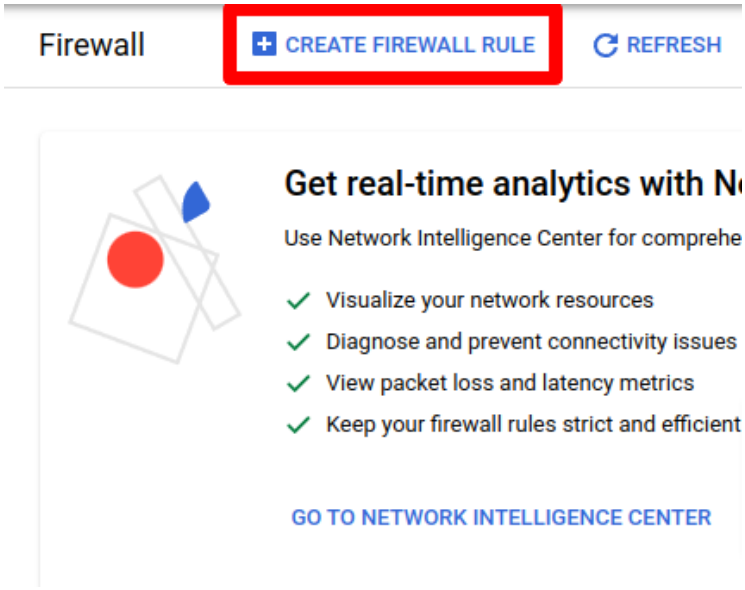
When learning new concepts, it is helpful to configure our learning environments in a way that allows us to focus on the topic at hand. In our case, we don't want to worry about whether the issues we are running into are network-related or due to the tools we are learning. This is not a networking course. As such, 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.** However, it will help you troubleshoot a CI/CD pipeline.

Creating a Firewall Rule

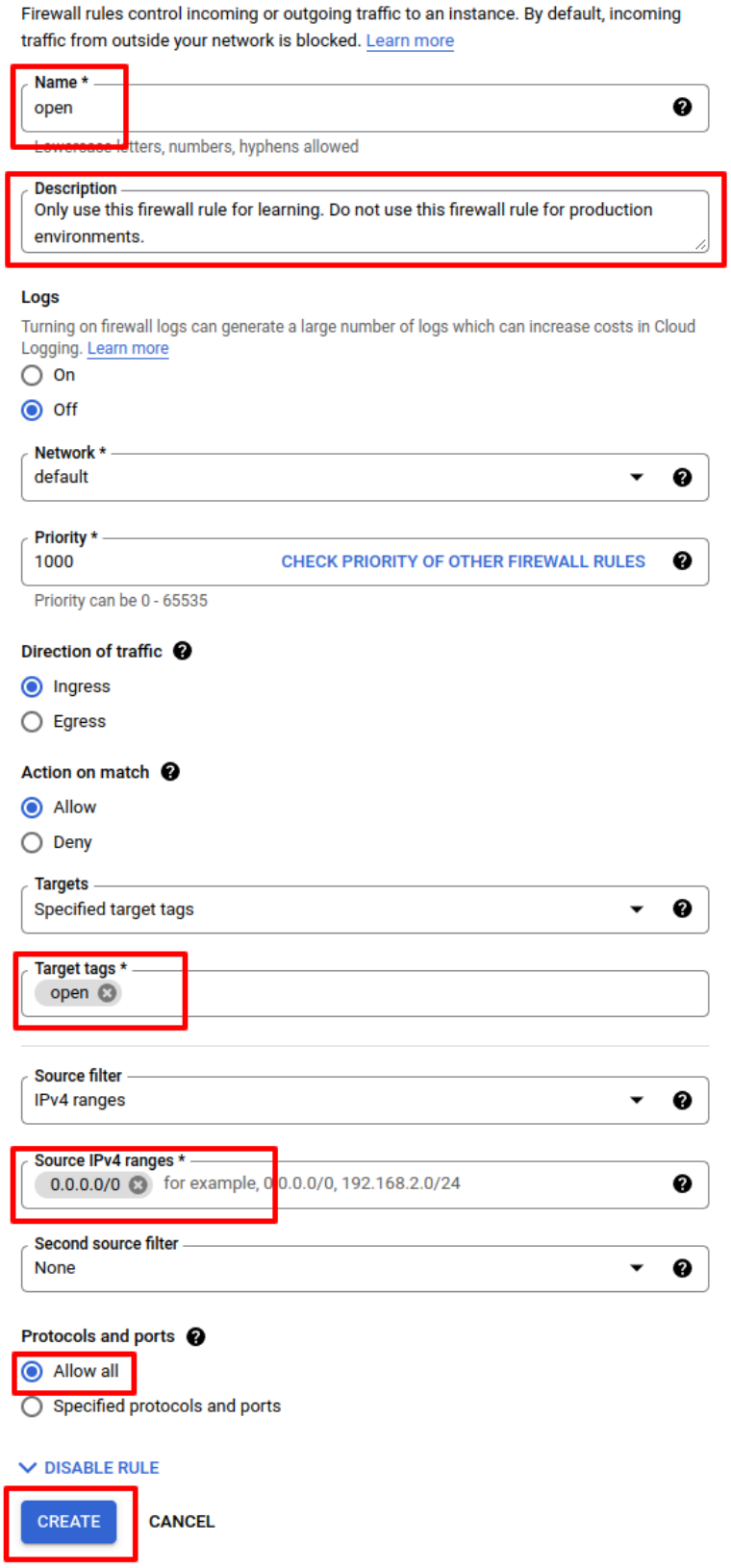
Under the **VPC network** go to **Firewall**.



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 it to our VM instance.



Refer to the following 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 need to remember this or look it up to add it to your VM. To make it easy, give this firewall rule the same tag as the name of the firewall rule.

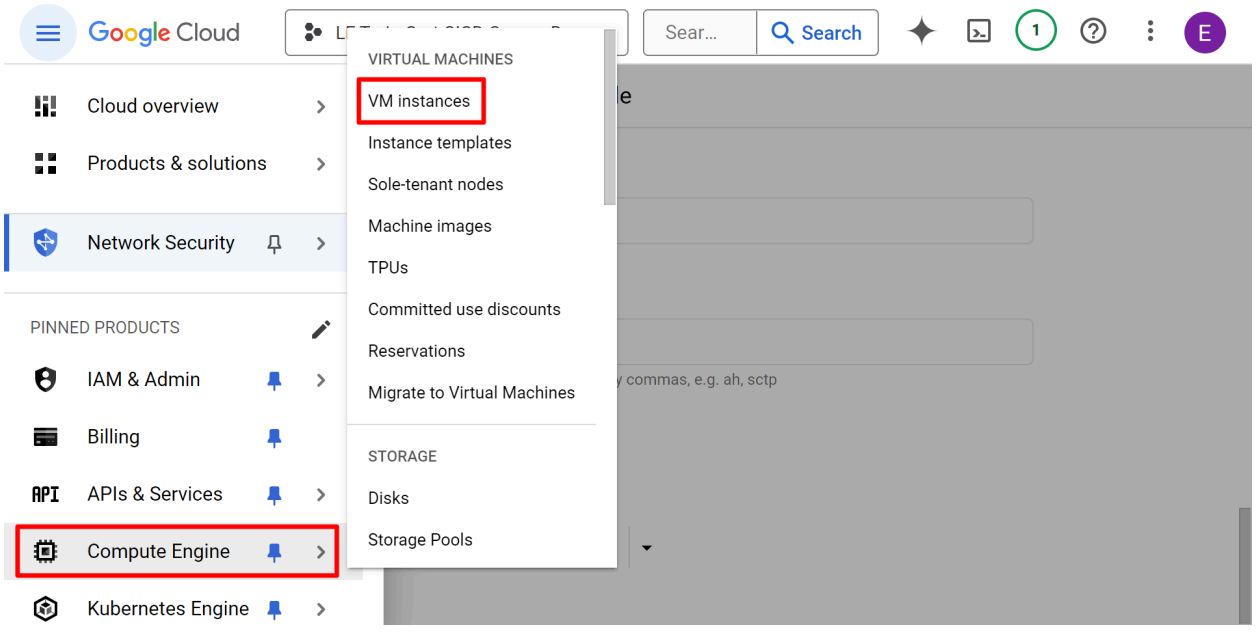
Under **Source IPv4 Ranges** we will enter `0.0.0.0/0`. `0.0.0.0/0` is networking for *all IP addresses*. This means that your VM can be connected to from *any computer.* As noted, this is highly insecure, but it is good for learning.

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

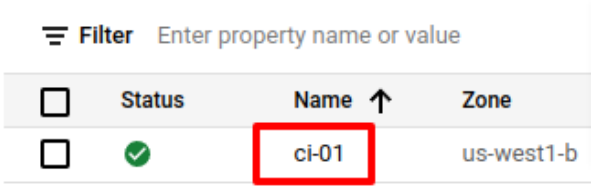
Adding the Firewall Rule to Our VM

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 this case, we want to apply it to our VM.

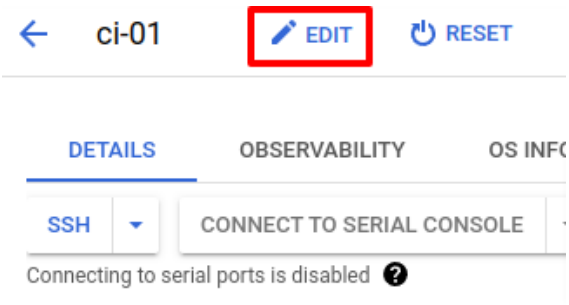
First go to **Compute Engine > VM instances**.



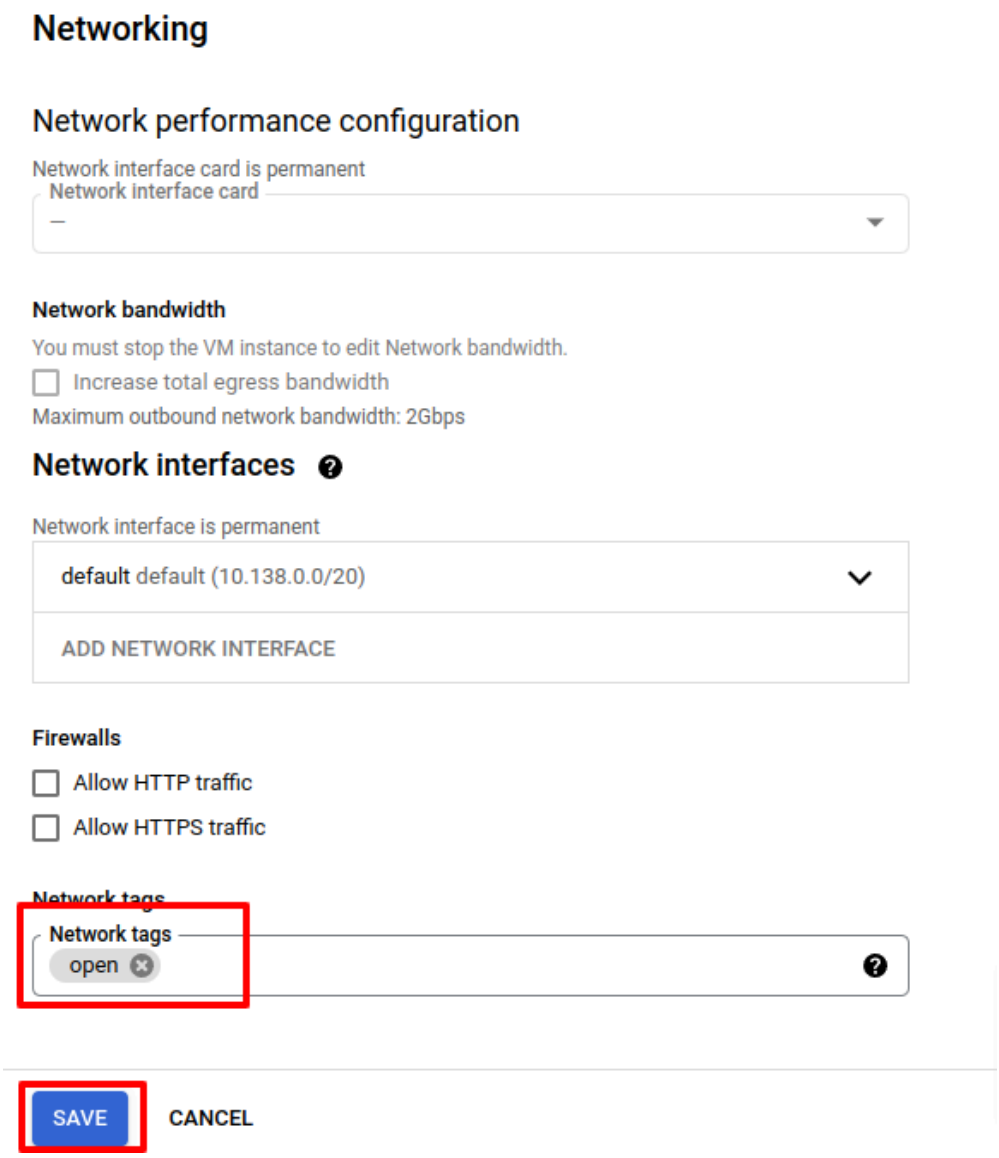
Click on your VM's name.



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



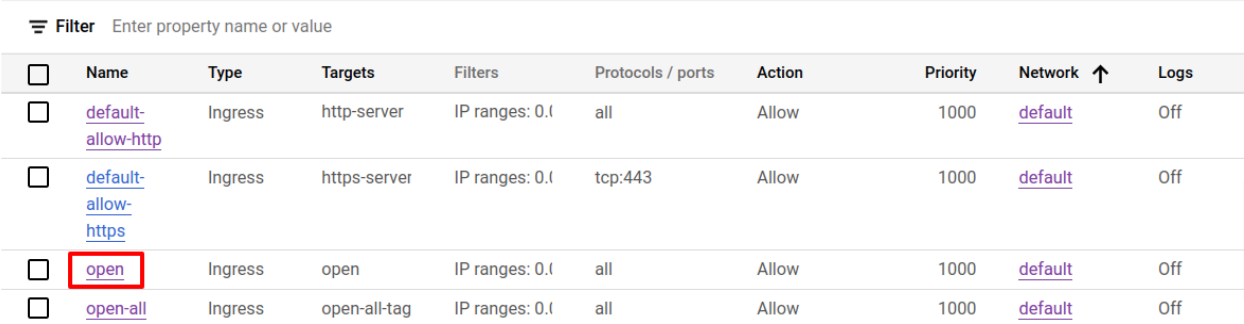
Scroll down to **Networking** if it isn't visible.



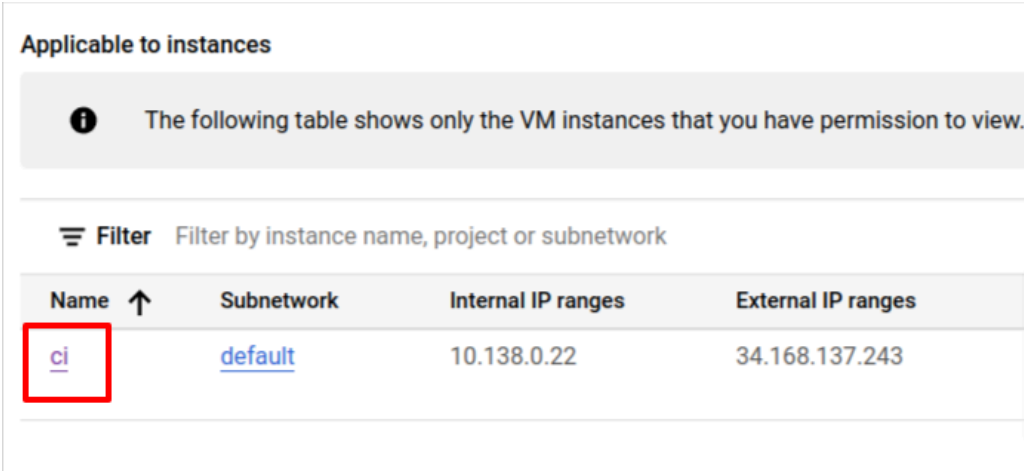
Enter **open** in the **Network tags** box. Remember, this is what we tagged our firewall rule earlier.

Verify the 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 in which they appear. The first command ensures that any existing Docker versions are removed, preventing conflicts 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
```

```
$ 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:           28.0.4
……..
Server: Docker Engine - Community
Engine:
Version:          28.0.4
……….
```

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.

Add your user to the **docker** group. This will make it more convenient to run commands, as you won't need to type **sudo** every time you run Docker.

WARNING: In a production environment, this has significant security implications that must be carefully weighed.

There is no one-size-fits-all answer as to whether you should add users to the **docker** group.

```
$ sudo usermod -aG docker $USER
```

```
$ newgrp docker
```

You should now be able to run **docker** commands without typing **sudo** first.

Verifying and Installing Git

Git comes pre-installed on most Linux distributions. To verify, run:

```
$ git version
```

If it is not installed, reference the official download and installation guides below to set up Git:

- Git - Downloads
- Installing Git

Summary

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