Preparing Your Google Cloud VM for W4705

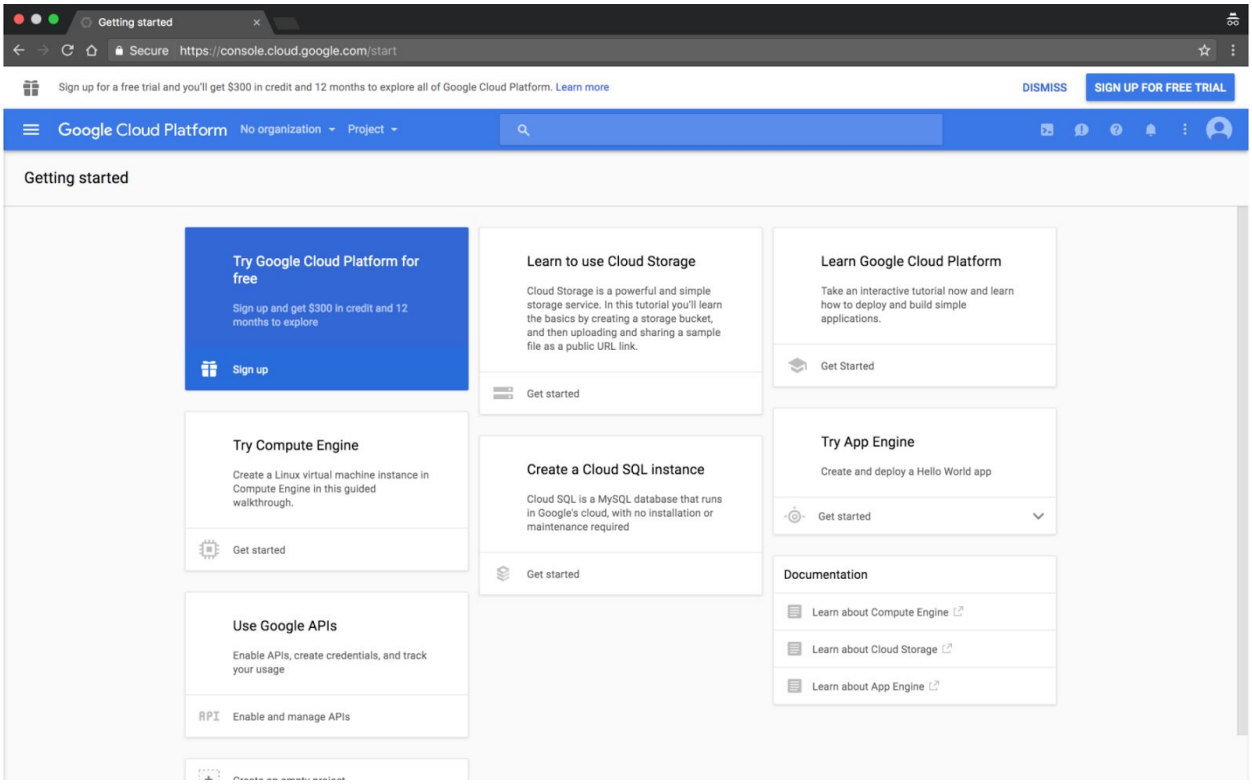
August 27, 2017

1. Get a *cloud.cs.columbia.edu* account

1. Sign up for a cloud Columbia CS account using this link. Note that is is an entirely new account and is not your normal Columbia account and is not your CS account. Your cloud Columbia CS account will be in the form: yourUNI@cloud.cs.columbia.edu.

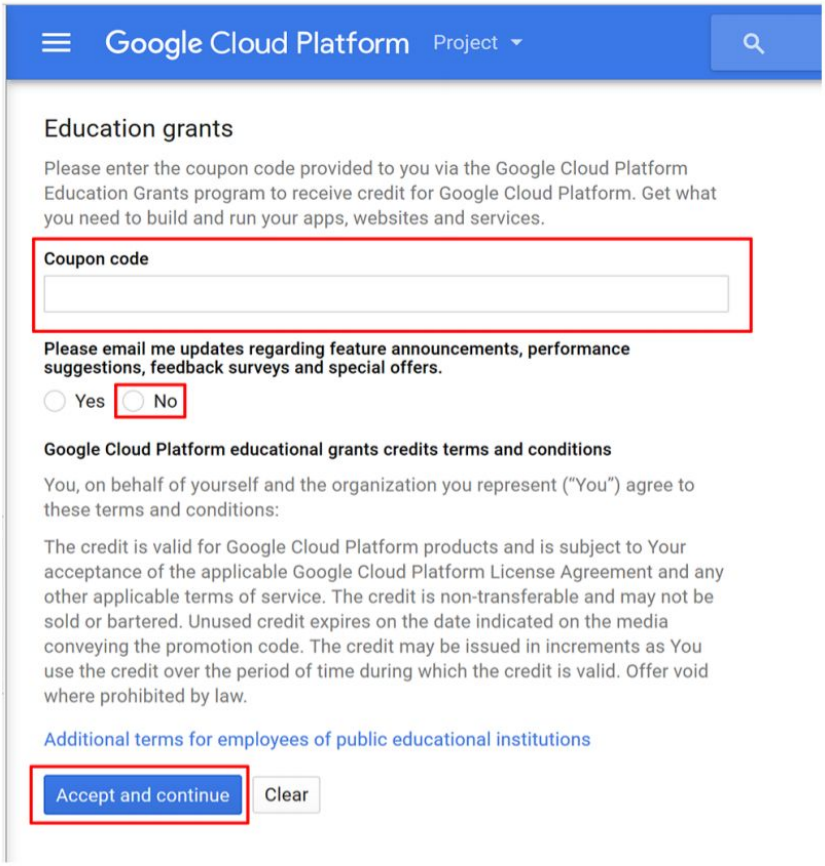
2. After you create your account, you should get an email with a temporary password to log into your cloud account. Log in to your Google Cloud Platform(GCP) account HERE using a private browsing window i.e. incognito (this is useful to prevent you from redeeming the coupon to a wrong account). Change your password.

On successful login, you should see the following page:



2. Redeem the Coupon code

1. Double check that you are logged in with yourUNI@cloud.cs.columbia.edu account! Also ensure that you are on an incognito browser window.
2. Check your email for a coupon code sent by the teaching staff via Canvas (Courseworks) to you.
3. Go to https://console.cloud.google.com/education
4. Accept any kind of agreement confirmation that Google asks you
5. On the "Education grants" screen, enter the coupon code



7. Select No for "Please email me updates ..." unless you actually want email updates on new offerings.

8. Click on Accept and continue.

9. You will be brought to the GCP web console. To check your coupon is redeemed successfully, check the menu at upper left corner -> Billing. You should see your coupon there.

3. Create a project

On the GCP console, Click the dropdown menu of projects, and click *Create project* (i.e. the + sign). Enter your own project name and click *Create*.

4. Generate a private/public key-pair

To gain access to the VMs you create in your GCP project, you will need to provide, to GCP, your "public key" that is paired with your "private key". Your private key, as the name implies, should be kept secret. On the other hand, your public key is broadcast publicly, and the VMs you create on GCP will use your public key to authenticate you, i.e. the holder of the private key corresponding to the known public key.
What this means for you is that you must copy your private key to every computer that you want to use, to access your Google VM. This makes it difficult to use public computers to do your assignments but provides security.

A full guide on how to generate key-pairs can be found here. However, not all the steps in this guide are necessary, so we reproduce below the bare minimum required steps for generating an SSH key-pair.

4.1 Terminal Setup

In order to perform GUI operations on your VM (such as viewing the plots you are asked to create in Homework 0), you will need a terminal that is capable of X11 forwarding. This allows your VM to use your local machine's display and peripherals for graphics.

Windows Users, download MobaXTerm. Note that MobaXTerm uses Linux-style commands.

Mac/Linux Users, Edit the sshd_config file (sudo vim /private/etc/ssh/sshd_config, or sudo vim /private/etc/sshd_config), and change X11Forwarding to yes (Uncomment if commented i.e. remove the leading #). Also make sure that you do ssh -Y uni@instance_ip, instead of just ssh uni@instance_ip, when trying to logon to the instance, as mentioned in Step 7 below.

4.2 Key Generation

1. Open a terminal window and type the following two commands:

```
ssh-keygen -t rsa -f ~/.ssh/my_w4705_key -C [MY_UNI]
chmod 400 ~/.ssh/my_w4705_key
```

The ssh-keygen command will create a private key file at ~/.ssh/my_w4705_key and a corresponding public key file at ~/.ssh/my_w4705_key.pub.The chmod command restricts the permissions on the private key—which must be kept secret — on your local machine so that only your user (and/or root) can view it.

1.5 (**Windows**). Windows users on MobaXTerm will need to start the ssh agent before running the next step. Execute:
eval 'ssh-agent -s'

If after running this command, you receive the error "**Could not open a connection to your authentication agent**." on running the next step, execute the following:
ssh-agent -s

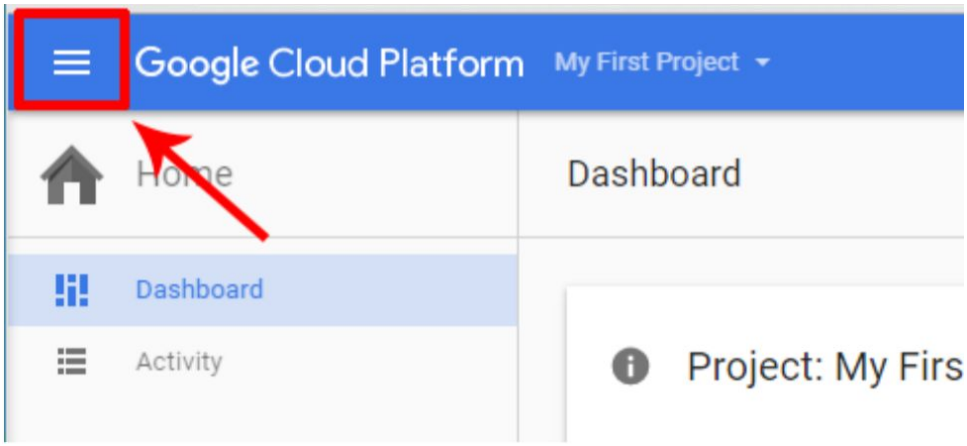
And then copy-paste the output of the above command into your command prompt. The above command will output a code, and you would need to run that code directly, by pasting it onto the command prompt and pressing enter.

2. For convenience, add the private key to your ssh-agent:
ssh-add ~/.ssh/my_w705_key

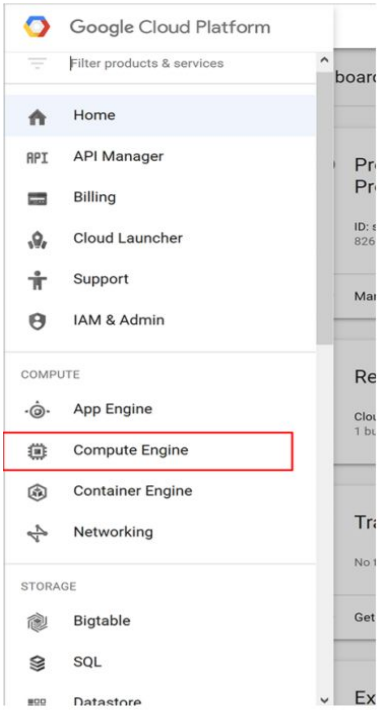
5. Add your public key to your GCP project

Now that you have your private/public key-pair, you need to add the public key to GCP. This will allow GCP VMs to authenticate anyone who has your private key.

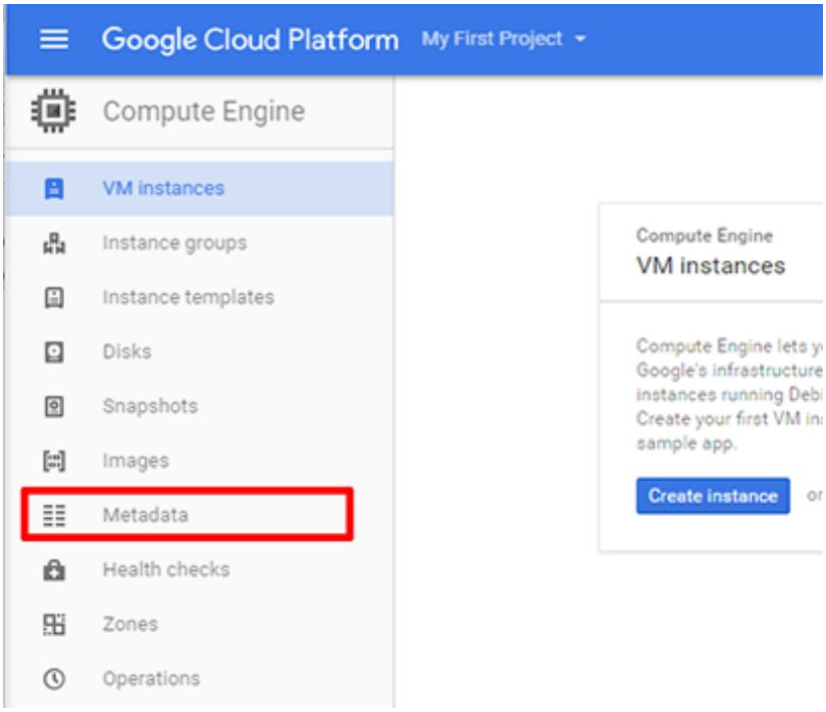
1. Navigate to your dashboard at https://console.cloud.google.com/ and click on the icon on the top left as shown below.



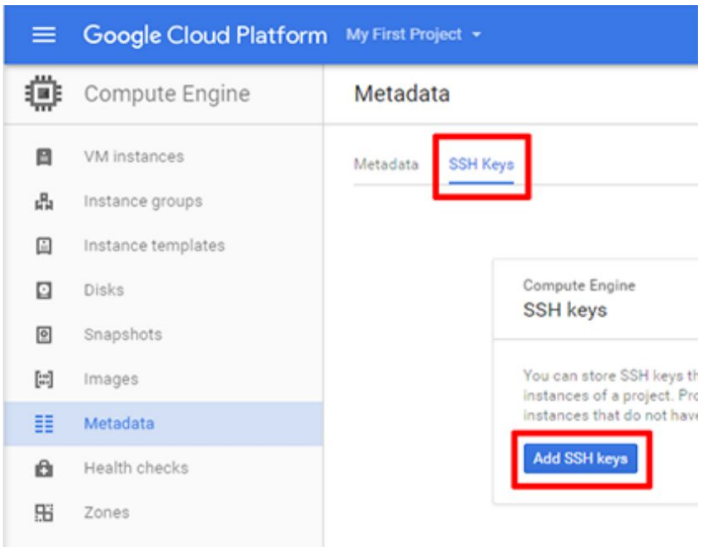
2. In the menu that expands, click on Compute Engine as depicted below.



3. You will now be on the Compute Engine portion of GCP which has its own menu on the left side. Click on Metadata on the menu.



4. Click on SSH Keys in the “Metadata” content area and then click on Add SSH keys.



5. Copy and paste the entire contents of the public key (my_w4705_key.pub) you generated in Section 4 into the “Enter entire key data” field. (You can open the public key in a text editor to view its contents.) The key you paste should begin with the text ssh-rsa.

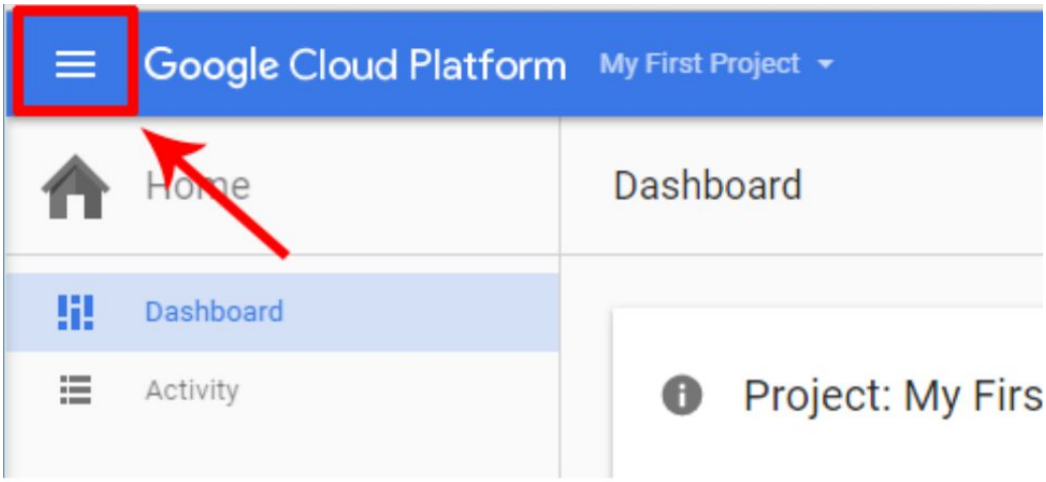
6. When you paste your public key, it automatically fills in the “Username” field. The “Username” field is the login username associated with the public key. You can change the “Username” field to your UNI by modifying the very last part of the public key that you pasted.

7. Click Save

6. Create virtual machine instance

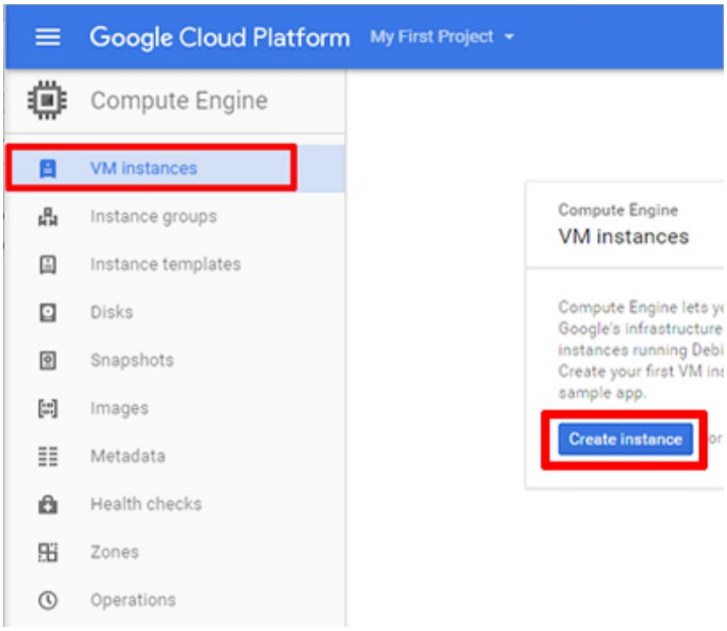
Now you will configure and create a virtual machine instance on GCP’S Compute Engine. This virtual machine is where you should test your programming assignments. You must exactly follow these instructions. Otherwise, your environment may differ from the graders’ VMs and cause issues with running your assignment submissions.

1. Navigate to your dashboard at https://console.cloud.google.com/ and click on the icon on the top left as shown below



2. In the menu that expands, click on Compute Engine

3. In the Compute Engine section, click on VM instances on the left menu. At this point, you may need to click on Enable billing, which will then ask you to choose a project to enable billing on. Select the appropriate project, and then click on Create instance.



4. Use the following settings:

- Zone: us-east1-d
- Machine type: Decide based on your requirements
- Boot disk: Ubuntu 14.04 LTS

Then click on Create.

PS: Till HW2, you would not require an instance with GPUs. Therefore, don't configure your instance with it, which will exhaust you coupon amount faster. If an assignment requires you to reserve a GPU, follow instructions mentioned in Step 10 below.

7. Accessing your virtual machine

You can access your GCP VM via SSH similar to how you access a CLIC machine via SSH. Some important points:

- The username you use must be the "Username" that you entered for your SSH key in Section 4.
- The IP address you connect to is the "External IP" for your VM instance. This IP will change every time you restart your instance, so take note.
- You must use the private key that you created in Section 4.
- Example: On Linux/Mac OS X/Windows 10 with Ubuntu (Windows' Bash), use the following command to SSH into the VM instance that was setup in this guide:

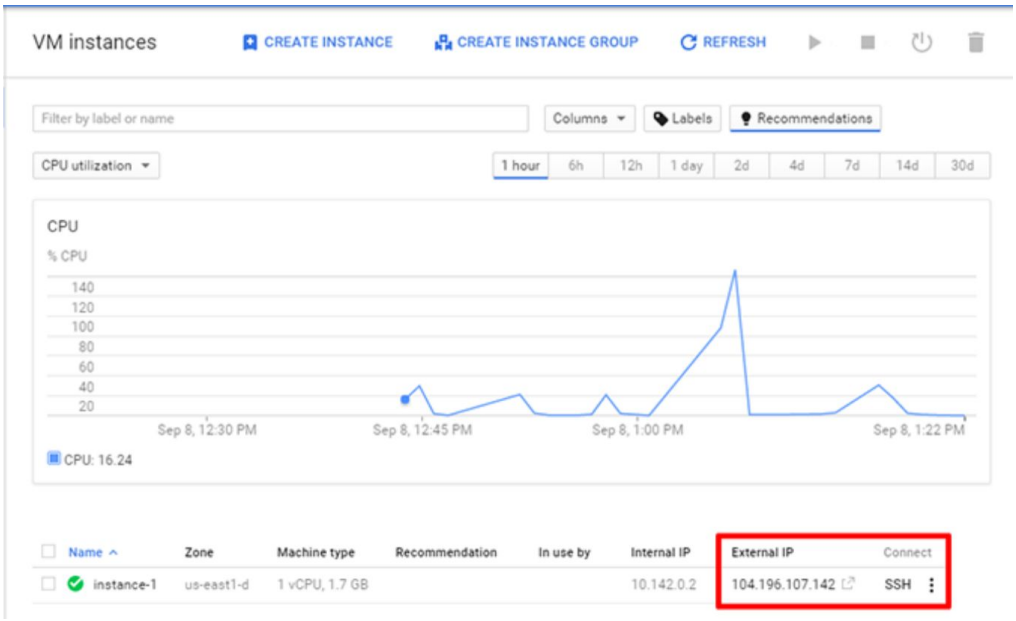
Windows: ssh <Your_UNI>@104.196.107.142

Mac: ssh -Y <Your_UNI>@104.196.107.142

Note that <Your_UNI> is the username that GCP parsed out from the public key that was pasted into their console, and that 104.196.107.142 is the external IP address of the VM instance that was launched.

For generic SSH client details, see GCP's guide at
https://cloud.google.com/compute/docs/instances/connecting-to-instance#standardssh

For quick access to look around, you can also use GCP's browser SSH client which you launch simply by clicking on the SSH button on the "VM instances" console. But the browser SSH client will not have X11 forwarding configured, which might be required based on the HW.



8. Setting up your development environment

1. SSH into your VM instance
2. Install gcc, perl and g++ using this command

   sudo apt-get update && sudo apt-get install gcc perl g++ linux-source linux-headers-$(uname -r) linux-image-extra-$(uname -r) linux-image-extra-virtual -y && sudo reboot

   PS: During reboot, your connection to the instance will be lost and you'll have to reconnect. Reconnect after 30 secs as reboot would take some time.

3. Install python and upgrade pip, if not already present

   1. *sudo apt-get install python-pip python-dev build-essential*
   2. *sudo pip install --upgrade pip*

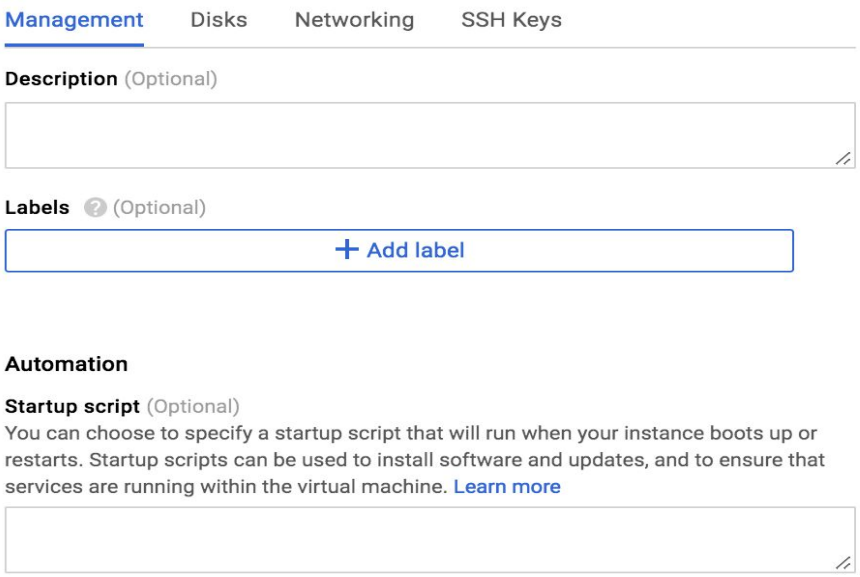
9. Shutting down your VM instance

Always ensure that you stop/delete your instance when you are not using it, otherwise you will keep on incurring instance costs and will exhaust your Cloud coupons. The coupons are limited to 1 per student and the coupon value is enough to make sure you complete your first 2 assignments, if you are using it judiciously. You can stop/delete your instance by clicking the 3 vertical dots next to "SSH", as shown in the red box, on the image above.

10. Additional instructions for reserving GPU instances (required after HW2)

9.1 Reserving instance with GPU

- Zone: us-east1-d
- Machine type: Decide memory based on your requirements.
  - Click customize for GPU options.
  - Select GPUs: 1 x NVIDIA Tesla K80 (Would require a 1 time quota approval)
- Boot disk: Ubuntu 14.04 LTS
- For configuring CUDA on instances with GPUs, click the drop down "Management, disks, networking, SSH keys". Scroll down to "Automation" section as shown below



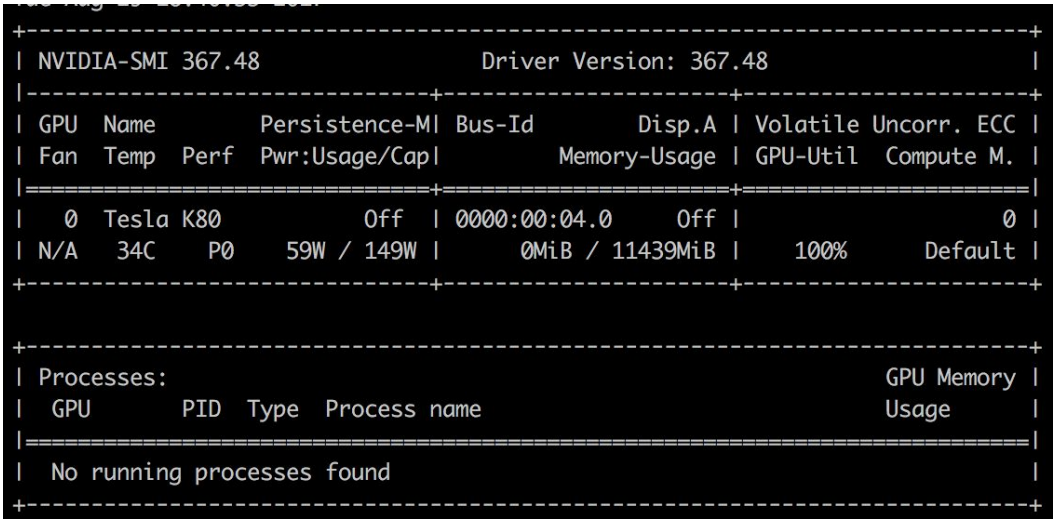
- Enter the following script in the "Startup script" section:

```
#!/bin/bash
echo "Checking for CUDA and installing."
# Check for CUDA and try to install.
if ! dpkg-query -W cuda; then
  curl -O
http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1
404/x86_64/cuda-repo-ubuntu1404_8.0.61-1_amd64.deb
  dpkg -i ./cuda-repo-ubuntu1404_8.0.61-1_amd64.deb
  apt-get update
  apt-get install cuda -y
  apt-get install linux-headers-$(uname -r) -y
fi
```

Then click on Create.

9.2 Checking CUDA installation and installing Keras with backend as TensorFlow

1. SSH into your VM instance
2. Type “nvidia-smi” and press enter
3. A successful CUDA installation should give you an output similar to the one below (It might take sometime after spawning the instance, for nvidia-smi to work):



3. In this course, we will be using Keras with TensorFlow backend. Therefore, install TensorFlow by executing :

sudo pip install --upgrade tensorflow-gpu

4. Install Keras

sudo pip install keras