**Ansible**

Ansible is an IT automation engine that automates cloud provisioning, configuration

management, application deployment, intra-service orchestration, and many other IT needs. It uses no agents and no additional custom security infrastructure, it uses a quite simple language (YAML, in the form of Ansible Playbooks) that allow you to describe your automation jobs in a way that approaches plain English.

**Assignment**

* Utilize the Dcloud DevNet Express Data Center v2 lab, or deploy your own three (3) Virtual Machines
* Configure Ansible server on VM 1 to deploy a webserver to VM2 and VM3 on port 8080 that displays the message: “Hello World from SJSU-X”, where X is 1 or 2 depending on which webserver.
* Include in the Ansible playbook, plays to deploy and un-deploy all the webserver resources.

Launching 3 EC2 instances (VMs) on AWS:

1. Visit https://aws.amazon.com/ec2/ Simply use your username and password to establish a free tier account.

2. Choose Launch a virtual machine with EC2 from the menu.

3. Enter a name for the VM in names and tags.

4. Under Amazon Machine Image, choose the desired operating system.

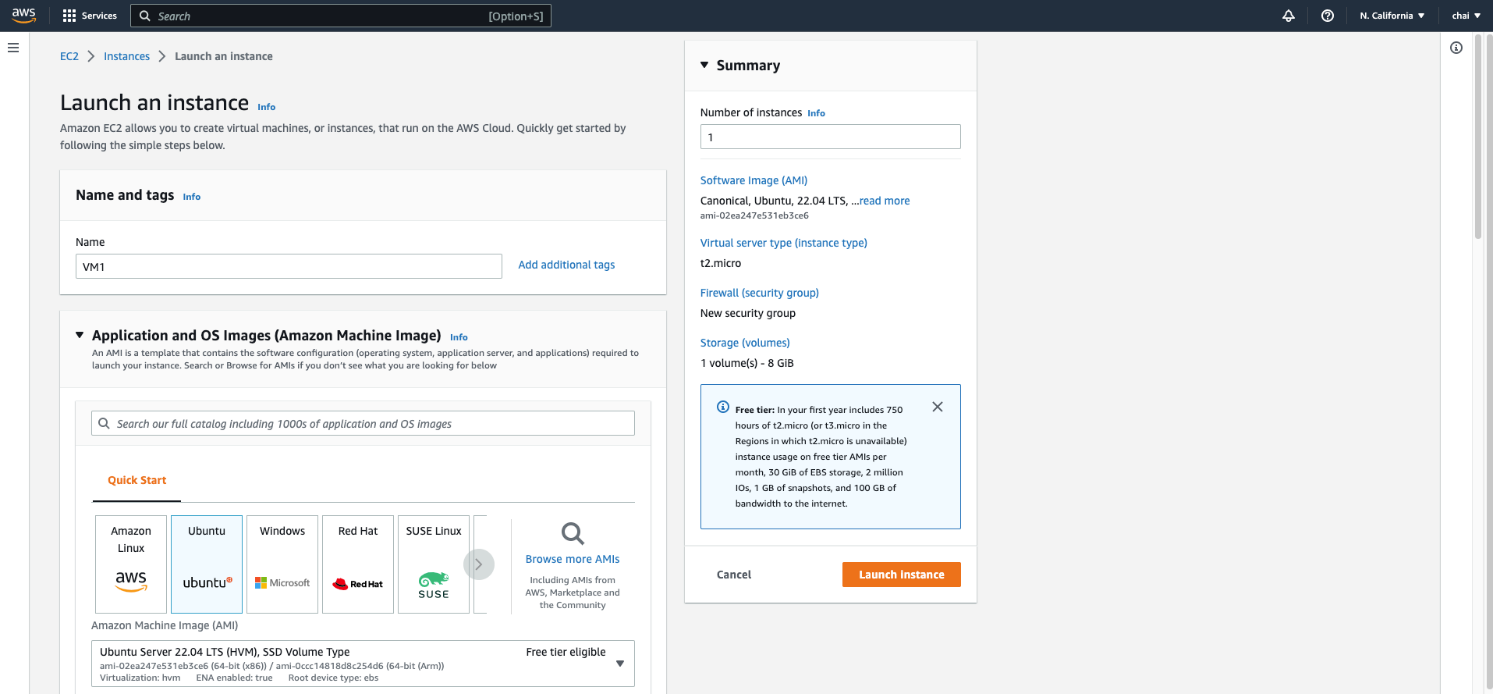
Choose Ubuntu 22.04.

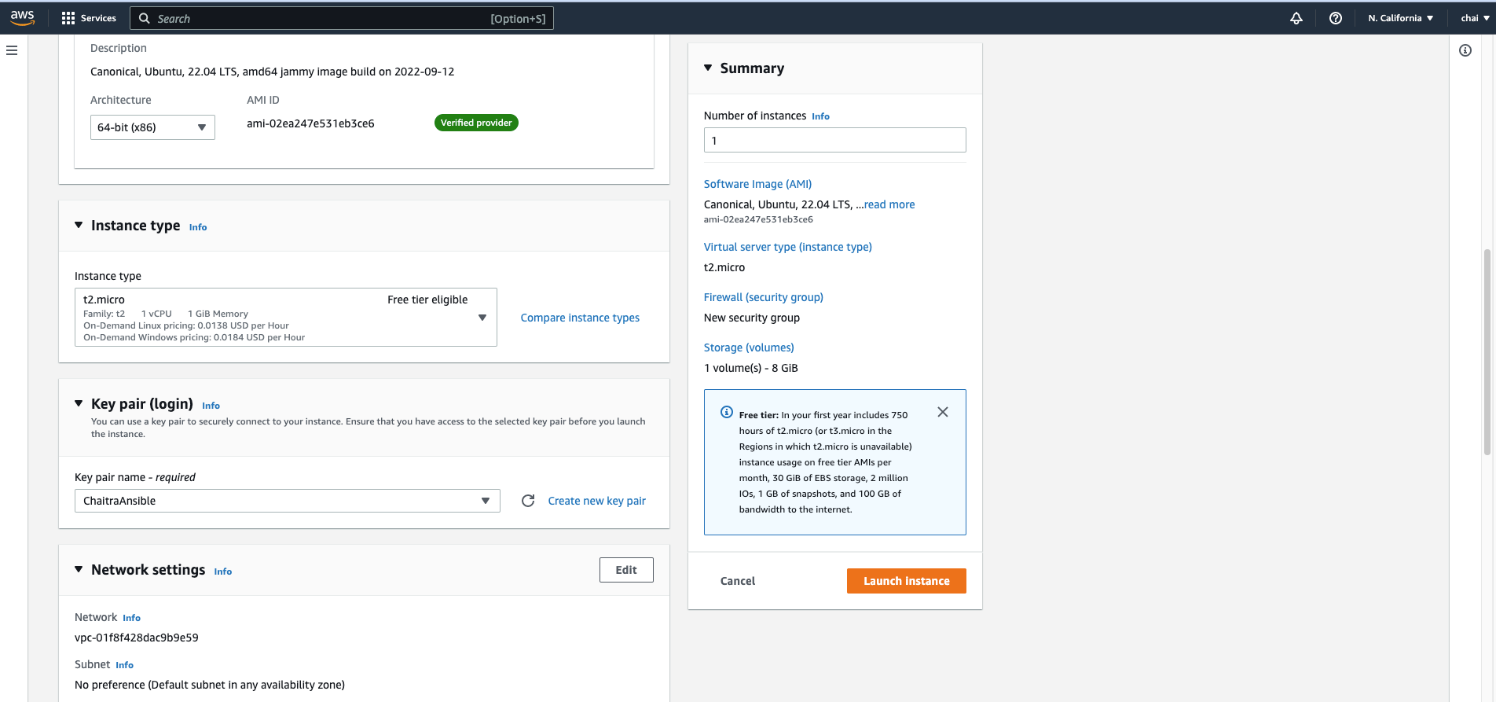
5. Choose the instance type t2.micro.

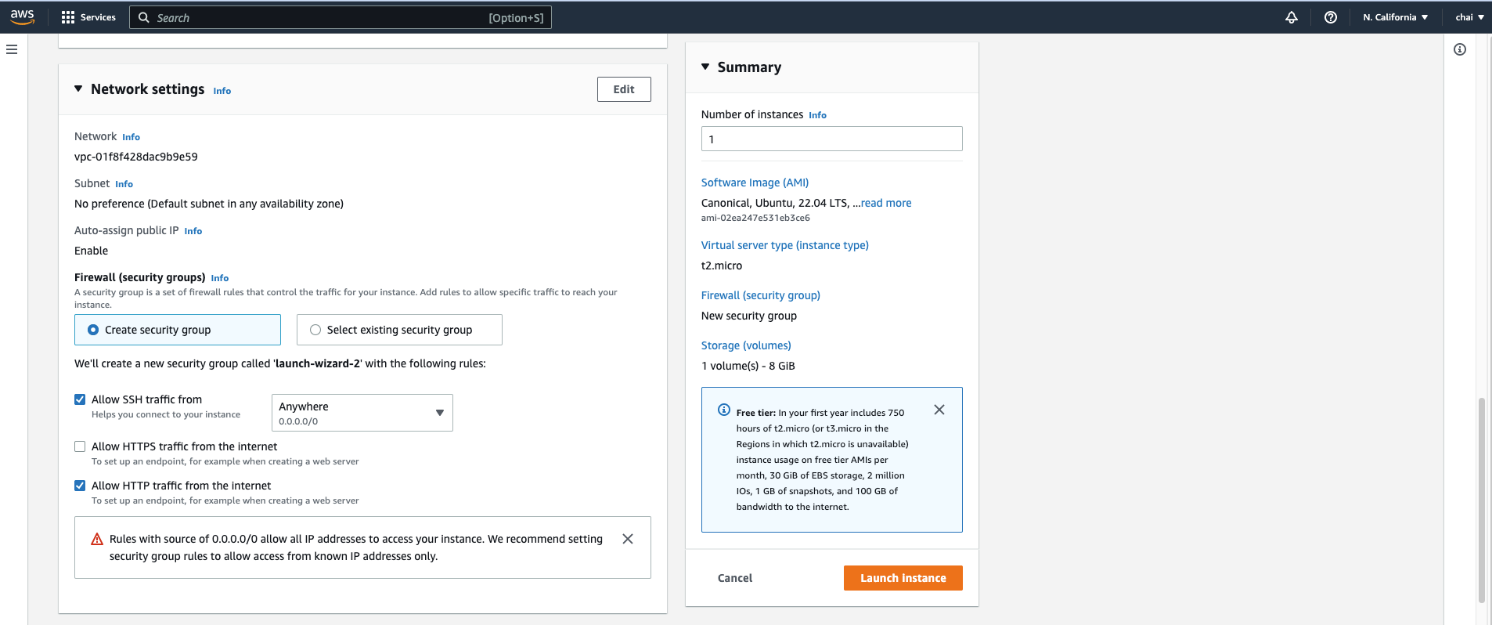
6. Under keypair, choose "Create New Key Pair," give the keypair file a name, and then click "Create."

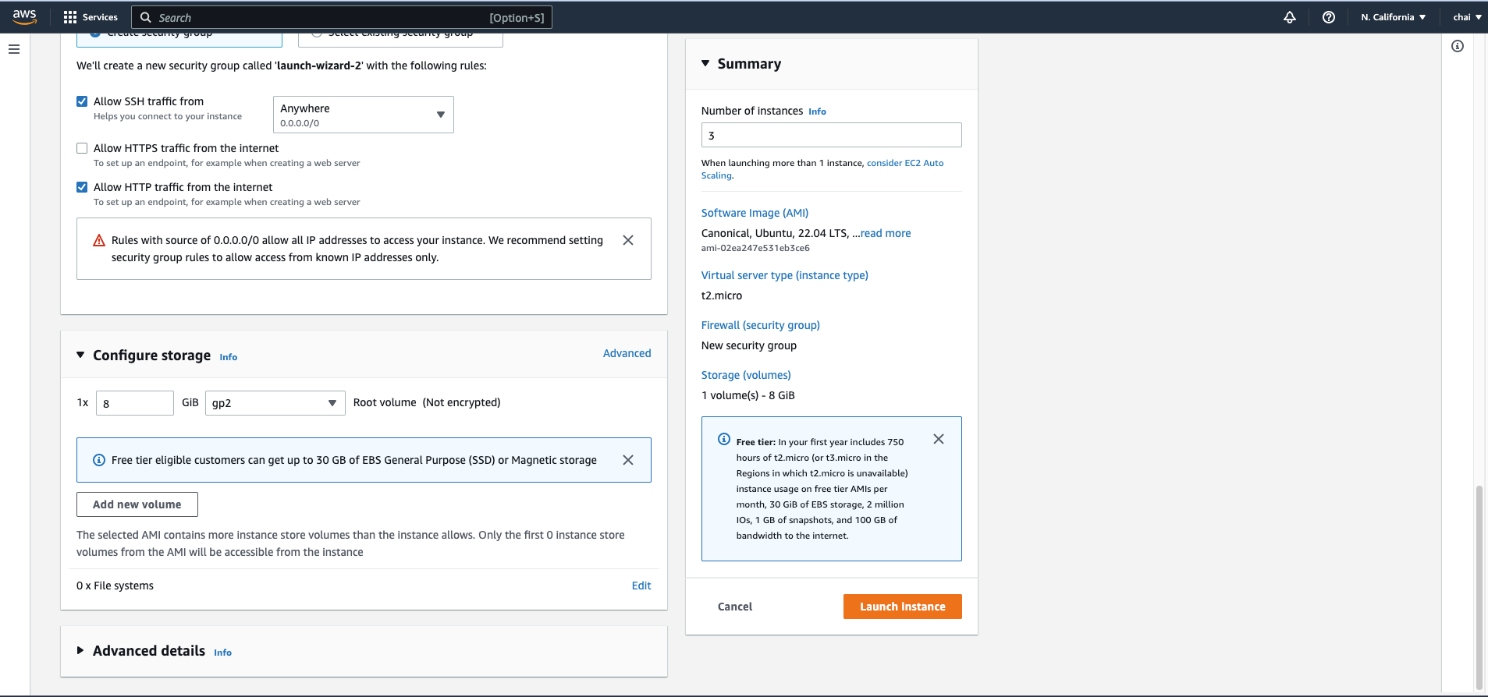
make sure to download it and save it in a safe place.

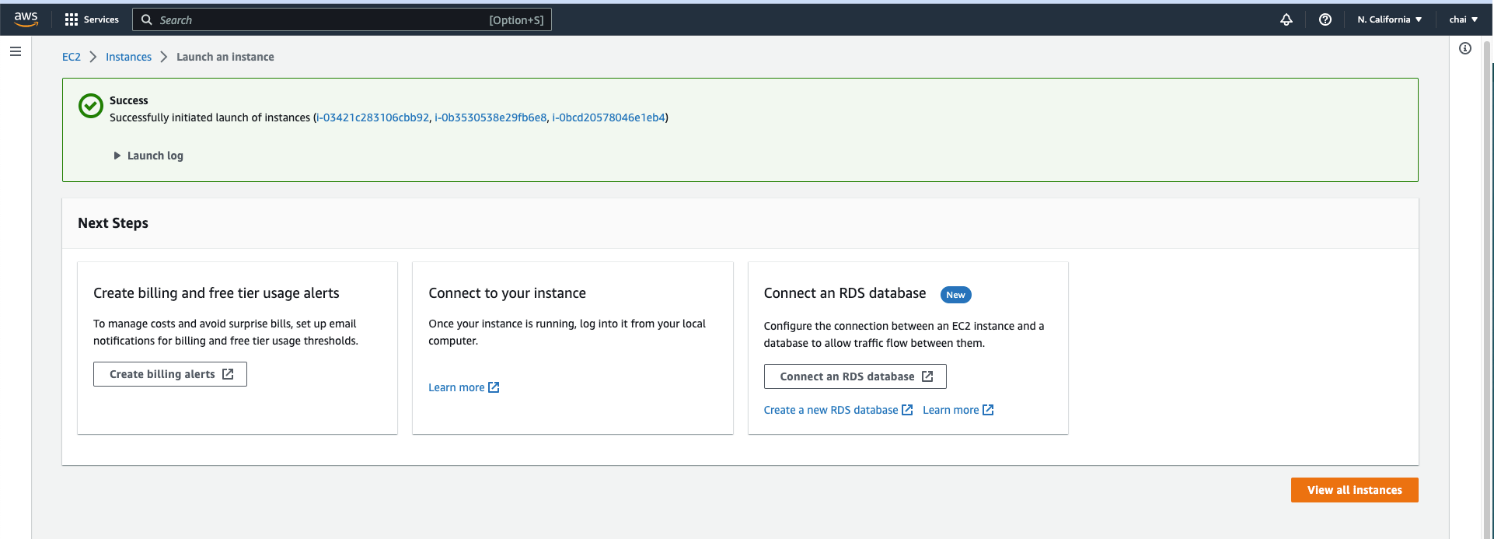
7. Establish a security group that permits HTTP and SSH traffic in network settings.







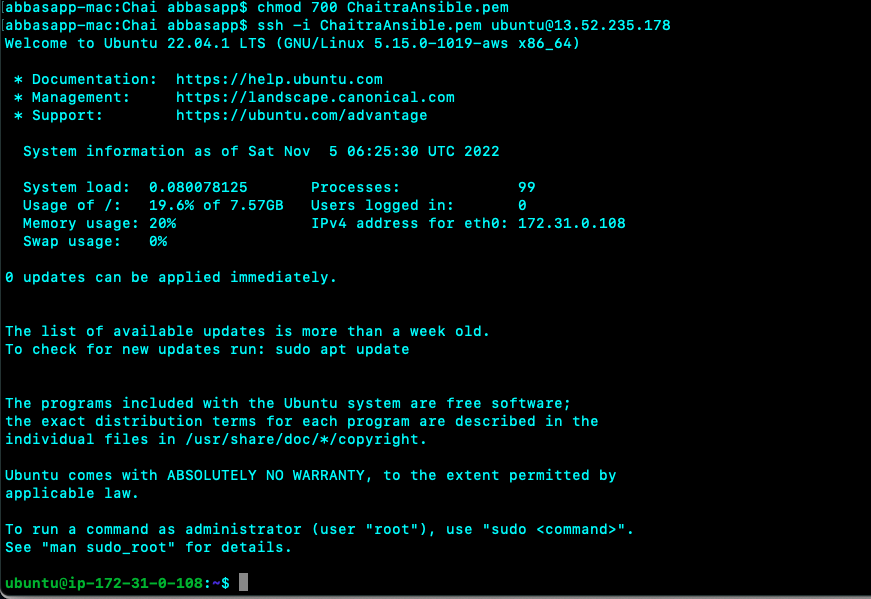




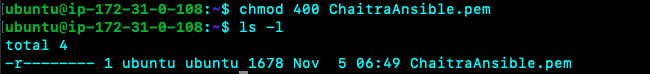
Connecting to EC2 instance using ssh:

1. In the terminal window use the ssh command to connect to your instance. Use the following command: ssh -i /path/key-pair-name.pem ubuntu@instance-public-dns-name

2. Enter yes when prompted whether to continue. Now we have successfully connected to the EC2 instance.







3. Repeat the above for the other two EC2 instances.

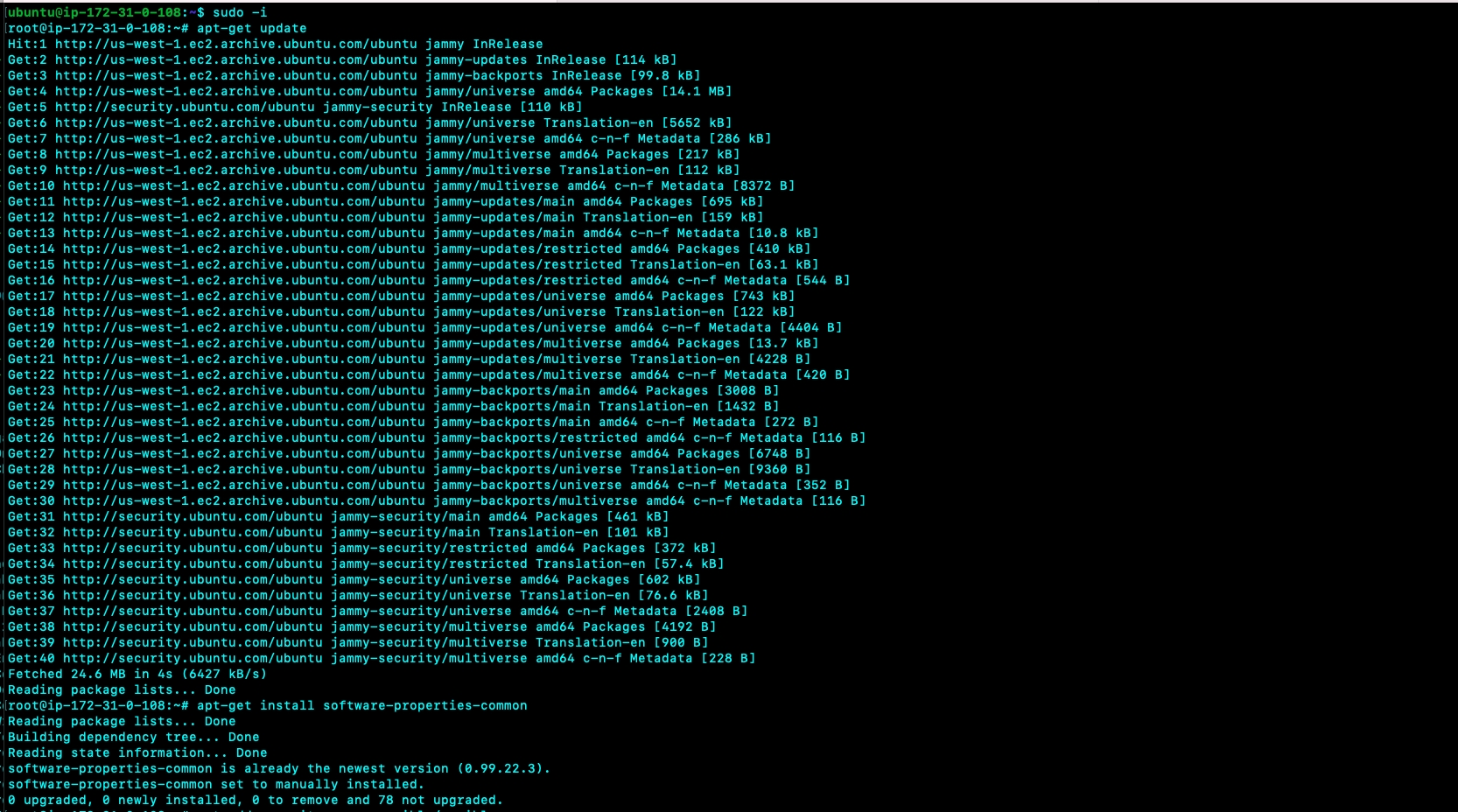
4. Copy the pem file that was downloaded when creating the user instance into the control node using scp command:

scp -i path/keypair.pem –r keypair.pem ubuntu@instance-public-dns-name

5. We must reduce the permission to this pem file using chmod.

Chmod 400 path/keypair.pem





Setting up ansible:

1. Check for python installation:

python3 --version

2. Install pip using these commands:

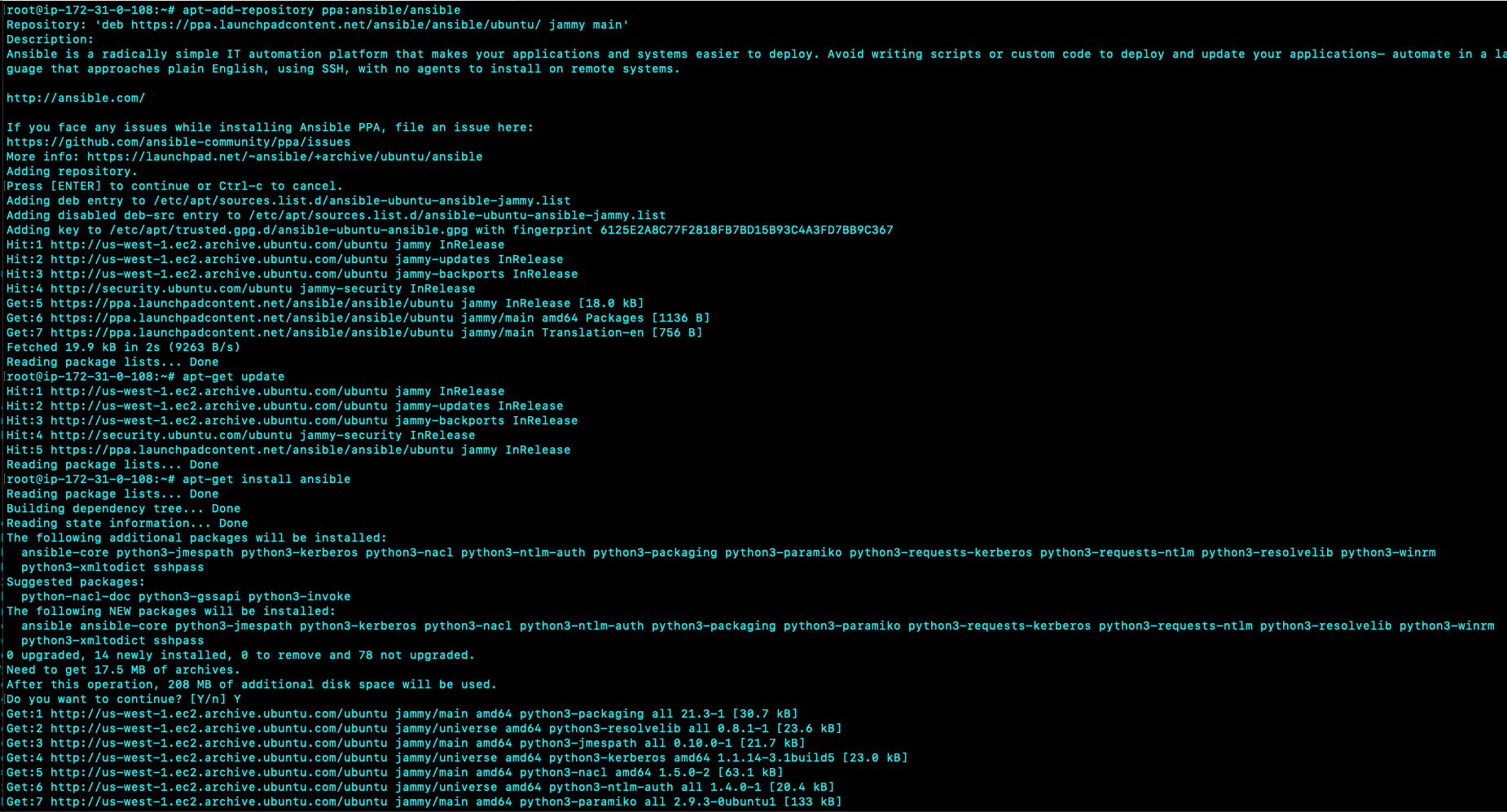
curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py python3 get-pip.py --user

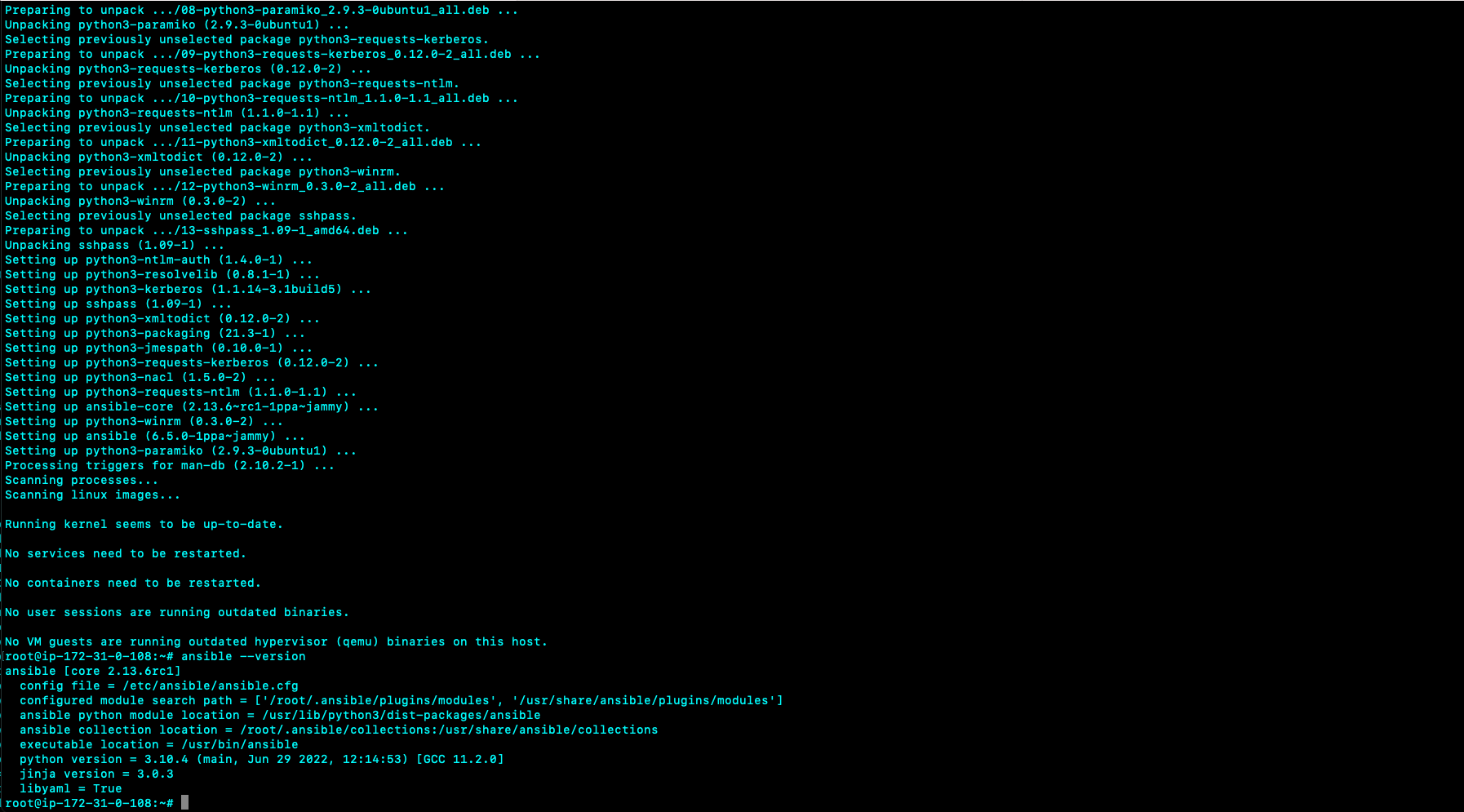
3. Now we install ansible as follows:

sudo -i  
apt-get update  
apt-get install software-properties-common  
apt-add-repository ppa:ansible/ansible  
apt-get update  
apt-get install ansible

4. Check the version :

ansible –version



Creating a playbook to deploy Apache server and bring up the webpage:

Writing the hosts' file:

1. We must write the hosts file to tell ansible which nodes/hosts to talk to.

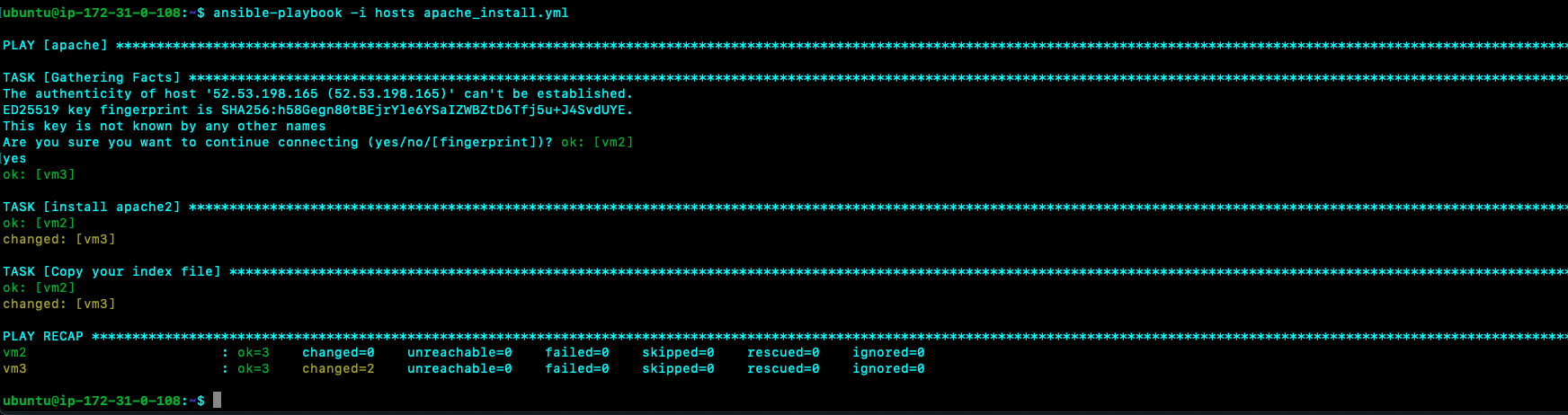
2. Create a file named hosts and include in the following code:

[apache]

vm1 ansible\_host=ip\_addr ansible\_ssh\_private\_key\_file=~/.ssh/keypair.pem

vm2 ansible\_host=ip\_addr ansible\_ssh\_private\_key\_file=~/.ssh/keypair.pem [all:vars]

ansible\_python\_interpreter=/usr/bin/python3 Here vm1 and vm2 are the names of the hosts. Ansible\_host is used to specify the IP address of the hosts. Ansible\_ssh\_private\_key\_file is the private key used by ssh.It is useful when using multiple keys and you do not want to use ssh-agent



Creating the playbook to install apache server on the hosts:

1. Create a file called apache\_install.yml

2. Add the following code:

---

- hosts: apache

become: true

tasks:

- name: install apache2

apt: name=apache2 update\_cache=yes state=latest

- name: Copy your index file

template:

src: "index.html"

dest: "/var/www/html/index.html"

In the above code, Task 1 is to install apache and task 2 is to copy the index.html to the webserver.

3. Create a index.html file and add the following HTML code:

<head>

<h1>

# Hello World from SJSU- {{inventory\_hostname\_short}}!

</h1>

<head>

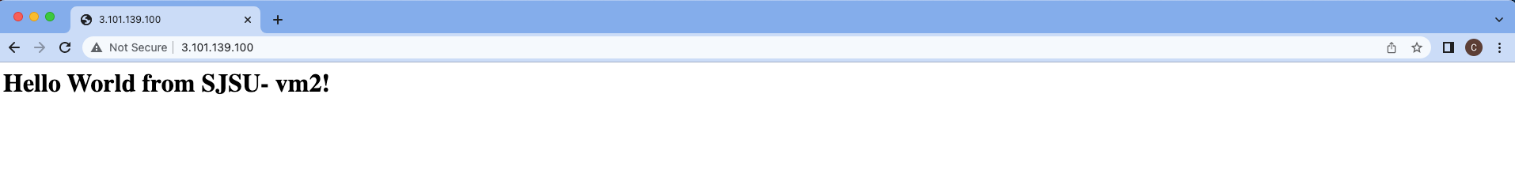
In the above code inventory\_hostname\_short is the current host being iterated over in the play.

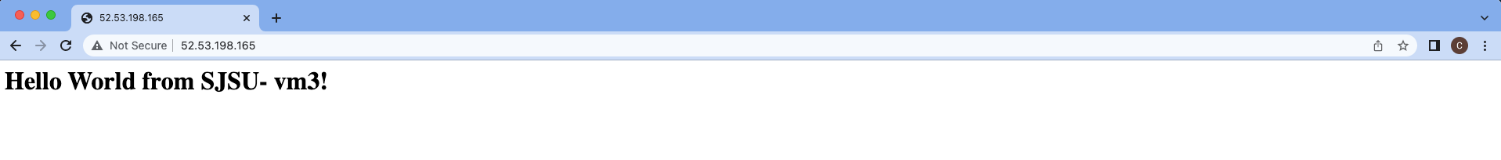
4. Now run the playbook with below command:

ansible-playbook -i hosts install\_apache.yml

Here -i overwrites the default inventory.

5. Then enter the IP address of both the hosts on the web browser to see if Hello World is displayed on them. The webpage is available on port 80.





Create a playbook to Un-deploy apache web server on the hosts:

1. Create a file named uninstall\_apache.yml.

2. Add the following code:

---

- hosts: apache

become: true

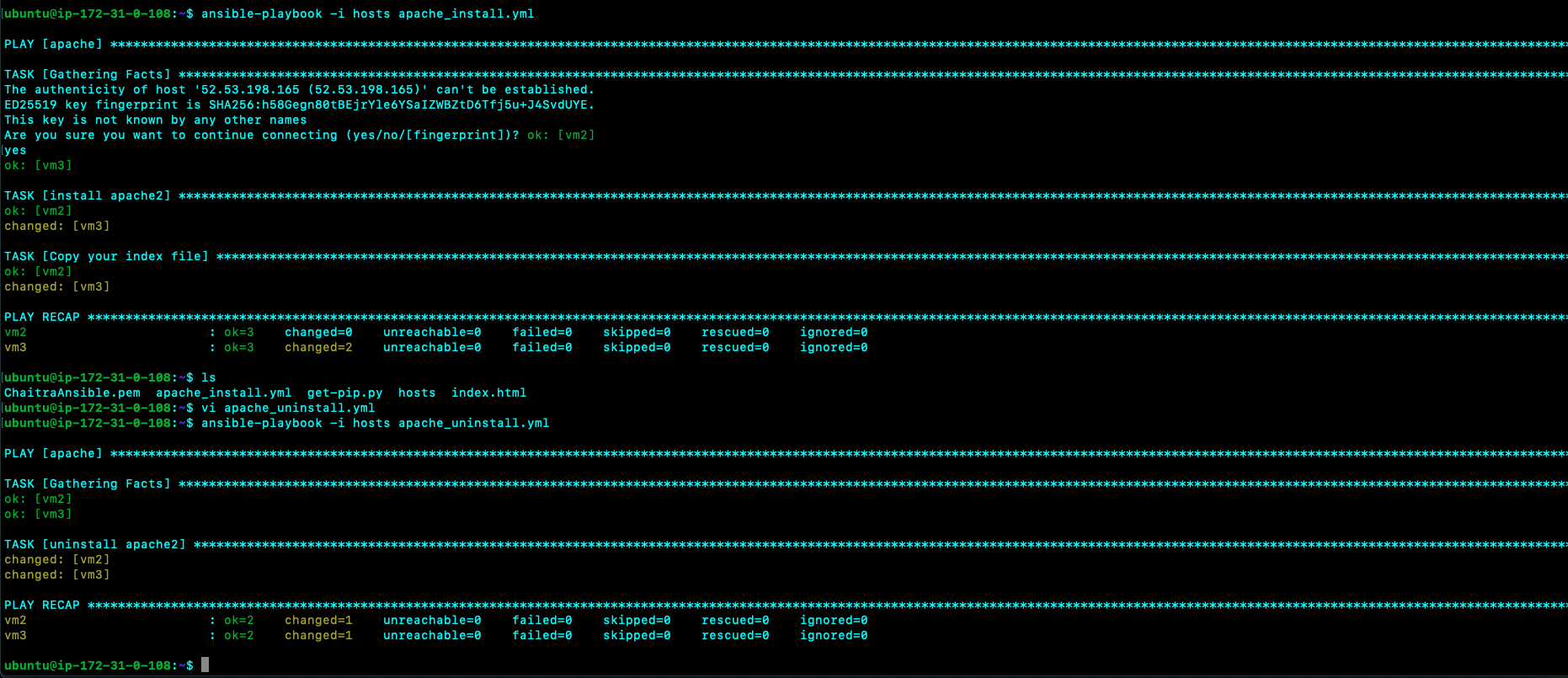
tasks:

- name: uninstall apache2 apt: name=apache2

autoremove=yes state=absent

3. Run the playbook using the following command:

ansible-playbook -i hosts uninstall\_apache.yml



Below are the screenshots when we try to access the webserver after uninstalling apache

