CSCL1030 Applied Lab 2 - Infrastructure Automation with Terraform and Ansible

Student Name: Vinod Kumar Dhanavath
Course: CSCL1030 –CloudOps Tools and Techniques
Instructor: Joba Hassan
Date: 4-02-2025

I. Introduction:

This lab exercise involved leveraging Terraform and Ansible to automate the provisioning and configuration of AWS infrastructure. The goal was to deploy a control node and four web servers, organizing them into distinct groups. This report provides a step-by-step account of the process, modifications made to the provided code, and evidence of successful execution through screenshots.

II. Infrastructure Deployment

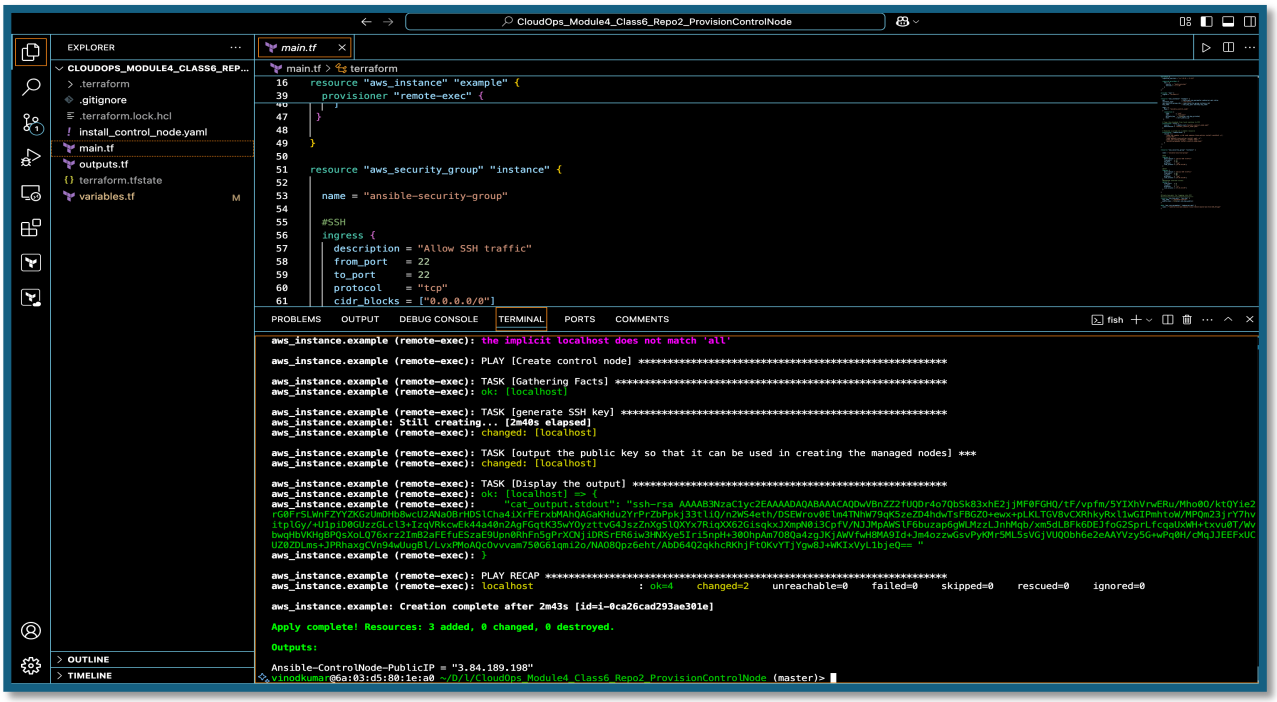
1. Cloning the Repository or Forked Repositories

The following repositories were cloned into my GitHub account:

- ProvisionControlNode
- ProvisionManagedNodes
- ControlNodeFiles

2. Setting Up the Control Node

Following the instructions from Class 6, the Control Node was provisioned using Terraform. The SSH key output was extracted and utilized to generate an **`id_rsa.pub`** key in the **ProvisionManagedNodes** repository. Following is the Control Node output with **Public Ip=”3.84.189.198”**



3. Deploying Managed Nodes

Four web servers were created and categorized into groups in the hosts.ini file.

[einstein] – The first two web servers

54.211.73.69

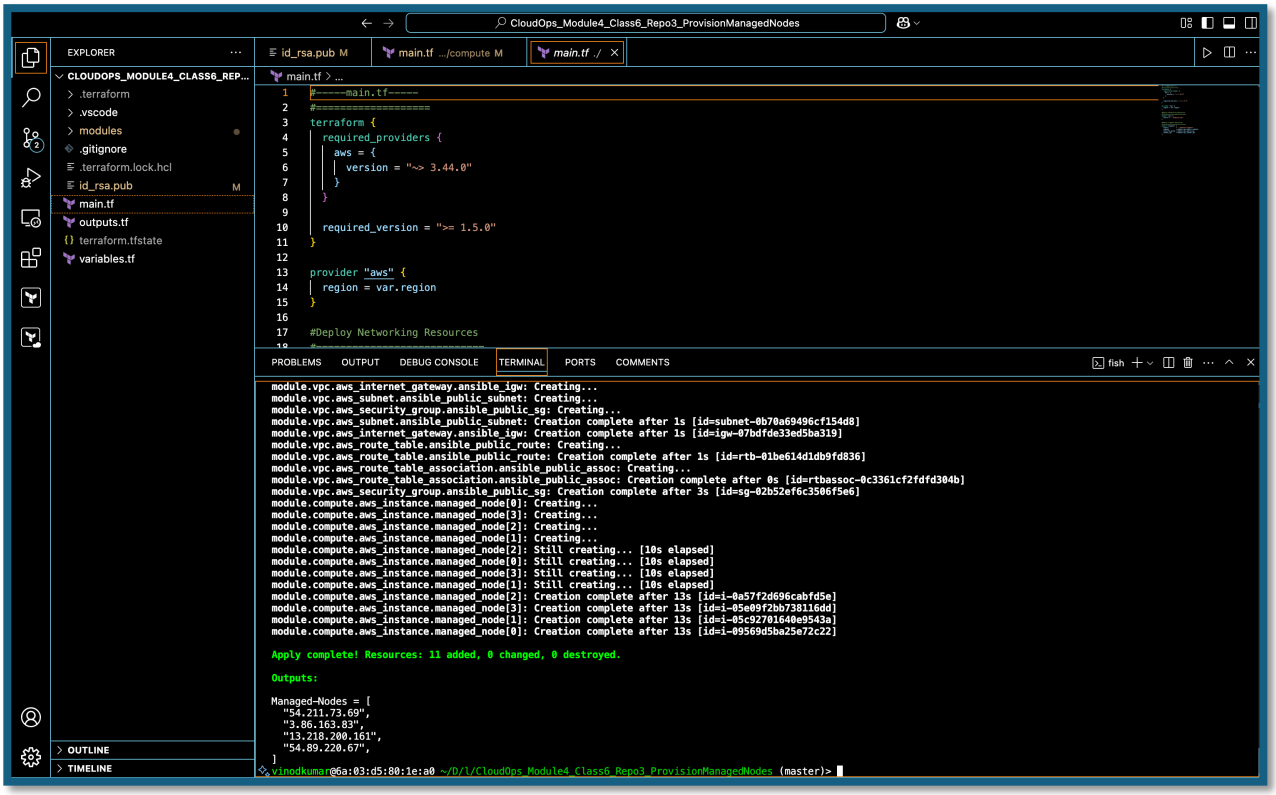
3.86.163.83

[Cloud] – The last two web servers

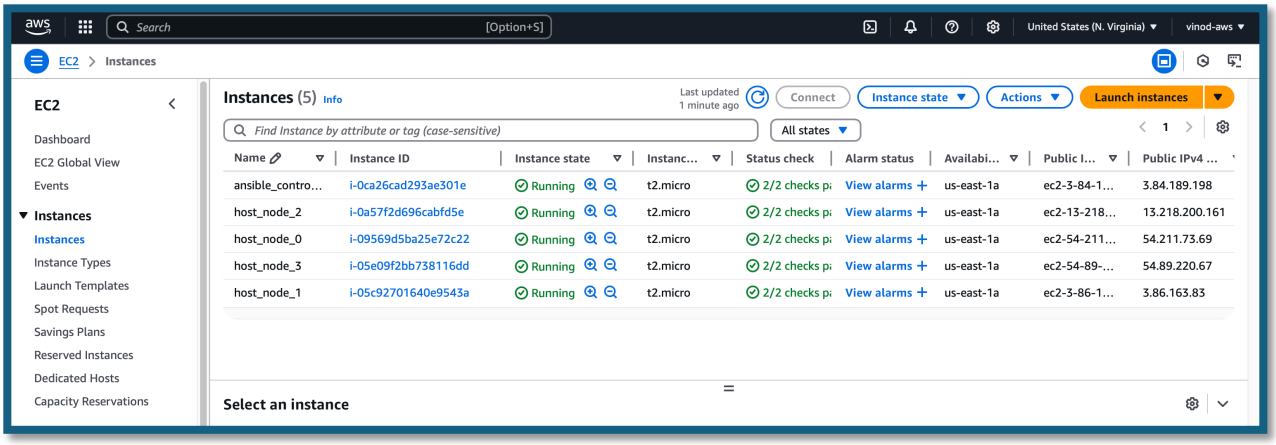
13.218.200.161

54.89.220.67

Following is the screenshot to demonstrate the all the managed nodes and their “Public Ips”.



Following is the screenshot for the AWS EC2 instance list showing all deployed servers with Public IPs.



III. Configuration with Ansible

4. Updating the Hosts File

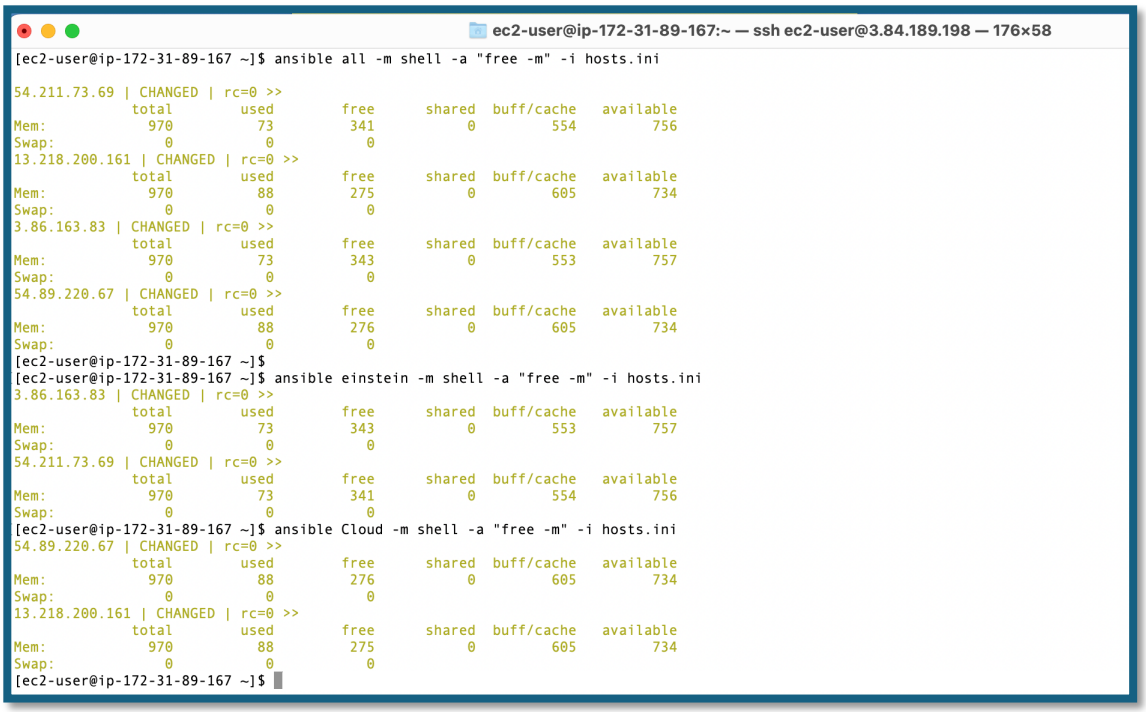
The `hosts.ini` file was configured as follows:

```
[ec2-user@ip-172-31-89-167 ~]$ vi hosts.ini
[ec2-user@ip-172-31-89-167 ~]$ ls
hosts.ini  install_control_node.yaml
[ec2-user@ip-172-31-89-167 ~]$ cat hosts.ini
[einstein]
54.211.73.69
3.86.163.83


[Cloud]
13.218.200.161
54.89.220.67
[ec2-user@ip-172-31-89-167 ~]$
```

Connected to all the managed nodes in the hosts.ini file resolving the python error adding the following line in the ansible.cfg on the default section.

interpreter_python = auto_silent

Following is the screenshot that establishes the connection to all the managed nodes also with group without Python error.



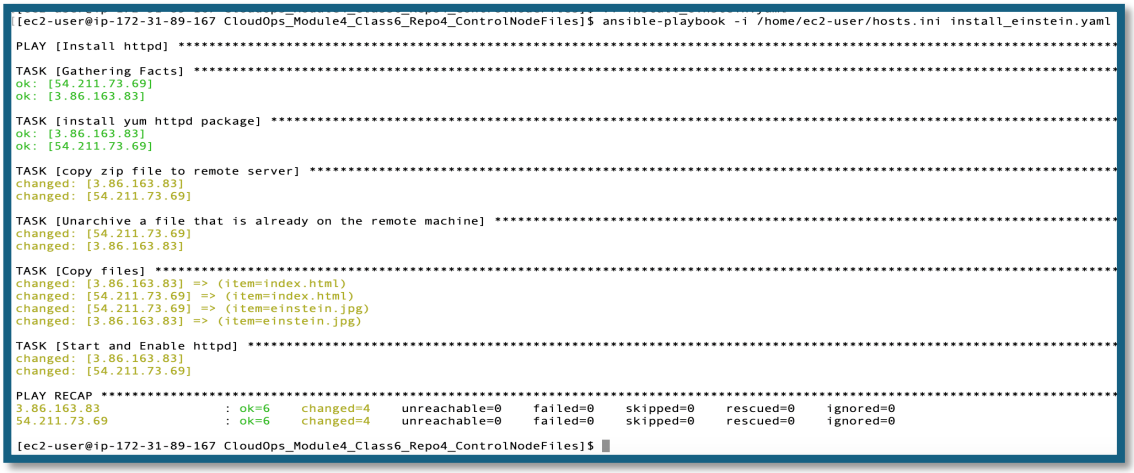
IV. Setting up Web Servers

Next, the code from the **ControlNodeFiles** repository has been successfully cloned and securely copied (SCP) to the home directory of the ec2-user on the control node.

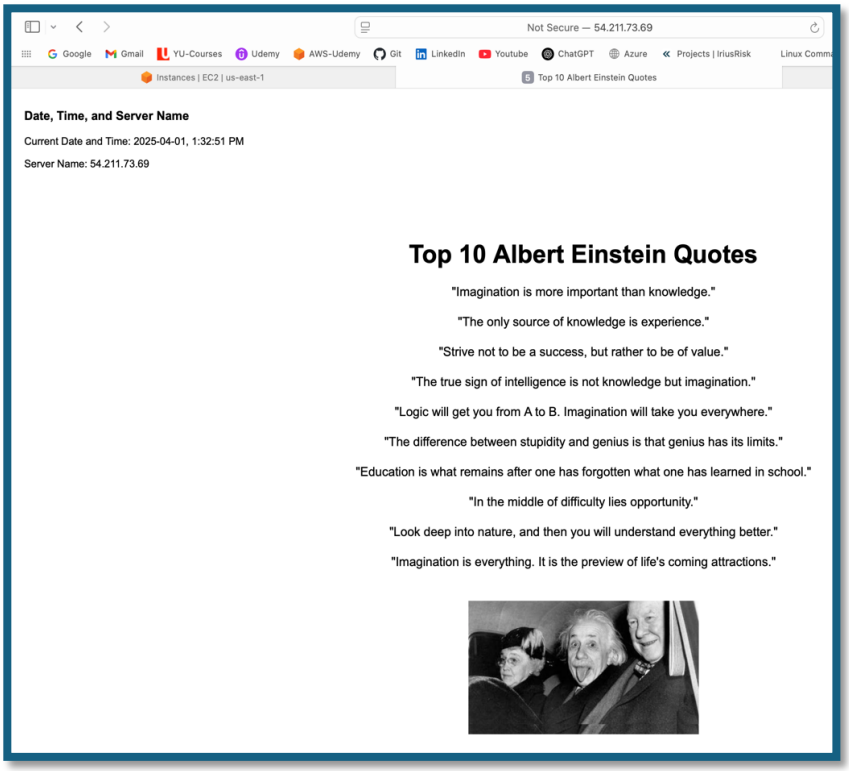
The install_httpd.yaml playbook was renamed to install_einstein. yaml and updated to target the **einstein** group.

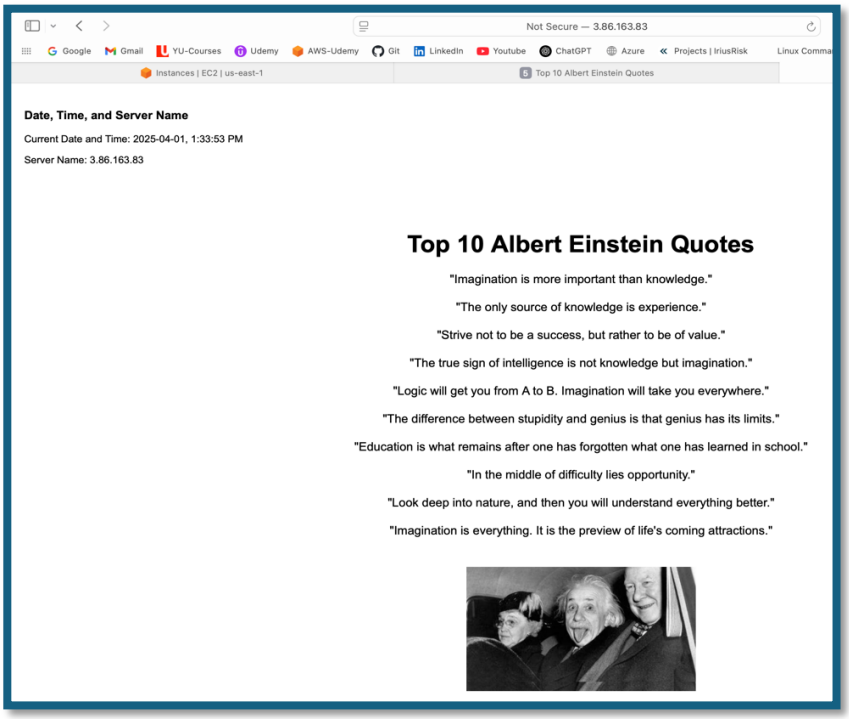
Executed using **ansible-playbook -i /home/ec2-user/hosts.ini install_einstein.yaml**

Execution output of install_einstein.yaml



Browser verification of Einstein web servers





5. Configuring the Cloud Group

Note: I have successfully cloned the CSCC1030 repository and used SCP to transfer it to my control node's home directory. I then moved it to the ControlNodeFiles folder within the same directory to execute the .yaml file.

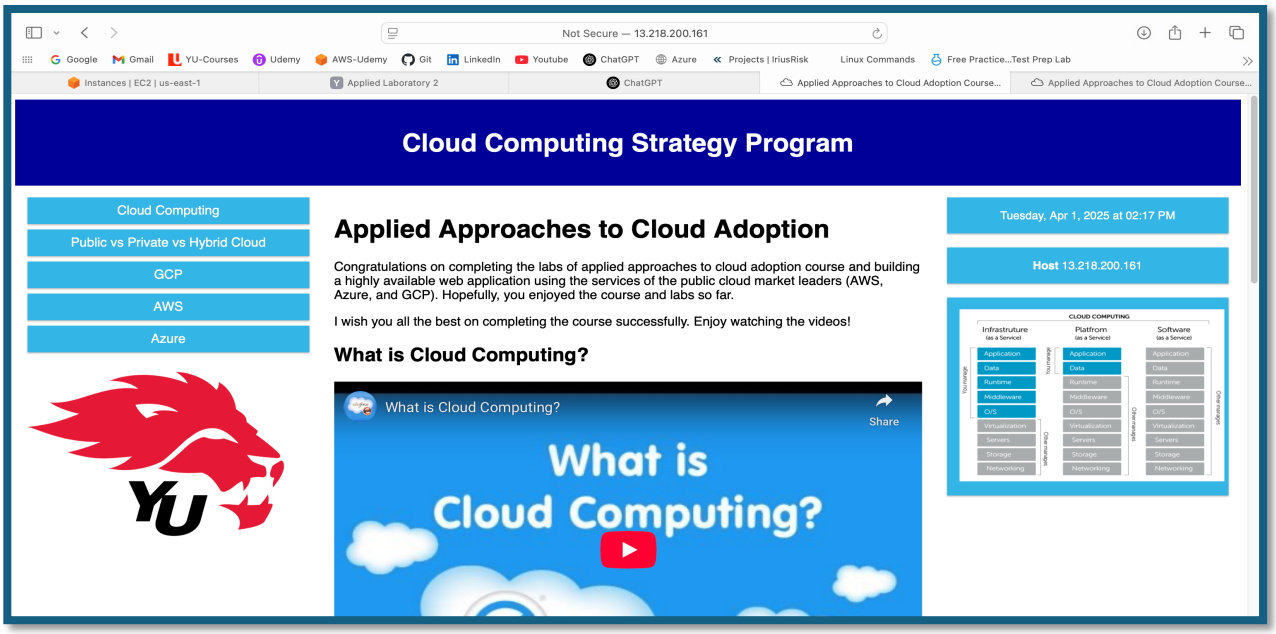
A new playbook **install_Cloud.yaml** was created to deploy the York University Cloud computing page to the **Cloud group.**

Executed using: **ansible-playbook -i ~/hosts.ini install_Cloud.yaml**

Execution output of install_Cloud.yaml

```
[ec2-user@ip-172-31-89-167 CloudOps_Module4_Class6_Repo4_ControlNodeFiles]$ vi install_Cloud.yaml
[ec2-user@ip-172-31-89-167 CloudOps_Module4_Class6_Repo4_ControlNodeFiles]$ ansible-playbook -i ~/hosts.ini install_Cloud.yaml

PLAY [Install and Deploy York U Cloud Page] ************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************
ok: [54.89.220.67]
ok: [13.218.200.161]

TASK [Install yum httpd package] ***********************************************************************************************************
ok: [54.89.220.67]
ok: [13.218.200.161]

TASK [Copy CSCC1030 repo zip to remote server] *********************************************************************************************
changed: [54.89.220.67]
changed: [13.218.200.161]

TASK [Unarchive CSCC1030 repo] *************************************************************************************************************
changed: [13.218.200.161]
changed: [54.89.220.67]

TASK [Copy Cloud Computing website files to /var/www/html/] ********************************************************************************
changed: [54.89.220.67]
changed: [13.218.200.161]

TASK [Start and Enable httpd] **************************************************************************************************************
changed: [54.89.220.67]
changed: [13.218.200.161]

PLAY RECAP *********************************************************************************************************************************
13.218.200.161             : ok=6    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
54.89.220.67               : ok=6    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[ec2-user@ip-172-31-89-167 CloudOps_Module4_Class6_Repo4_ControlNodeFiles]$
```

Browser verification of Cloud web servers



V. Final Confirmation

- Terraform successfully provisioned all required infrastructure components.
- Ansible playbooks were executed correctly, configuring the web servers as intended.
- The hosted web pages loaded successfully in browser tests.

VI. Conclusion

This lab demonstrated the seamless integration of Terraform and Ansible in deploying and managing AWS infrastructure. By automating provisioning and configuration, the workflow ensured efficiency and accuracy in setting up the control node and web servers. The results validate the correctness of the implementation, as evidenced by successful server deployment and web page accessibility.

VII. References:

https://learn.continue.yorku.ca/pluginfile.php/1142963/mod_lesson/page_contents/133502/CSCL1030%20Class%206%20Infrastructure%20as%20Code%20-%20Configuration%20Management%20Part%202%20.pdf
https://learn.continue.yorku.ca/mod/lesson/view.php?id=603346&pageid=133502

Thank you.