



Get Certified. Get Ahead.

PG DO - DEVOPS CERTIFICATION TRAINING

AUTOMATING INFRASTRUCTURE USING TERRAFORM

Git Repository:

https://github.com/sravan1990/simplilearn_automating_infrastructure_using_terraform.git

PRESENTED BY: VENKATA SRAVAN KUMAR CHIVUKULA

PG DO - DEVOPS CERTIFICATION TRAINING

PROJECT AGENDA

Use Terraform to provision infrastructure

DESCRIPTION

Nowadays, infrastructure automation is critical. We tend to put the most emphasis on software development processes, but infrastructure deployment strategy is just as important. Infrastructure automation not only aids disaster recovery, but it also facilitates testing and development.

Your organization is adopting the DevOps methodology and to automate provisioning of infrastructure there's a need to setup a centralized server for Jenkins.

Terraform is a tool that allows you to provision various infrastructure components. Ansible is a platform for managing configurations and deploying applications. It means you'll use Terraform to build a virtual machine, for example, and then use Ansible to install the necessary applications on that machine.

Considering the Organizational requirement, you are asked to automate the infrastructure using Terraform first and install other required automation tools in it.

Tools required: Terraform,
AWS account with security credentials,
Keypair

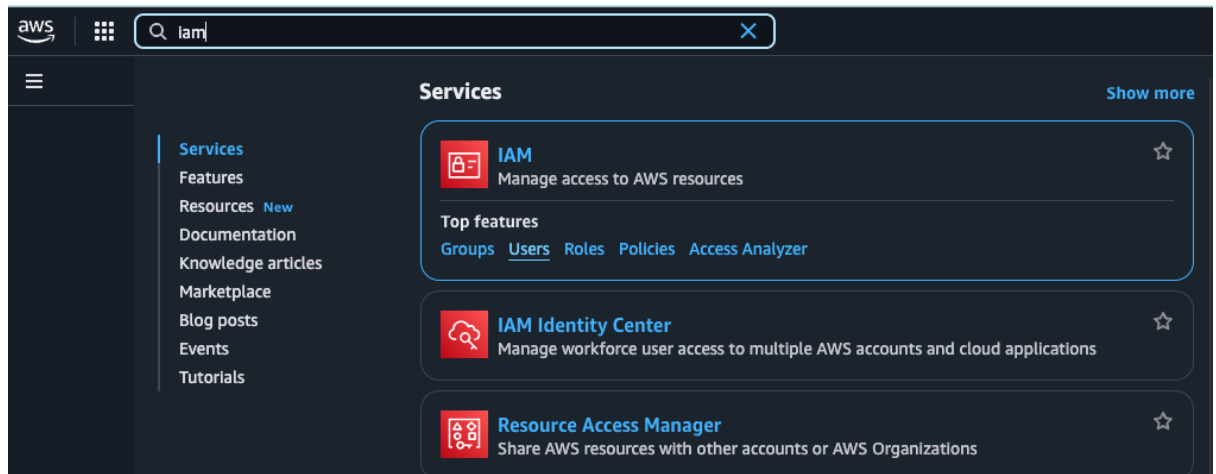
Expected Deliverables: Launch an EC2 instance using Terraform
Connect to the instance
Install Jenkins, Java and Python on the instance

EXECUTION STEPS

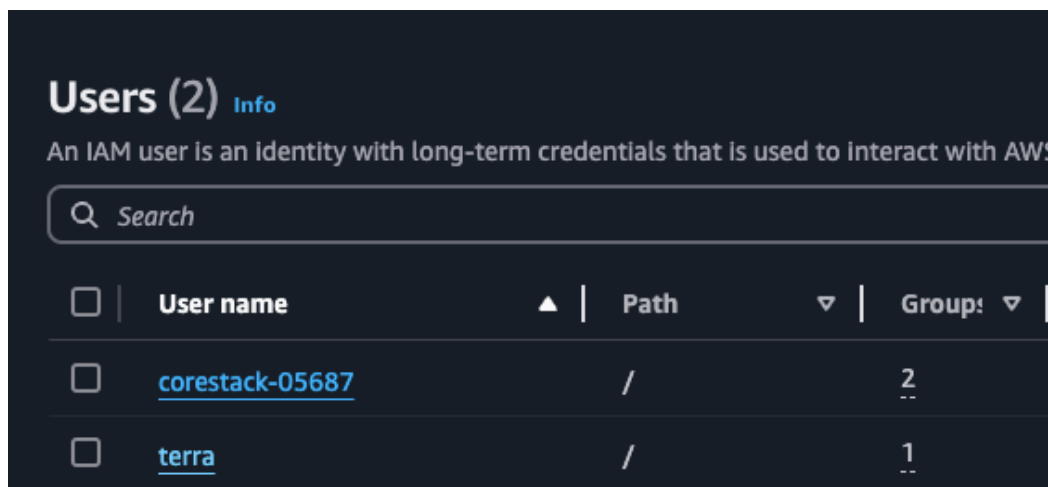
1. Creation of Access keys on AWS

To start the provisioning of the EC2 instance via terraform we need to establish a connection between terraform and AWS, this can be done by creating the access key and secret on AWS.

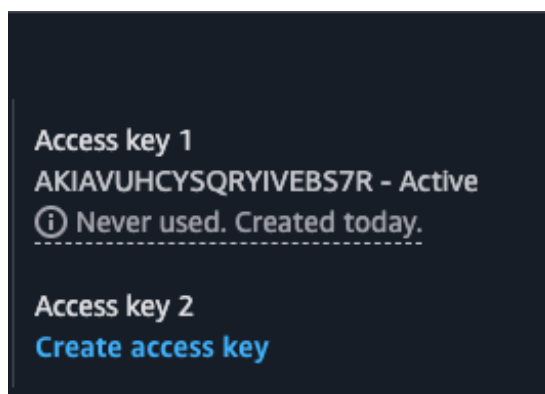
Login to the AWS dashboard, type **IAM** in the global search bar on the top left corner and select **Users** option under the IAM services when you do a mouseover.



On the resultant page you will see a list of all users. Select the user you would like to use



Then click on the Create Access Key on the next page



Once completed you will have the access key and secret to access AWS.

2. Create the terraform files

Check to see if terraform is installed

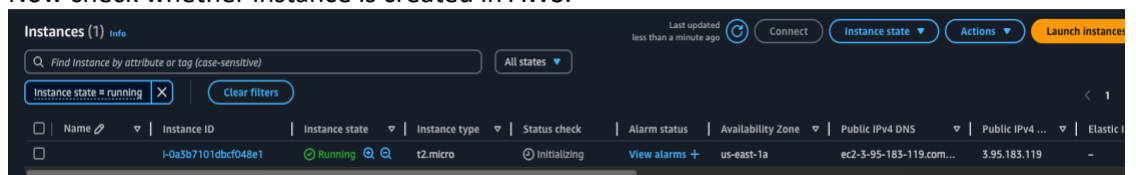
```
● sravankumar@MacBookAir terraform % terraform --version
Terraform v1.10.4
on darwin_arm64
+ provider registry.terraform.io/hashicorp/aws v5.83.0
○ sravankumar@MacBookAir terraform %
```

AWS Configure to save the access keys

```
○ sravankumar@MacBookAir terraform % aws configure
AWS Access Key ID [None]: AKIAVUHCYSQRYIVEBS7R
AWS Secret Access Key [None]: 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
Default region name [None]: US-east-1
Default output format [None]:
```

3. Launch an EC2 instance using Terraform

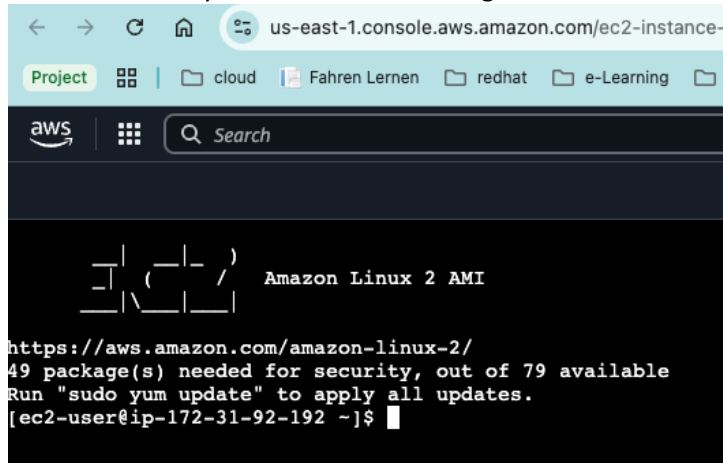
1. Create a **provider.tf** file. (Check the repository)
2. Create a **ec2.tf** file. (Check the repository)
3. initialize terraform using command :
\$ terraform init
4. Run plan :
\$ terraform plan
5. run apply :
\$ terraform apply
6. Now check whether instance is created in AWS.



| Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability Zone | Public IPv4 DNS | Public IPv4 ... | Elastic I |
|------|---------------------|----------------|---------------|--------------|---------------|-------------------|-------------------------|-----------------|-----------|
| | i-0a3b7101dbcf048e1 | Running | t2.micro | Initializing | View alarms + | us-east-1a | ec2-3-95-183-119.com... | 3.95.183.119 | - |

4. Connect to the instance

Connect to newly create instance using SSH.

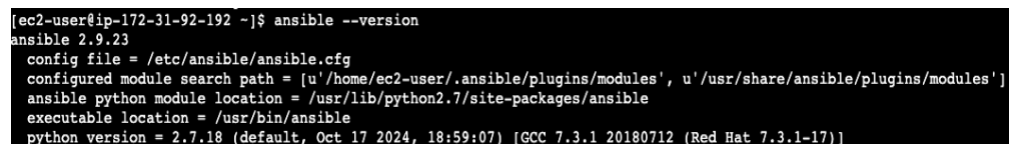


```
us-east-1.console.aws.amazon.com/ec2-instance-...  
Project | cloud | Fahren Lernen | redhat | e-Learning |  
aws | Search  
Amazon Linux 2 AMI  
https://aws.amazon.com/amazon-linux-2/  
49 package(s) needed for security, out of 79 available  
Run "sudo yum update" to apply all updates.  
[ec2-user@ip-172-31-92-192 ~]$
```

5. Install ansible on the instance

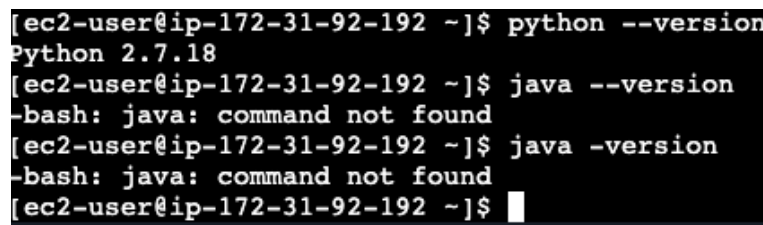
1. Install Ansible on the server

```
$ sudo yum update  
$ sudo amazon-linux-extras install ansible2
```



```
[ec2-user@ip-172-31-92-192 ~]$ ansible --version  
ansible 2.9.23  
config file = /etc/ansible/ansible.cfg  
configured module search path = [u'/home/ec2-user/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']  
ansible python module location = /usr/lib/python2.7/site-packages/ansible  
executable location = /usr/bin/ansible  
python version = 2.7.18 (default, Oct 17 2024, 18:59:07) [GCC 7.3.1 20180712 (Red Hat 7.3.1-17)]
```

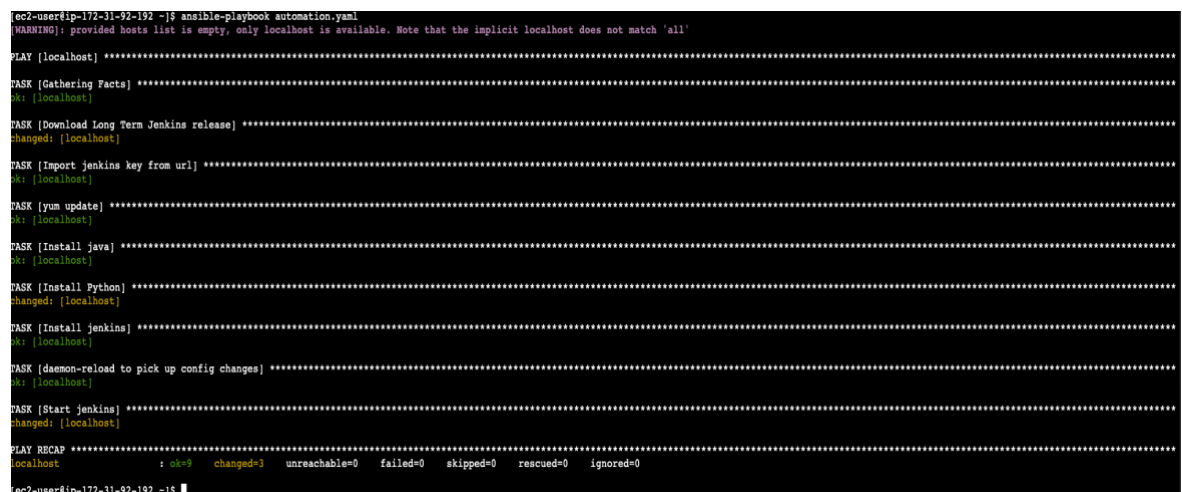
6. Create and Run an ansible playbook **automation.yaml** to install Java Python and Jenkins before running playbook.



```
[ec2-user@ip-172-31-92-192 ~]$ python --version  
Python 2.7.18  
[ec2-user@ip-172-31-92-192 ~]$ java --version  
-bash: java: command not found  
[ec2-user@ip-172-31-92-192 ~]$ java -version  
-bash: java: command not found  
[ec2-user@ip-172-31-92-192 ~]$
```

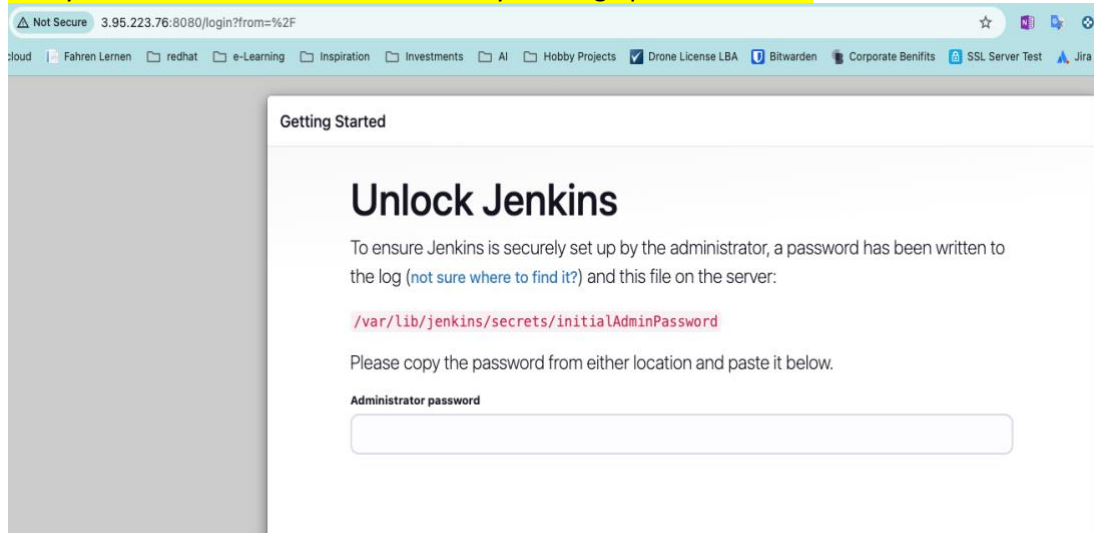
Run ansible playbook

```
$ ansible-playbook automation.yaml
```



```
[ec2-user@ip-172-31-92-192 ~]$ ansible-playbook automation.yaml  
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'  
PLAY [localhost] *****  
TASK [Gathering Facts] *****  
ok: [localhost]  
TASK [Download Long Term Jenkins release] *****  
changed: [localhost]  
TASK [Import jenkins key from url] *****  
ok: [localhost]  
TASK [yum update] *****  
ok: [localhost]  
TASK [Install java] *****  
ok: [localhost]  
TASK [Install Python] *****  
changed: [localhost]  
TASK [Install jenkins] *****  
ok: [localhost]  
TASK [daemon-reload to pick up config changes] *****  
ok: [localhost]  
TASK [Start jenkins] *****  
changed: [localhost]  
PLAY RECAP *****  
localhost : ok=9 changed=3 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0  
[ec2-user@ip-172-31-92-192 ~]$
```

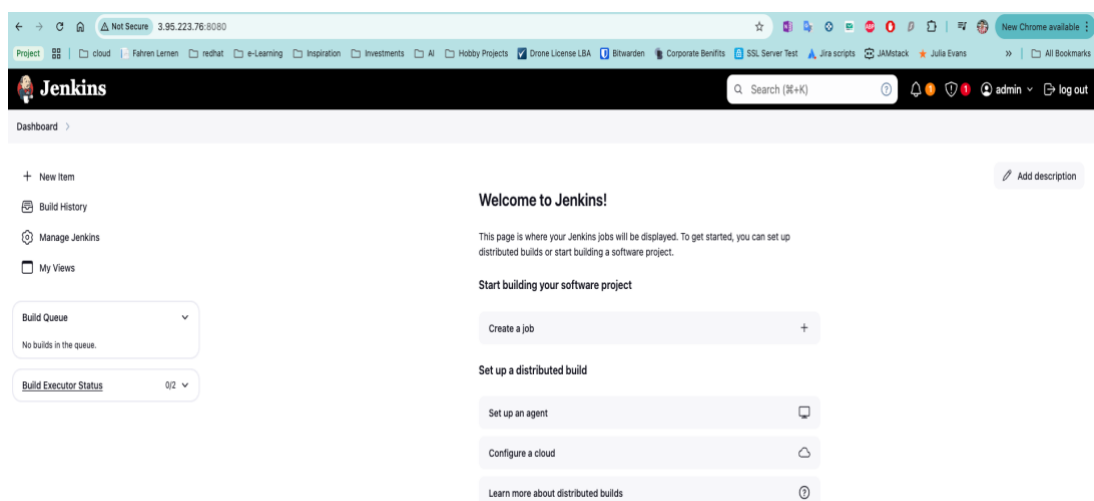
7. Verify whether Jenkins is installed or not by running `<public IP>:8080`



Jenkins Status

```
[ec2-user@ip-172-31-92-192 ~]$ systemctl status jenkins.service
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2025-01-10 01:51:58 UTC; 10min ago
     Main PID: 13487 (java)
    CGroup: /system.slice/jenkins.service
            └─13487 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins/war --httpPort=8080
```

Finally jenkins dashboard after installation was successful !!



1. Verify whether java installed or not by running:

`$ java -version`

```
[ec2-user@ip-172-31-92-192 ~]$ java -version
openjdk version "17.0.13" 2024-10-15 LTS
OpenJDK Runtime Environment Corretto-17.0.13.11.1 (build 17.0.13+11-LTS)
OpenJDK 64-Bit Server VM Corretto-17.0.13.11.1 (build 17.0.13+11-LTS, mixed mode, sharing)
[ec2-user@ip-172-31-92-192 ~]$
```

2. Verify whether Python installed or not by running:

`$ python --version`

```
[ec2-user@ip-172-31-92-192 ~]$ python --version
Python 2.7.18
[ec2-user@ip-172-31-92-192 ~]$
```

RISKS AND ERRORS DURING EXECUTION

POTENTIAL ERRORS AND PROBLEMS

1. Initially I was planning to install Ansible using **remote-exec** and **connection** block.

But this failed due to a weird issue with the private key. I tried couple of times but was unable to fix this and finally had to do this manually.

2. Initially I had an issue while starting the Jenkins server post installation.

```
[ec2-user@ip-172-31-92-192 ~]$ ansible-playbook automation.yaml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [localhost] *****************************************************************************************************************************************************************************************************
TASK [Gathering Facts] ****************************************************************************************************************************************************
ok: [localhost]

TASK [Download Long Term Jenkins release] *************************************************************************************
ok: [localhost]

TASK [Import Jenkins key from url] *********************************************************************************************
ok: [localhost]

TASK [yum update] *************************************************************************************************************
ok: [localhost]

TASK [Install Java] *************************************************************************************************************
ok: [localhost]

TASK [Install Python] *************************************************************************************************************
changed: [localhost]

TASK [Install Jenkins] *************************************************************************************************************
ok: [localhost]

TASK [daemon-reload to pick up config changes] ********************************************************************************
ok: [localhost]

TASK [Start Jenkins] *************************************************************************************************************
fatal: [localhost]: FAILED! => ('changed': false, 'msg': 'Unable to start service jenkins: Job for jenkins.service failed because the control process exited with error code. See 'systemctl status jenkins.service' and 'journalctl -b' for details.')

PLAY RECAP *************************************************************************************************************************************************************
localhost                  : ok=9  changed=1  unreachable=0  failed=0   skipped=0   rescued=0   ignored=0

[ec2-user@ip-172-31-92-192 ~]$
```

APPROPRIATE CORRECTIVE MEASURES

- For the issue with Jenkins not starting I reviewed the following
 - Security group : Inbound rules (8080 port should be allowed)
 - Checked the Jenkins configuration : init file to check startup script.
 - Verified the repo and URL used for installation was correct.

The issue was with the repo URL, which was corrected for Jenkins to start successfully.

CONCLUSION / RESULT

Successfully deployed an EC2 Instance on AWS cloud using terraform and installed automation tool Ansible and used ansible playbooks to install Java, Python and Jenkins on the ec2 instance.