“Course-end Project 1 - Automating Infrastructure using Terraform”

Project files in Github:

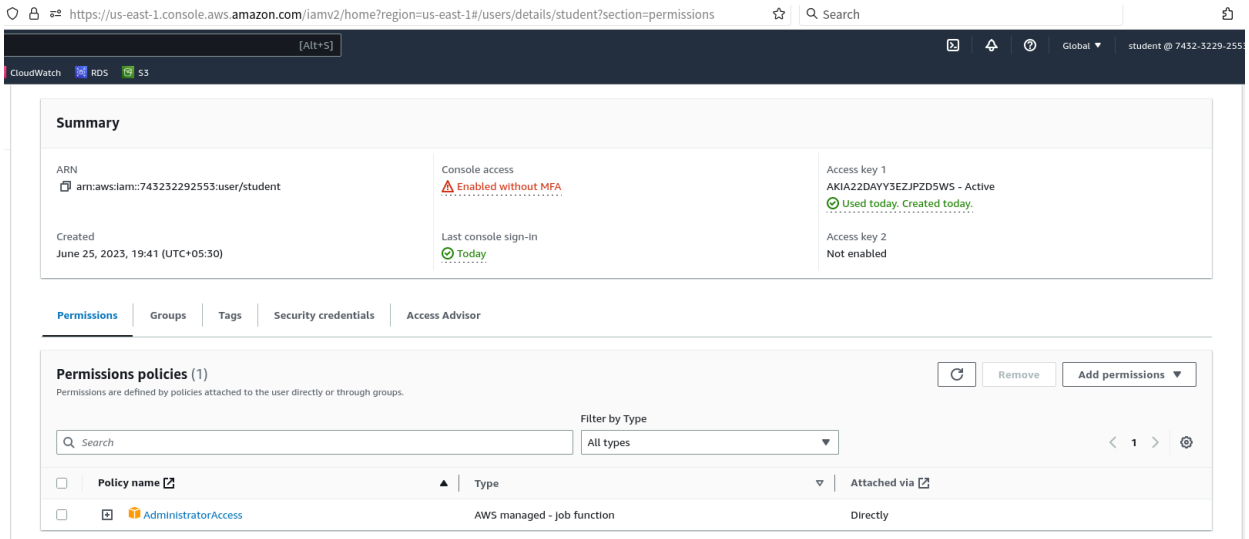
step 1:

Project environment
Base OS(laptop): Linux mint
AWS cli and Terraform installed on my laptop.

```
tridev@SuperiorLynux: ~
File  Edit  View  Search  Terminal  Help
tridev@SuperiorLynux:~$ dpkg -l | grep aws
ii  awscli                                       1.22.34-1
          all          Universal Command Line Environment for AWS
tridev@SuperiorLynux:~$ terraform --version
Terraform v1.5.0
on linux_amd64

Your version of Terraform is out of date! The latest version
is 1.5.1. You can update by downloading from https://www.terraform.io/downloads.
html
tridev@SuperiorLynux:~$
```

step 2:

Log into to AWS account with aws IAM user having full administrator permission (policy: AdministratorAccess applied), make sure the user has AWS access_key_ID and secret_access_key created and downloaded for use.



step 3:

In proj1 folder, create the following files and directory:

<u>files created:</u> init.sh, key.pem, main.tf, variables.tf , terraform.tfvars

<u>directory:</u> ansible which contains jenkins.sh and setup.yaml file.

All these files are to be used by terraform.

init.sh -> user data

key.pem -> aws private key file to create ec2

main.tf -> main file for terraform , which further utilizes variables defined in variables.tf and aws iam user's credential (accesskey,secret_access_key) goes to variables.tfvars

Following are the file contents:

```
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ ls
ansible  challenges-solution.txt  init.sh  key.pem  main.tf  student_user.txt  terraform.tfvars  variables.tf
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ ls ansible/
jenkins.sh  setup.yaml
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ cat init.sh
#!/bin/bash
sudo apt update -y
sudo apt upgrade -y
sudo apt install ansible -y
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ file key.pem
key.pem: PEM RSA private key
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ cat terraform.tfvars
access_key = "AKIA22DAYY3EZJPZD5WS"
secret_key = "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"
```

```
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ cat variables.tf
# https://developer.hashicorp.com/terraform/language/values/variables

variable "access_key" {
  description = "Access key to AWS console"
}
variable "secret_key" {
    description = "Secret key to AWS console"
}

variable "region" {
    description = "The region EC2 instances will be created in"
    default = "us-east-1"
}
variable "instance_name" {
    description = "Name of the instance to be created"
    default = "proj-ec2"
}

variable "instance_type" {
    default = "t2.micro"
}

variable "vpc_id" {
    default = "vpc-094935408df5c4b0b"
}

variable "subnet_id" {
    description = "The VPC subnet the instance(s) will use -> us-east-1a"
    default = "subnet-0f15680deb5806193"
}

variable "ami_id" {
    description = "the AMI image to use -> ubuntu 20 lts 64 bit"
    default = "ami-0261755bbcb8c4a84"
}

variable "number_of_instances" {
  description = "the number of instances to create"
  default = 1
}

variable "ami_key_pair_name" {
    description = "The key pair(.pem file) the ec2 instance is going to use"
    default = "key"  # warning/error note: DO NOT use .pem/extension
}
```

and $ cat main.tf

```
provider "aws" {
  access_key = "${var.access_key}"
  secret_key = "${var.secret_key}"
  region = "${var.region}"
}

resource "aws_security_group" "security_groups" {
  description = "Allow 8080, SSH, icmp traffic - inbound"
  vpc_id = "${var.vpc_id}"

  # inbound rule - allow jenkins traffic
```

```
  ingress {
    from_port   = 8080
    to_port     = 8080
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  # inbound rule - allow ssh traffic
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  # inbound rule - allow icmp/ping traffic
  ingress {
    from_port = -1
    to_port = -1
    protocol = "icmp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  # outbound rule - allow all traffic
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

resource "aws_instance" "main" {
  ami = "${var.ami_id}"
  count = "${var.number_of_instances}"
  #security_groups = "${var.security_groups}"
  vpc_security_group_ids = [aws_security_group.security_groups.id]
  subnet_id = "${var.subnet_id}"
  instance_type = "${var.instance_type}"
  key_name =  "${var.ami_key_pair_name}"

  user_data = "${file("init.sh")}"

  tags = {
```

```
    Name = "${var.instance_name}"
    Team = "Developers"
  }

}
```

inside ansible directory, file contents are:

```
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1/ansible$ cat jenkins.sh
#!/bin/bash
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
apt update -y
apt install jenkins -y
echo "============"
cat /var/lib/jenkins/secrets/initialAdminPassword > /tmp/jenkins-pswd.txt
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1/ansible$ cat setup.yaml
--- # install java and python3
- hosts: node
  become: true
  tasks:
    - name: install java and python3
      apt:
        name: openjdk-11-jdk,openjdk-11-jre,python3
        state: latest
    - name: setup jenkins
      command:
        cmd: bash /home/ubuntu/ansible/jenkins.sh
```

That’s all the project files. Let’s begin!

step 4:

<u>In the proj1 folder (main.tf is here only), run the terraform commands in the following order:</u>

$ terraform init

$ terraform plan

$ terraform apply –auto-approve

terraform init output:

```
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.5.0...
- Installed hashicorp/aws v5.5.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$
```

terraform plan output:

```
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.main[0] will be created
  + resource "aws_instance" "main" {
      + ami                                  = "ami-0261755bbcb8c4a84"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = "key"
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
```

```
    + placement_group                      = (known after apply)
    + placement_partition_number           = (known after apply)
    + primary_network_interface_id         = (known after apply)
    + private_dns                          = (known after apply)
    + private_ip                           = (known after apply)
    + public_dns                           = (known after apply)
    + public_ip                            = (known after apply)
    + secondary_private_ips                = (known after apply)
    + security_groups                      = (known after apply)
    + source_dest_check                    = true
    + spot_instance_request_id             = (known after apply)
    + subnet_id                            = "subnet-0f15680deb5806193"
    + tags                                 = {
        + "Name" = "proj-ec2"
        + "Team" = "Developers"
      }
    + tags_all                             = {
        + "Name" = "proj-ec2"
        + "Team" = "Developers"
      }
    + tenancy                              = (known after apply)
    + user_data                            = "938e29a3976ca4a98203a7373282e5206deac245"
    + user_data_base64                     = (known after apply)
    + user_data_replace_on_change          = false
    + vpc_security_group_ids               = (known after apply)
  }

# aws_security_group.security_groups will be created
+ resource "aws_security_group" "security_groups" {
    + arn                    = (known after apply)
    + description            = "Allow 8080, SSH, icmp traffic - inbound"
    + egress                 = [
```

```
+ egress                    = [
    + {
        + cidr_blocks      = [
            + "0.0.0.0/0",
          ]
        + description      = ""
        + from_port        = 0
        + ipv6_cidr_blocks = []
        + prefix_list_ids  = []
        + protocol         = "-1"
        + security_groups  = []
        + self             = false
        + to_port          = 0
      },
  ]
+ id                       = (known after apply)
+ ingress                  = [
    + {
        + cidr_blocks      = [
            + "0.0.0.0/0",
          ]
        + description      = ""
        + from_port        = -1
        + ipv6_cidr_blocks = []
        + prefix_list_ids  = []
        + protocol         = "icmp"
        + security_groups  = []
        + self             = false
        + to_port          = -1
      },
    + {
        + cidr_blocks      = [
            + "0.0.0.0/0",
```

```
            + cidr_blocks      = [
                + "0.0.0.0/0",
              ]
            + description      = ""
            + from_port        = 22
            + ipv6_cidr_blocks = []
            + prefix_list_ids  = []
            + protocol         = "tcp"
            + security_groups  = []
            + self             = false
            + to_port          = 22
          },
        + {
            + cidr_blocks      = [
                + "0.0.0.0/0",
              ]
            + description      = ""
            + from_port        = 8080
            + ipv6_cidr_blocks = []
            + prefix_list_ids  = []
            + protocol         = "tcp"
            + security_groups  = []
            + self             = false
            + to_port          = 8080
          },
      ]
    + name                   = (known after apply)
    + name_prefix            = (known after apply)
    + owner_id               = (known after apply)
    + revoke_rules_on_delete = false
    + tags_all               = (known after apply)
    + vpc_id                 = "vpc-094935408df5c4b0b"
  }

Plan: 2 to add, 0 to change, 0 to destroy.


Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run
"terraform apply" now.
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$
```

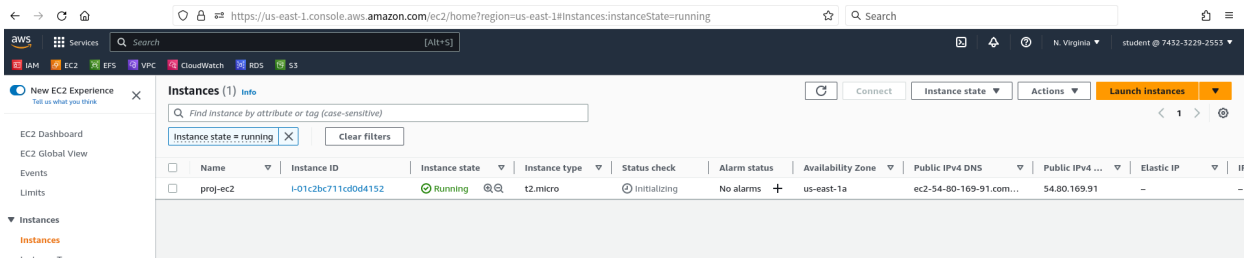
terraform apply –auto-approve output:

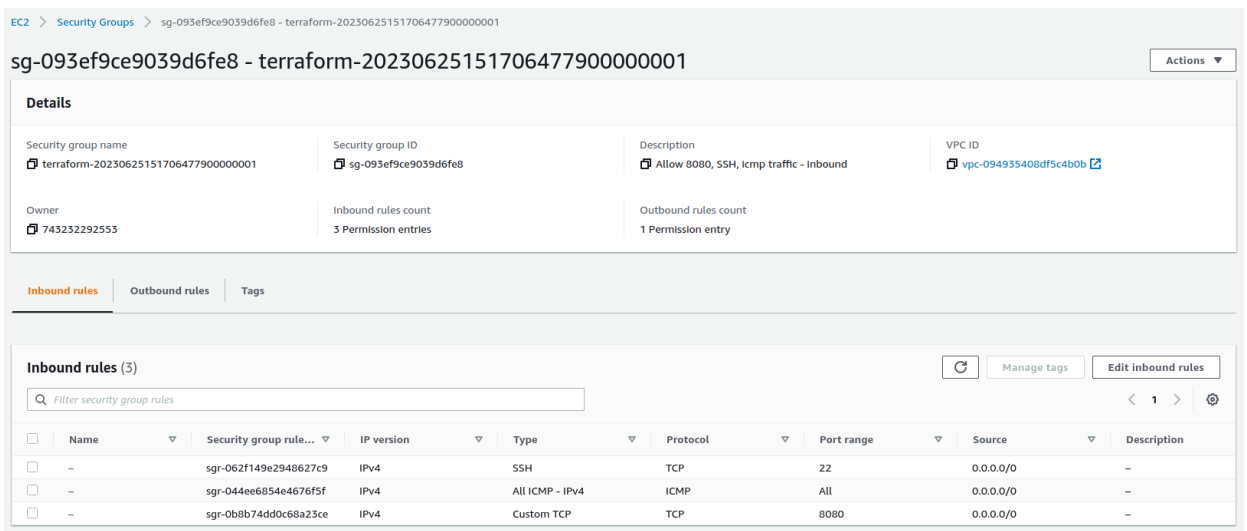
```
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ terraform apply --auto-approve

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.main[0] will be created
  + resource "aws_instance" "main" {
      + ami                                  = "ami-0261755bbcb8c4a84"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = "key"
```

Note: the output is so long, I am skipping to the end of the output, where it shows success or failure. Kindly bear.

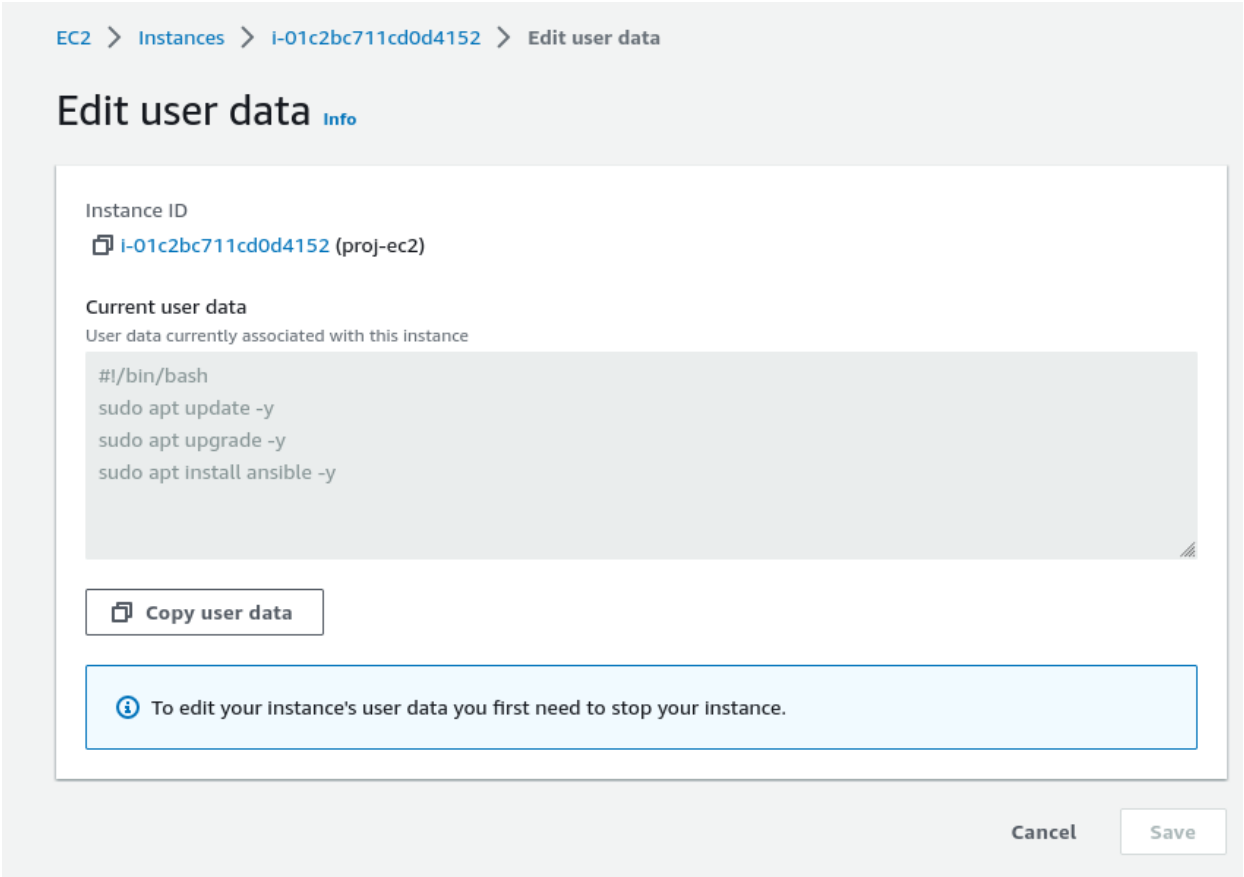
```
      + vpc_id                = "vpc-094935408df5c4b0b"
    }

Plan: 2 to add, 0 to change, 0 to destroy.
aws_security_group.security_groups: Creating...
aws_security_group.security_groups: Creation complete after 5s [id=sg-093ef9ce9039d6fe8]
aws_instance.main[0]: Creating...
aws_instance.main[0]: Still creating... [10s elapsed]
aws_instance.main[0]: Still creating... [20s elapsed]
aws_instance.main[0]: Still creating... [30s elapsed]
aws_instance.main[0]: Creation complete after 35s [id=i-01c2bc711cd0d4152]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$
```

As the result of terraform apply the following ec2 instance and security groupon AWS is created by terraform:





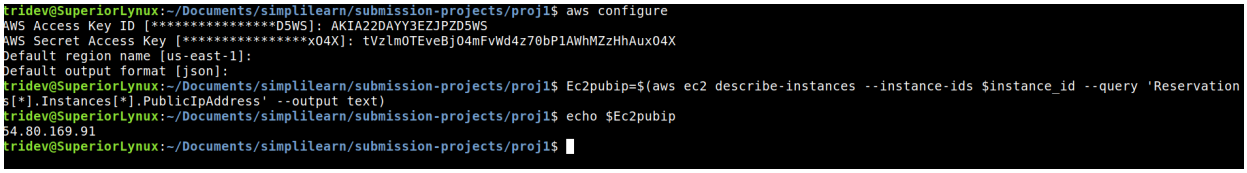
the userdata script installed ansible in our ec2 instance:



step 5:

Now that Terraform's part is over, the next part is ansible's work:

First we need to configure aws in order to grab the created EC2's public IP address:
run:
$ aws configure
$ Ec2pubip=$(aws ec2 describe-instances --instance-ids $instance_id --query 'Reservations[*].Instances[*].PublicIpAddress' --output text)
this stores ec2 public ip in Ec2pubip variable for further use.



step 6:

Let’s ssh to the the ec2 instance and get ansible pre-setup ready:
run:
$ ssh -i key.pem -o "StrictHostKeyChecking no" ubuntu@$Ec2pubip

```
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ ssh -i key.pem -o "StrictHostKeyChecking no" ubuntu@$Ec2pubip
Warning: Permanently added '54.80.169.91' (ED25519) to the list of known hosts.
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-1036-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sun Jun 25 15:27:45 UTC 2023

  System load:  0.0               Processes:             101
  Usage of /:   29.8% of 7.57GB   Users logged in:       0
  Memory usage: 27%               IPv4 address for eth0: 172.31.36.235
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


*** System restart required ***

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-36-235:~$
```

Now that we are logged into ec2, time to 1. check if ansible is installed, 2. make entry to /etc/ansible/hosts and 3. create ssh-keys without prompt and copy the id_rsa.pub to authorized_keys for ansible to use.
run:
$ which ansible
$ echo $'[node]\nlocalhost' | sudo tee -a /etc/ansible/hosts >> /dev/null
$ ssh-keygen -t rsa -N "" -f ~/.ssh/id_rsa
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
output:

```
ubuntu@ip-172-31-36-235:~$ which ansible
/usr/bin/ansible
ubuntu@ip-172-31-36-235:~$ echo $'[node]\nlocalhost' | sudo tee -a /etc/ansible/hosts  >> /dev/null
ubuntu@ip-172-31-36-235:~$ tail -3 /etc/ansible/hosts

[node]
localhost
ubuntu@ip-172-31-36-235:~$ ssh-keygen -t rsa -N "" -f ~/.ssh/id_rsa
Generating public/private rsa key pair.
Your identification has been saved in /home/ubuntu/.ssh/id_rsa
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:NzzTeEwc8L31x2MV9GQd4IYBD8IeWAKQCL5JU8guRpo ubuntu@ip-172-31-36-235
The key's randomart image is:
+---[RSA 3072]----+
|=.+o..+o ooo..o+=|
|o=.  ..o. oo+o o+|
|+=    . .  o+o. +|
|Eo+    . . =.  +o|
|oo      S B + .o+|
|         . =  . o|
|                 |
|                 |
|                 |
+----[SHA256]-----+
ubuntu@ip-172-31-36-235:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
ubuntu@ip-172-31-36-235:~$ cat ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCgyAOZhNpFjN+slmW1oFi0dTTKrX1BA5ifKPJNtwlNI4J3L4LVGP46qAHNMGSCGw2+andgQky6YPbU7jQ0fLZyTJcuNv+BKAtgonT0GPeFTdaJyUFLfJVchL
dVwOGNhIoiwD91X2xHCRtdsvwAs8b85a7/xxOt6S8Y2Ig9QwRh+Tmh0eU8R/ub+7gRNwkjbrF9AjEofjywP2wT1N0GtrimFfQbfnrumOtxQJAy3VYXO3Xna13uNcjnjnxZz1zE3UtOPupimmxfO3AoLhOZF6uo
ACWBfGfRlGJUI+BFPSKWGVD/ZiCjMTr5gKIdpCfqC9sy0825E7HgHpZkHwJ4P+Ip key
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCg5SIuXId2LLcQj9xcj2tJ/vM3TapzdtRsJl2jW6Sz9lL1Xi2UyNuJQWxT0a8FhmrtLPiooM0GhJkyU3g0YK/X7DgaWq3eDRc1Uqp7hlP2IBLhT+EO9zQvRP
2u1KIkj6+MpHEPtZPNpCgnnoQD6m3lFEeAajHU4ObozYvp7oYPYjEPd1GhCBfvzZPEL8CaGg3mayxx9H0cg9pErgJlK9DDT00iJQw9yuURbDe7y7NUWUnX5+AbFyqpxy7hB/zSPfAyV28EE7rNSoC1DTnzMRv
0X9DL+xQ1lP0wHe2+URdK7c9lC93lw2o9PgacEqpKlGhahX6/4N2W++l6BZ/ACOFbTi/ZcvF6wGgK56J1I1hJRaibM2BzQ7KBBWB0qOnb9pz9VwqaXt6L0GmYyPiuJhJhsiKwuuU2wXxJB01JI3rjoNaX61Zy/
4mRuUzV1SqzLbc9BHjf7Z3vT6NDC6wLuUO7gdARFtNKsALHdo/02QyiyYZus/UOSq6t7Pugrlf48M= ubuntu@ip-172-31-36-235
ubuntu@ip-172-31-36-235:~$
```

To avoid ansible from asking ssh prompt for first time let’s run the following:
$ ssh -o "StrictHostKeyChecking no" ubuntu@localhost
and test it out with (optional step): $ ansible node -m ping
output:

```
ubuntu@ip-172-31-36-235:~$ ssh -o "StrictHostKeyChecking no" ubuntu@localhost
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-1036-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sun Jun 25 15:33:34 UTC 2023

  System load:  0.0               Processes:             105
  Usage of /:   29.8% of 7.57GB   Users logged in:       1
  Memory usage: 28%               IPv4 address for eth0: 172.31.36.235
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '22.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


*** System restart required ***
Last login: Sun Jun 25 15:27:47 2023 from 223.233.85.230
ubuntu@ip-172-31-36-235:~$ ansible node -m ping
localhost | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ubuntu@ip-172-31-36-235:~$
```

Now let’s exit from the ec2 terminal and get back to our laptop by running $ exit twice.
Ansible pre-setup is complete.

step 7 and Verification:

Time for ansible magic as our final step.
Copy the jenkins.sh and setup.yaml scripts from ansible directory using scp to our EC2 instance.
run:
copies ansible related files: $ scp -i key.pem -r ansible ubuntu@$Ec2pubip:/home/ubuntu
ssh back to the ec2: $ ssh -i key.pem -o "StrictHostKeyChecking no" ubuntu@$Ec2pubip

```
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ scp -i key.pem -r ansible ubuntu@$Ec2pubip:/home/ubuntu
setup.yaml                                                                                    100%  282     1.3KB/s   00:00
jenkins.sh                                                                                    100%  451     2.0KB/s   00:00
tridev@SuperiorLynux:~/Documents/simplilearn/submission-projects/proj1$ ssh -i key.pem -o "StrictHostKeyChecking no" ubuntu@$Ec2pubip
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-1036-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sun Jun 25 15:37:11 UTC 2023

  System load:  0.0               Processes:             101
  Usage of /:   29.8% of 7.57GB   Users logged in:       0
  Memory usage: 30%               IPv4 address for eth0: 172.31.36.235
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '22.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


*** System restart required ***
Last login: Sun Jun 25 15:33:48 2023 from 127.0.0.1
ubuntu@ip-172-31-36-235:~$
```

Go inside the copied ansible folder and finally run the ansible playbook:

$ cd ansible

$ ansible-playbook setup.yaml

and Bingo!

verify with:

$ which java

$ which python3

$ which jenkins

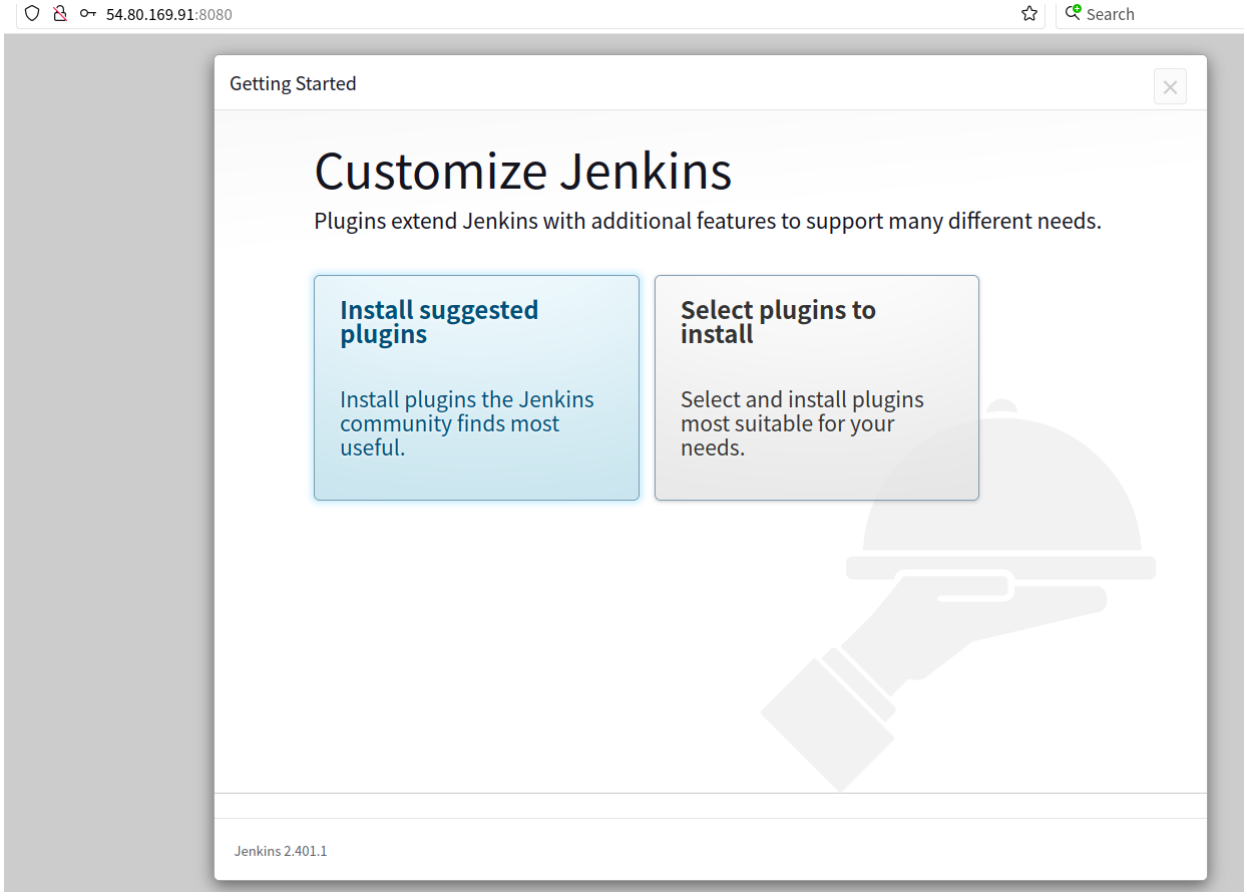
$ java -versions

$ systemctl status jenkins

```
ubuntu@ip-172-31-36-235:~$ cd ansible/
ubuntu@ip-172-31-36-235:~/ansible$ ansible-playbook setup.yaml

PLAY [node] *********************************************************************************************************************************

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

TASK [install java and python3] *************************************************************************************************************
changed: [localhost]

TASK [setup jenkins] ************************************************************************************************************************
changed: [localhost]

PLAY RECAP **********************************************************************************************************************************
localhost                  : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ubuntu@ip-172-31-36-235:~/ansible$ which jenkins
/usr/bin/jenkins
ubuntu@ip-172-31-36-235:~/ansible$ which python3
/usr/bin/python3
ubuntu@ip-172-31-36-235:~/ansible$ which jenkins
/usr/bin/jenkins
ubuntu@ip-172-31-36-235:~/ansible$ systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2023-06-25 15:41:14 UTC; 2min 8s ago
   Main PID: 29950 (java)
      Tasks: 36 (limit: 1141)
     Memory: 281.0M
     CGroup: /system.slice/jenkins.service
             └─29950 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Jun 25 15:40:26 ip-172-31-36-235 jenkins[29950]: 23420b70e025469bb521af9bc6bd7415
Jun 25 15:40:26 ip-172-31-36-235 jenkins[29950]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Jun 25 15:40:26 ip-172-31-36-235 jenkins[29950]: *************************************************************
Jun 25 15:40:26 ip-172-31-36-235 jenkins[29950]: *************************************************************
Jun 25 15:40:26 ip-172-31-36-235 jenkins[29950]: *************************************************************
Jun 25 15:41:14 ip-172-31-36-235 jenkins[29950]: 2023-06-25 15:41:14.353+0000 [id=29]        INFO        jenkins.InitReactorRunner$1#onAttained: Completed in>
Jun 25 15:41:14 ip-172-31-36-235 jenkins[29950]: 2023-06-25 15:41:14.384+0000 [id=22]        INFO        hudson.lifecycle.Lifecycle#onReady: Jenkins is fully>
```

```
ubuntu@ip-172-31-36-235:~/ansible$ java -version
openjdk version "11.0.19" 2023-04-18
OpenJDK Runtime Environment (build 11.0.19+7-post-Ubuntu-0ubuntu120.04.1)
OpenJDK 64-Bit Server VM (build 11.0.19+7-post-Ubuntu-0ubuntu120.04.1, mixed mode, sharing)
ubuntu@ip-172-31-36-235:~/ansible$ cat /tmp/jenkins-pswd.txt
23420b70e025469bb521af9bc6bd7415
ubuntu@ip-172-31-36-235:~/ansible$
```

Let’s verify if jenkins is actually running in web browser:
pasted the jenkins initial admin password from /tmp/jenkins-pswd.txt created by ansible.



verified…Project completed.