Indian Institute of Information Technology Vadodara

DevOps
Capstone Project

Aman Gangwar
202051020
Semester 6

DevOps
Capstone Project

Name - Aman Gangwar
Student ID - 202051020

Instance Configuration

Total Instances - 3

	jenkins-capstone	test-capstone	prod-capstone
Type	t2.medium	t2.micro	t2.micro
OS	Ubuntu 20.04	Ubuntu 20.04	Ubuntu 20.04
CPUs	2	1	1
RAM	4GB	1GB	1GB
Security Group	jenkins-capstone-sg	test-capstone-sg	prod-capstone-sg

Security Groups Configuration

Ingress (Inbound Rules)

Service\SG		jenkins-capstone-sg	test-capstone-sg	prod-capstone-sg
Jenkins	Port	8080	-	-
	Access	Anywhere	-	-
SSH	Port	22	22	22
	Access	My IP	My IP	My IP
HTTP	Port	-	80	80
	Access	-	Anywhere	Anywhere
Everything	Port	-	All	All
	Access	-	jenkins-capstone-sg	jenkins-capstone-sg

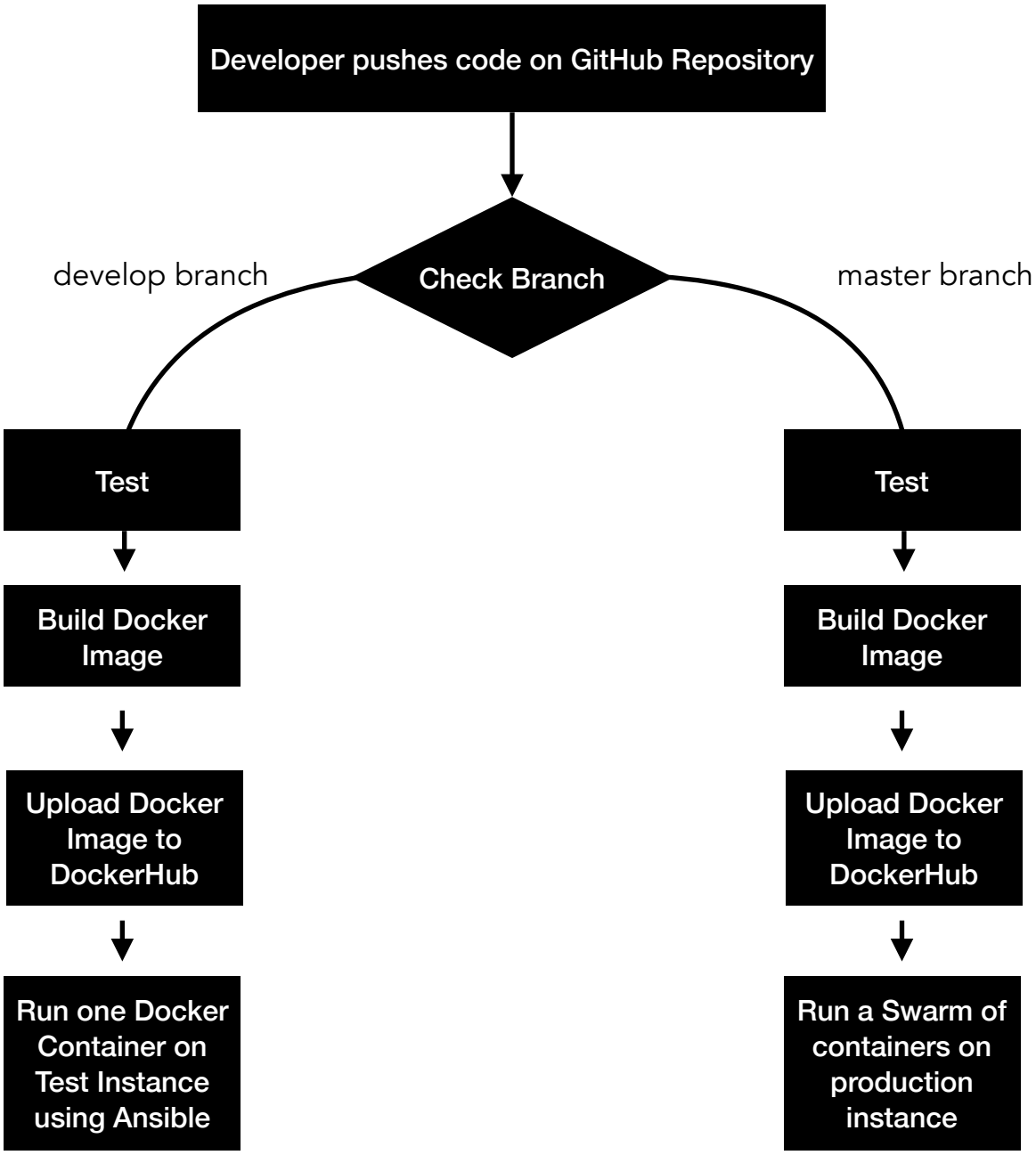
Egress (Outbound Rules)

Service\SG		jenkins-capstone-sg	test-capstone-sg	prod-capstone-sg
Everything	Port	All	All	All
	Access	Anywhere	Anywhere	Anywhere

Tools Used

1. Terraform
2. Git
3. Ansible
4. Jenkins
5. Docker
6. DockerHub
7. Docker Swarm (in place of Kubernetes)

Project Flow



IaC (Infrastructure as Code)

Terraform Script -

main.tf > resource "aws_key_pair" "capstone-key" > public_key

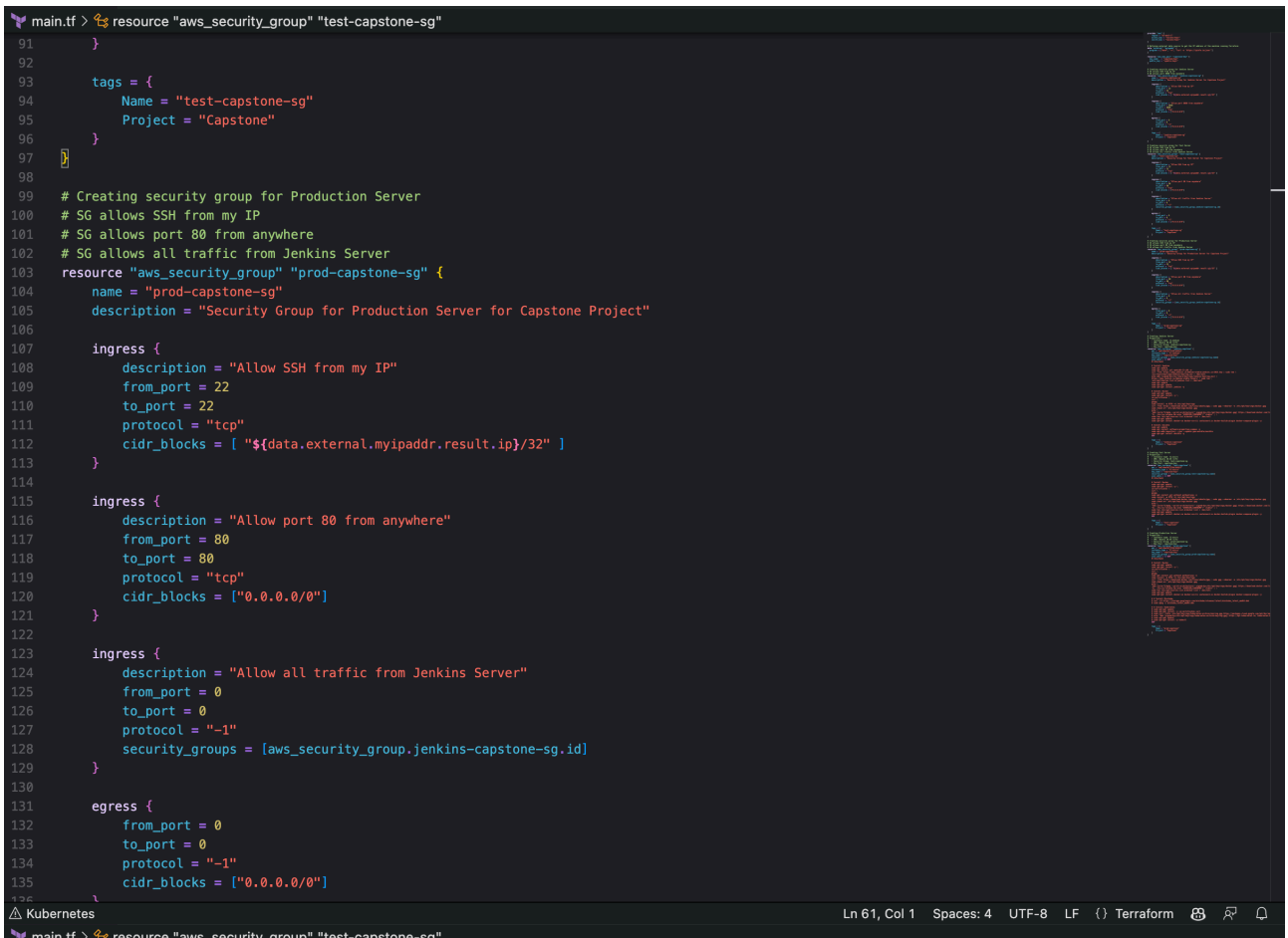
```
provider "aws" {
    region = "us-east-1"
    access_key = "<access-key>"
    secret_key = "<access-key>"
}

# Defining external data source to get the IP address of the machine running Terraform
data "external" "myipaddr" {
  program = ["bash", "-c", "curl -s 'https://ipinfo.io/json'"]
}

resource "aws_key_pair" "capstone-key" {
  key_name   = "capstone-key"
  public_key = "<public-key>"
}


# Creating security group for Jenkins Server
# SG allows SSH from my IP
# SG allows port 8080 from anywhere
resource "aws_security_group" "jenkins-capstone-sg" {
    name = "jenkins-capstone-sg"
    description = "Security Group for Jenkins Server for Capstone Project"

    ingress {
        description = "Allow SSH from my IP"
        from_port = 22
        to_port = 22
        protocol = "tcp"
        cidr_blocks = [ "${data.external.myipaddr.result.ip}/32" ]
    }

    ingress {
        description = "Allow port 8080 from anywhere"
        from_port = 8080
        to_port = 8080
        protocol = "tcp"
        cidr_blocks = ["0.0.0.0/0"]
    }

    egress {
        from_port = 0
        to_port = 0
        protocol = "-1"
        cidr_blocks = ["0.0.0.0/0"]
    }
```

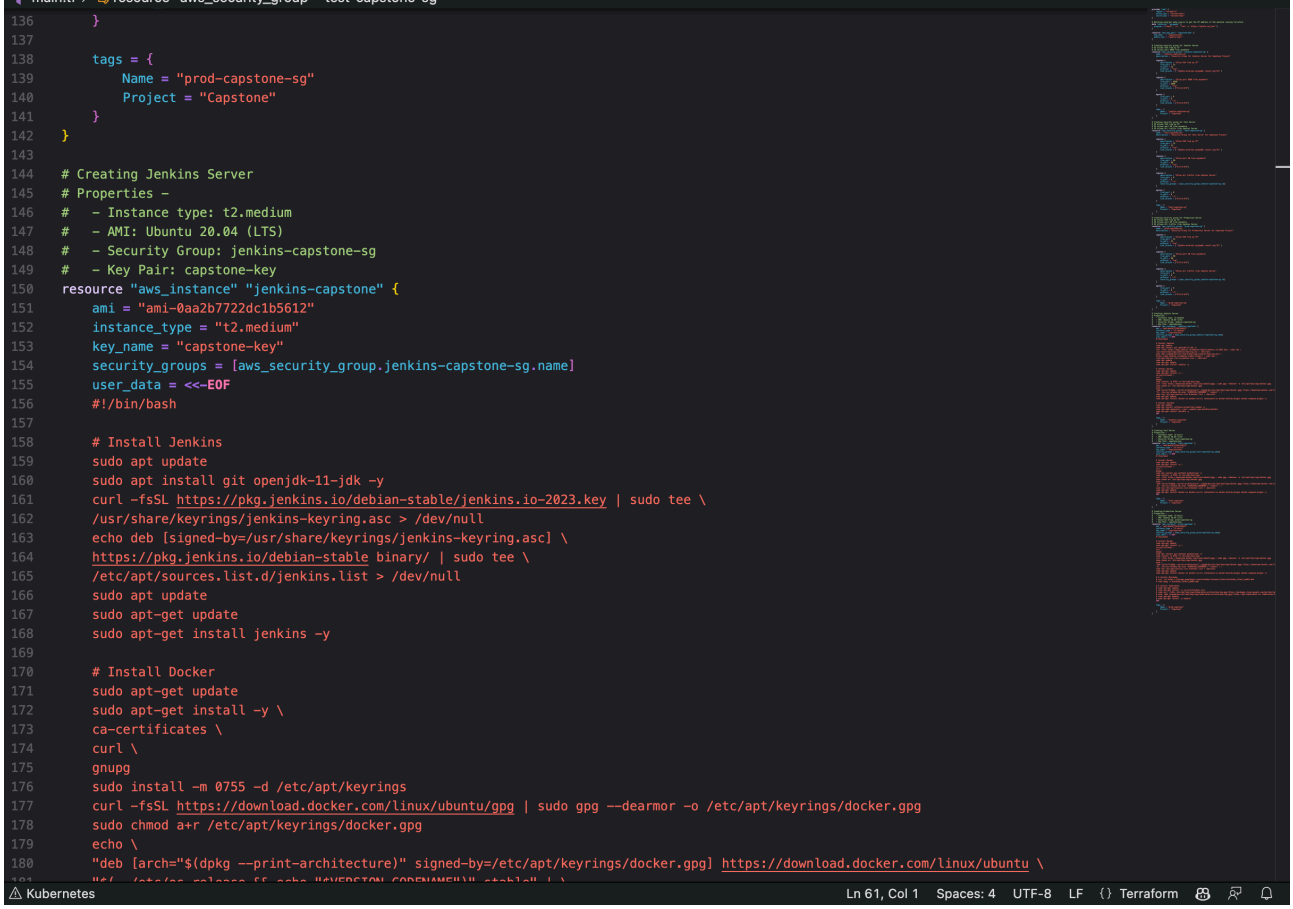
Kubernetes Ln 14, Col 29 Spaces: 4 UTF-8 LF {} Terraform

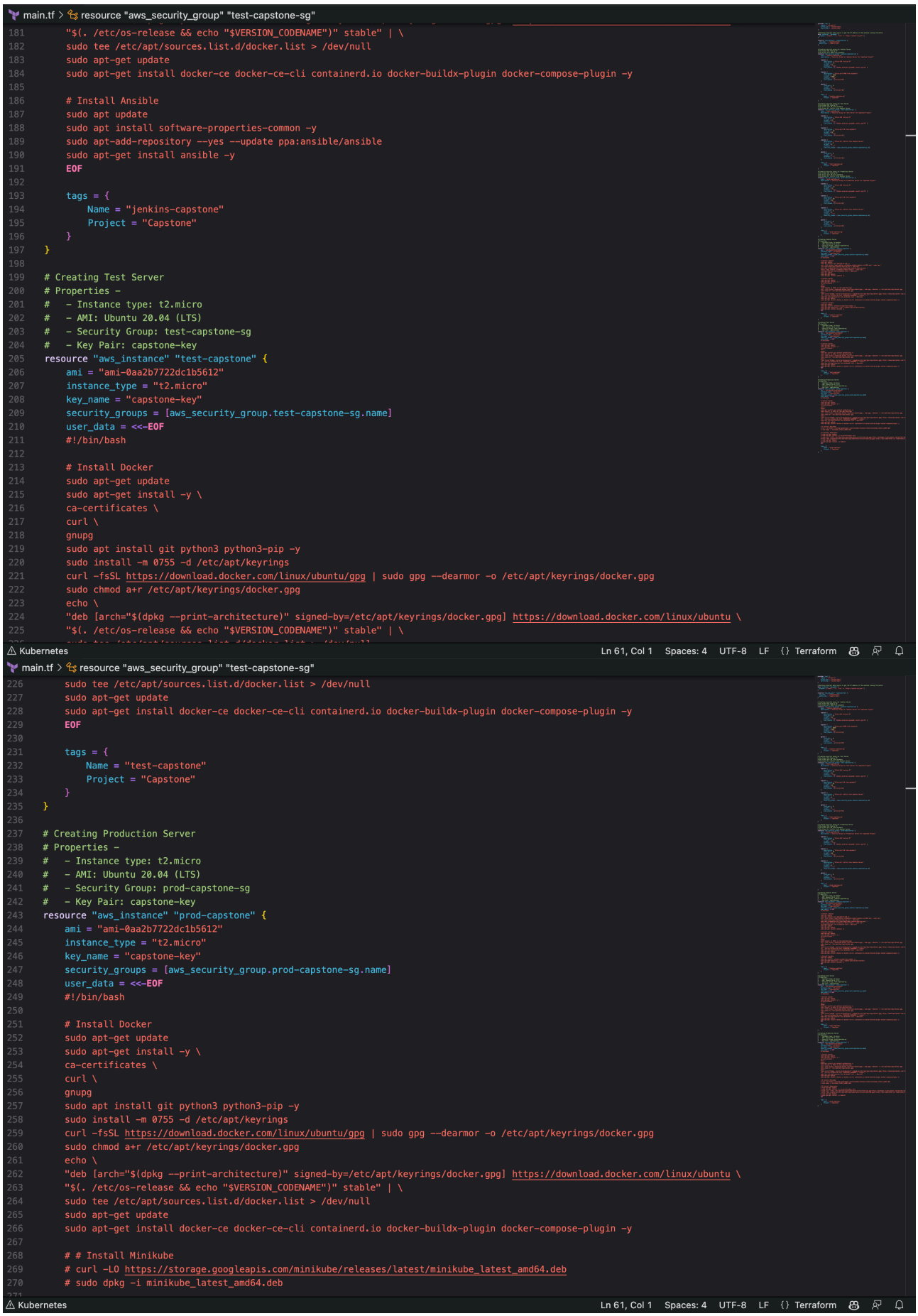
main.tf > resource "aws_security_group" "test-capstone-sg"

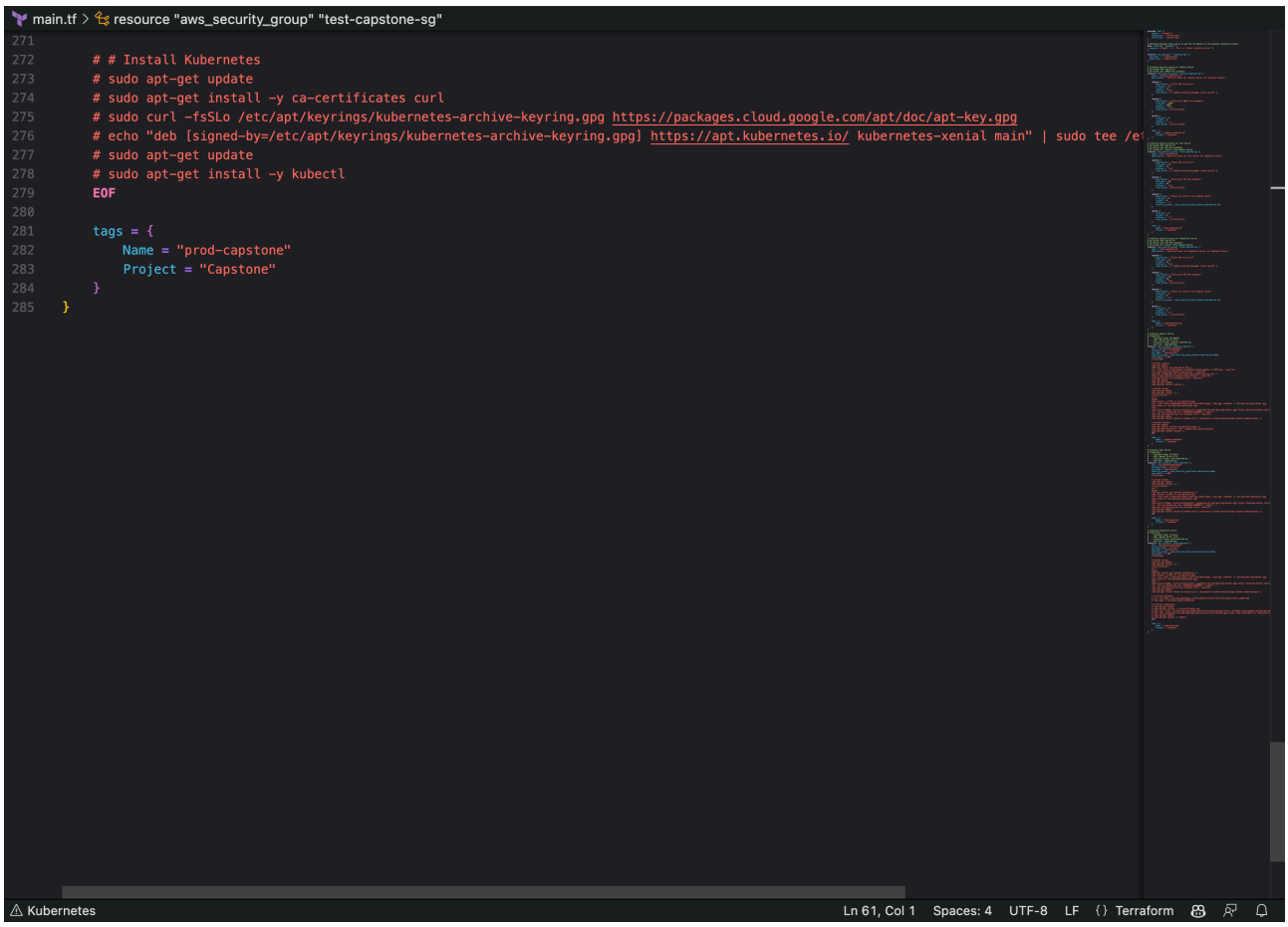
```
    }

    tags = {
        Name = "jenkins-capstone-sg"
        Project = "Capstone"
    }
}

# Creating security group for Test Server
# SG allows SSH from my IP
# SG allows port 80 from anywhere
# SG allows all traffic from Jenkins Server
resource "aws_security_group" "test-capstone-sg" {
    name = "test-capstone-sg"
    description = "Security Group for Test Server for Capstone Project"

    ingress {
        description = "Allow SSH from my IP"
        from_port = 22
        to_port = 22
        protocol = "tcp"
        cidr_blocks = [ "${data.external.myipaddr.result.ip}/32" ]
    }

    ingress {
        description = "Allow port 80 from anywhere"
        from_port = 80
        to_port = 80
        protocol = "tcp"
        cidr_blocks = ["0.0.0.0/0"]
    }

    ingress {
        description = "Allow all traffic from Jenkins Server"
        from_port = 0
        to_port = 0
        protocol = "-1"
        security_groups = [aws_security_group.jenkins-capstone-sg.id]
    }

    egress {
        from_port = 0
        to_port = 0
        protocol = "-1"
        cidr_blocks = ["0.0.0.0/0"]
```

Kubernetes Ln 61, Col 1 Spaces: 4 UTF-8 LF {} Terraform


main.tf > resource "aws_security_group" "test-capstone-sg"
}
tags = {
Name = "test-capstone-sg"
Project = "Capstone"
}
}
# Creating security group for Production Server
# SG allows SSH from my IP
# SG allows port 80 from anywhere
# SG allows all traffic from Jenkins Server
resource "aws_security_group" "prod-capstone-sg" {
name = "prod-capstone-sg"
description = "Security Group for Production Server for Capstone Project"
ingress {
description = "Allow SSH from my IP"
from_port = 22
to_port = 22
protocol = "tcp"
cidr_blocks = ["${data.external.myipaddr.result.ip}/32"]
}
ingress {
description = "Allow port 80 from anywhere"
from_port = 80
to_port = 80
protocol = "tcp"
cidr_blocks = ["0.0.0.0/0"]
}
ingress {
description = "Allow all traffic from Jenkins Server"
from_port = 0
to_port = 0
protocol = "-1"
security_groups = [aws_security_group.jenkins-capstone-sg.id]
}
egress {
from_port = 0
to_port = 0
protocol = "-1"
cidr_blocks = ["0.0.0.0/0"]
Kubernetes
Ln 61, Col 1
Spaces: 4
UTF-8
LF
Terraform


main.tf > resource "aws_security_group" "test-capstone-sg"
}
tags = {
Name = "prod-capstone-sg"
Project = "Capstone"
}
}
# Creating Jenkins Server
# Properties -
# - Instance type: t2.medium
# - AMI: Ubuntu 20.04 (LTS)
# - Security Group: jenkins-capstone-sg
# - Key Pair: capstone-key
resource "aws_instance" "jenkins-capstone" {
ami = "ami-0aa2b7722dc1b5612"
instance_type = "t2.medium"
key_name = "capstone-key"
security_groups = [aws_security_group.jenkins-capstone-sg.name]
user_data = <<-EOF
#!/bin/bash
# Install Jenkins
sudo apt update
sudo apt install git openjdk-11-jdk -y
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
sudo apt update
sudo apt-get update
sudo apt-get install jenkins -y
# Install Docker
sudo apt-get update
sudo apt-get install -y \
ca-certificates \
curl \
gnupg
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
echo \
"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
Kubernetes
Ln 61, Col 1
Spaces: 4
UTF-8
LF
Terraform


main.tf > resource "aws_security_group" "test-capstone-sg"
"$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y
# Install Ansible
sudo apt update
sudo apt install software-properties-common -y
sudo apt-add-repository --yes --update ppa:ansible/ansible
sudo apt-get install ansible -y
EOF
tags = {
Name = "jenkins-capstone"
Project = "Capstone"
}
}
# Creating Test Server
# Properties -
# - Instance type: t2.micro
# - AMI: Ubuntu 20.04 (LTS)
# - Security Group: test-capstone-sg
# - Key Pair: capstone-key
resource "aws_instance" "test-capstone" {
ami = "ami-0aa2b7722dc1b5612"
instance_type = "t2.micro"
key_name = "capstone-key"
security_groups = [aws_security_group.test-capstone-sg.name]
user_data = <<-EOF
#!/bin/bash
# Install Docker
sudo apt-get update
sudo apt-get install -y \
ca-certificates \
curl \
gnupg
sudo apt install git python3 python3-pip -y
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
echo \
"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
"$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
Kubernetes
Ln 61, Col 1
Spaces: 4
UTF-8
LF
Terraform
main.tf > resource "aws_security_group" "test-capstone-sg"
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y
EOF
tags = {
Name = "test-capstone"
Project = "Capstone"
}
}
# Creating Production Server
# Properties -
# - Instance type: t2.micro
# - AMI: Ubuntu 20.04 (LTS)
# - Security Group: prod-capstone-sg
# - Key Pair: capstone-key
resource "aws_instance" "prod-capstone" {
ami = "ami-0aa2b7722dc1b5612"
instance_type = "t2.micro"
key_name = "capstone-key"
security_groups = [aws_security_group.prod-capstone-sg.name]
user_data = <<-EOF
#!/bin/bash
# Install Docker
sudo apt-get update
sudo apt-get install -y \
ca-certificates \
curl \
gnupg
sudo apt install git python3 python3-pip -y
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
echo \
"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
"$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y
# # Install Minikube
# curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube_latest_amd64.deb
# sudo dpkg -i minikube_latest_amd64.deb
Kubernetes
Ln 61, Col 1
Spaces: 4
UTF-8
LF
Terraform



What is on GitHub?

- Code files
- Ansible Playbook named “test-deploy.yml” in ./scripts directory in repository
- Jenkinsfile named “Jenkinsfile” in root of repository
- Dockerfile to build docker image

What is happening?

- Developer pushes the code on a branch
- Jenkins checks which branch the code is pushed to
- Go to that branch
  - Search for a Jenkinsfile in the repository
  - Run the pipeline mentioned in the Jenkinsfile

Jenkinsfiles

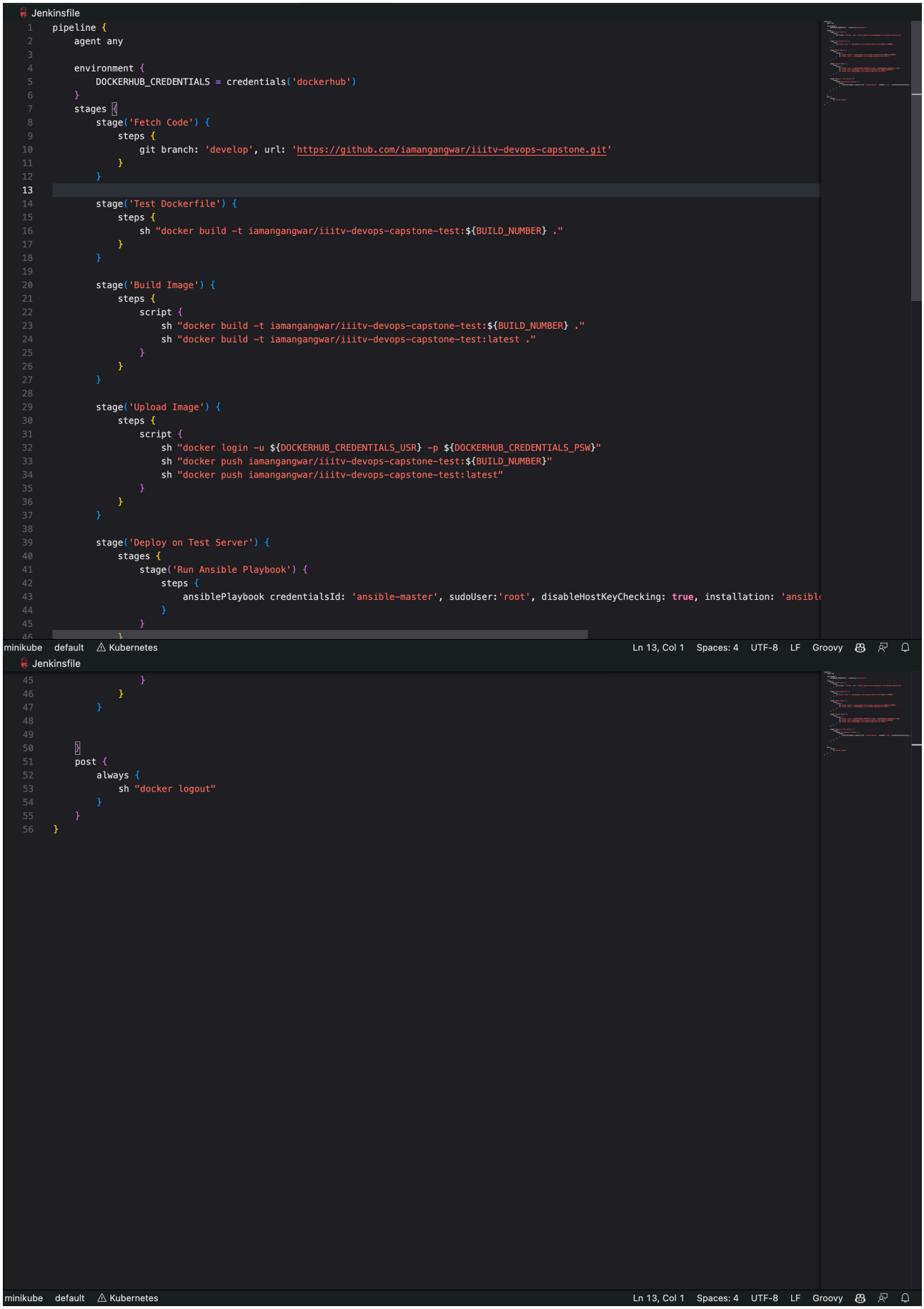
In develop branch

- Fetches the code of develop branch of https://github.com/iamangangwar/iiitv-devops-capstone.git (this repository is forked from the given repository, i.e. https://github.com/hshar/website.git)
- Performs testing
- Builds image using Dockerfile in the repository
  - Images built are
    - iamangangwar/iiitv-devops-capstone-test:${BUILD_NUMBER}
    - iamangangwar/iiitv-devops-capstone-test:latest
- Uploads the images to DockerHub to the corresponding repositories
- Search for ansible playbook named “test-deploy.yml” in scripts directory in repository
  - Ansible playbook does -
    - Remotely pull the image iamangangwar/iiitv-devops-capstone-test:latest from DockerHub
    - Run the container from the pulled image
    - If a container is already running
      - Remove the previous container
      - Run new container

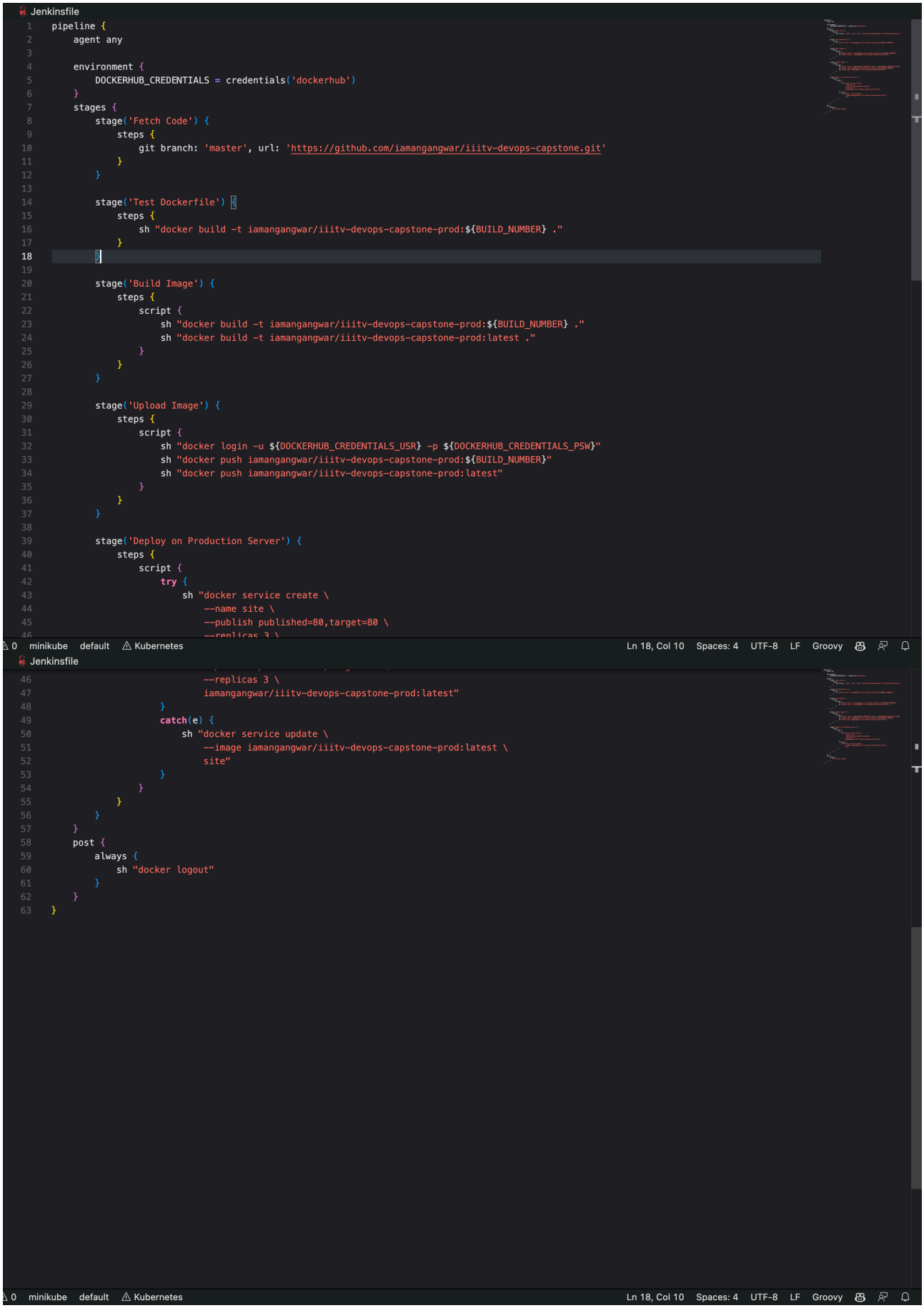
In master branch

- Fetches the code of develop branch of https://github.com/iamangangwar/iiitv-devops-capstone.git (this repository is forked from the given repository, i.e. https://github.com/hshar/website.git)
- Performs testing
- Builds image using Dockerfile in the repository
  - Images built are
    - iamangangwar/iiitv-devops-capstone-prod:${BUILD_NUMBER}
    - iamangangwar/iiitv-devops-capstone-prod:latest
- Uploads the images to DockerHub to the corresponding repositories
- If a swarm already doesn’t exist
  - Create a swarm
- Else
  - Update existing swarm

Jenkinsfile (in develop branch)



Jenkinsfile (in master branch)



Dockerfile

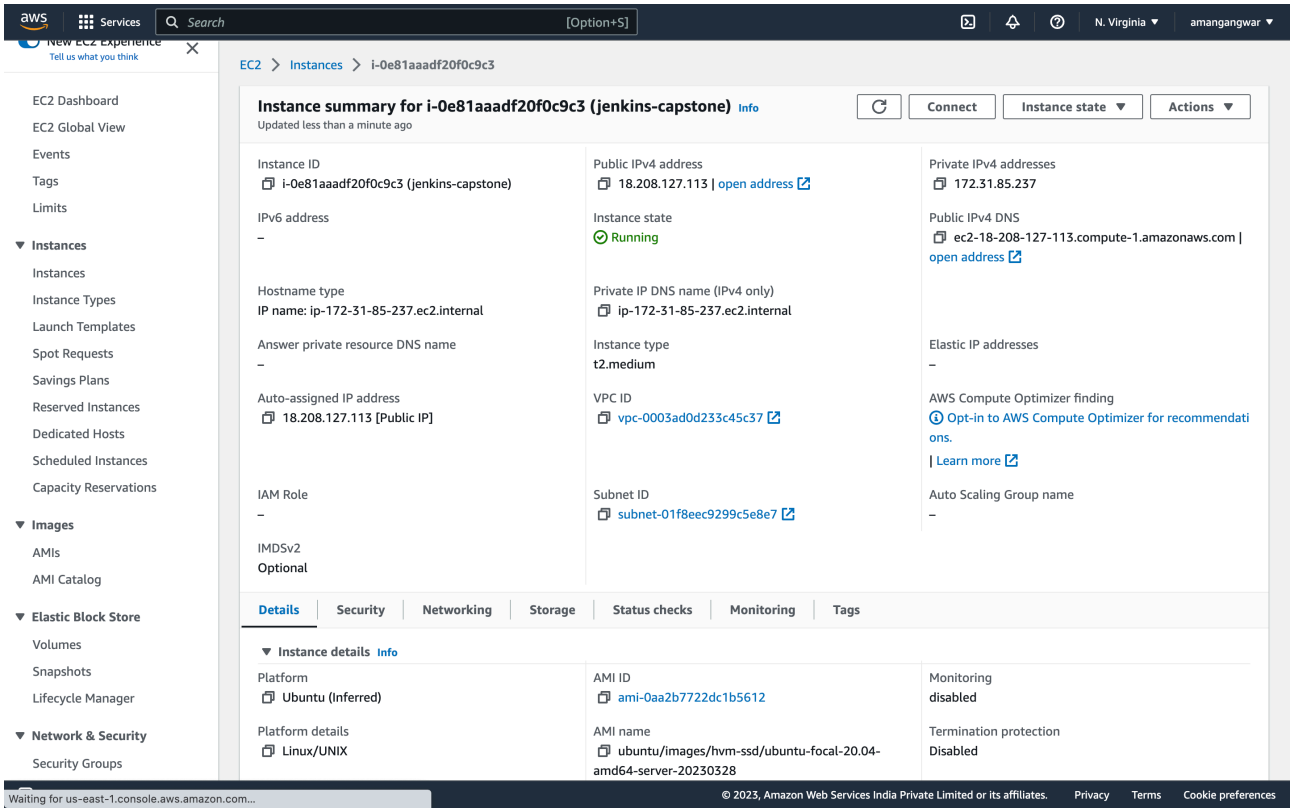
```dockerfile
FROM hshar/webapp

# Set the working directory to /var/www/html
WORKDIR /var/www/html

# Copy the source code into the container
COPY . .

# Expose port 80 for HTTP traffic
EXPOSE 80

# Start Apache web server when the container starts
CMD ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```

Ansible Playbook

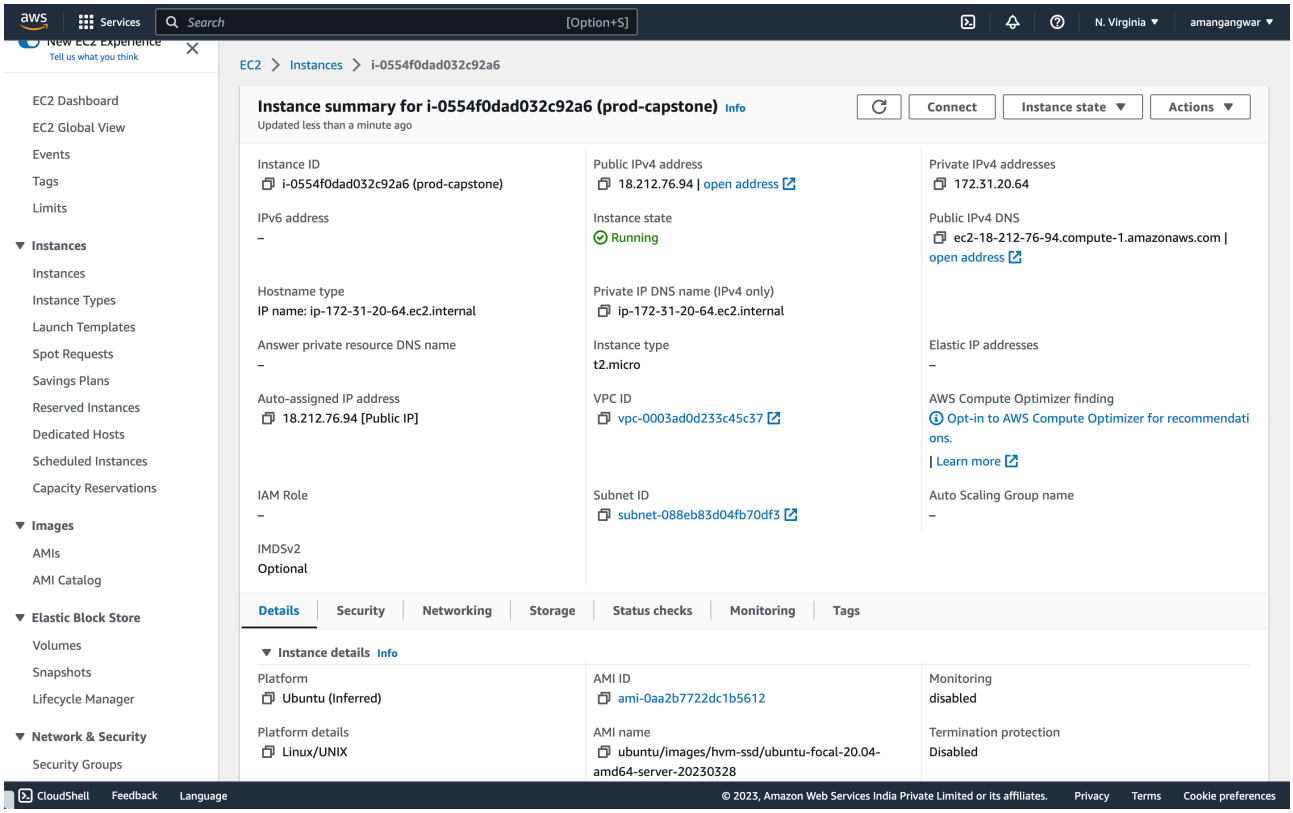
```yaml
- hosts: test
  tasks:
  - name: Pull Docker Image
    docker_image:
      name: iamangangwar/iiitv-devops-capstone-test:latest
      state: present
      source: pull
  - name: Run Container
    block:
    - name: Run Container
      docker_container:
        name: test
        image: iamangangwar/iiitv-devops-capstone-test:latest
        state: started
        ports:
          - "80:80"
    rescue:
    - name: Remove Previous Container
      docker_container:
        name: test
        state: absent
    - name: Run Container
      docker_container:
        name: test
        image: iamangangwar/iiitv-devops-capstone-test:latest
        state: started
        ports:
          - "80:80"
```

Instance Screenshots

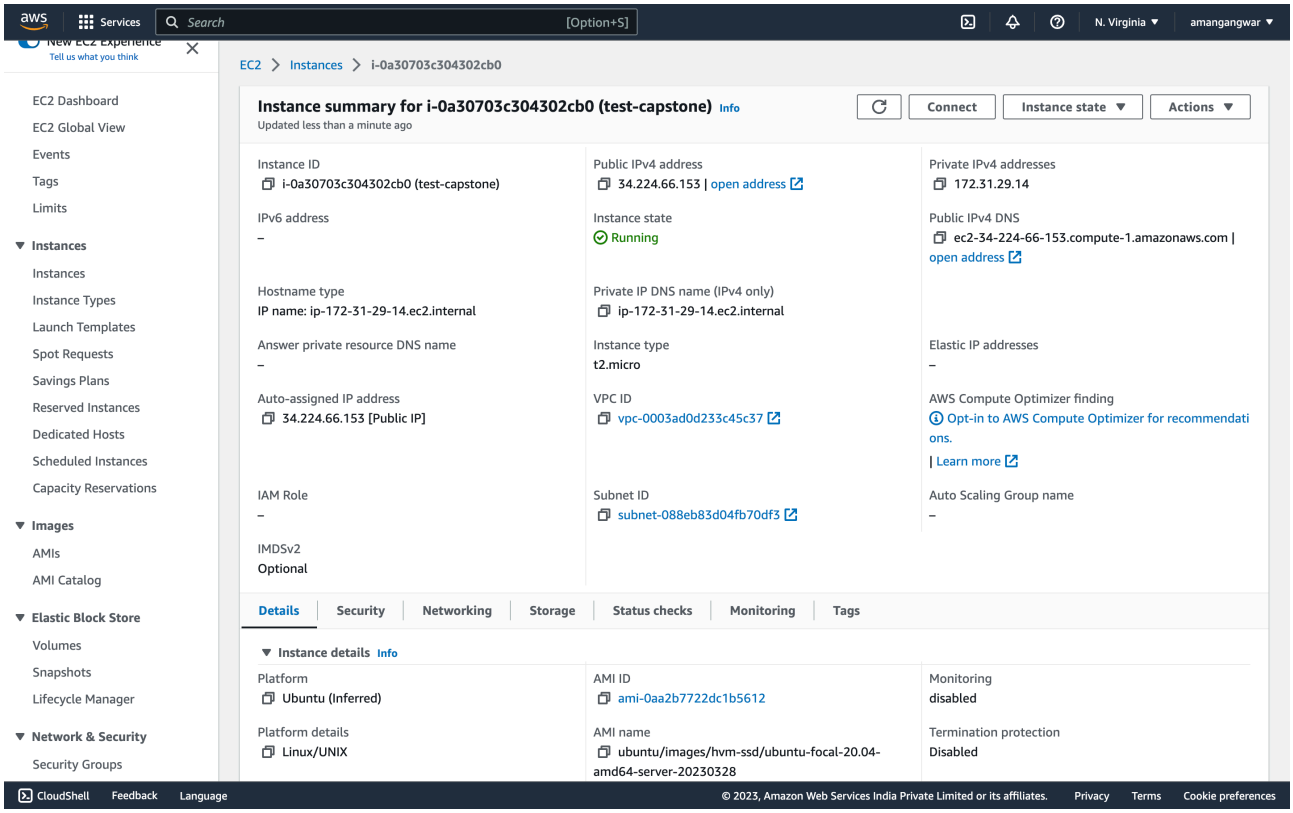
jenkins-capstone



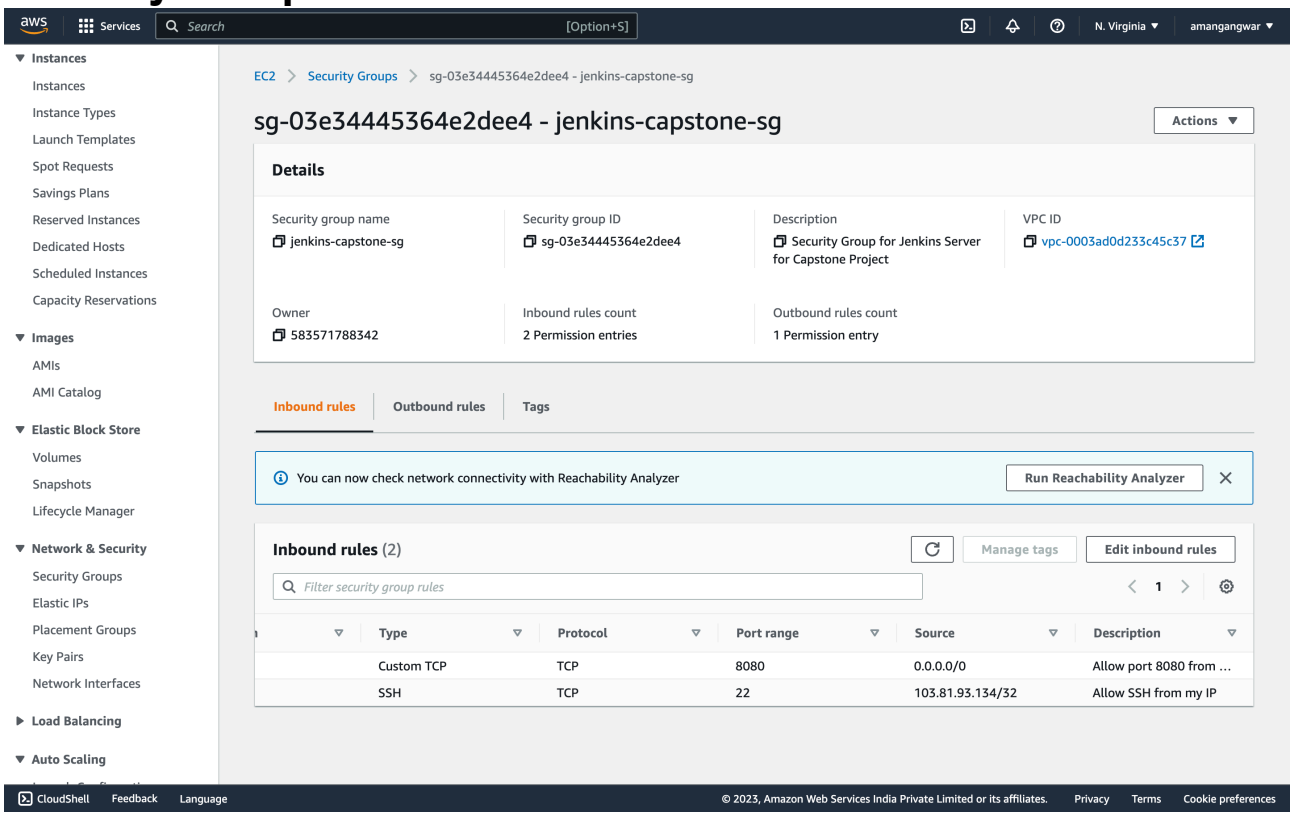
prod-capstone

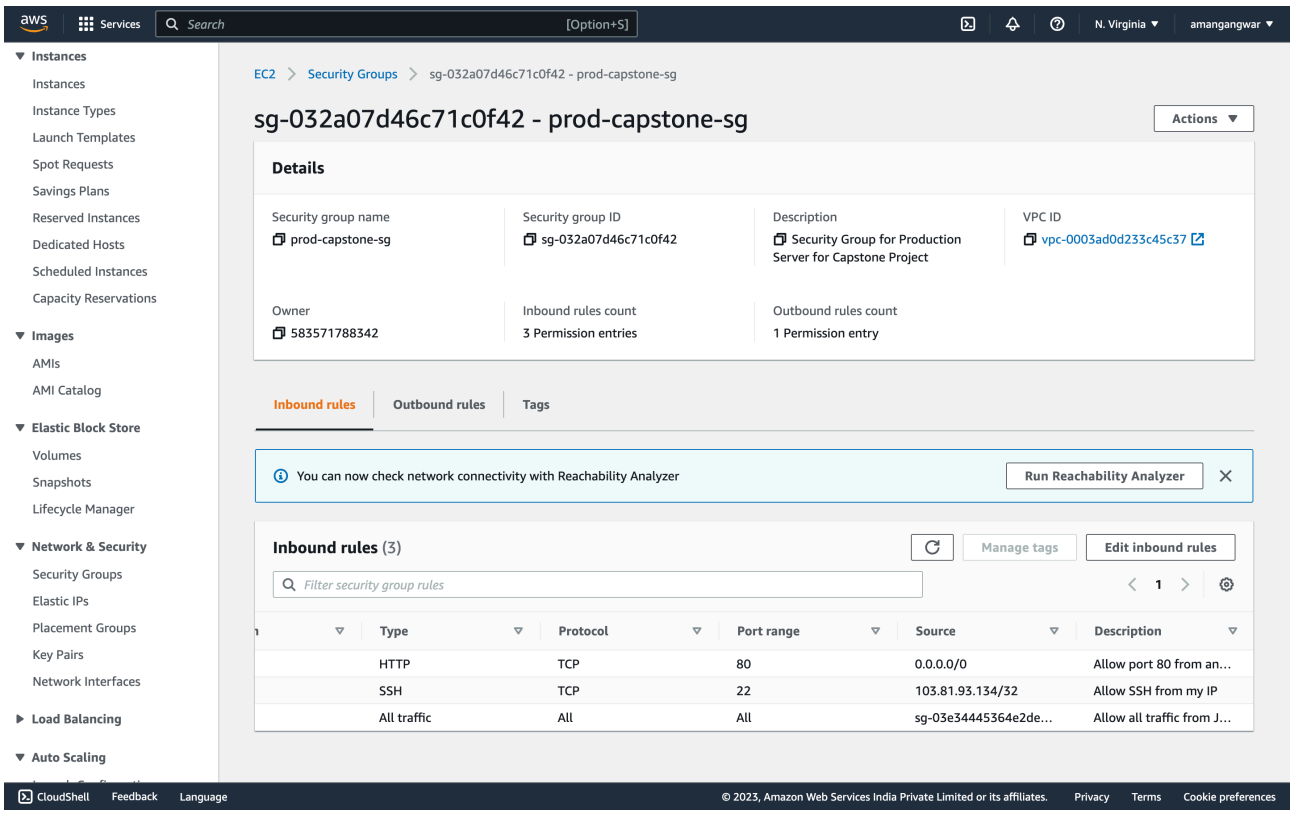


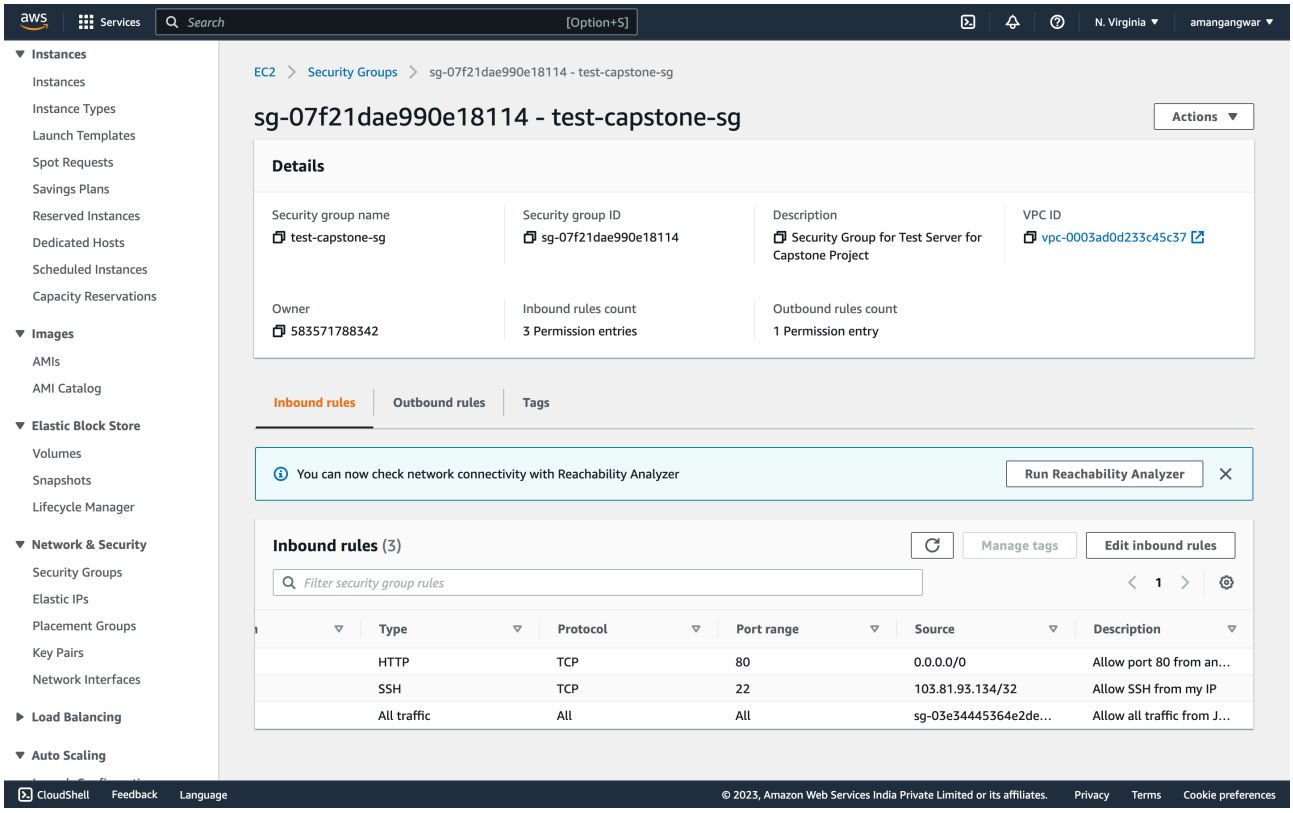
test-capstone



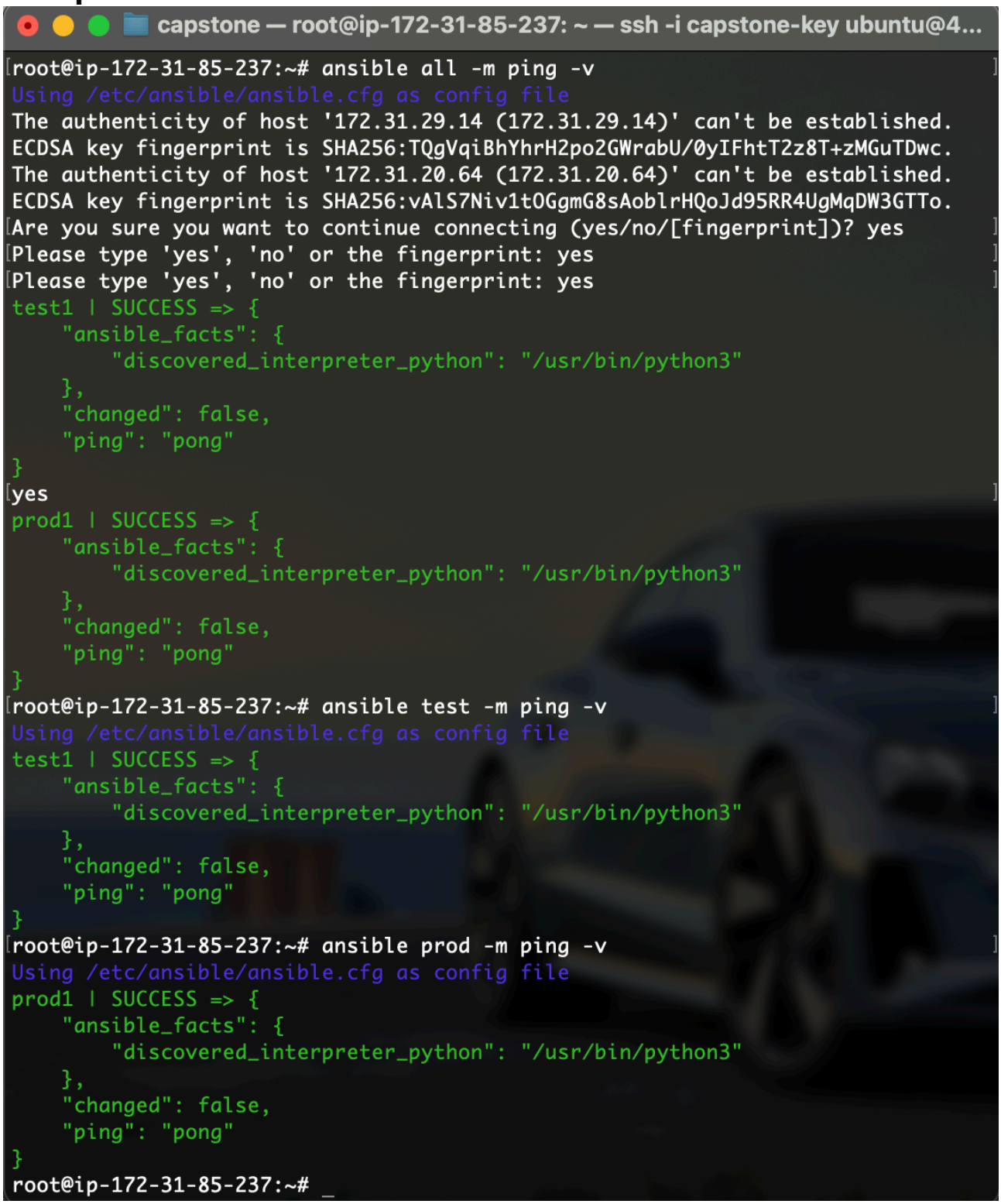
Security Group Screenshots




aws
Services
Search
[Option+S]
N. Virginia
amangangwar
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Scheduled Instances
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Auto Scaling
EC2 > Security Groups > sg-032a07d46c71c0f42 - prod-capstone-sg
sg-032a07d46c71c0f42 - prod-capstone-sg
Actions
Details
Security group name
prod-capstone-sg
Security group ID
sg-032a07d46c71c0f42
Description
Security Group for Production Server for Capstone Project
VPC ID
vpc-0003ad0d233c45c37
Owner
583571788342
Inbound rules count
3 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Tags
You can now check network connectivity with Reachability Analyzer
Run Reachability Analyzer
Inbound rules (3)
Manage tags
Edit inbound rules
Filter security group rules
1
Type
Protocol
Port range
Source
Description
HTTP
TCP
80
0.0.0.0/0
Allow port 80 from an...
SSH
TCP
22
103.81.93.134/32
Allow SSH from my IP
All traffic
All
All
sg-03e34445364e2de...
Allow all traffic from J...
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences


aws
Services
Search
[Option+S]
N. Virginia
amangangwar
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Scheduled Instances
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Auto Scaling
EC2 > Security Groups > sg-07f21dae990e18114 - test-capstone-sg
sg-07f21dae990e18114 - test-capstone-sg
Actions
Details
Security group name
test-capstone-sg
Security group ID
sg-07f21dae990e18114
Description
Security Group for Test Server for Capstone Project
VPC ID
vpc-0003ad0d233c45c37
Owner
583571788342
Inbound rules count
3 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Tags
You can now check network connectivity with Reachability Analyzer
Run Reachability Analyzer
Inbound rules (3)
Manage tags
Edit inbound rules
Filter security group rules
1
Type
Protocol
Port range
Source
Description
HTTP
TCP
80
0.0.0.0/0
Allow port 80 from an...
SSH
TCP
22
103.81.93.134/32
Allow SSH from my IP
All traffic
All
All
sg-03e34445364e2de...
Allow all traffic from J...
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

Setup Ansible Result

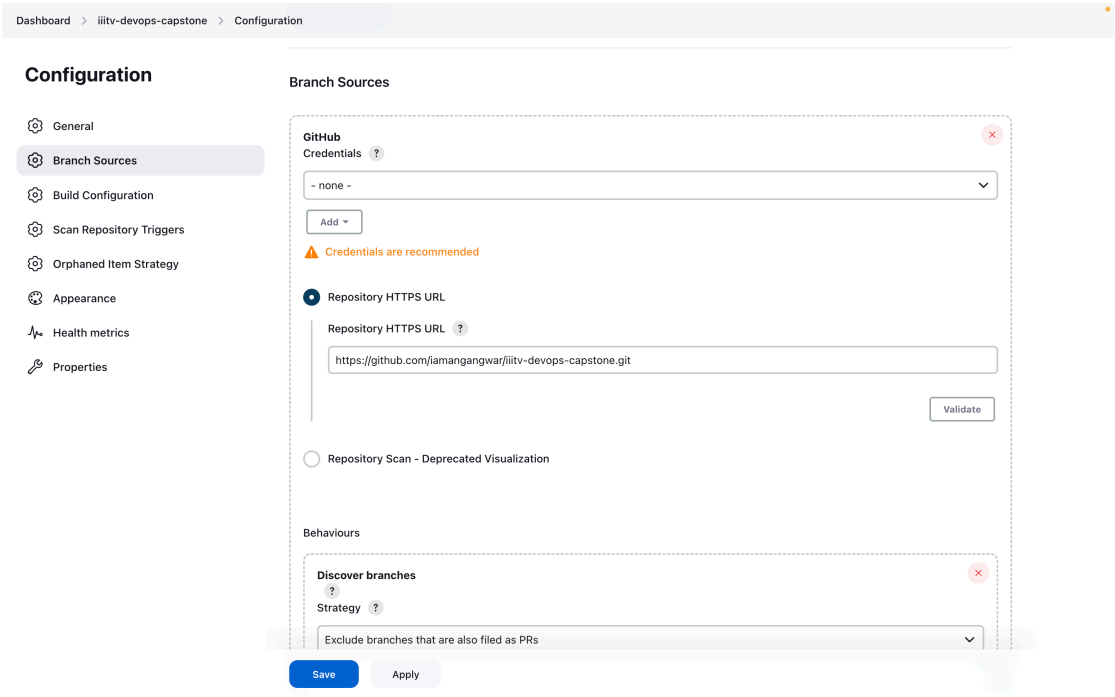


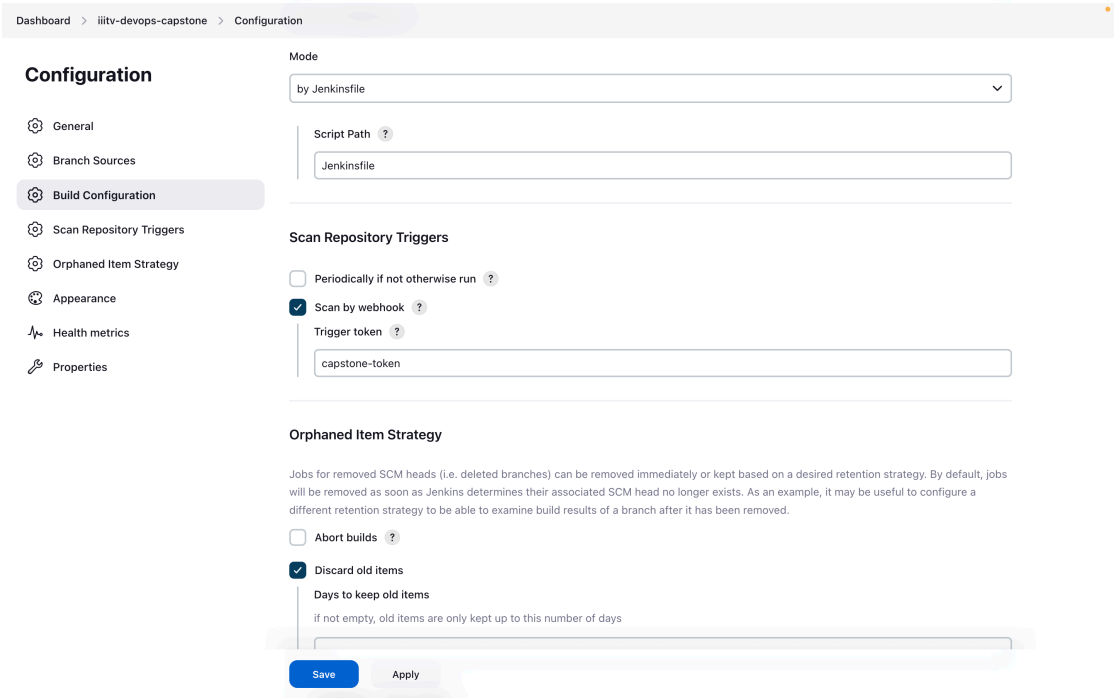
Docker Swarm Setup Result

Leader is ***jenkins-capstone*** instance, and worker is ***prod-capstone*** instance

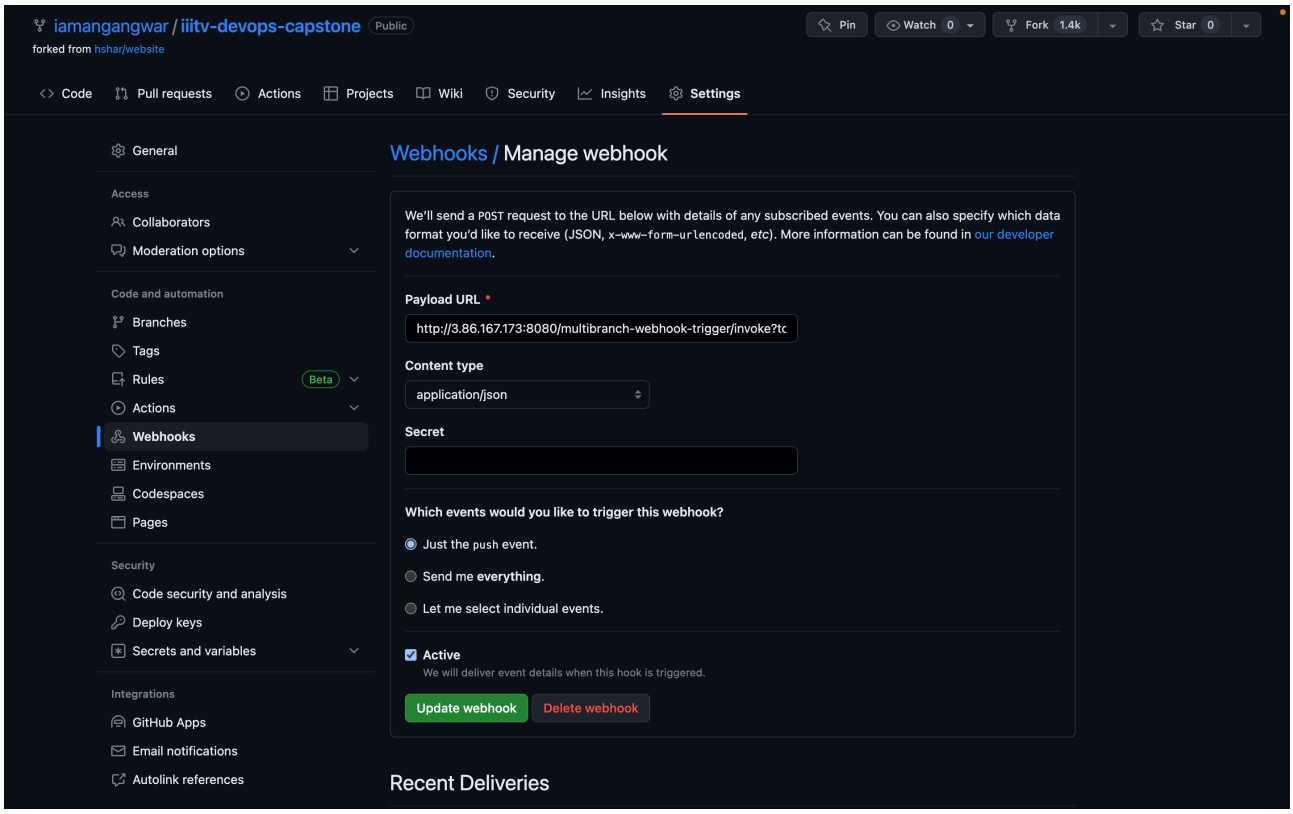
```
root@ip-172-31-85-237:~# docker node ls
ID                            HOSTNAME              STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
pzkrqiw95zf0dhdssz4lzxssp     ip-172-31-20-64       Ready     Active                          23.0.4
uf7nih98fq0ws9rybm2d3ejuz *   ip-172-31-85-237      Ready     Active         Leader           23.0.4
root@ip-172-31-85-237:~#
```

Jenkins Multibranch Pipeline Configuration

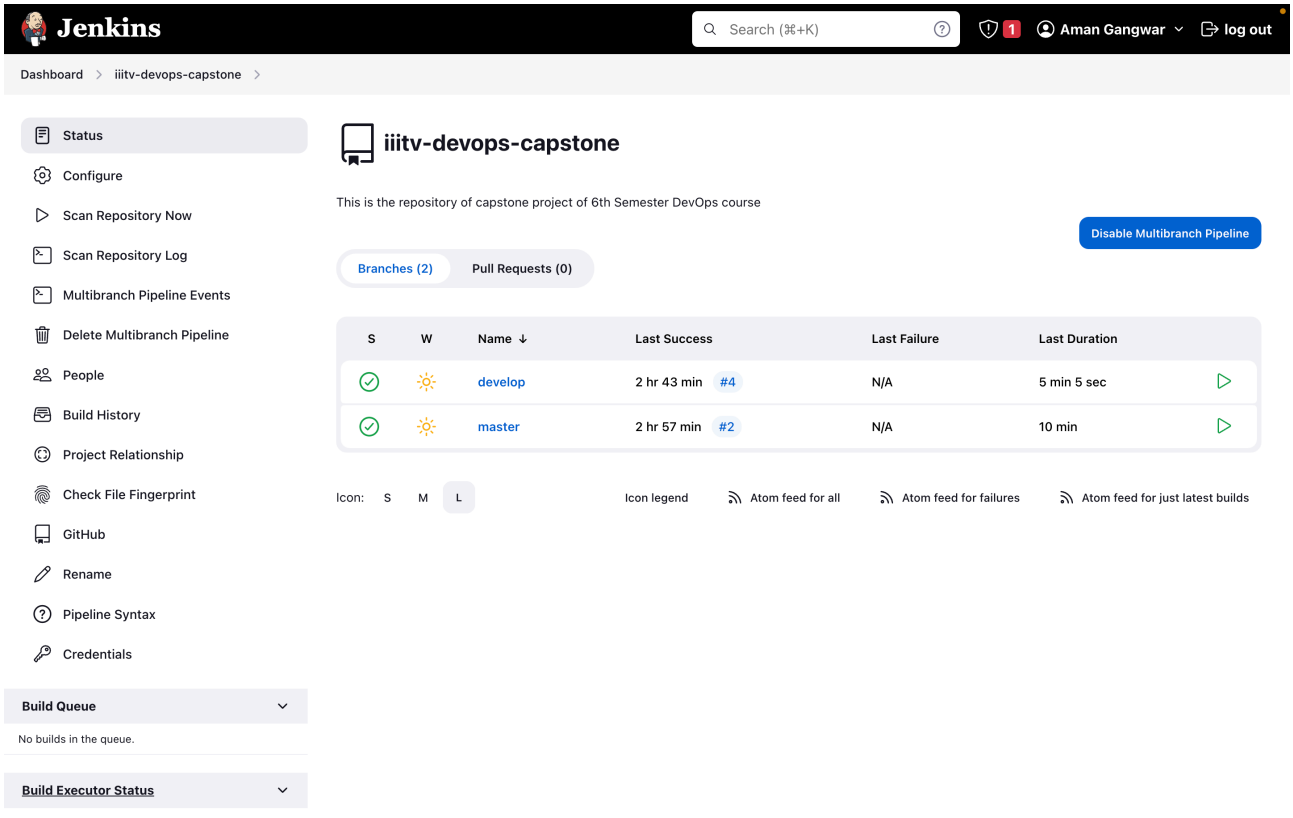




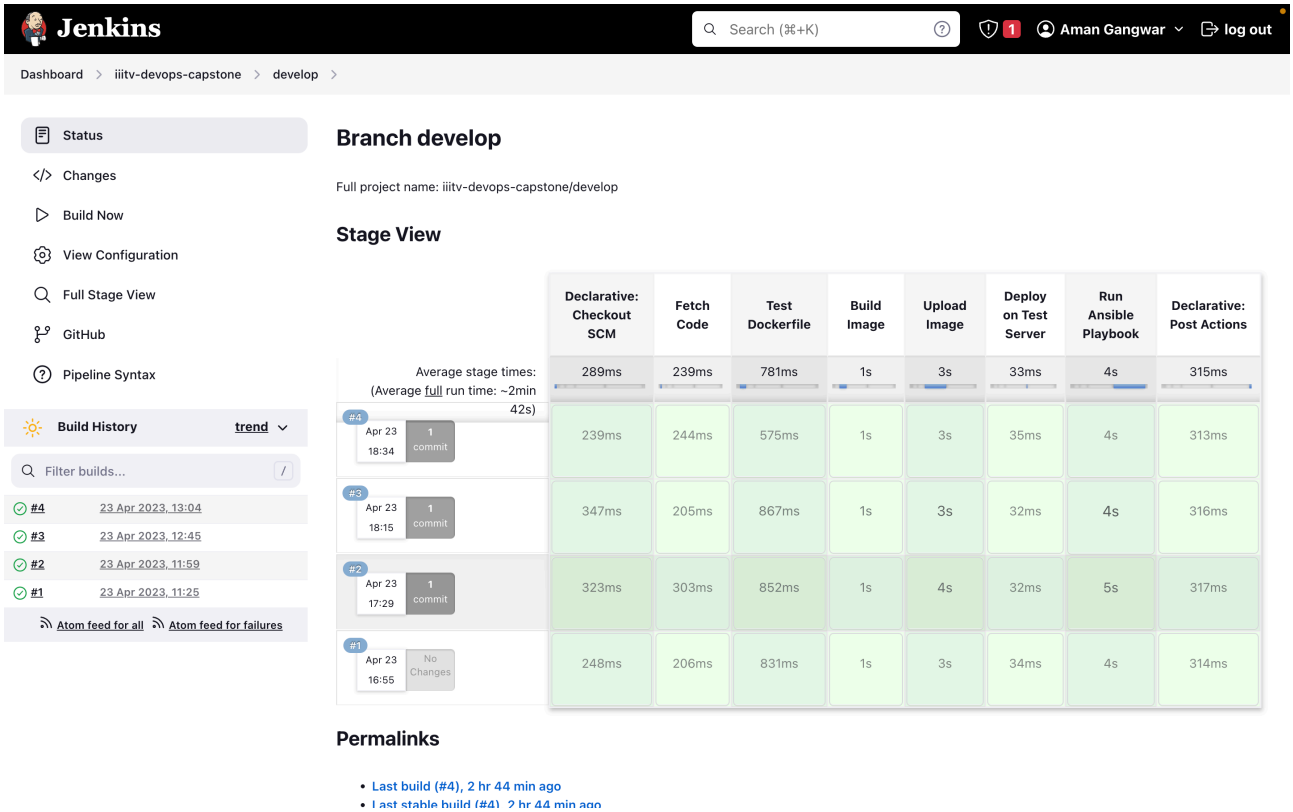
Github Webhook



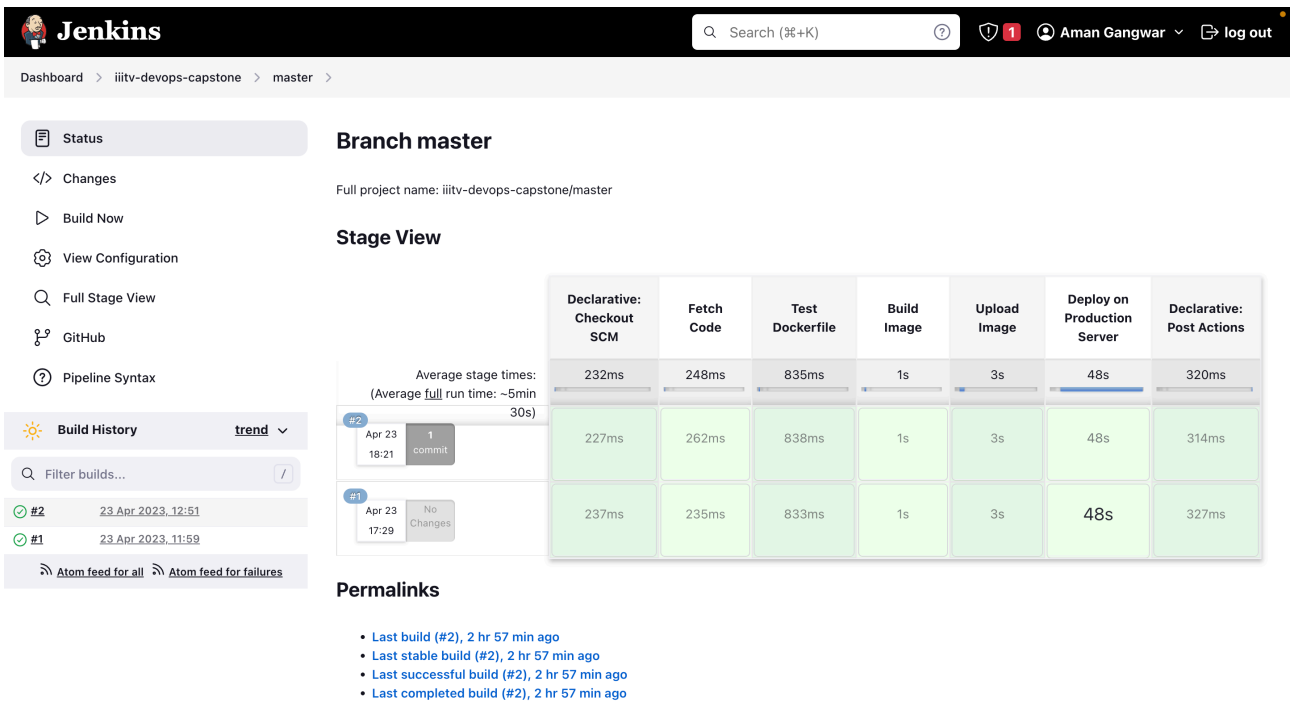
Jenkins Pipeline Screenshots



Develop branch -



Master branch -



Docker Containers running

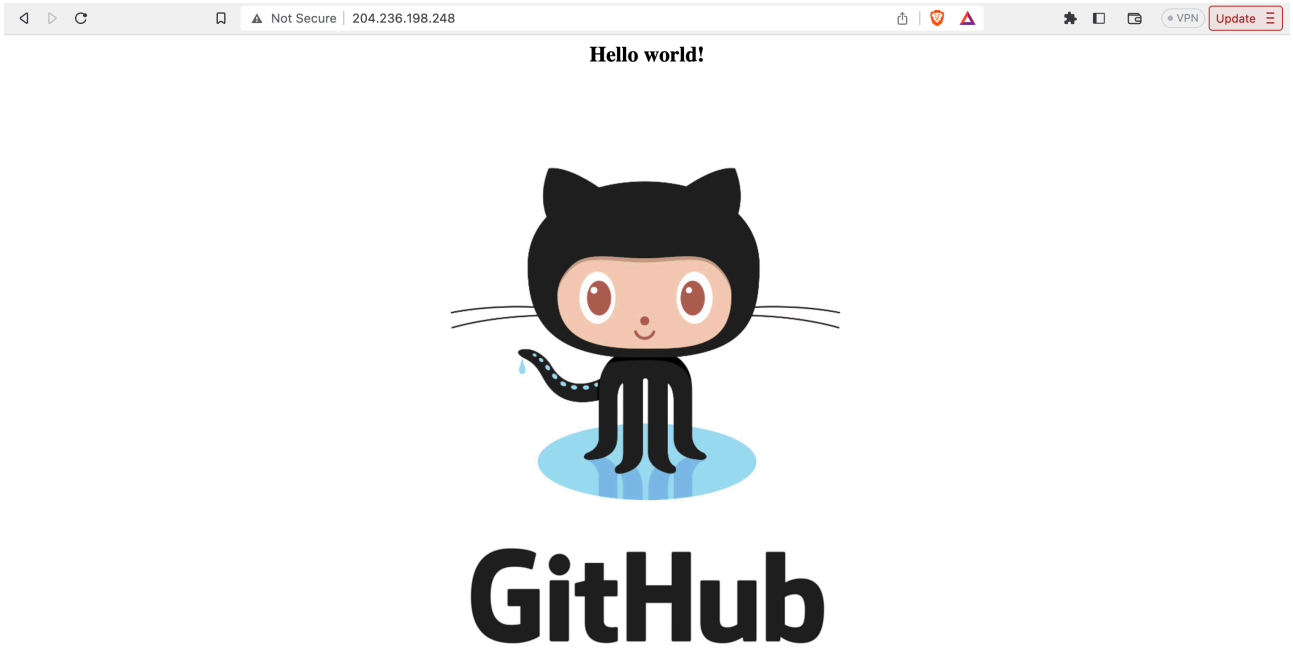
test-capstone instance

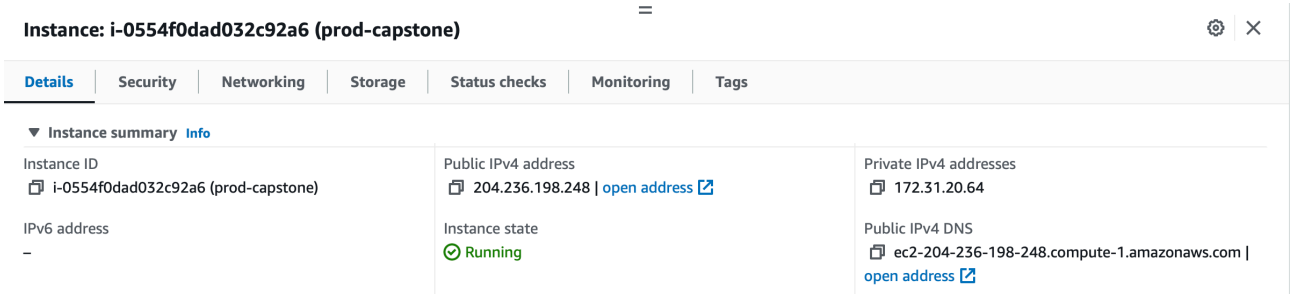
```
root@ip-172-31-29-14:~# docker ps
CONTAINER ID   IMAGE                                              COMMAND                  CREATED
   STATUS          PORTS                    NAMES
20c4a4fc997e   iamangangwar/iiitv-devops-capstone-test:latest   "/usr/sbin/apache2ct…"   23 minutes ago
   Up 23 minutes   0.0.0.0:80->80/tcp   test
```

Docker Swarm running

```
root@ip-172-31-85-237:~# docker ps
CONTAINER ID   IMAGE                                              COMMAND                  CREATED          STATUS          PORTS     NAMES
371b6755b1c0   iamangangwar/iiitv-devops-capstone-prod:latest   "/usr/sbin/apache2ct…"   42 minutes ago   Up 42 minutes   80/tcp    site.1.rzcc7lrd75htvux3ml3u7hj2b
630438a102e8   iamangangwar/iiitv-devops-capstone-prod:latest   "/usr/sbin/apache2ct…"   3 hours ago      Up 3 hours      80/tcp    site.3.ruphx3t4p35a3uc4u7payct2c
f977e41f8306   iamangangwar/iiitv-devops-capstone-prod:latest   "/usr/sbin/apache2ct…"   3 hours ago      Up 3 hours      80/tcp    site.2.qad5es95iqgvvqds083eyvkcb
root@ip-172-31-85-237:~#
```

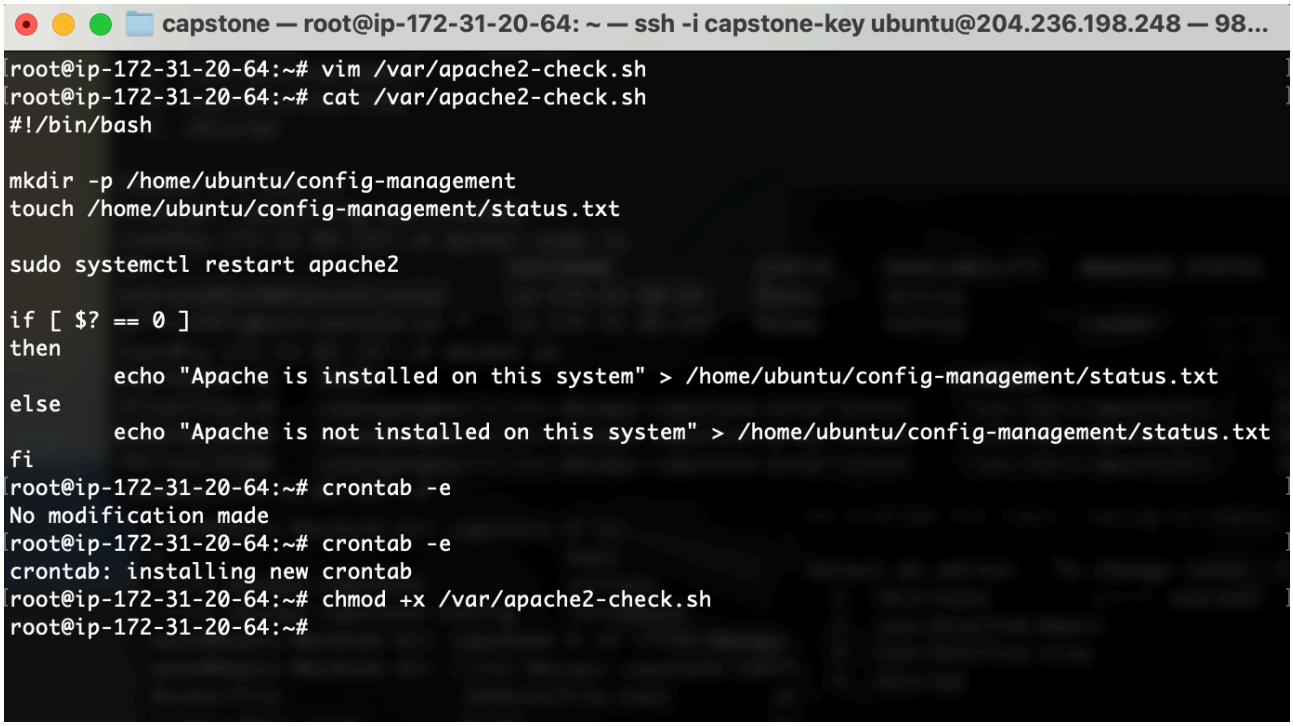
Running on Production Instance



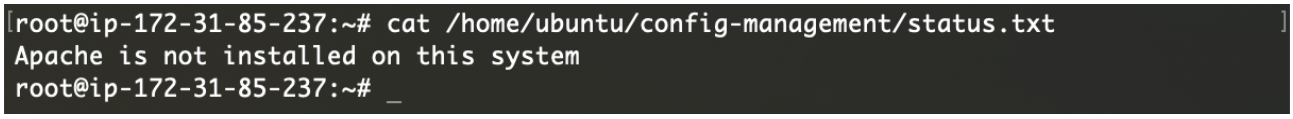


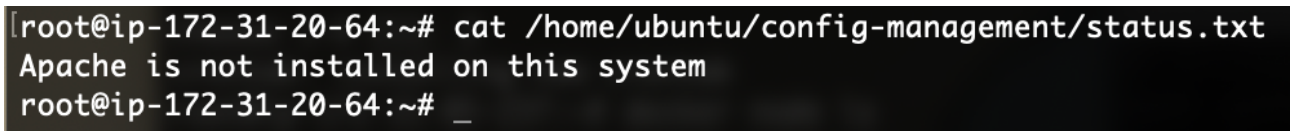
Creating status.txt in /home/ubuntu/config-management

Shell Script -



Files in all 3 instances -





Apache is not installed on any of the systems because I have deployed docker containers containing apache2 server, hence apache2 needs not be installed in the virtual machine.

All the scripts I used are present in my GitHub repository :)
https://github.com/iamangangwar/iiitv-devops-capstone-config.git