Milestone2

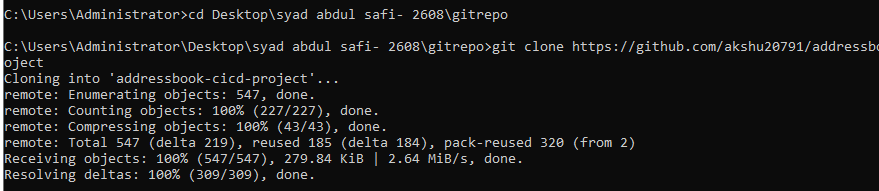
## **Syad abdul safi**

**Step 1:**

Create a new folder and clone this repo on it using this command

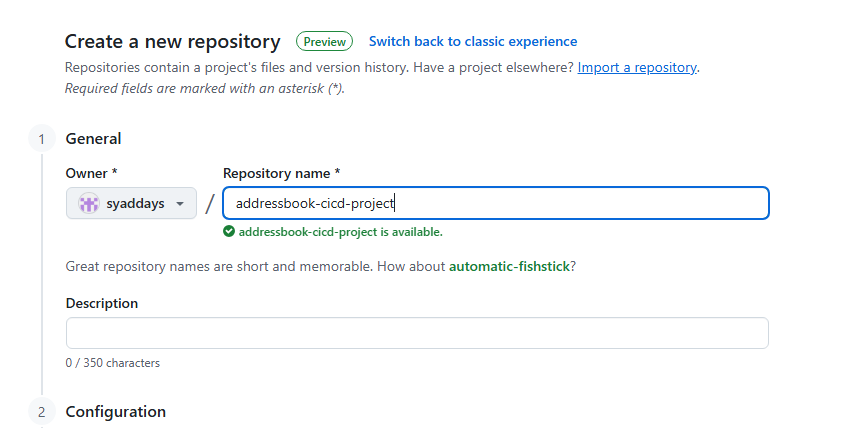
git clone <https://github.com/akshu20791/addressbook-cicd-project>

cd addressbook-cicd-project



**Step2:**

Create a new empty repository on your github: addressbook-cicd-project

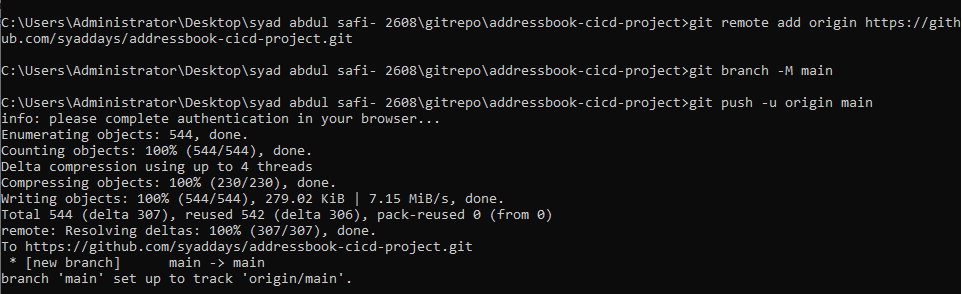


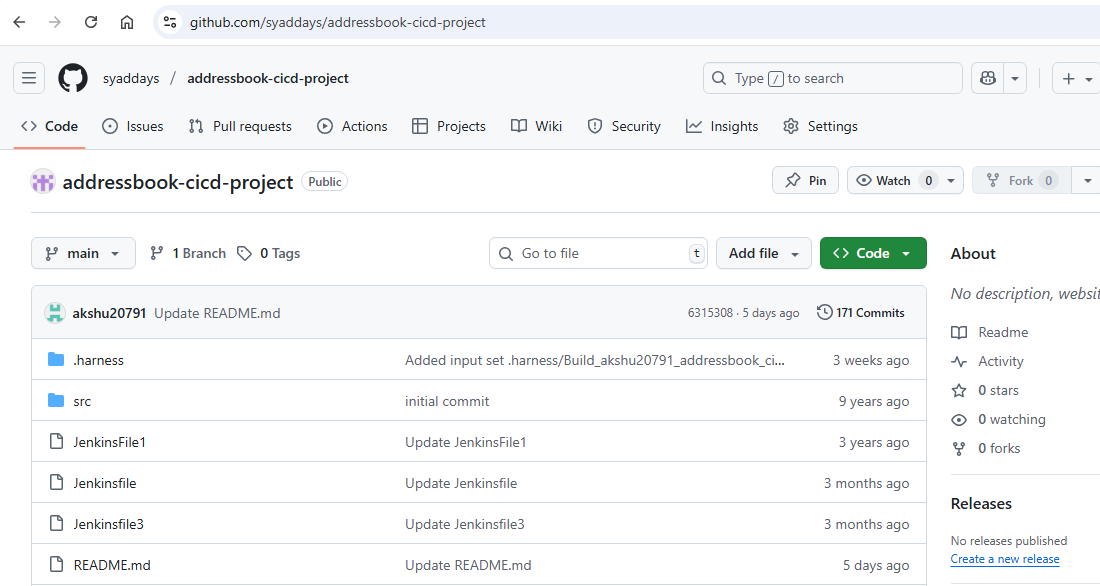
On the terminal right these commands to push this addressbook repo to your personal github repo

git remote add origin https://github.com/syaddays/addressbook-cicd-project.git

git branch -M main

git push -u origin main



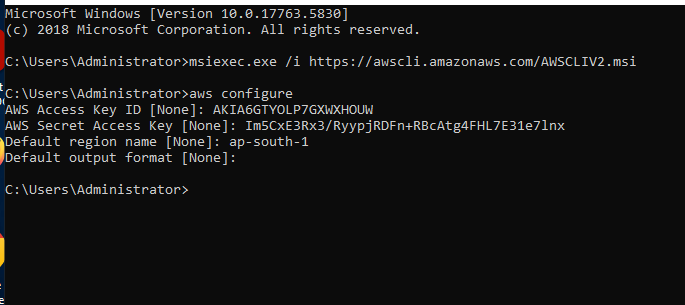


**Step 3 :** Install AWS CLI

msiexec.exe /i <https://awscli.amazonaws.com/AWSCLIV2.msi>

aws configure

now enter the details as asked

:

**2) Write a Terraform Script to Provision AWS Infrastructure**

Create a new folder name terraform\_files

Touch [main.tf](http://main.tf)

* Things to keep in mind:
* Check the region, ami, keypair

#1. Configure the AWS Provider

provider "aws" {

region = "ap-south-1" # You can choose a different region

}

# 2. Define a security group for the Jenkins Master

resource "aws\_security\_group" "jenkins\_sg" {

name = "jenkins-sg"

description = "Allow SSH, Jenkins UI, and HTTP traffic"

# Allow SSH from your current IP

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"] # For learning purposes only. In production, restrict this to your IP.

}

# Allow access to Jenkins UI

ingress {

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"]

}

}

# 3. Define a security group for the Application Node

resource "aws\_security\_group" "app\_node\_sg" {

name = "app-node-sg"

description = "Allow SSH and Tomcat access"

# Allow SSH from your current IP

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"] # For learning purposes only.

}

# Allow access to Tomcat

ingress {

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"]

}

}

# 4. Provision the Jenkins Master EC2 instance

resource "aws\_instance" "jenkins\_master" {

ami = "ami-02d26659fd82cf299" # Amazon Linux 2 AMI for us-east-1

instance\_type = "t2.micro"

key\_name = "mumbai" # IMPORTANT: Replace with your EC2 key pair name

vpc\_security\_group\_ids = [aws\_security\_group.jenkins\_sg.id]

tags = {

Name = "Jenkins-Master"

}

}

# 5. Provision the Application Node EC2 instance

resource "aws\_instance" "app\_node" {

ami = "ami-02d26659fd82cf299" # Amazon Linux 2 AMI for us-east-1

instance\_type = "t2.micro"

key\_name = "mumbai" # IMPORTANT: Replace with your EC2 key pair name

vpc\_security\_group\_ids = [aws\_security\_group.app\_node\_sg.id]

tags = {

Name = "Application-Node"

}

}

# 6. Output the public IP addresses of the instances

output "jenkins\_master\_public\_ip" {

value = aws\_instance.jenkins\_master.public\_ip

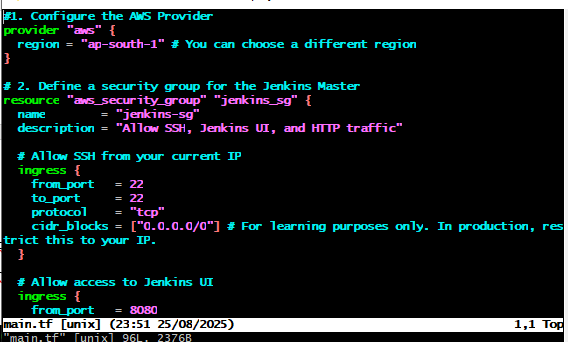
}

output "app\_node\_public\_ip" {

value = aws\_instance.app\_node.public\_ip

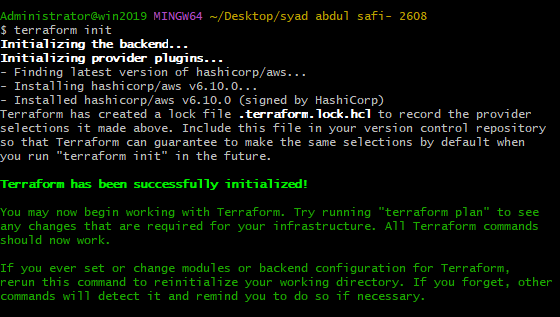
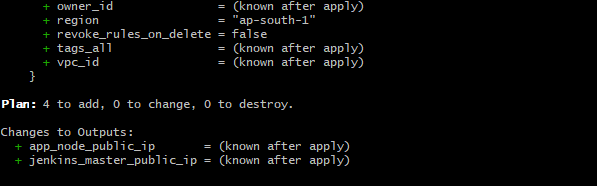
}

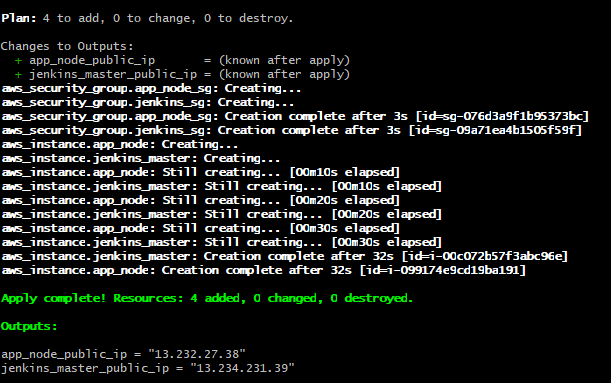




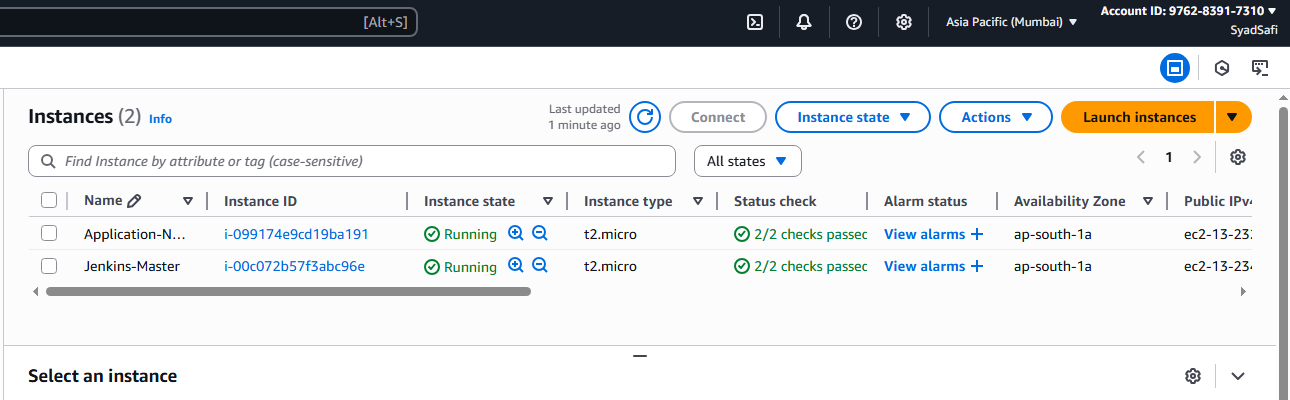
After that save the main.tf file

Now we will perform terraform commands to run the tf file

1. terraform init
2. terraform plan
3. terraform apply –auto-approve



Now we can see the instances running on our aws console:



**Step 4:**

Install ansible on Jenkins master

sudo su

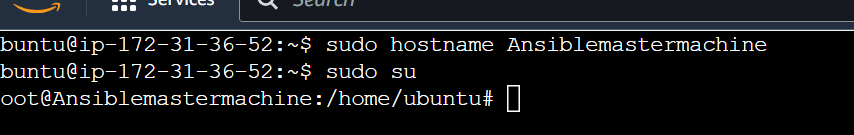
apt update -y

apt-get install -y software-properties-common

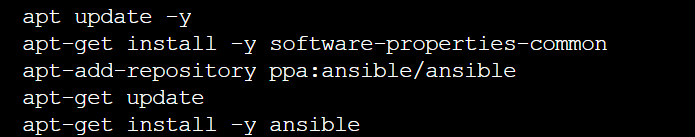
apt-add-repository ppa:ansible/ansible

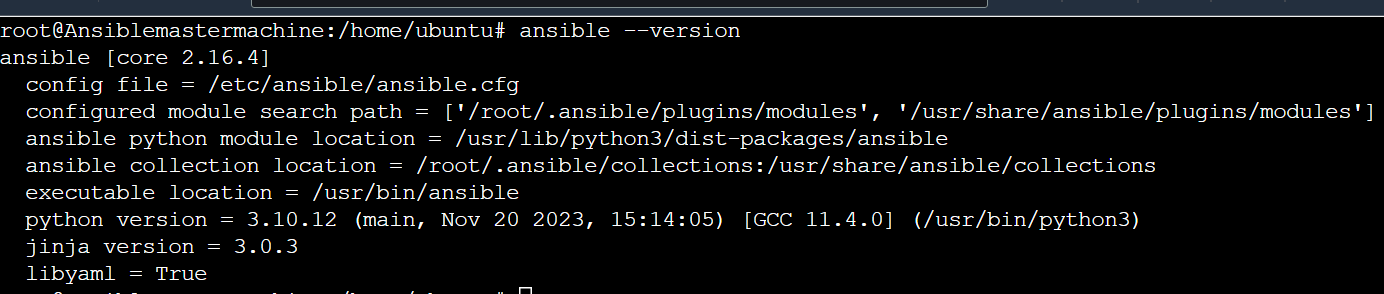
apt-get update

apt-get install -y ansible



# ansible --version





Install jekins

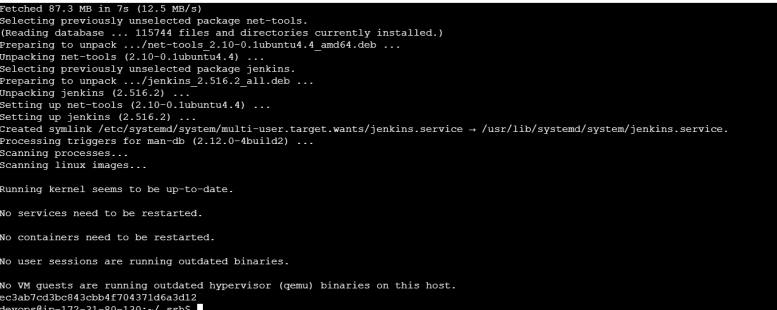
Install openjdk-17-jdk-y

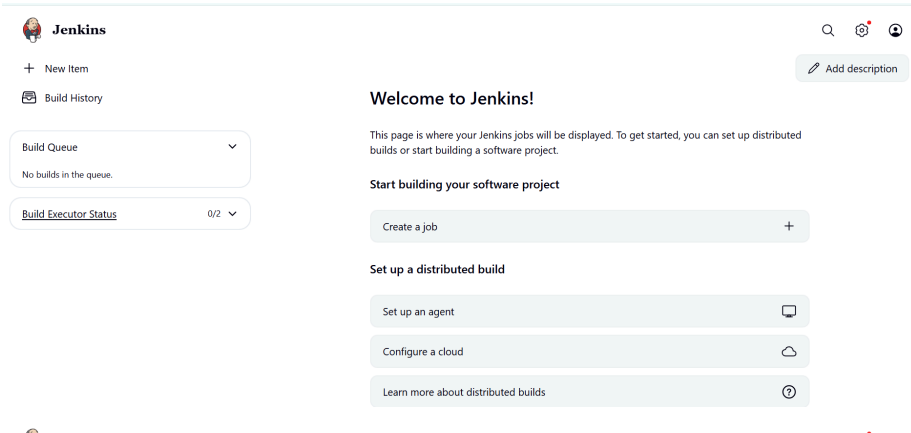
Wget <https://github.com/syaddays/deployments-script/raw/main/jenkins.sh>

Chmod +x Jenkins.sh

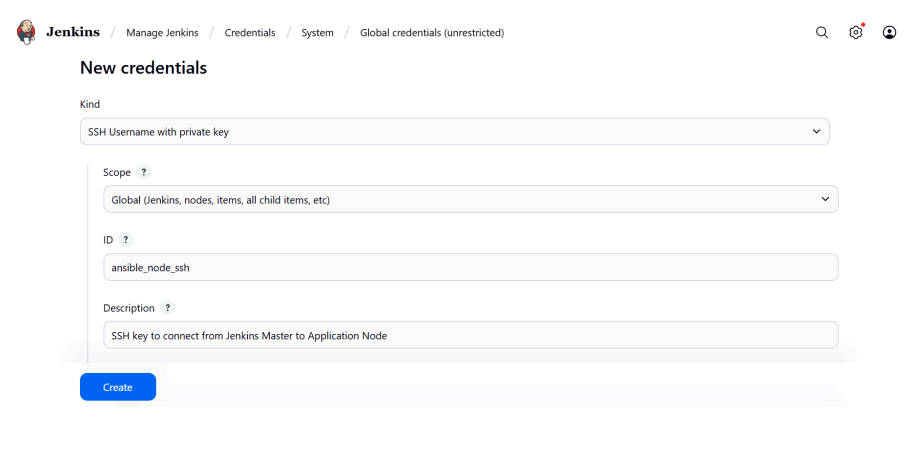
Chmod +x Jenkins.sh

Sudo ./Jenkins.sh

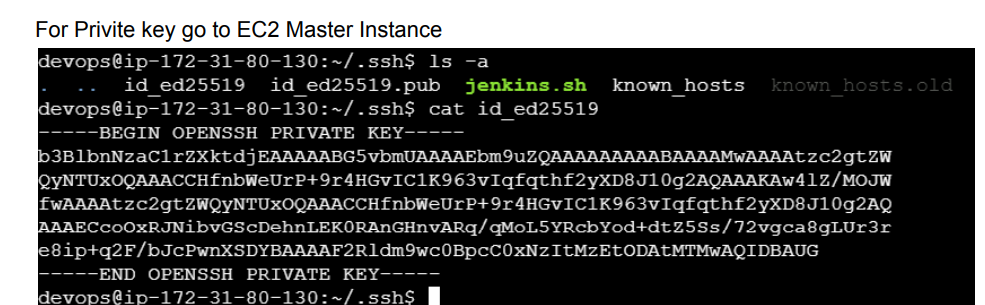


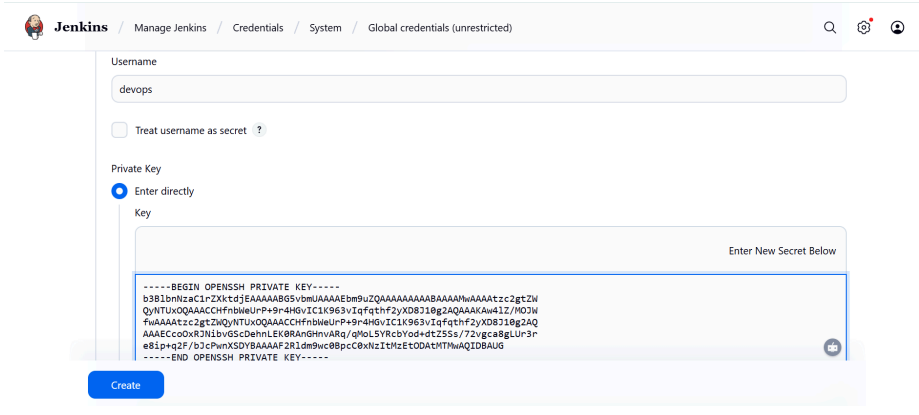


Set up new credentials

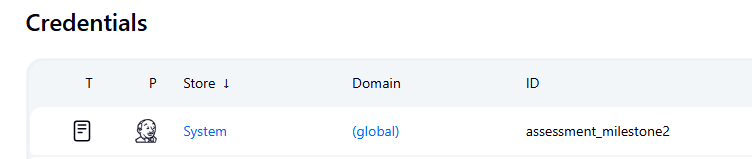


For private key go to ec2 instance





New creadentials will be created



On your master EC2:

mkdir addressbook\_repo

cd addressbook\_repo

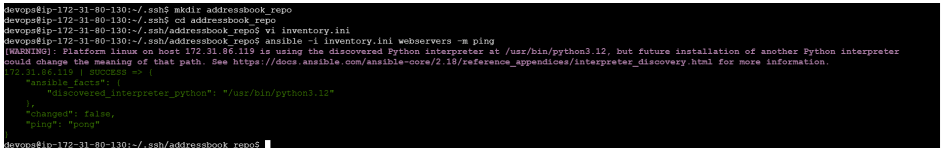
Inventory file content:- nodes private ip is placed in this file

vi inventory.ini

[webservers]

<private-id> ansible\_user=devops

ansible -i inventory.ini webservers -m ping



Maven must be installed on your Jenkins Master EC2 instance before you run the

pipeline, because the Jenkinsfile uses Maven commands (mvn clean package) to

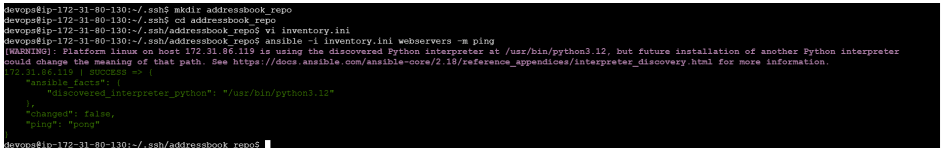
build the .war file.

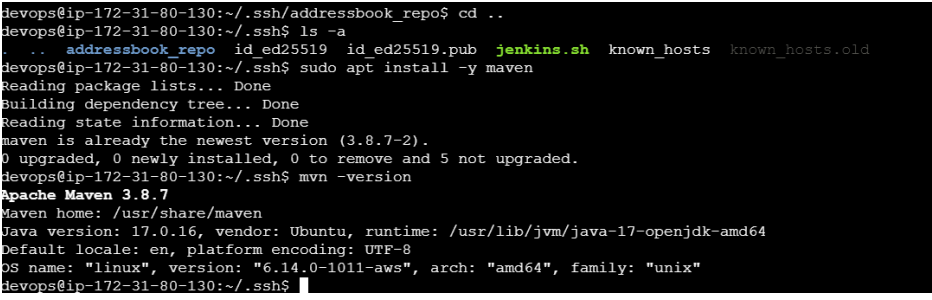
# Install Maven

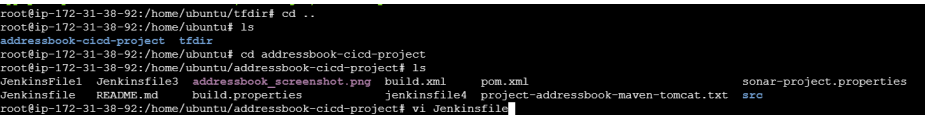
sudo apt install -y maven

# Verify installation

mvn –version







pipeline {

agent any

environment {

WAR\_FILE = "target/addressbook.war"

TOMCAT\_HOME = "/opt/tomcat9"

INVENTORY = "/home/devops/.ssh/addressbook\_repo/inventory.ini"

}

stages {

stage('Checkout Code') {

steps {

git branch: 'master', url: 'https://github.com/syaddays/addresbook-cicd

}

}

stage('Build with Maven') {

steps {

sh 'mvn clean package'

}

}

stage('Install Tomcat 9 via Ansible') {

steps {

writeFile file: 'tomcat.yml', text: '''

- hosts: webservers

become: true

tasks:

- name: Install unzip

apt:

name: unzip

state: present

update\_cache: yes

- name: Download Tomcat 9

get\_url:

url: https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.108/bin/apache-tomcat-9.0.108.zip

dest: /tmp/apache-tomcat-9.0.108.zip

- name: Extract Tomcat

unarchive:

src: /tmp/apache-tomcat-9.0.108.zip

dest: /opt/

remote\_src: yes

- name: Rename Tomcat folder

command: mv /opt/apache-tomcat-9.0.108 /opt/tomcat9

args:

creates: /opt/tomcat9

- name: Make Tomcat scripts executable

command: chmod +x /opt/tomcat9/bin/\*.sh

'''

sh "ansible-playbook -i ${INVENTORY} tomcat.yml"

}

}

stage('Deploy WAR to Tomcat') {

steps {

sh "ansible webservers -i ${INVENTORY} -m copy -a \"src=${WAR\_FILE} dest=${TOMCAT\_HOME}/webapps/addressbook.war\" --become"

sh "ansible webservers -i ${INVENTORY} -m shell -a \"${TOMCAT\_HOME}/bin/shutdown.sh || true\" --become"

sh "ansible webservers -i ${INVENTORY} -m shell -a \"${TOMCAT\_HOME}/bin/startup.sh\" --become"

}

}

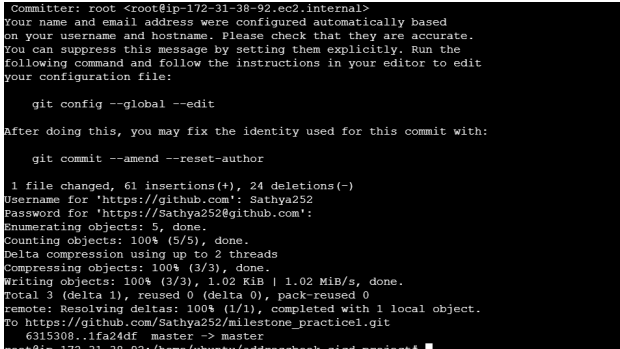
}}

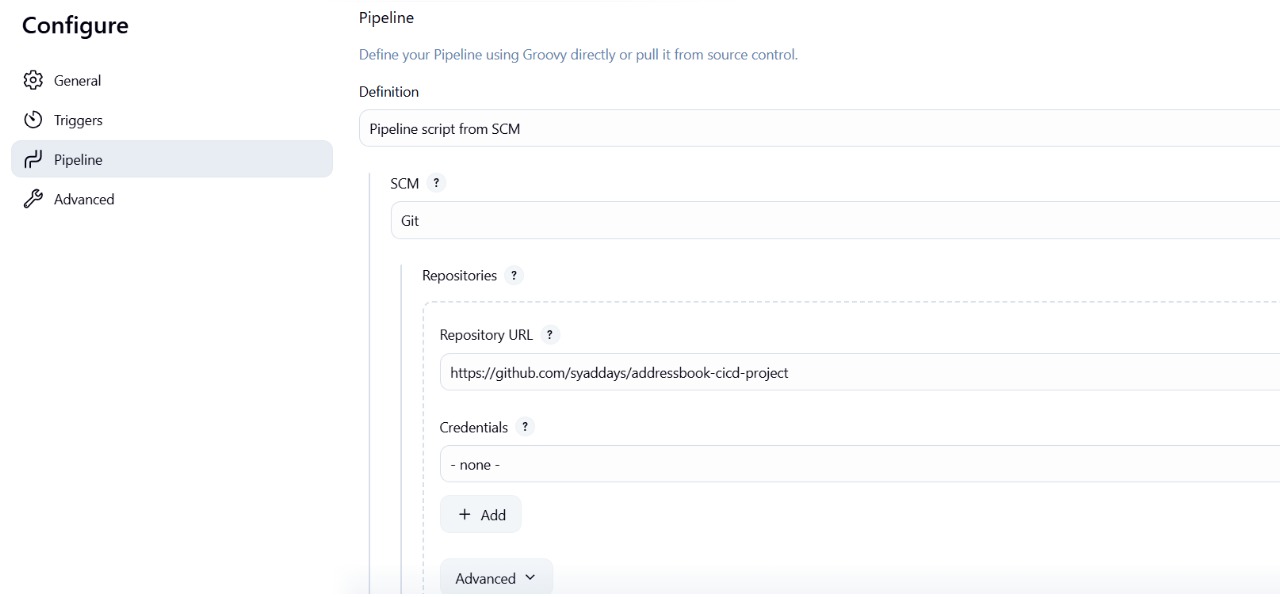
Push this Modified Jenkinsfile to your repo:

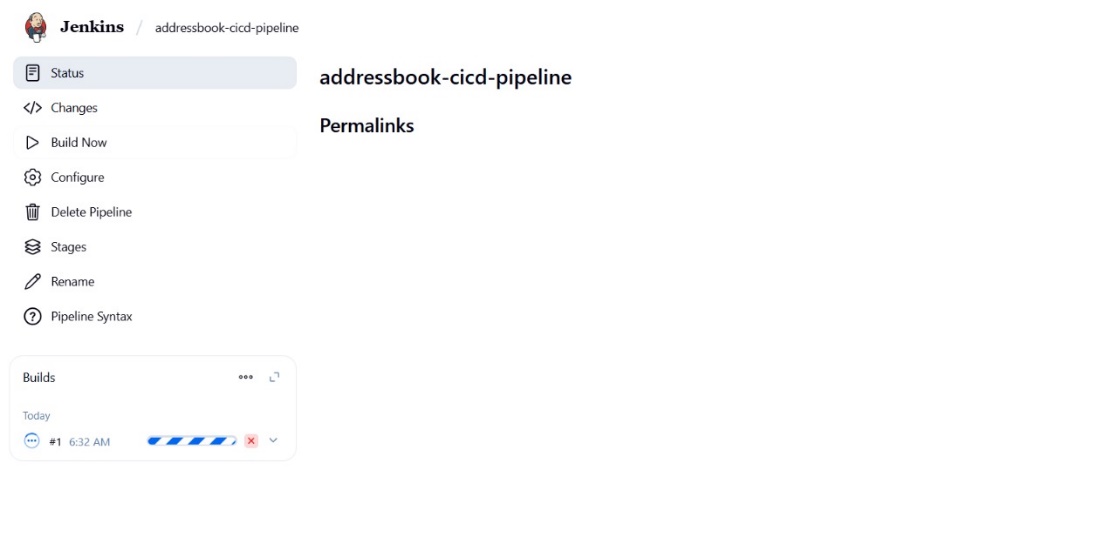
git add Jenkinsfile

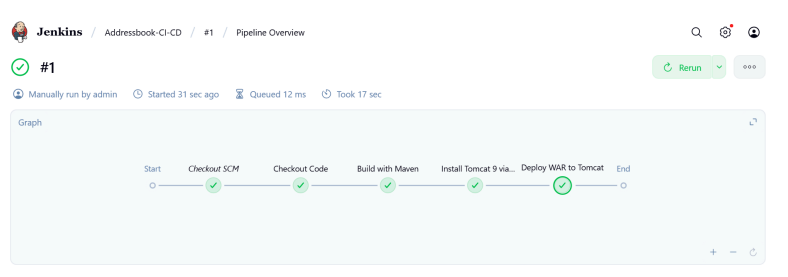
git commit -m "Final Jenkinsfile ready for master branch"

git push origin master

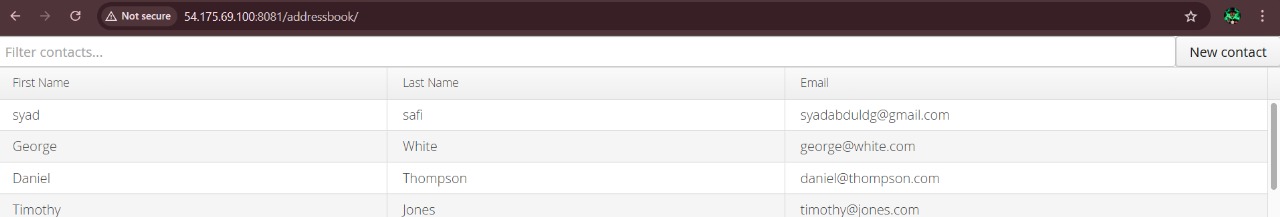








Go to public ip:addressbook to see the running application:



To add github webhook:

Got to repo settings->webhooks->

Which events would you like to trigger this webhook?

Select Just the push event->Click Add webhook

Give payload url and set content type as json

