Problem Description:-

As soon as the developer pushes the updated code on the GIT master branch, the code should be checked out, compiled, tested, packaged and containerized. A new test-server should be provisioned using terraform and should be automatically configured using Ansible with all the required software's and as soon as the server is available, the application must be deployed to the test-server automatically.

The deployment should then be tested using a test automation tool, and if the build is Successful, Prod server must be configured with all the software it should be pushed to the prod server. All this should happen automatically and should be triggered from a push to the GitHub master branch.

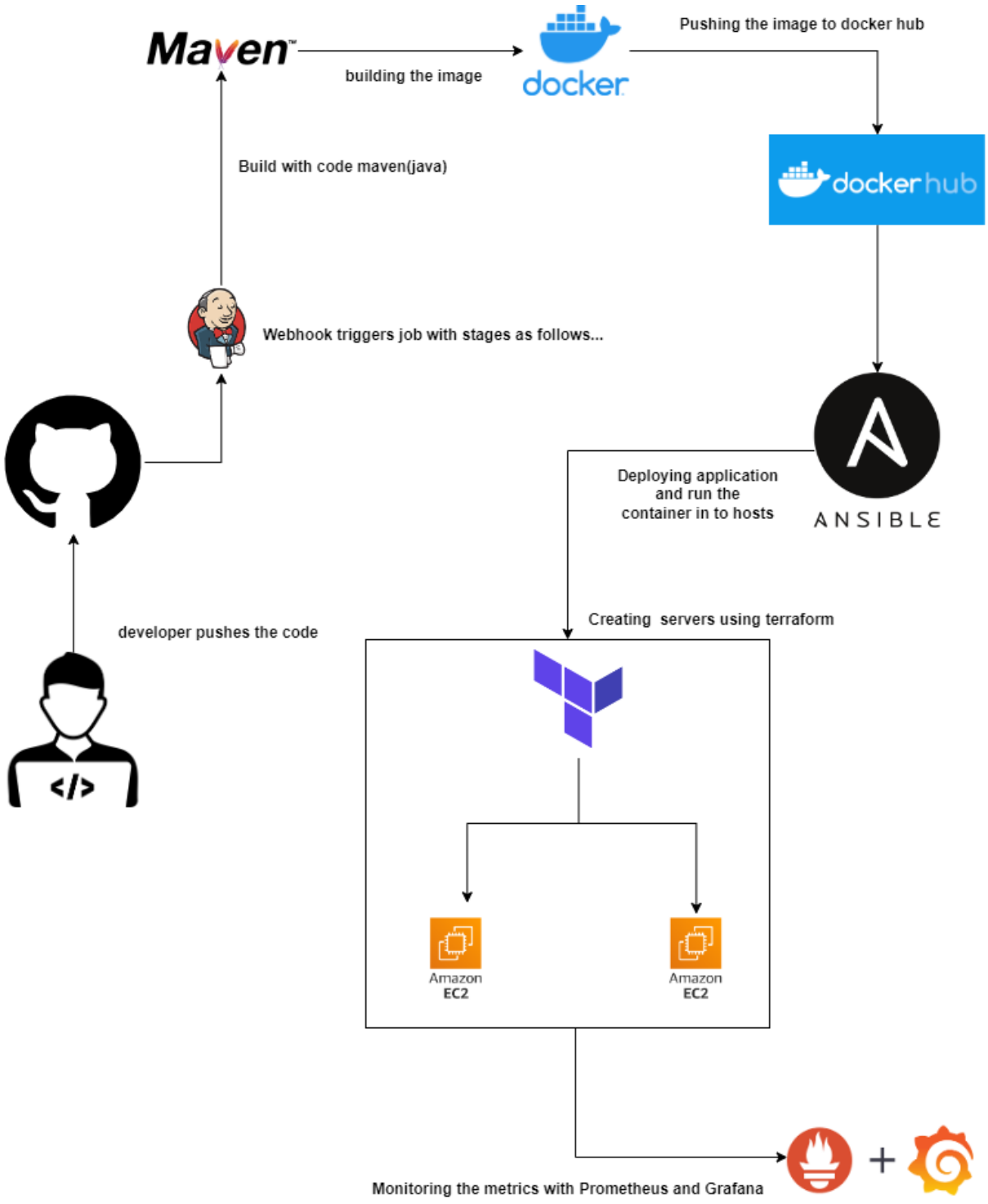
Continuous monitoring server must be configured to monitor the test as well as prod server using Prometheus and Grafana should be configured to display a dashboard with following

metrics.

1. CPU utilization
2. Disk Space Utilization
3. Total Available Memory

Problem Solution:-

Approach-



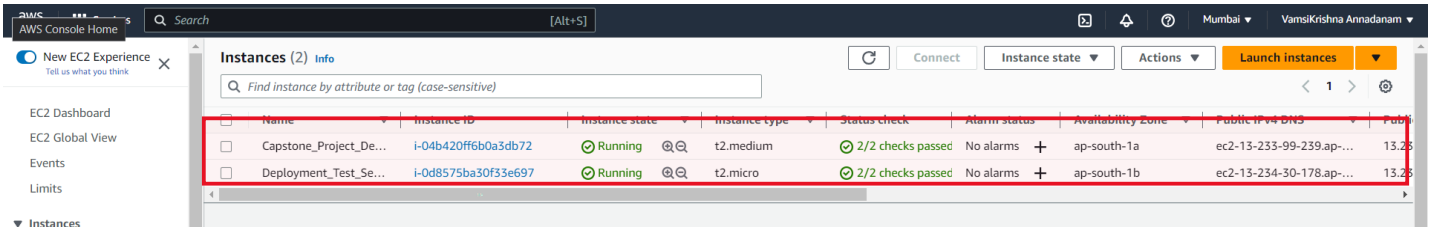
Source git hub - https://github.com/vamsikrishna918/Finance-Web-Application-E2E

Step 1:

1.A- EC2 instance setup

Creating 2 EC2 instanes name as below

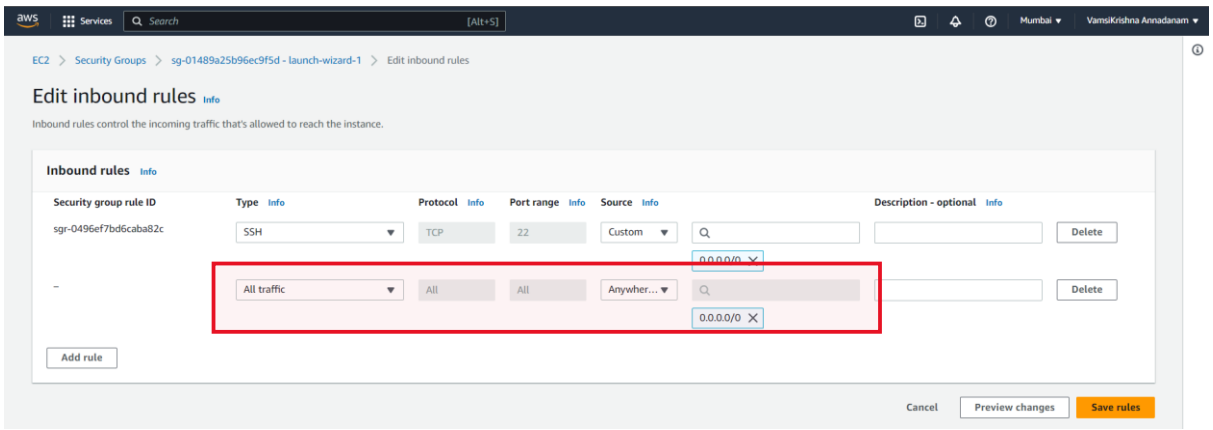
1. Capstone_Project_Development (T2.Medium with 30gib)
2. Deployment_Test_Server (T2.Micro with 8gib)



1.B

Setting up the security groups – allowing all Inbound traffic to both instances

Path: Security->Edit Inbound rules

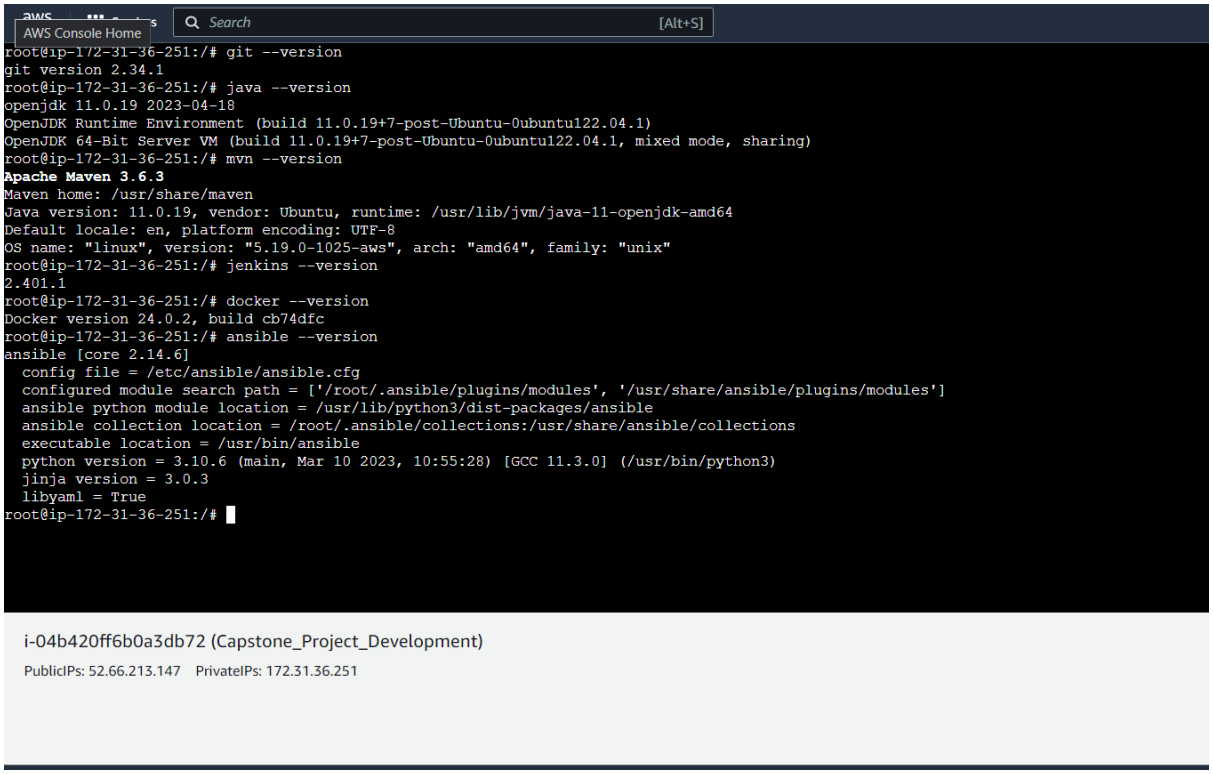


1.C Installing packages

Installing the necessary packages in instance (Capstone_Project_Development)

Follow the below sources.

1. Update packages – apt update
2. Git- https://git-scm.com/book/en/v2/Getting-Started-Installing-Git
3. Java- https://www.digitalocean.com/community/tutorials/how-to-install-java-with-apt-on-ubuntu-22-04
4. Maven- sudo apt install maven
5. Jenkins - https://www.jenkins.io/doc/book/installing/linux/#debianubuntu
6. Docker - https://docs.docker.com/engine/install/ubuntu/
7. Ansible- https://docs.ansible.com/ansible/latest/installation_guide/installation_distros.html#installing-ansible-on-debian
8. Terraform - https://developer.hashicorp.com/terraform/tutorials/aws-get-started/install-cli
9. aws cli - https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html



1.d - Password less authentication

Password less authetication b/w 2 servers (capstone_project_development (and) deployment_test_server)

On capstone_project_development instacne

Do command: ssh-keygen

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
/home/ubuntu/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
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:dTOLZQqBjiD62h/CdZPwb1mvsEz7Mkc25pcZdtR6TrI ubuntu@ip-172-31-53-60
The key's randomart image is:
+---[RSA 3072]----+
|       ..        |
|. .   .  .       |
|.. ..o  . . *  . |
|.   .o.. o * +. .|
| .  . = S + .. . |
| ... . o o=.o + o|
| oo .   *= o.= * |
|. .. . +o+o.+ E .|
|   ..   +=o.     |
```

It will generate public and private keys as below

```
ubuntu@ip-172-31-53-60:~/ansible$ ls /home/ubuntu/.ssh/
authorized_keys  id_rsa  id_rsa.pub  known_hosts
ubuntu@ip-172-31-53-60:~/ansible$
```

Copy the id_rsa.pub content, paste it in the deployment_test_server authotized keys file

On deployment_test_server instacne

Do command: ssh-keygen

```
ubuntu@ip-172-31-62-28:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
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:AFIPl190A+Zj2rUDuoNXoW8BUVp4QwkkudTphdDLbmw ubuntu@ip-172-31-62-28
The key's randomart image is:
+---[RSA 3072]----+
|  ..+ +B+OBoo    |
|   . =+.***. .   |
|     .o=o=*..    |
|      ..=B = .   |
|       oS + o    |
|        .E+ . .  |
|       .o+ o     |
|        . o      |
|                 |
+----[SHA256]-----+
ubuntu@ip-172-31-62-28:~$
```

```
ubuntu@ip-172-31-62-28:~$ ls ~/.ssh/
authorized_keys  id_rsa  id_rsa.pub
ubuntu@ip-172-31-62-28:~$ vim ~/.ssh/authorized_keys
```

Copy the id_rsa.pub content in capstone_project_development.

paste it in the deployment_test_server **authotized_keys file**

1.e ansible setup

On capstone_project_development instacne

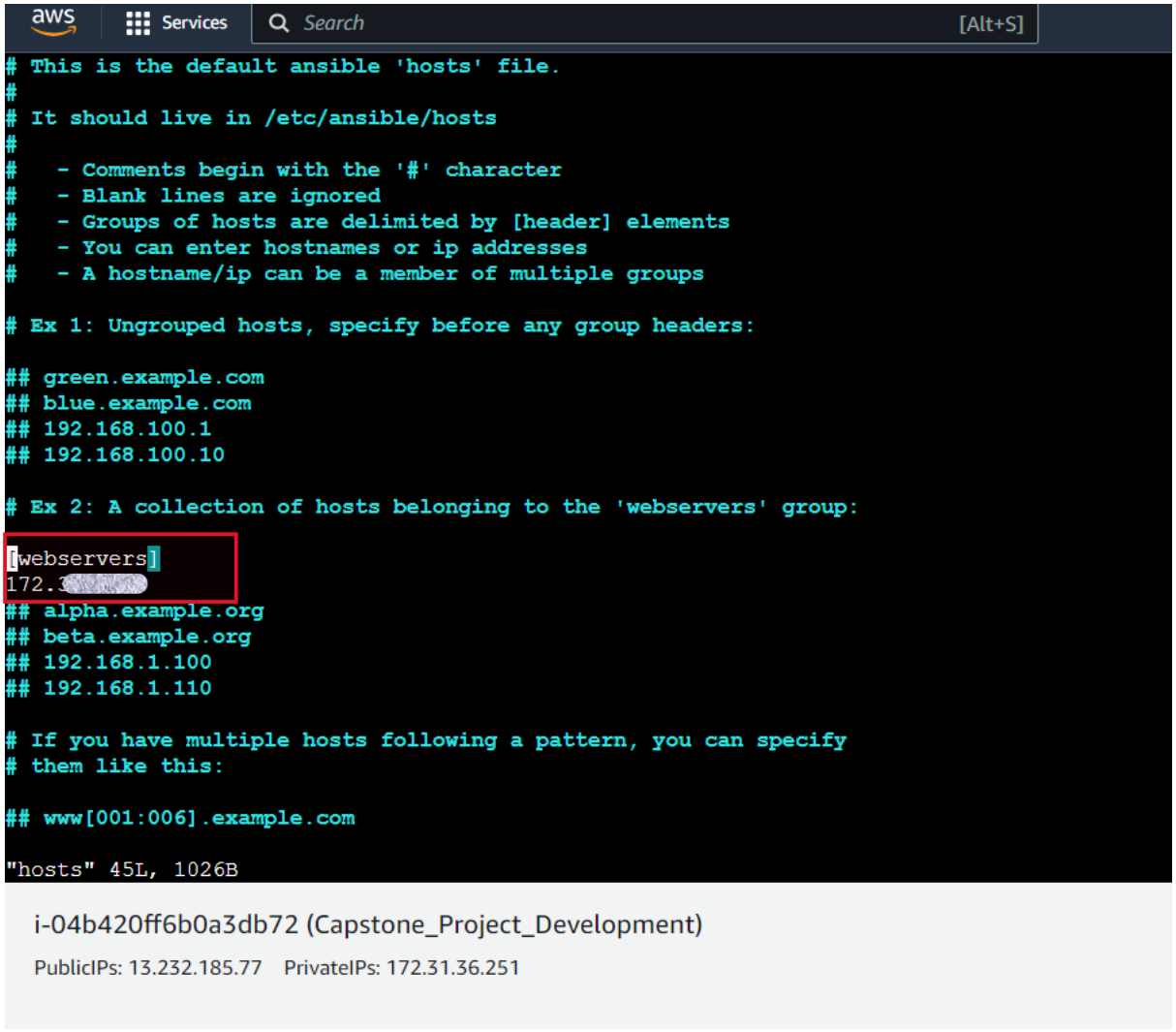
Do command: sudo su –

Cd /ect/ansible

```
root@ip-172-31-36-251:/etc# cd /etc
AWS Console Home
-251:/etc# cd ansible/
root@ip-172-31-36-251:/etc/ansible# ls
ansible.cfg  hosts  roles
root@ip-172-31-36-251:/etc/ansible#
```

Command:- Vi hosts

Insert the deployment server(deloyment_test_server) ip



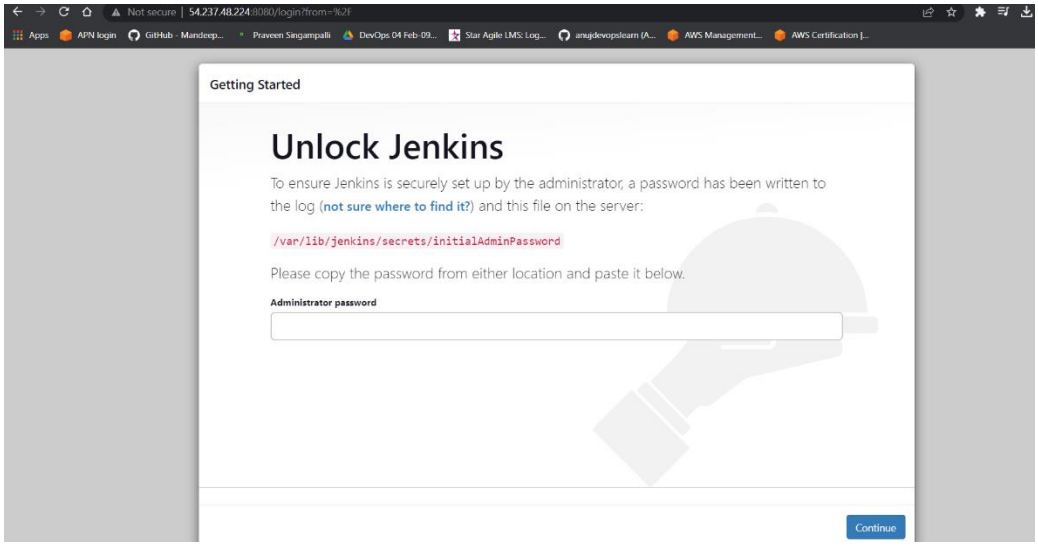
1.f Jenkins setup

Setting up the Jenkins (default running on 8080)

User- vams*****

Pas- pas*****

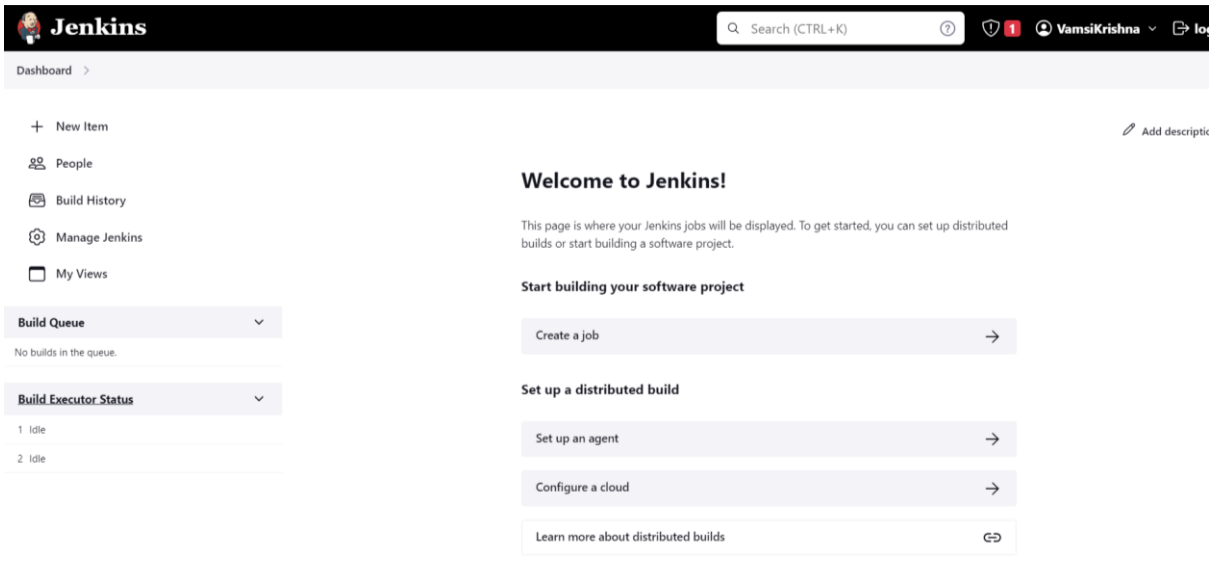
We can access Jenkins using http://IP :8080



Command :

Sudo cat /var/lib/Jenkins/secret/initialAdminPassword

After completing the initial setup



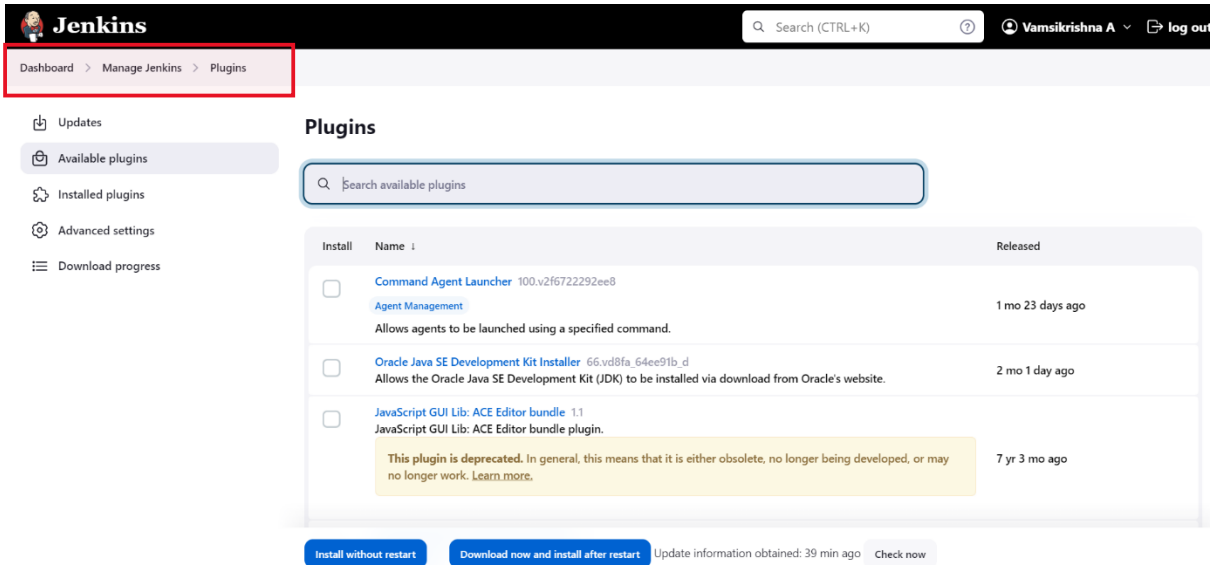
1.e

Plugin installation :

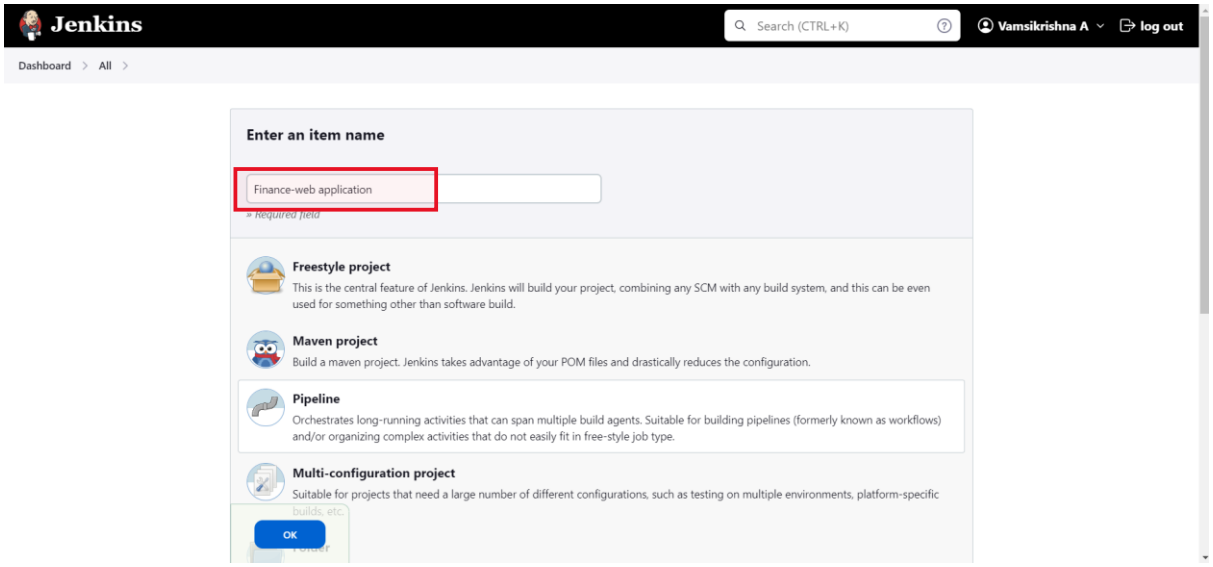
For integration of setup install the following plugins

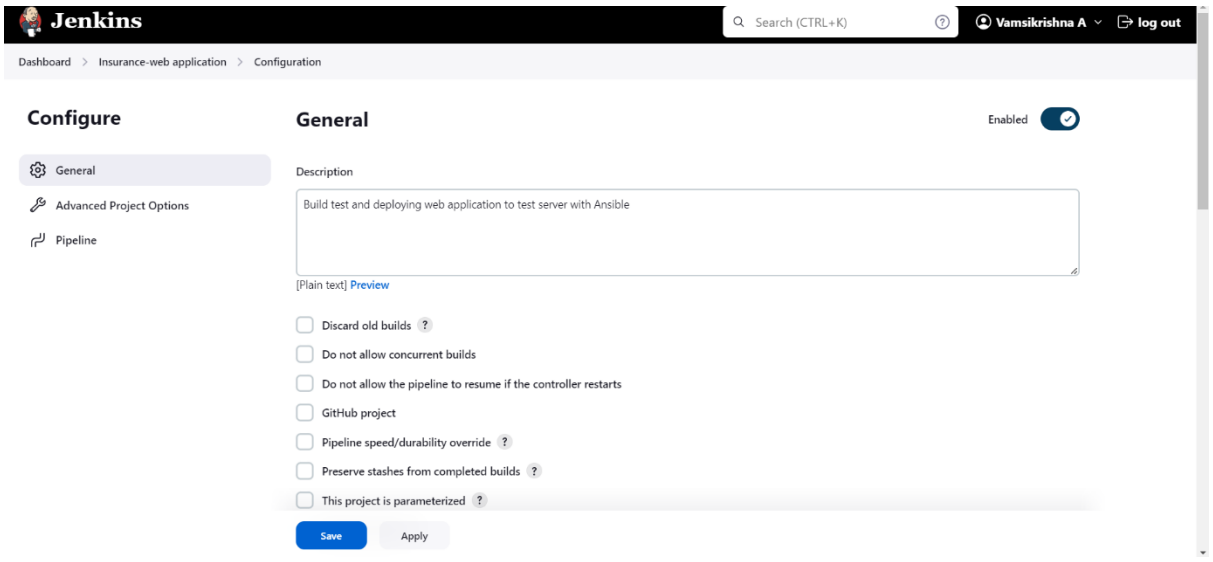
Path – dashboard->manage Jenkins-> plugins->available plugins

- Ansible
- Html publisher
- Docker
- Git
- Maven



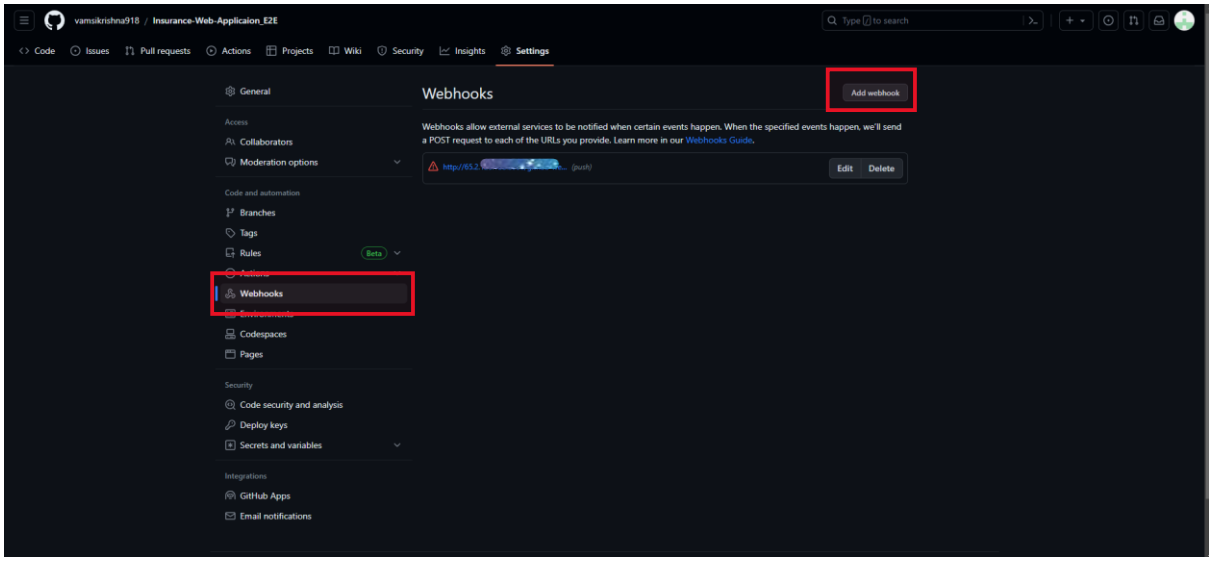
Step 2: Creating job



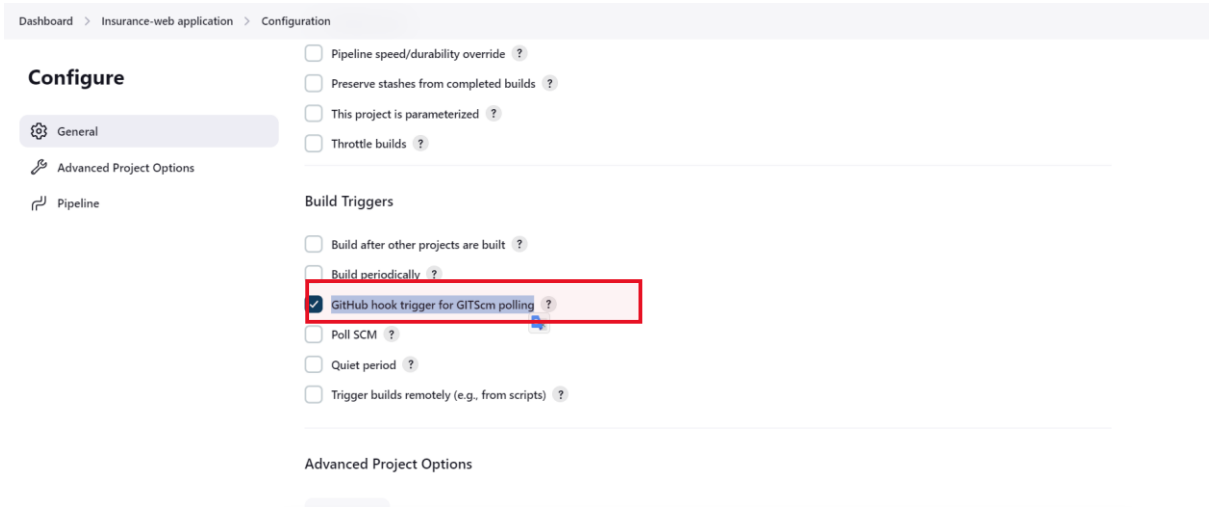


Setting up the webhook triggers- (The Jenkins will always watches the git repository if any commit happens it will automatically Runs the Jenkins job.)

GitHub -> repo-> settings-> webhooks-> add webhook -> add the URL of Jenkins



And in Jenkins check the **GitHub hook trigger for GITScm polling**



Create pipeline with the following stages:

1. Code checkout –
   checking out the code repo in **GitHub**
   use git:Git as sample step in syntax generator
2. Build –
   building the code, we have checkout with building tools like **maven**
3. publish the report-
   storing the generated reports in a directory
   use publishHTML:publishHTML reports as sample step in syntax generator
4. Image prune-
   docker image prune command allows you to clean up unused images
5. Build Docker image-
   building the docker image with Dockerfile
   #**Refer** Docker file in repository
6. Push Docker image to Hub-
   pushing the created image to docker registry (Docker Hub)
   use withCredentials:Bind credentials to variables as sample step in syntax generator

7. deploying to Test server –
   containerizing the image and deploying it into all hosts servers using Ansible.
   Use ansiblePlaybook:Invoke an ansible Playbook as sample step in syntax generator
   #Refer ansible-playbook file in repository

#**refer** the Jenkins file from git repo

Note- we have used Declarative pipeline syntax,

use **syntax generator/snippet generator** to generate the syntax as mentioned in steps.

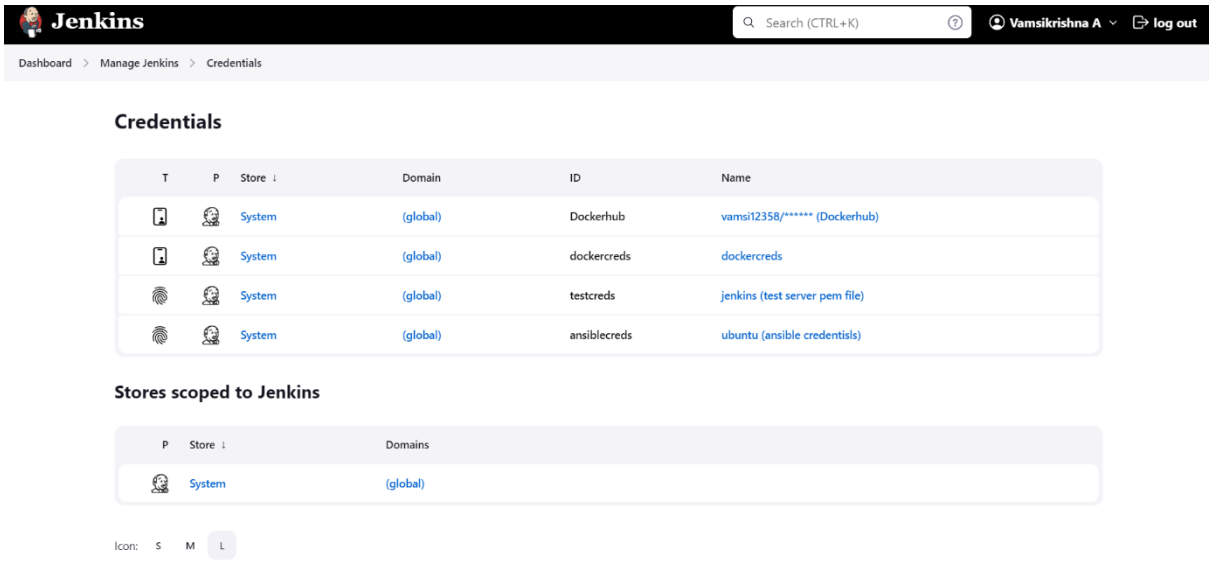
```
pipeline{
   agent any

   stages{
      stage("Code checkout"){
         steps{
            git branch: 'main', url: 'https://github.com/vamsikrishna918/Finance-Web-Application-E2E'          }
      }
       stage("Build"){
         steps{
            echo "****building with maven****"
            sh '''mvn clean package '''

         }
      }
      stage("publish the report")
      {
         steps{
```

```
        echo "generating test reports"
        publishHTML([allowMissing: false, alwaysLinkToLastBuild: false, keepAll: false, reportDir: '/var/lib/jenkins/workspace/insureme project/target/surefire-reports', reportFiles: 'index.html', reportName: 'HTML Report', reportTitles: '', useWrapperFileDirectly: true])
      }
    }
    stage("Image prune"){
      steps{
        echo "****deleting the previous images***"
        sh ' docker image prune -af '

      }
    }


    stage("Build Docker image"){
      steps{
        script {
         echo "****Creating Docker image****"
         sh 'docker build -t vamsi12358/insureme .'
         sh 'docker tag vamsi12358/insureme vamsi12358/insuremeapp:v7'
        }
      }
    }
    stage("Push Docker image to Hub"){
      steps{
```

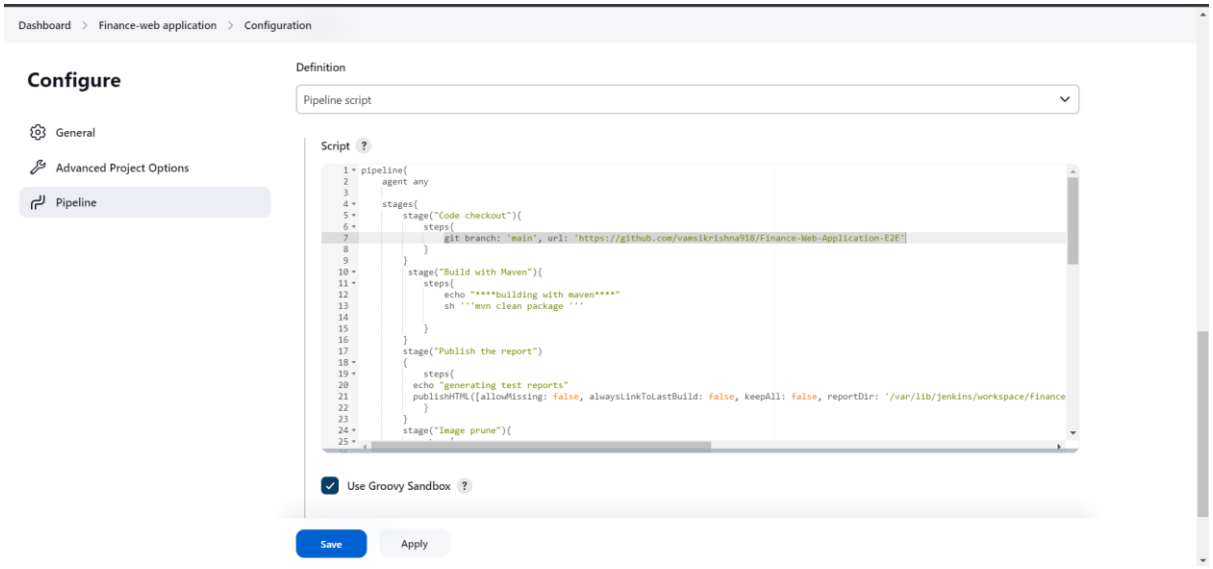
```
            script {
             echo "****Pushing Docker image to Hub****"

             withCredentials([string(credentialsId: 'dockercreds', variable:
'dockerhubpwd')]) {
              sh "docker login -u vamsi12358 -p ${dockerhubpwd} docker.io"
              sh 'docker push vamsi12358/insuremeapp:v7'
             }

            }
          }
        }
         stage("deploying to Test server"){
          steps{
            script {
             echo "****Deploying Application to Test server****"
             ansiblePlaybook become: true, credentialsId: 'ansiblecreds',
disableHostKeyChecking: true, inventory: '/etc/ansible/hosts', playbook:
'ansible-playbook.yml'
            }
          }
        }


    }
}
```

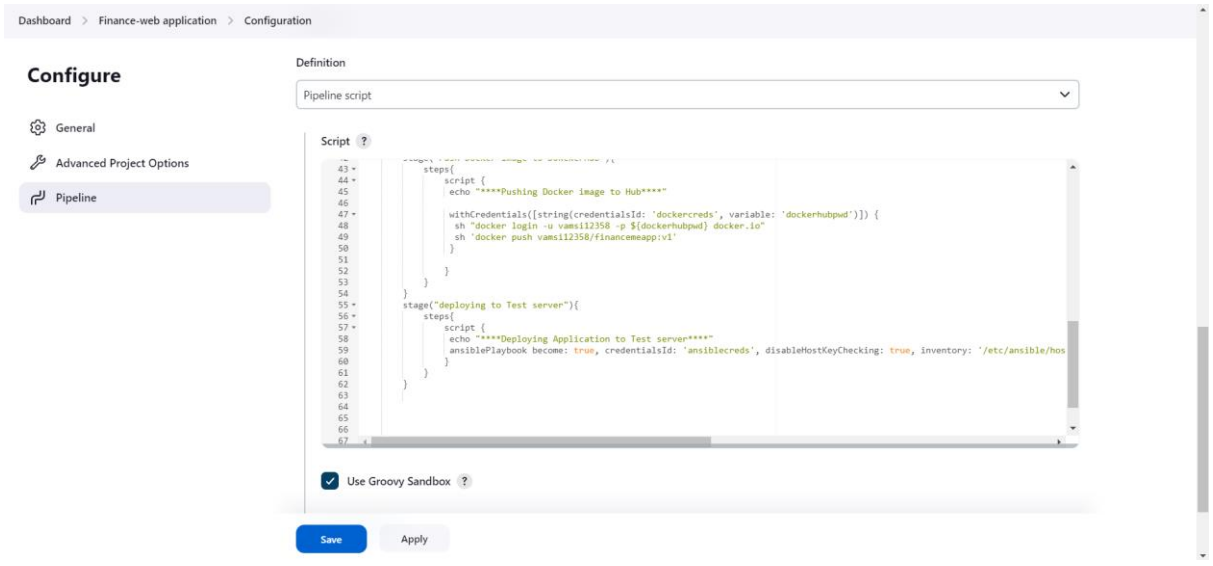
Credentials setup

- Docker
- Deploying server(deployment_test_server) ssh key setup

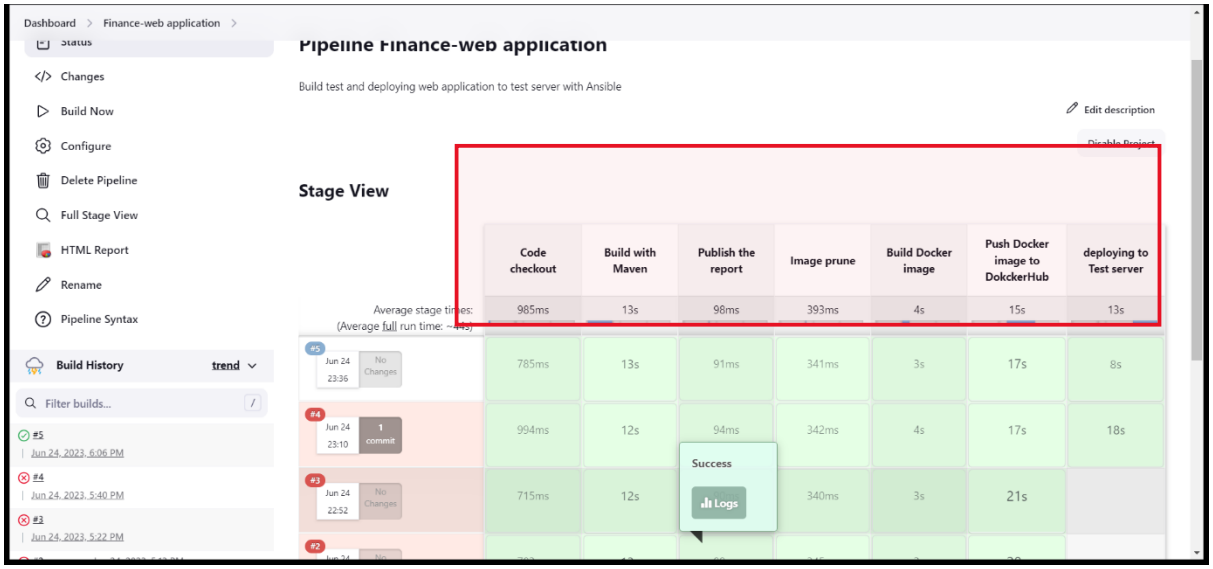


Jenkins pipeline - deploying to test server job



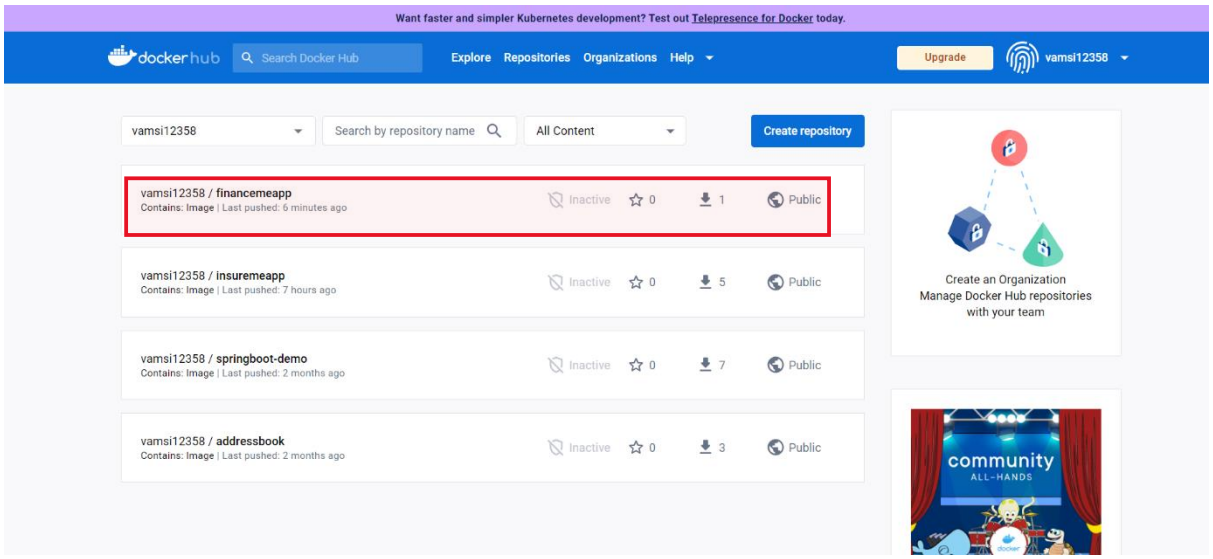


Build the job-



Docker hub-

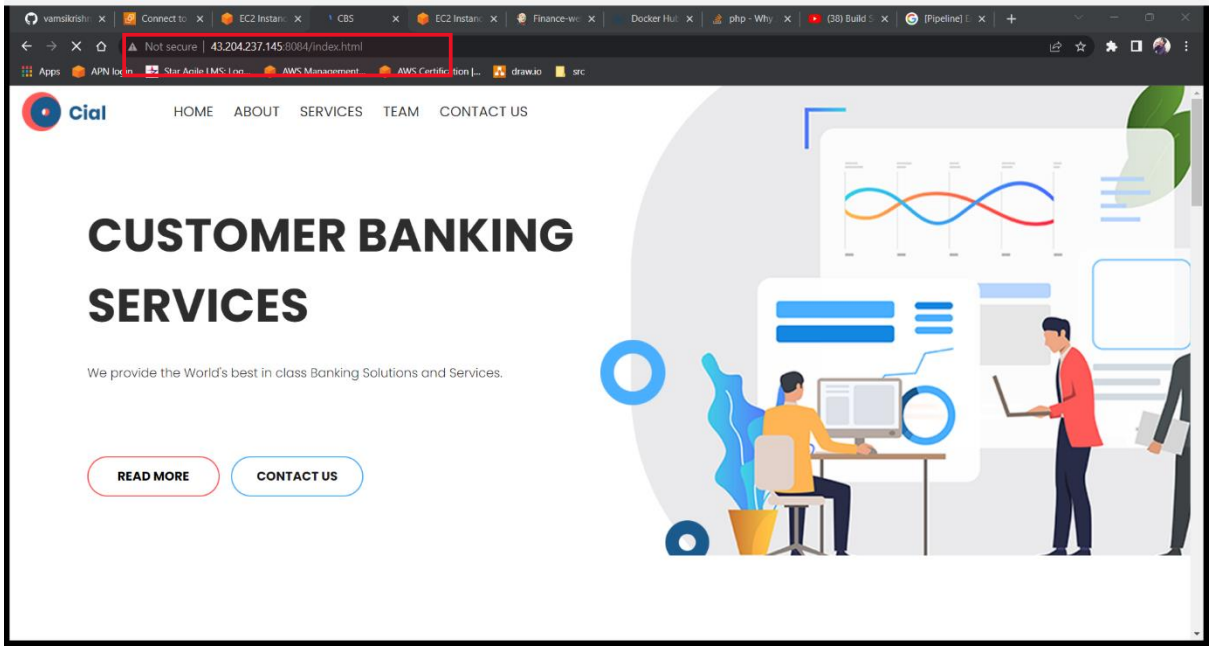
https://hub.docker.com/

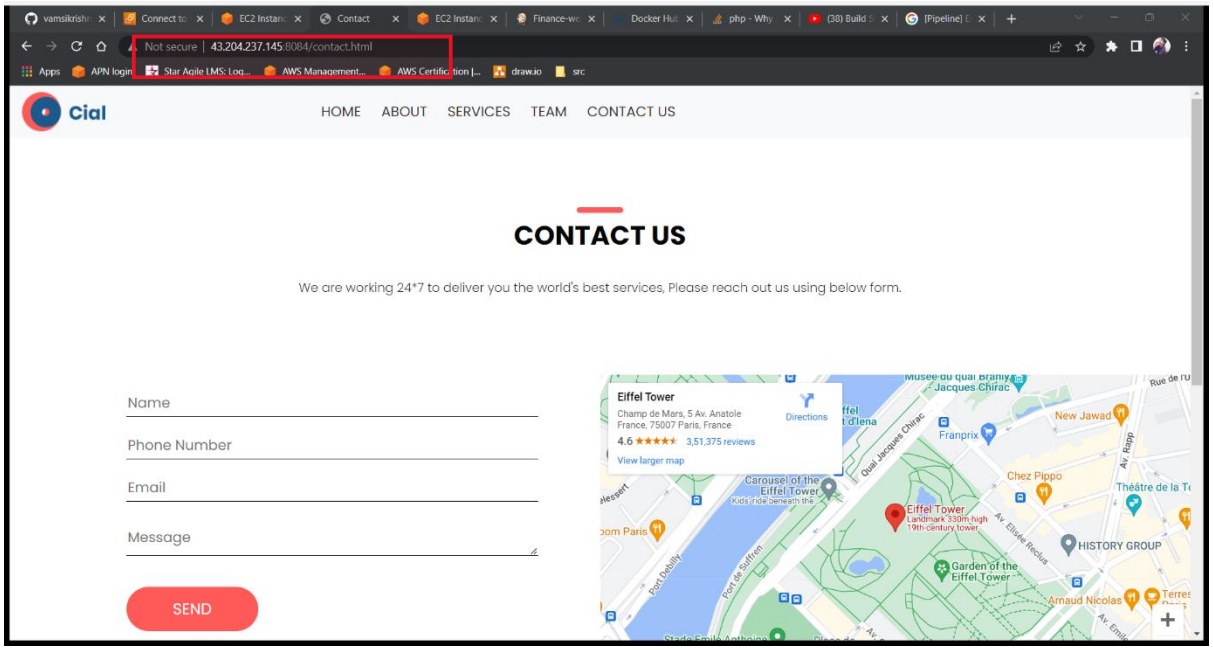


Deployed Application on test server

Accessing the web-applcation on test server

Deployment_test_server public ip:8084/





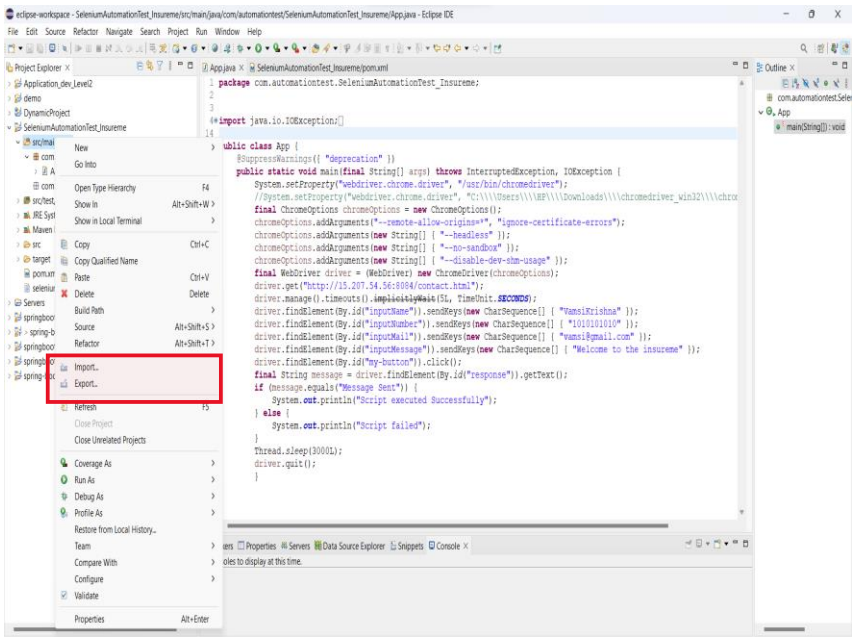
Test -Automation with selenium

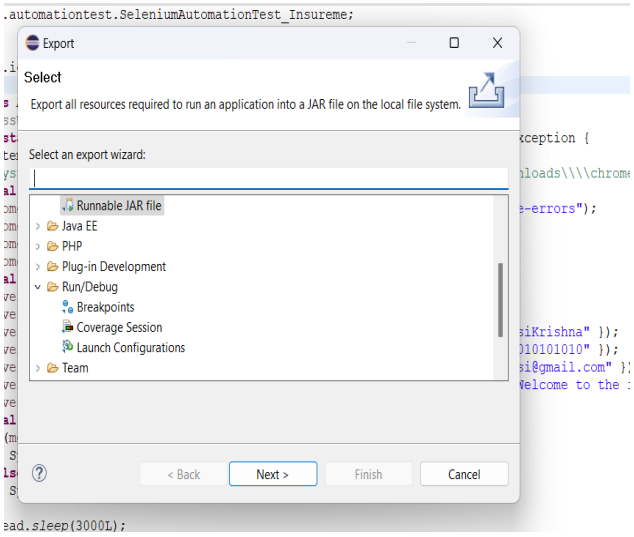
Refer - https://github.com/vamsikrishna918/Selenium-TestAutomation-Insureme

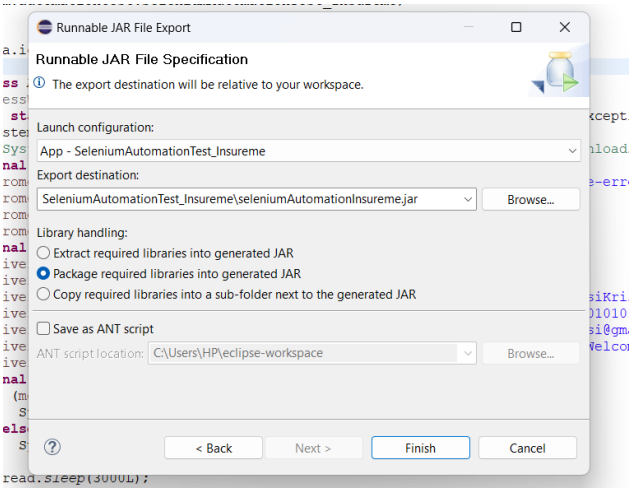
App.java × SeleniumAutomationTest_Insureme/pom.xml

```
4  import java.io.IOException;
5
6  import org.apache.commons.io.FileUtils;
7  import org.openqa.selenium.By;
8  import org.openqa.selenium.OutputType;
9  import org.openqa.selenium.TakesScreenshot;
10 import org.openqa.selenium.WebDriver;
11 import org.openqa.selenium.chrome.ChromeDriver;
12 import org.openqa.selenium.chrome.ChromeOptions;
13 import io.github.bonigarcia.wdm.WebDriverManager;
14
15 /**
16  * finance-me application test
17  *
18  */
19 public class App
20 {
21     public static void main( String[] args ) throws InterruptedException, IOException
22     {
23         //System.setProperty("webdriver.chrome.driver", "D:\\chrome-driver\\chromedriver.exe");
24         System.setProperty("webdriver.chrome.driver", "/usr/bin/chromedriver");
25
26         WebDriverManager.chromedriver().setup();
27         ChromeOptions chromeOptions = new ChromeOptions();
28
29         //To the scripts without opening GUI and need to process at bg use --headless
30         chromeOptions.addArguments("--headless");
31         chromeOptions.addArguments("--disable-dev-shm-usage");
32         chromeOptions.addArguments("--remote-allow-origins=*");
33         WebDriver driver = new ChromeDriver(chromeOptions);
34
35         System.out.println( "Selenium test scripts executed started ...." );
36
37
38         System.out.println("Opening finance-me Browser");
39         driver.get("http://3.110.221.50:8084/contact.html") ;
```

Export to Runnable jar







Push the jar to git hub, we can crate a job and use in pipeline

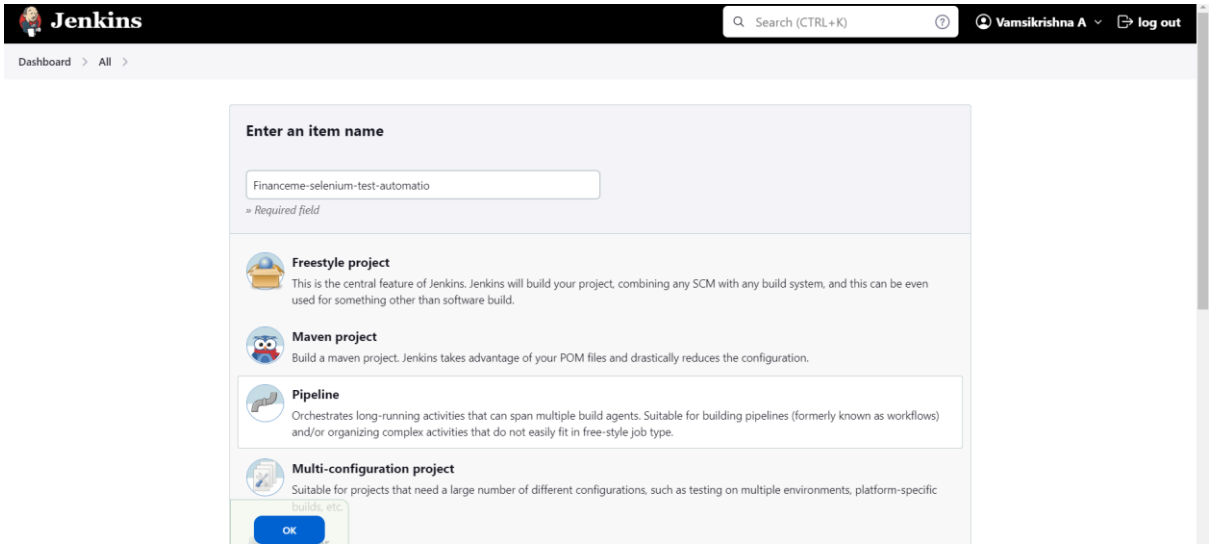
Installing the Chrome driver in our ubuntu instance
(Capstone_Project_Development)

On Capstone_Project_Development instance-

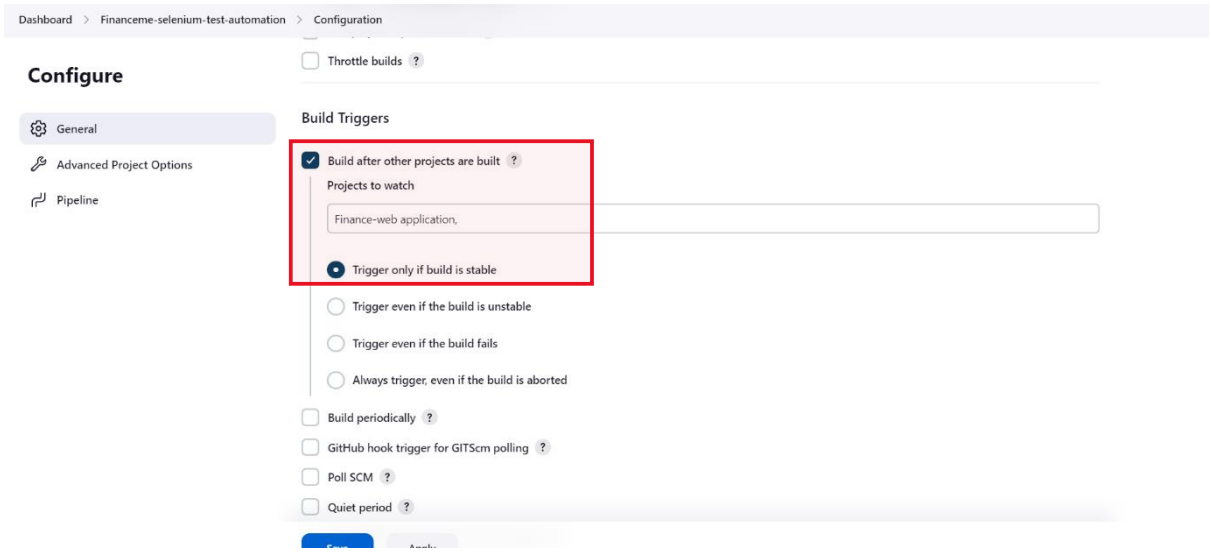
Run the below commands and install.

- sudo apt install -y unzip xvfb libxi6 libgconf-2-4
- sudo curl -sS -o - https://dl-ssl.google.com/linux/linux_signing_key.pub | apt-key add
- sudo bash -c "echo 'deb [arch=amd64] http://dl.google.com/linux/chrome/deb/ stable main' >> /etc/apt/sources.list.d/google-chrome.list"
- sudo apt -y update
- sudo apt -y install google-chrome-stable
- google-chrome --version
- wget https://chromedriver.storage.googleapis.com/114.0.5735.90/chromedriver_linux64.zip
- ls
- unzip chromedriver_linux64.zip
- ls
- sudo mv chromedriver /usr/bin/chromedriver
- sudo chown root:root /usr/bin/chromedriver
- sudo chmod +x /usr/bin/chromedriver
- which chromedriver

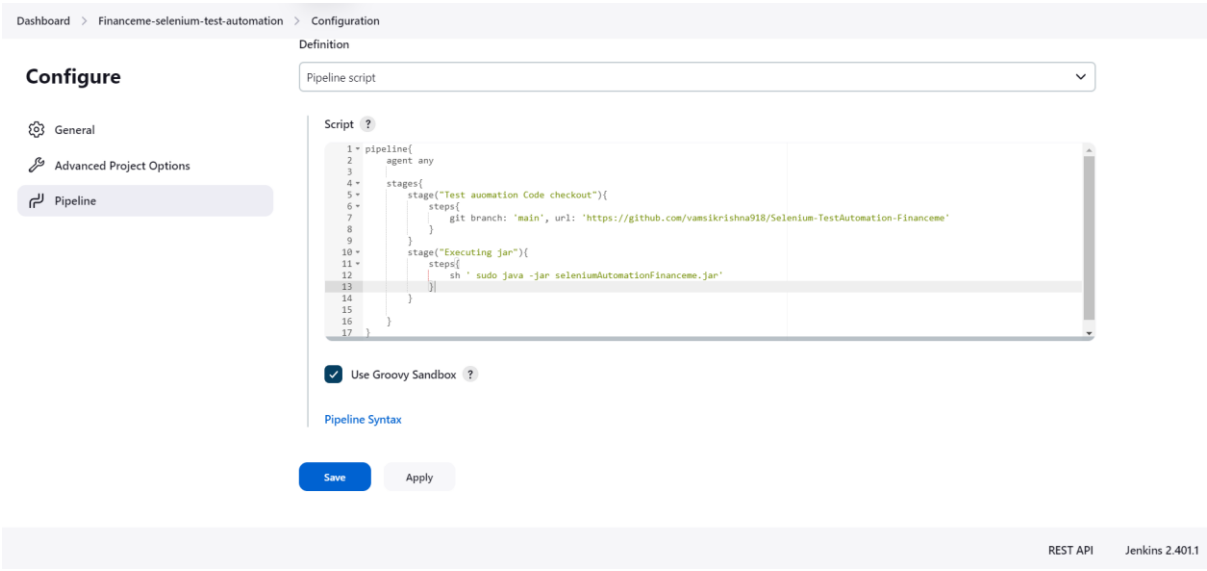
Creating a job for automation test-

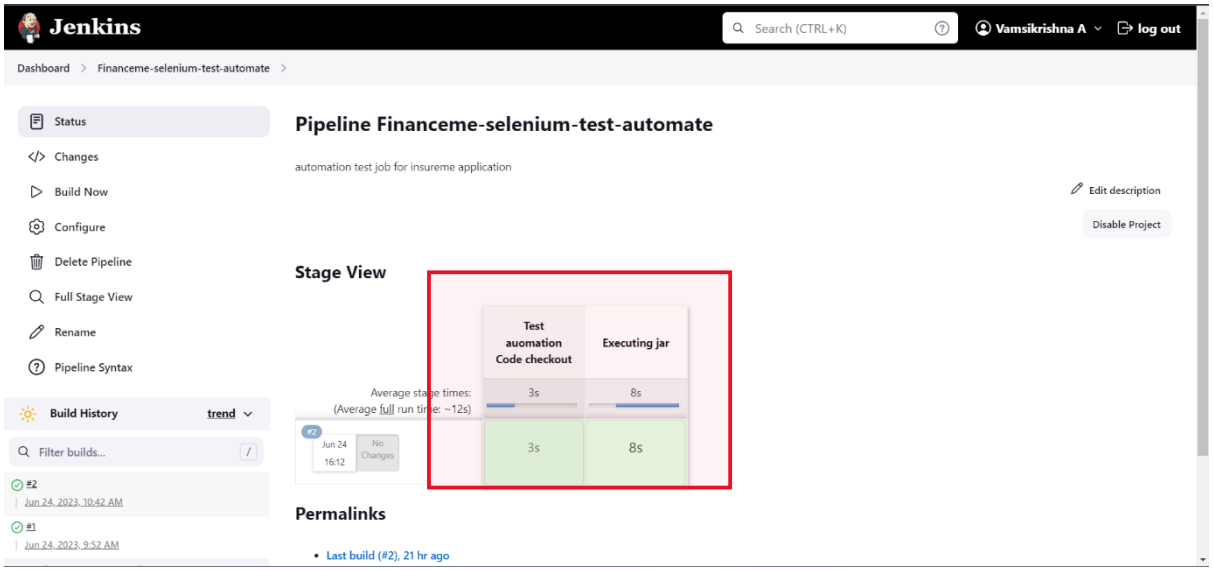


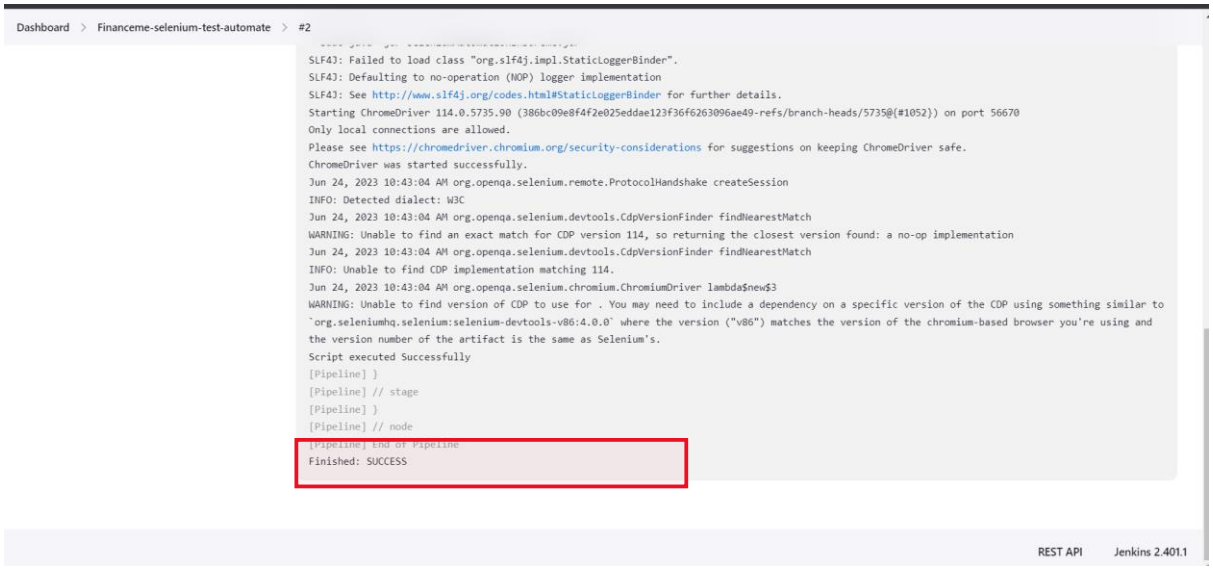
Setup the build trigger, which the automation job builds after the Finance me application deployed on test server(deployment_test_server).



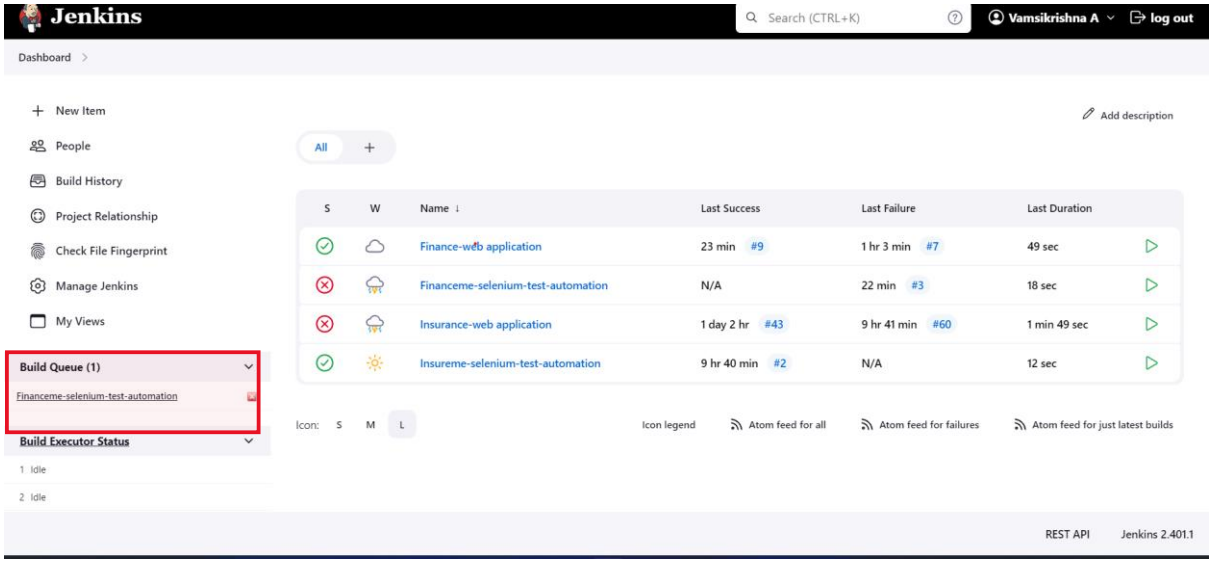
```
pipeline{
    agent any


    stages{
        stage("Test auomation Code checkout"){
            steps{
                git branch: 'main', url: 'https://github.com/vamsikrishna918/Selenium-TestAutomation-Financeme'
            }
        }
        stage("Executing jar"){
            steps{
                sh ' sudo java -jar seleniumAutomationFinanceme.jar'
            }
        }


    }
}
```



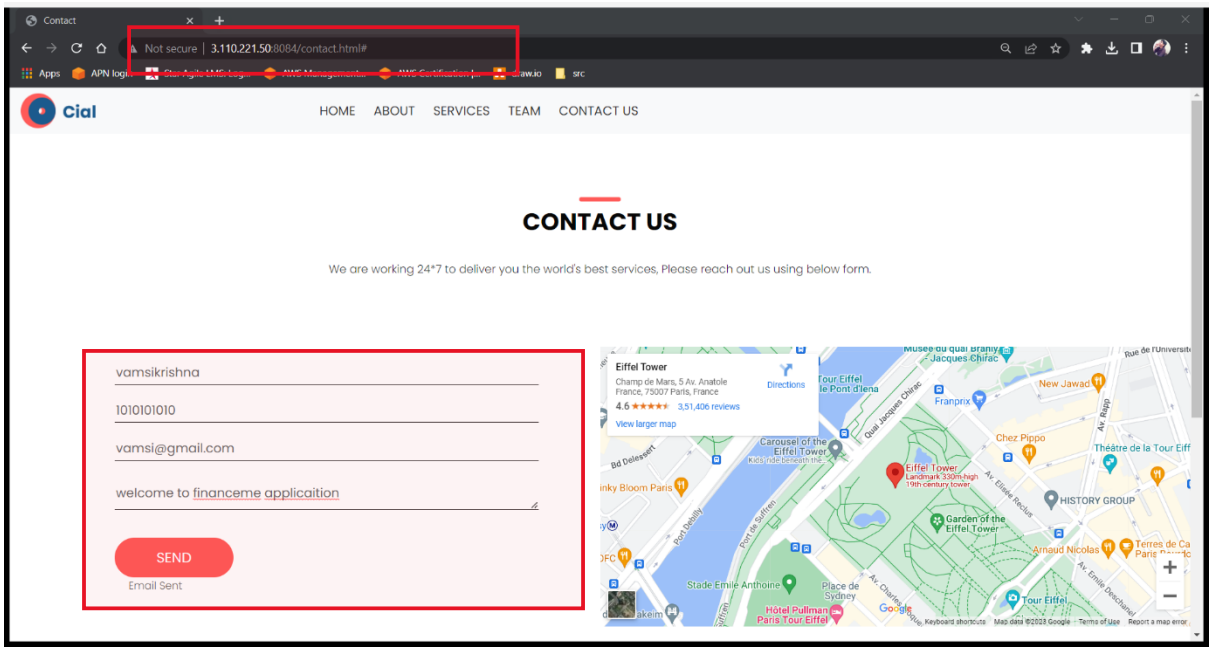

Jenkins
Search (CTRL+K)
Vamsikrishna A
log out
Dashboard
Financeme-selenium-test-automate
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Rename
Pipeline Syntax
Build History
trend
Filter builds...
#2
Jun 24, 2023, 10:42 AM
#1
Jun 24, 2023, 9:52 AM
Pipeline Financeme-selenium-test-automate
automation test job for insureme application
Edit description
Disable Project
Stage View
Test auomation Code checkout
Executing jar
Average stage times:
(Average full run time: ~12s)
3s
8s
Jun 24
16:12
No Changes
Permalinks
Last build (#2), 21 hr ago


Dashboard
Financeme-selenium-test-automate
#2
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Starting ChromeDriver 114.0.5735.90 (386bc09e8f4f2e025eddae123f36f6263096ae49-refs/branch-heads/5735@{#1052}) on port 56670
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Jun 24, 2023 10:43:04 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
Jun 24, 2023 10:43:04 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 114, so returning the closest version found: a no-op implementation
Jun 24, 2023 10:43:04 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Unable to find CDP implementation matching 114.
Jun 24, 2023 10:43:04 AM org.openqa.selenium.chromium.ChromiumDriver lambdanew3
WARNING: Unable to find version of CDP to use for . You may need to include a dependency on a specific version of the CDP using something similar to
`org.seleniumhq.selenium:selenium-devtools-v86:4.0.0` where the version ("v86") matches the version of the chromium-based browser you're using and
the version number of the artifact is the same as Selenium's.
Script executed Successfully
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
REST API
Jenkins 2.401.1


Dashboard > Finance-web application > #10
TASK [updating apt] ***
changed: [172.31.3.8]
TASK [Install Docker] ***
changed: [172.31.3.8]
TASK [Start Docker Service] ***
changed: [172.31.3.8]
TASK [Deploy Docker Container] **
changed: [172.31.3.8]
PLAY RECAP **
172.31.3.8 : ok=5 changed=4 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Triggering a new build of Financeme-selenium-test-automation #4
Finished: SUCCESS
REST API
Jenkins 2.401.1


Jenkins
Search (CTRL+K)
Vamsikrishna A
log out
Dashboard
New Item
People
Build History
Project Relationship
Check File Fingerprint
Manage Jenkins
My Views
Build Queue (1)
Financeme-selenium-test-automation
Build Executor Status
1 Idle
2 Idle
Add description
All
S
W
Name
Last Success
Last Failure
Last Duration
Finance-web application
23 min #9
1 hr 3 min #7
49 sec
Financeme-selenium-test-automation
N/A
22 min #3
18 sec
Insurance-web application
1 day 2 hr #43
9 hr 41 min #60
1 min 49 sec
Insureme-selenium-test-automation
9 hr 40 min #2
N/A
12 sec
Icon: S M L
Icon legend
Atom feed for all
Atom feed for failures
Atom feed for just latest builds
REST API
Jenkins 2.401.1

Sample executed on local machine-

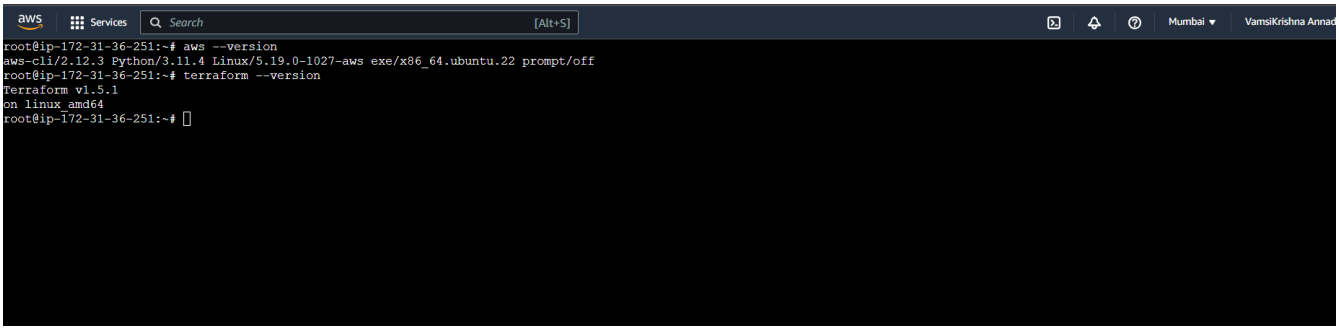


Terraform –

Creating a EC2 instance with terraform in IAC as a production server and deploying the Finance application on the server.

Installing the **terraform** and **AWS config** on the instance (Capstone_Project_Development)

- aws cli - https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html
- terraform - https://developer.hashicorp.com/terraform/tutorials/aws-get-started/install-cli

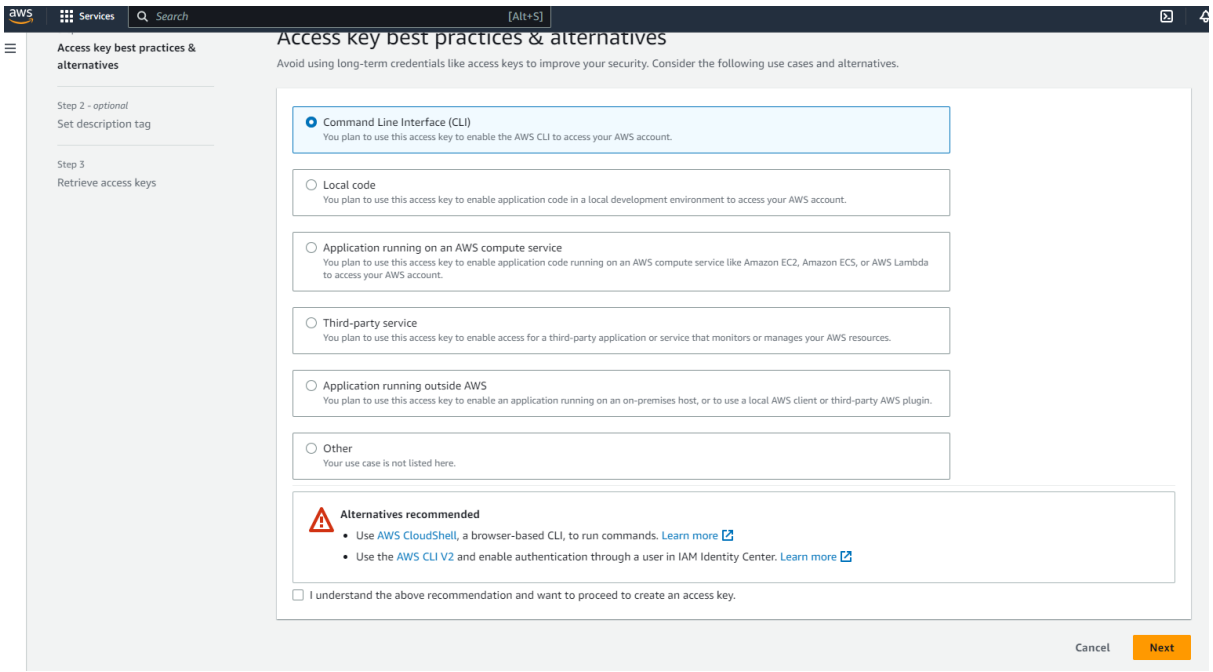


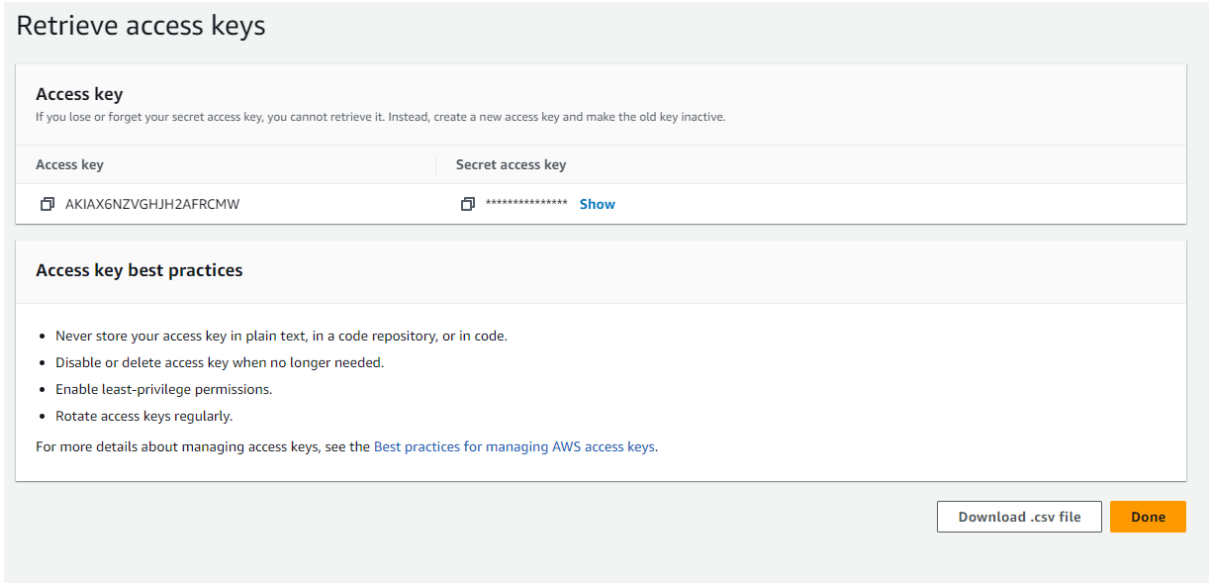
Configuring the AWS

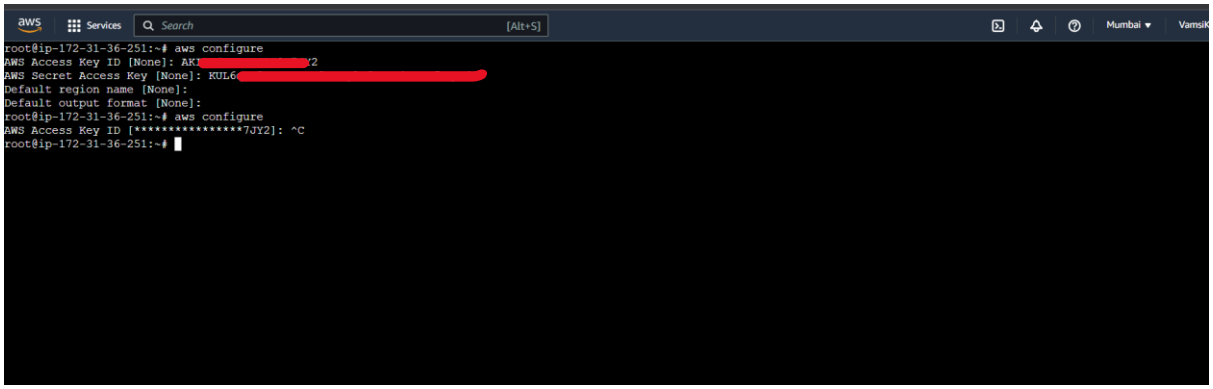
Setup aws user :

IAM-> Create a use and attach administrator access privilege roles

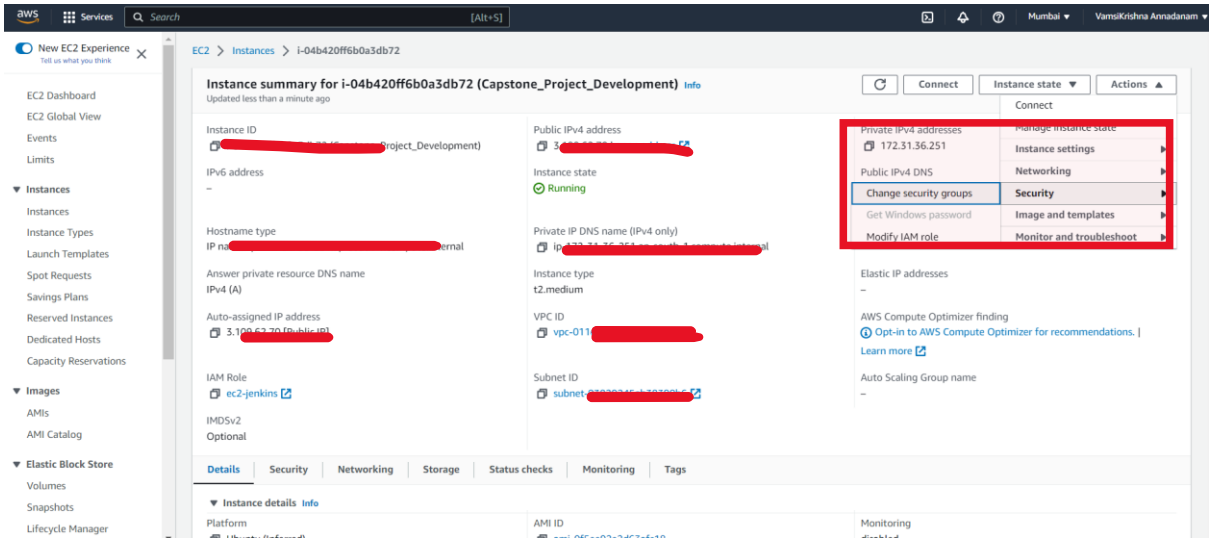
And setup the Credentials keys





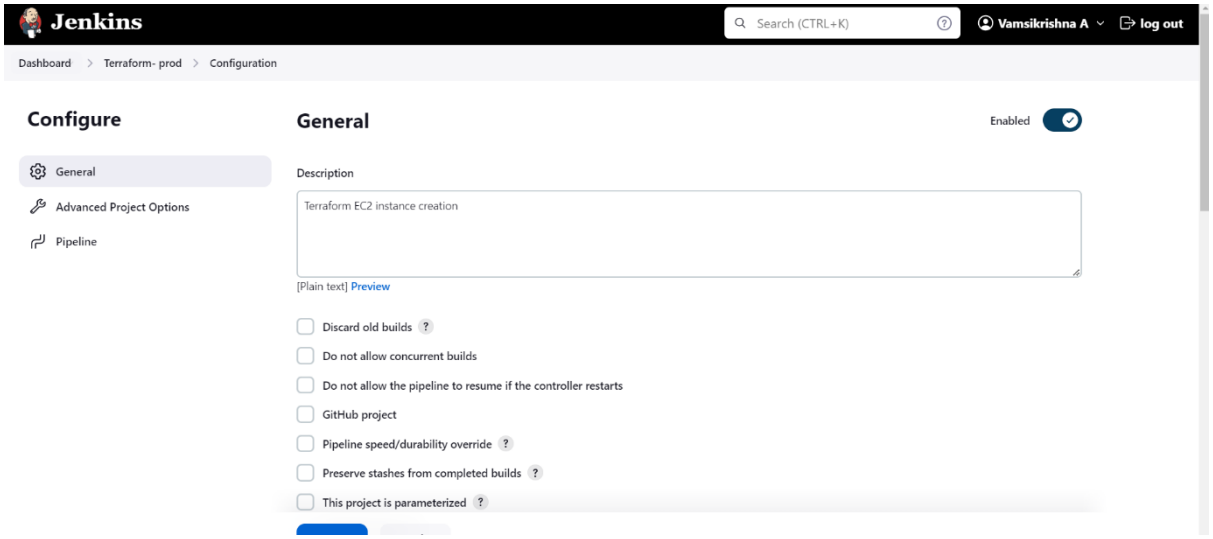


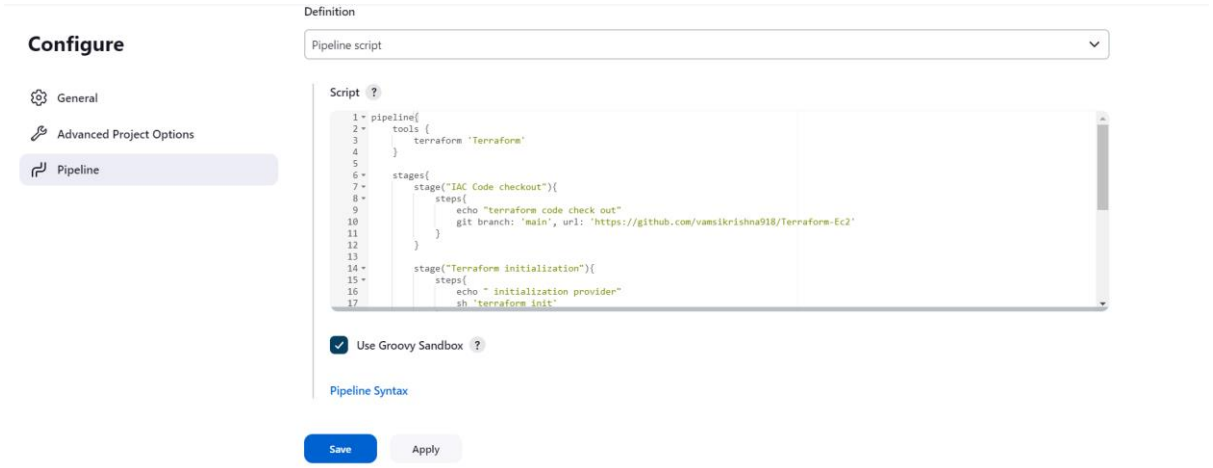
Create a IAM role for EC2 service and tag the role to Capstone_Project_Development



Terraform Job creation-

Refer - https://github.com/vamsikrishna918/Terraform-Ec2 for the script

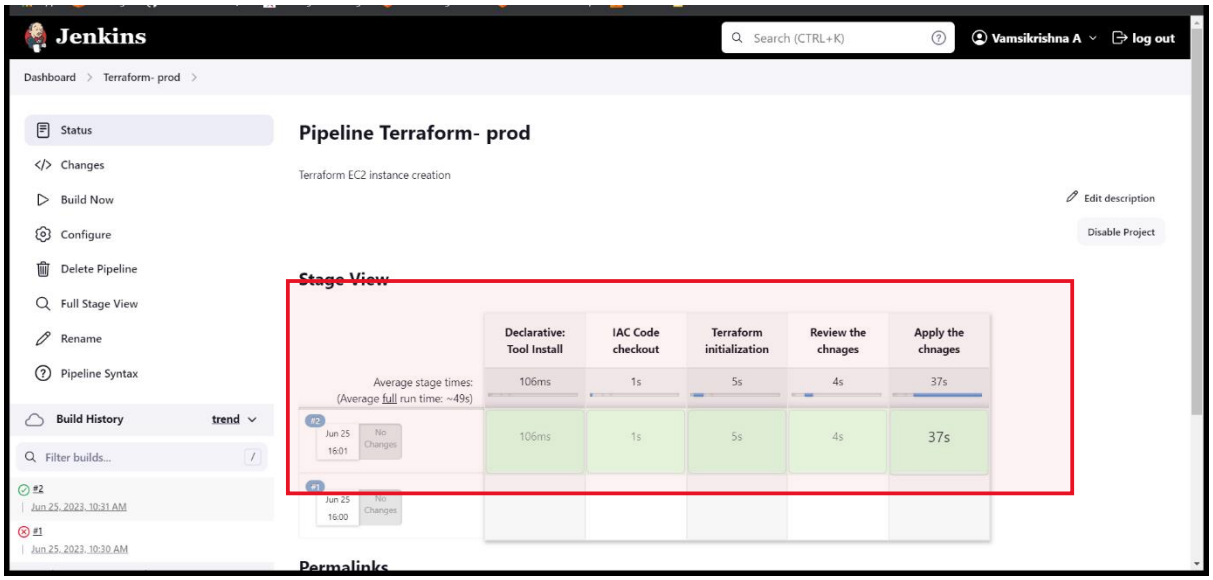




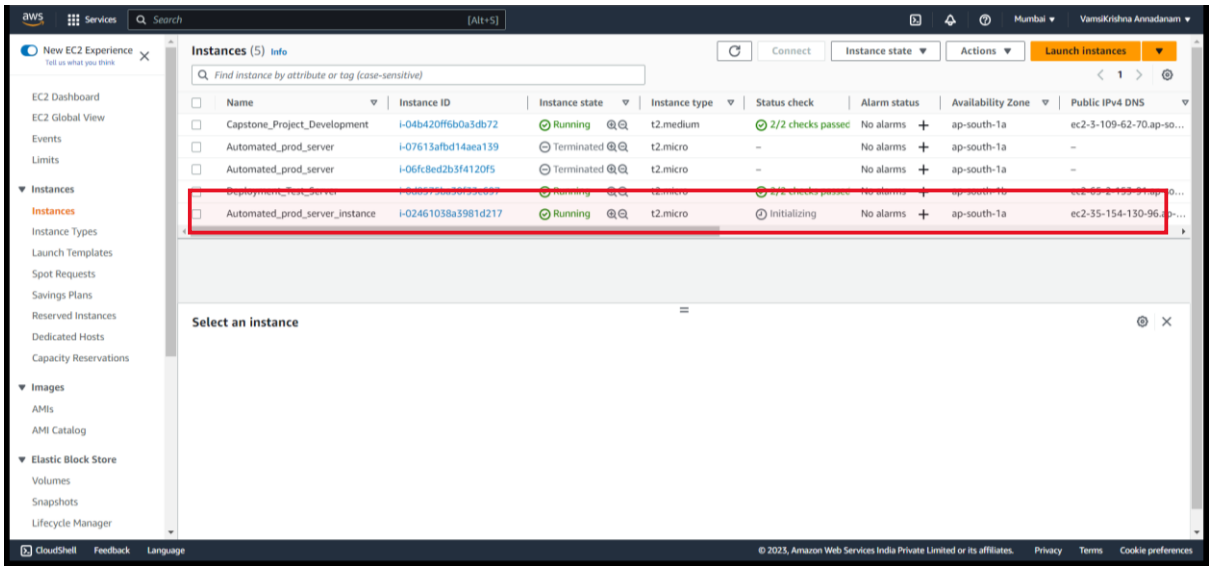

```
pipeline{
        agent any
  tools {
     terraform 'Terraform'
  }
```

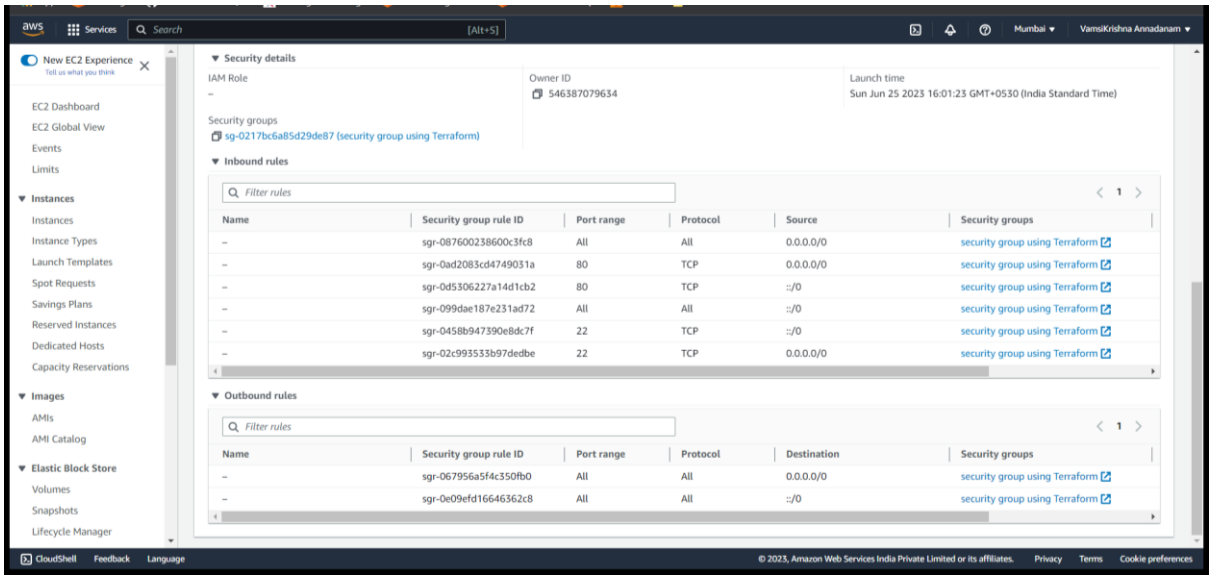
```
stages{
   stage("IAC Code checkout"){
      steps{
         echo "terraform code check out"
         git branch: 'main', url: 'https://github.com/vamsikrishna918/Terraform-Ec2'
      }
   }

   stage("Terraform initialization"){
      steps{
         echo " initialization provider"
         sh 'terraform init'
      }
   }

    stage("Review the chnages"){
      steps{
         echo " Review the infrastructure changes"
         sh 'terraform plan'
      }
   }
    stage("Apply the chnages"){
      steps{
         echo " Creating the infrastructure"
         sh 'terraform apply -auto-approve'
      }
   }

}
```

}


Jenkins
Search (CTRL+K)
Vamsikrishna A
log out
Dashboard > Terraform- prod >
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Rename
Pipeline Syntax
Build History
trend
Filter builds...
#2
Jun 25, 2023, 10:31 AM
#1
Jun 25, 2023, 10:30 AM
Pipeline Terraform- prod
Terraform EC2 instance creation
Edit description
Disable Project
Stage View
Declarative: Tool Install
IAC Code checkout
Terraform initialization
Review the chnages
Apply the chnages
Average stage times:
(Average full run time: ~49s)
106ms
1s
5s
4s
37s
Jun 25
16:01
No Changes
Jun 25
16:00
No Changes
Permalinks

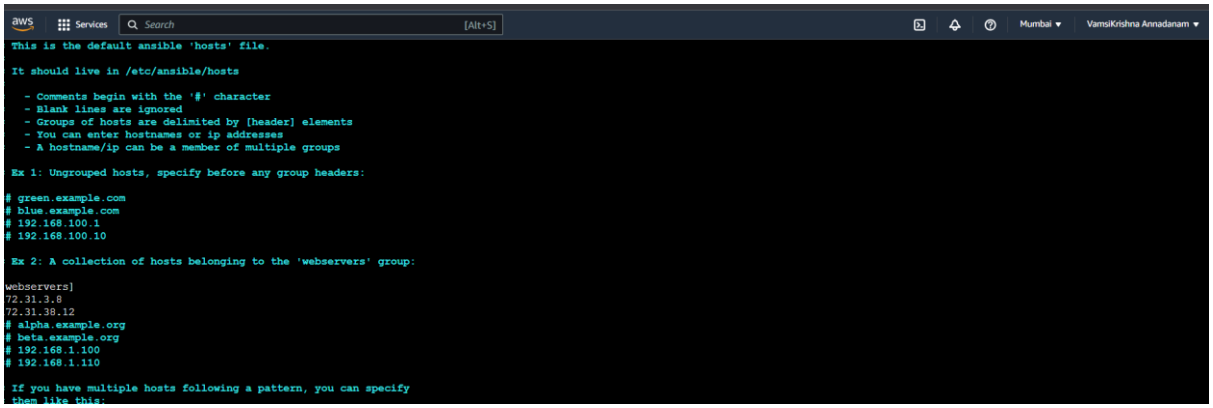

Dashboard > Terraform- prod > #2
= "vpc-011eb6390249210aa"
Plan: 2 to add, 0 to change, 0 to destroy.
aws_security_group.TF_SG: Creating...
aws_security_group.TF_SG: Creation complete after 2s [id=sg-0217bc6a85d29de87]
aws_instance.web: Creating...
aws_instance.web: Still creating... [10s elapsed]
aws_instance.web: Still creating... [20s elapsed]
aws_instance.web: Still creating... [30s elapsed]
aws_instance.web: Creation complete after 31s [id=i-02461038a3981d217]
Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
REST API
Jenkins 2.401.1

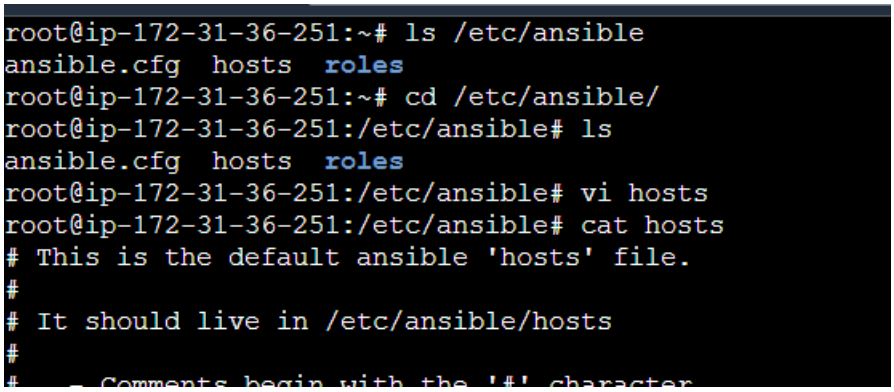




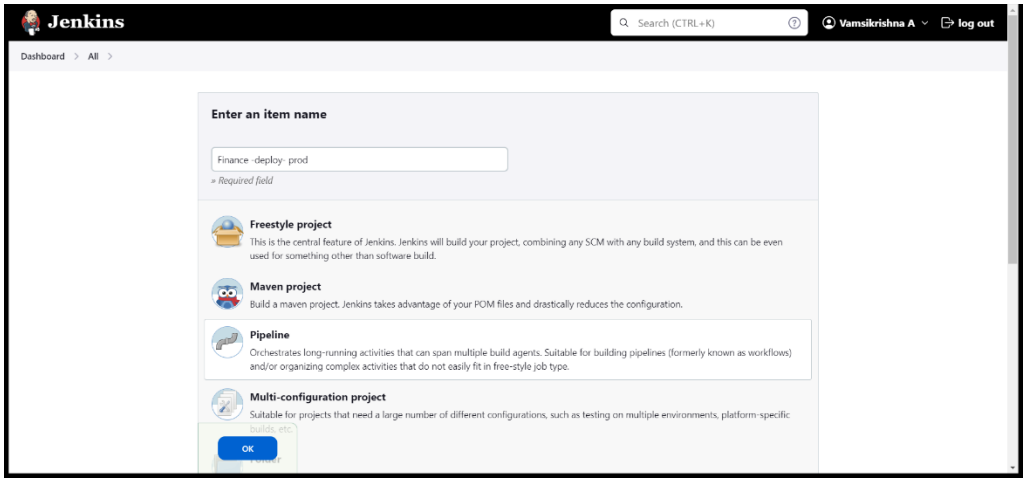
Setting up the Hosts

Insert the private ip of the automated_prod_server in host file on the (Capstone deployment server) as below

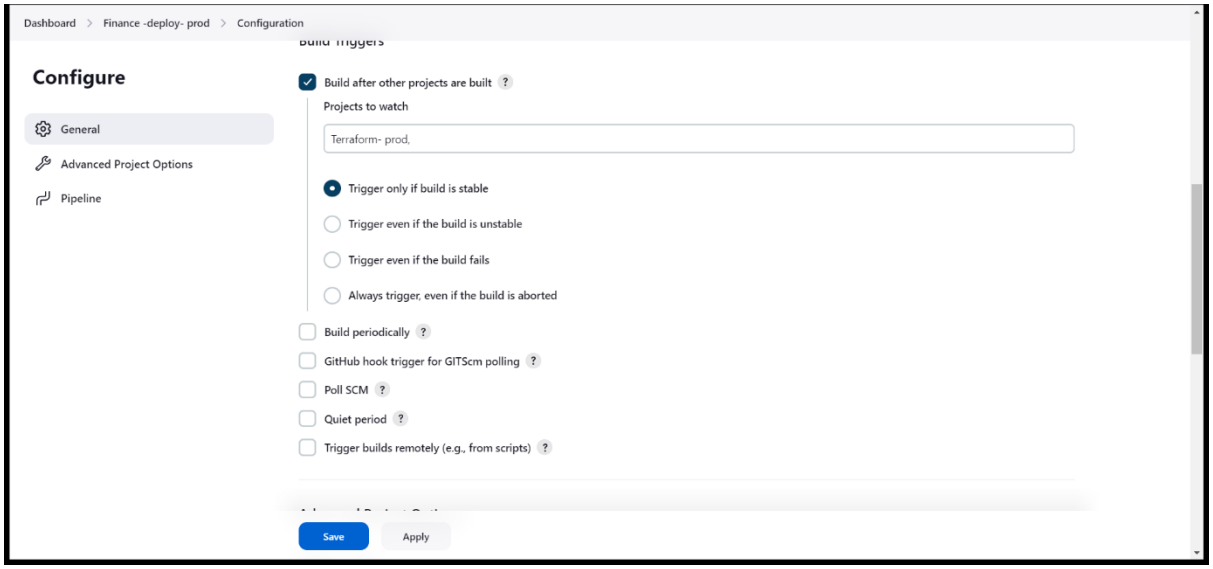


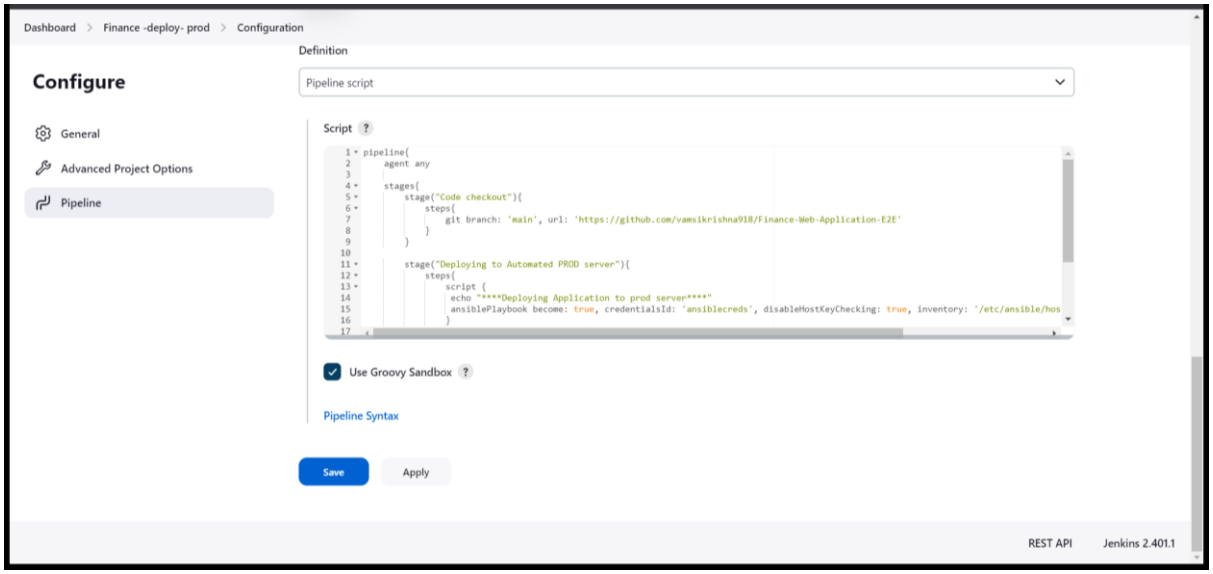


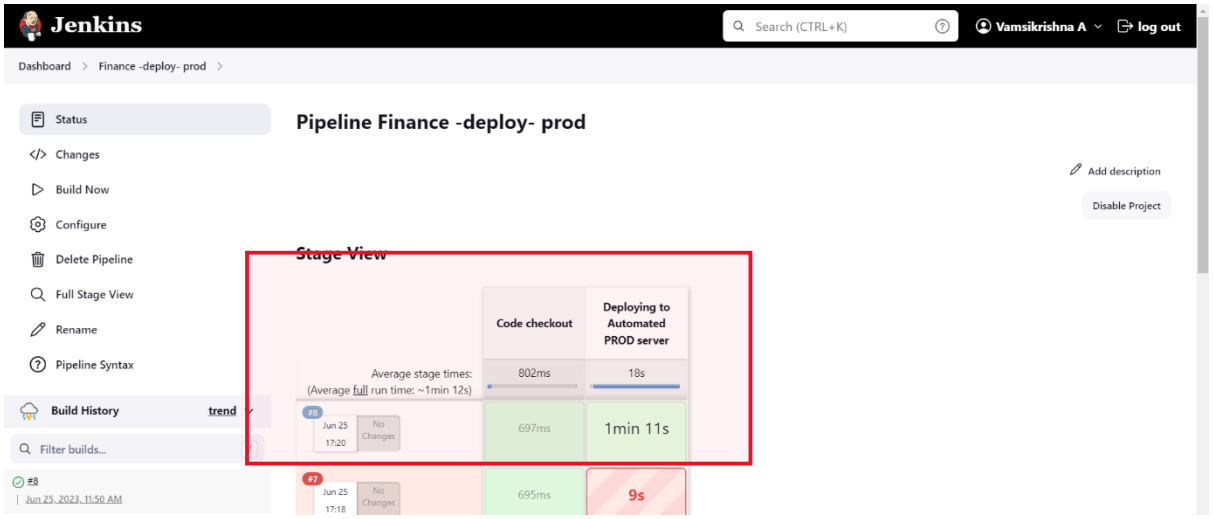
Finance me application deploy to automated prod server-

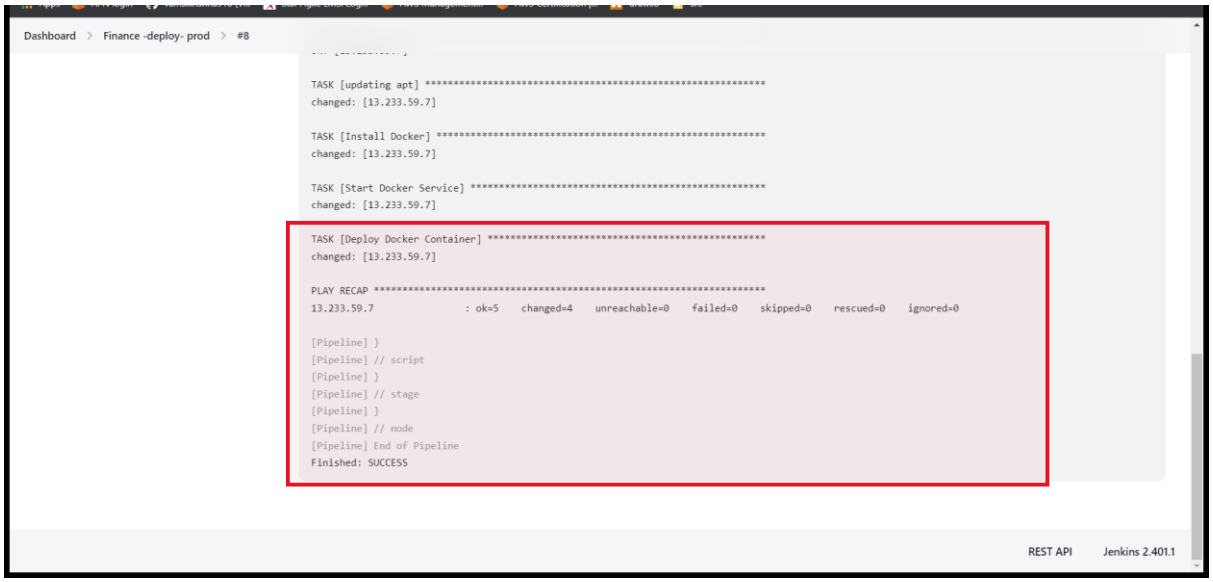


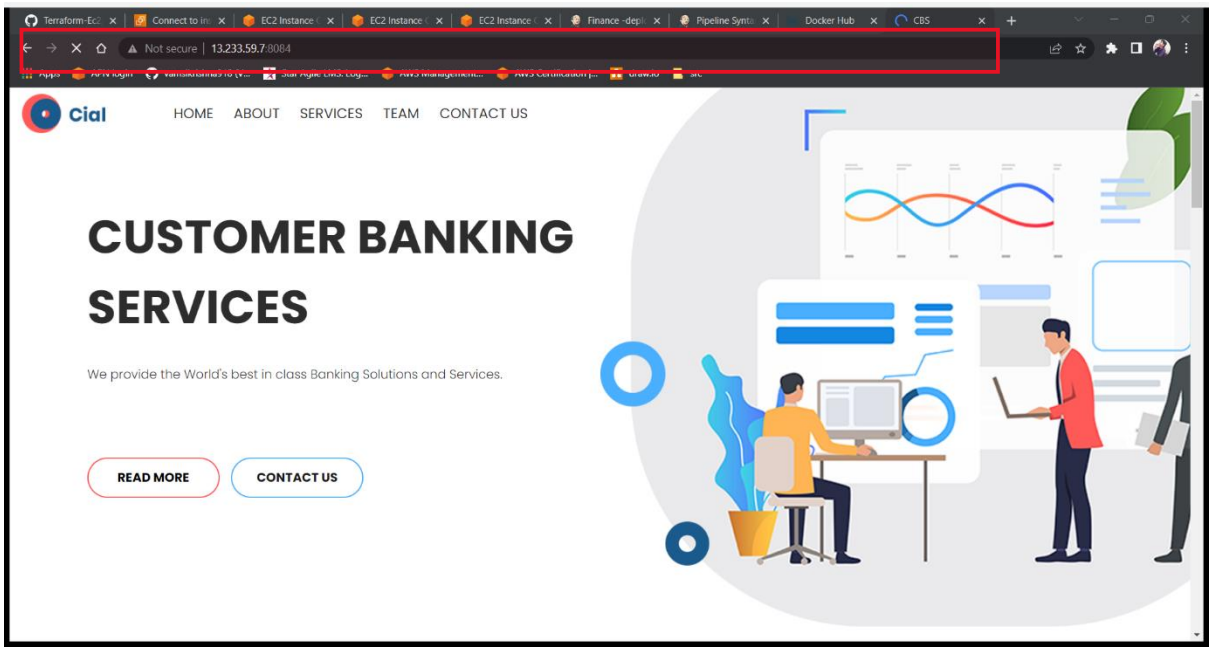
Build trigger setup – job builds after the automated instance creation


Dashboard > Finance -deploy- prod > Configuration
Configure
General
Advanced Project Options
Pipeline
Build after other projects are built
Projects to watch
Terraform- prod,
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Always trigger, even if the build is aborted
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Quiet period
Trigger builds remotely (e.g., from scripts)
Save
Apply


Dashboard > Finance -deploy- prod > Configuration
Configure
General
Advanced Project Options
Pipeline
Definition
Pipeline script
Script
pipeline{
agent any
stages{
stage("Code checkout"){
steps{
git branch: 'main', url: 'https://github.com/vamsikrishna918/Finance-Web-Application-E2E'
}
}
stage("Deploying to Automated PROD server"){
steps{
script {
echo "****Deploying Application to prod server****"
ansiblePlaybook become: true, credentialsId: 'ansiblecreds', disableHostKeyChecking: true, inventory: '/etc/ansible/hos
}
Use Groovy Sandbox
Pipeline Syntax
Save
Apply
REST API
Jenkins 2.401.1

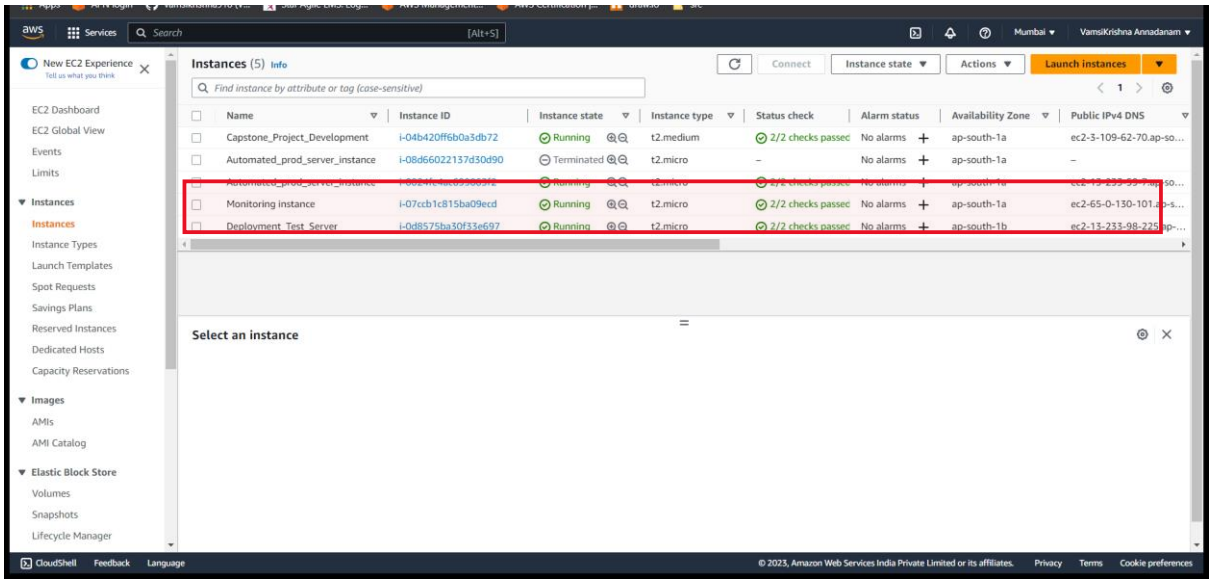

Jenkins
Search (CTRL+K)
Vamsikrishna A
log out
Dashboard
Finance -deploy- prod
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Rename
Pipeline Syntax
Build History
trend
Filter builds...
#8
Jun 25, 2023, 11:50 AM
Pipeline Finance -deploy- prod
Add description
Disable Project
Stage View
Code checkout
Deploying to Automated PROD server
Average stage times:
(Average full run time: ~1min 12s)
802ms
18s
#8
Jun 25
17:20
No Changes
697ms
1min 11s
#7
Jun 25
17:18
No Changes
695ms
9s


Dashboard
Finance -deploy- prod
#8
TASK [updating apt] **
changed: [13.233.59.7]
TASK [Install Docker] **
changed: [13.233.59.7]
TASK [Start Docker Service] **
changed: [13.233.59.7]
TASK [Deploy Docker Container] ***
changed: [13.233.59.7]
PLAY RECAP ***
13.233.59.7 : ok=5 changed=4 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
REST API
Jenkins 2.401.1


Not secure | 13.233.59.7:8084
Cial
HOME
ABOUT
SERVICES
TEAM
CONTACT US
CUSTOMER BANKING SERVICES
We provide the World's best in class Banking Solutions and Services.
READ MORE
CONTACT US

Monitoring the servers-

Creating a instance to monitor both test and prod server.

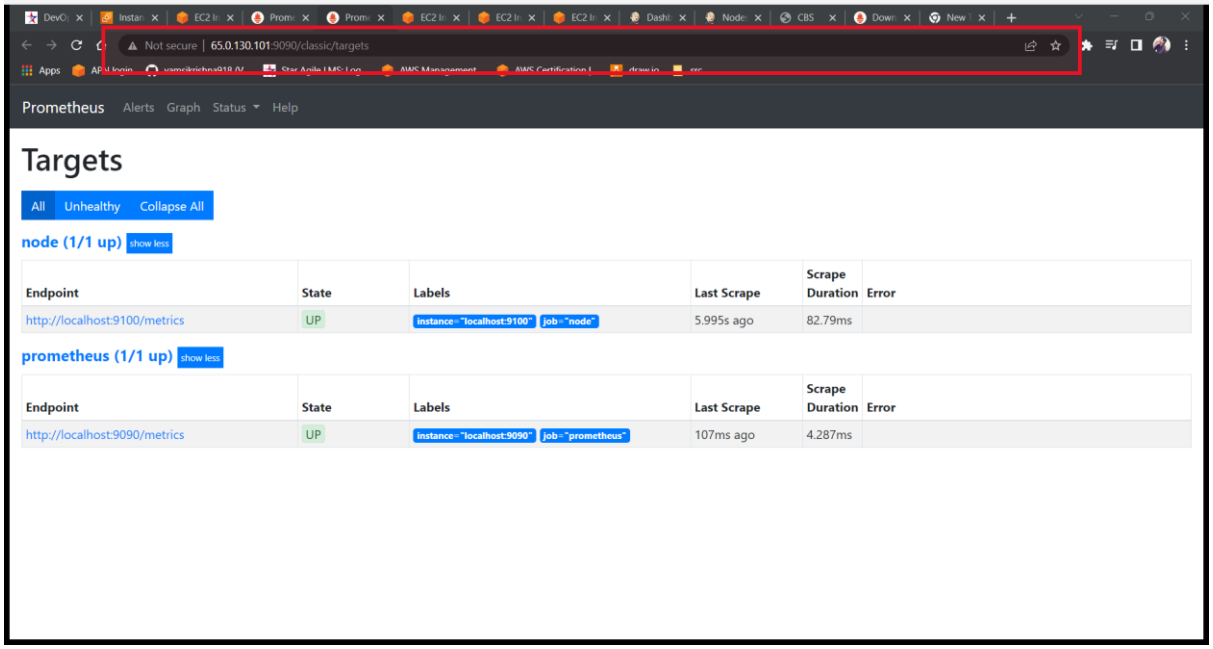


Installing Prometheus –

Command – apt update && apt install Prometheus

Default port of Prometheus – **9090**

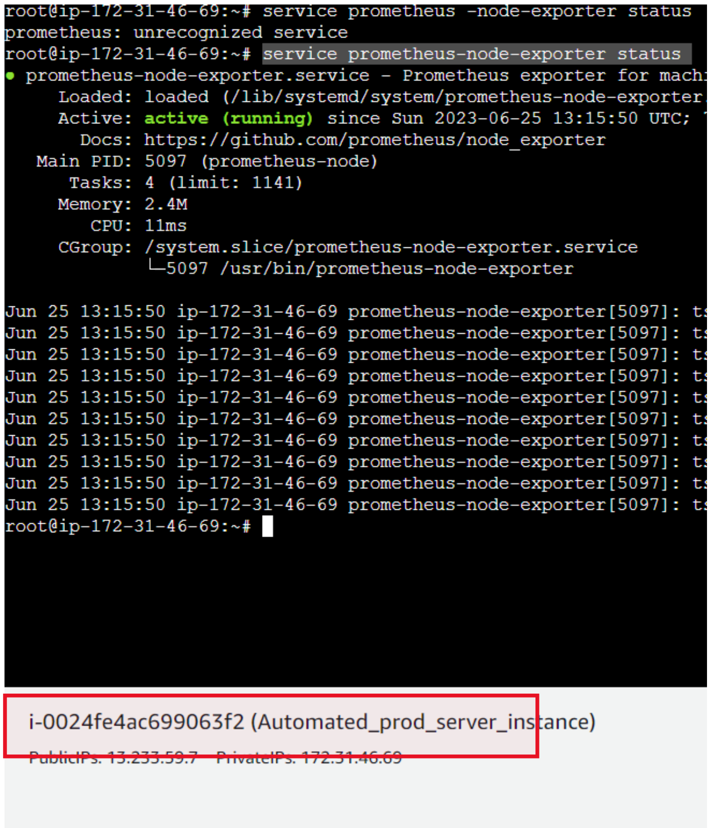
```
root@ip-172-31-37-231:/# service prometheus status
● prometheus.service - Monitoring system and time series database
     Loaded: loaded (/lib/systemd/system/prometheus.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2023-06-25 12:56:59 UTC; 3min 43s ago
       Docs: https://prometheus.io/docs/introduction/overview/
             man:prometheus(1)
   Main PID: 3601 (prometheus)
      Tasks: 7 (limit: 1141)
     Memory: 23.8M
        CPU: 439ms
     CGroup: /system.slice/prometheus.service
             └─3601 /usr/bin/prometheus

Jun 25 12:56:59 ip-172-31-37-231 prometheus[3601]: ts=2023-06-25T12:56:59.882Z caller=head.go:481 level=info component=tsdb msg="Replaying on-disk memory mappable chunks if any"
Jun 25 12:56:59 ip-172-31-37-231 prometheus[3601]: ts=2023-06-25T12:56:59.882Z caller=head.go:515 level=info component=tsdb msg="On-disk memory mappable chunks replay completed" dur
Jun 25 12:56:59 ip-172-31-37-231 prometheus[3601]: ts=2023-06-25T12:56:59.882Z caller=head.go:521 level=info component=tsdb msg="Replaying WAL, this may take a while"
Jun 25 12:56:59 ip-172-31-37-231 prometheus[3601]: ts=2023-06-25T12:56:59.882Z caller=head.go:592 level=info component=tsdb msg="WAL segment loaded" segment=0 maxSegment=0
Jun 25 12:56:59 ip-172-31-37-231 prometheus[3601]: ts=2023-06-25T12:56:59.883Z caller=head.go:598 level=info component=tsdb msg="WAL replay completed" checkpoint_replay_duration=48.
Jun 25 12:56:59 ip-172-31-37-231 prometheus[3601]: ts=2023-06-25T12:56:59.884Z caller=main.go:850 level=info fs_type=EXT4_SUPER_MAGIC
Jun 25 12:56:59 ip-172-31-37-231 prometheus[3601]: ts=2023-06-25T12:56:59.885Z caller=main.go:853 level=info msg="TSDB started"
Jun 25 12:56:59 ip-172-31-37-231 prometheus[3601]: ts=2023-06-25T12:56:59.885Z caller=main.go:980 level=info msg="Loading configuration file" filename=/etc/prometheus/prometheus.yml
Jun 25 12:56:59 ip-172-31-37-231 prometheus[3601]: ts=2023-06-25T12:56:59.892Z caller=main.go:1017 level=info msg="Completed loading of configuration file" filename=/etc/prometheus/
Jun 25 12:56:59 ip-172-31-37-231 prometheus[3601]: ts=2023-06-25T12:56:59.892Z caller=main.go:795 level=info msg="Server is ready to receive web requests."
```



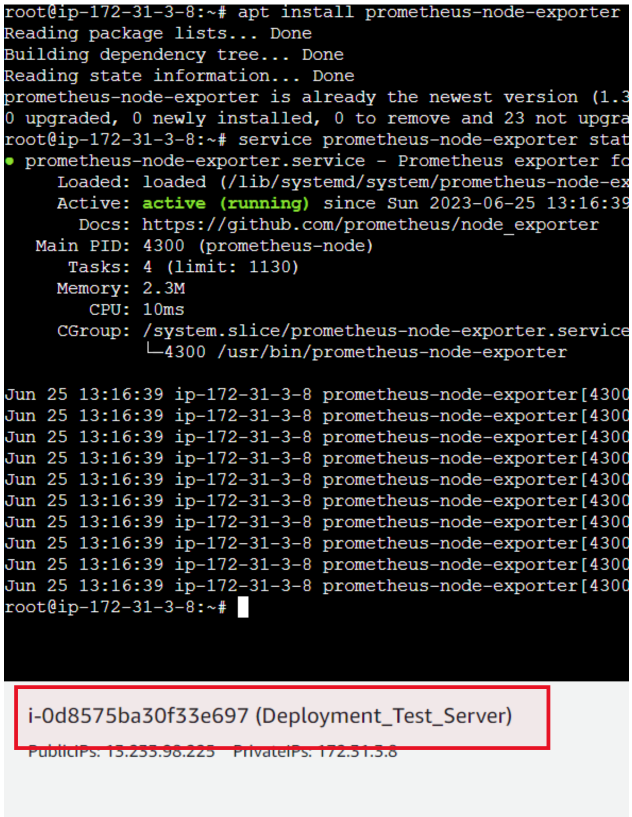
We need to monitor the test and prod servers,

Install **Node exporter** on both servers which acts as a agent to get the metrics

Command - apt install prometheus-node-exporter



```
root@ip-172-31-46-69:~# service prometheus -node-exporter status
prometheus: unrecognized service
root@ip-172-31-46-69:~# service prometheus-node-exporter status
● prometheus-node-exporter.service - Prometheus exporter for machi
     Loaded: loaded (/lib/systemd/system/prometheus-node-exporter
     Active: active (running) since Sun 2023-06-25 13:15:50 UTC;
       Docs: https://github.com/prometheus/node_exporter
   Main PID: 5097 (prometheus-node)
      Tasks: 4 (limit: 1141)
     Memory: 2.4M
        CPU: 11ms
     CGroup: /system.slice/prometheus-node-exporter.service
             └─5097 /usr/bin/prometheus-node-exporter

Jun 25 13:15:50 ip-172-31-46-69 prometheus-node-exporter[5097]: t
Jun 25 13:15:50 ip-172-31-46-69 prometheus-node-exporter[5097]: t
Jun 25 13:15:50 ip-172-31-46-69 prometheus-node-exporter[5097]: t
Jun 25 13:15:50 ip-172-31-46-69 prometheus-node-exporter[5097]: t
Jun 25 13:15:50 ip-172-31-46-69 prometheus-node-exporter[5097]: t
Jun 25 13:15:50 ip-172-31-46-69 prometheus-node-exporter[5097]: t
Jun 25 13:15:50 ip-172-31-46-69 prometheus-node-exporter[5097]: t
Jun 25 13:15:50 ip-172-31-46-69 prometheus-node-exporter[5097]: t
Jun 25 13:15:50 ip-172-31-46-69 prometheus-node-exporter[5097]: t
Jun 25 13:15:50 ip-172-31-46-69 prometheus-node-exporter[5097]: t
root@ip-172-31-46-69:~#
```

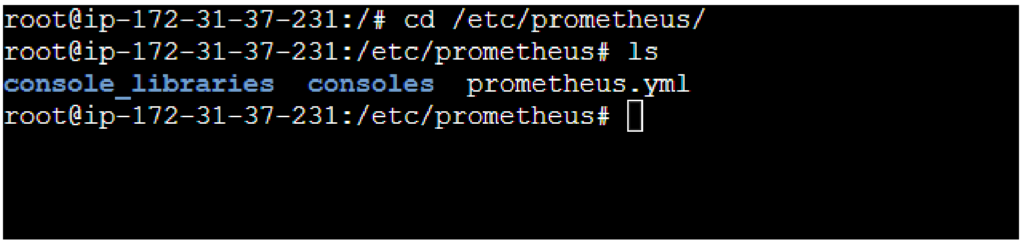
```
root@ip-172-31-3-8:~# apt install prometheus-node-exporter
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
prometheus-node-exporter is already the newest version (1.3
0 upgraded, 0 newly installed, 0 to remove and 23 not upgra
root@ip-172-31-3-8:~# service prometheus-node-exporter stat
● prometheus-node-exporter.service - Prometheus exporter fo
     Loaded: loaded (/lib/systemd/system/prometheus-node-ex
     Active: active (running) since Sun 2023-06-25 13:16:39
       Docs: https://github.com/prometheus/node_exporter
   Main PID: 4300 (prometheus-node)
      Tasks: 4 (limit: 1130)
     Memory: 2.3M
        CPU: 10ms
     CGroup: /system.slice/prometheus-node-exporter.service
             └─4300 /usr/bin/prometheus-node-exporter

Jun 25 13:16:39 ip-172-31-3-8 prometheus-node-exporter[4300
Jun 25 13:16:39 ip-172-31-3-8 prometheus-node-exporter[4300
Jun 25 13:16:39 ip-172-31-3-8 prometheus-node-exporter[4300
Jun 25 13:16:39 ip-172-31-3-8 prometheus-node-exporter[4300
Jun 25 13:16:39 ip-172-31-3-8 prometheus-node-exporter[4300
Jun 25 13:16:39 ip-172-31-3-8 prometheus-node-exporter[4300
Jun 25 13:16:39 ip-172-31-3-8 prometheus-node-exporter[4300
Jun 25 13:16:39 ip-172-31-3-8 prometheus-node-exporter[4300
Jun 25 13:16:39 ip-172-31-3-8 prometheus-node-exporter[4300
Jun 25 13:16:39 ip-172-31-3-8 prometheus-node-exporter[4300
root@ip-172-31-3-8:~#
```


On monitoring instance-

Check out the Prometheus directory

Command – cd /etc/Prometheus/

And edit the **.yml** file and insert the **ip's** which need to be monitored.



```
root@ip-172-31-37-231:/# cd /etc/prometheus/
root@ip-172-31-37-231:/etc/prometheus# ls
console_libraries  consoles  prometheus.yml
root@ip-172-31-37-231:/etc/prometheus#
```

Scrape : configs

- job_name: any name

  # If prometheus-node-exporter is installed, grab stats about the local

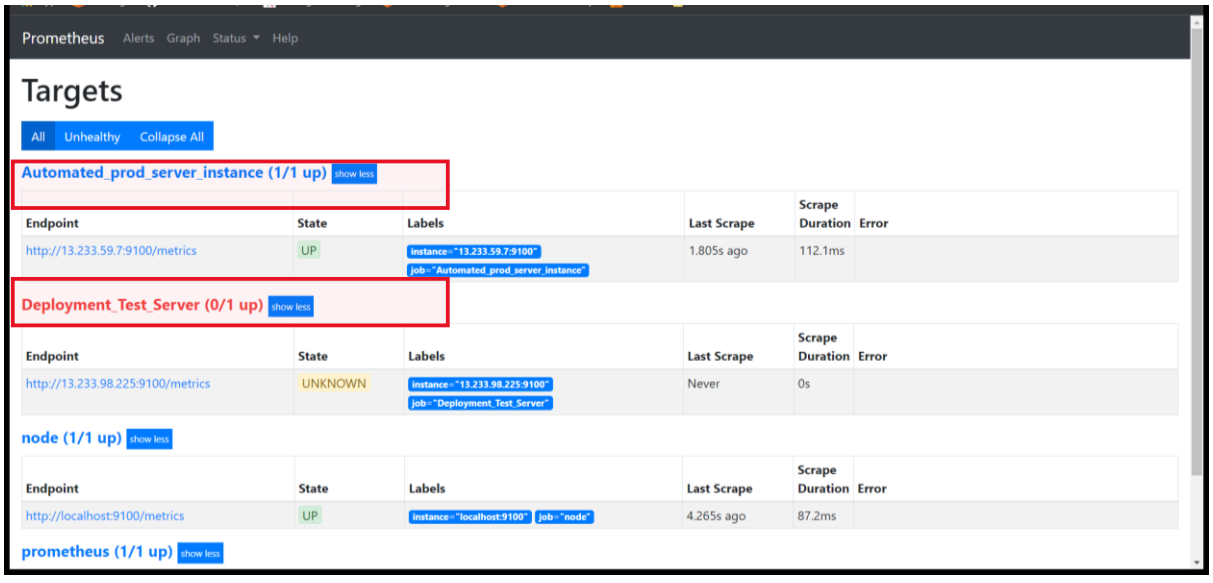
  # machine by default.

static_configs:

- targets: ['IP(to be monitored):9100']

Restart the Prometheus –

Command- service Prometheus restart



Visualization of metrics with Grafana server –

On monitoring instance-

Installation of Grafana- https://grafana.com/docs/grafana/latest/setup-grafana/installation/debian/

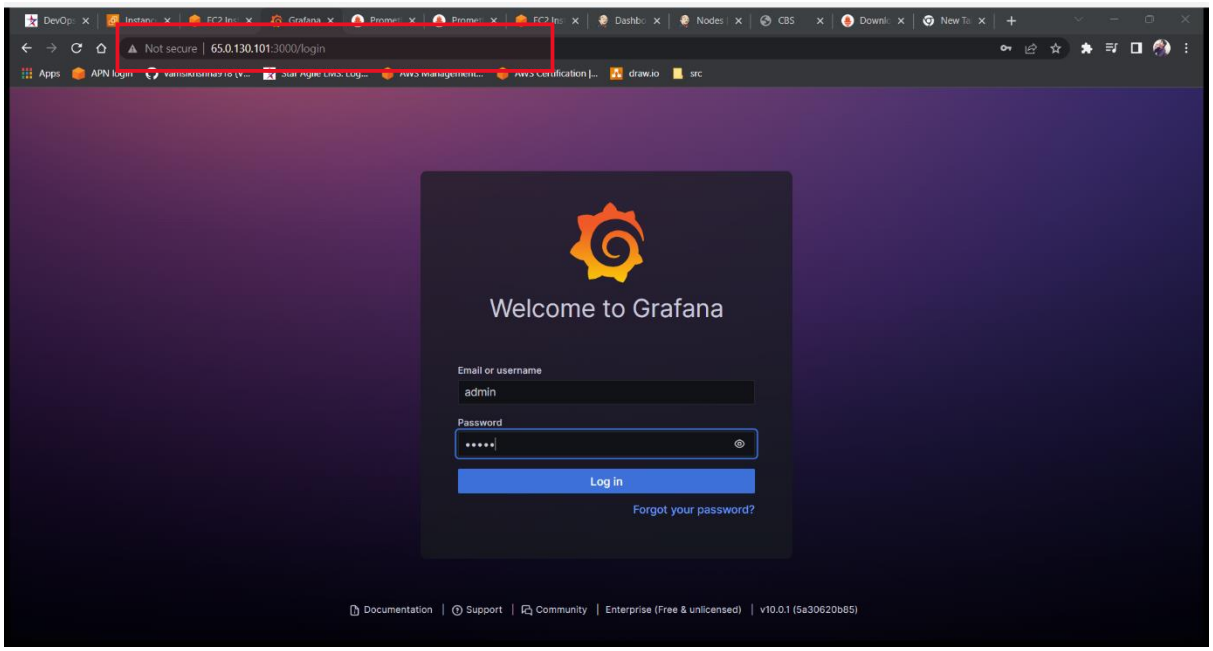
```
root@ip-172-31-37-231:/# grafana server --version
Version 10.0.1 (commit: 5a30620b85, branch: HEAD)
root@ip-172-31-37-231:/# service grafana server restart
grafana: unrecognized service
root@ip-172-31-37-231:/# service grafana-server restart
root@ip-172-31-37-231:/#
```

i-07ccb1c815ba09ecd (Monitoring instance)

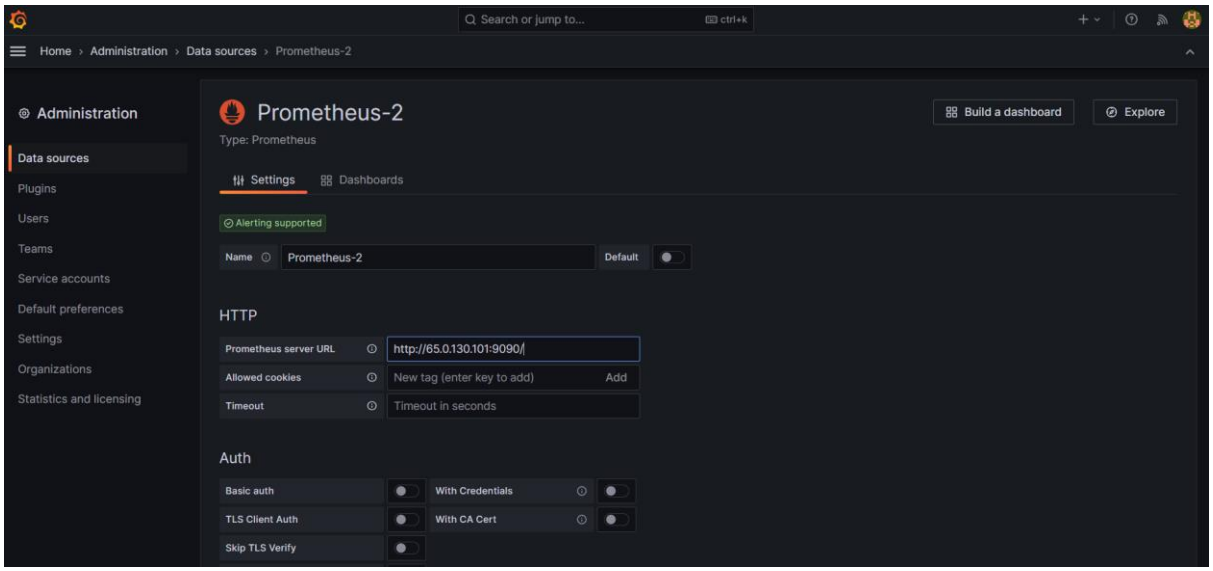
PublicIPs: 65.0.130.101 PrivateIPs: 172.31.37.231

Default port of Grafana – 3000

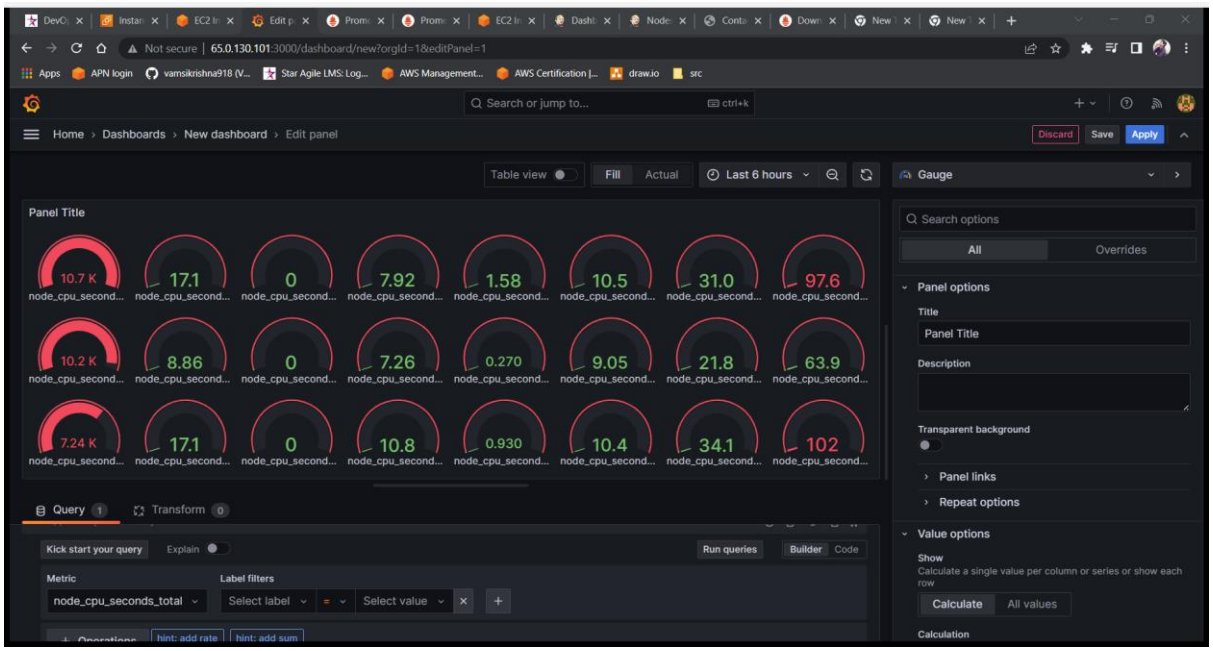
Default username and password - admin & admin



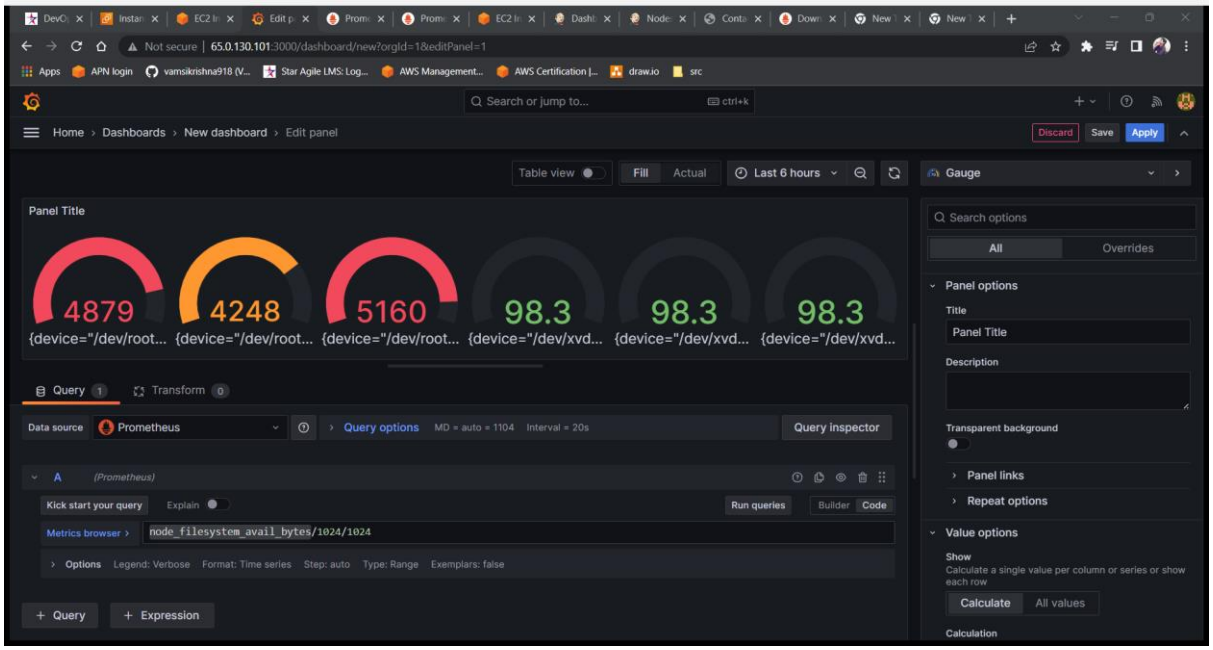
Add the data source:



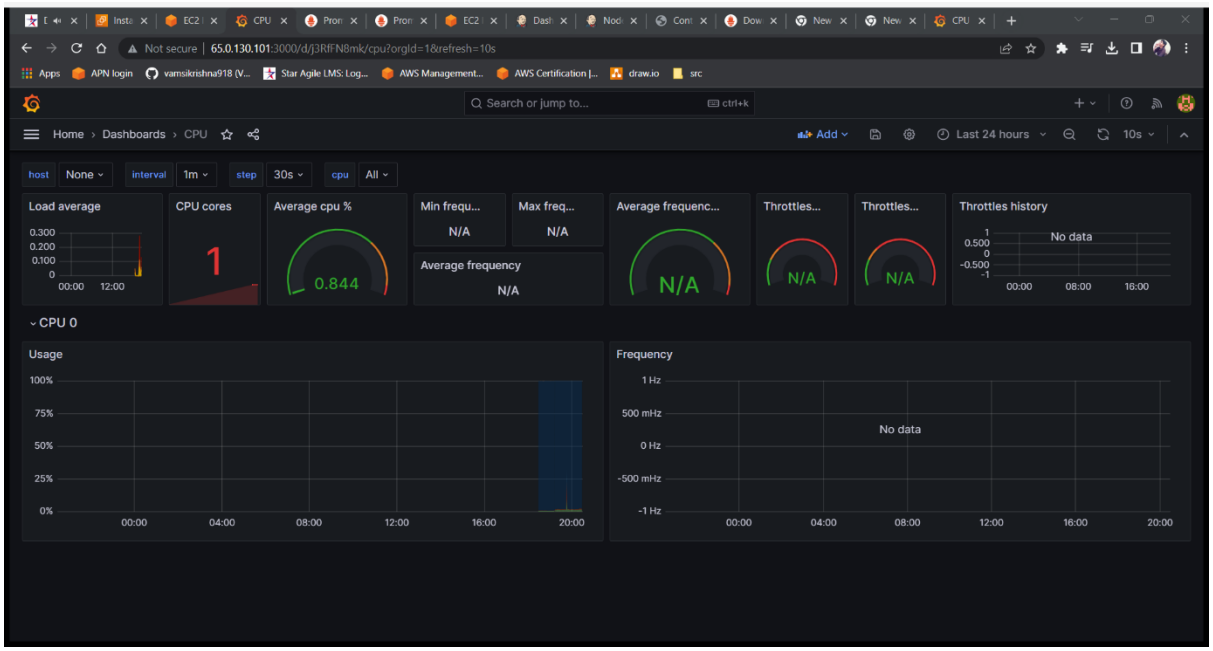
CPU metrics – query (Node_cpu_seconds_total)



Disk Utilization – query (node_filesystem_avail_bytes)



CPU dashboard import- https://grafana.com/grafana/dashboards/9617-cpu/



Memory metrics – https://grafana.com/grafana/dashboards/3002-memory/