# CAPSTONE PROJECT

## Banking and Finance Domain Project – FinanceMe

## SUBMITTED BY: GOKUL C

## DATE: 05-03-2024





gokul.chandrasekharan01@gmail.com

# CONTENT OF THE PROJECT

**FinanceMe** is a Global leading Banking and Financial services provider based out of Germany. The company offers products and services like Banking, Funds Management, Loans, Debit Cards and Credits Cards, Investment Banking etc. Initially the company was using a Monolithic application architecture, As the company grown, It started facing difficulties in managing the application infrastructure and application deployments and Scaling of application when the traffic load increases.

**FinanceMe** has decided to opt for microservice architecture for its applications and decided to go DevOps by implementing necessary automations using CICD. **FinanceMe** has decided to use AWS as primary cloud services provider to create servers, databases and application deployments.

The company's goal is to deliver the product updates frequently to production automatically with High quality & Reliability. They also want to accelerate software delivery speed, quality and reducing feedback time between developers and testers.

Currently, they are facing following problems, because of various technologies involved in the project.

✓ Building Complex Monolithic Application is difficult.
✓ Manual efforts to test various components/modules of the project
✓ Incremental builds are difficult to manage, test and deploy.
✓ It was not possible to scale up individual modules independently.
✓ Creation of infrastructure and configure it manually is very time consuming
✓ Continuous manual monitoring the application is quite challenging.

In order to implement a POC, you are requested to develop a mavenized microservice using spring boot and in memory h2 database.

1. a microservice which exposes below mentioned endpoints as APIs and uses pre configured AWS RDS – mysql database to store the data.

a. /createAccount (HTTP Method: POST) (Request Body: JSON)
b. /updateAccount/{account no.} (HTTP Method : PUT ) (Request Body : JSON)



gokul.chandrasekharan01@gmail.com

c. /viewPolicy/{account no.}  (HTTP Method: GET) (No Request Body )
d. /deletePolicy/{account no.}  (HTTP Method: DELETE) (No Request Body)

2. Write necessary Junit testcase.
3. Generate HTML report using TestNG.
4. Push your code into your GitHub Repository.

**Note: Preload some data into the database.**

Later, you need to implement Continuous Integration & Continuous Deployment using following tools:

✓ Git - For version control for tracking changes in the code files
✓ Maven – For Continuous Build
✓ Jenkins - For continuous integration and continuous deployment
✓ Docker - For deploying containerized applications
✓ Ansible - Configuration management tools
✓ Selenium - For automating tests on the deployed web application
✓ Terraform - For creation of infrastructure.
✓ Prometheus and Grafana – For Automated Monitoring and Report Visualization

This project will be about how to test the services and deploy code to dev/stage/prod etc, just on a click of button.

**Business challenge/requirement**
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.


gokul.chandrasekharan01@gmail.com

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

Link for the Solution of **FinanceMe** project code is attached below. Use it to validate your solution.
**https://github.com/StarAgileDevOpsTraining/star-agile-banking-finance.git**

**Note:** To have a detailed information about running the application and exposed APIs, Input/Output format, Refer to the README.md in the GitHub repository.

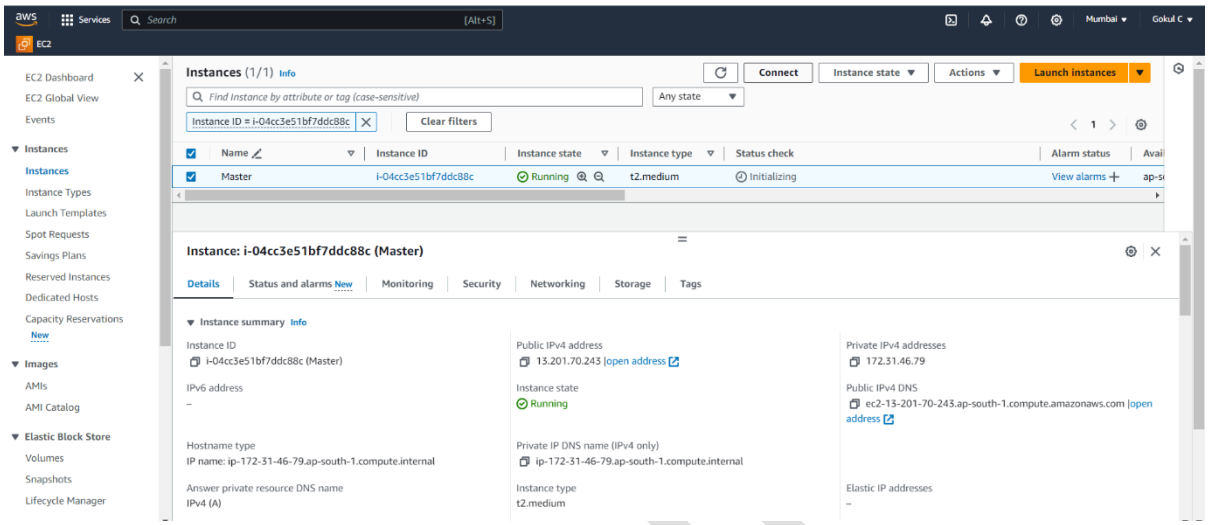




gokul.chandrasekharan01@gmail.com

# PROJECT SOLUTION

Creating master instance to perform the above project




Installation of all the necessary dependencies

JAVA INSTALLATION

```
root@ip-172-31-46-79:/home/ubuntu# java --version
openjdk 11.0.22 2024-01-16
OpenJDK Runtime Environment (build 11.0.22+7-post-Ubuntu-0ubuntu222.04.1)
OpenJDK 64-Bit Server VM (build 11.0.22+7-post-Ubuntu-0ubuntu222.04.1, mixed mode, sharing)
root@ip-172-31-46-79:/home/ubuntu#
```

i-04cc3e51bf7ddc88c (Master)

PublicIPs: 13.201.70.243 PrivateIPs: 172.31.46.79

MAVEN INSTALLATION

```
root@ip-172-31-46-79:/home/ubuntu# mvn --version
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.22, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "6.2.0-1018-aws", arch: "amd64", family: "unix"
root@ip-172-31-46-79:/home/ubuntu#
```

i-04cc3e51bf7ddc88c (Master)

PublicIPs: 13.201.70.243 PrivateIPs: 172.31.46.79



gokul.chandrasekharan01@gmail.com

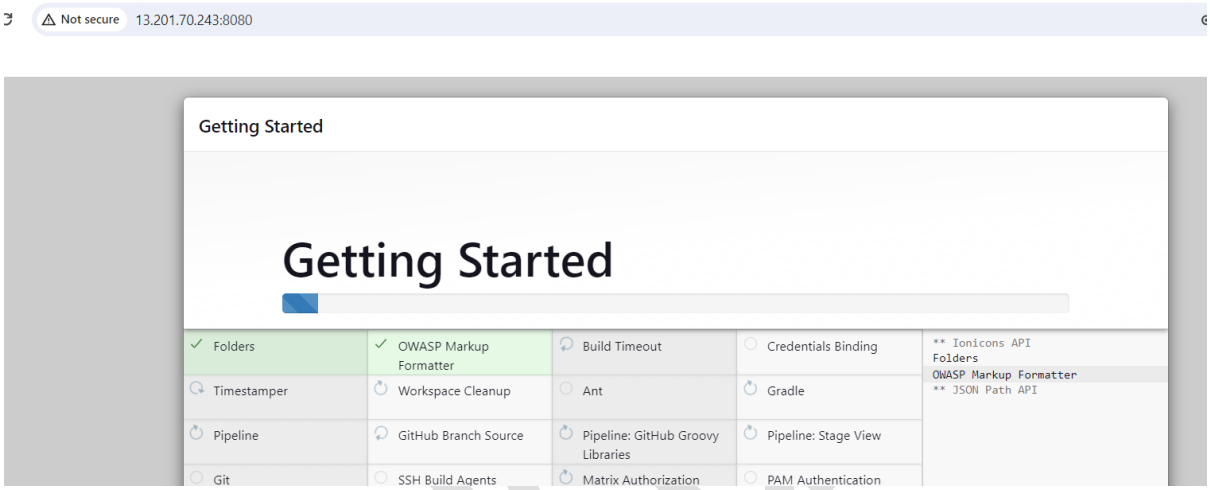
## JENKINS INSTALLATION

```
root@ip-172-31-46-79:/home/ubuntu# jenkins --version
2.440.1
root@ip-172-31-46-79:/home/ubuntu#
```

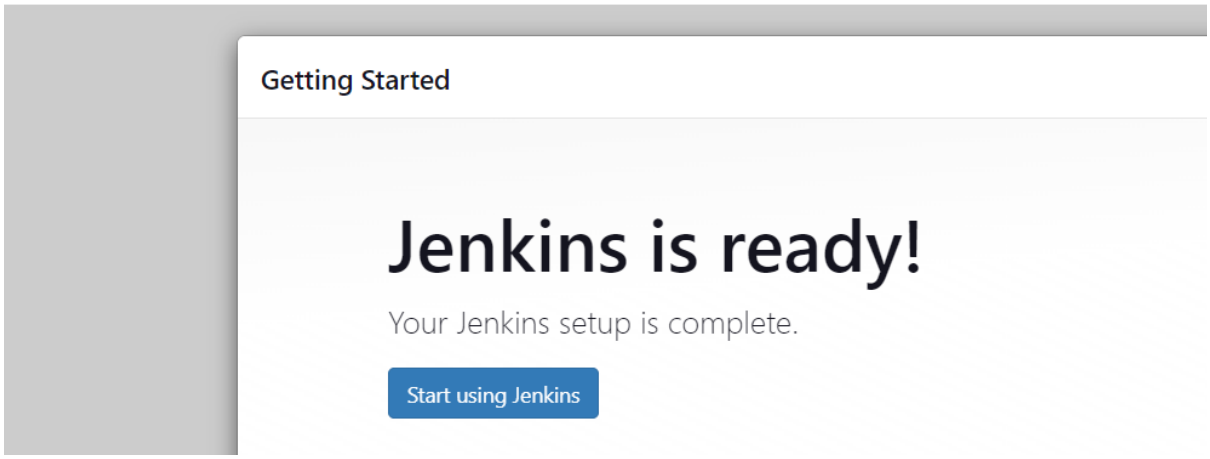
i-04cc3e51bf7ddc88c (Master)

PublicIPs: 13.201.70.243  PrivateIPs: 172.31.46.79

## Opening in default port 8080










gokul.chandrasekharan01@gmail.com

## DOCKER INSTALLATION

```
root@ip-172-31-46-79:/home/ubuntu# docker --version
Docker version 24.0.5, build 24.0.5-0ubuntu1~22.04.1
root@ip-172-31-46-79:/home/ubuntu#
```

i-04cc3e51bf7ddc88c (Master)

PublicIPs: 13.201.70.243 PrivateIPs: 172.31.46.79

## ANSIBLE INSTALLATION

```
root@ip-172-31-46-79:/home/ubuntu# ansible --version
ansible [core 2.16.4]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.10.12 (main, Nov 20 2023, 15:14:05) [GCC 11.4.0] (/usr/bin/python3)
  jinja version = 3.0.3
  libyaml = True
root@ip-172-31-46-79:/home/ubuntu#
```

i-04cc3e51bf7ddc88c (Master)

PublicIPs: 13.201.70.243 PrivateIPs: 172.31.46.79

CloudShell Feedback

## TERRAFORM INSTALLATION

```
root@ip-172-31-46-79:/home/ubuntu# terraform --version
Terraform v1.7.4
on linux_amd64
root@ip-172-31-46-79:/home/ubuntu#
```

i-04cc3e51bf7ddc88c (Master)

PublicIPs: 13.201.70.243 PrivateIPs: 172.31.46.79

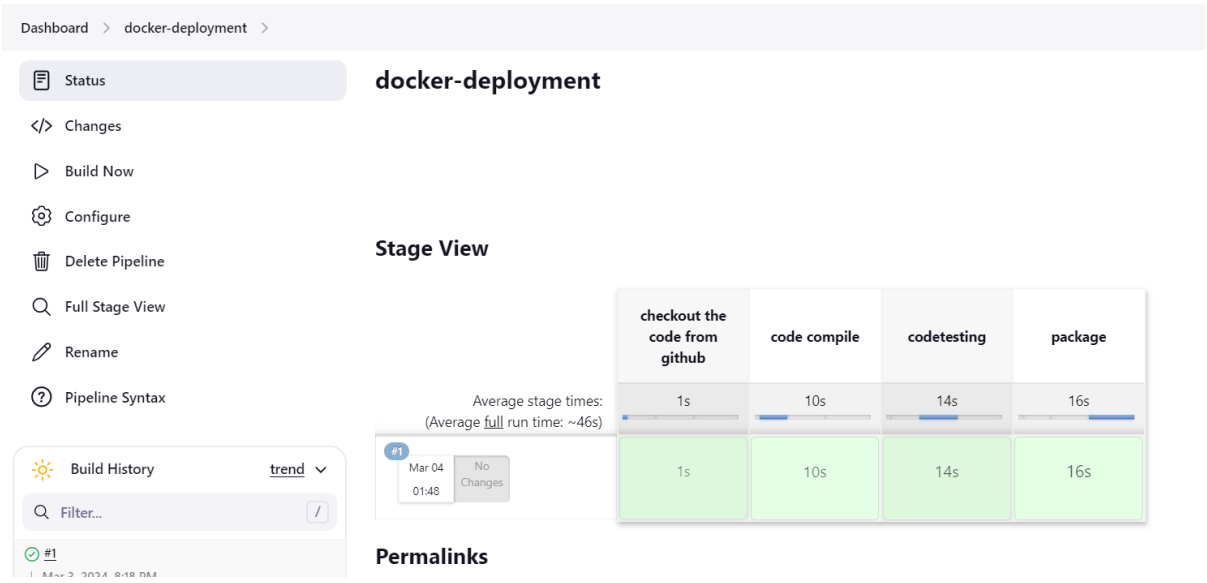


gokul.chandrasekharan01@gmail.com

# Docker deployment

Created a new pipeline job for the docker deployment to perform checked out, compiled, tested, packaged and containerized.

```
pipeline{
    agent any
    stages{
        stage('checkout the code from github'){
            steps{
                git url: 'https://github.com/StarAgileDevOpsTraining/star-agile-banking-finance.git', branch:"master"
                echo 'github url checkout'
            }
        }

        stage('code compile'){
            steps{
                echo 'starting compiling'
                sh 'mvn compile'
            }
        }

        stage('codetesting'){
            steps{
                sh 'mvn test'
            }
        }

        stage('package'){
            steps{
                sh 'mvn clean package'
            }
        }
```

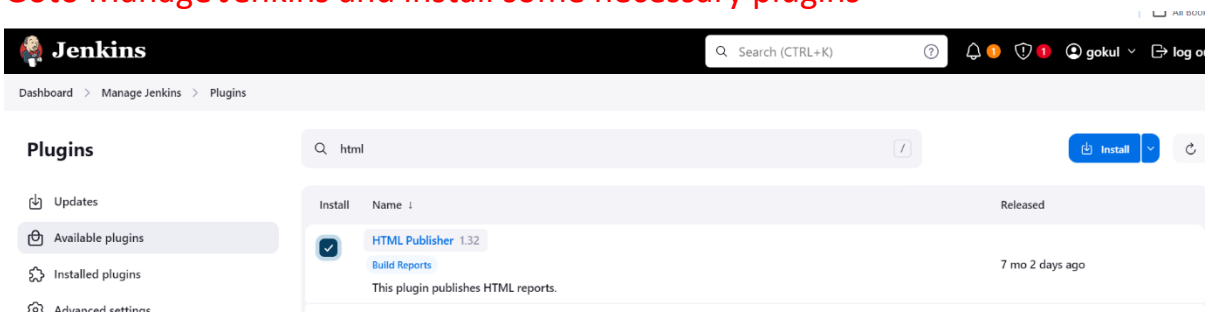


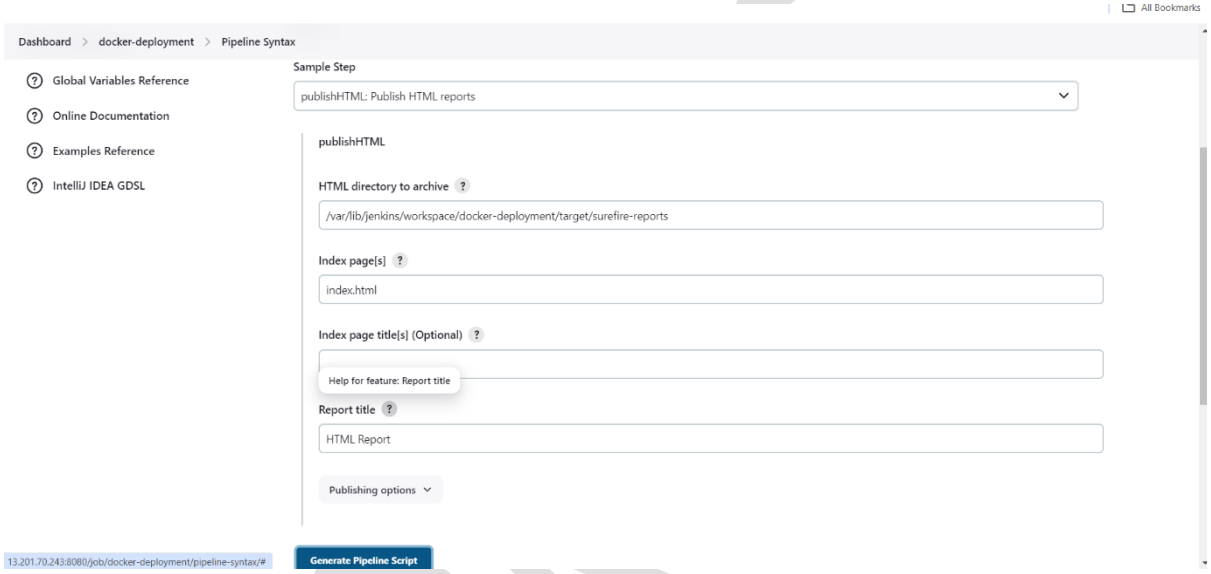



gokul.chandrasekharan01@gmail.com

# HTML Report creation

## Goto Manage Jenkins and install some necessary plugins




```
24      stage('package'){
25          steps{
26              sh 'mvn clean package'
27          }
28      }
29
30      stage('HTML Report'){
31          steps{publishHTML([allowMissing: false, alwaysLinkToLastBuild: false, keepAll: false, reportDir: '/var/lib/jenkins/workspace/dc
32      }
33  }
34  }
35  }
```

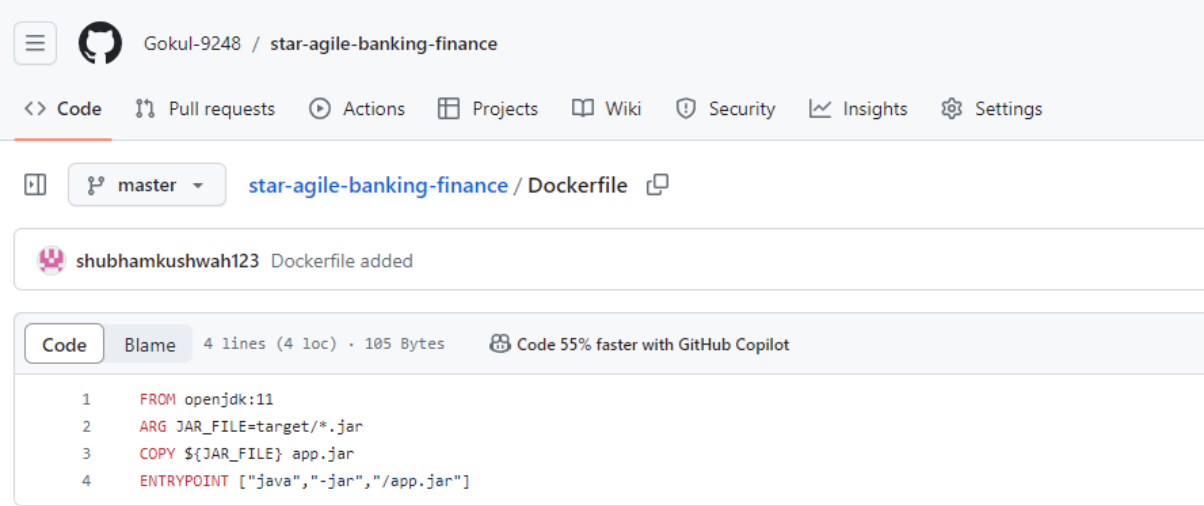
```
[Pipeline] publishHTML
[htmlpublisher] Archiving HTML reports...
[htmlpublisher] Archiving at PROJECT level /var/lib/jenkins/workspace/docker-deployment/target/surefire-reports to /var/lib/jenkins/jobs/docker-deployment/htmlreports/HTML_20Report
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```



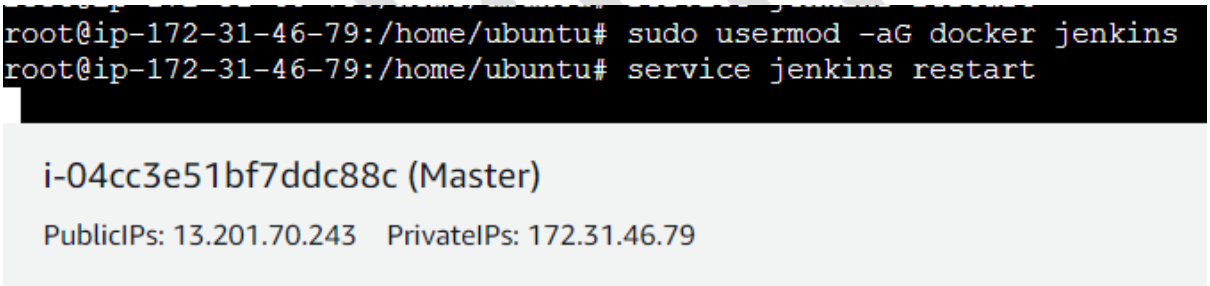
gokul.chandrasekharan01@gmail.com

## Building docker image

Before that verifying the **dockerfile** is present in our repo and its correct




Giving necessary sudo root permissions to Jenkins





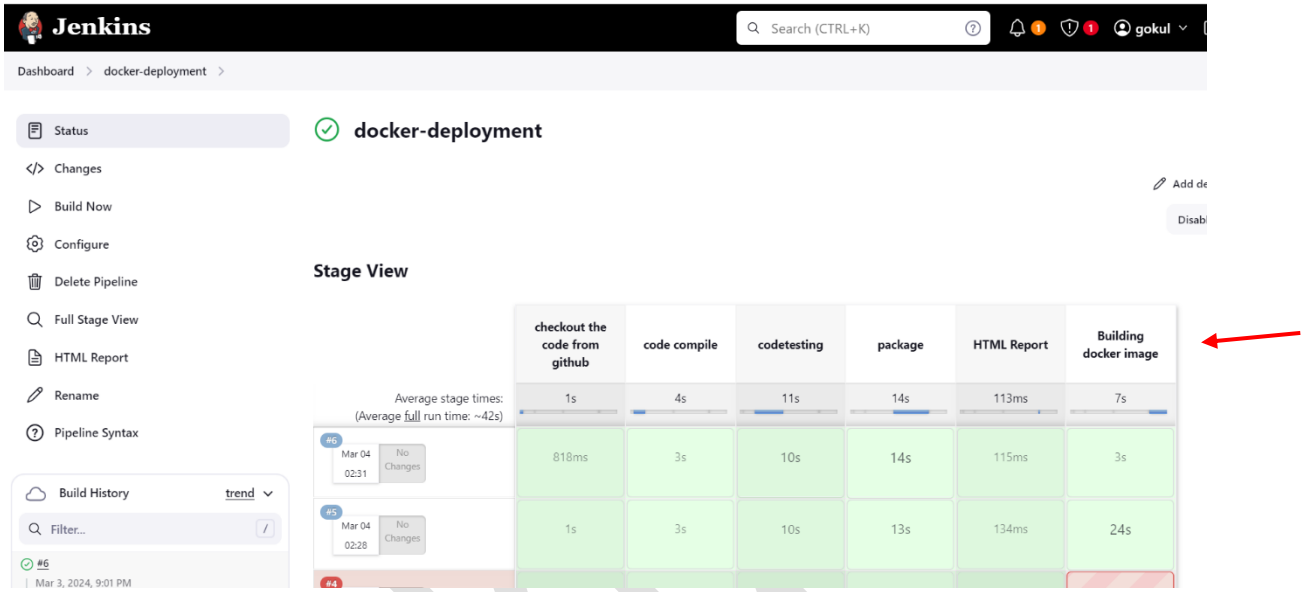




gokul.chandrasekharan01@gmail.com

## Creating the docker image for our FinanceMe application

```
        }
    }
    stage('Building docker image'){
        steps{
            sh 'docker build -t gokul1311/financeme:1.0 . '
        }
    }
  }
}
}
```

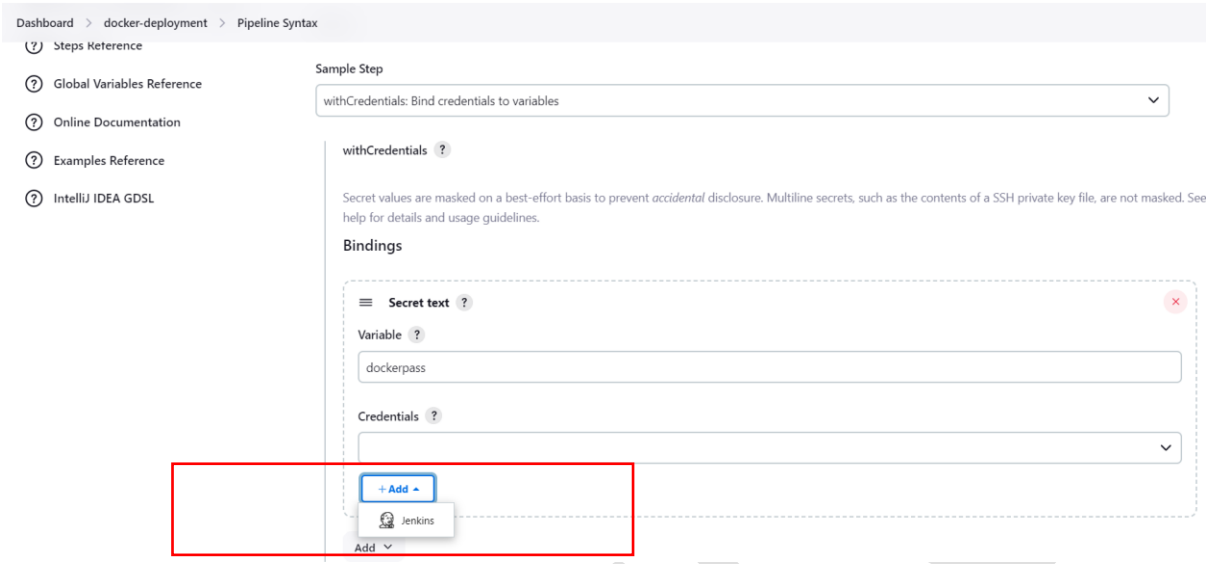




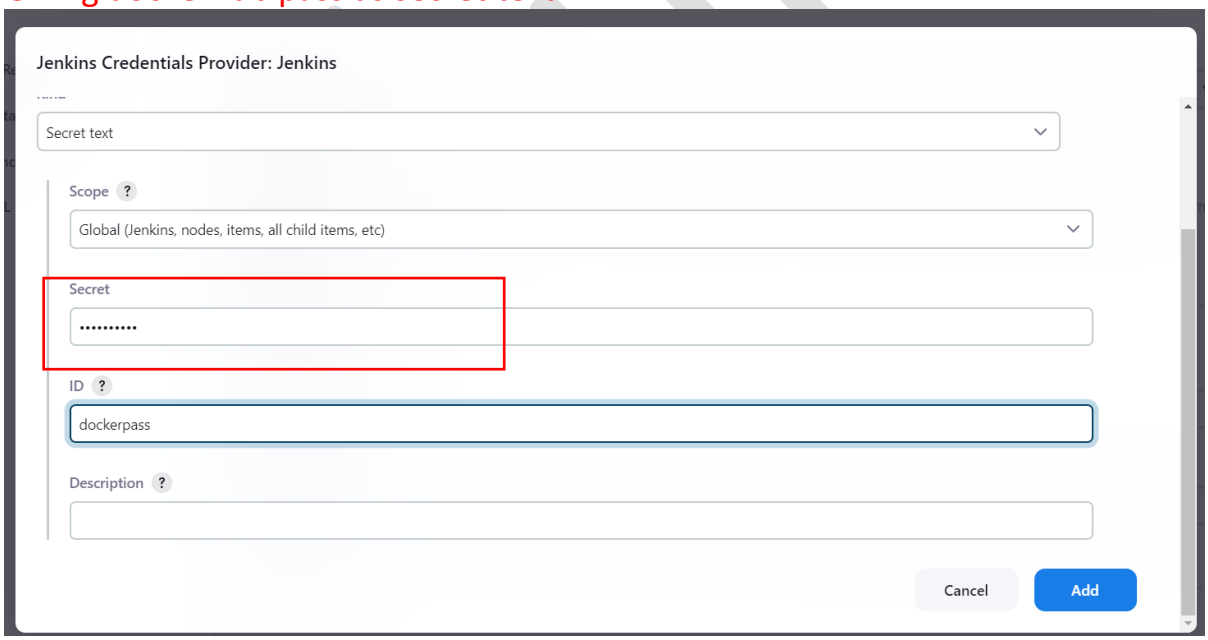

gokul.chandrasekharan01@gmail.com

# Dockerhub login and image pushing

## Setting dockerhub username password for Jenkins




## Giving dockerhub pass as secret text



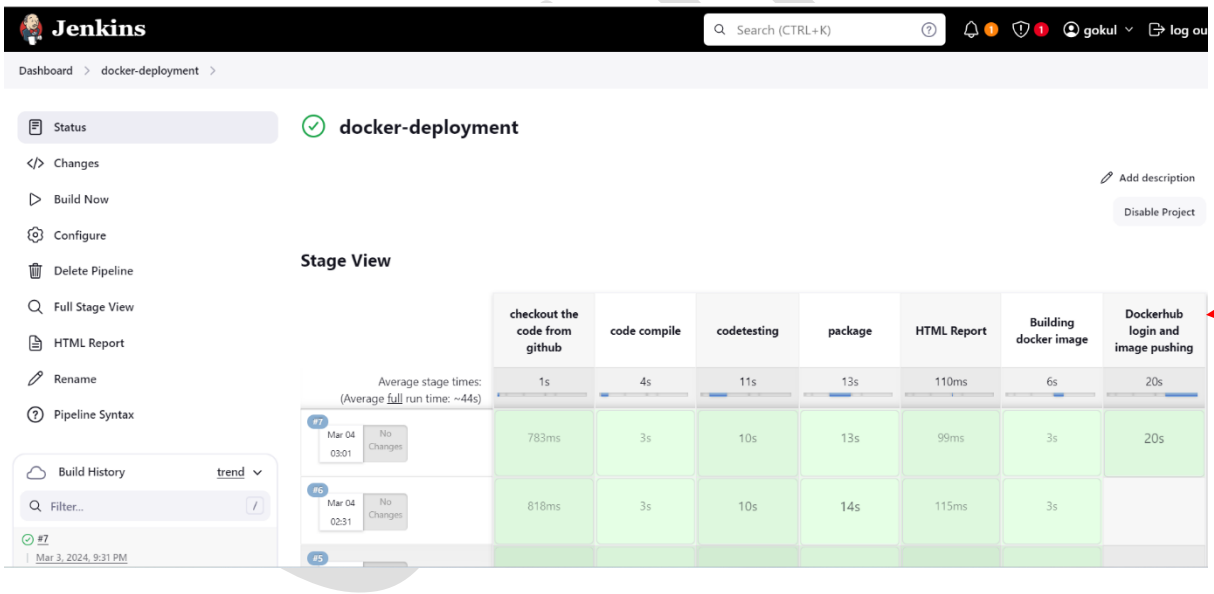



gokul.chandrasekharan01@gmail.com

# Dockerhub login and image pushing

Script ?

```
            sh 'mvn test'
        }
    }

    stage('package'){
        steps{
            sh 'mvn clean package'
        }
    }

    stage('HTML Report'){
        steps{publishHTML([allowMissing: false, alwaysLinkToLastBuild: false, keepAll: false, reportDir: '/var/lib/jenkins/workspace

        }
    }
    stage('Building docker image'){
        steps{
            sh 'docker build -t gokul1311/financeme:1.0 .'
        }
    }

    stage('Dockerhub login and image pushing'){
        steps{withCredentials([string(credentialsId: 'dockerpass', variable: 'dockerpass')]) {
            sh 'docker login -u gokul1311 -p ${dockerpass}'
            sh 'docker push gokul1311/financeme:1.0'
        }
    }
    }
}
}
```

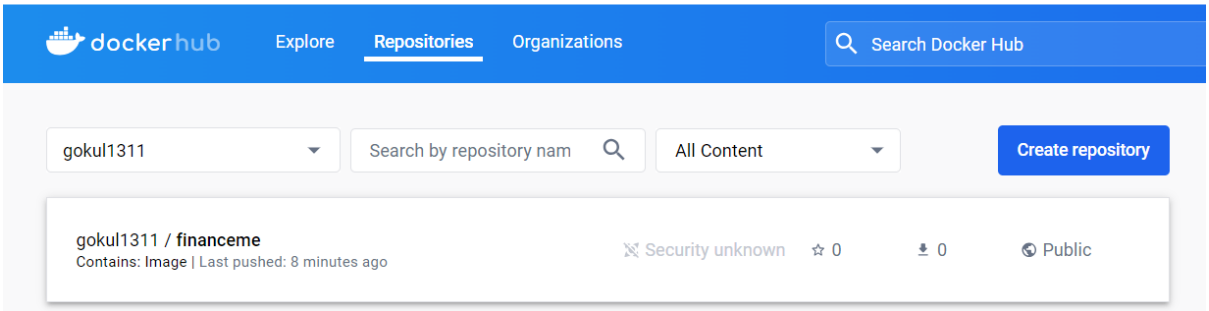



```
03127cdb479b: Mounted from library/openjdk
9c742cd6c7a5: Mounted from library/openjdk
e5eb581cbe58: Pushed
1.0: digest: sha256:b0be22459381f888ba412a98d62ea0973d5c114c845bc44bd780a68a31c4b131 size: 2007
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```



gokul.chandrasekharan01@gmail.com

## Verifying the Dockerhub




# Docker containerizing the application

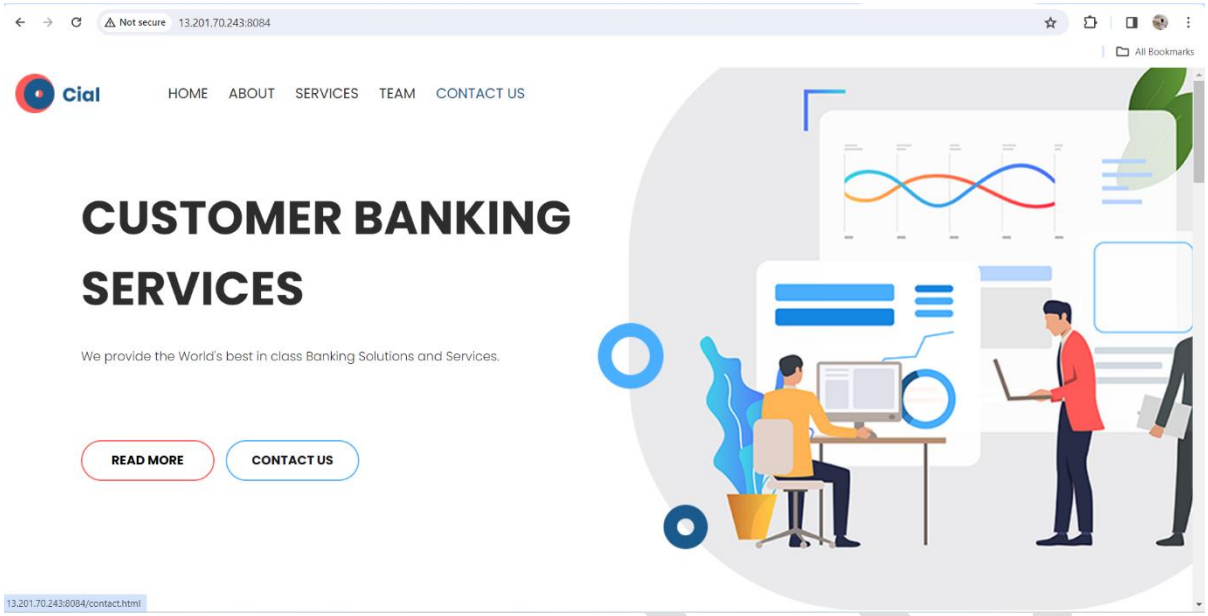
## Container creation and port expose


```
0312fcdb479b: Layer already exists
9c742cd6c7a5: Layer already exists
16aef6cc7fee: Pushed
1.0: digest: sha256:0c8cb90f146452459d09a93e9dc6750c2f613ccf14b82386caddafe9b9a2cdaa size: 2007
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (containerizing the application)
[Pipeline] sh
+ docker run -itd -p 8084:8081 gokul1311/financeme:1.0
7067eb2d95d4803e2c421fa97087d595354e46da75aeae65377f1174d0d3ae57
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```



gokul.chandrasekharan01@gmail.com

Verifying the application





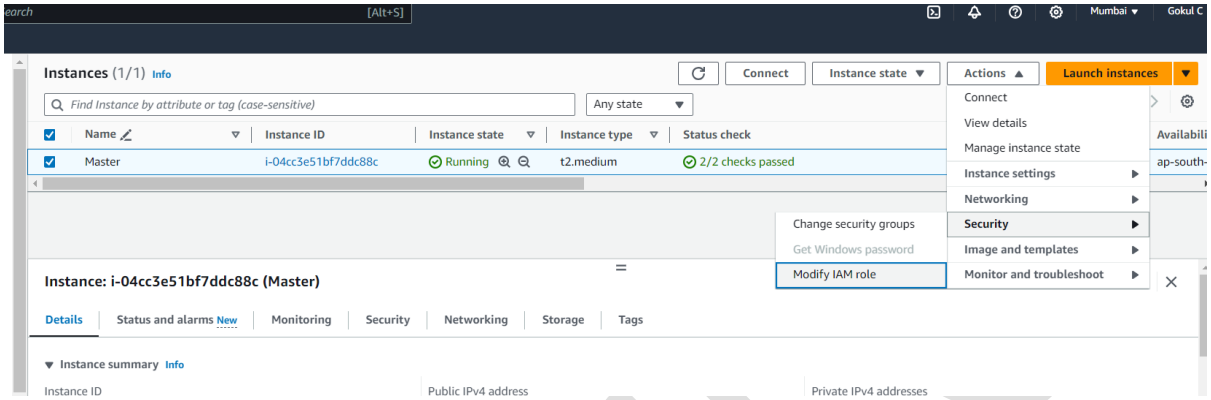



gokul.chandrasekharan01@gmail.com

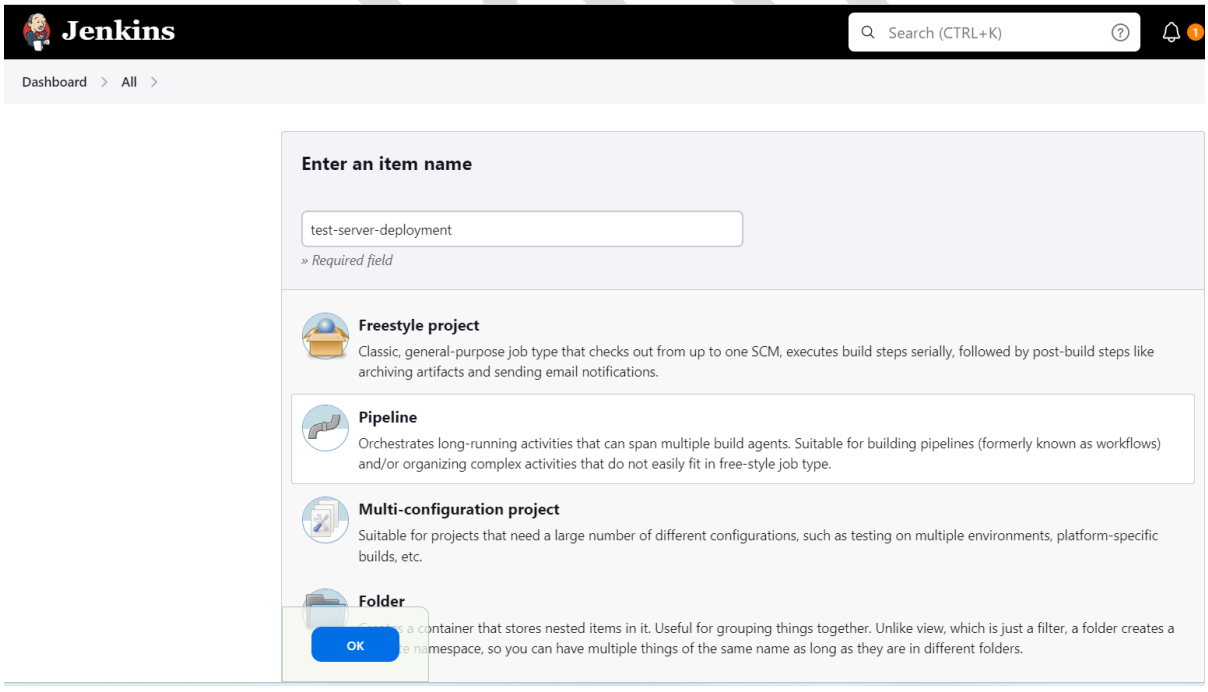
# Test server Deployment

## Test server creation using Terraform

Before that we have to give the EC2 full access permission to our master machine using IAM role.



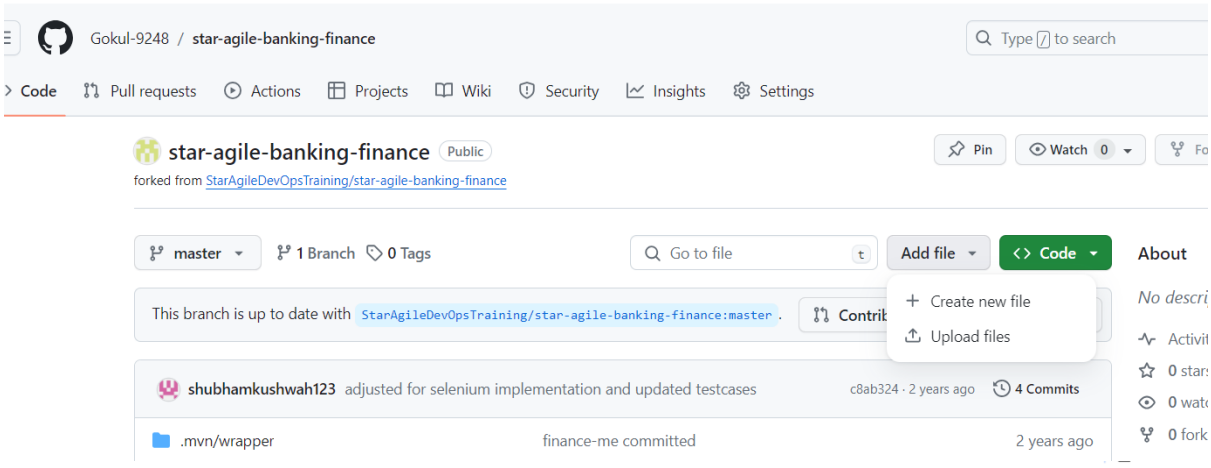

Creating new Pipeline job for test server deployment

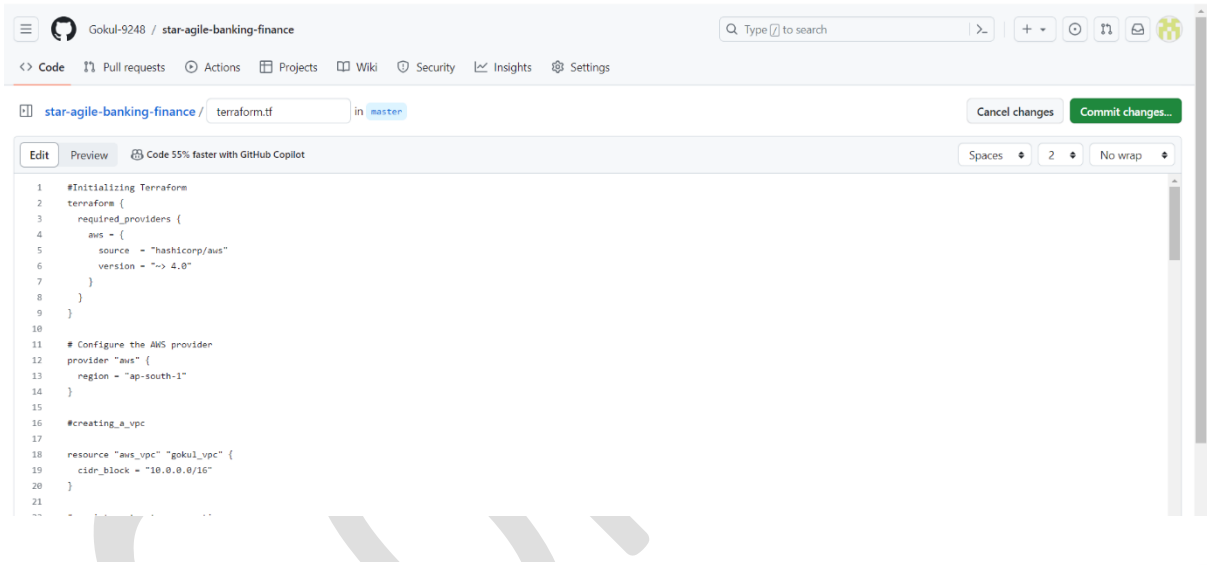


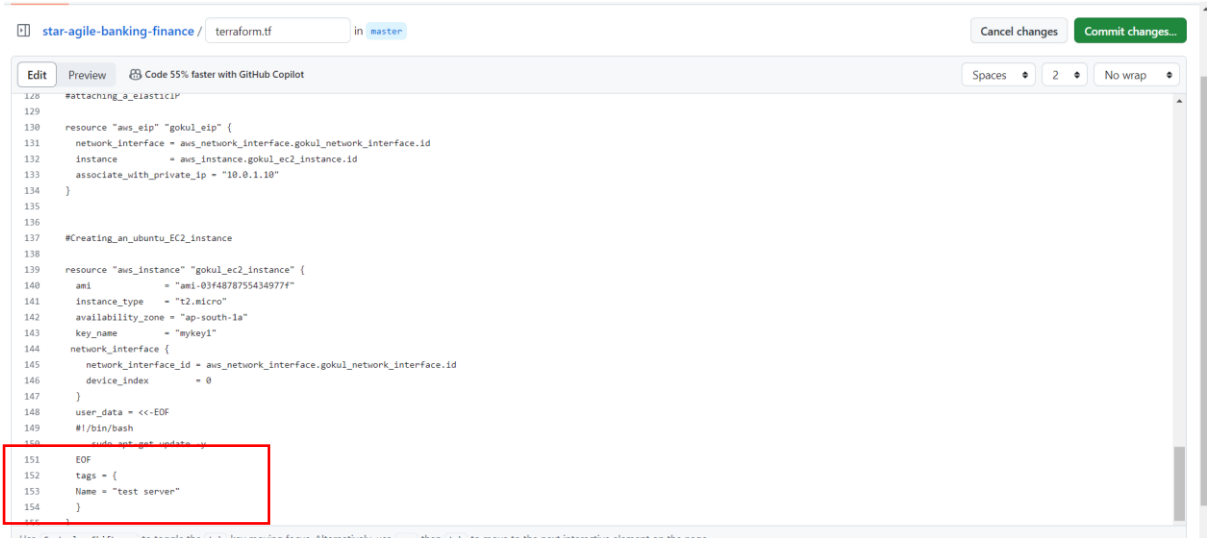



gokul.chandrasekharan01@gmail.com

We don’t have terraform.tf file in our repository for terraform instance creation, so we are creating aa new terraform file.

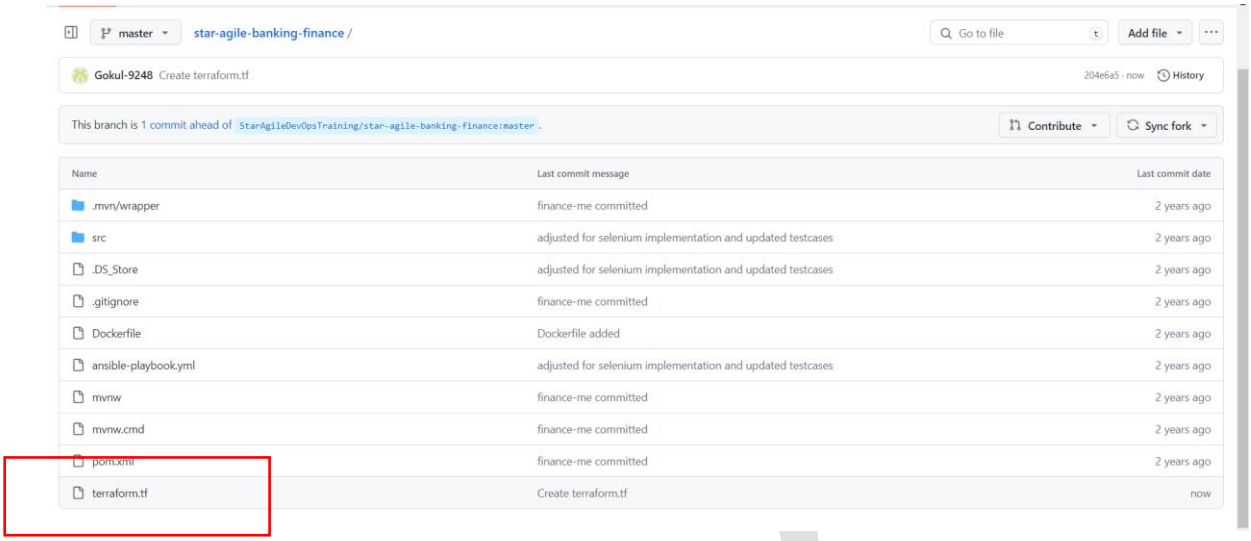


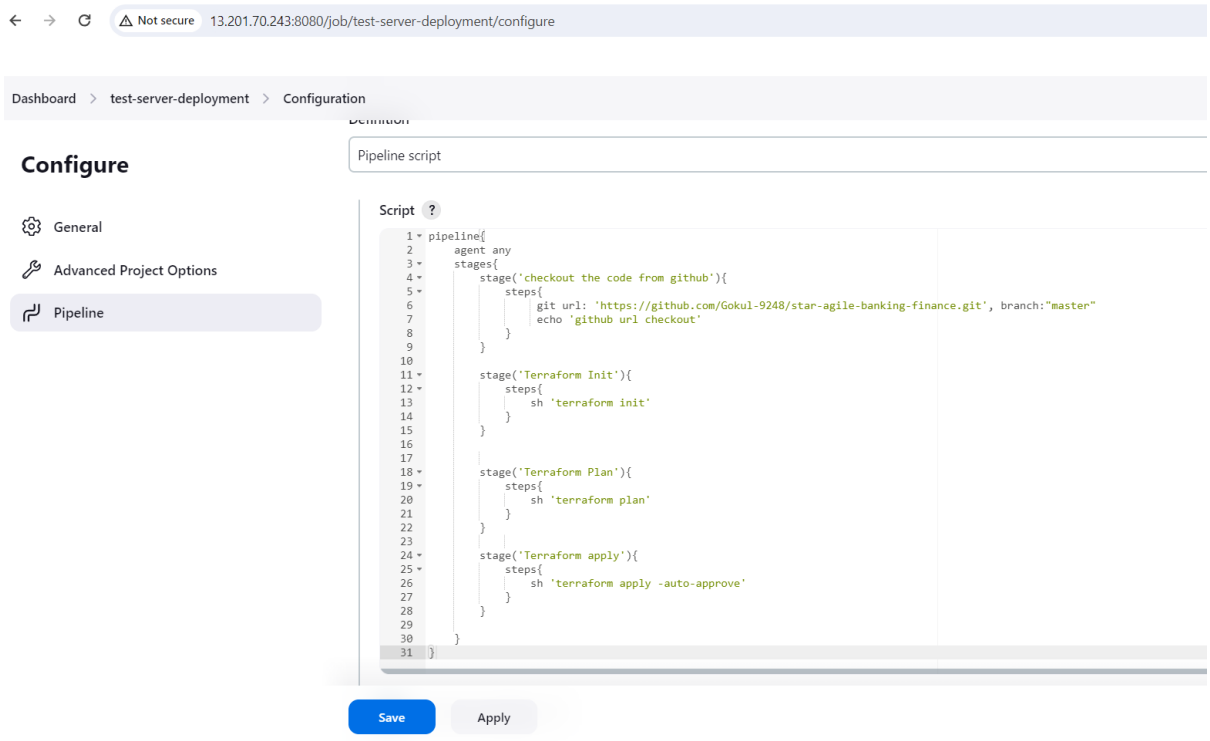






gokul.chandrasekharan01@gmail.com

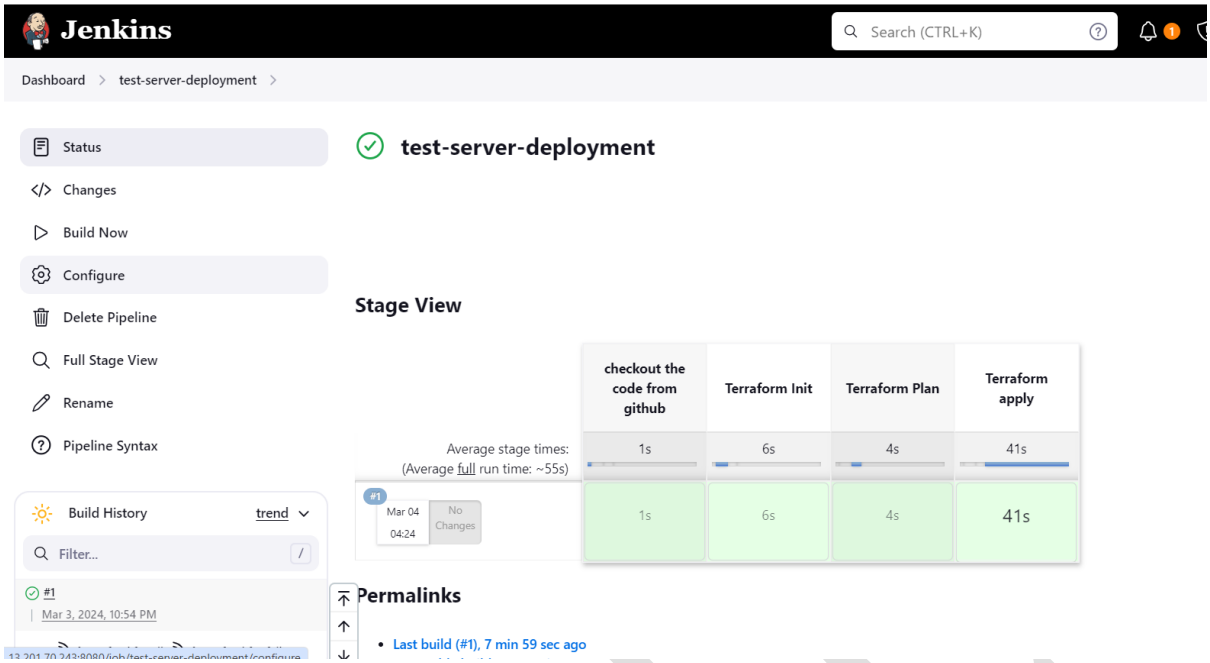


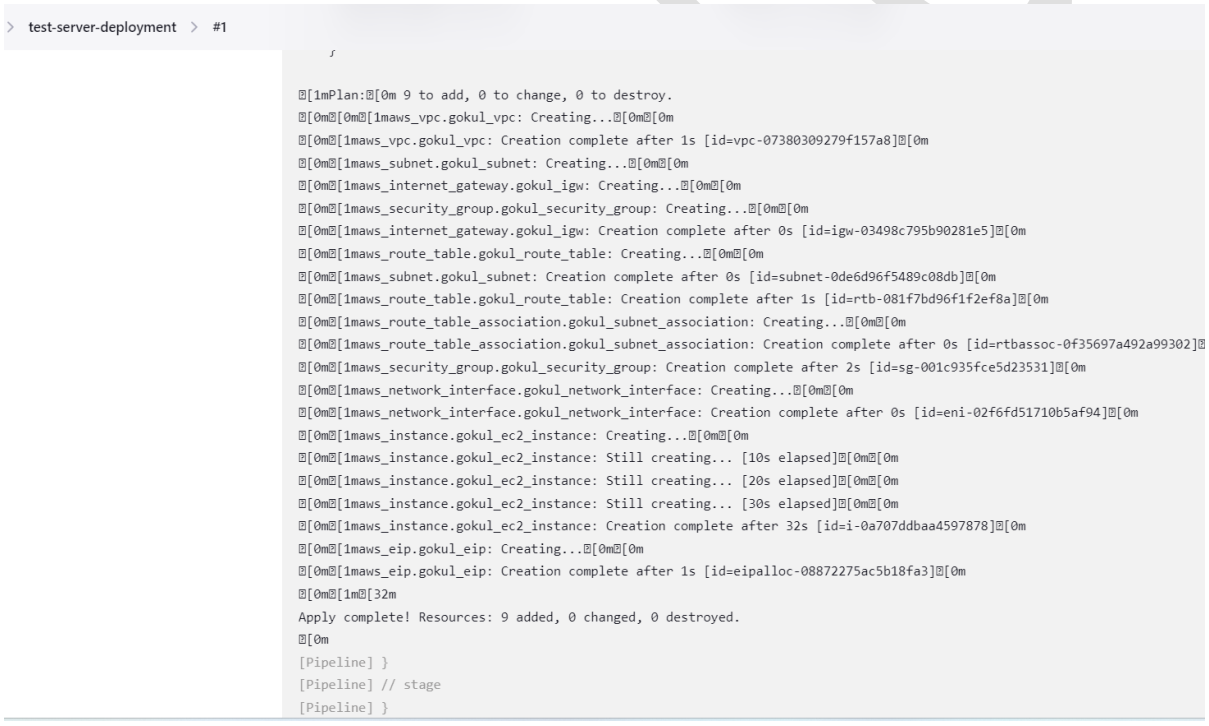

## Pipeline for the terraform execution






gokul.chandrasekharan01@gmail.com

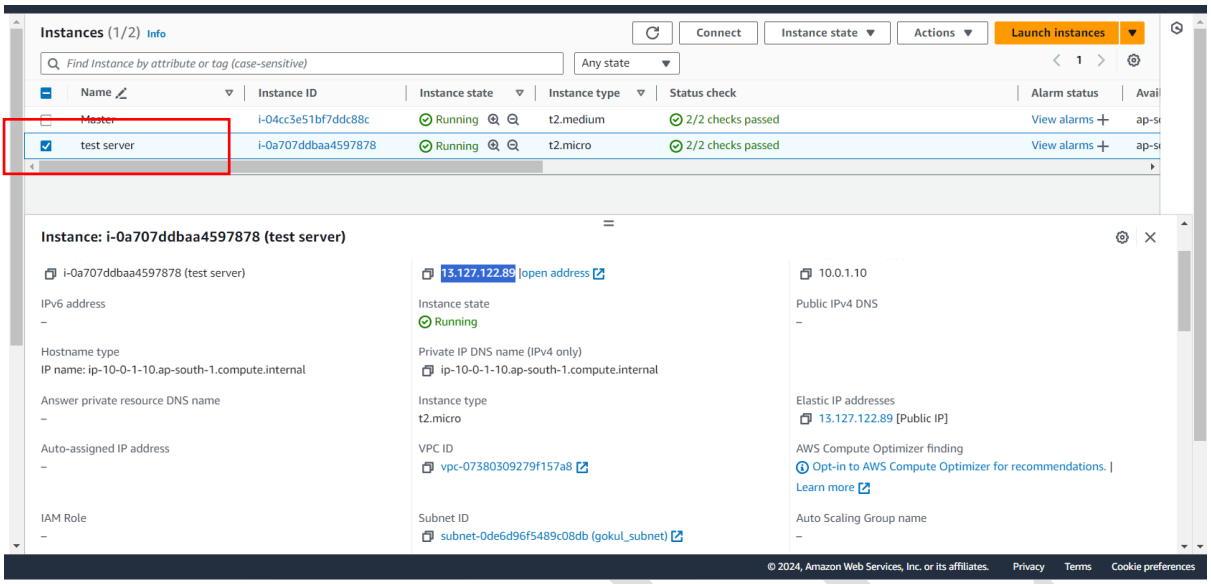





gokul.chandrasekharan01@gmail.com

# Verifying the terraform instance creation



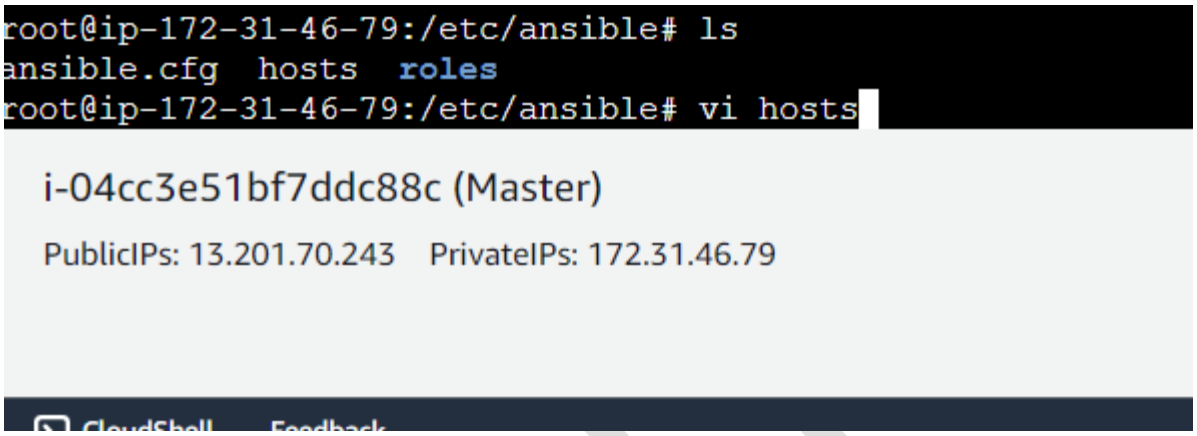


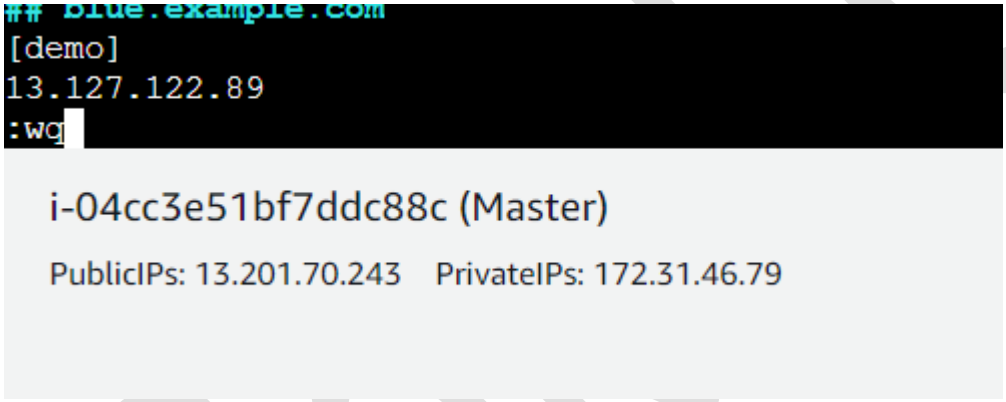



gokul.chandrasekharan01@gmail.com

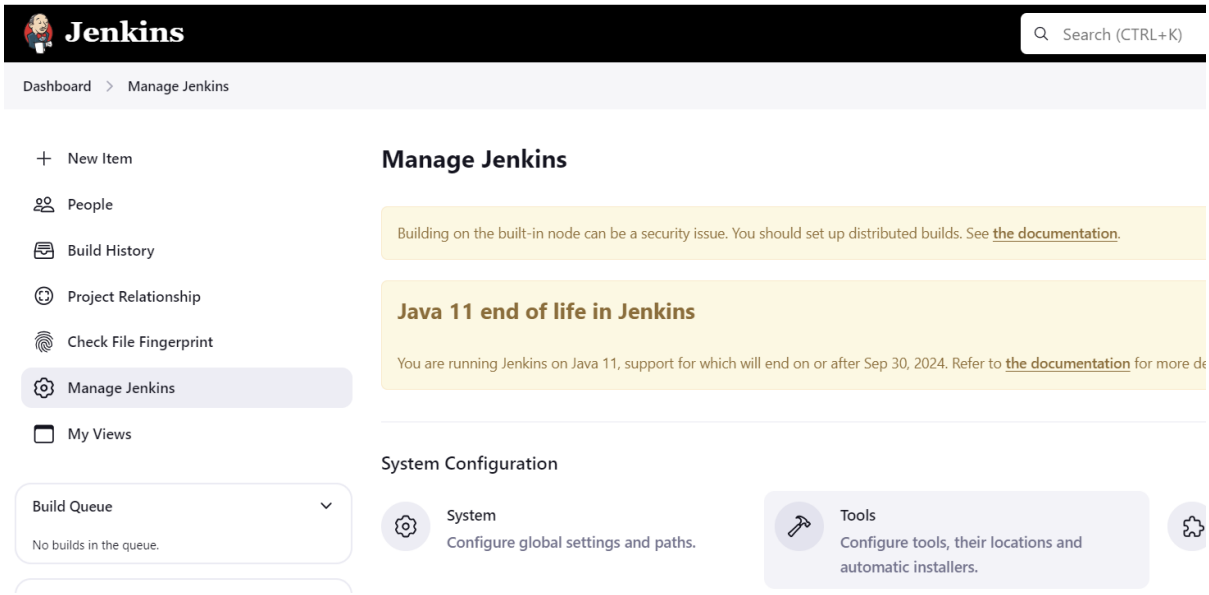
# Ansible deployment on test server

Setting up the above test-server ip address to ansible hosts



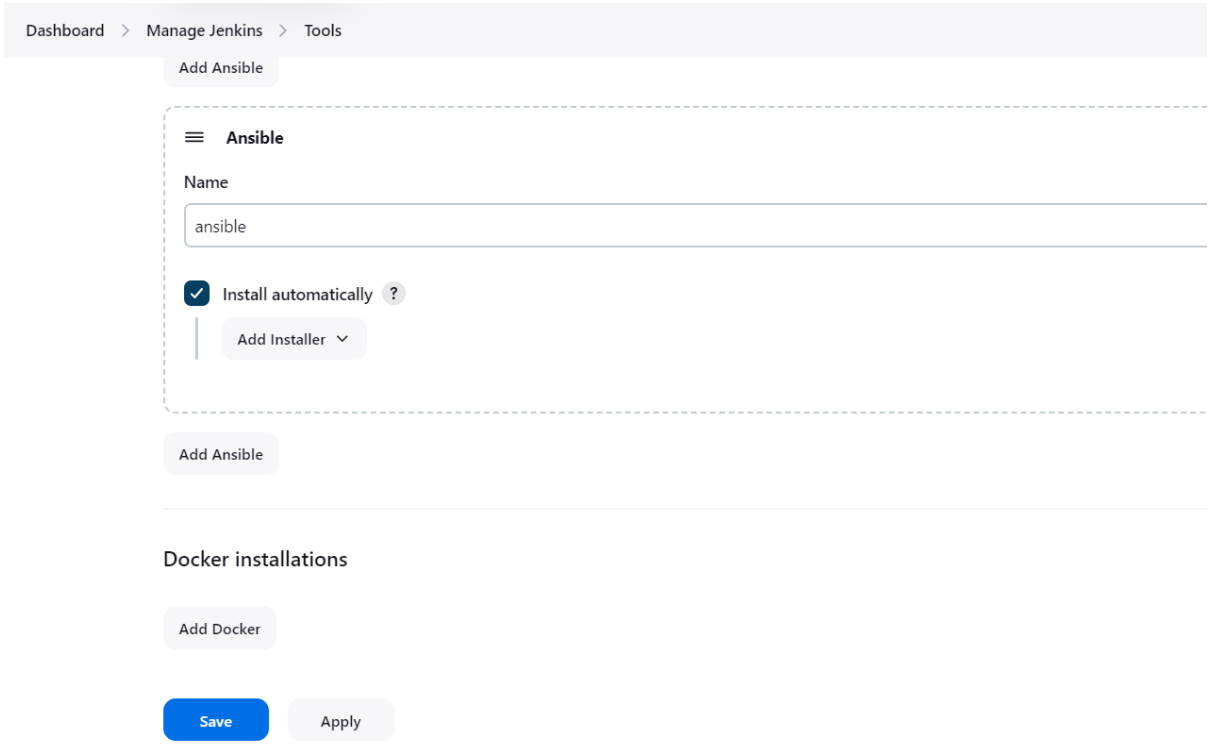




Setting the Jenkins tools for ANSIBLE
Manage Jenkins > tools >

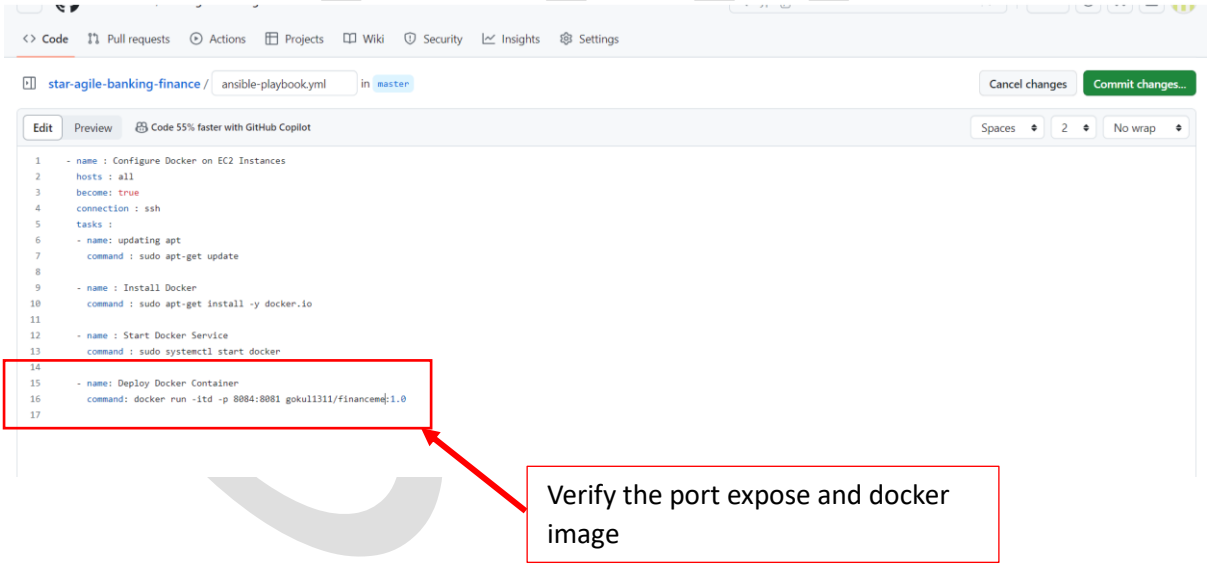





gokul.chandrasekharan01@gmail.com




Verifying ansible playbook in our repository

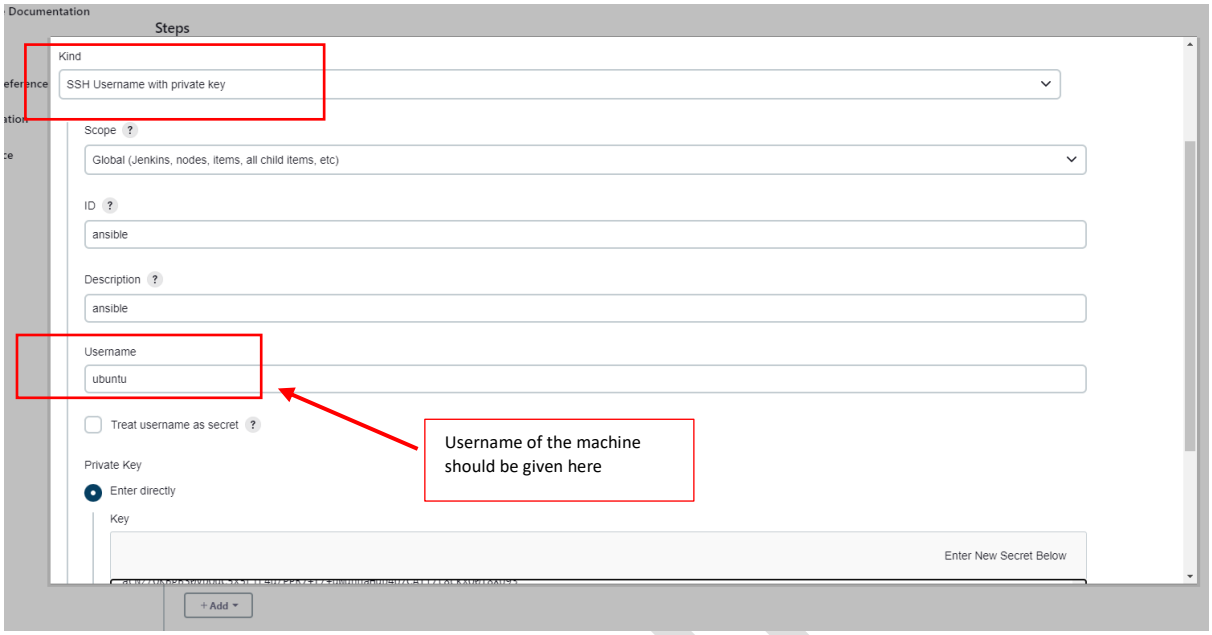


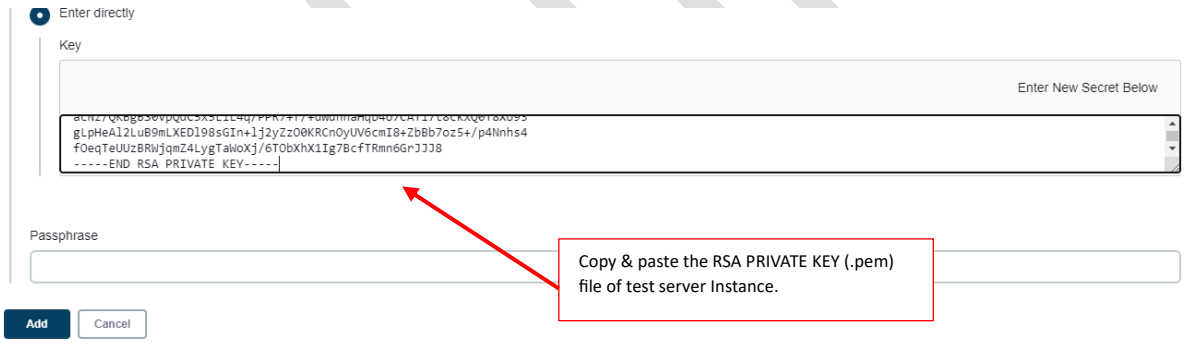

Verify the port expose and docker image

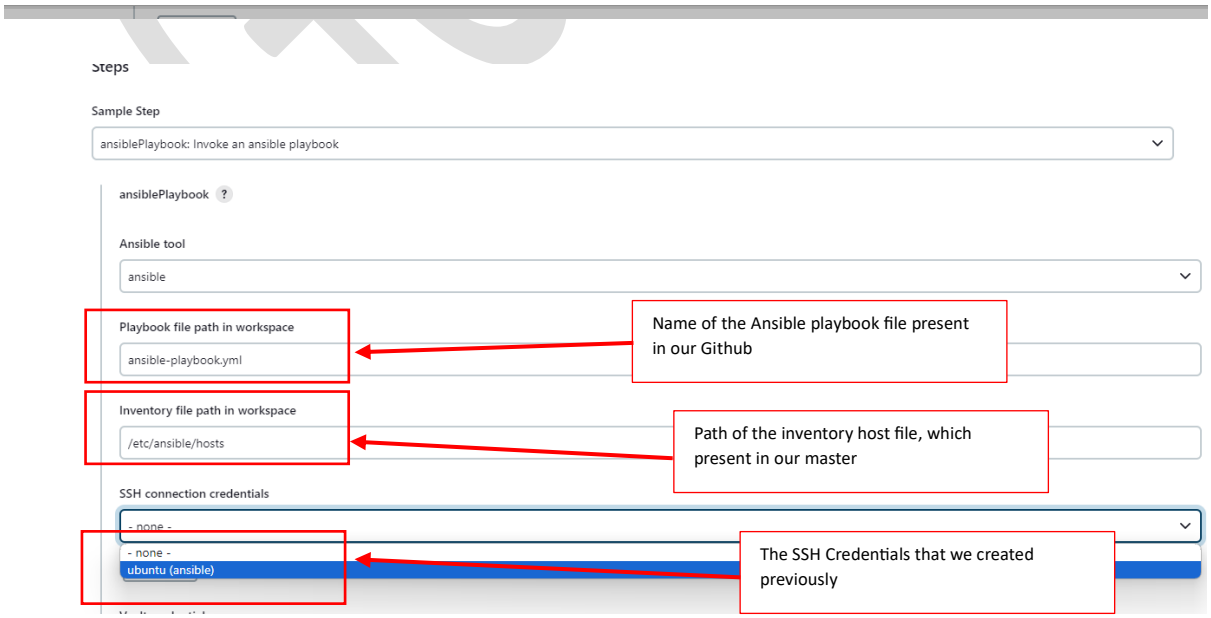


gokul.chandrasekharan01@gmail.com

# Creating the SSH Credentials of test-server and giving to Ansible, so that it can perform the activities on test server

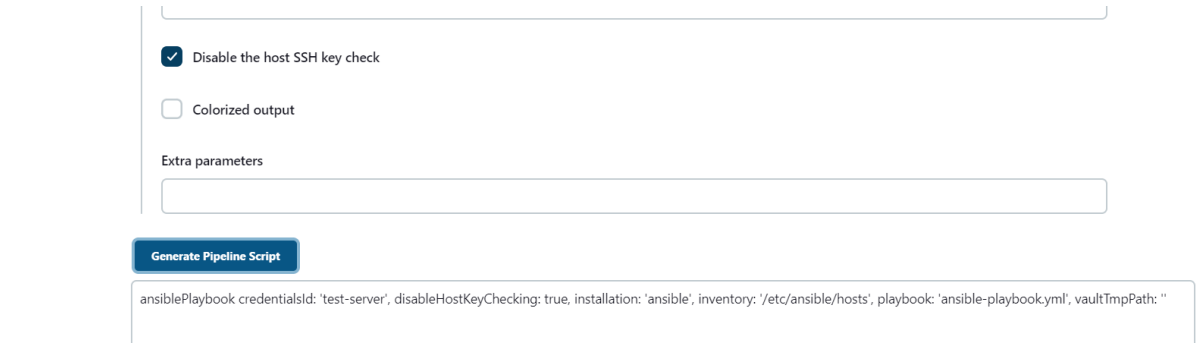


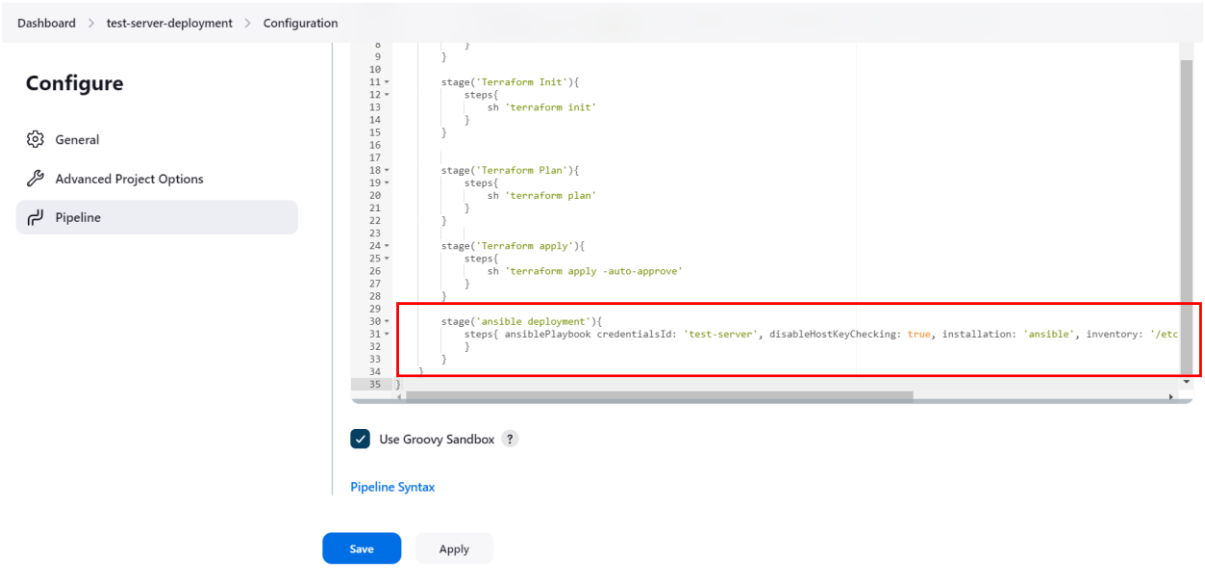



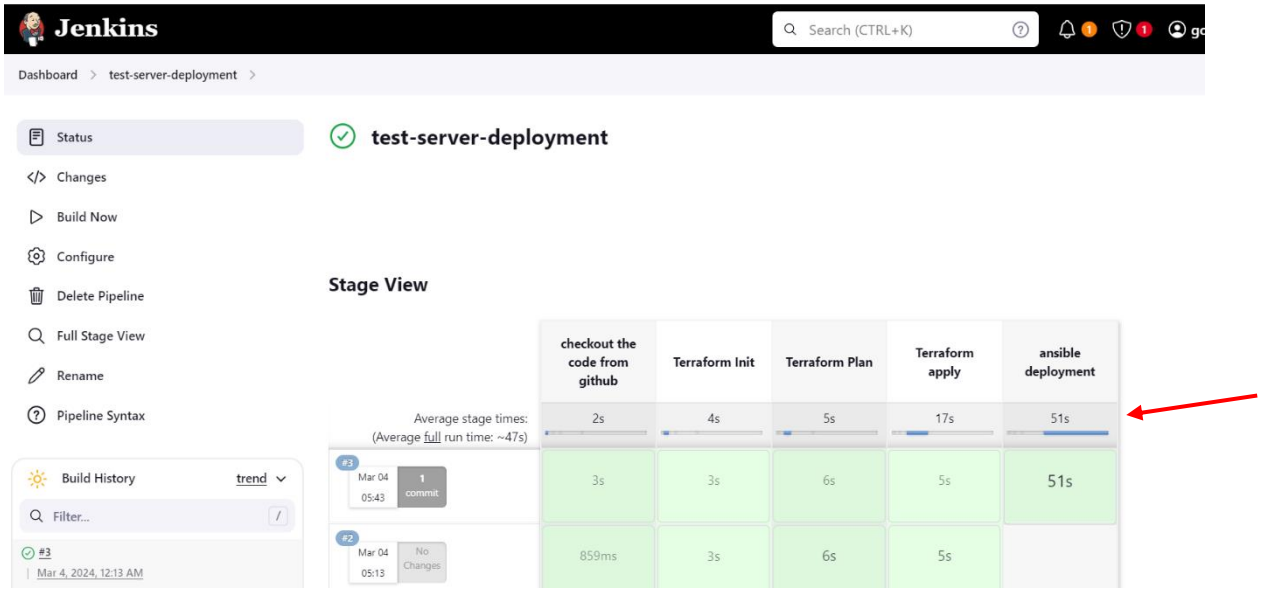






gokul.chandrasekharan01@gmail.com






gokul.chandrasekharan01@gmail.com

test-server-deployment > #3

```
[0m[1m[32mNo changes.[0m[1m Your infrastructure matches the configuration.[0m

[0mTerraform has compared your real infrastructure against your configuration
and found no differences, so no changes are needed.
[0m[1m[32m
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
[0m
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (ansible deployment)
[Pipeline] ansiblePlaybook
[test-server-deployment] $ ansible-playbook ansible-playbook.yml -i /etc/ansible/hosts --private-key /var/lib/jenkins/workspace/test-server-
deployment/ssh16822539509420140445.key -u ubuntu

PLAY [Configure Docker on EC2 Instances] ***********************************

TASK [Gathering Facts] *****************************************************
ok: [13.127.122.89]

TASK [updating apt] ********************************************************
changed: [13.127.122.89]

TASK [Install Docker] ******************************************************
changed: [13.127.122.89]

TASK [Start Docker Service] ************************************************
changed: [13.127.122.89]
```

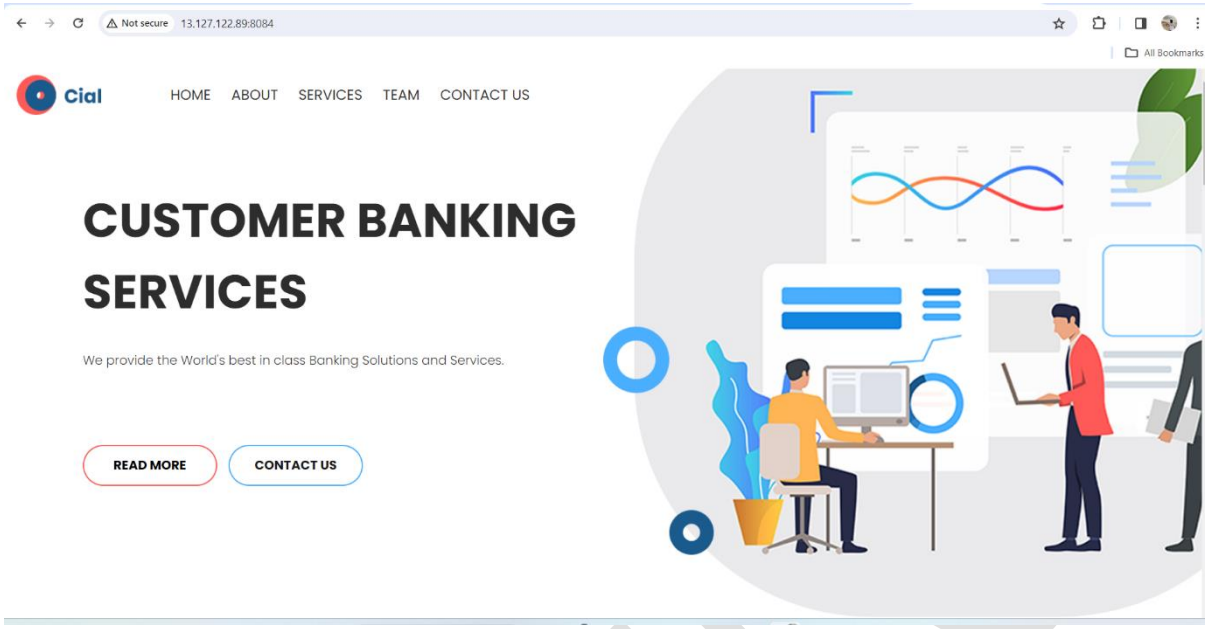
test-server-deployment > #3

```
ok: [13.127.122.89]

TASK [updating apt] ********************************************************
changed: [13.127.122.89]

TASK [Install Docker] ******************************************************
changed: [13.127.122.89]

TASK [Start Docker Service] ************************************************
changed: [13.127.122.89]

TASK [Deploy Docker Container] *********************************************
changed: [13.127.122.89]

PLAY RECAP *****************************************************************
13.127.122.89              : ok=5    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```



gokul.chandrasekharan01@gmail.com

Verifying the ansible deployment using the public IP of test server that we created using terraform and the post expose of our application



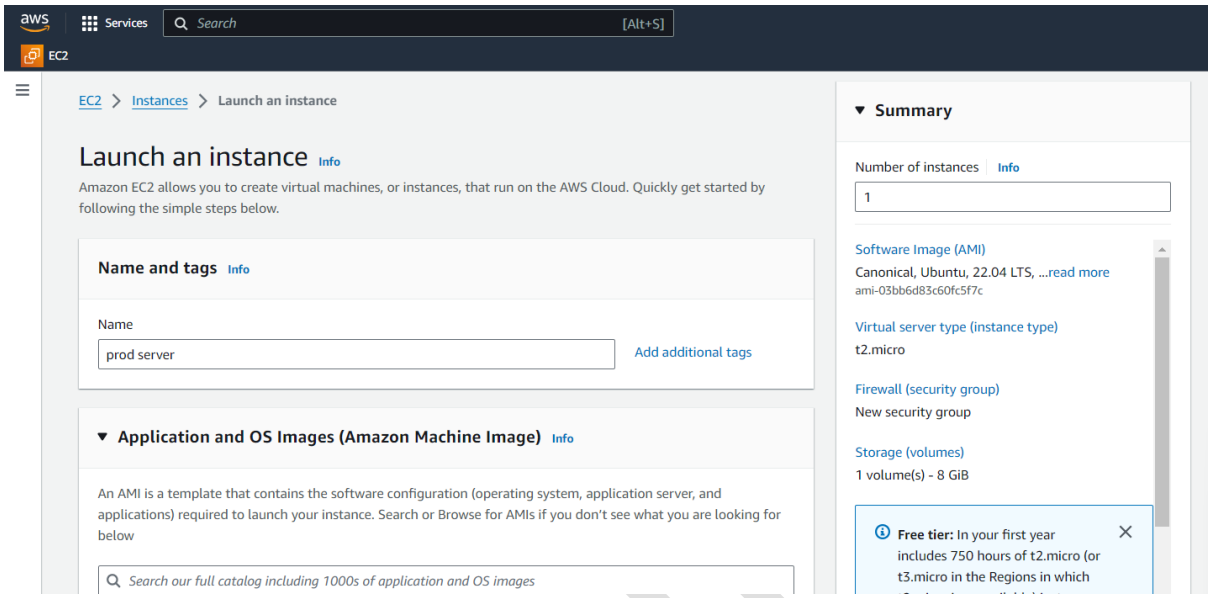


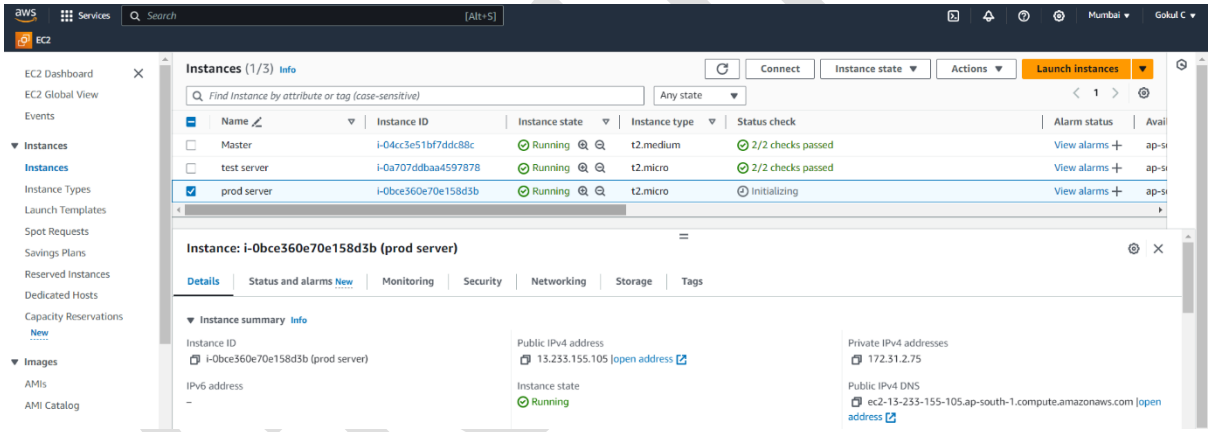



gokul.chandrasekharan01@gmail.com

# Prod server deployment

## Setting the prod server instance



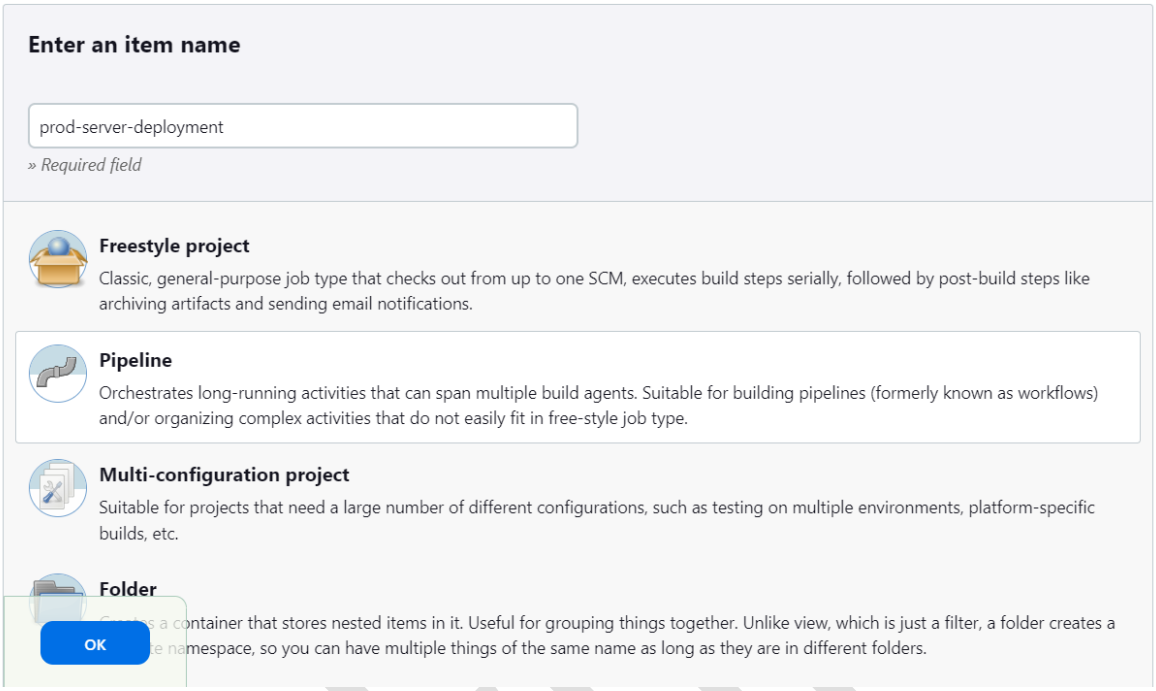






gokul.chandrasekharan01@gmail.com

# Ansible deployment to Prod server

## Creating a new pipeline job for prod server deployment



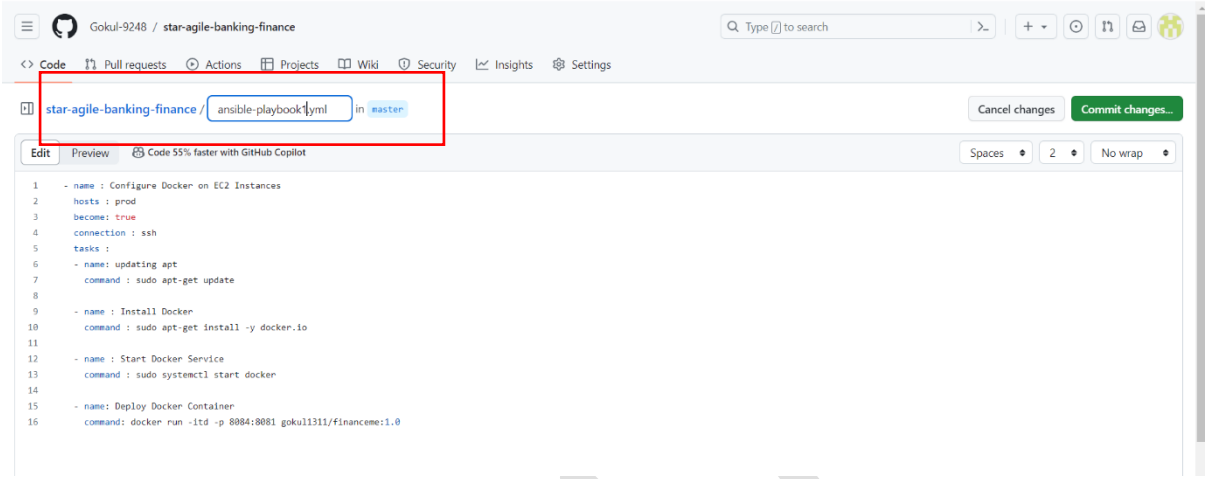

## Before that we have to configure the prod server details to ansible master

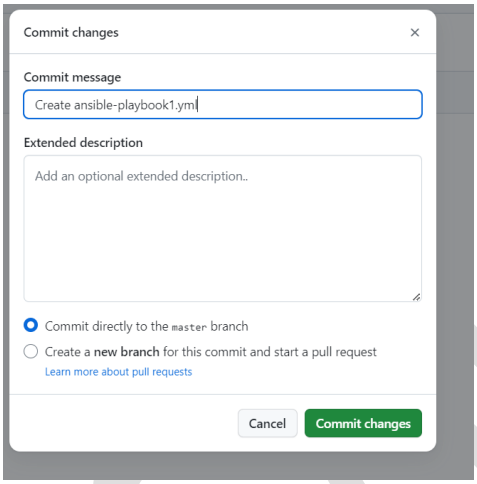
Added the prod server to ansible-hosts as [ prod ]

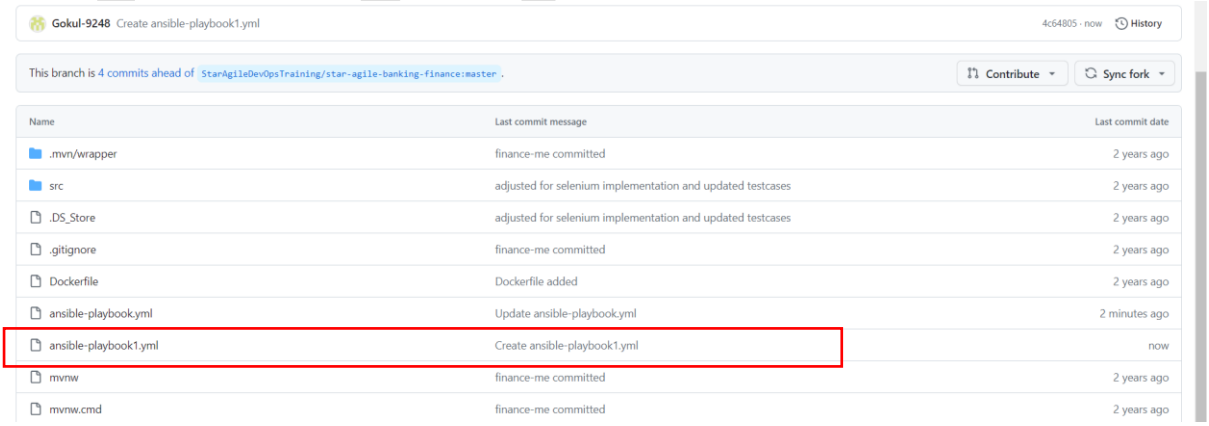



gokul.chandrasekharan01@gmail.com

Created a new ansible playbook for prod server deployment as: ansible-playbook1.yml



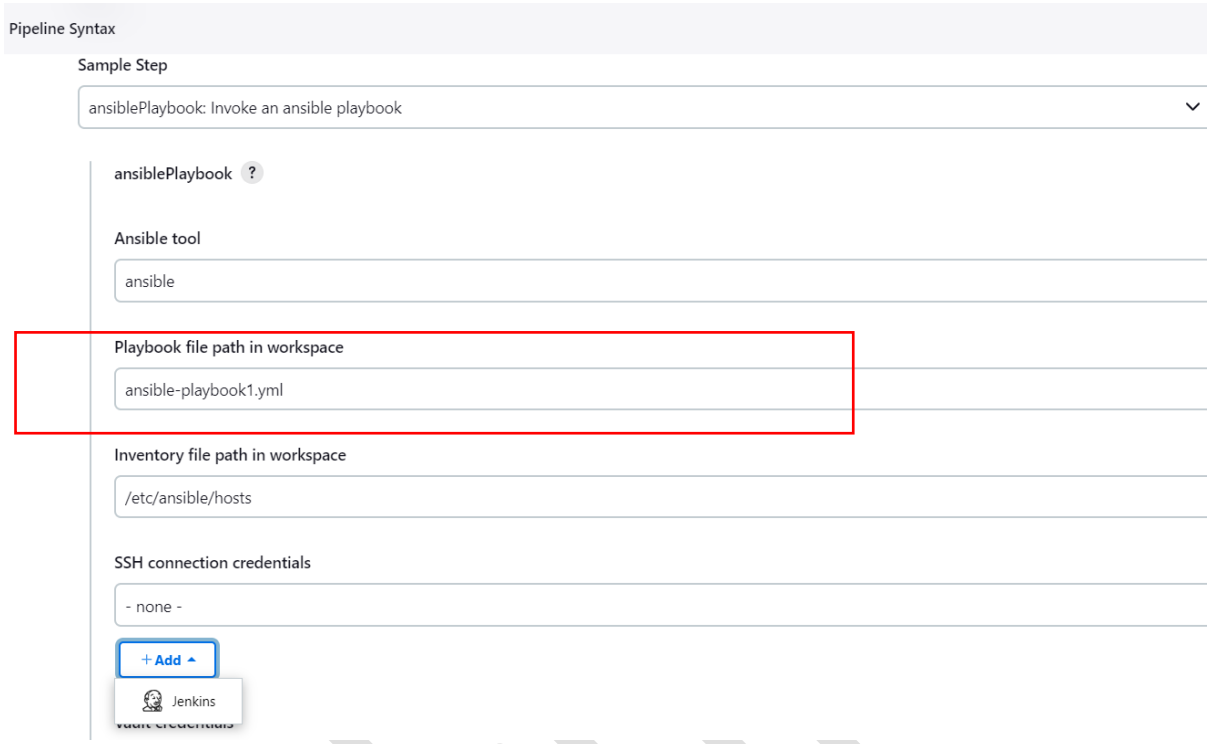






gokul.chandrasekharan01@gmail.com

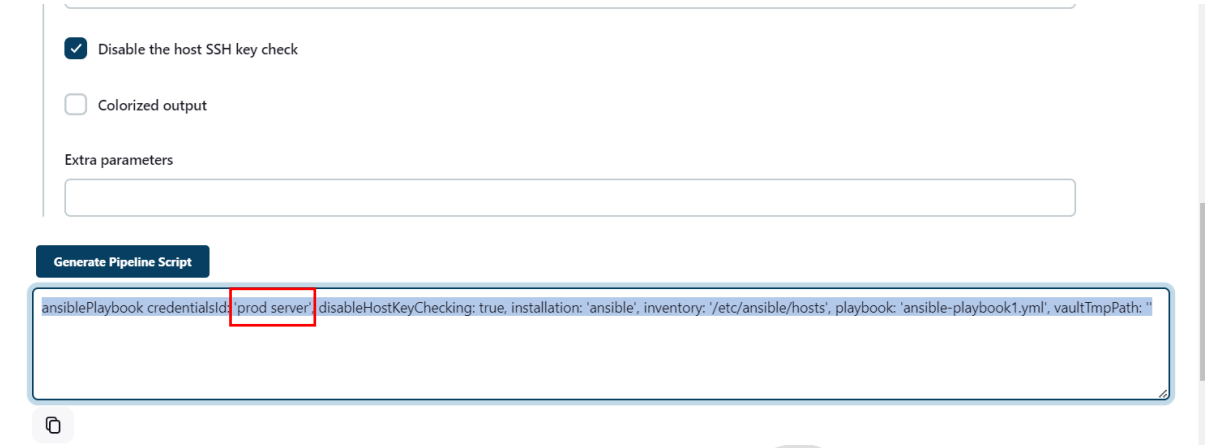
## Creating the SSH Credentials of prod server and giving to Ansible, so that it can perform the activities on prod server

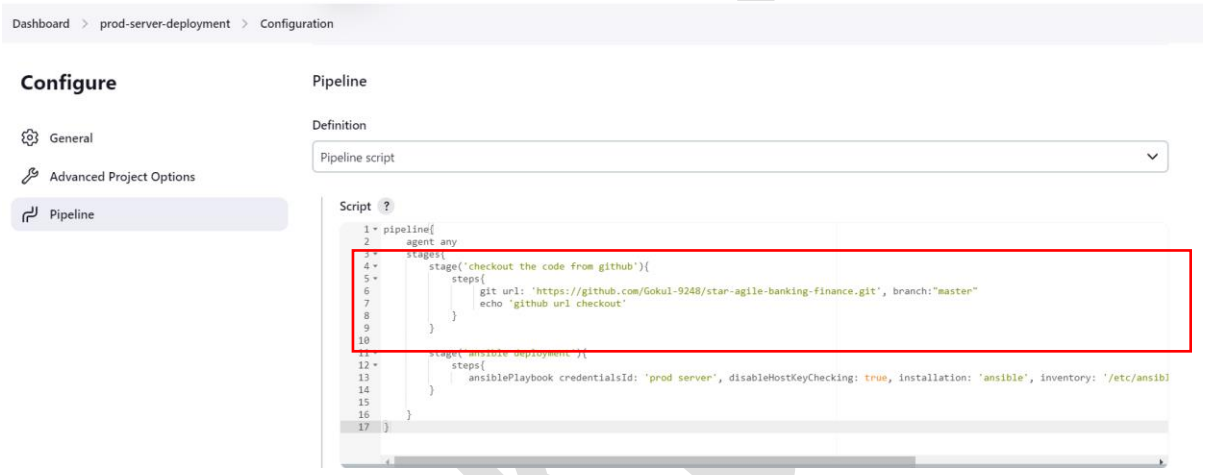


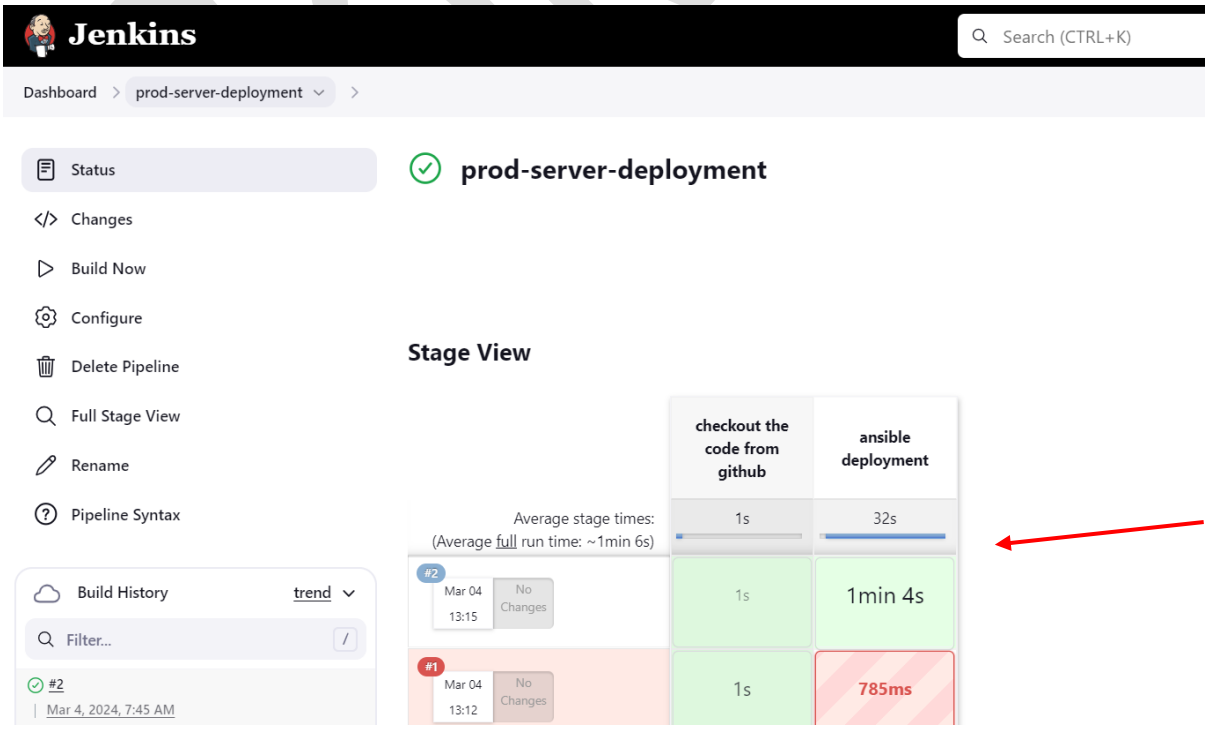




gokul.chandrasekharan01@gmail.com



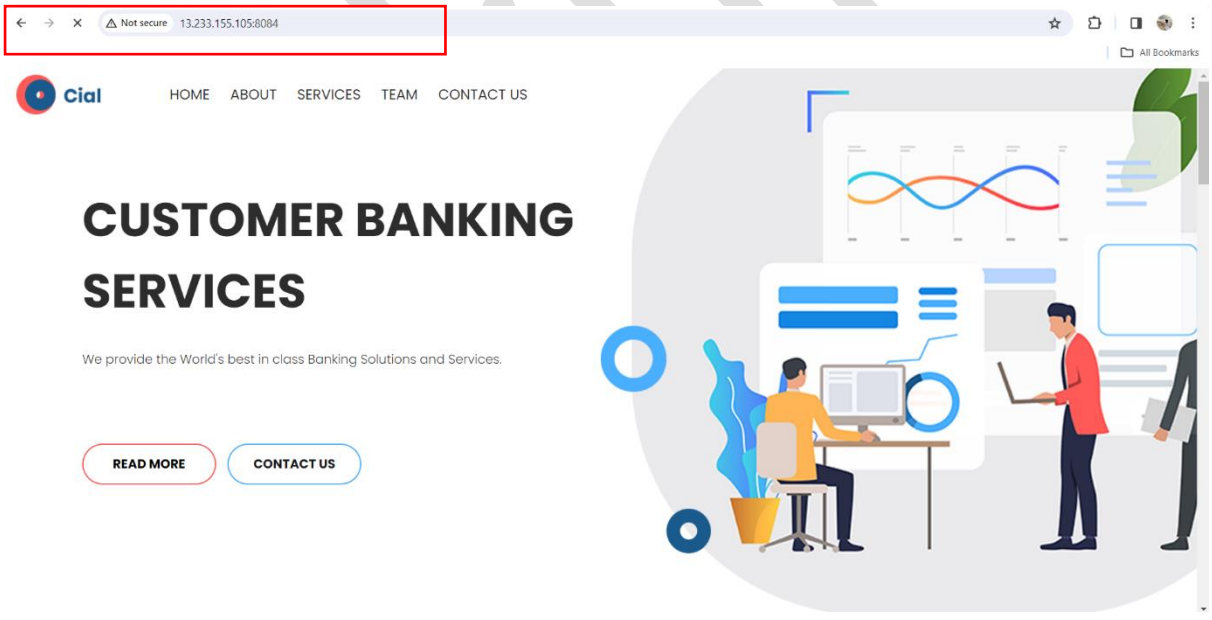






gokul.chandrasekharan01@gmail.com

## Verifying the prod server deployment

```
[Pipeline] ansibleplaybook
[prod-server-deployment] $ ansible-playbook ansible-playbook1.yml -i /etc/ansible/hosts --private-key /var/lib/jenkins/workspace/prod-server-deployment/ssh1580196668429747903.key -u ubuntu

PLAY [Configure Docker on EC2 Instances] ***********************************

TASK [Gathering Facts] *****************************************************
ok: [13.233.155.105]

TASK [updating apt] ********************************************************
changed: [13.233.155.105]

TASK [Install Docker] ******************************************************
changed: [13.233.155.105]

TASK [Start Docker Service] ************************************************
changed: [13.233.155.105]

TASK [Deploy Docker Container] *********************************************
changed: [13.233.155.105]

PLAY RECAP *****************************************************************
13.233.155.105             : ok=5    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

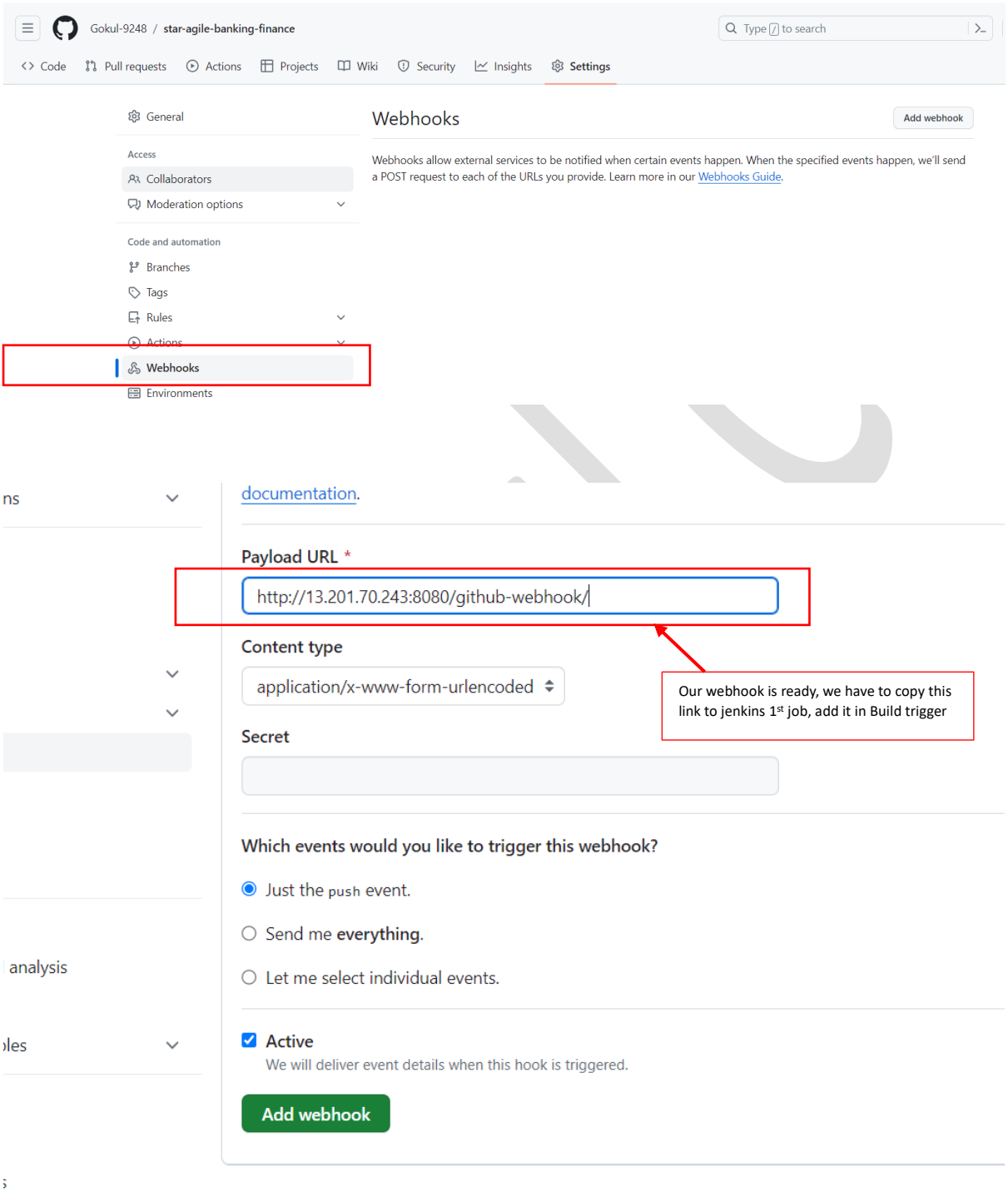





gokul.chandrasekharan01@gmail.com

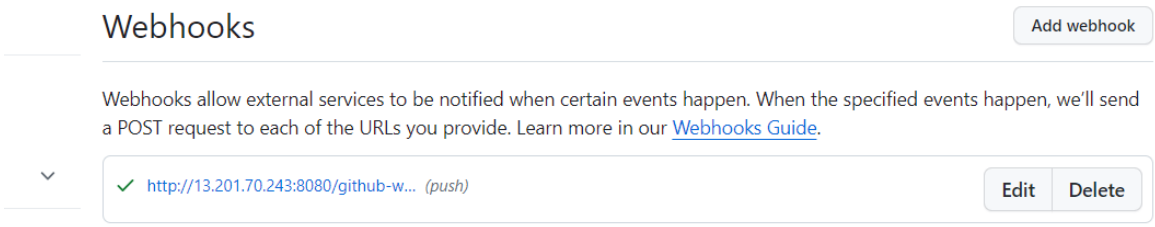
# Automation of workflows

Creating a WEBHOOK to our repo for the automatic triggering actions






gokul.chandrasekharan01@gmail.com




We have to edit our pipeline and Github files so that there won't be any errors caused by duplicate containers

```
45              }
46          }
47          }
48              stage('stop existing containers'){
49              steps{
50                  sh 'docker stop $(docker ps -a -q)'
51              }
52          }
53
54          stage('deleting existing containers'){
55              steps{
56                  sh 'docker rm $(docker ps -a -q)'
57              }
58          }
59
60      stage('containerizing the application'){
61              steps{
62                  sh 'docker run -itd -p 8084:8081 gokul1311/financeme:1.0'
63              }
64          }
65
66      }
67  }
```

star-agile-banking-finance / ansible-playbook.yml in master — Cancel changes

Edit | Preview | Code 55% faster with GitHub Copilot — Spaces 2

```
1   - name : Configure Docker on EC2 Instances
2     hosts : demo
3     become: true
4     connection : ssh
5     tasks :
6     - name: updating apt
7       command : sudo apt-get update
8
9     - name : Install Docker
10      command : sudo apt-get install -y docker.io
11
12    - name : Start Docker Service
13      command : sudo systemctl start docker
14
15    - name : stop existing containers
16      shell: docker stop $(docker ps -a -q)
17
18    - name : delete all containers
19      shell: docker rm $(docker ps -a -q)
20
21
22    - name: Deploy Docker Container
23      command: docker run -itd -p 8084:8081 gokul1311/financeme:1.0
24
```

Use Control + Shift + m to toggle the tab key moving focus. Alternatively, use esc then tab to move to the next interactive element on the page.



gokul.chandrasekharan01@gmail.com

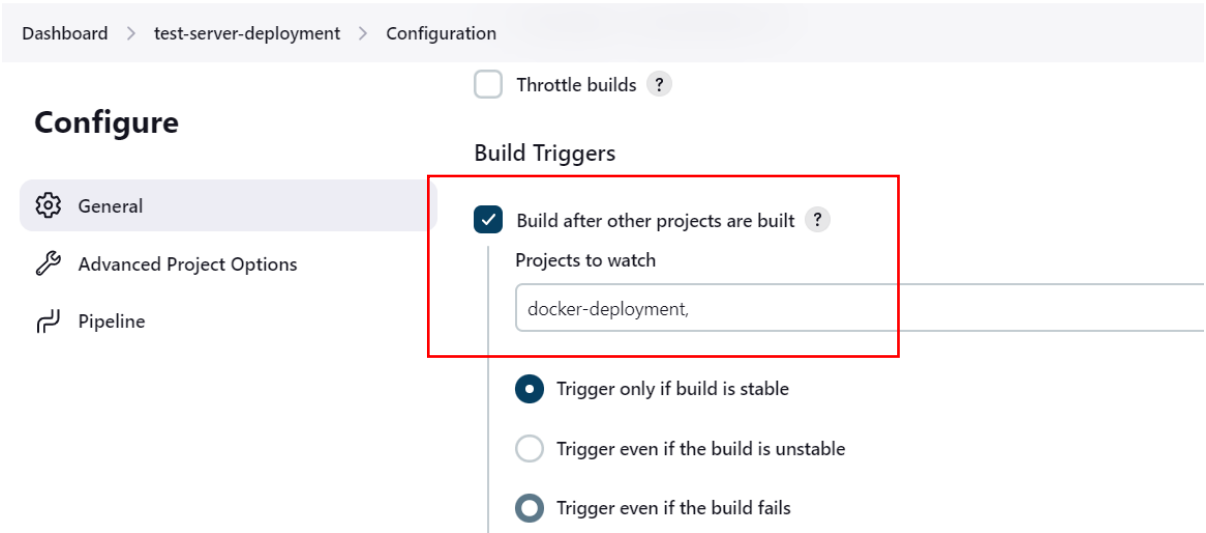
star-agile-banking-finance / ansible-playbook1.yml in master

Edit Preview Code 55% faster with GitHub Copilot

```
- name : Configure Docker on EC2 Instances
  hosts : prod
  become: true
  connection : ssh
  tasks :
  - name: updating apt
    command : sudo apt-get update

  - name : Install Docker
    command : sudo apt-get install -y docker.io

  - name : Start Docker Service
    command : sudo systemctl start docker

  - name : stop existing containers
    shell: docker stop $(docker ps -a -q)

  - name : delete all containers
    shell: docker rm $(docker ps -a -q)

  - name: Deploy Docker Container
    command: docker run -itd -p 8084:8081 gokul1311/financeme:1.0

```

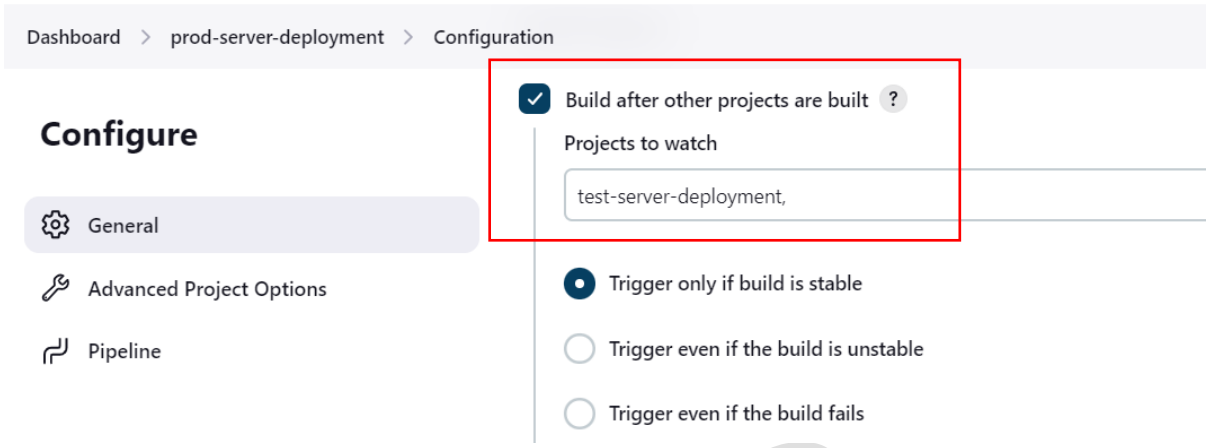
## Build triggers our Jenkins job

Build triggers in the order 1st 2nd and 3rd



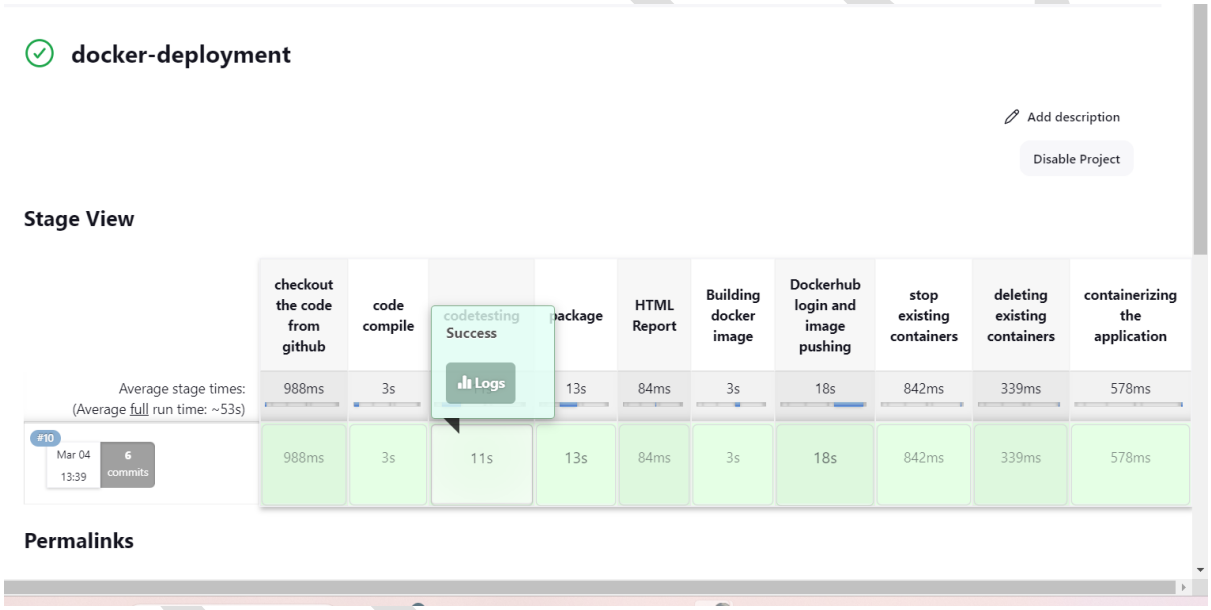



gokul.chandrasekharan01@gmail.com




## Verifying the workflow by pushing a commit to the github

Our 1st job is triggered automatically by the github push

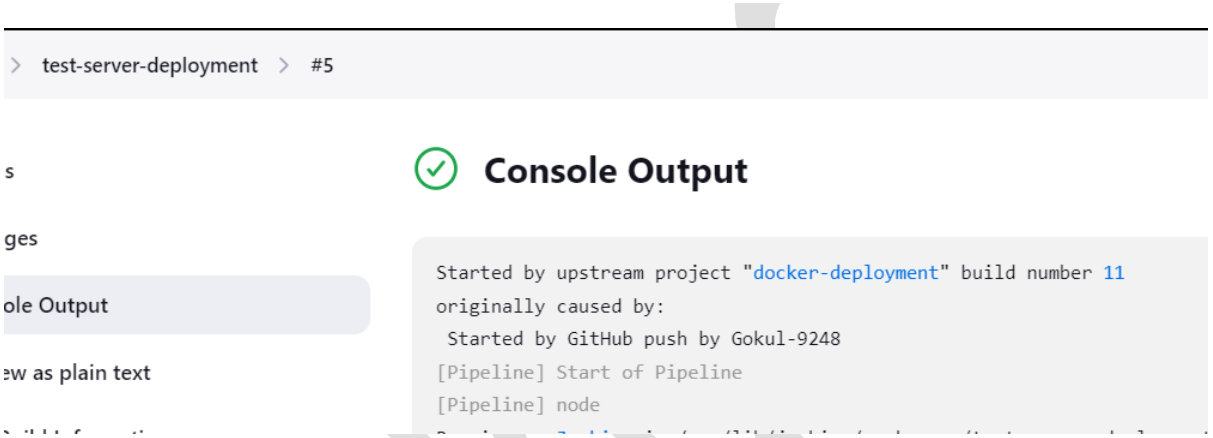




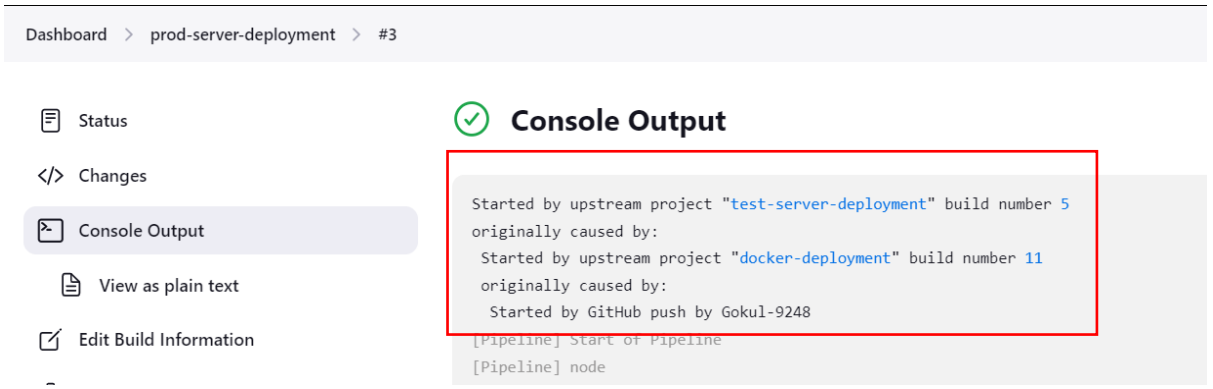


gokul.chandrasekharan01@gmail.com

```
+ docker run -itd -p 8084:8081 gokul1311/financeme:1.0
2841d05c8279956dedf05d5ac6e2a794bfbc4464d8c72b06a51c76ad8fb8dcb6
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Triggering a new build of test-server-deployment #5
Finished: SUCCESS
```




```
PLAY RECAP *********************************************************************
13.127.122.89              : ok=7    changed=6    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Triggering a new build of prod-server-deployment #3
Finished: SUCCESS
```

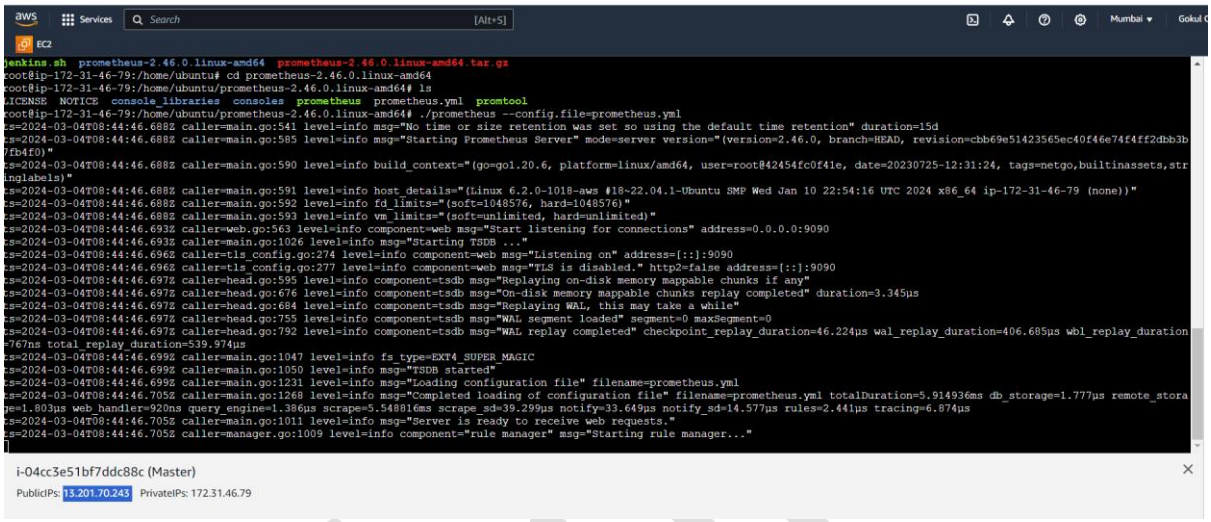




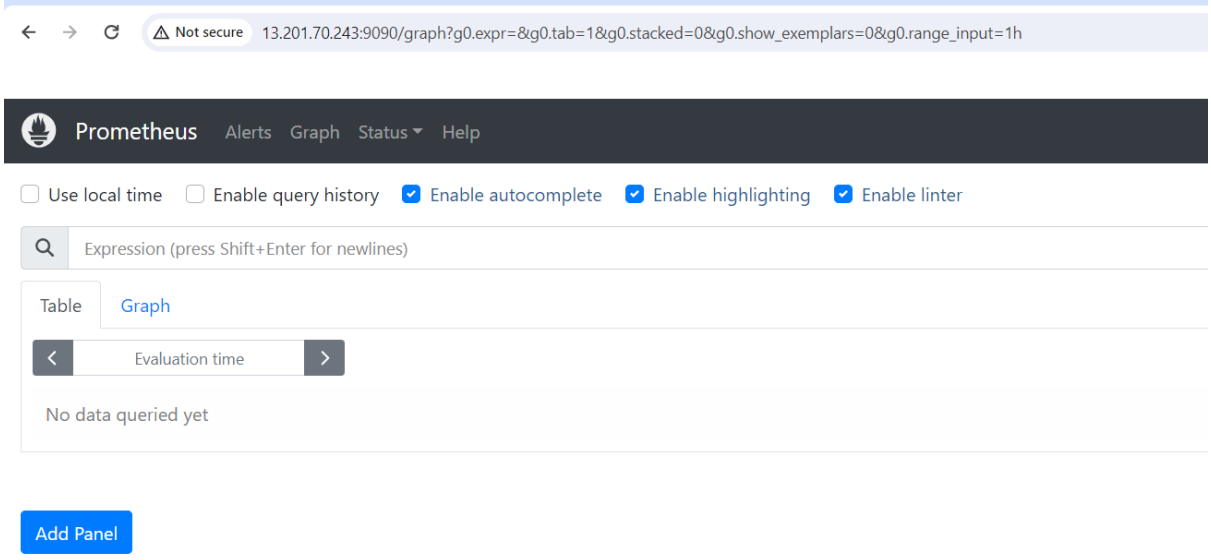

gokul.chandrasekharan01@gmail.com

# Continuous monitoring using PROMETHEUS & GRAFANA

## PROMETHEUS INSTALLATION




## Checking the Prometheus on default port 9090






gokul.chandrasekharan01@gmail.com

## NODE EXPORTER Installation in both test server & prod server

```
root@ip-10-0-1-10:/home/ubuntu# docker run -d -p 9100:9100 --name=node_exporter prom/node-exporter
Unable to find image 'prom/node-exporter:latest' locally
latest: Pulling from prom/node-exporter
2abcce694348: Pull complete
455fd88e5221: Pull complete
324153f2810a: Pull complete
Digest: sha256:4cb2b9019f1757be8482419002cb7afe028fdba35d47958829e4cfeaf6246d80
Status: Downloaded newer image for prom/node-exporter:latest
39111f612066b136abd1717cdabd3e4893812277f1362782907a4baeebfbd8ce
root@ip-10-0-1-10:/home/ubuntu#
```

i-0a707ddbaa4597878 (test server)

PublicIPs: 13.127.122.89 PrivateIPs: 10.0.1.10

```
root@ip-172-31-2-75:/home/ubuntu# docker run -d -p 9100:9100 --name=node_exporter prom/node-exporter
Unable to find image 'prom/node-exporter:latest' locally
latest: Pulling from prom/node-exporter
2abcce694348: Pull complete
455fd88e5221: Pull complete
324153f2810a: Pull complete
Digest: sha256:4cb2b9019f1757be8482419002cb7afe028fdba35d47958829e4cfeaf6246d80
Status: Downloaded newer image for prom/node-exporter:latest
c1d01cd9b048dd5a90209f9616a42446e25c284f5ca19f3955188513c8437d7d
root@ip-172-31-2-75:/home/ubuntu#
```

i-0bce360e70e158d3b (prod server)

PublicIPs: 13.233.155.105 PrivateIPs: 172.31.2.75

## Verifying in node exporter. default port 9100


# Node Exporter

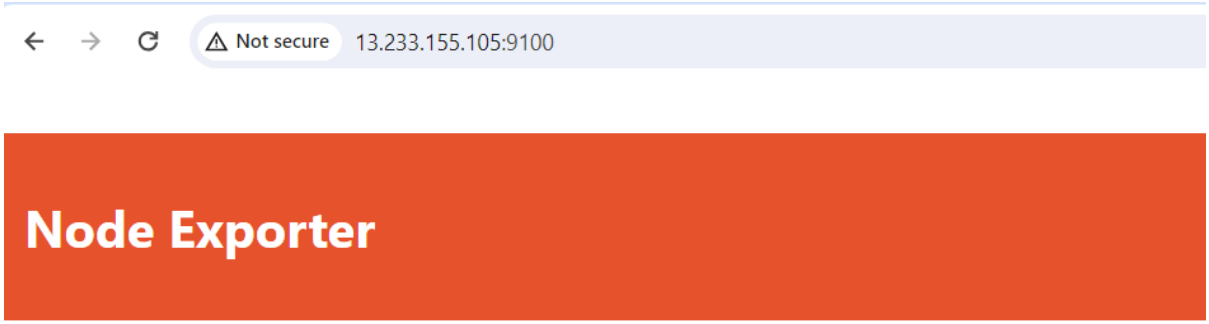
### Prometheus Node Exporter

Version: (version=1.7.0, branch=HEAD, revision=7333465abf9efba81876303bb57e6fadb946041b)

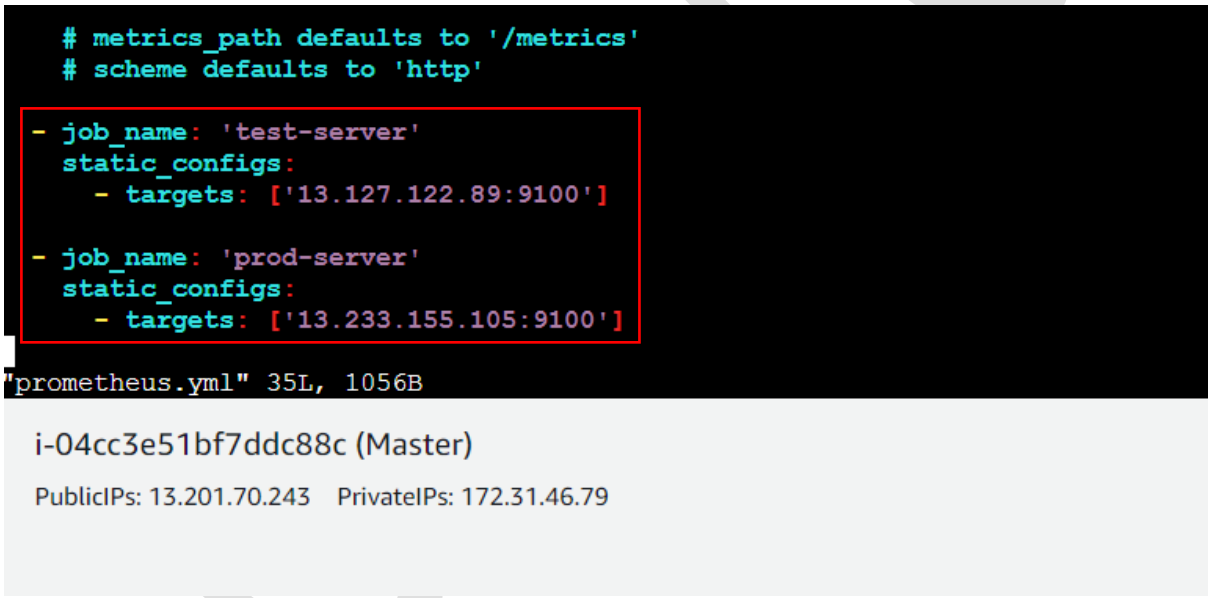
- Metrics

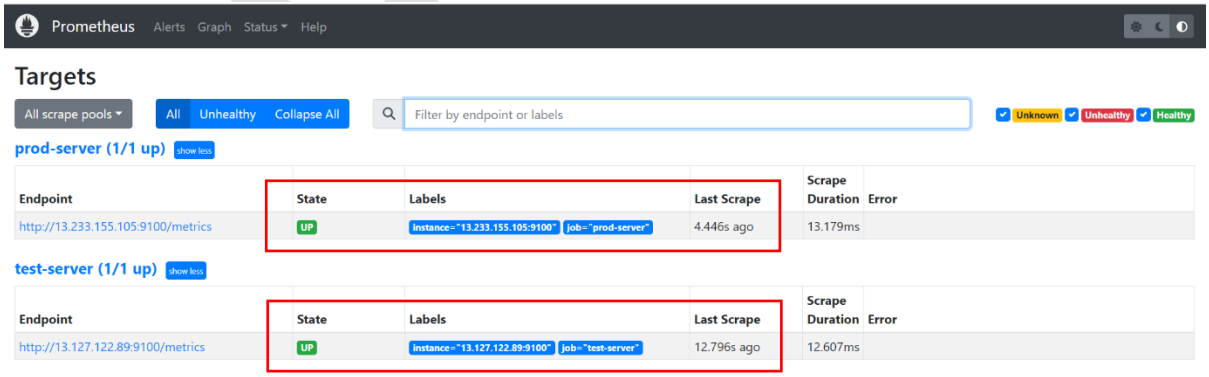


gokul.chandrasekharan01@gmail.com




## Configuring the Prometheus targets

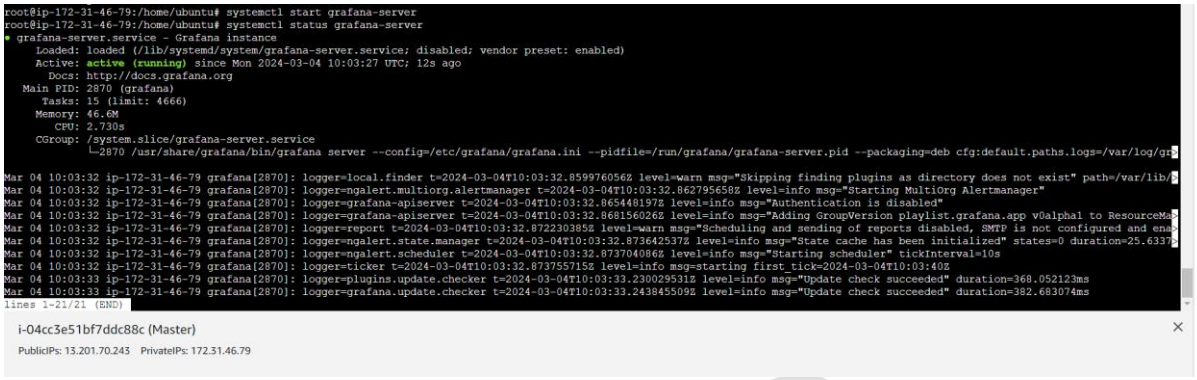


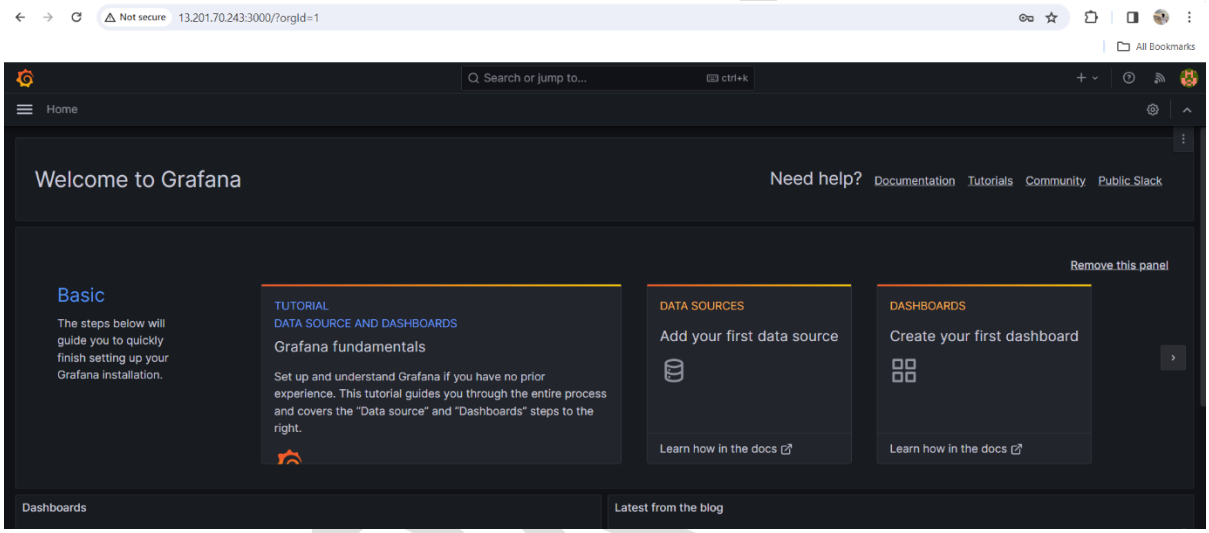



gokul.chandrasekharan01@gmail.com

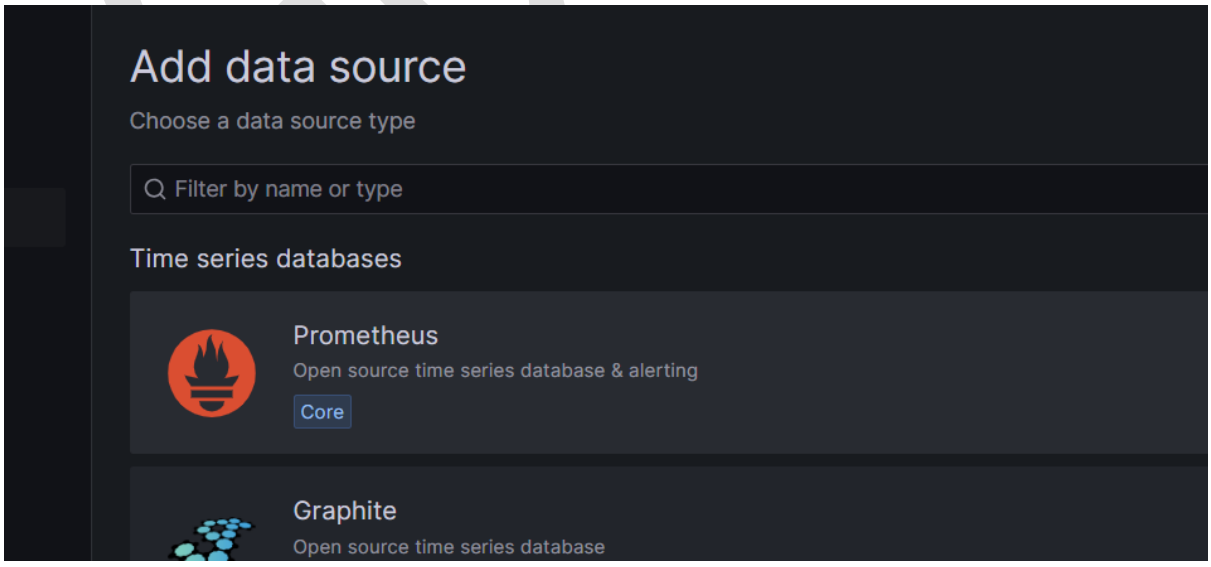
## GRAFANA INSTALLATION




## Accessing Grafana on default port 3000



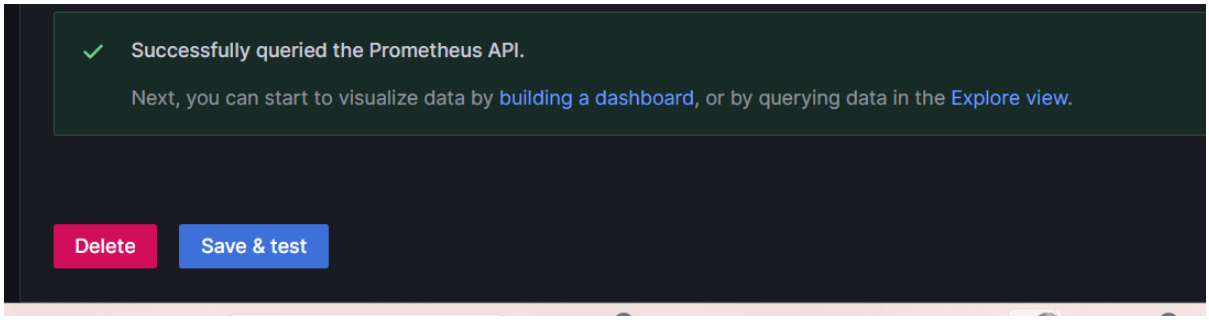

## Adding Prometheus as Data source





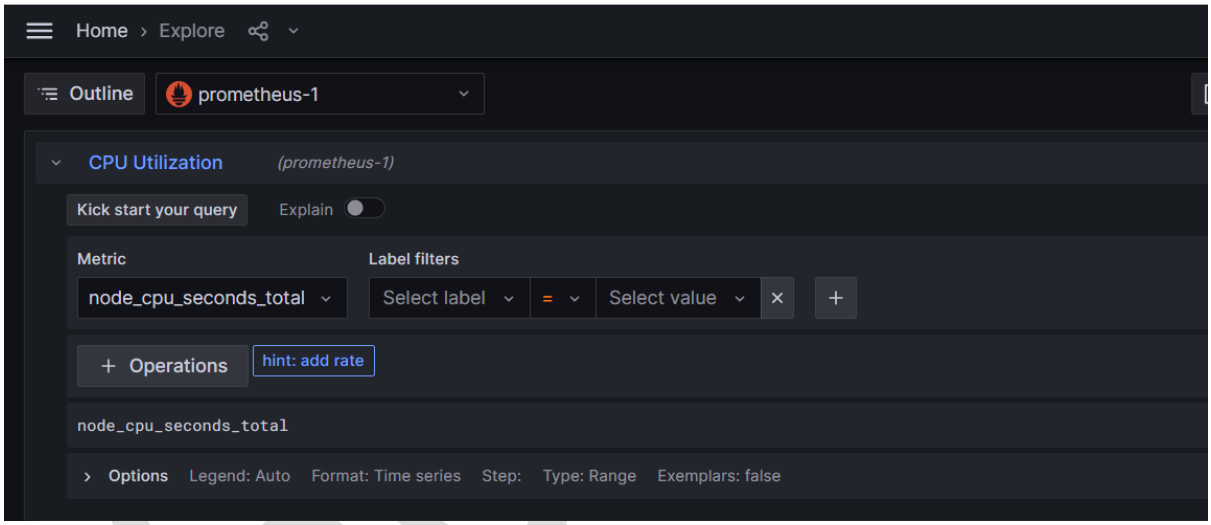

gokul.chandrasekharan01@gmail.com

We have to give Prometheus url as the data source




# Dashboard creations

## 1. CPU Utilization

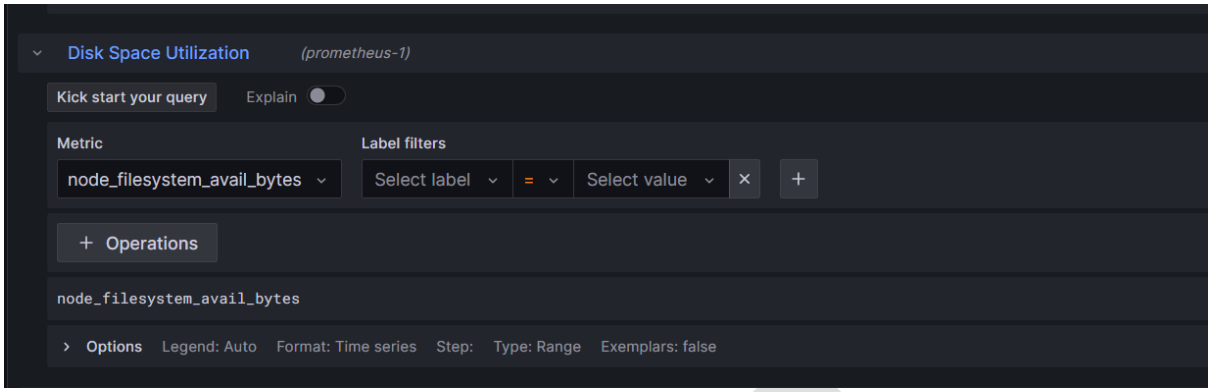


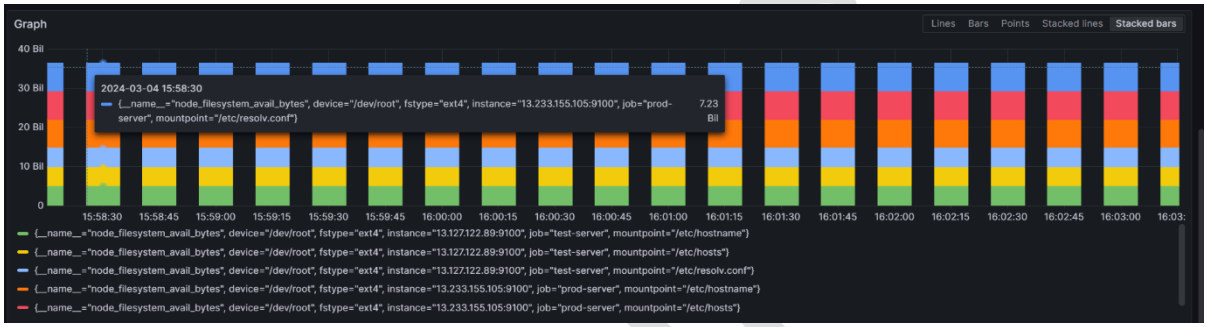




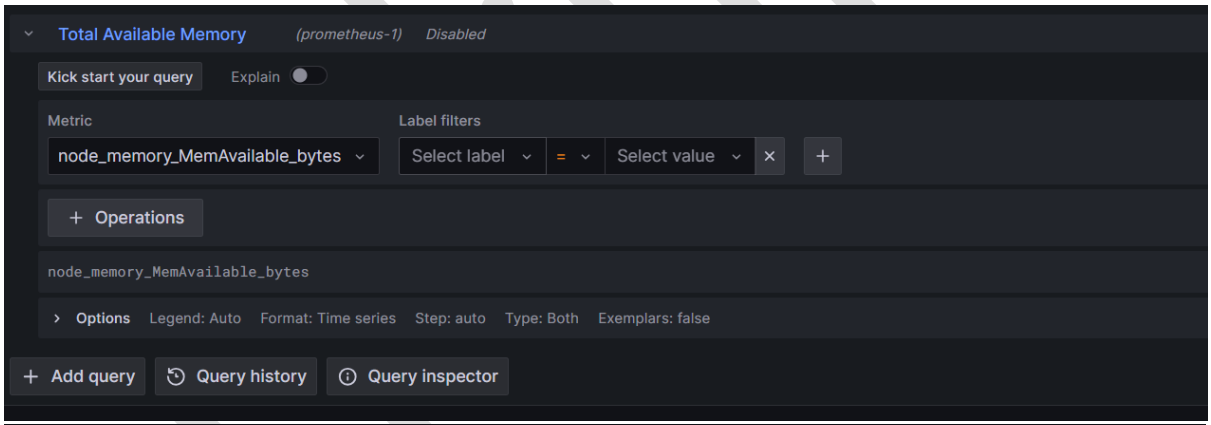
gokul.chandrasekharan01@gmail.com

## 2. Disk space utilization




## 3. Total available memory







gokul.chandrasekharan01@gmail.com