

CAPSTONE PROJECT

**Banking and Finance Domain
Project – FinanceMe**

**SUBMITTED BY: GOKUL C
DATE: 05-03-2024**

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)

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.

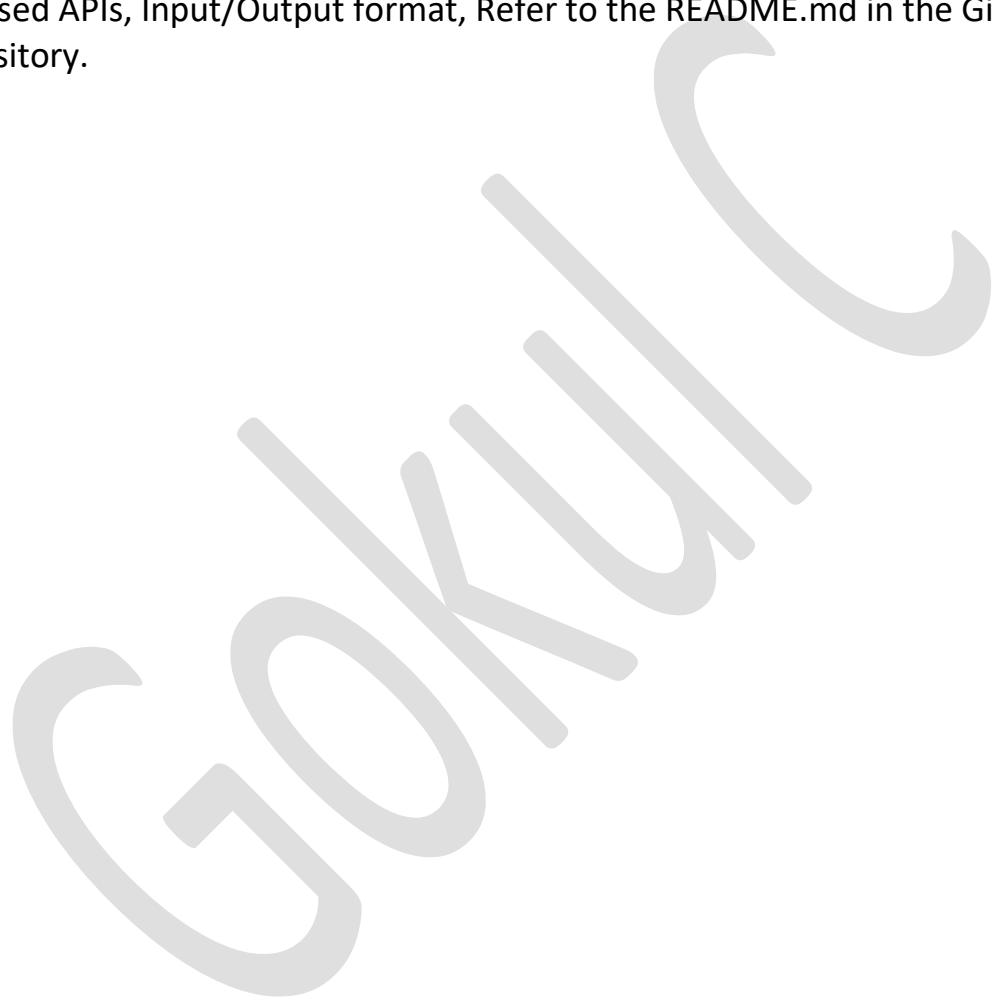
The deployment should then be tested using a test automation tool, and if the build is Successful, Prod server must be configured with all the software it should be pushed to the prod server. All this should happen automatically and should be triggered from a push to the GitHub master branch. Continuous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

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.



PROJECT SOLUTION

Creating master instance to perform the above project

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with options like EC2 Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations (New), Images, AMIs, AMI Catalog, and Elastic Block Store. The main area displays a table titled 'Instances (1/1) Info'. It shows one instance: 'Master' (Instance ID: i-04cc3e51bf7ddc88c, Status: Running, Instance type: t2.medium). Below the table, there's a detailed view for the 'Master' instance, showing its instance summary, network interfaces (Public IPv4 address: 13.201.70.243, Private IP DNS name: ip-172-31-46-79.ap-south-1.compute.internal), and other details like Hostname type, IP name, and Answer private resource DNS name.

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-0ubuntu22.04.1)
OpenJDK 64-Bit Server VM (build 11.0.22+7-post-Ubuntu-0ubuntu22.04.1, 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

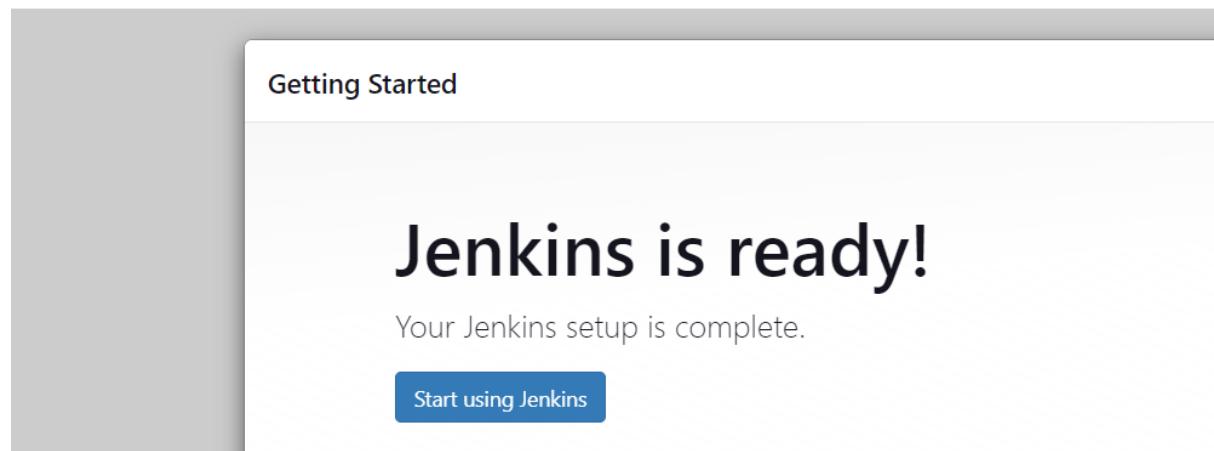
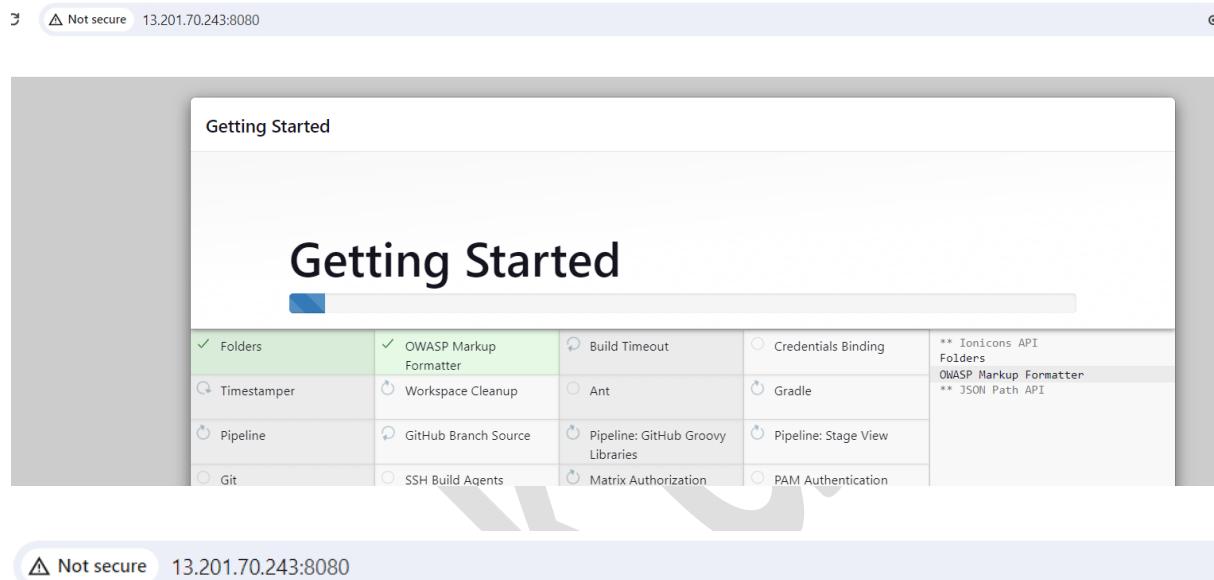
JENKINS INSTALLATION

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

i-04cc3e51bf7ddc88c (Master)

Public IPs: 13.201.70.243 Private IPs: 172.31.46.79

Opening in default port 8080



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)

Public IPs: 13.201.70.243 Private IPs: 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)

Public IPs: 13.201.70.243 Private IPs: 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)

Public IPs: 13.201.70.243 Private IPs: 172.31.46.79

Docker deployment

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

The screenshot shows a CI/CD pipeline named "docker-deployment". At the top, there is a code editor window displaying the Jenkinsfile (Jenkins Pipeline script). The pipeline consists of four stages: "checkout the code from github", "code compile", "codetesting", and "package". Each stage contains specific Jenkins commands like git checkout, mvn compile, mvn test, and mvn clean package.

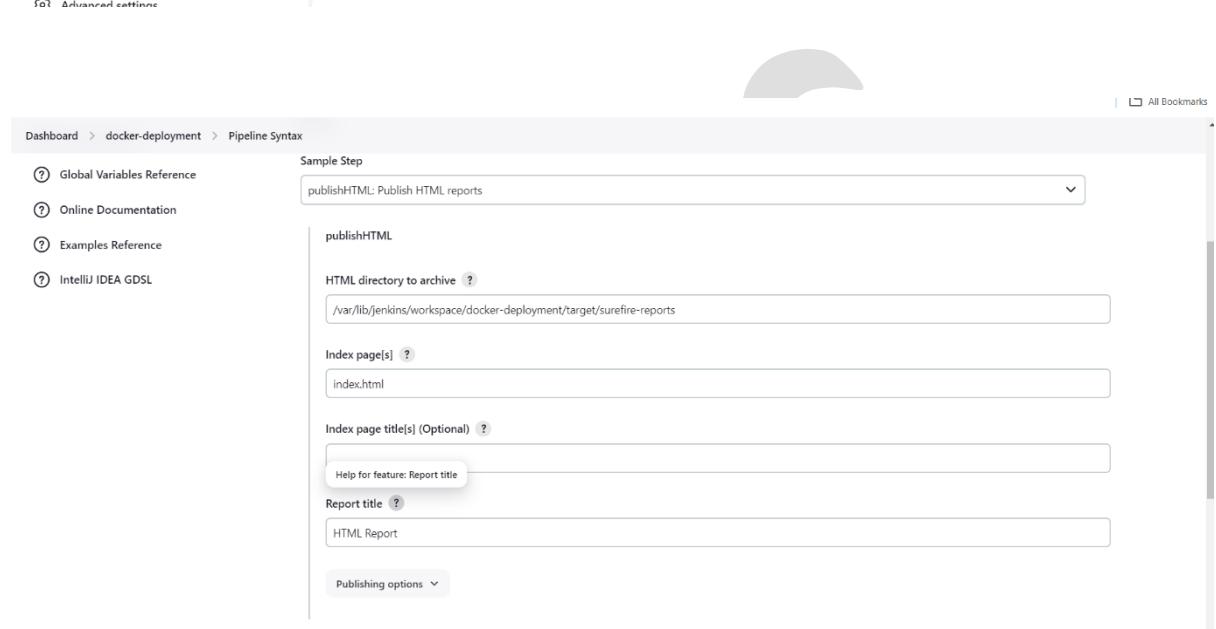
Below the code editor is a navigation bar with links: Dashboard > docker-deployment >. The main area displays the pipeline's status. It shows the pipeline is currently running, with the last build being #1, which started on Mar 04 at 01:48 and has "No Changes". The pipeline has four stages: "checkout the code from github" (1s), "code compile" (10s), "codetesting" (14s), and "package" (16s). A "Stage View" table provides a summary of these times. On the left, there are various management options: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Rename, and Pipeline Syntax. Below these are sections for Build History (with a trend dropdown) and Permalinks.

HTML Report creation

Goto Manage Jenkins and install some necessary plugins



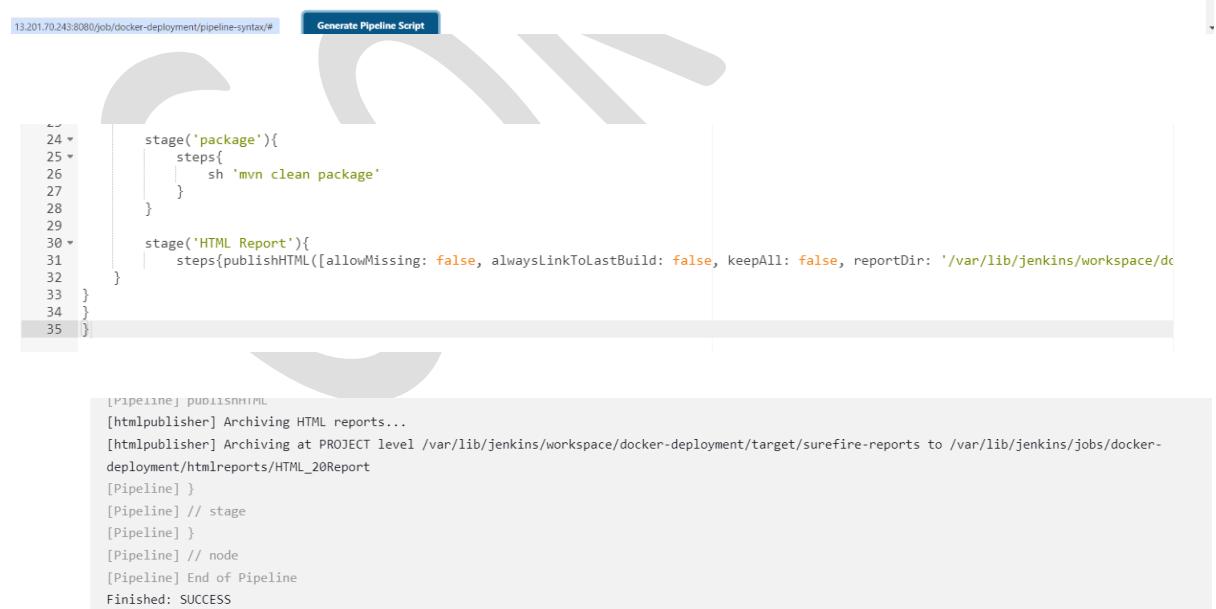
The screenshot shows the Jenkins Manage Plugins page. A search bar at the top right contains the text "html". Below it, a table lists the "HTML Publisher" plugin by "Build Reports". The plugin version is 1.32, it was released 7 months and 2 days ago, and its description is "This plugin publishes HTML reports." An "Install" button is visible on the right.



The screenshot shows the Jenkins Pipeline Syntax configuration for a job named "docker-deployment". Under the "Sample Step" section, a "publishHTML" step is selected. The configuration includes:

- HTML directory to archive: "/var/lib/jenkins/workspace/docker-deployment/target/surefire-reports"
- Index page(s): "index.html"
- Index page title(s) (Optional): "Report title"
- Report title: "HTML Report"

A "Publishing options" dropdown is also present.



The screenshot shows the generated Pipeline Script and its execution log. The script is as follows:

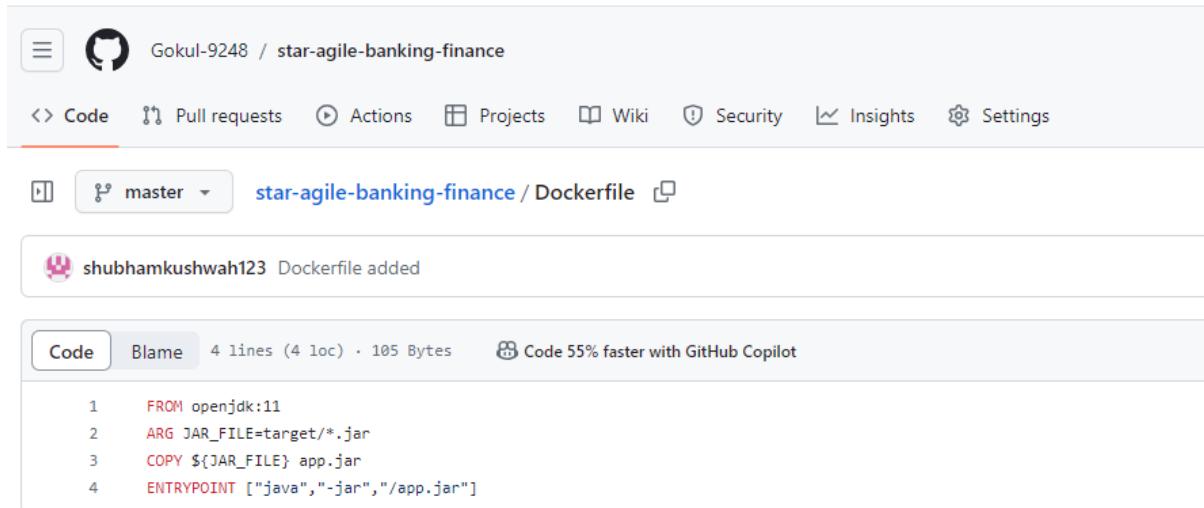
```
13.201.70.243:8080/job/docker-deployment/pipeline-syntax/# 13.201.70.243:8080/job/docker-deployment/pipeline-syntax/# Generate Pipeline Script
24 *
25 stage('package'){
26   steps{
27     sh 'mvn clean package'
28   }
29 }
30 *
31 stage('HTML Report'){
32   steps{publishHTML([allowMissing: false, alwaysLinkToLastBuild: false, keepAll: false, reportDir: '/var/lib/jenkins/workspace/docker-deployment/htmlreports/HTML_20Report'])
33 }
34 }
35 }
```

The execution log shows:

```
[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
```

Building docker image

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



The screenshot shows a GitHub repository page for 'star-agile-banking-finance'. The 'Dockerfile' tab is selected. A commit message from 'shubhamkushwah123' is visible, stating 'Dockerfile added'. The Dockerfile content is displayed:

```
FROM openjdk:11
ARG JAR_FILE=target/*.jar
COPY ${JARFILE} app.jar
ENTRYPOINT ["java","-jar","/app.jar"]
```

Giving necessary sudo root permissions to Jenkins

```
root@ip-172-31-46-79:/home/ubuntu# sudo usermod -aG docker jenkins
root@ip-172-31-46-79:/home/ubuntu# service jenkins restart
```

i-04cc3e51bf7ddc88c (Master)

Public IPs: 13.201.70.243 Private IPs: 172.31.46.79

Creating the docker image for our FinanceMe application

```
33
34
35 }
36 }
37 stage('Building docker image'){
38   steps{
39     sh 'docker build -t gokul1311/financeme:1.0 . '
40   }
41 }
```

The screenshot shows the Jenkins Pipeline interface for a job named "docker-deployment". On the left, there's a sidebar with various pipeline management options like Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, HTML Report, Rename, and Pipeline Syntax. Below that is a "Build History" section showing two builds: #6 (Mar 04, 02:31) and #5 (Mar 04, 02:28). The main area is titled "Stage View" and displays a grid of stages for each build. The columns are labeled: checkout the code from github, code compile, codetesting, package, HTML Report, and Building docker image. Each cell contains a timeline bar indicating the duration of that stage. A red arrow points to the "Building docker image" stage in the #5 build, which took 24 seconds. The overall Jenkins UI has a dark theme.

	checkout the code from github	code compile	codetesting	package	HTML Report	Building docker image
#6 Mar 04 02:31	1s	4s	11s	14s	113ms	7s
#5 Mar 04 02:28	818ms	3s	10s	14s	115ms	3s
#4	1s	3s	10s	13s	134ms	24s

Dockerhub login and image pushing

Setting dockerhub username password for Jenkins

The screenshot shows the Jenkins Pipeline Syntax configuration page. On the left, there's a sidebar with links to Global Variables Reference, Online Documentation, Examples Reference, and IntelliJ IDEA GDSL. The main area shows a 'Sample Step' section with a dropdown menu set to 'withCredentials: Bind credentials to variables'. Below it, the 'withCredentials' step is expanded, showing a 'Bindings' section. A 'Secret text' binding is defined with a variable named 'dockerpass' and a credential named 'Jenkins'. A red box highlights the 'Add' button and the 'Jenkins' credential entry.

Giving dockerhub pass as secret text

The screenshot shows the 'Jenkins Credentials Provider: Jenkins' dialog. It has fields for 'Secret text', 'Scope', 'Secret', 'ID', and 'Description'. The 'Secret' field contains '*****' and is highlighted with a red box. The 'ID' field is set to 'dockerpass' and is also highlighted with a red box. At the bottom right are 'Cancel' and 'Add' buttons.

Dockerhub login and image pushing

```
Script ?  
20  
21  
22  
23  
24 v  
25 v  
26  
27  
28  
29  
30 v  
31  
32  
33  
34  
35 v  
36 v  
37  
38  
39  
40  
41 v  
42 v  
43  
44  
45  
46  
47  
48  
49 }  
  
    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'
    }
}
}
```

Jenkins

Dashboard > docker-deployment >

Status: **green** docker-deployment

Changes: Build Now

Configure: Add description, Disable Project

Delete Pipeline

Full Stage View

HTML Report

Rename

Pipeline Syntax

Build History: trend, Filter..., #7, Mar 3, 2024, 9:31 PM

Average stage times: (Average full run time: ~44s)

checkout the code from github	code compile	codetesting	package	HTML Report	Building docker image	Dockerhub login and image pushing
1s	4s	11s	13s	110ms	6s	20s
783ms	3s	10s	13s	99ms	3s	20s
818ms	3s	10s	14s	115ms	3s	

Stage View: #7, Mar 04, 03:01, No Changes, 783ms, 3s, 10s, 13s, 99ms, 3s, 20s

Stage View: #6, Mar 04, 02:31, No Changes, 818ms, 3s, 10s, 14s, 115ms, 3s,

Stage View: #5, [redacted]

```
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
```

Verifying the Dockerhub

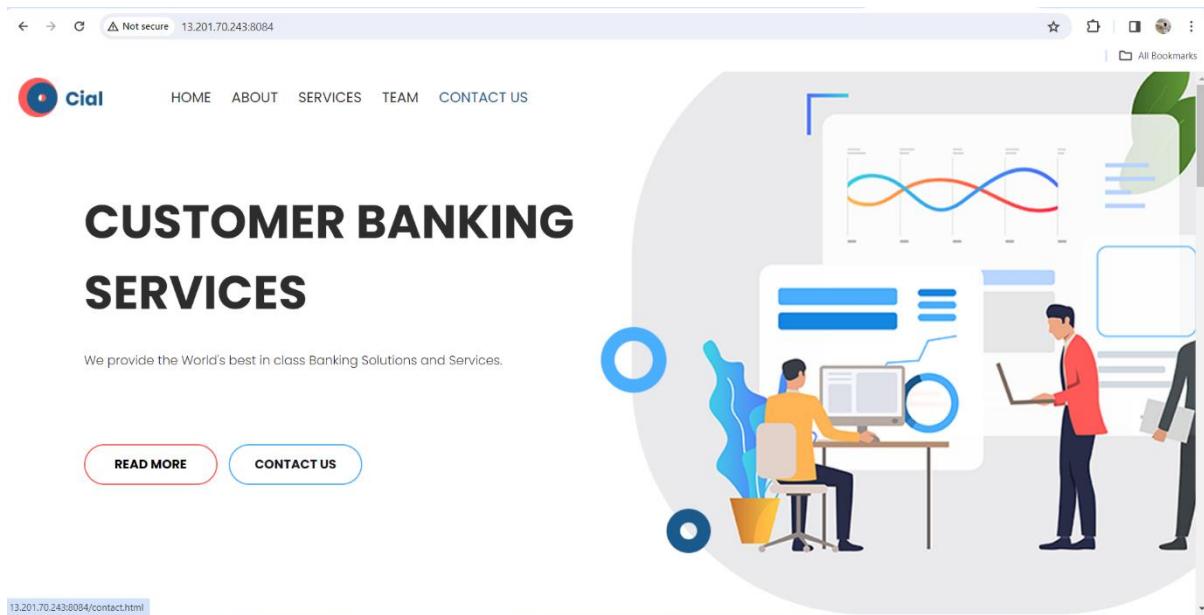
The screenshot shows the Docker Hub interface. At the top, there are tabs for 'Explore', 'Repositories' (which is underlined), and 'Organizations'. A search bar at the top right contains the text 'Search Docker Hub'. Below the search bar, there's a dropdown menu set to 'gokul1311', a search input field with 'Search by repository name', and a dropdown for 'All Content'. A blue button on the right says 'Create repository'. The main content area displays a repository card for 'gokul1311 / financeme'. The card includes the repository name, a note that it 'Contains: Image | Last pushed: 8 minutes ago', a 'Security unknown' badge, a star count of '0', a download count of '0', and a 'Public' status.

Docker containerizing the application Container creation and port expose

The screenshot shows a Jenkins Pipeline script editor. The script is written in Groovy and defines several stages: 'HTML Report', 'Building docker image', 'Dockerhub login and image pushing', and 'containerizing the application'. The 'containerizing the application' stage is highlighted with a red box. Below the script, there are buttons for 'Use Groovy Sandbox', 'Save', and 'Apply'. The output console at the bottom shows the execution of the pipeline, with the 'containerizing the application' stage also highlighted with a red box. The output text includes log entries like 'Layer already exists', 'Pushed', and 'digest: sha256...', followed by '[Pipeline]' logs for each stage.

```
Script ?  
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'])}  
32 }  
33 }  
34 stage('Building docker image'){  
35     steps{  
36         sh 'docker build -t gokul1311/financeme:1.0 .'  
37     }  
38 }  
39 }  
40 }  
41 stage('Dockerhub login and image pushing'){  
42     steps{withCredentials([string(credentialsId: 'dockerpass', variable: 'dockerpass')]) {  
43         sh 'docker login -u gokul1311 -p ${dockerpass}'  
44         sh 'docker push gokul1311/financeme:1.0'  
45     }  
46 }  
47 }  
48 stage('containerizing the application'){  
49     steps{  
50         sh 'docker run -itd -p 8084:8081 gokul1311/financeme:1.0'  
51     }  
52 }  
53 }  
54 }  
55 }  
56 }  
  
Use Groovy Sandbox ?  
Save Apply  
  
0512/cdb4/9b: Layer already exists  
9c742cd6c7a5: Layer already exists  
16ae6fcc7fee: Pushed  
1.0: digest: sha256:0c8cb90f146452459d09a93e9dc6750c2f613ccf14b82386cadafe9b9a2cd  
aa 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
```

Verifying the application

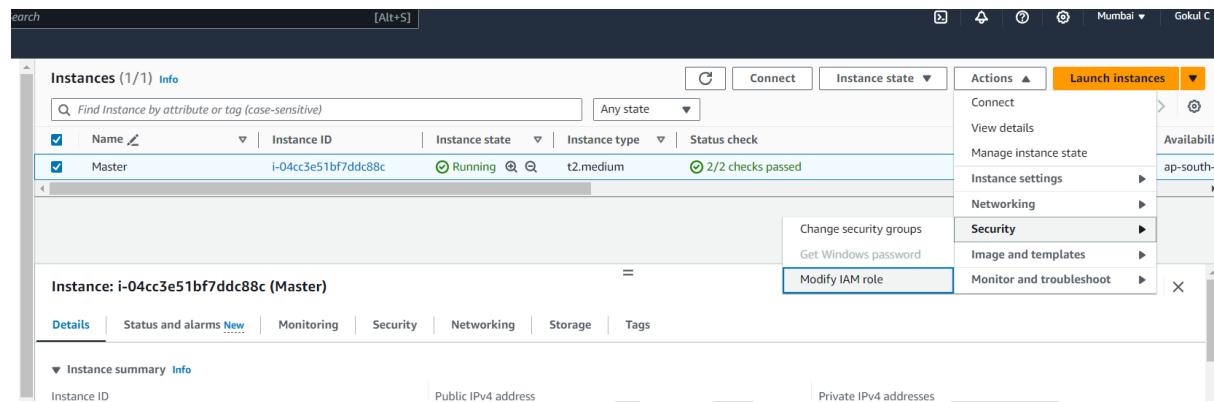


Gokul

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

Enter an item name

test-server-deployment
» Required field

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

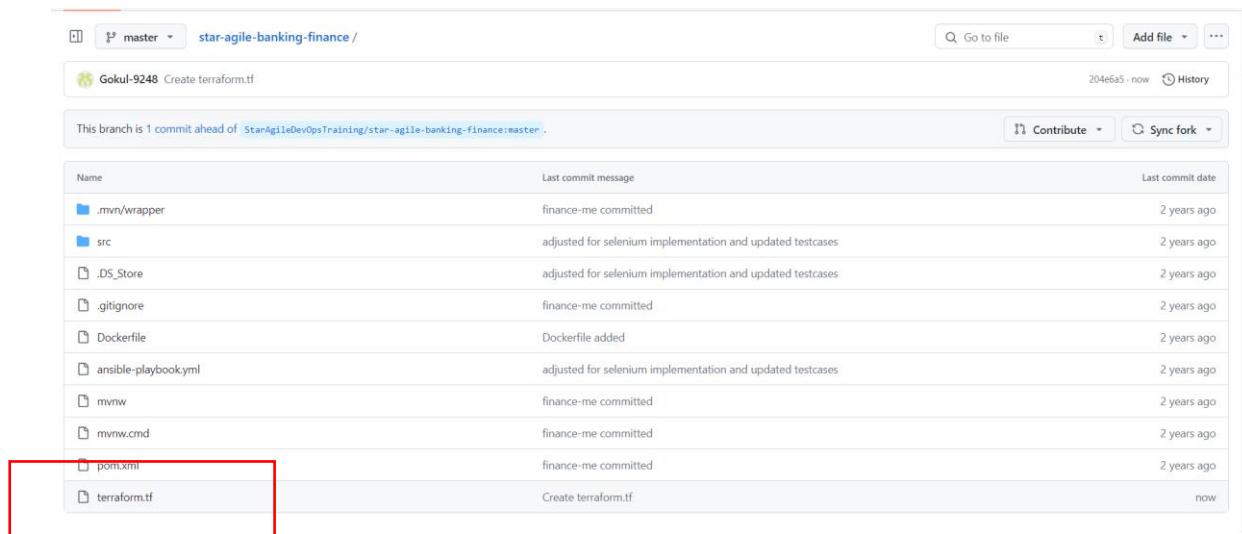
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
A container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

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

The screenshot shows two GitHub repository pages for "star-agile-banking-finance". The top page displays the repository's main information, including its public status, contributors, and recent commits. The bottom page shows the code editor for a new file named `terraform.tf`. The code is a Terraform configuration for creating an AWS VPC and attaching an elastic IP. A red box highlights the final part of the code where the EC2 instance is configured with a tag named "test server".

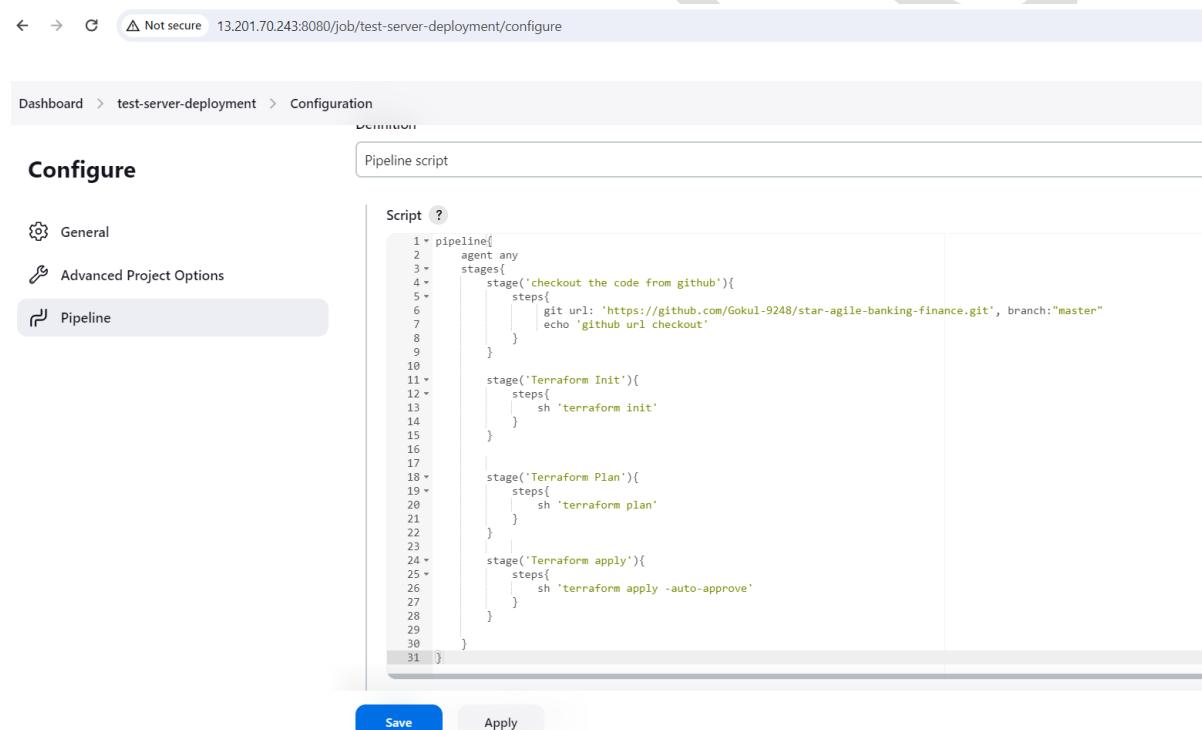
```
1 #Initializing Terraform
2 terraform {
3     required_providers {
4         aws = {
5             source  = "hashicorp/aws"
6             version = "~> 4.0"
7         }
8     }
9 }
10
11 # Configure the AWS provider
12 provider "aws" {
13     region = "ap-south-1"
14 }
15
16 #creating_a_vpc
17
18 resource "aws_vpc" "gokul_vpc" {
19     cidr_block = "10.0.0.0/16"
20 }
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37 #attaching_a_elasticIP
38
39
40 resource "aws_elip" "gokul_elip" {
41     network_interface = aws_network_interface.gokul_network_interface.id
42     instance          = aws_instance.gokul_ec2_instance.id
43     associate_with_private_ip = "10.0.1.10"
44 }
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
```



This screenshot shows a GitHub repository named 'star-agile-banking-finance'. The 'master' branch is selected. A red box highlights the commit 'Create terraform.tf' at the bottom of the list. The commit message is 'Create terraform.tf', the author is 'Gokul-9248', and it was committed 'now'.

Name	Last commit message	Last commit date
.mvn/wrapper	finance-me committed	2 years ago
src	adjusted for selenium implementation and updated testcases	2 years ago
.DS_Store	adjusted for selenium implementation and updated testcases	2 years ago
.gitignore	finance-me committed	2 years ago
Dockerfile	Dockerfile added	2 years ago
ansible-playbook.yml	adjusted for selenium implementation and updated testcases	2 years ago
mvnw	finance-me committed	2 years ago
mvnw.cmd	finance-me committed	2 years ago
pom.xml	finance-me committed	2 years ago
terraform.tf	Create terraform.tf	now

Pipeline for the terraform execution



This screenshot shows the Jenkins Pipeline configuration for a job named 'test-server-deployment'. The 'Pipeline' tab is selected. The pipeline script is defined as follows:

```

1 pipeline{
2   agent any
3   stages{
4     stage('checkout the code from github'){
5       steps{
6         git url: 'https://github.com/Gokul-9248/star-agile-banking-finance.git', branch:"master"
7         echo 'github url checkout'
8       }
9     }
10    stage('Terraform Init'){
11      steps{
12        sh 'terraform init'
13      }
14    }
15    stage('Terraform Plan'){
16      steps{
17        sh 'terraform plan'
18      }
19    }
20    stage('Terraform apply'){
21      steps{
22        sh 'terraform apply -auto-approve'
23      }
24    }
25  }
26}
27
28
29
30
31

```

At the bottom, there are 'Save' and 'Apply' buttons.

Jenkins

Dashboard > test-server-deployment >

Status Changes Build Now Configure Delete Pipeline Full Stage View Rename Pipeline Syntax

test-server-deployment

Stage View

Average stage times: (Average full run time: ~55s)

checkout the code from github	Terraform Init	Terraform Plan	Terraform apply
1s	6s	4s	41s

#1 Mar 04 04:24 No Changes

Build History trend Filter... #1 Mar 3, 2024, 10:54 PM

Permalinks Last build (#1), 7 min 59 sec ago

```

> test-server-deployment > #1

@ [1mPlan: 0m 9 to add, 0 to change, 0 to destroy.
@ [0m@[0m@[1aws_vpc.gokul_vpc: Creating...
@ [0m@[1aws_vpc.gokul_vpc: Creation complete after 1s [id=vpc-07380309279f157a8]@[0m
@ [0m@[1aws_subnet.gokul_subnet: Creating...
@ [0m@[1aws_internet_gateway.gokul_igw: Creating...
@ [0m@[1aws_security_group.gokul_security_group: Creating...
@ [0m@[1aws_internet_gateway.gokul_igw: Creation complete after 0s [id=igw-03498c795b90281e5]@[0m
@ [0m@[1aws_route_table.gokul_route_table: Creating...
@ [0m@[1aws_subnet.gokul_subnet: Creation complete after 0s [id=subnet-0de6d96f5489c08db]@[0m
@ [0m@[1aws_route_table.gokul_route_table: Creation complete after 1s [id=rtb-081f7bd96f1f2ef8a]@[0m
@ [0m@[1aws_route_table_association.gokul_subnet_association: Creating...
@ [0m@[1aws_route_table_association.gokul_subnet_association: Creation complete after 0s [id=rtbassoc-0f35697a492a99302]@[0m
@ [0m@[1aws_security_group.gokul_security_group: Creation complete after 2s [id=sg-001c935fce5d23531]@[0m
@ [0m@[1aws_network_interface.gokul_network_interface: Creating...
@ [0m@[1aws_network_interface.gokul_network_interface: Creation complete after 0s [id=en-02f6fd51710b5af94]@[0m
@ [0m@[1aws_instance.gokul_ec2_instance: Creating...
@ [0m@[1aws_instance.gokul_ec2_instance: Still creating... [10s elapsed]@[0m@[0m
@ [0m@[1aws_instance.gokul_ec2_instance: Still creating... [20s elapsed]@[0m@[0m
@ [0m@[1aws_instance.gokul_ec2_instance: Still creating... [30s elapsed]@[0m@[0m
@ [0m@[1aws_instance.gokul_ec2_instance: Creation complete after 32s [id=i-0a707ddbaa4597878]@[0m
@ [0m@[1aws_eip.gokul_eip: Creating...
@ [0m@[1aws_eip.gokul_eip: Creation complete after 1s [id=eipalloc-08872275ac5b18fa3]@[0m
@ [0m@[1m@[32m
Apply complete! Resources: 9 added, 0 changed, 0 destroyed.
@ [0m
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
```

Verifying the terraform instance creation

The screenshot shows the AWS CloudWatch Metrics console. A metric named "test" is selected, showing a single data point with a value of 1. The x-axis represents time from 2024-01-01T00:00:00Z to 2024-01-01T01:00:00Z. The y-axis represents the metric value.

Time	Value
2024-01-01T00:00:00Z	1

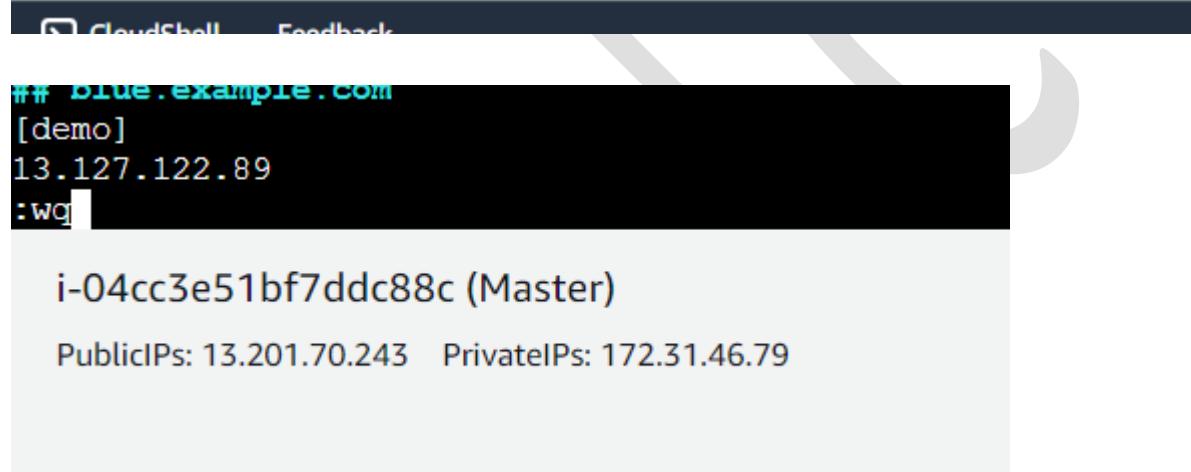
Ansible deployment on test server

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

```
root@ip-172-31-46-79:/etc/ansible# ls  
ansible.cfg  hosts  roles  
root@ip-172-31-46-79:/etc/ansible# vi hosts
```

i-04cc3e51bf7ddc88c (Master)

PublicIPs: 13.201.70.243 PrivateIPs: 172.31.46.79



Setting the Jenkins tools for ANSIBLE
Manage Jenkins > tools >

The screenshot shows the Jenkins Manage Jenkins page. The left sidebar has links: "+ New Item", "People", "Build History", "Project Relationship", "Check File Fingerprint", "Manage Jenkins" (which is selected and highlighted in grey), and "My Views". The main content area has a heading "Manage Jenkins" and a warning message: "Building on the built-in node can be a security issue. You should set up distributed builds. See [the documentation](#)". Below this is a box titled "Java 11 end of life in Jenkins" with the message: "You are running Jenkins on Java 11, support for which will end on or after Sep 30, 2024. Refer to [the documentation](#) for more details". The right side of the page has sections for "System Configuration" with "System" and "Tools" options, and a "Build Queue" section which is currently empty.

Dashboard > Manage Jenkins > Tools

Add Ansible

Ansible

Name: ansible

Install automatically ? Add Installer ▾

Add Ansible

Docker installations

Add Docker

Save **Apply**

Verifying ansible playbook in our repository

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

Edit Preview Code 55% faster with GitHub Copilot Cancel changes Commit changes...

```

1 - name : Configure Docker on EC2 Instances
2 hosts : all
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: Deploy Docker Container
16     command: docker run -itd -p 8084:8081 gokul1311/financeme:1.0
17

```

Verify the port expose and docker image

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

The screenshot shows two Jenkins Pipeline configuration pages. The top page is for creating an SSH credential named 'ansible'. The bottom page shows how this credential is used in an 'ansiblePlaybook' step.

Top Page: Creating an SSH Credential

- Kind:** SSH Username with private key (highlighted with a red box)
- Scope:** Global (Jenkins, nodes, items, all child items, etc)
- ID:** ansible
- Description:** ansible
- Username:** ubuntu (highlighted with a red box)
- Treat username as secret:**
- Private Key:** Enter directly (radio button selected)
- Key:** (Text area containing RSA PRIVATE KEY content)
- Passphrase:** (Text area for passphrase)
- Notes:** Username of the machine should be given here

Bottom Page: Using the SSH Credential in an Ansible Step

- Sample Step:** ansiblePlaybook: Invoke an ansible playbook
- ansiblePlaybook:** ansible
- Ansible tool:** ansible
- Playbook file path in workspace:** ansible-playbook.yml (highlighted with a red box)
- Inventory file path in workspace:** /etc/ansible/hosts (highlighted with a red box)
- SSH connection credentials:** - none - (highlighted with a red box)
- Notes:** Name of the Ansible playbook file present in our Github, Path of the inventory host file, which present in our master, The SSH Credentials that we created previously

<input checked="" type="checkbox"/> Disable the host SSH key check
<input type="checkbox"/> Colorized output
Extra parameters
<input type="text"/>
Generate Pipeline Script
ansiblePlaybook credentialsId: 'test-server', disableHostKeyChecking: true, installation: 'ansible', inventory: '/etc/ansible/hosts', playbook: 'ansible-playbook.yml', vaultTmpPath: ''

Dashboard > test-server-deployment > Configuration

Configure

General

Advanced Project Options

Pipeline

```
o 9
10 }
11 = stage('Terraform Init'){
12 =   steps{
13 =     sh 'terraform init'
14 =   }
15 }
16
17
18 = stage('Terraform Plan'){
19 =   steps{
20 =     sh 'terraform plan'
21 =   }
22 }
23
24 = stage('Terraform apply'){
25 =   steps{
26 =     sh 'terraform apply -auto-approve'
27 =   }
28 }
29
30 = stage('ansible deployment'){
31 =   steps{
32 =     ansiblePlaybook credentialsId: 'test-server', disableHostKeyChecking: true, installation: 'ansible', inventory: '/etc'
33 =   }
34 }
35 }
```

Use Groovy Sandbox [?](#)

[Pipeline Syntax](#)

[Save](#) [Apply](#)

Stage View

checkout the code from github	Terraform Init	Terraform Plan	Terraform apply	ansible deployment
2s	4s	5s	17s	51s

Average stage times:
(Average full run time: ~47s)

#3 Mar 04 05:43 1 commit

#2 Mar 04 05:13 No Changes

Build History trend ▾ Filter... #3 Mar 4, 2024, 12:13 AM

```
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/ssh1682253950942014045.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

...[0m [Dockerizing : success]
ok: [13.127.122.89]

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

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

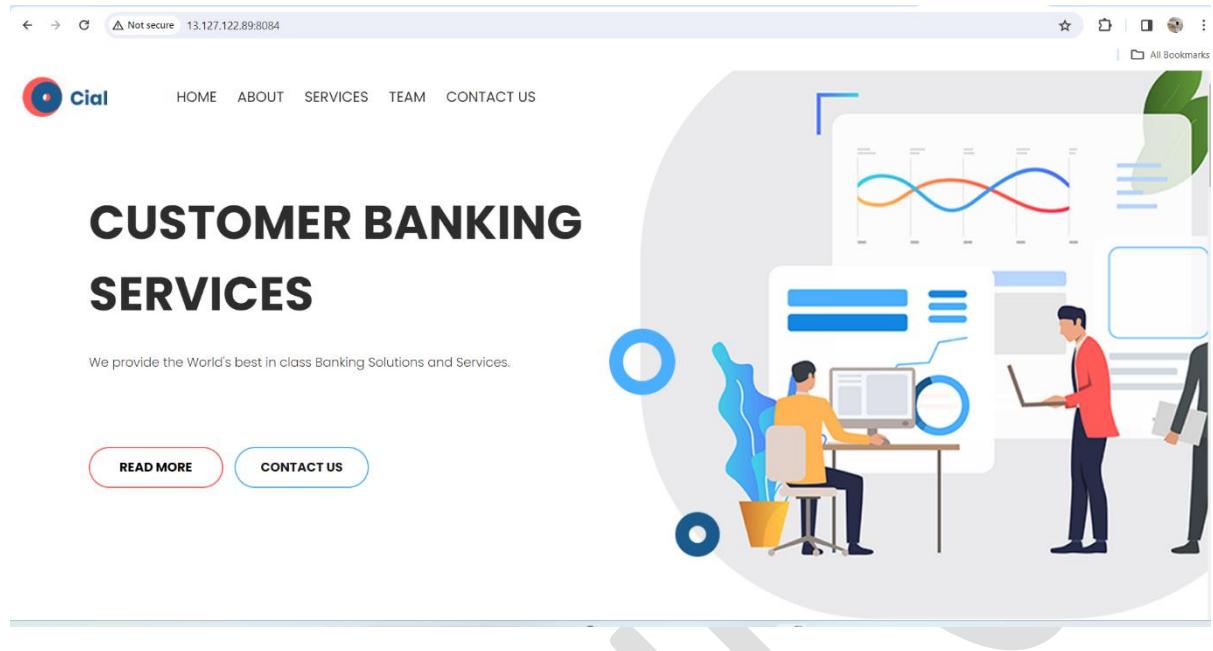
TASK [Start Docker Service] *****
changed: [13.127.122.89]

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
```

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



Gokul

Prod server deployment

Setting the prod server instance

The screenshot shows the 'Launch an instance' wizard in the AWS EC2 console. The 'Name and tags' section has 'prod server' entered in the Name field. The 'Software Image (AMI)' section shows 'Canonical, Ubuntu, 22.04 LTS, ami-03bb6d83c60fcf7c'. The 'Virtual server type (instance type)' is set to 't2.micro'. A summary panel on the right indicates 1 instance will be launched. A tooltip for the free tier is visible.

The screenshot shows the 'Instances' page in the AWS EC2 console. The 'prod server' instance is listed in the table, showing it is running and assigned the public IP 13.233.155.105. The instance ID is i-0bce360e70e158d3b.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Available
Master	i-04cc5e51bf7ddc8bc	Running	t2.medium	2/2 checks passed	View alarms +	ap-south-1
test server	i-0a707ddbaa4597878	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1
prod server	i-0bce360e70e158d3b	Running	t2.micro	Initializing	View alarms +	ap-south-1

Ansible deployment to Prod server

Creating a new pipeline job for prod server deployment

Enter an item name

» Required field

 **Freestyle project**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

 **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

 **Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

 **Folder**
A folder is a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
OK

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

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

```
## [openSUSE]
## green.example.com
## blue.example.com
[demo]
13.127.122.89

[prod]
13.233.155.105
:wg
```

i-04cc3e51bf7ddc88c (Master)

Public IPs: 13.201.70.243 Private IPs: 172.31.46.79

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

The screenshot shows the GitHub interface for a repository named 'star-agile-banking-finance'. A red box highlights the file 'ansible-playbook1.yml' in the master branch. Below it, a modal window titled 'Commit changes' is open, containing a commit message 'Create ansible-playbook1.yml'. A large red box highlights this commit message. The commit message field is filled with 'Create ansible-playbook1.yml'. The modal also includes fields for 'Extended description' (empty) and two radio button options: 'Commit directly to the master branch' (selected) and 'Create a new branch for this commit and start a pull request'. At the bottom are 'Cancel' and 'Commit changes' buttons. In the background, the repository's commit history is visible, showing several commits, including one from 'Gokul-9248' with the message 'Create ansible-playbook1.yml'.

Name	Last commit message	Last commit date
.mvn/wrapper	finance-me committed	2 years ago
src	adjusted for selenium implementation and updated testcases	2 years ago
.DS_Store	adjusted for selenium implementation and updated testcases	2 years ago
.gitignore	finance-me committed	2 years ago
Dockerfile	Dockerfile added	2 years ago
ansible-playbook.yml	Update ansible-playbook.yml	2 minutes ago
ansible-playbook1.yml	Create ansible-playbook1.yml	now
mvnw	finance-me committed	2 years ago
mvnw.cmd	finance-me committed	2 years ago

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

Pipeline Syntax

Sample Step

ansiblePlaybook: Invoke an ansible playbook

ansiblePlaybook ?

Ansible tool

ansible

Playbook file path in workspace

ansible-playbook1.yml

Inventory file path in workspace

/etc/ansible/hosts

SSH connection credentials

- none -

+ Add ▾

Jenkins



Jenkins Credentials Provider: Jenkins

ubuntu

Treat username as secret ?

Private Key

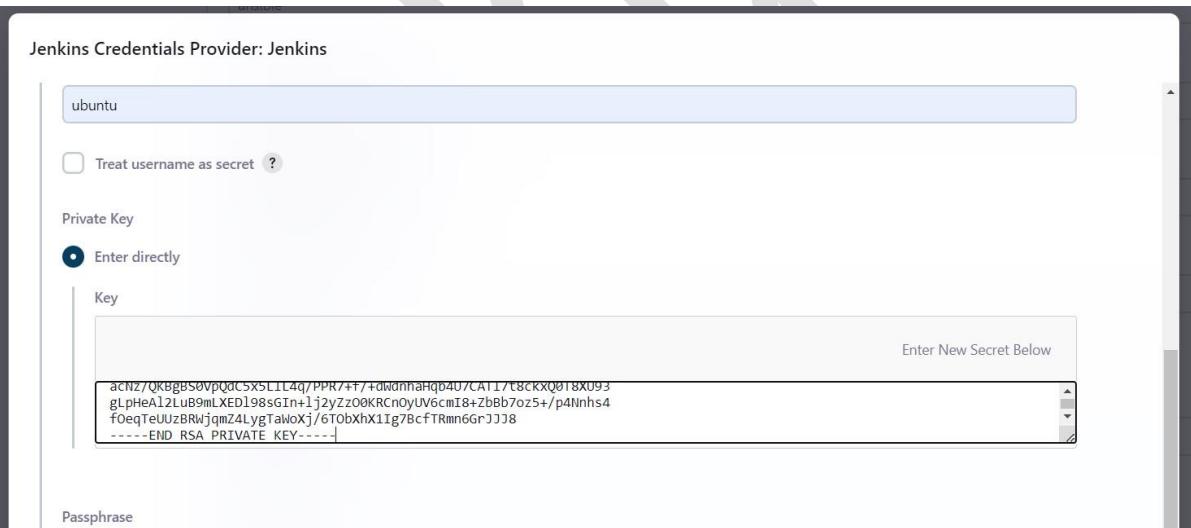
Enter directly

Key

acNz/QRBgBS0VpQdC5x5L1L4qj/PPR/4f/+dw0mnaHqb4U/CAT17t8CKXQ018X093
gLpHeA12LuB9mLXE198sGIn+1j2yZz00kRCnOyUV6cm18+zbBb7oz5+/p4Nhhs4
f0eqTeUzBRwjqmZ4LygTaWoXj/6TObxhX1Ig7BcfTRmm6GrJJJ8
-----END RSA PRIVATE KEY-----

Enter New Secret Below

Passphrase



Disable the host SSH key check
 Colorized output
 Extra parameters

Generate Pipeline Script

```
ansiblePlaybook credentialsId: 'prod server' disableHostKeyChecking: true, installation: 'ansible', inventory: '/etc/ansible/hosts', playbook: 'ansible-playbook1.yml', vaultTmpPath: ''
```

Dashboard > prod-server-deployment > Configuration

Configure

Pipeline

General

Advanced Project Options

Pipeline

Definition

Pipeline script

```
1 pipeline{
2     agent any
3     stages{
4         stage('checkout the code from github'){
5             steps{
6                 git url: 'https://github.com/Gokul-9248/star-agile-banking-finance.git', branch:"master"
7                 echo 'github url checkout'
8             }
9         }
10     }
11     stage('ansible deployment'){
12         steps{
13             ansiblePlaybook credentialsId: 'prod server', disableHostKeyChecking: true, installation: 'ansible', inventory: '/etc/ansible/hosts', playbook: 'ansible-playbook1.yml', vaultTmpPath: ''
14         }
15     }
16 }
17 }
```



Search (CTRL+K)

Dashboard > prod-server-deployment >

Status

prod-server-deployment

</> Changes

▷ Build Now

⚙ Configure

Delete Pipeline

Full Stage View

Rename

Pipeline Syntax

Stage View

Average stage times:
(Average full run time: ~1min 6s)



Verifying the prod server deployment

```
[Pipeline] Ansible Playbook
[prod-server-deployment] $ ansible-playbook ansible-playbook1.yml -i /etc/ansible/hosts --private-key /var/lib/jenkins/workspace/prod-server-deployment/ssh15801966688429747903.key -u ubuntu

PLAY [Configure Docker on EC2 Instances] ****
TASK [Gathering Facts] ****
ok: [13.233.155.105]

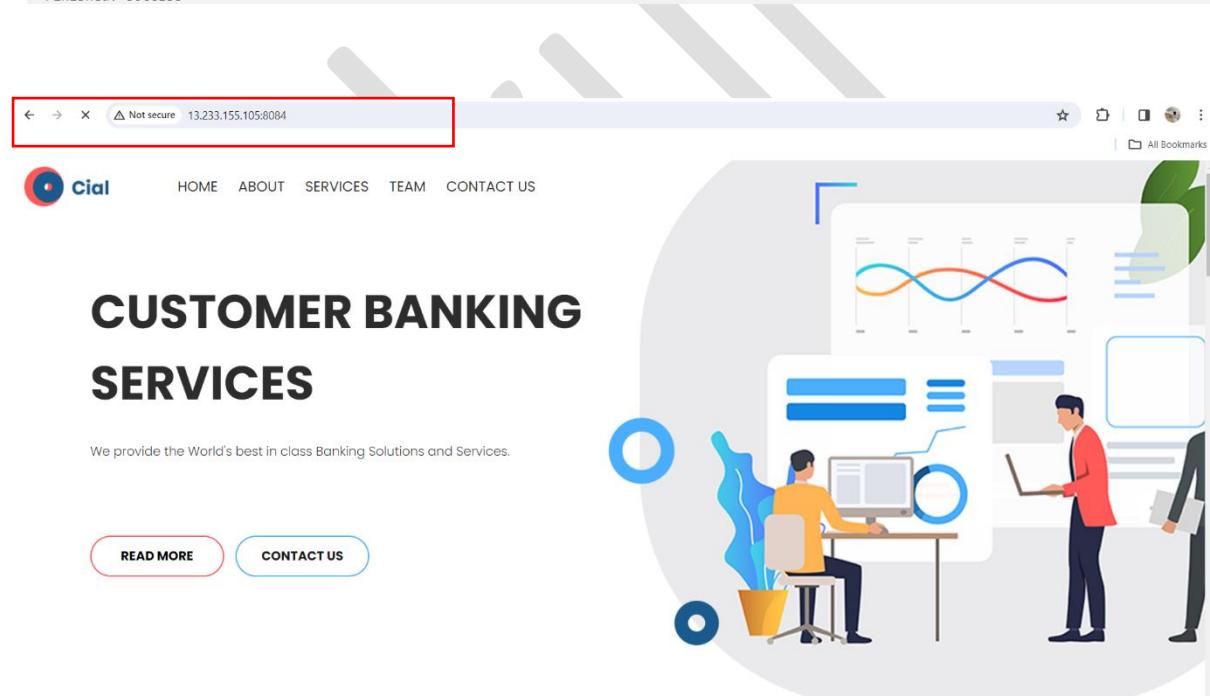
TASK [updating apt] ****
changed: [13.233.155.105]

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
```



The screenshot shows a web browser displaying a landing page for a company named 'Cial'. The URL in the address bar is 'Not secure 13.233.155.105:8084'. The page has a header with a logo and navigation links for HOME, ABOUT, SERVICES, TEAM, and CONTACT US. Below the header, there is a large section titled 'CUSTOMER BANKING SERVICES' with the subtext 'We provide the World's best in class Banking Solutions and Services.' At the bottom of this section are two buttons: 'READ MORE' and 'CONTACT US'. To the right of this text is a large, stylized circular graphic featuring three people (two men and one woman) working on computers, surrounded by abstract shapes like circles and lines.

Automation of workflows

Creating a WEBHOOK to our repo for the automatic triggering actions

The screenshot shows the GitHub repository settings for 'star-agile-banking-finance'. The 'Webhooks' tab is selected. A red box highlights the 'Webhooks' section in the sidebar. Another red box highlights the 'Payload URL *' input field, which contains the value 'http://13.201.70.243:8080/github-webhook/'. A callout bubble points to this field with the text 'Our webhook is ready, we have to copy this link to jenkins 1st job, add it in Build trigger'. The 'Content type' dropdown is set to 'application/x-www-form-urlencoded'. The 'Secret' field is empty. Under 'Which events would you like to trigger this webhook?', the 'Just the push event.' option is selected. The 'Active' checkbox is checked, with the note 'We will deliver event details when this hook is triggered.' Below the form is a green 'Add webhook' button.

Webhooks

Add webhook

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

✓ http://13.201.70.243:8080/github-w... (push)

Edit Delete

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    }
49
50    stage('stop existing containers'){
51        steps{
52            sh 'docker stop $(docker ps -a -q)'
53        }
54
55    stage('deleting existing containers'){
56        steps{
57            sh 'docker rm $(docker ps -a -q)'
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.

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

Edit Preview Code 55% faster with GitHub Copilot

```
1 - name : Configure Docker on EC2 Instances
2   hosts : prod
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    - name: Deploy Docker Container
22      command: docker run -itd -p 8084:8081 gokul1311/financeme:1.0
23
```

Build triggers our Jenkins job

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

Dashboard > test-server-deployment > Configuration

Throttle builds ?

Configure

Build Triggers

General

Advanced Project Options

Pipeline

Build after other projects are built ?

Projects to watch

docker-deployment,

Trigger only if build is stable

Trigger even if the build is unstable

Trigger even if the build fails

Configure General Advanced Project Options Pipeline Build after other projects are built [?](#)

Projects to watch

test-server-deployment,

 Trigger only if build is stable Trigger even if the build is unstable Trigger even if the build fails**Verifying the workflow by pushing a commit to the github****Our 1st job is triggered automatically by the github push****docker-deployment**[Add description](#)[Disable Project](#)**Stage View**

	checkout the code from github	code compile	codetesting Success	package	HTML Report	Building docker image	Dockerhub login and image pushing	stop existing containers	deleting existing containers	containerizing the application
Average stage times: (Average full run time: ~53s)	988ms	3s	Success	13s	84ms	3s	18s	842ms	339ms	578ms
#10 Mar 04 13:39	988ms	3s	11s	13s	84ms	3s	18s	842ms	339ms	578ms

Permalinks

docker-deployment > #11

Console Output

»

 Output as plain text

Started by GitHub push by Gokul-9248
 [Pipeline] Start of Pipeline
 [Pipeline] node

Running on Jenkins in /var/lib/jenkins/workspace/docker-deployment
 [Pipeline] {

```
+ docker run -itd -p 8084:8081 gokul1311/financeme:1.0
2841d05c8279956dedf05d5ac6e2a794bfbcc4464d8c72b06a51c76ad8fb8dc6
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Triggering a new build of test-server-deployment #5
Finished: SUCCESS
```

> test-server-deployment > #5

Console Output

ges

ole Output

ew as plain text

View in browser

Changes [13.127.122.89]

```
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
```

Dashboard > prod-server-deployment > #3

Status

Changes

Console Output

View as plain text

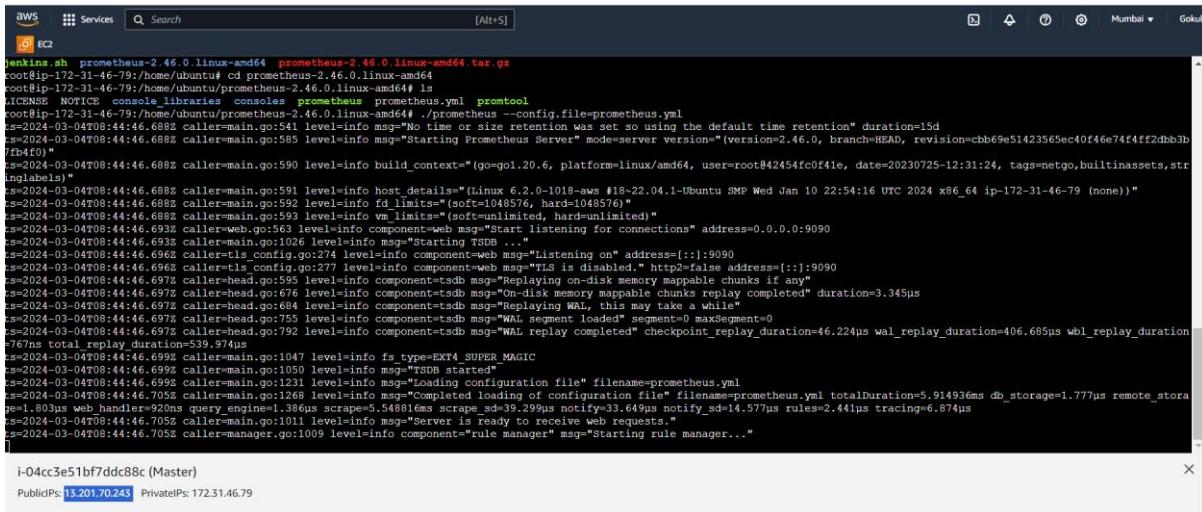
Edit Build Information

Console Output

```
Started by upstream project "test-server-deployment" build number 5
originally caused by:
Started by upstream project "docker-deployment" build number 11
originally caused by:
Started by GitHub push by Gokul-9248
[Pipeline] Start of Pipeline
[Pipeline] node
```

Continuous monitoring using PROMETHEUS & GRAFANA

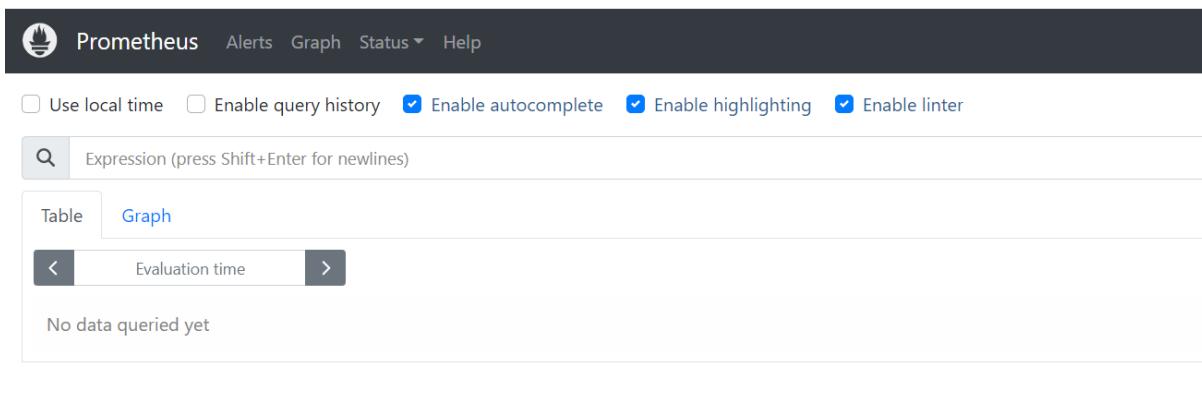
PROMETHEUS INSTALLATION



```
aws Services Search [Alt+S] Jenkins.sh prometheus-2.46.0.linux-amd64.tar.gz
cd /tmp/prometheus-2.46.0.linux-amd64
tar -xvf prometheus-2.46.0.linux-amd64.tgz
cd prometheus-2.46.0.linux-amd64
ls
LICENSE NOTICE console libraries consoles prometheus prometheus.yml promtool
root@ip-172-31-46-79:/home/ubuntu/prometheus-2.46.0.linux-amd64# ./prometheus --config.file=prometheus.yaml
ts=2024-03-04T08:44:46.688Z caller=main.go:541 level=info msg="No time or size retention was set so using the default time retention" duration=15d
ts=2024-03-04T08:44:46.688Z caller=main.go:585 level=info msg="Starting Prometheus Server" mode=server version="(version=2.46.0, branch=HEAD, revision=ccb69e51423565ec40f46e74f4ff2dbb3b7fb4f0)"
ts=2024-03-04T08:44:46.688Z caller=main.go:591 level=info build_context="(go=g01.20.6, platform=linux/amd64, user=root@842454fc0f1e, date=20230725-12:31:24, tags=netg0,builtinassets,stringslabels)"
ts=2024-03-04T08:44:46.688Z caller=main.go:591 level=info host_details="(Linux 6.2.0-1018-aws #18~22.04.1-Ubuntu SMP Wed Jan 10 22:54:16 UTC 2024 x86_64 ip-172-31-46-79 (none))"
ts=2024-03-04T08:44:46.688Z caller=main.go:592 level=info fd limits="(soft=1048576, hard=1048576)"
ts=2024-03-04T08:44:46.688Z caller=main.go:593 level=info vm limits="(soft=unlimited, hard=unlimited)"
ts=2024-03-04T08:44:46.693Z caller=main.go:563 level=info component=web msg="Starting listening for connections" address=0.0.0.0:9090
ts=2024-03-04T08:44:46.693Z caller=main.go:1026 level=info msg="Starting TSDR ... "
ts=2024-03-04T08:44:46.696Z caller=cls config.go:274 level=info component=web msg="Listening on" address=[::]:9090
ts=2024-03-04T08:44:46.696Z caller=cls config.go:277 level=info component=web msg="TLS is disabled." http2=false address=[::]:9090
ts=2024-03-04T08:44:46.697Z caller=head.go:595 level=info component=tssdb msg="Replaying on-disk memory mappable chunks if any"
ts=2024-03-04T08:44:46.697Z caller=head.go:676 level=info component=tssdb msg="on-disk memory mappable chunks replay completed" duration=3.345μs
ts=2024-03-04T08:44:46.697Z caller=head.go:684 level=info component=tssdb msg="Replaying WAL, this may take a while"
ts=2024-03-04T08:44:46.697Z caller=head.go:755 level=info component=tssdb msg="WAL segment loaded" segment=0 maxSegment=0
ts=2024-03-04T08:44:46.697Z caller=head.go:792 level=info component=tssdb msg="WAL replay completed" checkpoint_replay_duration=46.224μs wal_replay_duration=406.605μs wal_replay_start_time=1767ns total_replay_duration=539.974μs
ts=2024-03-04T08:44:46.699Z caller=main.go:1047 level=info fs_type=EXT4_SUPER_MAGIC
ts=2024-03-04T08:44:46.699Z caller=main.go:1050 level=info msg="TSDB started"
ts=2024-03-04T08:44:46.699Z caller=main.go:1123 level=info msg="Loading configuration file" filename=prometheus.yaml
ts=2024-03-04T08:44:46.705Z caller=main.go:1269 level=info msg="Completed loading of configuration file" filename=prometheus.yaml totalDuration=5.914936ms db_storage=1.777μs remote_stora
ne=1.803μs web_handlers=920ms query_engine=1.386μs scrape_sd=39.299μs notify_sd=33.649μs notify_sd=14.577μs rules=2.441μs tracing=6.874μs
ts=2024-03-04T08:44:46.705Z caller=main.go:1011 level=info msg="Server is ready to receive web requests."
ts=2024-03-04T08:44:46.705Z caller=manager.go:1009 level=info component="rule manager" msg="Starting rule manager..."
```

Checking the Prometheus on default port 9090

Not secure 13.201.70.243:9090/graph?g0.expr=&g0.tab=1&g0.stacked=0&g0.show_exemplars=0&g0.range_input=1h



Prometheus Alerts Graph Status Help

Use local time Enable query history Enable autocomplete Enable highlighting Enable linter

Expression (press Shift+Enter for newlines)

Table Graph

< Evaluation time >

No data queried yet

Add Panel

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: sha256:4cb2b9019f1757be8482419002cb7afe028fdb35d47958829e4cfeaf6246d80
Status: Downloaded newer image for prom/node-exporter:latest
39111f61206b136abd1717cdabd3e4893812277f1362782907a4baeebfbd8ce
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: sha256:4cb2b9019f1757be8482419002cb7afe028fdb35d47958829e4cfeaf6246d80
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

← → ⌂ Not secure **13.127.122.89:9100**

Node Exporter

Prometheus Node Exporter

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

- [Metrics](#)

Node Exporter

Prometheus Node Exporter

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

- [Metrics](#)

Configuring the Prometheus targets

```
# metrics_path defaults to '/metrics'
# scheme defaults to 'http'

- job_name: 'test-server'
  static_configs:
    - targets: ['13.127.122.89:9100']

- job_name: 'prod-server'
  static_configs:
    - targets: ['13.233.155.105:9100']
```

"prometheus.yml" 35L, 1056B

i-04cc3e51bf7ddc88c (Master)

Public IPs: 13.201.70.243 Private IPs: 172.31.46.79

Prometheus Alerts Graph Status Help

Targets

All scrape pools ▾ All Unhealthy Collapse All Filter by endpoint or labels Unknown Unhealthy Healthy

prod-server (1/1 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://13.233.155.105:9100/metrics	UP	instance="13.233.155.105:9100" job="prod-server"	4.446s ago	13.179ms	

test-server (1/1 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://13.127.122.89:9100/metrics	UP	instance="13.127.122.89:9100" job="test-server"	12.796s ago	12.607ms	

GRAFANA INSTALLATION

```
root@ip-172-31-46-79:/home/ubuntu# systemctl start grafana-server
root@ip-172-31-46-79:/home/ubuntu# systemctl status grafana-server
● grafana-server.service - Grafana instance
   Loaded: loaded (/lib/systemd/system/grafana-server.service; disabled; vendor preset: enabled)
     Active: active (running) since Mon 2024-03-04 10:32:27 UTC; 12s ago
       Docs: http://docs.grafana.org
      Main PID: 2870 (grafana)
        Tasks: 15 (limit: 4666)
       Memory: 46.6M
          CPU: 2.730s
         CGroup: /system.slice/grafana-server.service
                  └─2870 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/grafana/grafana-server.pid --packaging=deb cfg=default.paths.logs=/var/log/grafana

Mar 04 10:03:32 ip-172-31-46-79 grafana[2870]: logger-local.finder t=2024-03-04T10:03:32.859976056Z level=warn msg="Skipping finding plugins as directory does not exist" path=/var/lib/...
Mar 04 10:03:32 ip-172-31-46-79 grafana[2870]: logger-nogalert.multiorg.alertmanager t=2024-03-04T10:03:32.862795658Z level=info msg="Starting MultiOrg Alertmanager"
Mar 04 10:03:32 ip-172-31-46-79 grafana[2870]: logger-grafana-apiserver t=2024-03-04T10:03:32.865440197Z level=info msg="Authentication is disabled"
Mar 04 10:03:32 ip-172-31-46-79 grafana[2870]: logger-grafana-apiserver t=2024-03-04T10:03:32.868156026Z level=info msg="Adding GroupVersion playlist.grafana.app v0alpha1 to ResourceMap"
Mar 04 10:03:32 ip-172-31-46-79 grafana[2870]: logger-report t=2024-03-04T10:03:32.868230385Z level=warn msg="Scheduling and sending of reports disabled, SMTP is not configured and enab...
Mar 04 10:03:32 ip-172-31-46-79 grafana[2870]: logger-nogalert.state.manager t=2024-03-04T10:03:32.873642537Z level=info msg="State cache has been initialized" states=0 duration=25.63375...
Mar 04 10:03:32 ip-172-31-46-79 grafana[2870]: logger-nogalert.scheduler t=2024-03-04T10:03:32.873704066Z level=info msg="Starting scheduler" tickInterval=10s
Mar 04 10:03:32 ip-172-31-46-79 grafana[2870]: logger-ticker t=2024-03-04T10:03:32.873755715Z level=info msg="Starting first tick=2024-03-04T10:03:40Z
Mar 04 10:03:33 ip-172-31-46-79 grafana[2870]: logger-plugins.update.checker t=2024-03-04T10:03:33.230029531Z level=info msg="Update check succeeded" duration=368.052123ms
Mar 04 10:03:33 ip-172-31-46-79 grafana[2870]: logger-grafana.update.checker t=2024-03-04T10:03:33.243845509Z level=info msg="Update check succeeded" duration=382.683074ms
lines 1-21721 (END)
```

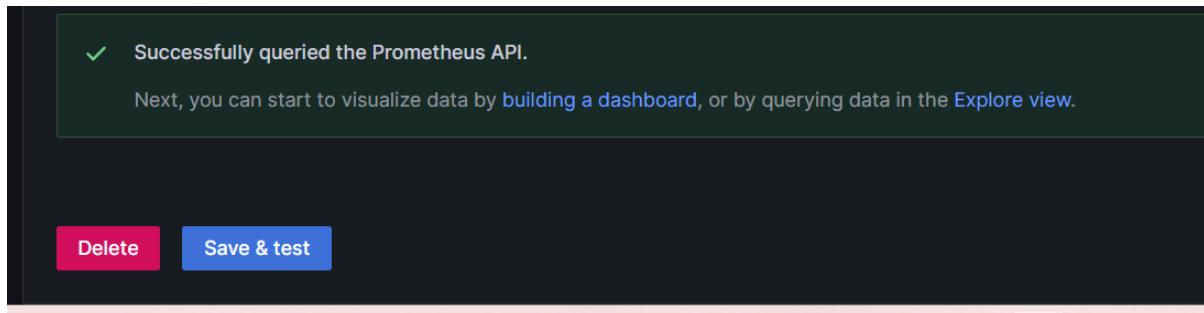
Accessing Grafana on default port 3000

The screenshot shows the Grafana landing page. At the top, there's a navigation bar with links for 'Documentation', 'Tutorials', 'Community', and 'Public Slack'. Below the header, there's a 'Welcome to Grafana' message. On the left, a 'Basic' section provides a 'Tutorial' link to 'Grafana fundamentals'. In the center, there are two main panels: 'DATA SOURCES' (with a 'Add your first data source' button) and 'DASHBOARDS' (with a 'Create your first dashboard' button). At the bottom, there's a 'Latest from the blog' section.

Adding Prometheus as Data source

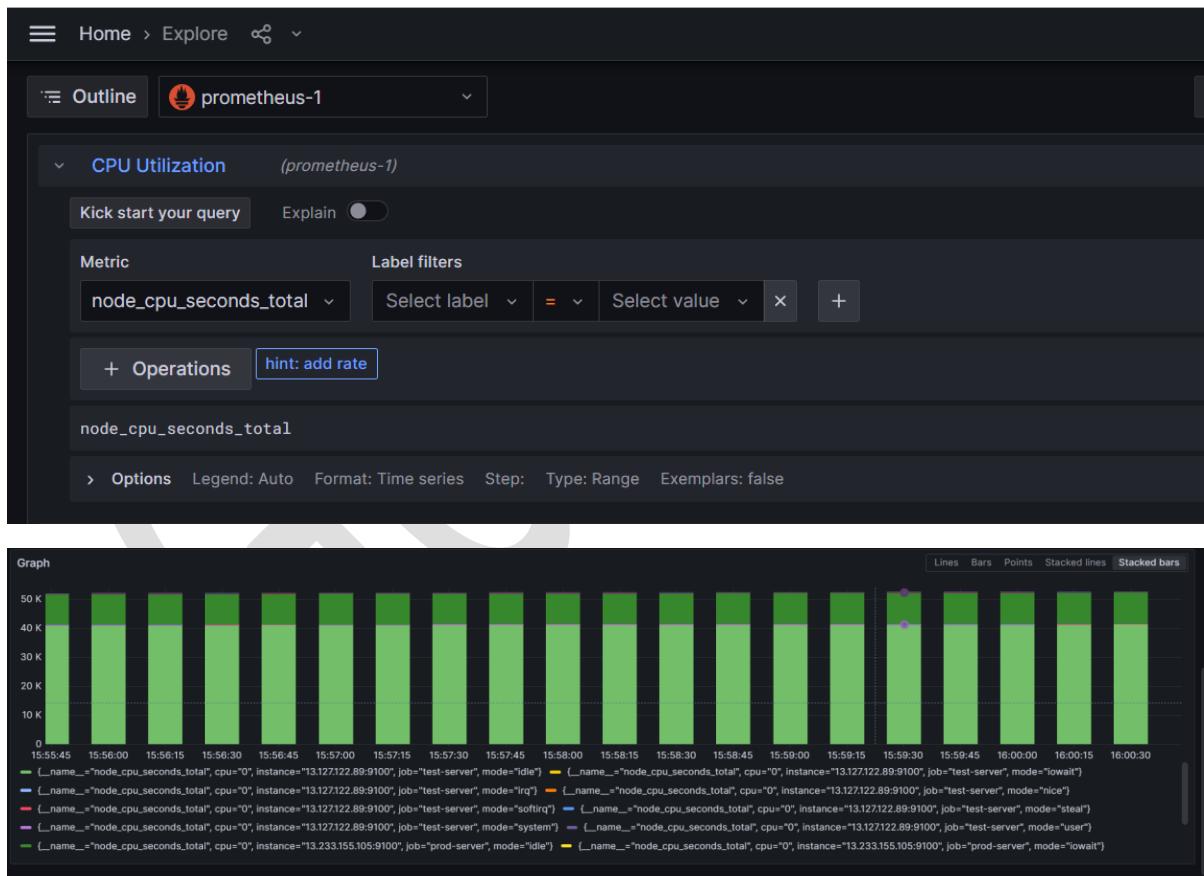
The screenshot shows the 'Add data source' configuration page. It starts with a search bar for 'Filter by name or type'. Below it, there's a list of 'Time series databases' with icons. The 'Prometheus' entry is highlighted, showing its details: 'Open source time series database & alerting' and a 'Core' button. Below this, there's another entry for 'Graphite' with the same description.

We have to give Prometheus url as the data source

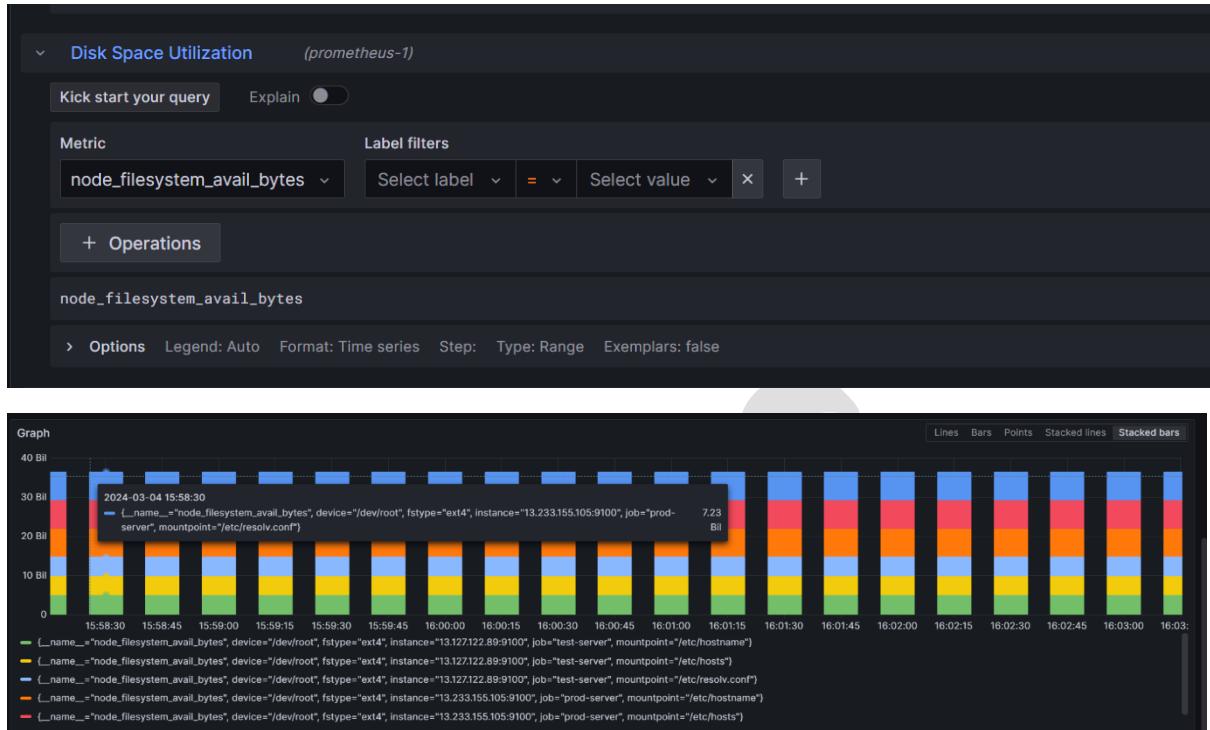


Dashboard creations

1. CPU Utilization



2. Disk space utilization



3. Total available memory

