

Banking and Finance Domain

A PROJECT REPORT

SUBMITTED BY

ANMOL UPADHAYA

DEVOPS Course

SUBMITTED TO

Star Agile

APR & 2025

Certification Project – Finance Me

Banking and Finance Domain

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.

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

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

Step1: Create a Terraform – Server to Deploy a DEV ENV, TEST Server, and Prod Server for Further Process.

The screenshot displays the AWS CloudShell interface. The terminal window shows the following command sequence:

```
root@Terraform-Server:~# wget -O https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
--2025-04-22 05:49:58-- https://apt.releases.hashicorp.com/gpg
Resolving apt.releases.hashicorp.com (apt.releases.hashicorp.com)... 18.160.10.45, 18.160.10.71, 18.160.10.126, ...
Connecting to apt.releases.hashicorp.com (apt.releases.hashicorp.com)|18.160.10.45|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3980 [application/octet-stream]
Saving to: "STDOUT"
[Progress Bar] 100%[=====] 3.89K --.-KB/s   in 0s
2025-04-22 05:49:58 (1023 MB/s) - written to stdout [3980/3980]

root@Terraform-Server:~# echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com \$lsb_release -cs" main | sudo tee /etc/apt/sources.list.d/hashicorp.list
deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com jammy main
root@Terraform-Server:~# sudo apt update && sudo apt install terraform
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Get:5 https://apt.releases.hashicorp.com jammy InRelease [12.9 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [652 kB]
Get:8 https://apt.releases.hashicorp.com jammy/main amd64 Packages [177 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [107 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [2499 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [409 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [18.5 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [3336 kB]
Get:17 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [2222 kB]
```

```
root@Terraform-Server:~# terraform -version
Terraform v1.11.4
on linux_amd64
root@Terraform-Server:~# |
```

Step2: Create Accesskey and secret key in AWS for terraform to connect

The screenshot shows the AWS IAM Users page. The left sidebar has 'Identity and Access Management (IAM)' selected. Under 'Access management', 'Users' is selected, showing two entries: 'ANMOL' and 'Terraform'. Both users have a status of 'Active' and were created on '2024-04-22'. The 'Create user' button is visible at the top right of the user list.

User name	Path	Group	Last activity	MFA	password age	Console last sign-in
ANMOL	/	0	-	-	103 days	-
Terraform	/	0	-	-	-	-

us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#users/details/Terraform/create-access-key

Step 1
 Access key best practices & alternatives
Step 2 - optional
 Set description tag
Step 3
 Retrieve access keys

Access key best practices & alternatives Info

When using long-term credentials like access keys to improve your security, consider the following use cases and alternatives.

Use case

Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

Local code
You plan to use this access key in local application code running on your computer to access your AWS account.

Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 11:24 AM 22-Apr-25

Create access key | IAM | Global +

us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#users/details/Terraform/create-access-key

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 11:24 AM 22-Apr-25

IAM > Users > Terraform > Create access key

Access key created
This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Retrieve access keys Info

Step 2 - optional
 Set description tag
Step 3
 Retrieve access keys

Access key

Access key	Secret access key
<input type="text" value="AKIA34AMDIBPXJGPFH72"/>	<input type="password" value="*****"/> Show

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key best practices

- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the [best practices for managing AWS access keys](#).

[Download .CSV file](#) [Done](#)

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 11:25 AM 22-Apr-25

```
root@Terraform-Server:~# aws configure
AWS Access Key ID [None]: AKIA34AMDIBPXJGPFH72
AWS Secret Access Key [None]: OLCvNNigHMrRH+eFhrs8cUd59L1RNEVkMDpc
leUo
Default region name [None]:
Default output format [None]:
root@Terraform-Server:~#
root@Terraform-Server:~#
```



Step3: Write terraform configuration file to create an EC2 server and install ansible on it.

```
root@Terraform-Server:~/myproject
root@Terraform-Server:~# mkdir myproject
root@Terraform-Server:~# cd myproject
root@Terraform-Server:~/myproject# vim aws_infra.tf
root@Terraform-Server:~/myproject# vim aws_infra.tf
root@Terraform-Server:~/myproject# |
```



```
resource "aws_instance" "myec2-Dev" {
  ami           = "ami-0f9de6e2d2f067fca"
  instance_type = "t2.medium"
  key_name      = "mykey"
  subnet_id     = aws_subnet.subnet-1.id
  security_groups = [aws_security_group.sg-1.id]
  tags = [
    { Name = "Project-Dev" }
  ]
}

resource "aws_instance" "myec2-test" {
  ami           = "ami-0f9de6e2d2f067fca"
  instance_type = "t2.micro"
  key_name      = "mykey"
  subnet_id     = aws_subnet.subnet-1.id
  security_groups = [aws_security_group.sg-1.id]
  tags = [
    { Name = "Project-test" }
  ]
}

resource "aws_instance" "myec2-prod" {
  ami           = "ami-0f9de6e2d2f067fca"
  instance_type = "t2.micro"
  key_name      = "mykey"
  subnet_id     = aws_subnet.subnet-1.id
  security_groups = [aws_security_group.sg-1.id]
  tags = [
    { Name = "Project-prod" }
  ]
}
-- INSERT --
```

```
}
```

Plan: 10 to add, 0 to change, 0 to destroy.

```
aws_vpc.vpc: Creating...
aws_vpc.vpc: Creation complete after 1s [id=vpc-029b8b1c6e3a326a9]
aws_security_group.sg: Creating...
aws_route_table.rtb: Creating...
aws_internet_gateway.gw: Creating...
aws_subnet.subnet-1: Creating...
aws_internet_gateway.gw: Creation complete after 0s [id=igw-0e77aa1441cfcc24c5]
aws_route_table.rtb: Creation complete after 0s [id=rtb-091575de87a7547ca]
aws_route.route: Creating...
aws_route.route: Creation complete after 1s [id=r-rtb-091575de87a75a7ca1080289494]
aws_security_group.sg: Creation complete after 2s [id=sg-05f17fae89461e8a6]
aws_subnet.subnet-1: Still creating... [10s elapsed]
aws_subnet.subnet-1: Creation complete after 11s [id=subnet-072096b470819a337]
aws_instance.myec2-test: Creating...
aws_instance.myec2-Dev: Creating...
aws_route_table_association.a: Creating...
aws_instance.myec2-prod: Creating...
aws_route_table_association.a: Creation complete after 0s [id=rtbassoc-05c4ceefad583d63d]
aws_instance.myec2-test: Still creating... [10s elapsed]
aws_instance.myec2-Dev: Still creating... [10s elapsed]
aws_instance.myec2-prod: Still creating... [10s elapsed]
aws_instance.myec2-test: Still creating... [20s elapsed]
aws_instance.myec2-prod: Still creating... [20s elapsed]
aws_instance.myec2-test: Still creating... [30s elapsed]
aws_instance.myec2-Dev: Still creating... [30s elapsed]
aws_instance.myec2-prod: Still creating... [30s elapsed]
aws_instance.myec2-test: Creation complete after 32s [id=i-06ac2ba04af3b368d]
aws_instance.myec2-Dev: Creation complete after 32s [id=i-04ec1221755216d81]
aws_instance.myec2-test: Still creating... [40s elapsed]
aws_instance.myec2-test: Creation complete after 42s [id=i-0d5dbcb9e68dce752]
```

Apply complete! Resources: 10 added, 0 changed, 0 destroyed.

```
root@Terraform-Server:~/myproject#
```

The screenshot shows the AWS EC2 Instances page with four running instances listed:

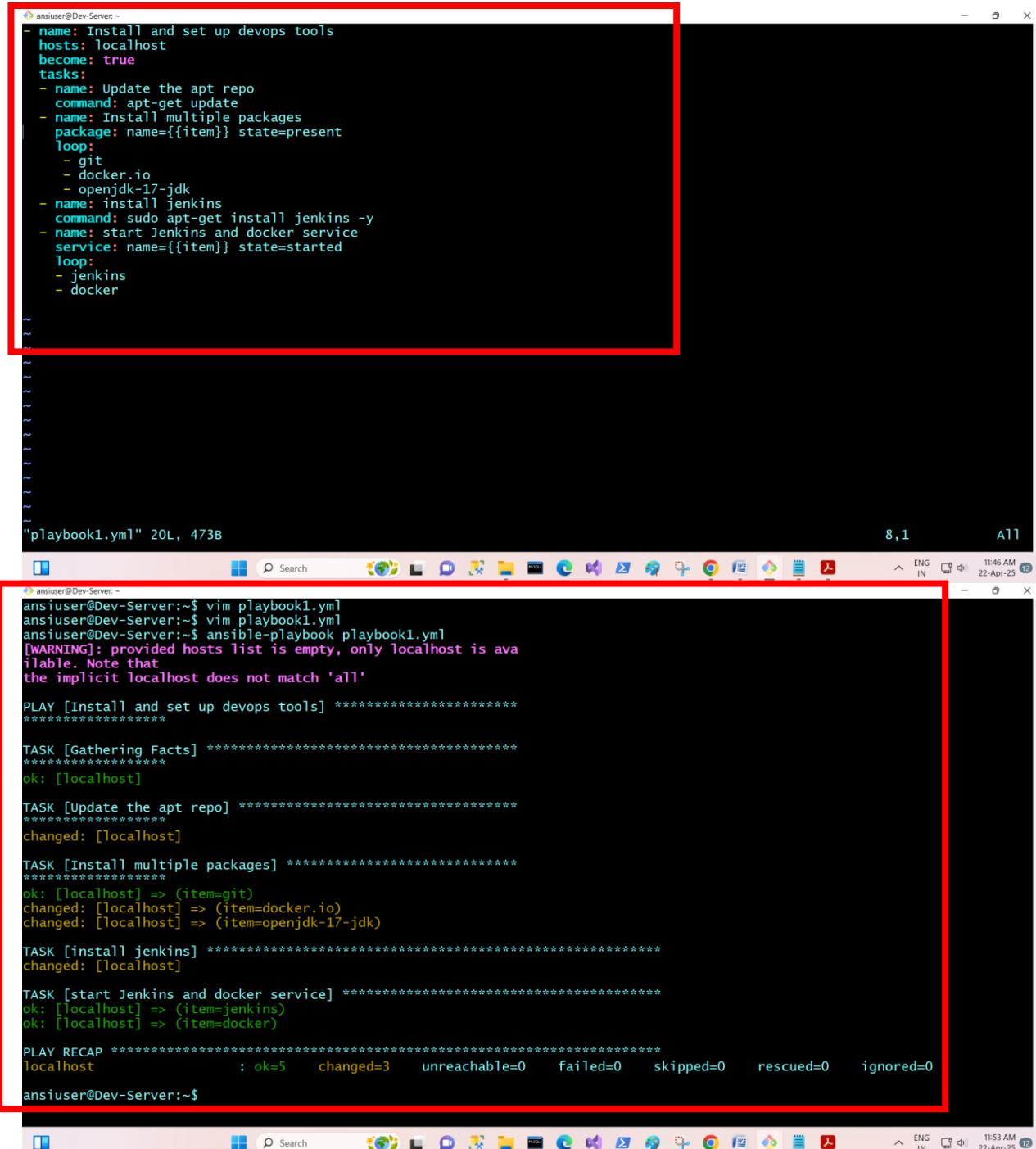
Name	Instance ID	Instance state	Instance type	Status check
Project-test	i-0d5dbcb9e68dce752	Running	t2.micro	Initializing
Project-Dev	i-04ec1221755216d81	Running	t2.medium	Initializing
Project-prod	i-06ac2ba04af3b368d	Running	t2.micro	Initializing
Terraform-Server	i-07f0aa5c3fb527350	Running	t2.micro	2/2 checks passed

Each instance has a green circular icon with a checkmark and a question mark icon next to it.

Step4: Install all the tools for CICD like (Ansible, Jenkins, Docker, Etc...) on DEV-Server

```
root@Dev-Server:~# ansible --version
ansible [core 2.17.10]
  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, Feb 4 2025, 14:57:36) [GCC 11.4.0] (/usr/bin/python3)
  jinja version = 3.0.3
  libyaml = True
root@Dev-Server:~#
```

Step5: Create a Playbook for Install DevOps Tools on Dev-Server



```
ansiuser@Dev-Server:~$ vim playbook1.yml
ansiuser@Dev-Server:~$ vim playbook1.yml
ansiuser@Dev-Server:~$ ansible-playbook playbook1.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Install and set up devops tools] ****
*****
TASK [Gathering Facts] ****
ok: [localhost]
TASK [Update the apt repo] ****
changed: [localhost]
TASK [Install multiple packages] ****
ok: [localhost] => (item=git)
changed: [localhost] => (item=docker.io)
changed: [localhost] => (item=openjdk-17-jdk)
TASK [install jenkins] ****
changed: [localhost]
TASK [start Jenkins and docker service] ****
ok: [localhost] => (item=jenkins)
ok: [localhost] => (item=docker)

PLAY RECAP ****
localhost                  : ok=5      changed=3      unreachable=0      failed=0      skipped=0      rescued=0      ignored=0
ansiuser@Dev-Server:~$
```

Step6: Create a Playbook2 for Install DevOps Tools on Test-Server

```
ansiuser@Dev-Server:~$ Vim ansible.cfg
ansiuser@Dev-Server:~$ ls
ansible.cfg myinventory playbook1.yml
ansiuser@Dev-Server:~$ vim playbook2.yml
ansiuser@Dev-Server:~$ ansible-playbook playbook2.yml

PLAY [Install and Set up Devops tools] ****
*****
TASK [gathering Facts] ****
[WARNING]: Platform linux on host 10.0.1.103 is using the discovered Python interpreter at /usr/bin/python3.10, but future installation of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.17/reference_appendices/interpreter_discovery.html for more information.
ok: [10.0.1.103]

TASK [update the apt repo] ****
*****
^changed: [10.0.1.103]

TASK [Install Multiple Packages] ****
*****
ok: [10.0.1.103] => {item=git}
changed: [10.0.1.103] => {item=docker.io}
changed: [10.0.1.103] => {item=openjdk-17-jdk}

TASK [Create a Jenkins root directory] ****
*****
changed: [10.0.1.103]

PLAY RECAP ****
*****
10.0.1.103 : ok=4    changed=3    unreachable=0
              failed=0   skipped=0   rescued=0   ignored=0
```

Step7: Create a Playbook2 for Install DevOps Tools on PROD-Server

```
[ansiuser@Dev-Server: ~]
[webserver]
10.0.1.103
10.0.1.120
~
```

```
thon
interpreter at /usr/bin/python3.10, but future installation of another
Python
interpreter could change the meaning of that path. See
https://docs.ansible.com/ansible-
core/2.17/reference_appendices/interpreter_discovery.html for more info
rmation.
ok: [10.0.1.120]
[WARNING]: Platform linux on host 10.0.1.103 is using the discovered Py
thon
interpreter at /usr/bin/python3.10, but future installation of another
Python
interpreter could change the meaning of that path. See
https://docs.ansible.com/ansible-
core/2.17/reference_appendices/interpreter_discovery.html for more info
rmation.
ok: [10.0.1.103]

TASK [update the apt repo] ****
***** changed: [10.0.1.103]
changed: [10.0.1.120]

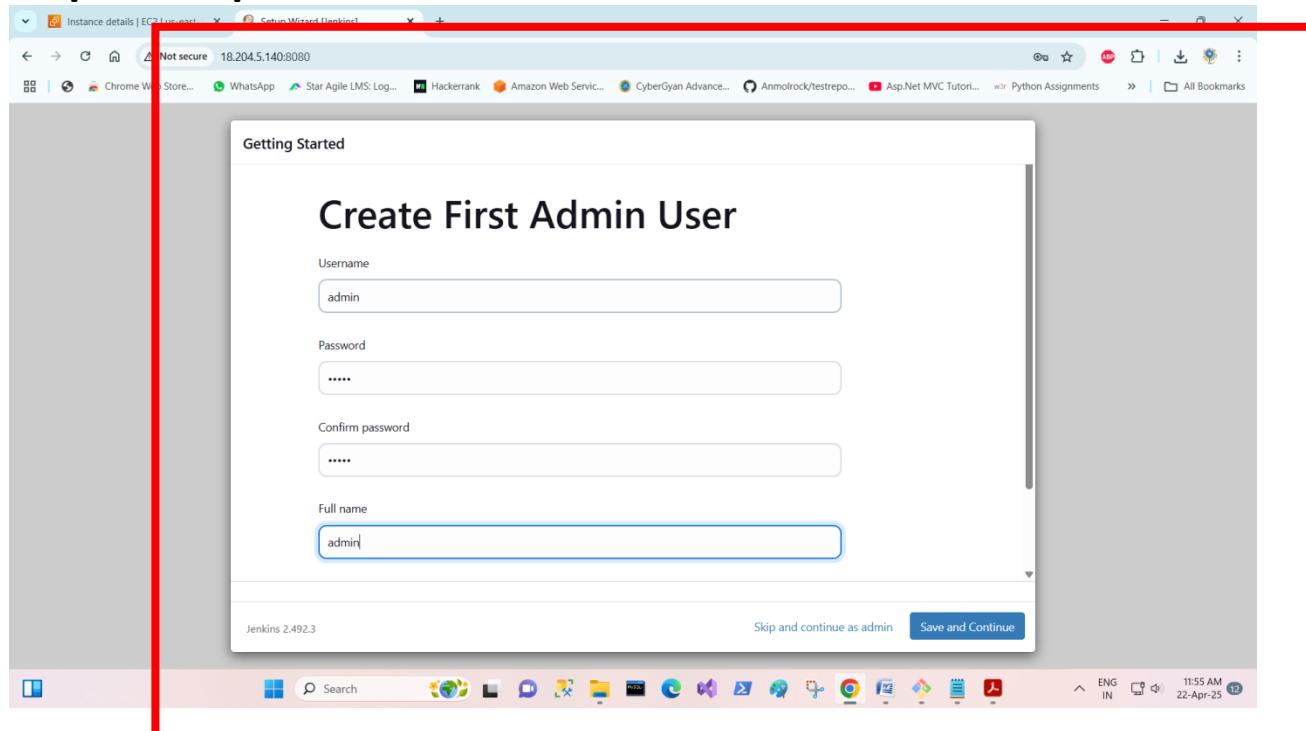
TASK [Install Multiple Packages] ****
***** ok: [10.0.1.120] => {item=git}
ok: [10.0.1.103] => {item=git}
ok: [10.0.1.103] => {item=docker.io}
ok: [10.0.1.103] => {item=openjdk-17-jdk}
changed: [10.0.1.120] => {item=docker.io}
changed: [10.0.1.120] => {item=openjdk-17-jdk}

TASK [Create a Jenkins root directory] ****
ok: [10.0.1.103]
changed: [10.0.1.120]

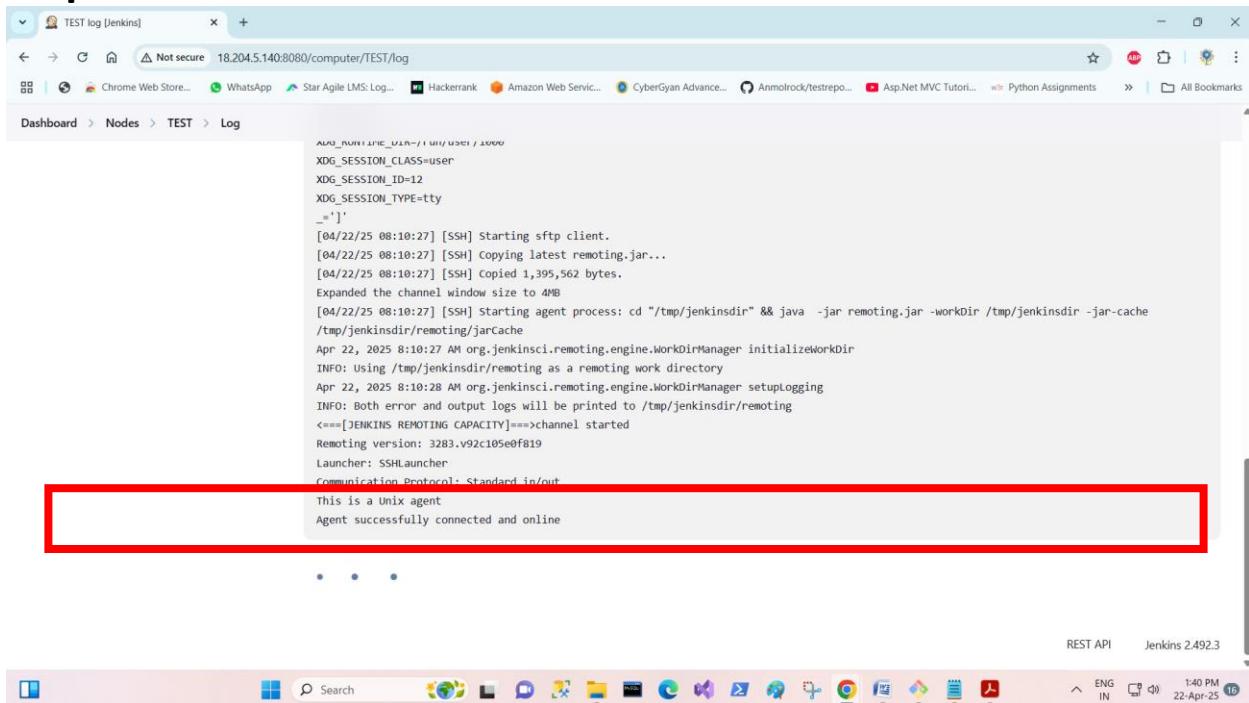
PLAY RECAP ****
10.0.1.103      : ok=4    changed=1    unreachable=0    failed=0
10.0.1.120      : ok=4    changed=3    unreachable=0    failed=0
skipped=0        rescued=0       ignored=0

ansiusr@Dev-Server:~$
```

Step8: Setup Jenkins Dashboard



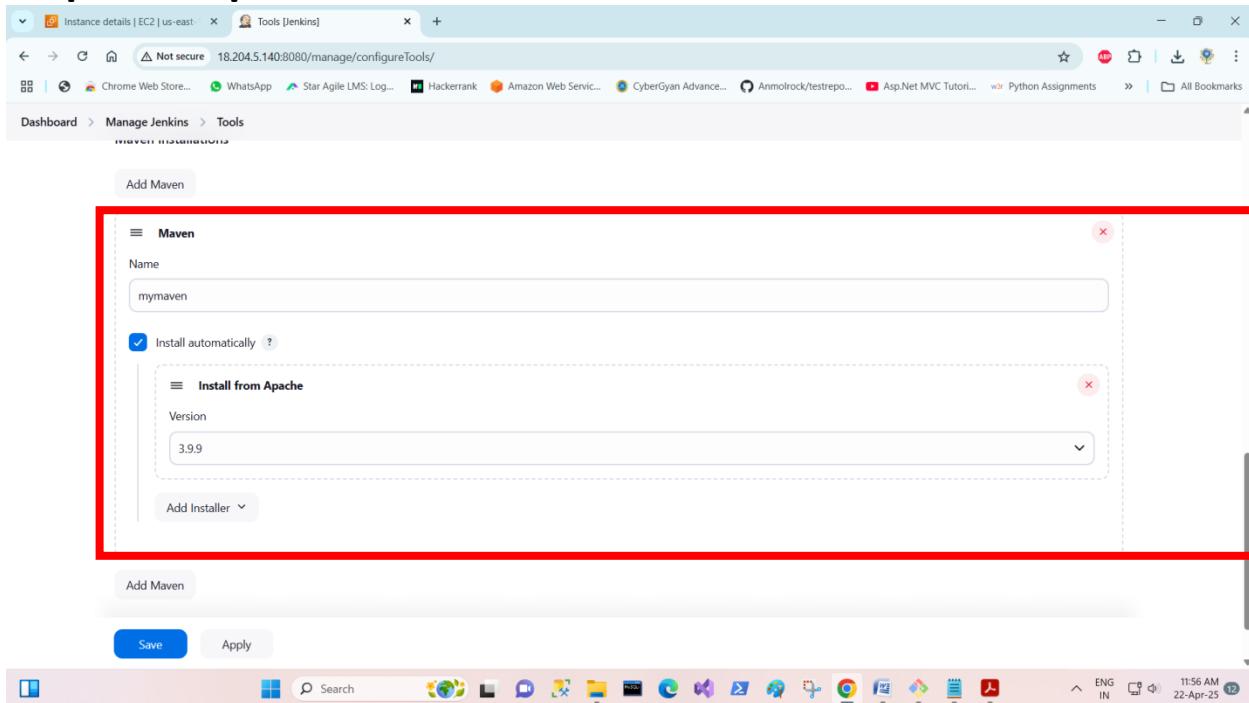
Step9: Then Create a Worker Nodes in Jenkins



A screenshot of a web browser displaying the Jenkins log for a node named "TEST". The log shows the configuration of an SSH agent and its successful connection to the Jenkins master. A red box highlights the final message indicating the agent is successfully connected and online.

```
AUG_22_2025 08:10:27] [SSH] Starting sftp client.  
[04/22/25 08:10:27] [SSH] Copying latest remoting.jar...  
[04/22/25 08:10:27] [SSH] Copied 1,395,562 bytes.  
Expanded the channel window size to 4MB  
[04/22/25 08:10:27] [SSH] Starting agent process: cd "/tmp/jenkinsdir" && java -jar remoting.jar -workDir /tmp/jenkinsdir -jar-cache /tmp/jenkinsdir/remoting/jarCache  
Apr 22, 2025 8:10:27 AM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir  
INFO: Using /tmp/jenkinsdir/remoting as a remoting work directory  
Apr 22, 2025 8:10:28 AM org.jenkinsci.remoting.engine.WorkDirManager setupLogging  
INFO: Both error and output logs will be printed to /tmp/jenkinsdir/remoting  
<===[JENKINS REMOTING CAPACITY]==>channel started  
Remoting version: 3283.v92c105e0f819  
Launcher: SSHLauncher  
Communication Protocol: Standard in/out  
This is a unix agent  
Agent successfully connected and online
```

Step9: Setup Maven in Jenkins



A screenshot of a web browser showing the "Add Maven" configuration page in Jenkins. The "Name" field is set to "mymaven" and the "Install automatically" checkbox is checked. A red box highlights the "Install from Apache" section where the "Version" dropdown is set to "3.9.9".

Add Maven

Maven

Name: mymaven

Install automatically

Install from Apache

Version: 3.9.9

Add Installer

Add Maven

Save Apply

Step10: Create a Pipeline Project in Jenkins

The screenshot shows two consecutive steps in the Jenkins interface:

Step 1: New Item Creation

The top window shows the Jenkins 'New Item' dialog. A red box highlights the main input area where 'Banking-finance-Project1' is typed into the 'Enter an item name' field. Below it, a list of item types is shown:

- 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**

An 'OK' button is visible at the bottom of the dialog.

Step 2: Pipeline Configuration

The bottom window shows the Jenkins 'Configuration' screen for the newly created 'Banking-finance-Project1'. A red box highlights the 'General' tab. On the left, a sidebar lists 'General', 'Triggers', 'Pipeline', and 'Advanced' tabs. The 'General' tab is active, showing the following configuration:

- Description**:

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 CI/CD. FinanceMe has decided to use AWS as primary cloud vendor provider to create various databases and application deployments.
- Enabled**: A checked checkbox.
- Options**:
 - Discard old builds
 - Do not allow concurrent builds
 - Do not allow the pipeline to resume if the controller restarts
 - Github project
 - Pipeline speed/durability override
- Buttons**: 'Save' and 'Apply' buttons at the bottom.

Configure Pipeline

Definition

Pipeline script

```
1 pipeline{  
2     agent any  
3     tools{  
4         maven 'mymaven'  
5     }  
6     stages{  
7         stage('clone Repo')  
8             steps{  
9                 git 'https://github.com/Sonal0409/banking-finance-Project1.git'  
10            }  
11        }  
12    }  
13}
```

Save Apply

Configure Pipeline

Definition

Pipeline script

```
29     stage('Build Image')  
30     {  
31         steps{  
32             sh 'docker build -t myimage:dev .'  
33         }  
34     }  
35     stage('Deploy the Image')  
36     {  
37         steps{  
38             sh 'docker run -d -P myimage:dev '  
39         }  
40     }  
41 }  
42 }
```

Use Groovy Sandbox ?

Pipeline Syntax

Advanced

Save Apply

```

[Pipeline] 
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] {
[Pipeline] stage
[Pipeline] {
[Pipeline] ( Deploy the Image)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker run -d -P myproject1:1
0284552f56cbf746f27842fcf60dc102d22c917a9108bea66e0e3bef14da374e
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

REST API Jenkins 2.492.3

BANKING-finance-Project1

Status: Green

Changes: 0

Build Now: Build Now

Configure: Configure

Delete Pipeline: Delete Pipeline

Stages: Stages

Rename: Rename

Pipeline Syntax: Pipeline Syntax

Builds

Build	Timestamp	Result
#1	6:33 AM	Success

Filter: Filter

Today: #1 6:33 AM

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.

Permalinks

- Last build (#1), 12 min ago



```
ansisuser@Dev-Server:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                         NAMES
3ef933712ce2        myimage:dev      "java -jar /app.jar"   13 seconds ago    Up 13 seconds    0.0.0.0:32773->8081/tcp, :::32773->8081/tcp   sad_perlman
612ad4c4ac68c       myproject1:4     "java -jar /app.jar"   3 minutes ago     Up 3 minutes     0.0.0.0:32772->8081/tcp, :::32772->8081/tcp   lucid_carver
8f78f7645c59a       myproject1:3     "java -jar /app.jar"   19 minutes ago    Up 19 minutes    0.0.0.0:32771->8081/tcp, :::32771->8081/tcp   loving_shamir
74f6b104e05e        myproject1:2     "java -jar /app.jar"   21 minutes ago    Up 21 minutes    0.0.0.0:32770->8081/tcp, :::32770->8081/tcp   kind_murdock
ba66623f9bf8        efaaa330db184   "java -jar /app.jar"   2 hours ago      Up 2 hours       0.0.0.0:32769->8081/tcp, :::32769->8081/tcp   vigorous_good
0284552f56cb        efaaa330db184   "java -jar /app.jar"   2 hours ago      Up 2 hours       0.0.0.0:32768->8081/tcp, :::32768->8081/tcp   focused_kalam
ansiuser@Dev-Server:~$
```

Step11: Monitoring Using Prometheus and Grafana

The screenshot shows the Jenkins Plugins page. A red box highlights the search results for 'prome'. The results list three items:

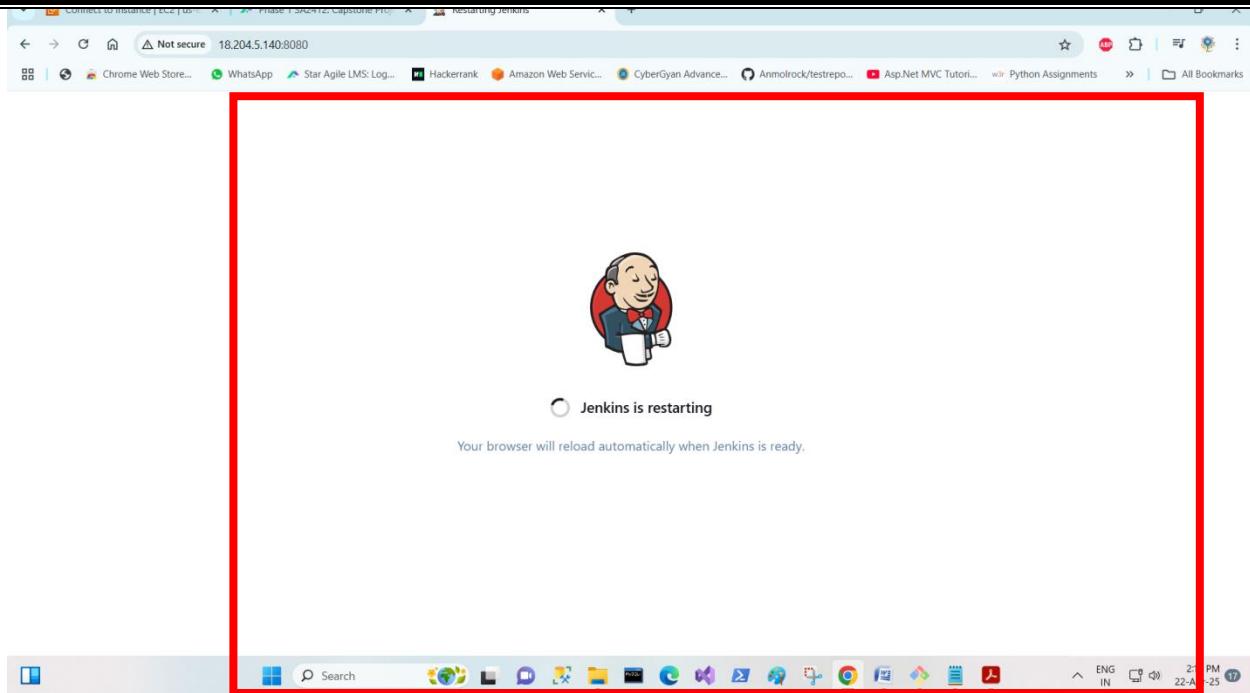
- Prometheus metrics**: Version 0.19.v50953a_c560dd, Released 2 mo 2 days ago. Description: Jenkins Prometheus Plugin expose an endpoint (default /prometheus) with metrics where a Prometheus Server can scrape.
- Otel agent host metrics monitoring**: Version 1.4.1, Released 20 days ago. Description: This plugin allows monitoring of Jenkins agents by deploying Prometheus node exporters and Otel collectors to them and linking to a Grafana dashboard displaying those gathered metrics.
- Cortex Metrics 1.0.1**: Adds the ability to publish run results to Cortex directly using the Prometheus push endpoint. Released 4 yr 1 mo ago.

On the left sidebar, 'Available plugins' is selected. At the bottom right, it says REST API Jenkins 2.492.3.

The screenshot shows the Jenkins Plugins page. A red box highlights the 'Download progress' section. It lists various Jenkins components and their download status:

Component	Status
Ionicons API	Success
Folders	Success
OWASP Markup Formatter	Success
ASM API	Success
JSON Path API	Success
Structs	Success
Pipeline: Step API	Success
Token Macro	Success
Build Timeout	Success
bouncycastle API	Success
Credentials	Success
Plain Credentials	Success
Variant	Success
SSH Credentials	Success
Credentials Binding	Success

On the left sidebar, 'Download progress' is selected. At the bottom right, it says REST API Jenkins 2.492.3.



```
# root@prometheus: ~
# my_global_config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label 'job=<job_name>' to any timeseries scraped from this config.
  - job_name: "prometheus"

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

    static_configs:
      - targets: ["localhost:9090"]
    - job_name: 'Jenkins_Server'
      metrics_path: '/prometheus'
      static_config:
        - targets: ['18.204.5.140:8080']
-
```

A screenshot of a terminal window displaying a Prometheus configuration file. The configuration includes global settings for scraping and evaluation intervals, an alerting section with a single static configuration for an alertmanager, and a scrape_configs section that defines a job named 'prometheus' which scrapes the local host at port 9090 and a job named 'Jenkins_Server' at port 8080. A red box highlights the entire configuration text.

The screenshot shows the Prometheus Query interface. The URL is 54.163.9.153:9090/query. The main area has a red border and displays the message "No data queried yet". There is a "Add query" button at the bottom left.



The screenshot shows the Prometheus Status > Target health interface. It lists two targets:

Target	Endpoint	Labels	Last scrape	State
Jenkins_Server	http://18.204.5.140:8080/prometheus	instance="18.204.5.140:8080" job="Jenkins_Server"	8.653s ago	21ms UP
prometheus	http://localhost:9090/metrics	instance="localhost:9090" job="prometheus"	5.106s ago	4ms UP



```

● grafana-server.service - Grafana instance
   Loaded: loaded (/lib/systemd/system/grafana-server.service; disabled; vendor preset: enabled)
   Active: active (running) since Tue Apr 22 09:03:19 UTC; 5s ago
     Docs: http://docs.grafana.org
 Main PID: 2649 (grafana)
   Tasks: 6 (limit: 1129)
    Memory: 211.4M
      CPU: 1.158s
     CGroup: /system.slice/grafana-server.service
             └─2649 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/grafana/grafana-server.pid --packaging=deb cfg:def>

Apr 22 09:03:24 ip-172-31-30-193 grafana[2649]: logger=migrator t=2025-04-22T09:03:24.752984873Z level=info msg="Executing migration" id="create index IDX_lo>
Apr 22 09:03:24 ip-172-31-30-193 grafana[2649]: logger=migrator t=2025-04-22T09:03:24.7539445Z level=info msg="Migration successfully executed" id="create in>
Apr 22 09:03:24 ip-172-31-30-193 grafana[2649]: logger=migrator t=2025-04-22T09:03:24.761265368Z level=info msg="Executing migration" id="copy login_attempt>
Apr 22 09:03:24 ip-172-31-30-193 grafana[2649]: logger=migrator t=2025-04-22T09:03:24.761860482Z level=info msg="Migration successfully executed" id="copy log>
Apr 22 09:03:24 ip-172-31-30-193 grafana[2649]: logger=migrator t=2025-04-22T09:03:24.769473192Z level=info msg="Executing migration" id="drop login_attempt_t>
Apr 22 09:03:24 ip-172-31-30-193 grafana[2649]: logger=migrator t=2025-04-22T09:03:24.770346315Z level=info msg="Migration successfully executed" id="drop To>
Apr 22 09:03:24 ip-172-31-30-193 grafana[2649]: logger=migrator t=2025-04-22T09:03:24.779912122Z level=info msg="Executing migration" id="create user auth_t>
Apr 22 09:03:24 ip-172-31-30-193 grafana[2649]: logger=migrator t=2025-04-22T09:03:24.780512672Z level=info msg="Migration successfully executed" id="create >
Apr 22 09:03:24 ip-172-31-30-193 grafana[2649]: logger=migrator t=2025-04-22T09:03:24.788020937Z level=info msg="Executing migration" id="create index IDX_us>
Apr 22 09:03:24 ip-172-31-30-193 grafana[2649]: logger=migrator t=2025-04-22T09:03:24.790194446Z level=info msg="Migration successfully executed" id="create >
```

```

Lines 1-21/21 (END)



Home > Connections > Data sources > prometheus

Incremental querying (beta)

Disable recording rules (beta)

Other

Custom query parameters Example: max\_source\_resolution=5m&timeout=

HTTP method POST

Use series endpoint

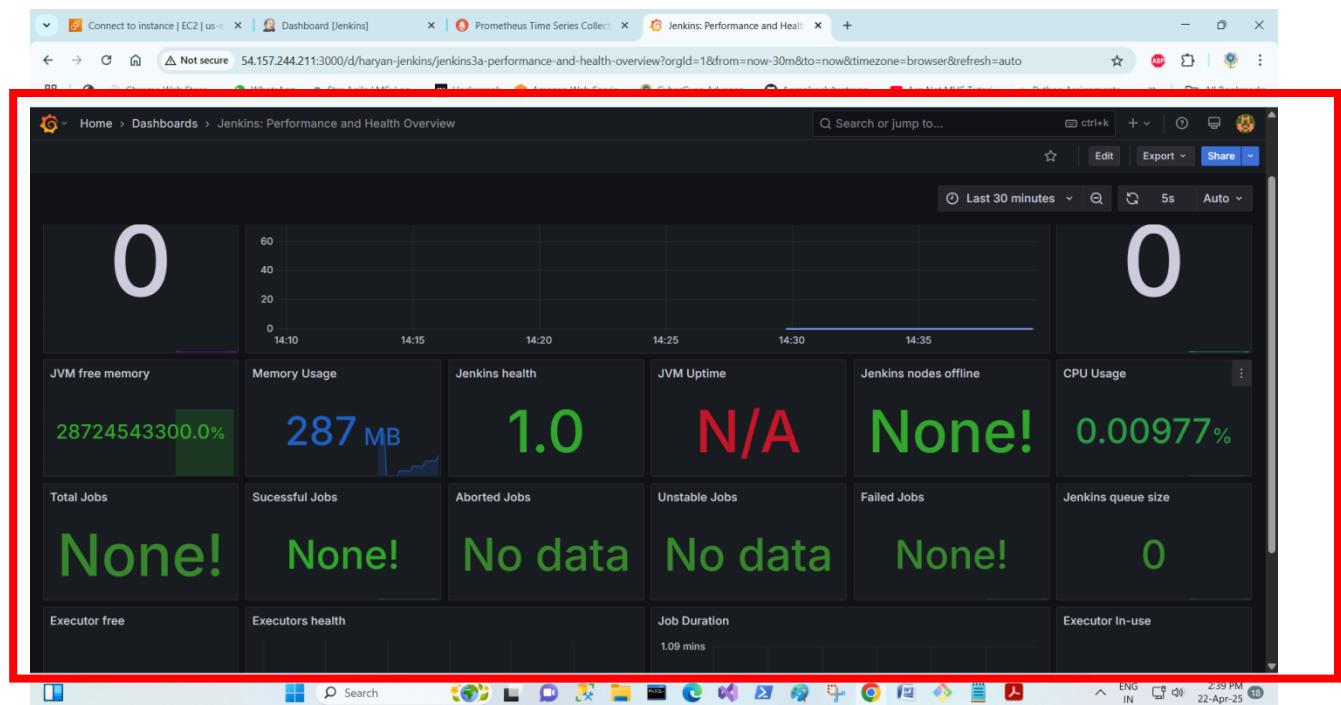
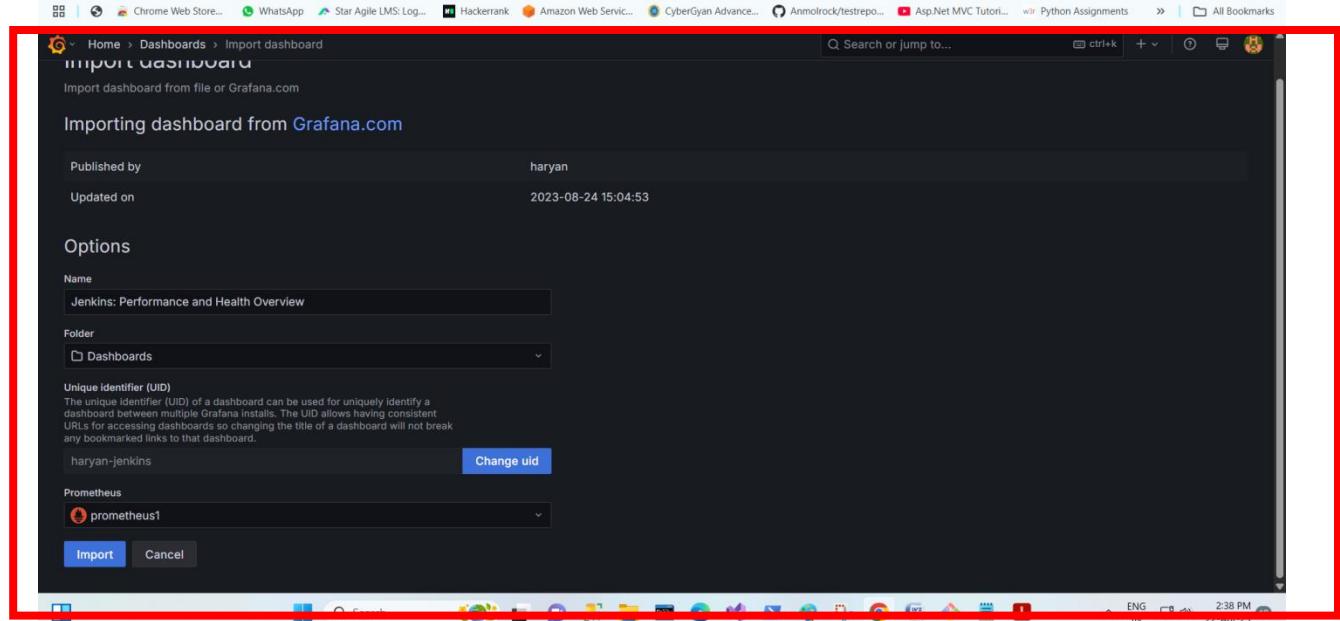
Exemplars + Add

✓ Successfully queried the Prometheus API.

Next, you can start to visualize data by [building a dashboard](#), or by querying data in the [Explore view](#).

Delete Save & test





## Output→

The screenshot shows a web browser window with a red border around the main content area. The address bar displays the URL `18.204.5.140:8080`. The page title is "CBS". The header features a logo with the letters "CIAL" and navigation links for HOME, ABOUT, SERVICES, TEAM, and CONTACT US. The main section has a large heading "CUSTOMER BANKING SERVICES" and a subtext "We provide the World's best in class Banking Solutions and Services." Below the heading are two buttons: "READ MORE" (red) and "CONTACT US" (blue). To the right is a stylized illustration of two people working with a computer and a laptop, surrounded by charts and graphs.

The screenshot shows a web browser window with a red border around the main content area. The address bar displays the URL `18.204.5.140:8080/index.html`. The page title is "CBS". The main content area is divided into four colored boxes: blue ("ACCOUNTING"), red ("ADVISOR"), blue ("INVESTMENT"), and blue ("FINANCIAL"). Each box contains a "Read More" button. Below this section is a heading "OUR TEAM AND EXPERTS" followed by a quote: "It is a long established fact that a reader will be distracted by the readable content of a". At the bottom are four circular profile pictures of men, each with a "Read More" button below it. The taskbar at the bottom shows various application icons.