

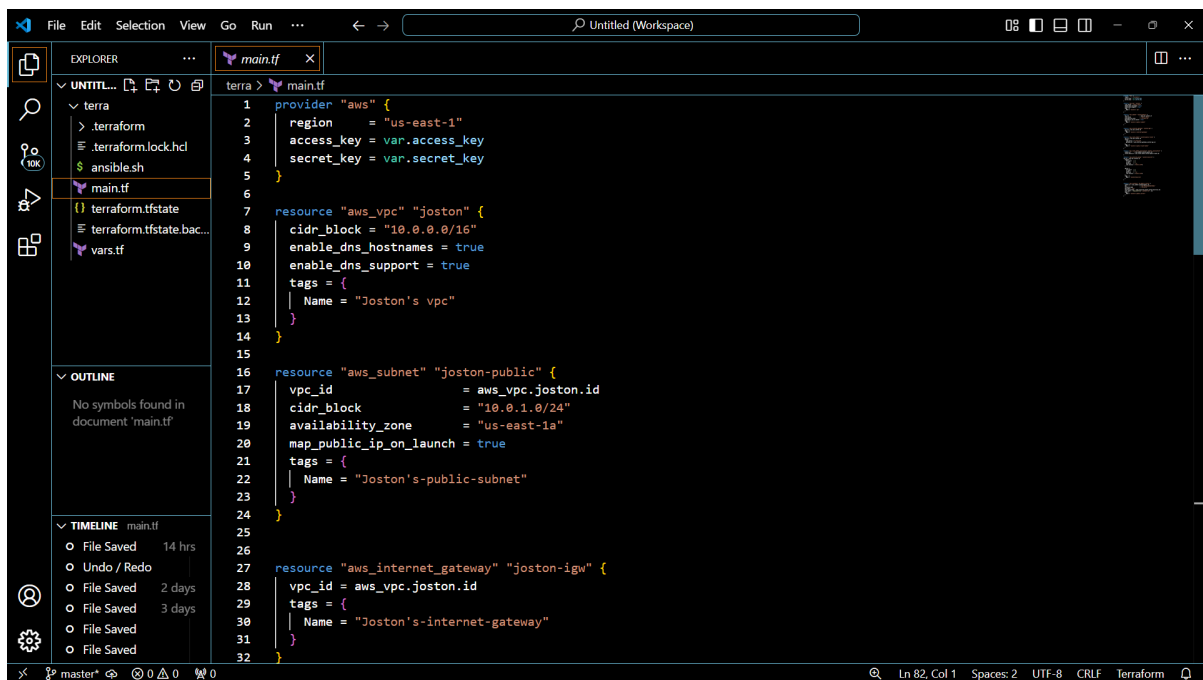
PROJECT-1

CERTIFICATION PROJECT – FINANCE ME

BANKING AND FINANCE DOMAIN

STEP-1


FIRST WRITE THE TERRAFORM FILE TO CREATE VPC,SUBNET,INTERNET GATEWAY AND ONE UNBUNTU INSTANCE AND INSTALL ASNIBLE ON THE INSTANCE.



```

1 provider "aws" {
2   region = "us-east-1"
3   access_key = var.access_key
4   secret_key = var.secret_key
5 }
6
7 resource "aws_vpc" "joston" {
8   cidr_block = "10.0.0.0/16"
9   enable_dns_hostnames = true
10  enable_dns_support = true
11  tags = {
12    Name = "Joston's vpc"
13  }
14 }
15
16 resource "aws_subnet" "joston-public" {
17   vpc_id = aws_vpc.joston.id
18   cidr_block = "10.0.1.0/24"
19   availability_zone = "us-east-1a"
20   map_public_ip_on_launch = true
21   tags = {
22     Name = "Joston's-public-subnet"
23   }
24 }
25
26 resource "aws_internet_gateway" "joston-igw" {
27   vpc_id = aws_vpc.joston.id
28   tags = {
29     Name = "Joston's-internet-gateway"
30   }
31 }
32

```



```

34 resource "aws_route_table" "joston-public-route" {
35   vpc_id = aws_vpc.joston.id
36   routes {
37     cidr_block = "0.0.0.0/0"
38     gateway_id = aws_internet_gateway.joston-igw.id
39   }
40   tags = {
41     Name = "Joston's-public-route-table"
42   }
43 }
44
45 resource "aws_route_table_association" "public_association" {
46   subnet_id = aws_subnet.joston-public.id
47   route_table_id = aws_route_table.joston-public-route.id
48 }
49
50 resource "aws_security_group" "joston-security" {
51   vpc_id = aws_vpc.joston.id
52   ingress {
53     from_port = 0
54     to_port = 0
55     protocol = "-1"
56     cidr_blocks = ["0.0.0.0/0"]
57   }
58   egress {
59     from_port = 0
60     to_port = 0
61     protocol = "-1"
62     cidr_blocks = ["0.0.0.0/0"]
63   }
64   tags = {
65     Name = "Joston-Security"
66   }
67 }
68
69
70 resource "aws_instance" "my_public_server" {
71   subnet_id = aws_subnet.joston-public.id
72   ami = "ami-0e2c8ca4b6378d8c"
73   instance_type = "t2.micro"
74   key_name = "jst"
75   security_groups = [aws_security_group.joston-security.id]
76   user_data = templatefile("./ansible.sh", {})
77   tags = {
78     Name = "joston's_public_server"
79   }
80 }
81

```

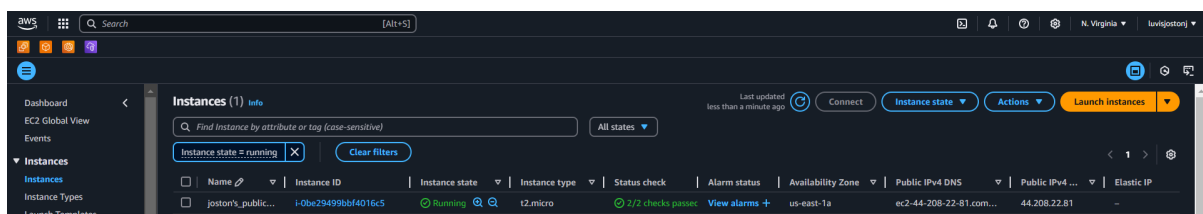
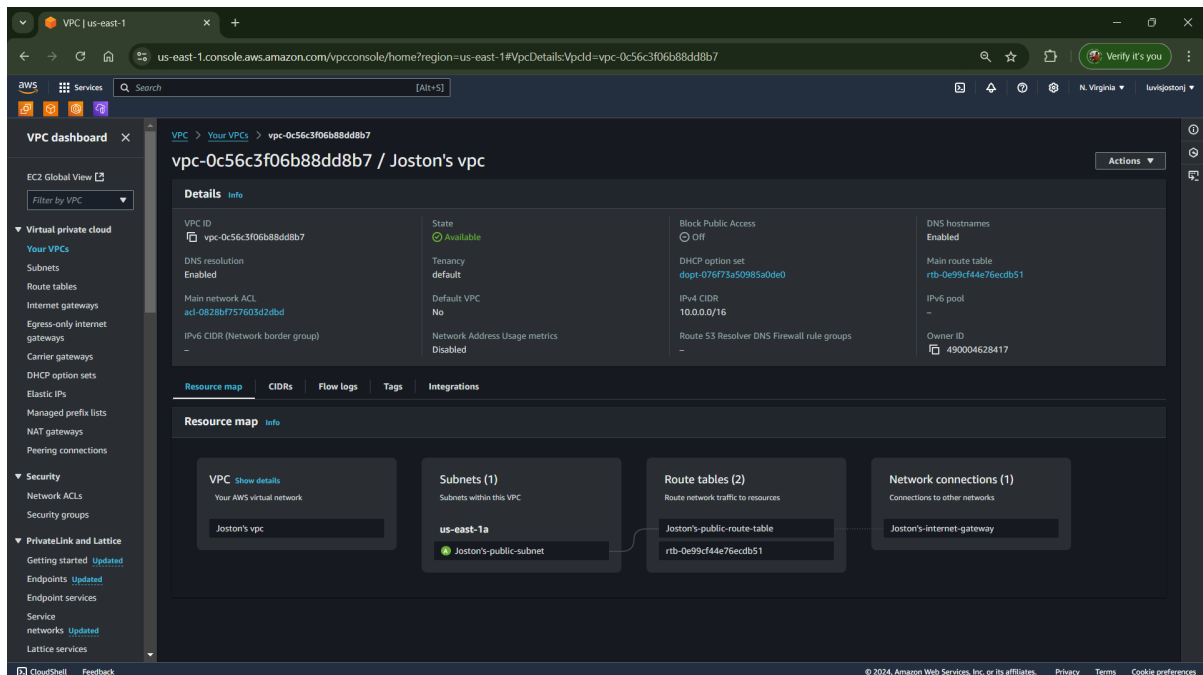


IM GONNA ATTACH TEMPLATEFILE FILE TO TERRAFORM FILE TO INSTALL ANSIBLE ON THE UBUNTU INSTANCE THAT WE GONNA CREATE USING TERRAFORM.

```
File Edit Selection View Go Run Terminal Help
$ ansible
1 #!/bin/bash
2 terra > $ ansible
3 sudo apt update
4 sudo apt install software-properties-common
5 sudo add-apt-repository --yes --update ppa:ansible/ansible
6 sudo apt install ansible -y
```

NOW APPLY, TERRAFORM FILE CREATE ENTIRE INFRASTRUCTURE FOR YOU IT WILL CREATE VPC,SUBNET AND ROUTE TABLES AND INTERNET GATEWAY AND AND ONE INSTANCE AND ATTACH SECURITY GROUP FOR IT AND INSTALL ANSIBLE ON IT.

```
Plan: 7 to add, 0 to change, 0 to destroy.
aws_vpc.joston: Creating...
aws_vpc.joston: Still creating... [10s elapsed]
aws_vpc.joston: Creation complete after 16s [id=vpc-0c56c3f06b88dd8b7]
aws_subnet.joston-public: Creating...
aws_internet_gateway.joston-igw: Creating...
aws_security_group.joston-security: Creating...
aws_route_table.joston-public-route: Creating...
aws_route_table.joston-public-route: Creation complete after 3s [id=rtb-0c723942c8f5b2ec8]
aws_security_group.joston-security: Creation complete after 7s [id=sg-846b0f0c1fd3dae96]
aws_subnet.joston-public: Still creating... [10s elapsed]
aws_subnet.joston-public: Creation complete after 13s [id=subnet-01550f454cef52277]
aws_route_table_association.joston-public-association: Creating...
aws_instance.my_public_server: Creating...
aws_route_table_association.joston-public-association: Creation complete after 1s [id=rtbassoc-00f0f0556194d4269]
aws_instance.my_public_server: Still creating... [10s elapsed]
aws_instance.my_public_server: Creation complete after 15s [id=i-0bc29499bbf4016c5]
Apply complete! Resources: 7 added, 0 changed, 0 destroyed.
PS C:\Users\luis> terraform j\Joston\Terraform
```



STEP-2

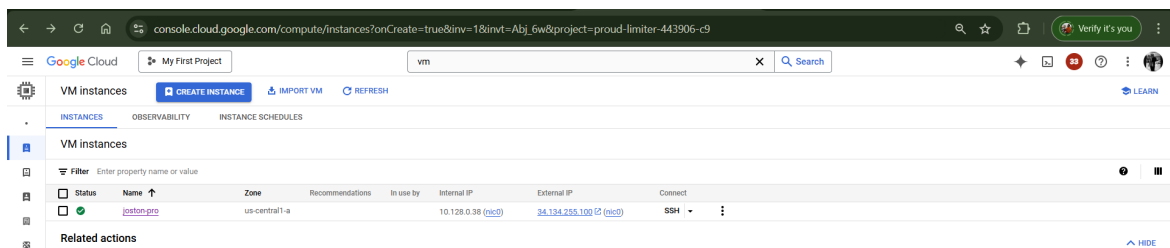


NOW CONNECT TO THAT INSTANCE THAT YOU CREATED FROM TERRAFORM AND YOU HAVE INSTALLED ANSIBLE ON IT.

ANSIBLE CONTROLLER

```
root@ip-10-0-1-226:~# ansible --version
ansible [core 2.17.7]
  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 = 3.12.5 (main, Sep 11 2024, 14:17:37) [GCC 13.2.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
root@ip-10-0-1-226:~#
```

I HAVE CONNECT ONE UBUNTU INSTANCE FROM GCP AS A ANSIBLE WORKER VIA SSH.



```
joston@ip-10-0-1-226:~$ ansible -m ping joston
joston:
  ansible_facts: {
    "discovered_interpreter_python": "/usr/bin/python3.8"
  },
  "changed": false,
  "ping": "pong"
joston@ip-10-0-1-226:~$
```

START WTRING THE ANSIBLE PLYBOOK.YML FILE TO INSTALL JENKINS,DOCKER,GIT ,GIVE PERMISSION TO JENKINS USER TO RUN DOCKER COMMANDS AND START THE JENKINS AND DOCKER SERVICES.

To install jenkins im gonna use jenkins.sh file and im gonna execute this .sh file using playbook.

```
joston@ip-10-0-1-226:~$ vi jenkins.sh

sudo apt update
sudo apt install openjdk-17-jre-headless -y
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/" | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins -y
```

```
- name: my playbook to do the project no-1
  hosts: joston
  become: true
  tasks:
    - name: update the packages
      command: apt update
    - name: copy the jenkins.sh file to ansible worker
      copy: src=/home/joston/jenkins.sh dest=/home/joston
    - name: installing jenkins using jenkins.sh file
      command: bash jenkins.sh
    - name: install docker and git on ansible worker
      package: name={{item}} state=present
      loop:
        - docker.io
        - git
    - name: start the jenkins and docker services
      service: name={{item}} state=started
      loop:
        - jenkins
        - docker
    - name: give permission jenkins user to run docker cmds
      command: chmod 777 /var/run/docker.sock
```

STEP-3

NOW RUN THE PLAYBOOK FILE.

```
joston@ip-10-0-1-226:~$ ls
ansible.cfg  jenkins.sh  myinventory  playbook.yml
joston@ip-10-0-1-226:~$ ansible-playbook playbook.yml

PLAY [my playbook to do the project no-1] *****

TASK [Gathering Facts] *****
ok: [34.134.255.100]

TASK [update the packages] *****
changed: [34.134.255.100]

TASK [copy the jenkins.sh file to ansible worker] *****
changed: [34.134.255.100]

TASK [installing jenkins using jenkins.sh file] *****
changed: [34.134.255.100]

TASK [install docker and git on ansible worker] *****
changed: [34.134.255.100] => (item=docker.io)
ok: [34.134.255.100] => (item=git)

TASK [start the jenkins and docker services] *****
ok: [34.134.255.100] => (item=jenkins)
ok: [34.134.255.100] => (item=docker)

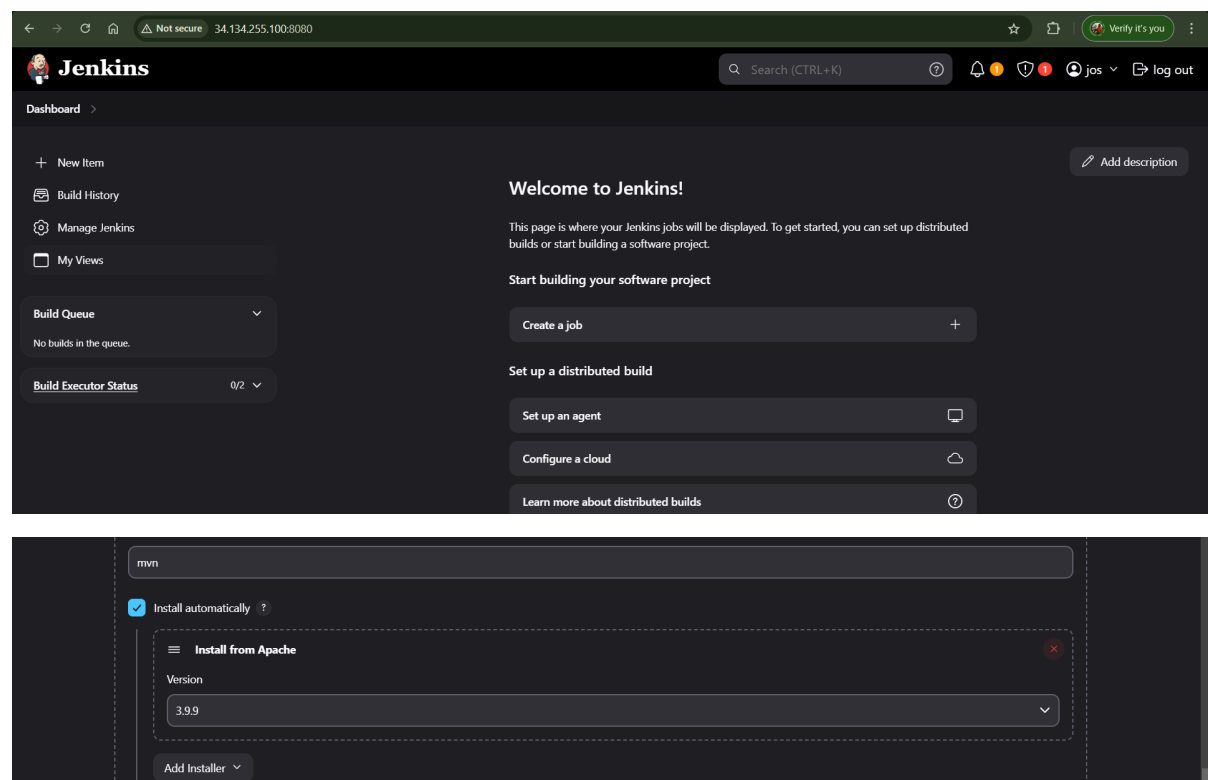
TASK [give permission jenkins user to run docker commads] *****
changed: [34.134.255.100]

PLAY RECAP *****
34.134.255.100      : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

joston@ip-10-0-1-226:~$
```

STEP-4

NOW CONNECT TO JENKINS AND AND CONFIGURE MAVEN TOOL.

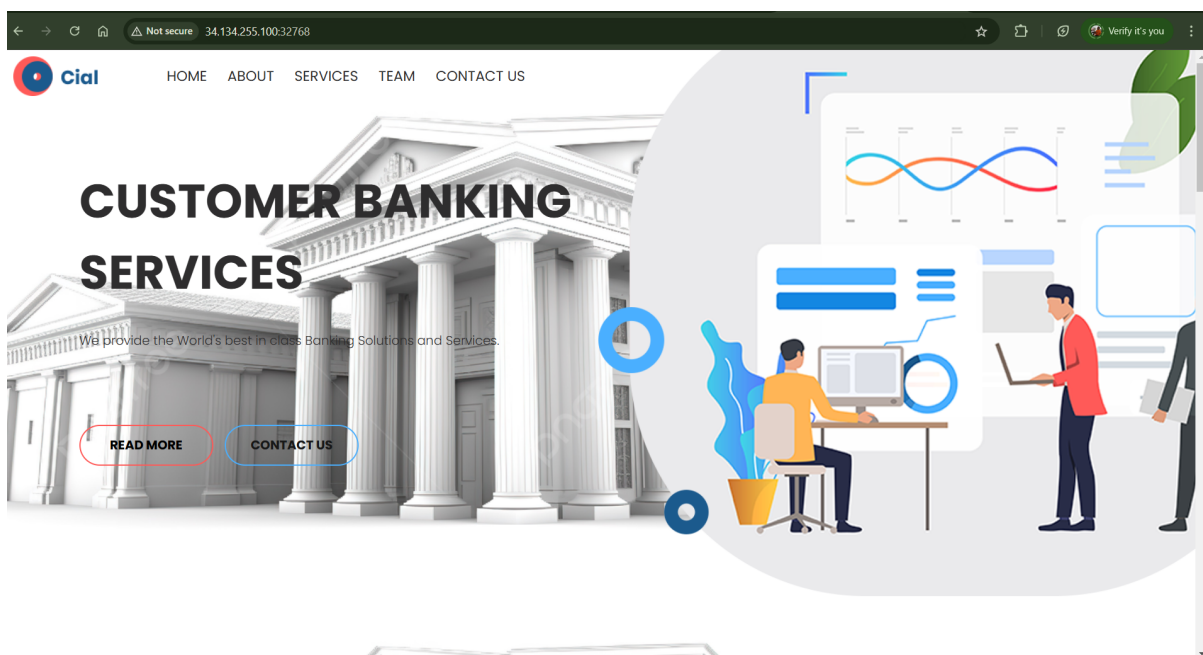


The screenshot shows the Jenkins Dashboard in a web browser. The dashboard includes a sidebar with links to 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. The main content area displays 'Welcome to Jenkins!' and 'Start building your software project' with buttons for 'Create a job', 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'. Below the dashboard, a configuration window for 'mvn' is shown, with 'Install automatically' checked and 'Install from Apache' selected. The version '3.9.9' is chosen from a dropdown menu, and an 'Add Installer' button is visible at the bottom.


```
ssh.cloud.google.com/v2/ssh/projects/proud-limiter-443906-c9/zones/us-central1-a/instances/joston-pro?authuser=0&hl=en_US&projectNumber=115255736637&useAdminProxy=true
SSH-in-browser
joston@joston-pro:~$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
joston latest 425391a8a855 11 minutes ago 690MB
openjdk 11 47a932d998b7 2 years ago 654MB
joston@joston-pro:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
69906a8db11 joston "java -jar /app.jar" 11 minutes ago Up 11 minutes 0.0.0.0:32768->8081/tcp, :::32768->8081/tcp quizzical_gould
joston@joston-pro:~$
```

You can see created image and container with port number.

Now access the banking app from internet using container port.



We successfully clone the repo and test the code and build the code and built a docker image and Created a docker container from that image.

STEP-6

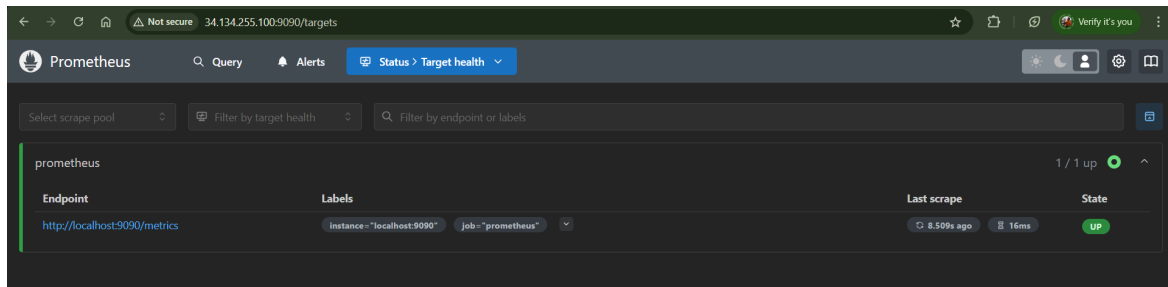
NOW INSTALL PROMETHEUS AND GRAFANA TO MONITER THE JENKINS.

I HAVE INSTALLED PROMETHEUS IN MY INSTANCE.

```
joston@joston-pro:~/tmp/prometheus-3.0.0-linux-amd64$ bash 4.sh
Created symlink /etc/systemd/system/multi-user.target.wants/prometheus.service → /etc/systemd/system/prometheus.service.
* prometheus.service - Prometheus
   Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2024-12-13 11:10:12 UTC; 464ms ago
     Main PID: 20662 (prometheus)
       Tasks: 8 (limit: 4680)
      Memory: 27.7M
     CGroup: /system.slice/prometheus.service
            └─20662 /usr/local/bin/prometheus --config.file /etc/prometheus/prometheus.yml --storage.tsdb.path /var/lib/prometheus/ --web.console.templates=/etc/prometheus/consoles --web.conso

Dec 13 11:10:12 joston-pro prometheus[20662]: time=2024-12-13T11:10:12.442Z level=INFO source=main.go:770 msg="Leaving GOMAXPROCS=2: CPU quota undefined" component=automaxprocs
Dec 13 11:10:12 joston-pro prometheus[20662]: time=2024-12-13T11:10:12.449Z level=INFO source=web.go:650 msg="Start listening for connections" component=web address=0.0.0.0:9090
Dec 13 11:10:12 joston-pro prometheus[20662]: time=2024-12-13T11:10:12.452Z level=INFO source=main.go:1231 msg="Starting TSDB ..."
Dec 13 11:10:12 joston-pro prometheus[20662]: time=2024-12-13T11:10:12.482Z level=INFO source=tls_config.go:347 msg="Listening on" component=web address=[::]:9090
Dec 13 11:10:12 joston-pro prometheus[20662]: time=2024-12-13T11:10:12.482Z level=INFO source=tls_config.go:350 msg="TLS is disabled." component=web http2=false address=[::]:9090
Dec 13 11:10:12 joston-pro prometheus[20662]: time=2024-12-13T11:10:12.496Z level=INFO source=head.go:688 msg="Replaying on-disk memory mappable chunks if any" component=tsdb
Dec 13 11:10:12 joston-pro prometheus[20662]: time=2024-12-13T11:10:12.496Z level=INFO source=head.go:715 msg="On-disk memory mappable chunks replay completed" component=tsdb duration=2.933ps
Dec 13 11:10:12 joston-pro prometheus[20662]: time=2024-12-13T11:10:12.496Z level=INFO source=head.go:723 msg="Replaying WAL, this may take a while" component=tsdb
Dec 13 11:10:12 joston-pro prometheus[20662]: time=2024-12-13T11:10:12.498Z level=INFO source=head.go:795 msg="WAL segment loaded" component=tsdb segment=0 maxSegment=0
Dec 13 11:10:12 joston-pro prometheus[20662]: time=2024-12-13T11:10:12.498Z level=INFO source=head.go:832 msg="WAL replay completed" component=tsdb checkpoint_replay_duration=72.407ps wal_repla
Since 1-13/13 (23%)
```

Its up and running.



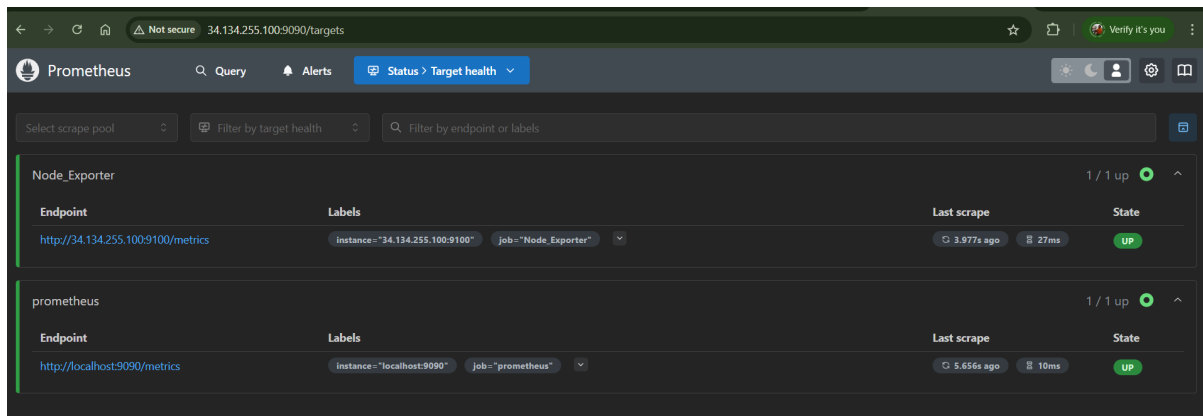
NOW INSTALL NODE EXPORTER TO COLLECT THE METRICS FROM JENKINS.

```
joston@joston-pro:~$ sudo systemctl status node_exporter
● node_exporter.service - Node Exporter
   Loaded: loaded (/etc/systemd/system/node_exporter.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2024-12-13 11:14:45 UTC; 8s ago
     Main PID: 20809 (node_exporter)
       Tasks: 4 (limit: 4680)
      Memory: 2.1M
     CGroup: /system.slice/node_exporter.service
            └─20809 /usr/local/bin/node_exporter

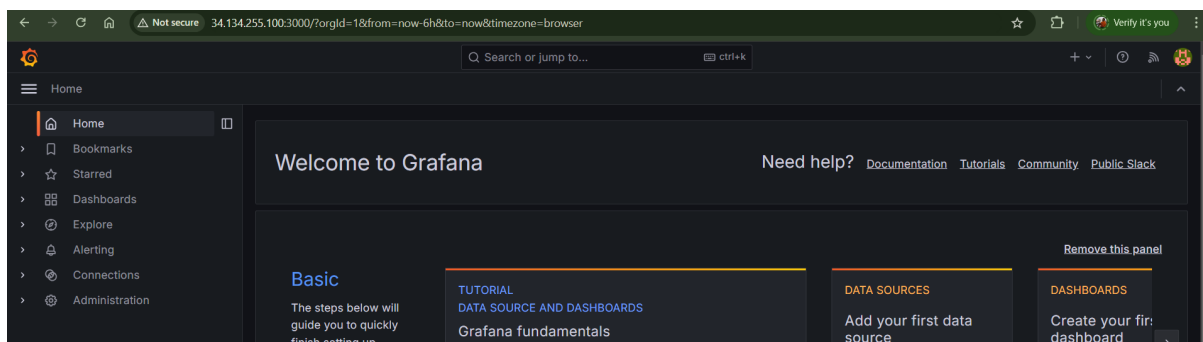
Dec 13 11:14:45 joston-pro node_exporter[20809]: ts=2024-12-13T11:14:45.266Z caller=node_exporter.go:118 level=info collector=time
Dec 13 11:14:45 joston-pro node_exporter[20809]: ts=2024-12-13T11:14:45.266Z caller=node_exporter.go:118 level=info collector=timex
Dec 13 11:14:45 joston-pro node_exporter[20809]: ts=2024-12-13T11:14:45.266Z caller=node_exporter.go:118 level=info collector=udp_queues
Dec 13 11:14:45 joston-pro node_exporter[20809]: ts=2024-12-13T11:14:45.266Z caller=node_exporter.go:118 level=info collector=uname
Dec 13 11:14:45 joston-pro node_exporter[20809]: ts=2024-12-13T11:14:45.266Z caller=node_exporter.go:118 level=info collector=vmstat
Dec 13 11:14:45 joston-pro node_exporter[20809]: ts=2024-12-13T11:14:45.266Z caller=node_exporter.go:118 level=info collector=watchdog
Dec 13 11:14:45 joston-pro node_exporter[20809]: ts=2024-12-13T11:14:45.266Z caller=node_exporter.go:118 level=info collector=xfs
Dec 13 11:14:45 joston-pro node_exporter[20809]: ts=2024-12-13T11:14:45.266Z caller=node_exporter.go:118 level=info collector=xfs
Dec 13 11:14:45 joston-pro node_exporter[20809]: ts=2024-12-13T11:14:45.267Z caller=tls_config.go:313 level=info msg="Listening on" address=[::]:9100
Dec 13 11:14:45 joston-pro node_exporter[20809]: ts=2024-12-13T11:14:45.267Z caller=tls_config.go:316 level=info msg="TLS is disabled." http2=false address=[::]:9100
joston@joston-pro:~$
```

Now create a job for node exporter to collect metrics and give ip to targets.

```
- targets: ['localhost:9090']
- job_name: 'Node_Exporter'
  scrape_interval: 5s
  static_configs:
    - targets: ['34.134.255.100:9100']
```



NOW INSTALL GRAFANA TO DIPLAY THE METRICS NODE EXPORTER HAVE COLLECTED.

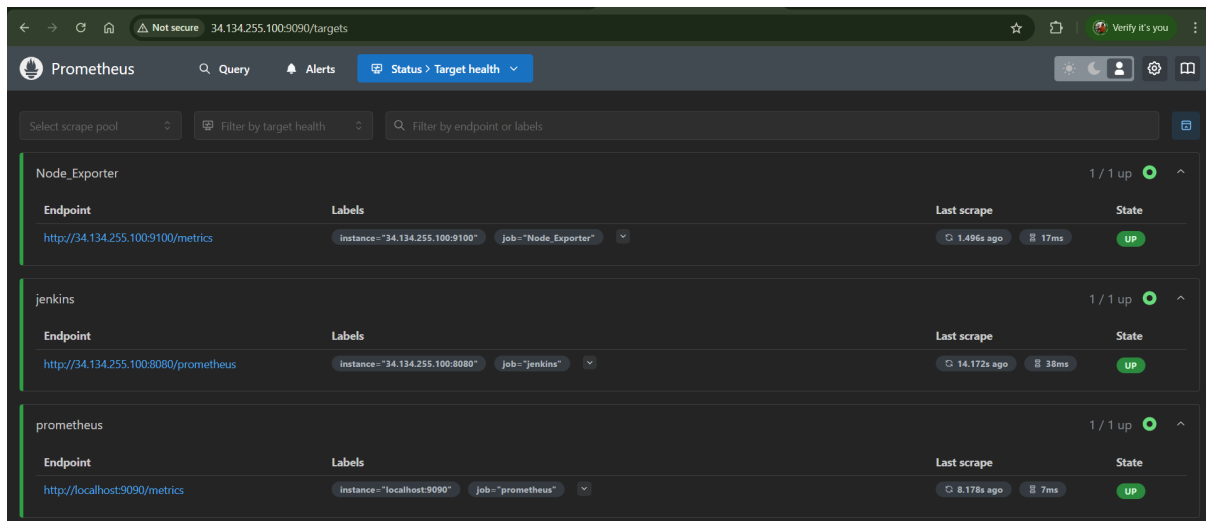
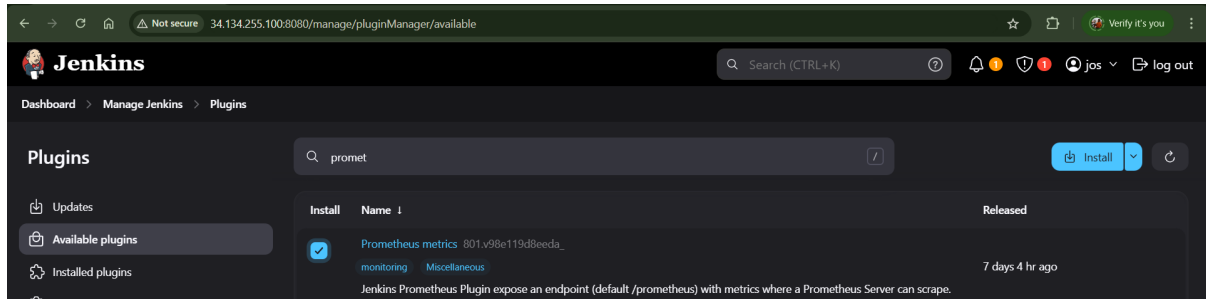




AND JENKINS JOB TO PROMETHEUS YML FILE TO COLLECT METRIC FROM JENKINS.

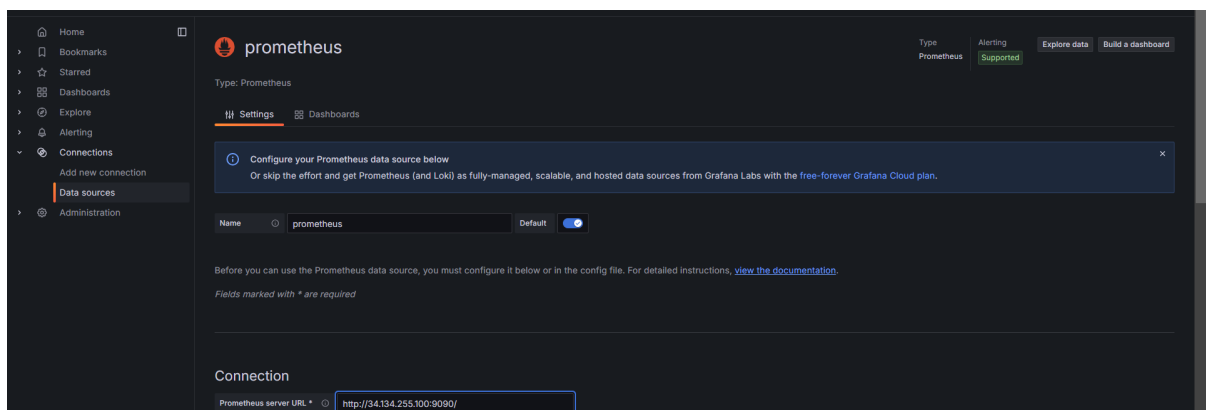
```
- targets: ['34.134.255.100:9100']
- job_name: 'jenkins'
metrics_paths: '/prometheus'
static_configs:
- targets: ['34.134.255.100:8080']
```

And install prometheus metrics plugin in jenkins dashboard.



STEP-7

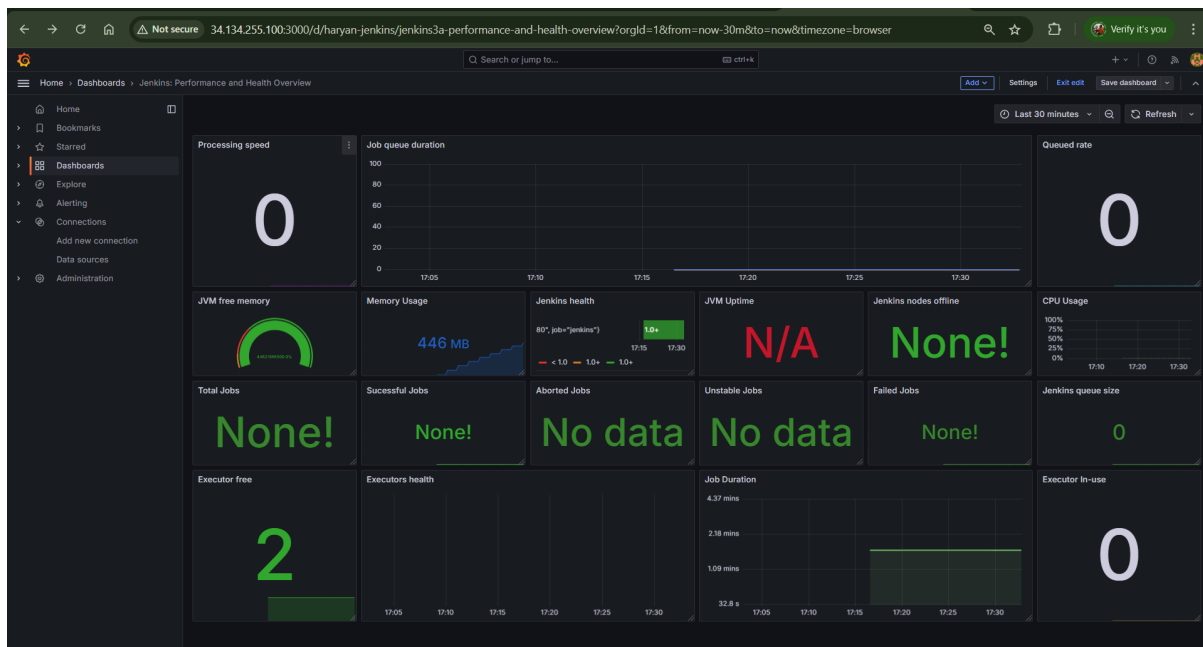
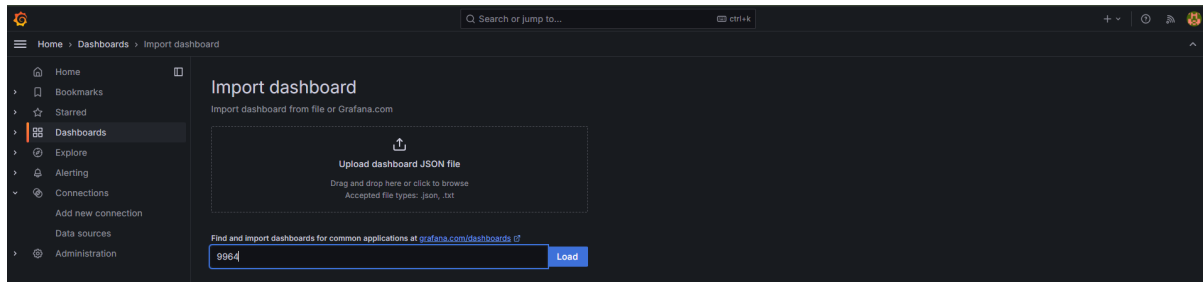
NOW NAVIGATE TO GRAFANA AND CLICK ON CONNECTION ADD THE DATASOURCE AS PROMETHEUS AND GIVE THE URL OF PROMETHEUS AND CLICK ON SAVE AND TEST.



STEP-8

NOW IMPORT THE DASHBOARD TO MONITOR JENKINS.

USING 9964 ID NUMBER.



You can see the metrics that are collected from prometheus using grafana dashboard.

See its monitoring.

1. CPU utilization

2. Disk Space Utilization

3. Total Available Memory