

Installed jdk,maven,docker

```
9 sudo systemctl enable node_exporter
10 sudo systemctl status node_exporter
11 history
12 apt install openjdk-11-jdk -y
13 apt install maven -y
14 apt install docker.io -y
15 history
root@Jenkins:~# java --version
openjdk 11.0.23 2024-04-16
OpenJDK Runtime Environment (build 11.0.23+9-post-Ubuntu-1ubuntu122.04.1)
OpenJDK 64-Bit Server VM (build 11.0.23+9-post-Ubuntu-1ubuntu122.04.1, mixed mode, sharing)
root@Jenkins:~# maven --version
Command 'maven' not found, did you mean:
  command 'aven' from deb surverx-aven (1.4.1-1)
Try: apt install <deb name>
root@Jenkins:~# mvn --version
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.23, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "6.5.0-1022-aws", arch: "amd64", family: "unix"
root@Jenkins:~# docker --version
Docker version 24.0.7, build 24.0.7-0ubuntu2-22.04.1
root@Jenkins:~# 
```

i-0716d1d48322ad151 (Jenkins)

PublicIPs: 13.233.36.148 PrivateIPs: 172.31.5.82

Terraform Installed

```
No containers need to be restarted.
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@Jenkins:~# cat tf.sh
#!/bin/bash
sudo apt-get update && sudo apt-get install -y gnupg software-properties-common
wget -O https://apt.releases.hashicorp.com/gpg | \
gpg --dearmoz | \
sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg > /dev/null
gpg --no-default-keyring \
--keyring /usr/share/keyrings/hashicorp-archive-keyring.gpg \
--fingerprint
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
sudo apt update
sudo apt-get install terraform
root@Jenkins:~# terraform --version
Terraform v1.9.2
on linux_amd64
root@Jenkins:~# 
```

i-0716d1d48322ad151 (Jenkins)

PublicIPs: 13.233.36.148 PrivateIPs: 172.31.5.82

Ansible Installed

```
No services need to be restarted.  
No containers need to be restarted.  
No user sessions are running outdated binaries.  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
root@Jenkins:~$ ansible --version  
ansible [core 2.16.8]  
  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@Jenkins:~$ cat an.sh  
#!/bin/bash  
sudo apt update  
sudo apt install software-properties-common  
sudo add-apt-repository --yes --update ppa:ansible/ansible  
sudo apt install ansible  
root@Jenkins:~$   
  
i-0716d1d48322ad151 (Jenkins)  
PublicIPs: 13.233.36.148 PrivateIPs: 172.31.5.82
```

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 4:43 PM 17/07/2024

Jenkins Installed

```
Processing triggers for man-db (2.10.2-1) ...  
Scanning processes...  
Scanning linux images...  
Running kernel seems to be up-to-date.  
No services need to be restarted.  
No containers need to be restarted.  
No user sessions are running outdated binaries.  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
root@Jenkins:~$ cat jn.sh  
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  
root@Jenkins:~$ jenkins --version  
2.452.3  
root@Jenkins:~$   
  
i-0716d1d48322ad151 (Jenkins)  
PublicIPs: 13.233.36.148 PrivateIPs: 172.31.5.82
```

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 4:45 PM 17/07/2024

Added Jenkins users to sudo access

```
GNU nano 6.2 /etc/sudoers.tmp

# Host alias specification
# User alias specification
# Cmnd alias specification
# User privilege specification
root    ALL=(ALL:ALL) ALL
jenkins ALL=(ALL) NOPASSWD:ALL
# Members of the admin group may gain root privileges
$admin  ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "@include" directives:
@includedir /etc/sudoers.d

^G Help      ^C Write Out  ^W Where Is   ^X Cut       ^T Execute   ^C Location  M-^U Undo   M-^P Set Mark  M-] To Bracket
^X Exit     ^F Read File  ^R Replace   ^U Paste     ^O Justify   ^V Go To Line M-^D Redo   M-^N Copy      ^C Where Was

i-0716d1d48322ad151 (Jenkins)
PublicIPs: 13.233.36.148 PrivateIPs: 172.31.5.82
```

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 4:47 PM 17/07/2024

Add docker to Jenkins group and check ansible working

```
No containers need to be restarted.
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@Jenkins:~# cat jn.sh
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
root@Jenkins:~# jenkins --version
2.452.3
root@Jenkins:~# visudo
root@Jenkins:~# visudo
root@Jenkins:~# sudo usermod -aG docker jenkins
root@Jenkins:~# ansible -m ping localhost
localhost | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
root@Jenkins:~#
```

i-0716d1d48322ad151 (Jenkins)
PublicIPs: 13.233.36.148 PrivateIPs: 172.31.5.82

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 4:48 PM 17/07/2024

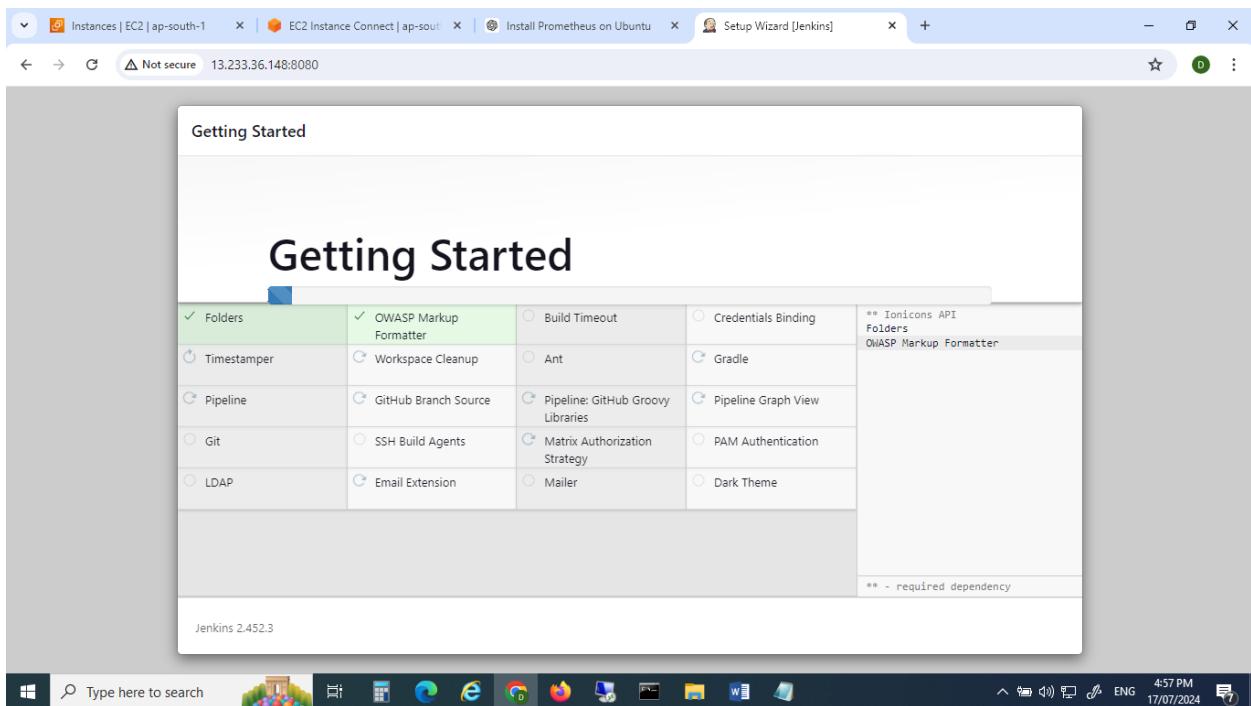
Create role with ec2 full access

The screenshot shows the AWS IAM console with a role named 'rollfort' created on July 15, 2024, at 00:03 UTC+05:30. The role has two attached policies: 'arn:aws:iam::189455135272:role/rollfort' and 'arn:aws:iam::189455135272:instance-profile/rollfort'. The 'Permissions' tab is active, showing one managed policy named 'AmazonEC2FullAccess' attached to the role.

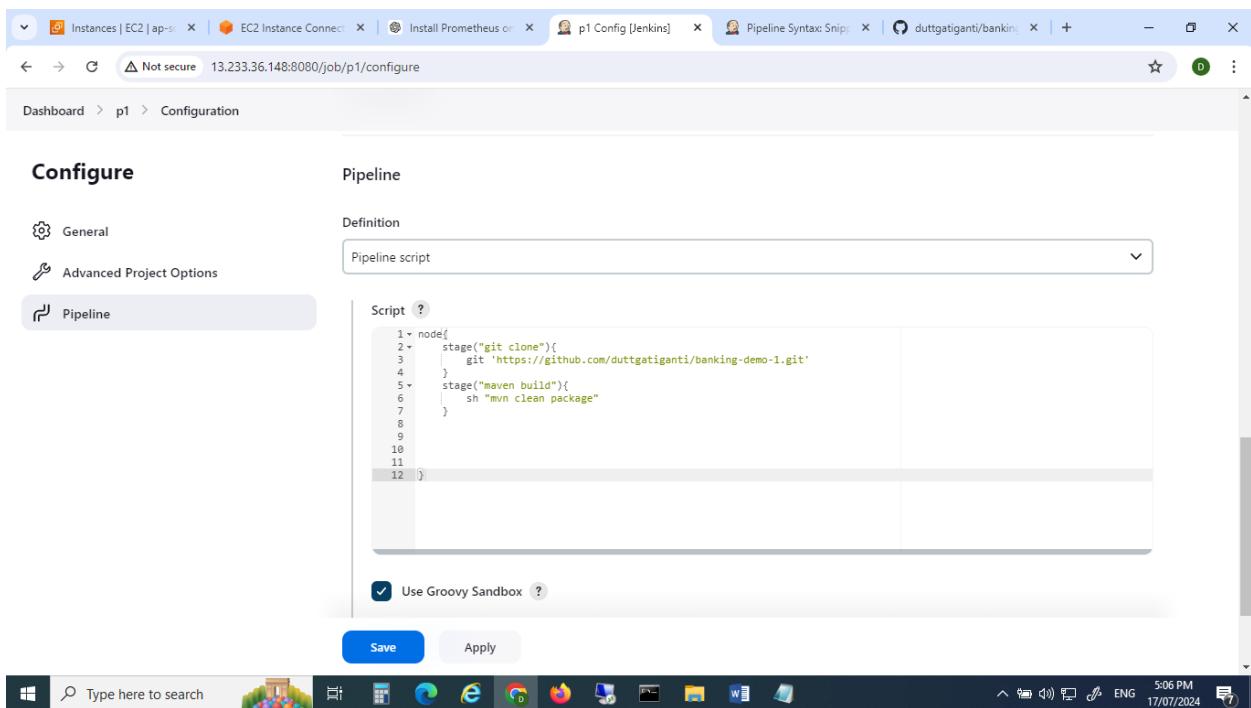
Attach role to ec2

The screenshot shows the AWS EC2 Instances page with an instance ID of 'i-0716d1d48322ad151'. A 'Modify IAM role' dialog is open, showing the 'rollfort' role selected under the 'IAM role' dropdown. The 'Update IAM role' button is highlighted in orange at the bottom right of the dialog.

Access Jenkins



Configure git clone and build



Build success

The screenshot shows the Jenkins Stage View for a pipeline named 'p1'. The 'git clone' stage took 1s and the 'maven build' stage took 22s. Build #2 was run at 17:04 on Jul 17, 2024, and build #1 was run at 17:02 on Jul 17, 2024. Both builds are listed as 'No Changes'. The 'maven build' stage for build #1 is highlighted in red with a 'failed' status. On the left, there's a sidebar with options like 'Configure', 'Delete Pipeline', 'Full Stage View', 'Stages', 'Rename', and 'Pipeline Syntax'. Below that is a 'Build History' section showing the last two builds. At the bottom, there's a Windows taskbar.

Configure Docker build

The screenshot shows the Jenkins Pipeline configuration for project 'p1'. The 'Pipeline' tab is selected. The 'Script' field contains the following Groovy code:

```
1 <node>
2   stage("git clone"){
3     git 'https://github.com/duttgatiganti/banking-demo-1.git'
4   }
5   stage("maven build"){
6     sh "mvn clean package"
7   }
8   stage("Docker Image") {
9     sh "docker build -t dutt1/primg:${BUILD_NUMBER} ."
10    sh "docker tag dutt1/primg:${BUILD_NUMBER} dutt1/primg:v1.3"
11  }
```

Below the script, there's a checkbox for 'Use Groovy Sandbox' which is checked. At the bottom, there are 'Save' and 'Apply' buttons. The Windows taskbar is visible at the bottom.

Docker build success

The screenshot shows the Jenkins Pipeline Stage View for a job named 'p1'. The stage view displays three stages: 'git clone', 'maven build', and 'Docker Image'. The 'git clone' stage took 1s, 'maven build' took 12s, and 'Docker Image' took 24s. Build history shows three builds: #3 (Jul 17, 2024, 11:40 AM), #2 (Jul 17, 2024, 11:34 AM), and #1 (Jul 17, 2024, 11:32 AM). Build #1 failed after 3s. A 'Permalinks' section provides links to the last build and the last stable build.

Stage	Time
git clone	1s
maven build	12s
Docker Image	24s

Build History:

- #3 | Jul 17, 2024, 11:40 AM
- #2 | Jul 17, 2024, 11:34 AM
- #1 | Jul 17, 2024, 11:32 AM

Permalinks:

- Last build (#3), 1 min 29 sec ago
- Last stable build (#3), 1 min 29 sec ago

Configure terraform stage and docker login

The screenshot shows the Jenkins Pipeline configuration for job 'p1 Config'. The 'Pipeline' tab is selected. The pipeline script defines a 'Terraform stage' with commands for initializing, formatting, validating, planning, and applying Terraform. It also includes a 'withCredentials' block for Docker login and push. A 'Pipeline Syntax' link is available for reference.

```
1/  withCredentials([string(credentialsId: 'dock', variable: 'dock')]) {  
2/    sh "docker login -u dutt.getigant@gmail.com -p ${dock}"  
3/    sh "docker push dutt1/primg:v1.3"  
4/  }  
5/  stage("Terraform stage") {  
6/    sh "terraform init"  
7/    sh "terraform fmt"  
8/    sh "terraform validate"  
9/    sh "terraform plan"  
10/   sh "terraform apply --auto-approve"  
11/ }  
12/ }
```

Configuration Options:

- General
- Advanced Project Options
- Pipeline (selected)

Buttons: Save, Apply

Configure ansible

The screenshot shows the Jenkins Pipeline configuration page for a project named 'p1'. The 'Pipeline' tab is selected. The pipeline script is defined as follows:

```
script {
    stage('Terraform Stage') {
        sh "terraform init"
        sh "terraform fmt"
        sh "terraform validate"
        sh "terraform plan"
        sh "terraform apply --auto-approve"
    }
    stage("Deploy to Server using ansible"){
        ansiblePlaybook become: true, credentialsId: 'ansible', disableHostKeyChecking: true, installation: 'ansible'
    }
}
```

Below the script, there is a checkbox labeled 'Use Groovy Sandbox' which is checked. At the bottom of the page are 'Save' and 'Apply' buttons.

Script for Prod server

The screenshot shows a GitHub browser window displaying the Terraform configuration file 'banking.tf' for a repository named 'banking-demo-1'. The file contains the following code:

```
resource "aws_instance" "ec" {
    ami = "ami-0c2af51e265bd5e0e"
    instance_type = "t2.micro"
    associate_public_ip_address = true
    subnet_id = "subnet-07289ec292e705287"
    vpc_security_group_ids = [aws_security_group.sg.id]
    key_name = "KP1805"

    user_data = <<-EOF
#!/bin/bash
sudo apt-get update -y
EOF

    tags = [
        Name = "Prod Server"
    ]
}
```

Terraform apply success

Screenshot of a Jenkins pipeline named 'p1' showing the Stage View. The pipeline consists of five stages: git clone, maven build, Docker Image, Docker login and push, and Terraform stage. The first stage, 'git clone', has a status of 'Success' and a duration of 1s. The second stage, 'maven build', has a duration of 12s. The third stage, 'Docker Image', has a duration of 11s. The fourth stage, 'Docker login and push', has a duration of 17s. The fifth stage, 'Terraform stage', has a duration of 58s. A tooltip indicates an average stage time of 1s and a full run time of ~1min 1s. The pipeline history shows three builds: #7 (Jul 17, 17:34, 1 commit, Success), #6 (Jul 17, 17:22, No Changes, 710ms), and #5 (Jul 17, 17:18, No Changes, 675ms). Build #5 failed after 59s.

Configure ansible credentials,hosts with ssh keys

Screenshot of an AWS CloudShell session showing Ansible configuration. The session is running on an EC2 instance connected via SSH. The command history includes:

```

## db-[99:101]-node.example.com
# ex 3: A collection of database servers in the 'dbservers' group:
## [dbservers]
## db01.intranet.mydomain.net
## db02.intranet.mydomain.net
## 10.25.1.56
## 10.25.1.57

# ex4: Multiple hosts arranged into groups such as 'Debian' and 'openSUSE':
## [Debian]
## alpha.example.org
## beta.example.org

## [openSUSE]
## green.example.com
## blue.example.com
[n]
3.109.1.187
root@Jenkins:/etc/ansible#

```

The session ends with the prompt 'root@Jenkins:/etc/ansible#'. Below the session, the AWS CloudShell interface is shown, including the AWS logo, services menu, search bar, and navigation buttons. The status bar at the bottom indicates the session is running on an EC2 instance with PublicIPs: 13.233.36.148 and PrivateIPs: 172.31.5.82.

The screenshot shows the Jenkins Pipeline Syntax configuration page for a job named 'p1'. The 'Sample Step' dropdown is set to 'ansiblePlaybook: Invoke an ansible playbook'. The configuration fields are as follows:

- Ansible tool**: ansible
- Playbook file path in workspace**: ansible-playbook.yml
- Inventory file path in workspace**: /etc/ansible/hosts
- SSH connection credentials**: ubuntu (ansible)
- Vault credentials**: None

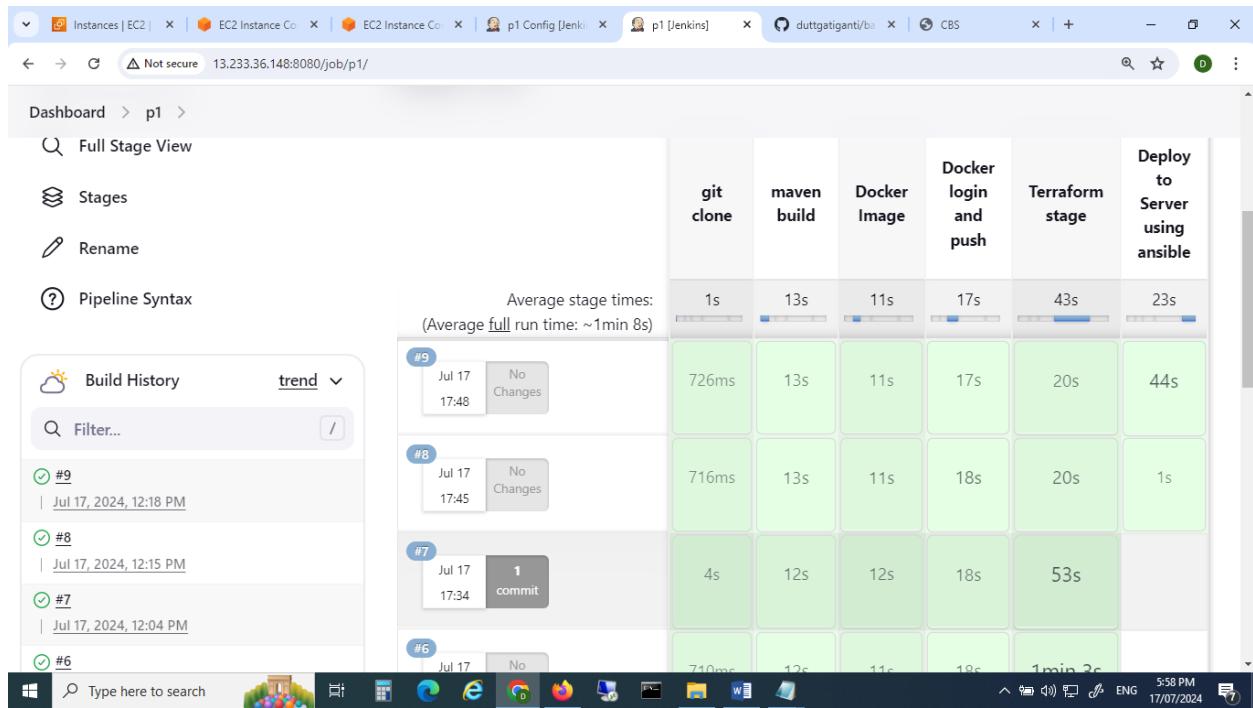
The browser address bar shows the URL: 13.233.36.148:8080/job/p1/pipeline-syntax/.

The screenshot shows the Jenkins Configuration page for a job named 'p1'. The left sidebar shows 'Configure' under 'Pipeline'. The main area displays a Groovy script for the pipeline:

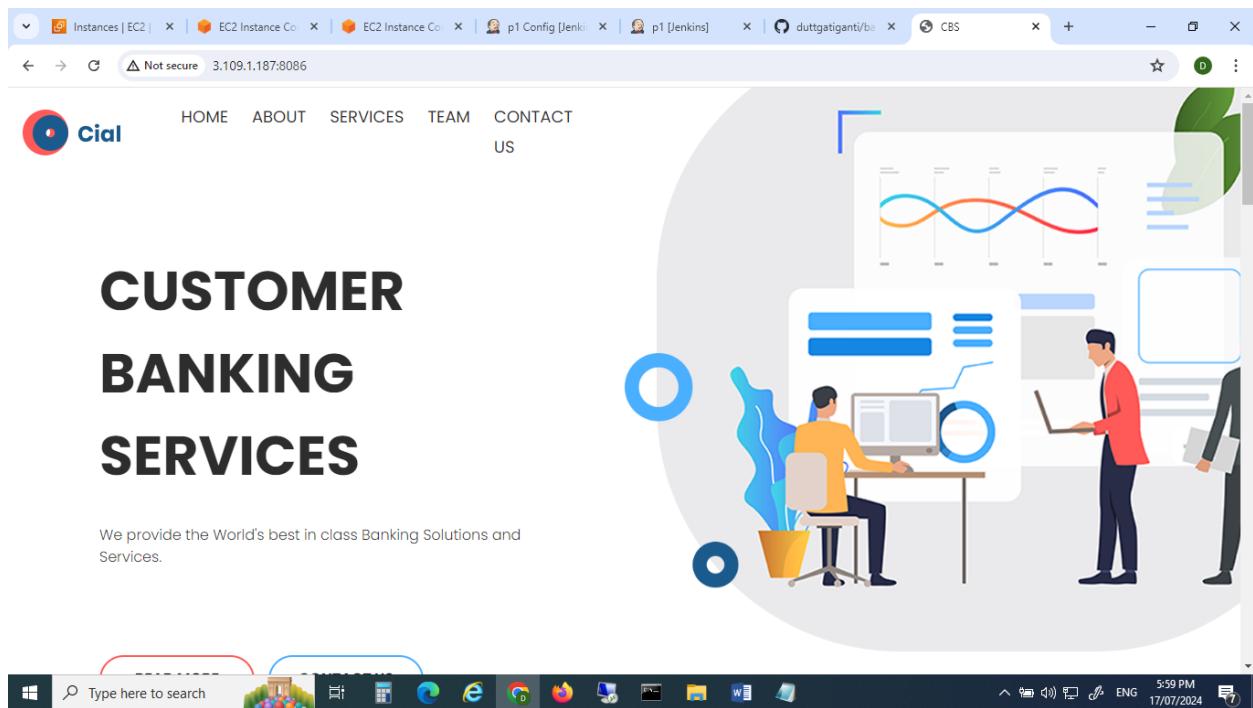
```
21 }
22 -> stage("Terraform stage") {
23
24     sh "terraform init"
25     sh "terraform fmt"
26     sh "terraform validate"
27     sh "terraform plan"
28     sh "terraform apply --auto-approve"
29
30
31
32 }
33 -> stage("Deploy to Server using ansible"){
34     ansiblePlaybook become: true, credentialsId: 'ansible', disableHostKeyChecking: true
35 }
36
37 }
```

A checkbox labeled 'Use Groovy Sandbox' is checked. Below the script, there are 'Save' and 'Apply' buttons. The browser address bar shows the URL: 13.233.36.148:8080/job/p1/configure.

Ansible stage success



Successfully deployed



Github-webhook configuration

A screenshot of a web browser window showing the 'Webhooks / Manage webhook' configuration page for the 'duttgatiganti / banking-demo-1' repository on GitHub. The URL in the address bar is 'github.com/duttgatiganti/banking-demo-1/settings/hooks/490637422'. The page includes a sidebar with options like General, Access, Collaborators, Moderation options, Code and automation, Branches, Tags, Rules, Actions, Webhooks (which is selected), Environments, Codespaces, and Pages. The main content area shows a form for setting up a webhook, with fields for 'Payload URL' (set to 'http://13.233.36.148:8080/github-webhook/'), 'Content type' (set to 'application/x-www-form-urlencoded'), and 'Secret'. A note at the top explains that a POST request will be sent to the URL with event details.

The screenshot shows the GitHub settings page for the repository 'duttgatiganti / banking-demo-1'. The 'Webhooks' tab is selected. A single webhook is listed with the URL <http://13.233.36.148:8080/github-w...> and the event type '(push)'. The status indicates 'Last delivery was successful.' There are 'Edit' and 'Delete' buttons next to the webhook entry.

Sir if I configure web-hook at initial Jenkins is triggering for every troubleshooting change.

So I configured web hook at last

The screenshot shows the Jenkins job configuration page for 'p1'. Under the 'General' section of the 'Configure' tab, the 'Build Triggers' section is visible. The 'GitHub hook trigger for GITScm polling' checkbox is checked. Other options like 'Build after other projects are built', 'Build periodically', 'Poll SCM', 'Quiet period', and 'Trigger builds remotely' are available but not selected. At the bottom of the configuration page, there are 'Save' and 'Apply' buttons.

Changes pushed

```

Instances | EC2 | EC2 Instance Co... | EC2 Instance Co... | p1 [Jenkins] | p1 [Jenkins] | banking-demo-1 | CBS
C:\Windows\System32\cmd.exe
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
      modified:   src/main/resources/static/index.html

no changes added to commit (use "git add" and/or "git commit -a")

D:\star\proj\Src\banking-demo-1>git add .

D:\star\proj\Src\banking-demo-1>git commit -m "sc"
[master bbaa866] sc
 1 file changed, 1 insertion(+), 1 deletion(-)

D:\star\proj\Src\banking-demo-1>git push
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 630 bytes | 630.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/duttgatiganti/banking-demo-1.git
  fea67e0..bbaa866 master -> master

D:\star\proj\Src\banking-demo-1>
```

#10 Jul 17, 2024, 12:43 PM Changes 17:48 726ms 13s 11s 17s 20s 44s

#8 Jul 17, 17:45 No Changes 716ms 13s 11s 18s 20s 1s

#9 Jul 17, 12:18 PM No Changes 716ms 13s 11s 18s 20s 1s

Type here to search 6:13 PM 17/07/2024

Triggered automatically

Dashboard > p1 > Full Stage View

Stages

Rename

Pipeline Syntax

GitHub Hook Log

Build History trend

Filter...

#10 Jul 17 18:13 2 commits 972ms 12s

#9 Jul 17 17:48 No Changes 726ms 13s 11s 17s 20s 44s

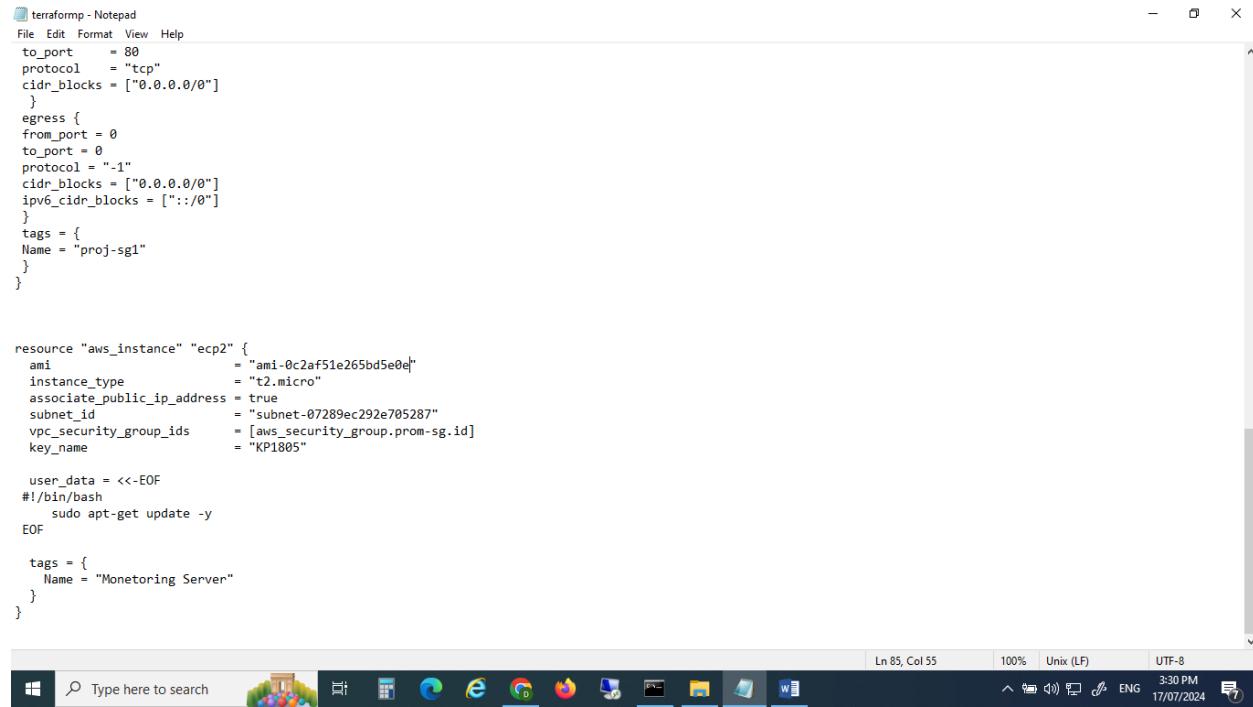
#8 Jul 17 17:45 No Changes 716ms 13s 11s 18s 20s 1s

Average stage times:
(Average full run time: ~1min 8s)

git clone maven build Docker Image Docker login and push Terraform stage Deploy to Server using ansible

Type here to search 6:13 PM 17/07/2024

Created Monetoring server

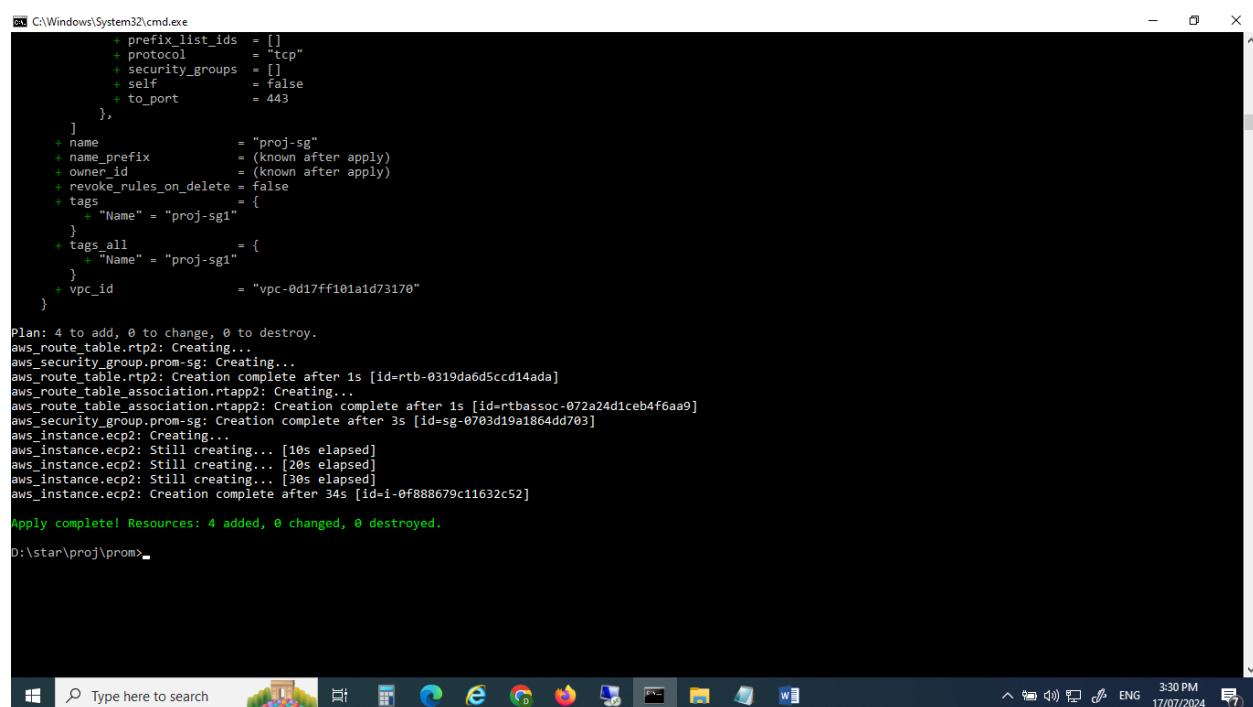


```
terraform - Notepad
File Edit Format View Help
to_port      = 80
protocol     = "tcp"
cidr_blocks = ["0.0.0.0/0"]
}
egress {
from_port   = 0
to_port     = 0
protocol   = "-1"
cidr_blocks = ["0.0.0.0/0"]
ipv6_cidr_blocks = [":/:0"]
}
tags = {
Name = "proj-sg1"
}
}

resource "aws_instance" "ecp2" {
ami           = "ami-0c2af51e265bd5e0e"
instance_type = "t2.micro"
associate_public_ip_address = true
subnet_id     = "subnet-07289ec292e705287"
vpc_security_group_ids = [aws_security_group.prom-sg.id]
key_name      = "KP1805"

user_data = <<-EOF
#!/bin/bash
sudo apt-get update -y
EOF

tags = {
Name = "Monetoring Server"
}
}
```



```
C:\Windows\System32\cmd.exe
+ prefix_list_ids  = []
+ protocol        = "tcp"
+ security_groups = []
+ self             = false
+ to_port          = 443
},
]
+
name           = "proj-sg"
name_prefix    = "(known after apply)"
owner_id       = "(known after apply)"
revoke_rules_on_delete = false
tags           = [
+ "Name" = "proj-sg1"
]
tags_all       = {
+ "Name" = "proj-sg1"
}
vpc_id         = "vpc-0d17ff101a1d73170"
}

Plan: 4 to add, 0 to change, 0 to destroy.
aws_route_table.rtb2: Creating...
aws_security_group.prom-sg: Creating...
aws_route_table_association.rtapp2: Creating...
aws_route_table_association.rtapp2: Creation complete after 1s [id=rtb-0319da6d5cccd14ada]
aws_route_table_association.rtapp2: Creation complete after 1s [id=rtbassoc-072a24diceb4f6aa9]
aws_security_group.prom-sg: Creation complete after 3s [id=sg-0703d19a1864dd703]
aws_instance.ecp2: Creating...
aws_instance.ecp2: Still creating... [20s elapsed]
aws_instance.ecp2: Still creating... [20s elapsed]
aws_instance.ecp2: Still creating... [20s elapsed]
aws_instance.ecp2: Creation complete after 34s [id=i-0f888679c11632c52]

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

D:\star\proj>
```

Commands executed for installing Prometheus

A screenshot of a Windows desktop environment. At the top, there's a taskbar with icons for File Explorer, Task View, Start, and others. Below the taskbar, a browser window is open to the EC2 Instance Connect URL. The main area shows a terminal session titled 'AWS Services' with the title bar 'Instances | EC2 | ap-south-1'. The terminal window displays a series of commands being run on an Ubuntu server via SSH. The commands include navigating to /etc/prometheus, editing prometheus.yml, restarting the Prometheus service, installing Grafana, and enabling it. The session ends with a history command. Below the terminal, a message box shows the instance ID (i-07a1e9e88cbad09b), public IP (13.233.45.195), and private IP (172.31.8.200). The bottom of the screen features the Windows Start button and a taskbar with various pinned applications.

```
29 sudo systemctl status prometheus
30 cd /etc/prometheus
31 ls
32 vi prometheus.yml
33 cd
34 sudo systemctl restart prometheus
35 sudo systemctl status prometheus
36 sudo apt update
37 sudo apt install -y software-properties-common
38 sudo add-apt-repository "deb https://packages.grafana.com/oss/deb stable main"
39 wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
40 sudo apt update
41 sudo apt install -y grafana
42 sudo systemctl start grafana-server
43 sudo systemctl enable grafana-server
44 sudo systemctl status grafana-server
45 history
46 cd
47 apt update -y
48 hostnamectl set-hostname Mon
49 bash
50 cd
51 history
root@Mon:~#
```

i-07a1e9e88cbad09b (Mon Server)
PublicIPs: 13.233.45.195 PrivateIPs: 172.31.8.200

Prometheus service configured

A screenshot of a Windows desktop environment, similar to the previous one. A browser window shows the EC2 Instance Connect URL. The main area is a terminal session titled 'AWS Services' with the title bar 'Instances | EC2 | ap-south-1'. The terminal window displays commands to verify the Prometheus service configuration. It includes navigating to the Prometheus directory, listing files, and viewing the contents of prometheus.service. The session ends with a history command. Below the terminal, a message box shows the instance ID (i-07a1e9e88cbad09b), public IP (13.233.45.195), and private IP (172.31.8.200). The bottom of the screen features the Windows Start button and a taskbar with various pinned applications.

```
root@Mon:~# cd prometheus-2.46.0.linux-amd64
root@Mon:~/prometheus-2.46.0.linux-amd64# ls
LICENSE NOTICE
root@Mon:~/prometheus-2.46.0.linux-amd64# cd
root@Mon:~# sudo cat /etc/systemd/system/prometheus.service
[Unit]
Description=Prometheus
Wants=network-online.target
After=network-online.target

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/usr/local/bin/prometheus \
--config.file /etc/prometheus/prometheus.yml \
--storage.tsdb.path /var/lib/prometheus/ \
--web.console.templates=/etc/prometheus/consoles \
--web.console.libraries=/etc/prometheus/console_libraries

[Install]
WantedBy=multi-user.target
root@Mon:~#
```

i-07a1e9e88cbad09b (Mon Server)
PublicIPs: 13.233.45.195 PrivateIPs: 172.31.8.200

Prometheus status

```

Last login: Wed Jul 17 15:16:15 2024 from 13.233.177.3
ubuntu@Mon-:~$ sudo su
root@Mon:/home/ubuntu# cd
root@Mon-:~# sudo systemctl status prometheus
● prometheus.service - Prometheus
   Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2024-07-17 15:39:51 UTC; 12min ago
       Main PID: 387 (prometheus)
          Tasks: 8 (limit: 4666)
            Memory: 84.6M
              CPU: 676ms
            CGroup: /system.slice/prometheus.service
                └─ 387 /usr/local/bin/prometheus --config.file /etc/prometheus/prometheus.yml --storage.tsdb.path /var/lib/prometheus/ --web.console.templates=/etc/p

Jul 17 15:39:53 Mon prometheus[387]: ts=2024-07-17T15:39:53.622Z caller=head.go:684 level=info component=tsdb msg="Replaying WAL, this may take a while"
Jul 17 15:39:53 Mon prometheus[387]: ts=2024-07-17T15:39:53.678Z caller=head.go:755 level=info component=tsdb msg="WAL segment loaded" segment=0 maxSegment=1
Jul 17 15:39:53 Mon prometheus[387]: ts=2024-07-17T15:39:53.679Z caller=head.go:755 level=info component=tsdb msg="WAL segment loaded" segment=1 maxSegment=1
Jul 17 15:39:53 Mon prometheus[387]: ts=2024-07-17T15:39:53.679Z caller=head.go:792 level=info component=tsdb msg="WAL replay completed" checkpoint_replay_duration=0.000000s
Jul 17 15:39:53 Mon prometheus[387]: ts=2024-07-17T15:39:53.680Z caller=main.go:1047 level=info fs type=EXT4 SUPER MAGIC
Jul 17 15:39:53 Mon prometheus[387]: ts=2024-07-17T15:39:53.682Z caller=main.go:1050 level=info msg="TSDB started"
Jul 17 15:39:53 Mon prometheus[387]: ts=2024-07-17T15:39:53.684Z caller=main.go:1231 level=info msg="Loading configuration file" filename=/etc/prometheus/prometheus.yml
Jul 17 15:39:53 Mon prometheus[387]: ts=2024-07-17T15:39:53.689Z caller=main.go:1268 level=info msg="Completed loading of configuration file" filename=/etc/prometheus/prometheus.yml
Jul 17 15:39:53 Mon prometheus[387]: ts=2024-07-17T15:39:53.690Z caller=main.go:1011 level=info msg="Server is ready to receive web requests."
Jul 17 15:39:53 Mon prometheus[387]: ts=2024-07-17T15:39:53.690Z caller=manager.go:1000 level=info component="rule manager" msg="Starting rule manager..."
lines 1-20/20 (END)
[1]+  Stopped                  sudo systemctl status prometheus
root@Mon:~# 
```

i-07a1e9e88cbadf09b (Mon Server)
PublicIPs: 65.0.177.56 PrivateIPs: 172.31.8.200

Hosts added in monitoring server

```

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: ["172.31.5.82:9100"]
      - targets: ["172.31.10.123:9100"]
root@Mon:/etc/prometheus# 
```

i-07a1e9e88cbadf09b (Mon Server)
PublicIPs: 13.233.45.195 PrivateIPs: 172.31.8.200

Grafana configured

The screenshot shows a CloudShell window with several tabs open. The active tab displays terminal output from an EC2 instance. The command `vi prometheus.yml` is run, followed by `sudo systemctl status grafana-server`. The output shows the service is active and running. Subsequent log entries show various system and application events, including Prometheus update checks and Grafana usage statistics.

```
root@Mon:/etc/prometheus# vi prometheus.yml
root@Mon:/etc/prometheus# sudo systemctl status grafana-server
● grafana-server.service - Grafana instance
   Loaded: loaded (/lib/systemd/system/grafana-server.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2024-07-17 12:53:41 UTC; 11min ago
       Docs: http://docs.grafana.org
      Main PID: 366 (grafana)
        Tasks: 14 (limit: 4666)
       Memory: 176.5M
          CPU: 2.305s
         CGroup: /system.slice/grafana-server.service
                 └─366 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/grafana/grafana-server.pid --packaging

Jul 17 12:53:46 Mon grafana[366]: logger=grafana.update.checker t=2024-07-17T12:53:46.338588202Z level=info msg="Update check succeeded" duration=1.000ms
Jul 17 12:53:46 Mon grafana[366]: logger=grafana-apiserver t=2024-07-17T12:53:46.458216193Z level=info msg="Adding GroupVersion playlist.grafana"
Jul 17 12:53:46 Mon grafana[366]: logger=grafana-apiserver t=2024-07-17T12:53:46.458980662Z level=info msg="Adding GroupVersion featuretoggle.grafana"
Jul 17 12:54:31 Mon grafana[366]: logger=context userId=1 orgId=1 uname=admin t=2024-07-17T12:54:31.249288803Z level=info msg="Request Completed"
Jul 17 12:55:02 Mon grafana[366]: logger=infra.usagestats t=2024-07-17T12:55:02.919368941Z level=info msg="Usage stats are ready to report"
Jul 17 12:56:41 Mon grafana[366]: logger=live t=2024-07-17T12:56:41.111071874Z level=info msg="Initialized channel handler" channel=grafana/dash
Jul 17 12:59:48 Mon grafana[366]: logger=context userId=1 orgId=1 uname=admin t=2024-07-17T12:59:48.170386891Z level=info msg="Request Completed"
Jul 17 13:01:53 Mon grafana[366]: logger=context userId=1 orgId=1 uname=admin t=2024-07-17T13:01:53.189949751Z level=info msg="Request Completed"
Jul 17 13:03:45 Mon grafana[366]: logger=cleanup t=2024-07-17T13:03:45.940153767Z level=info msg="Completed cleanup jobs" duration=7.338996ms
Jul 17 13:03:46 Mon grafana[366]: logger=plugins.update.checker t=2024-07-17T13:03:46.728617444Z level=info msg="Update check succeeded" duration=0.000ms
lines 1-21/21 (END)
```

i-07a1e9e88cbad09b (Mon Server)
PublicIPs: 13.233.45.195 PrivateIPs: 172.31.8.200

CloudShell Feedback Type here to search © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 6:35 PM 17/07/2024

Added Data source to grafana

The screenshot shows the 'Data sources' configuration page in Grafana. A new data source named 'prometheus' is being added. The 'Connection' section specifies the Prometheus server URL as 'http://localhost:9090'. The 'Authentication' section indicates no authentication methods are chosen. The browser's address bar shows the URL '13.233.45.195:3000/connections/datasources/edit/bds0opug9pbld'.

Not secure 13.233.45.195:3000/connections/datasources/edit/bds0opug9pbld

Configure your Prometheus data source below
Or skip the effort and get Prometheus (and Loki) as fully-managed, scalable, and hosted data sources from Grafana Labs with the free-forever Grafana Cloud plan.

Name: prometheus Default:

Before you can use the Prometheus data source, you must configure it below or in the config file. For detailed instructions, [view the documentation](#).

Fields marked with * are required

Connection

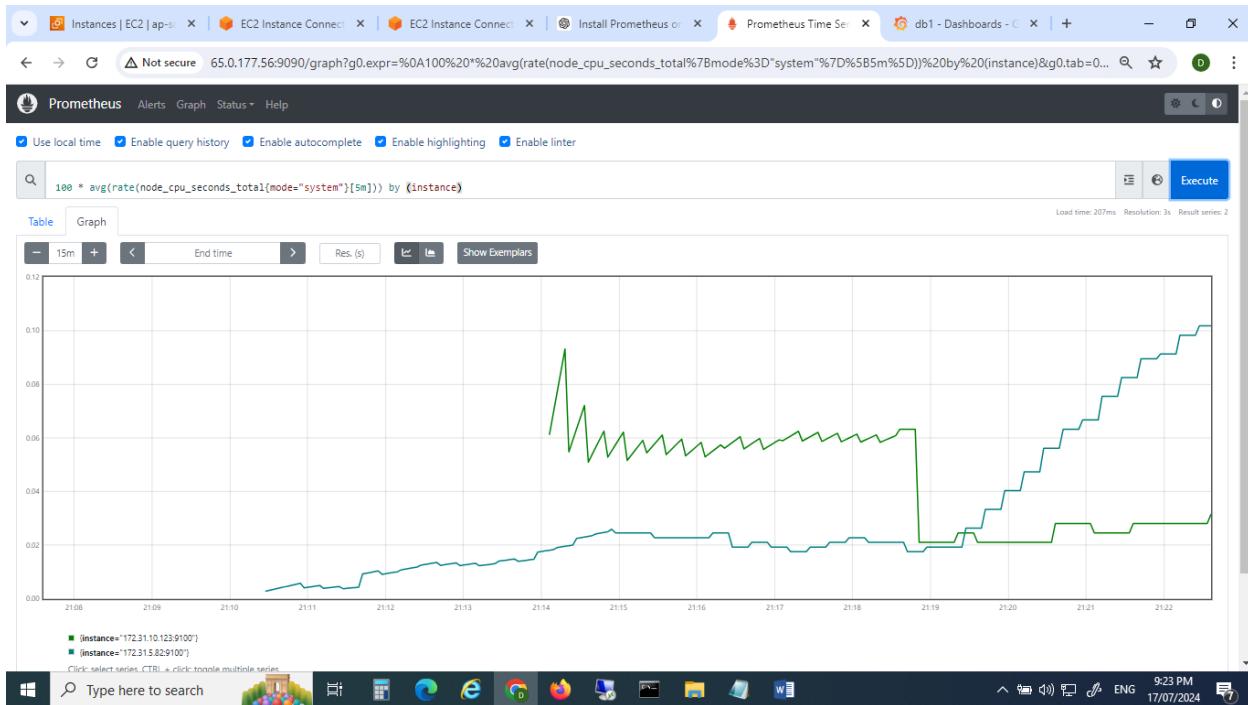
Prometheus server URL *

Authentication

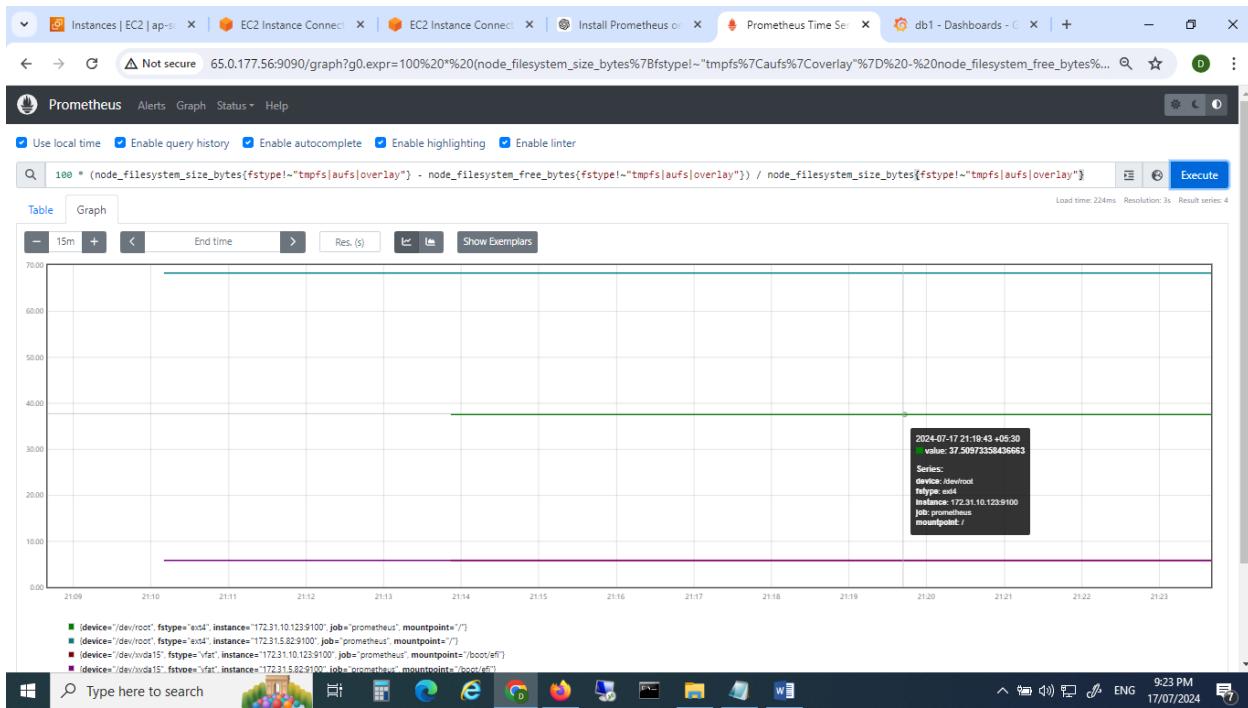
Authentication methods
Choose an authentication method to access the data source

Type here to search © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 6:36 PM 17/07/2024

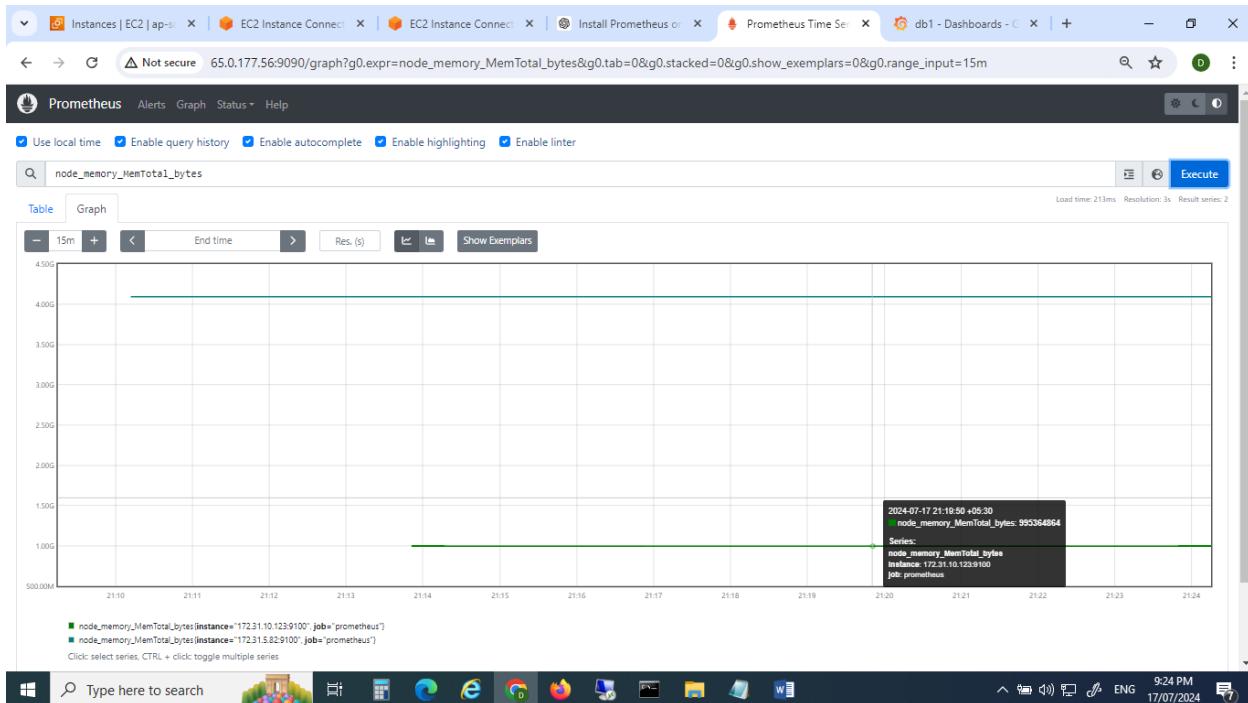
CPU utilization from Prometheus



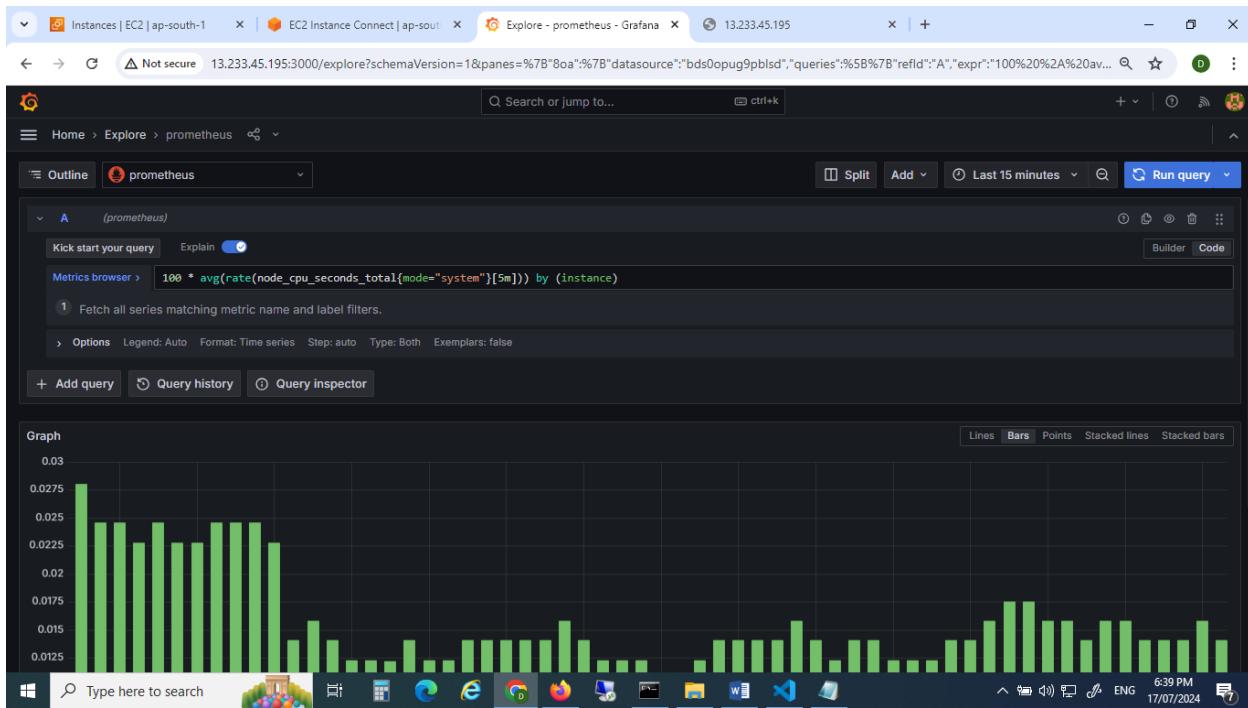
Disk Space Utilization



Total Available Memory



View of data in grafana



Added Dash board to grafana

