Finance Me

Banking and Finance Domain

Finance me is a project which is used for banking services. It uses java code for developing application. The application is automated using CI/CD pipeline. The flow of the project is explained in detail.

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.

Created an EC2 instance with name Master-server and configured java, maven, docker, Jenkins, Ansible and Terraform on the instance, this server acts as master server, which is used to implement CI/CD pipeline script.

These are details of master-server

Public IP: *13.233.102.84*

IAM Role: srija with EC2 Full Access.

Java Version: jdk-17

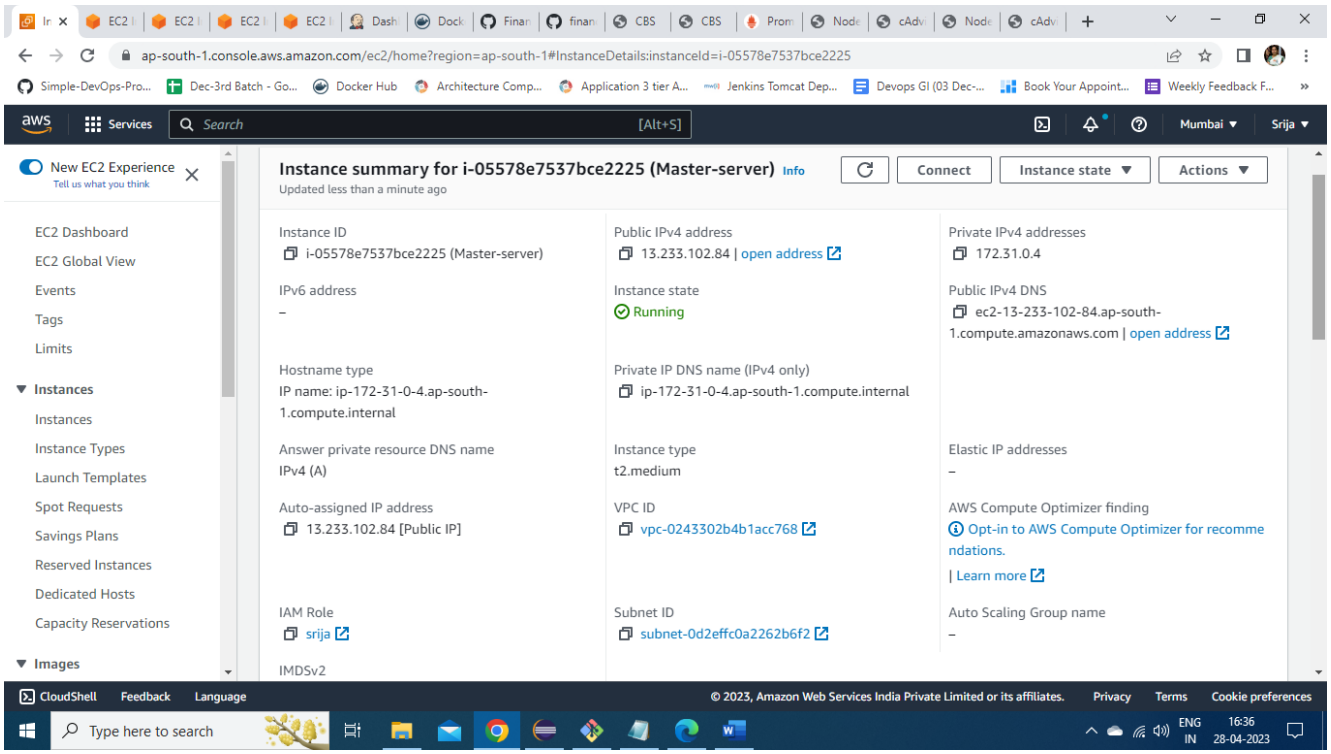
Maven version: maven 3.6.3.

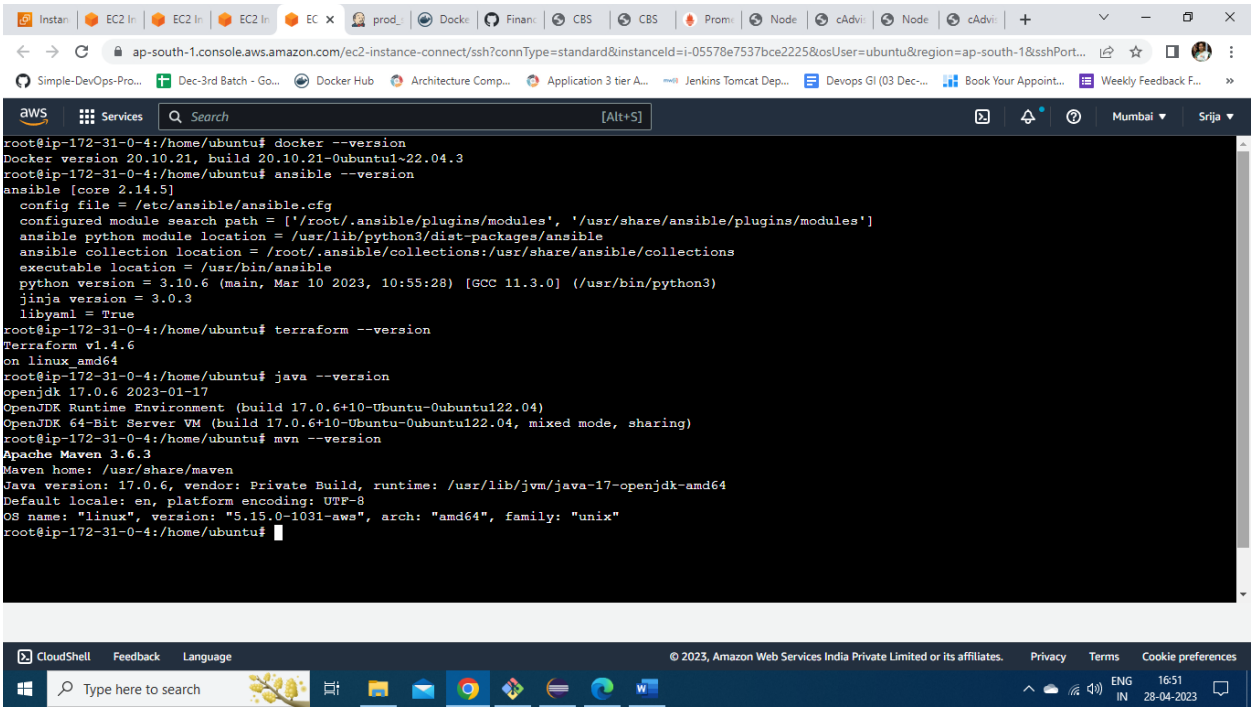
Ansible Version: 2.14.5

Terraform Version: v1.4.6

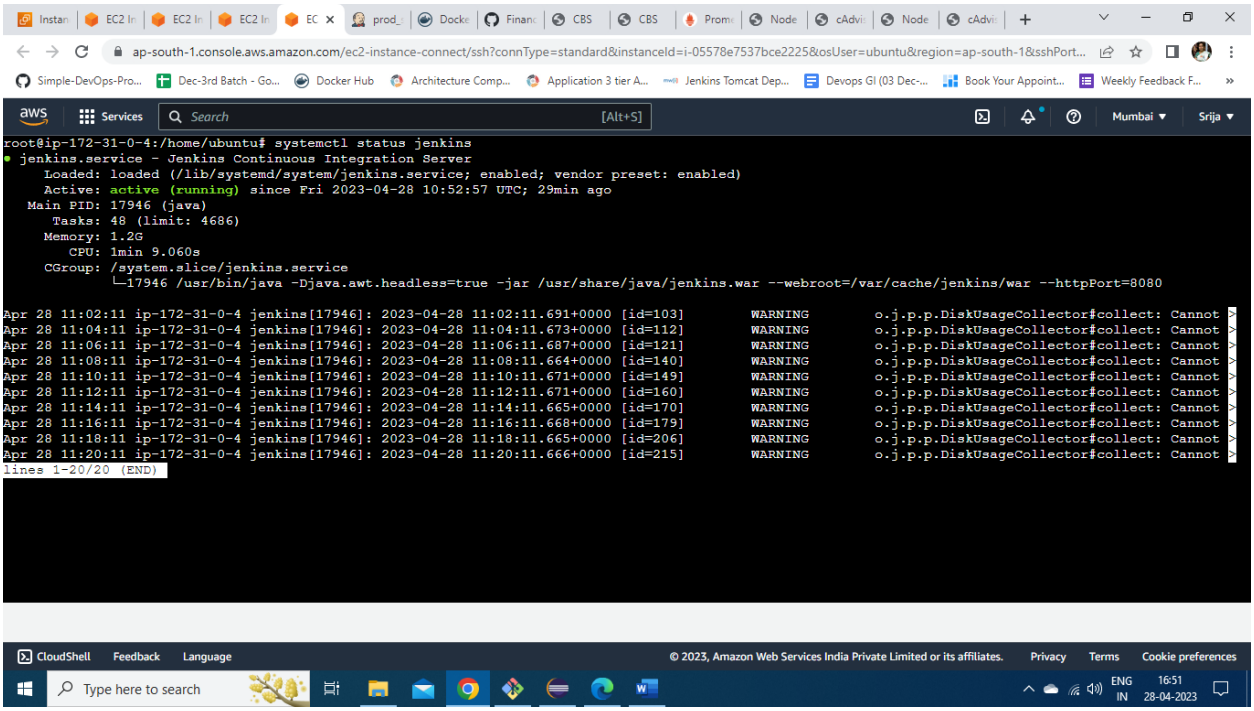
Docker Version: 20.10.21

Jenkins configured on URL: *13.233.102.84:8080*



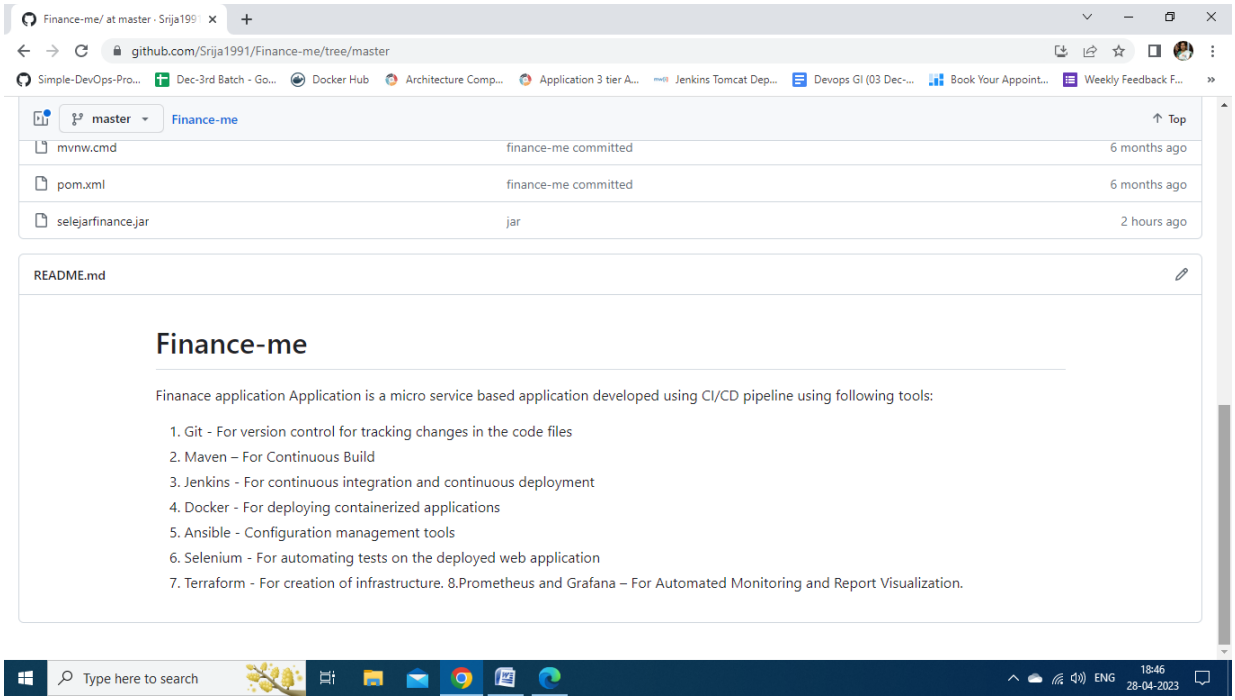


Here are the screenshots of master-server connected on browser displaying all the versions.



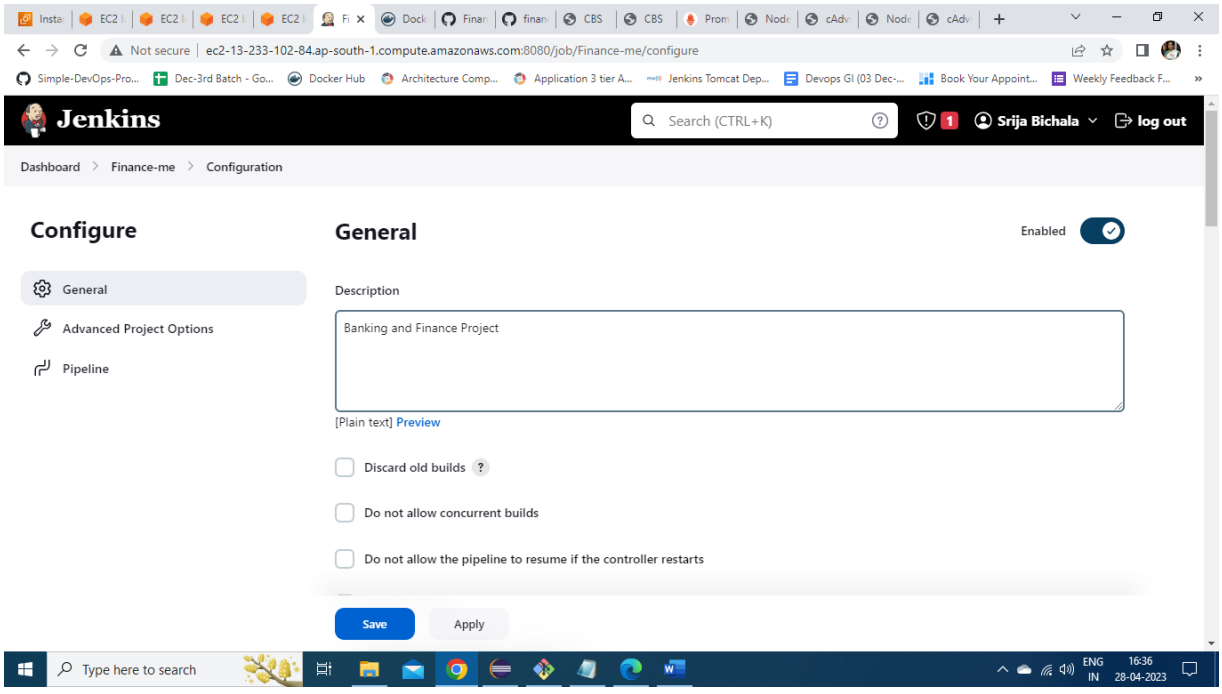
The code is pushed onto Github and the repository URL is:

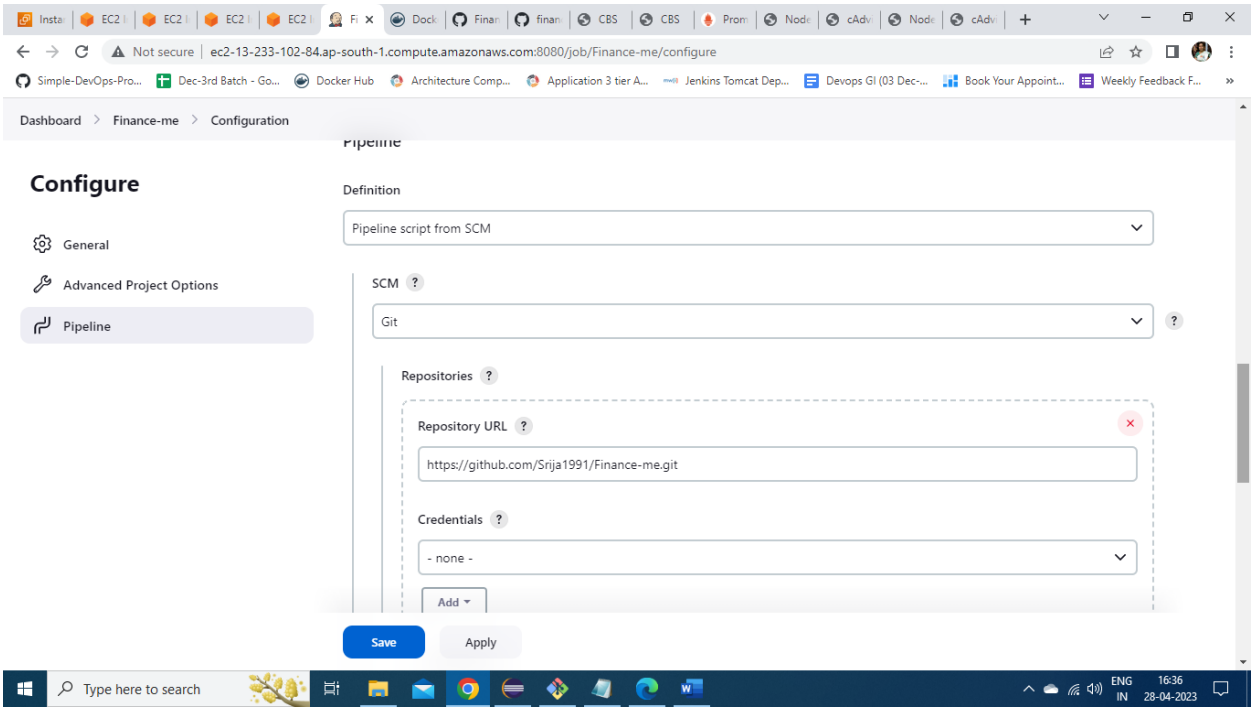
https://github.com/Srija1991/Finance-me.git



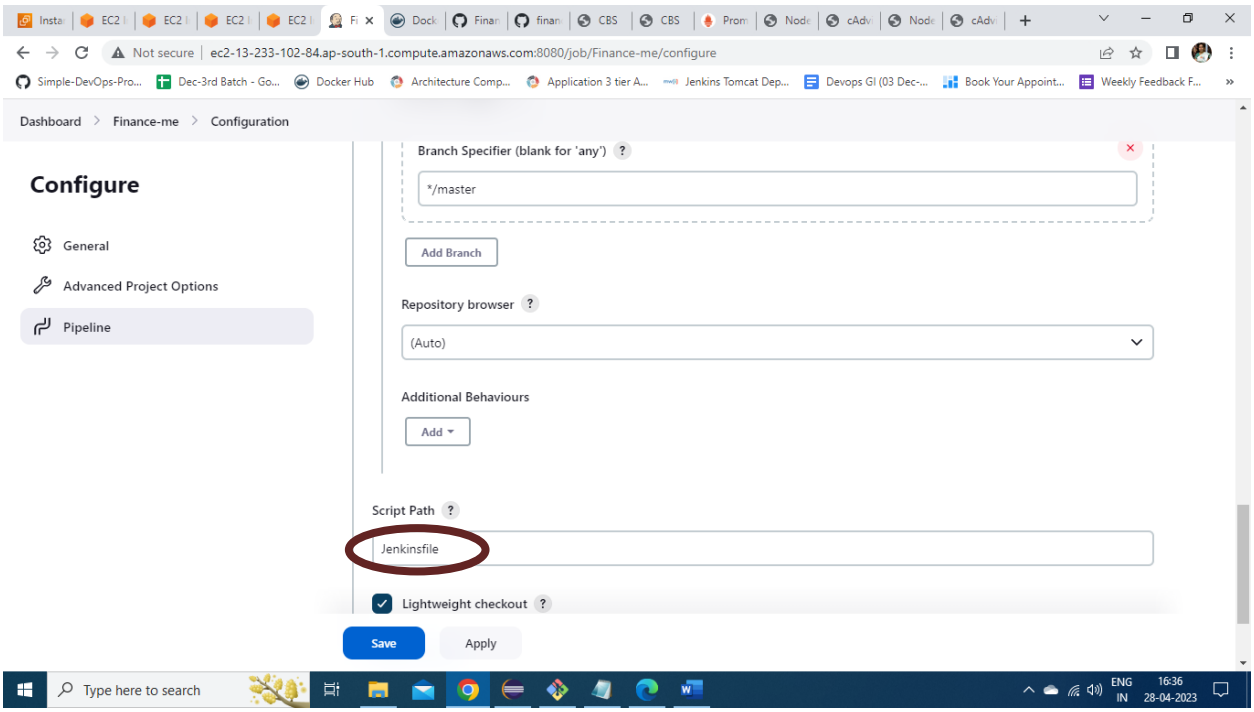
The new job named Finance-me is created and added pipelineSCM for pipeline definition.

The SCM used here is Git and URL is given as shown.

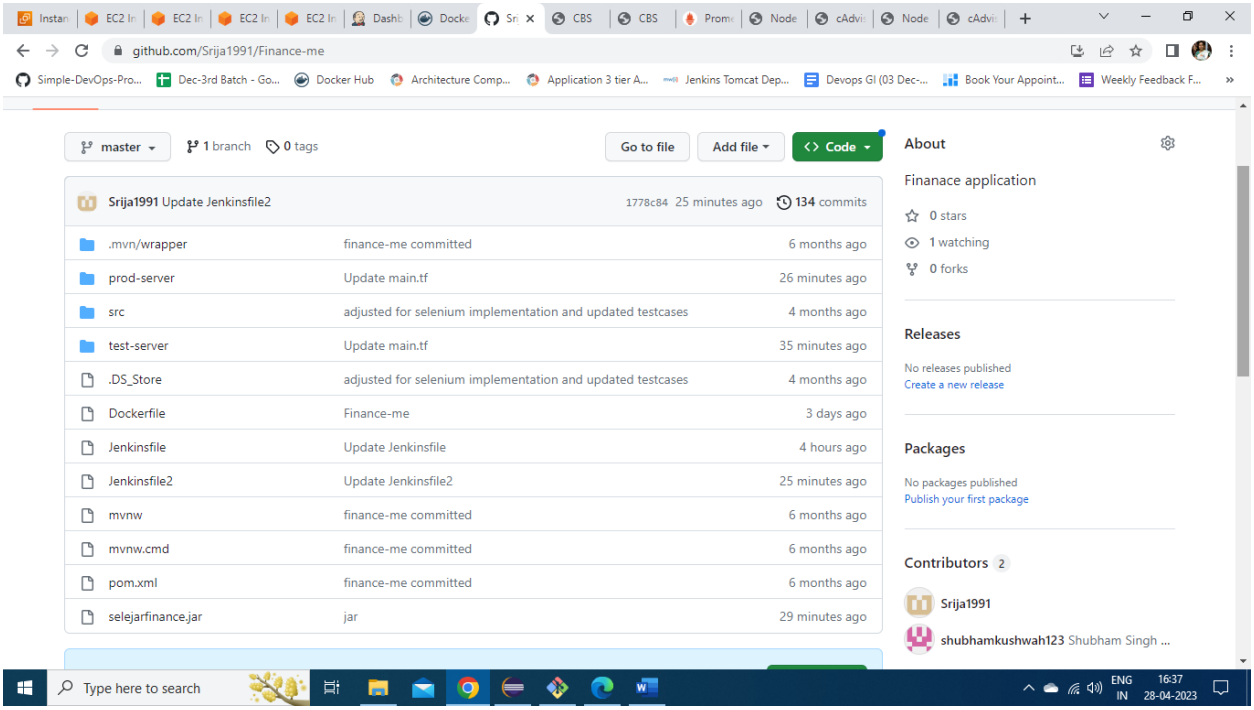




The pipeline script is provided through file named *Jenkinsfile*.

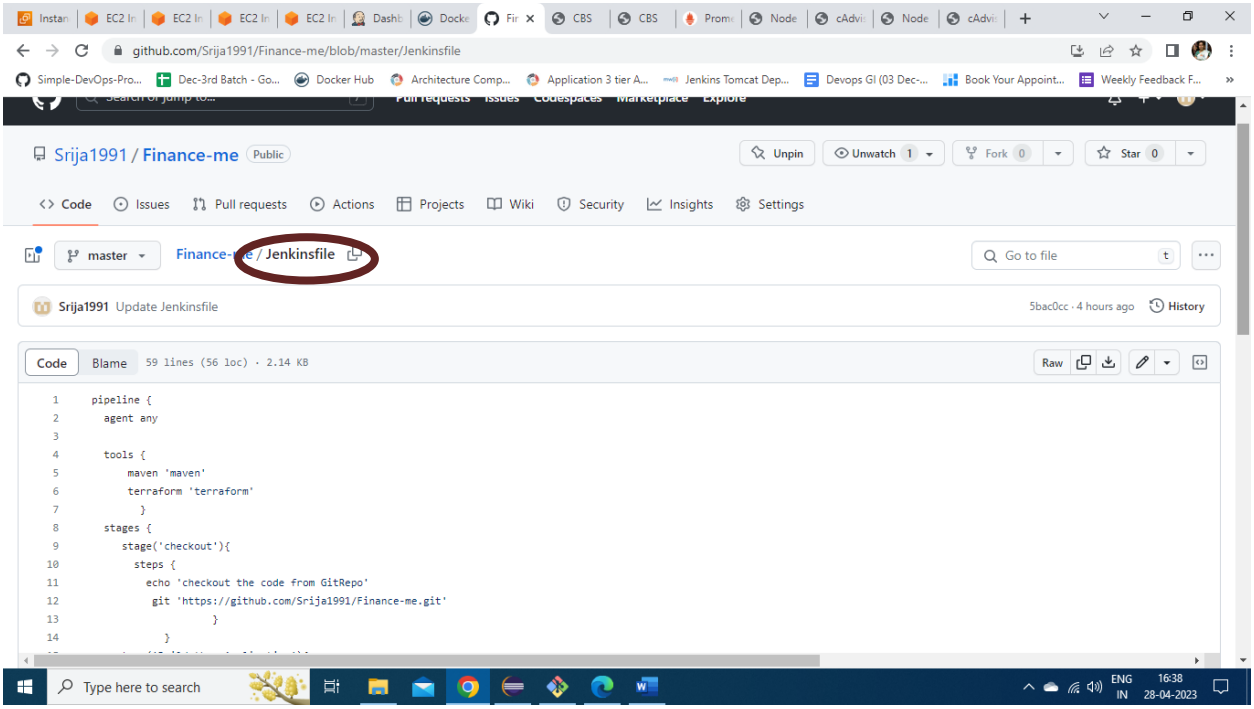


The repository is shown here with Dockerfile, Jenkinsfile, Jenkinsfile2 , folders test-server and prod-server for creating and deploying application onto test-server and production server respectively.

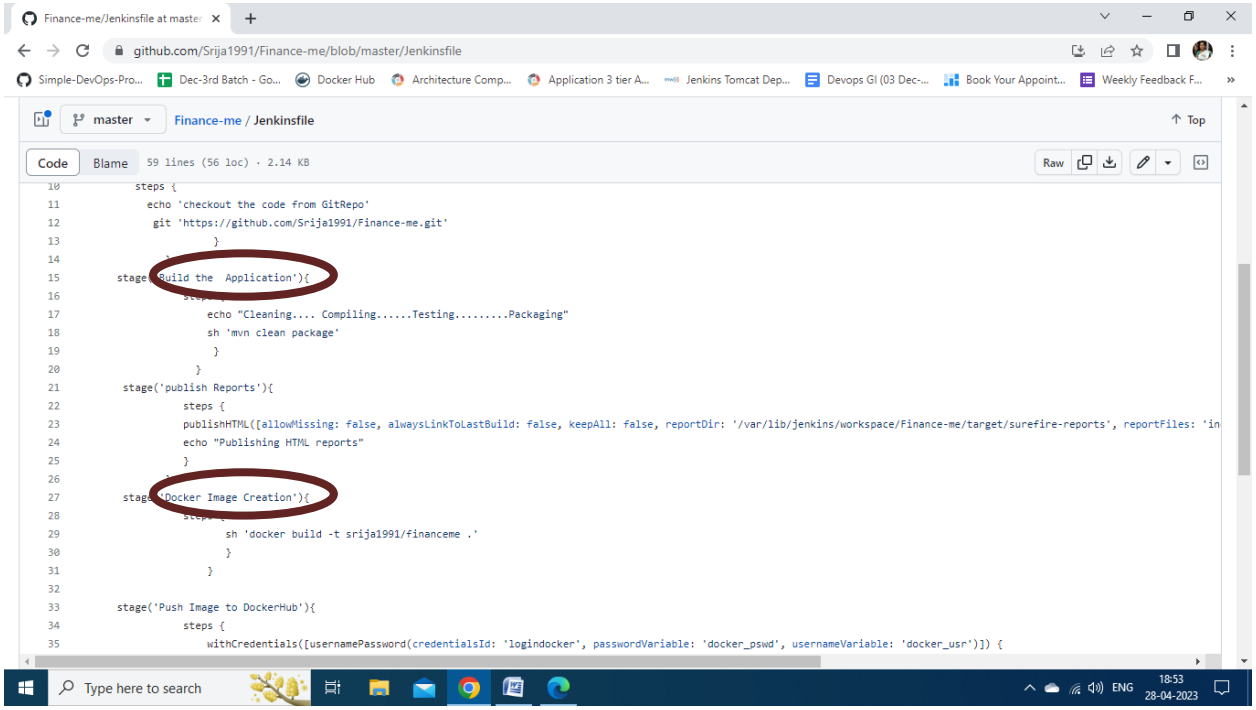


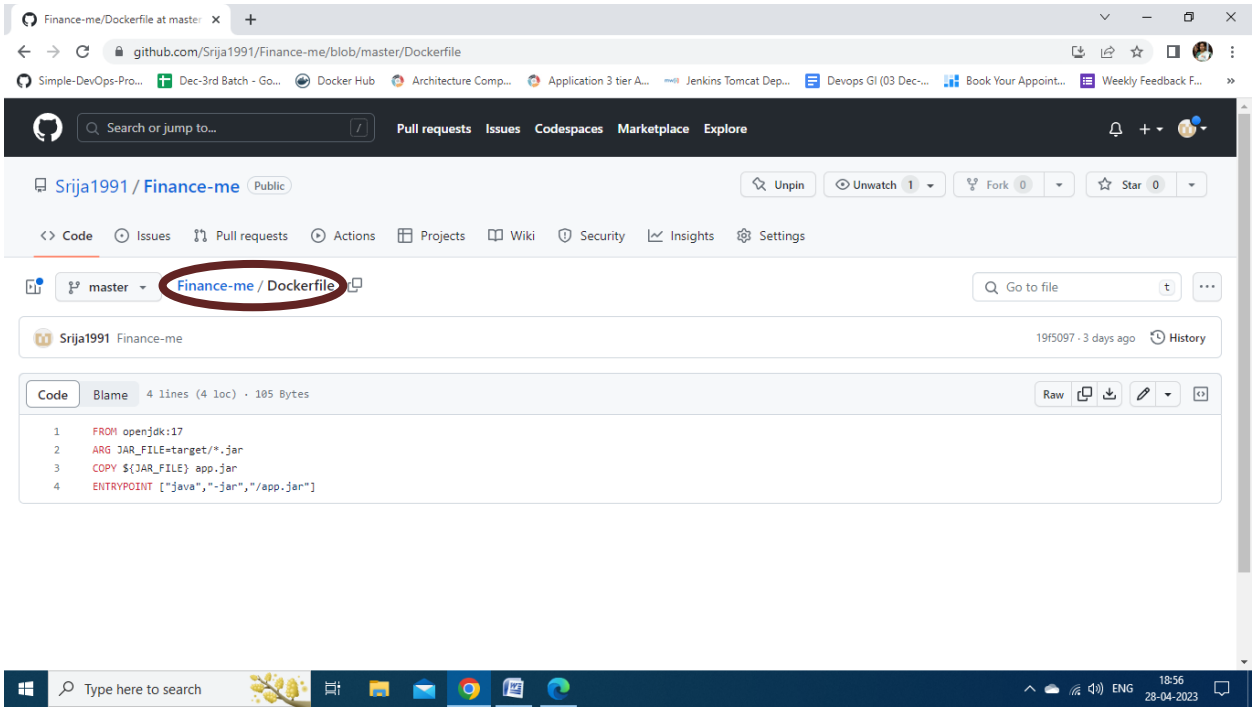
Here is the pipeline script written in file named Jenkinsfile. The flow of the script is as follows:

1. Checkout the code from git.

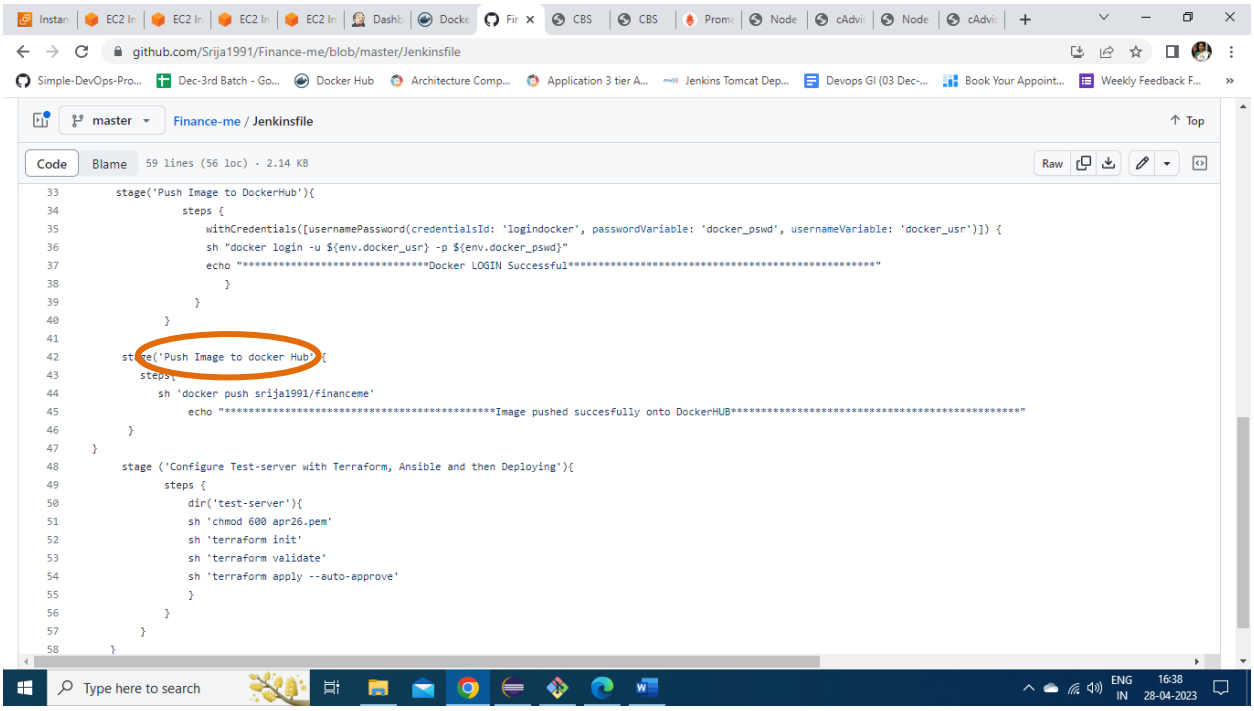


2. Build the application using maven build tool using command "*mvn clean package*".
3. Publish the HTML reports.
4. Create the docker using jar file and Dockerfile.
5. Push image to docker hub using appropriate credentials.



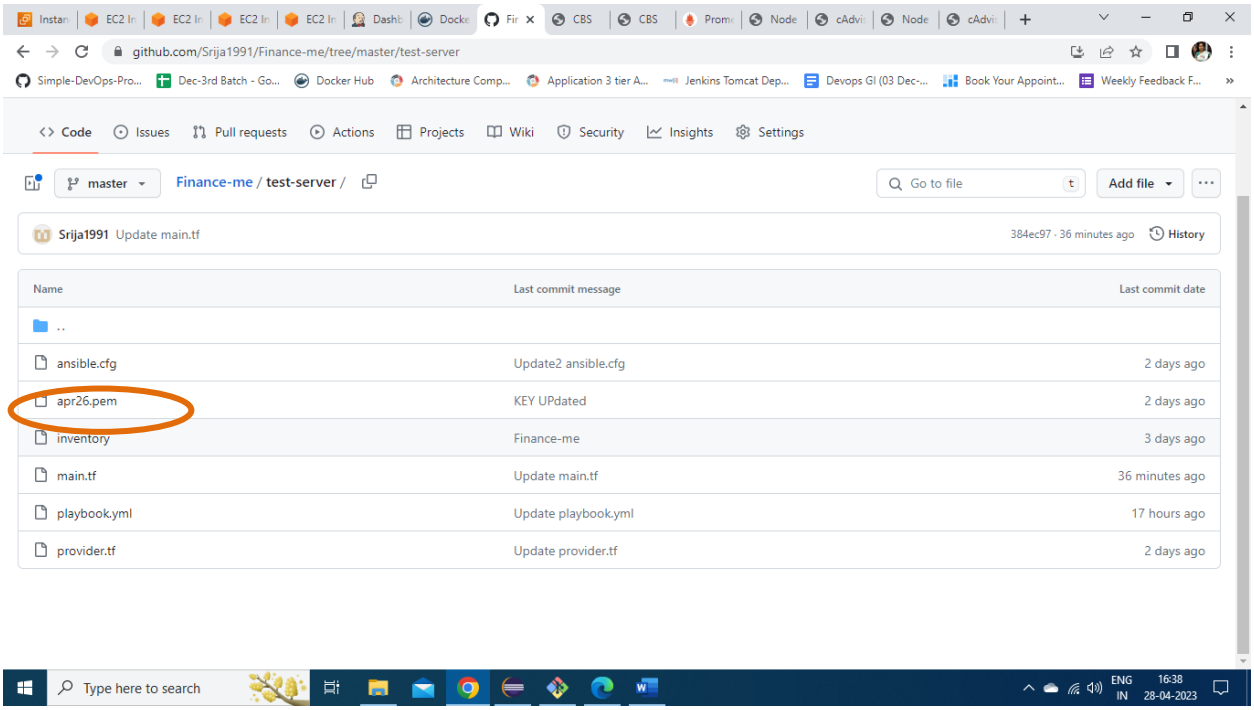


6. Terraform is configured to create a test-server and deploy application using Ansible playbook.

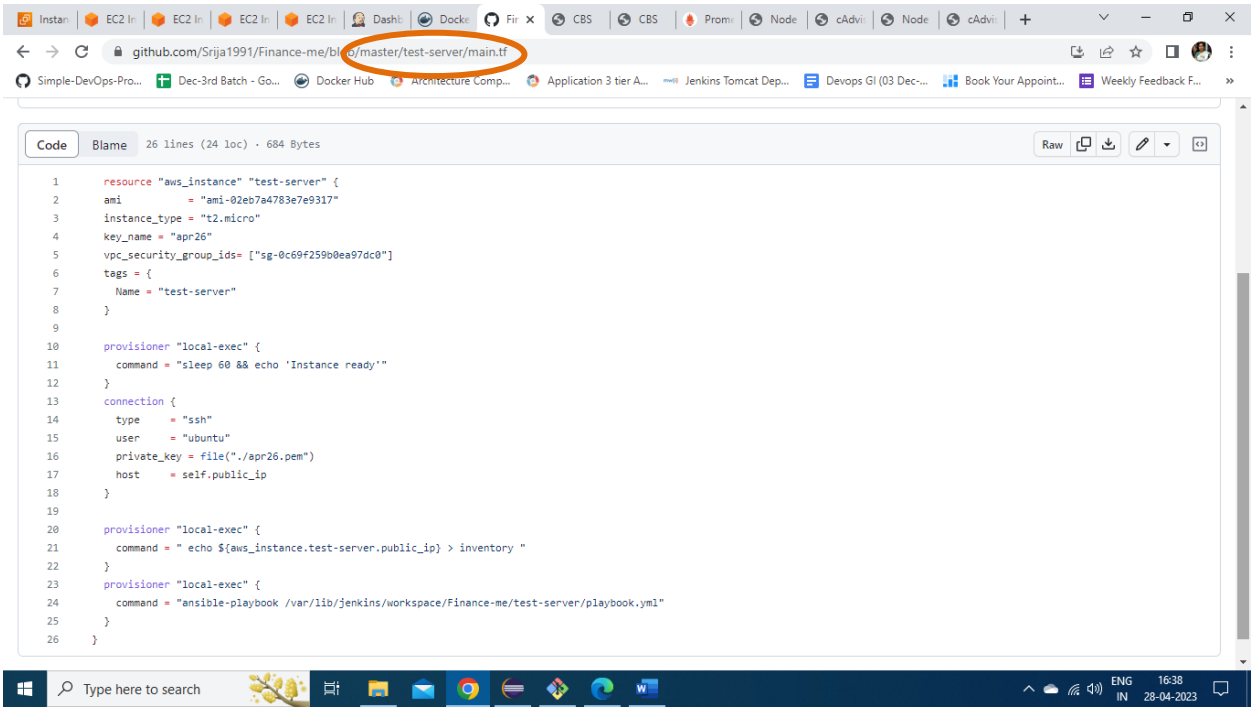


```
33      stage('Push Image to DockerHub'){
34              steps {
35                  withCredentials([usernamePassword(credentialsId: 'logindocker', passwordVariable: 'docker_pswd', usernameVariable: 'docker_usr')]) {
36                  sh "docker login -u ${env.docker_usr} -p ${env.docker_pswd}"
37                  echo "*****************************Docker LOGIN Successful*****************************************************"
38                     }
39                 }
40              }
41
42      stage('Push Image to docker Hub'){
43         steps{
44            sh 'docker push srija1991/financeme'
45                echo "*********************************************Image pushed succesfully onto DockerHUB*************************************************"
46       }
47  }
48      stage ('Configure Test-server with Terraform, Ansible and then Deploying'){
49            steps {
50                dir('test-server'){
51                sh 'chmod 600 apr26.pem'
52                sh 'terraform init'
53                sh 'terraform validate'
54                sh 'terraform apply --auto-approve'
55                }
56            }
57        }
58    }
```

Here is the test-server folder which has *main.tf* file to create an EC2 instance in the region mentioned and attach the keypair *"apr26*".



Name	Last commit message	Last commit date
..		
ansible.cfg	Update2 ansible.cfg	2 days ago
apr26.pem	KEY UPdated	2 days ago
inventory	Finance-me	3 days ago
main.tf	Update main.tf	36 minutes ago
playbook.yml	Update playbook.yml	17 hours ago
provider.tf	Update provider.tf	2 days ago

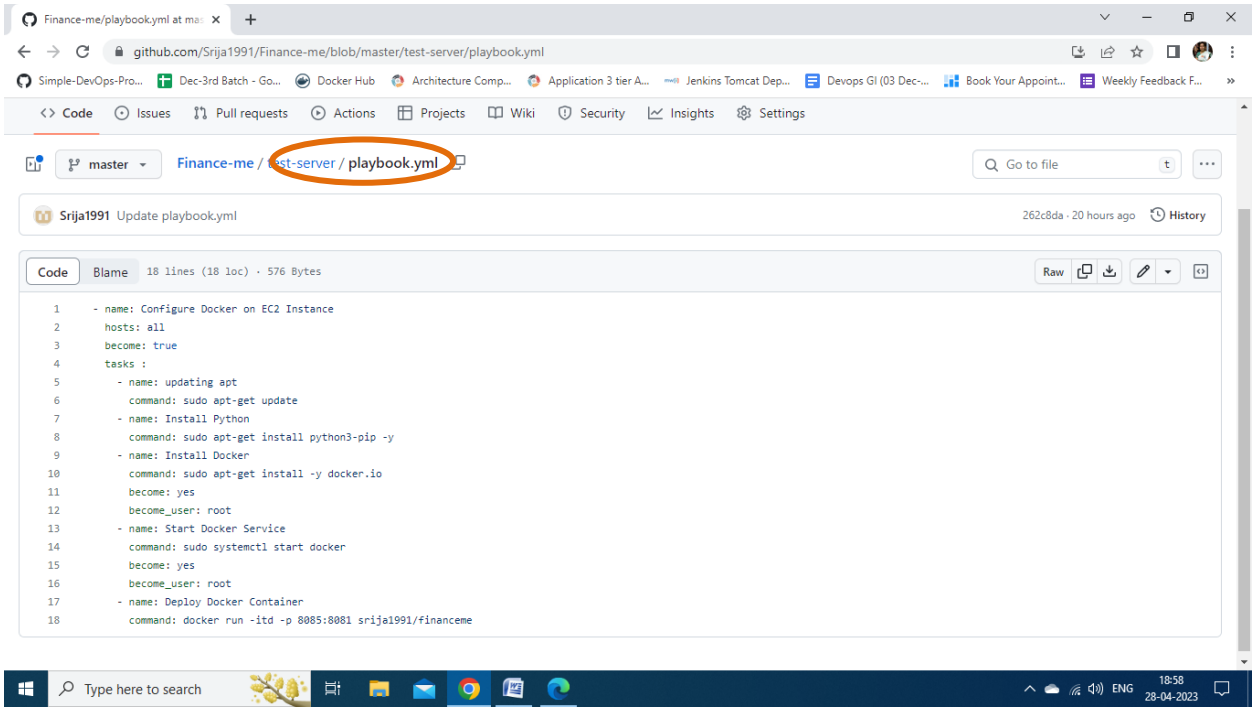


github.com/Srija1991/Finance-me/blob/master/test-server/main.tf

Code | Blame | 26 lines (24 loc) · 684 Bytes | Raw

```
resource "aws_instance" "test-server" {
ami           = "ami-02eb7a4783e7e9317"
instance_type = "t2.micro"
key_name = "apr26"
vpc_security_group_ids= ["sg-0c69f259b0ea97dc0"]
tags = {
  Name = "test-server"
}

provisioner "local-exec" {
  command = "sleep 60 && echo 'Instance ready'"
}
connection {
  type     = "ssh"
  user     = "ubuntu"
  private_key = file("./apr26.pem")
  host     = self.public_ip
}

provisioner "local-exec" {
  command = " echo ${aws_instance.test-server.public_ip} > inventory "
}
provisioner "local-exec" {
  command = "ansible-playbook /var/lib/jenkins/workspace/Finance-me/test-server/playbook.yml"
}
}
```

Here is the playbook to configure the *test-server* with python and docker.



github.com/Srija1991/Finance-me/blob/master/test-server/playbook.yml

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

master | Finance-me / test-server / playbook.yml | Go to file

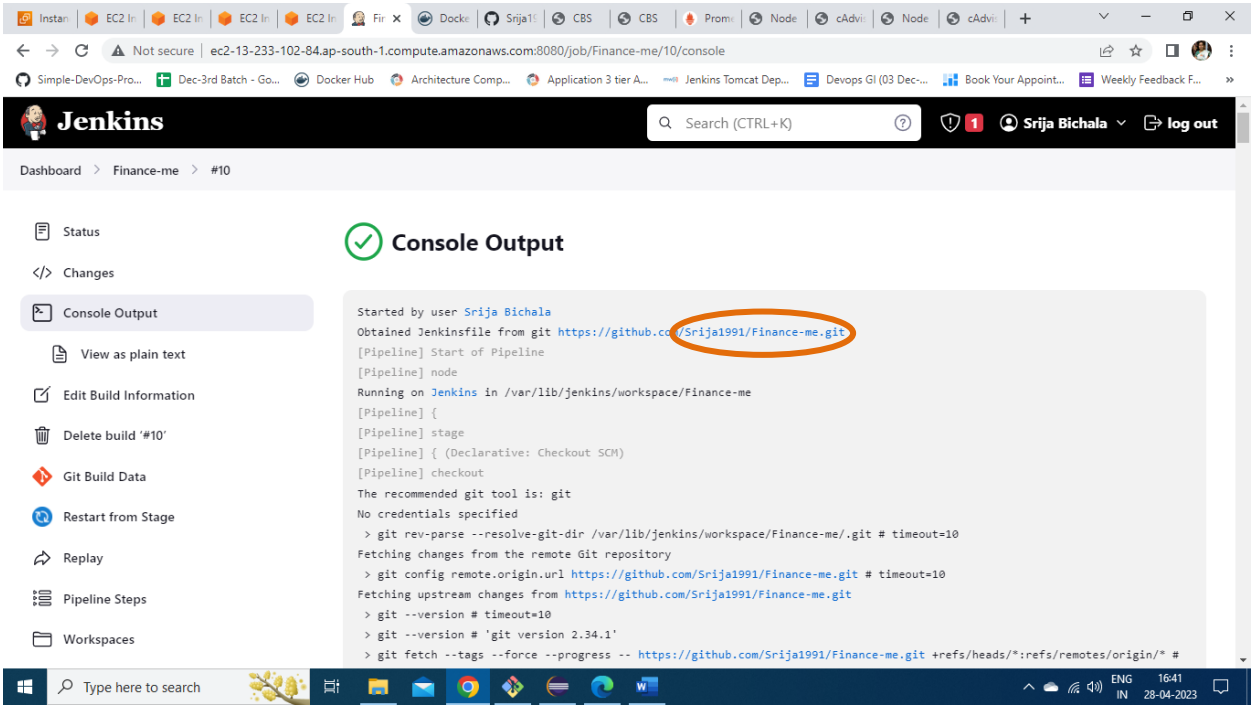
Srija1991 Update playbook.yml — 262c8da · 20 hours ago | History

Code | Blame | 18 lines (18 loc) · 576 Bytes | Raw

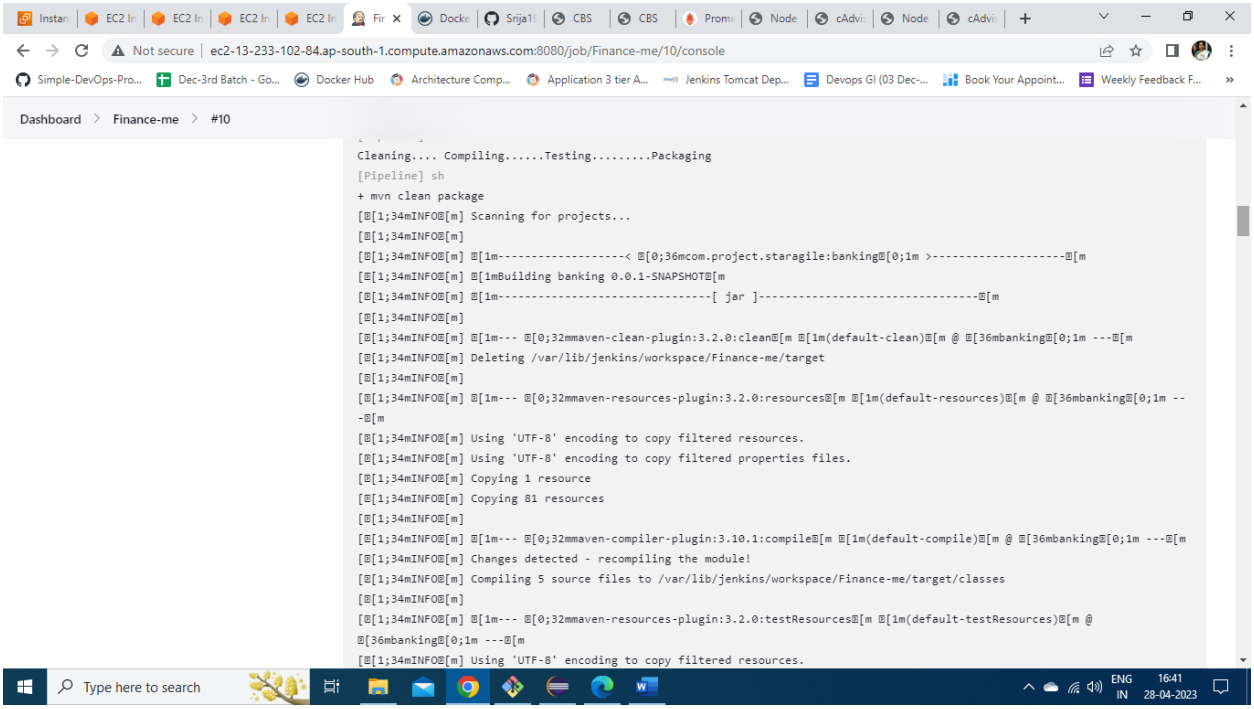
```
- name: Configure Docker on EC2 Instance
  hosts: all
  become: true
  tasks :
    - name: updating apt
      command: sudo apt-get update
    - name: Install Python
      command: sudo apt-get install python3-pip -y
    - name: Install Docker
      command: sudo apt-get install -y docker.io
      become: yes
      become_user: root
    - name: Start Docker Service
      command: sudo systemctl start docker
      become: yes
      become_user: root
    - name: Deploy Docker Container
      command: docker run -itd -p 8085:8081 srija1991/financeme
```

A job(Finance-me) is created which then triggers selenium test(Selenium Test) job and then deploy the application on to production server using (Prod-server) job.

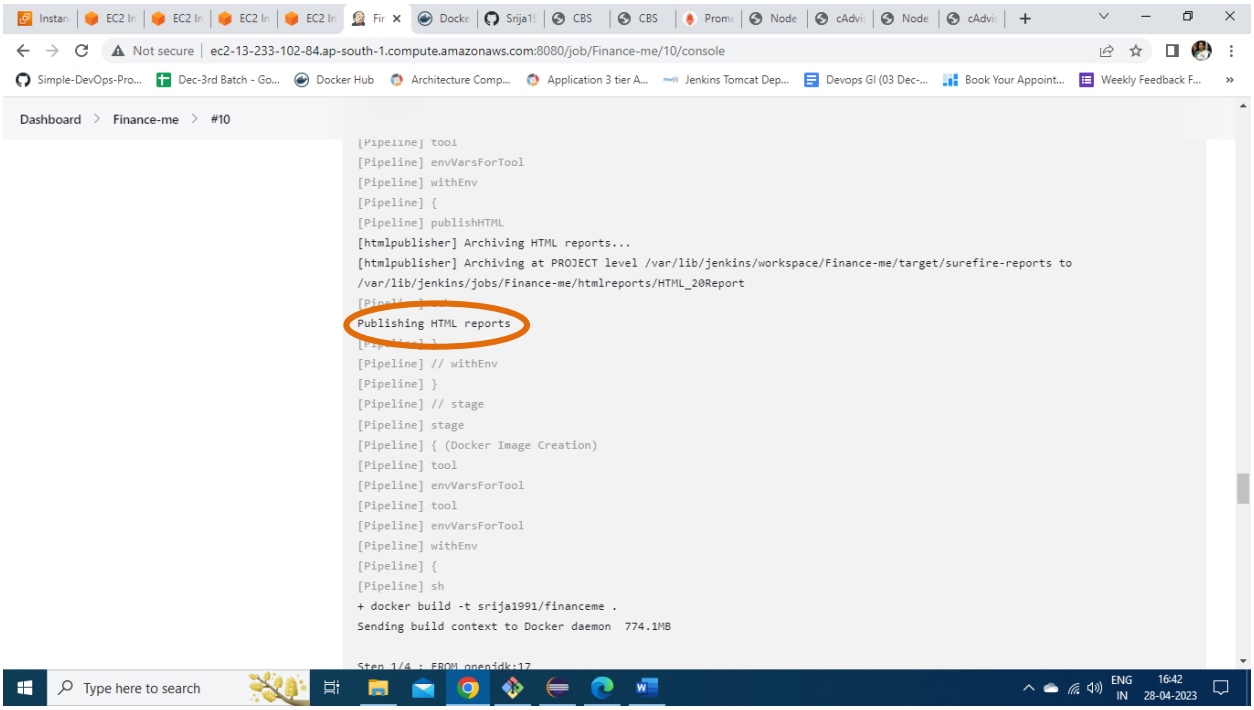
Here is the screenshot of Finance-me job executed successfully at build number 10.



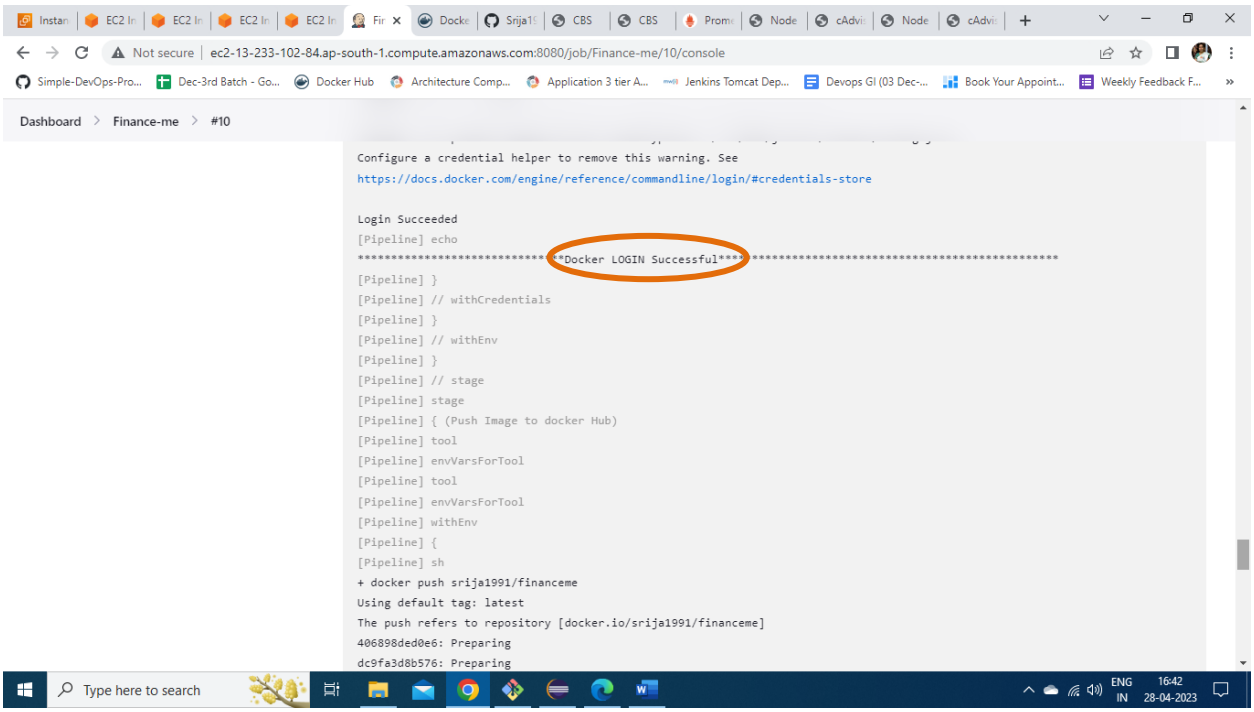
Created jar file displaying "*build success*".



Generated HTML reports.



Created a docker image and pushed onto Docker hub successfully.



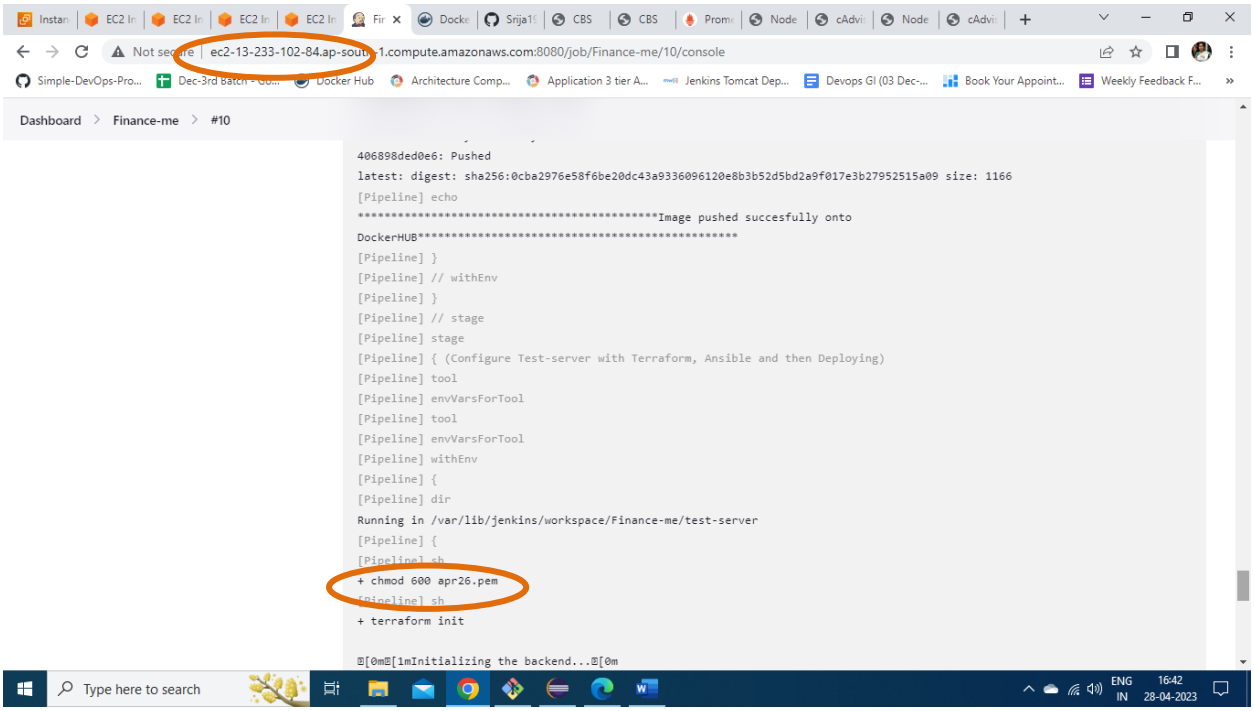
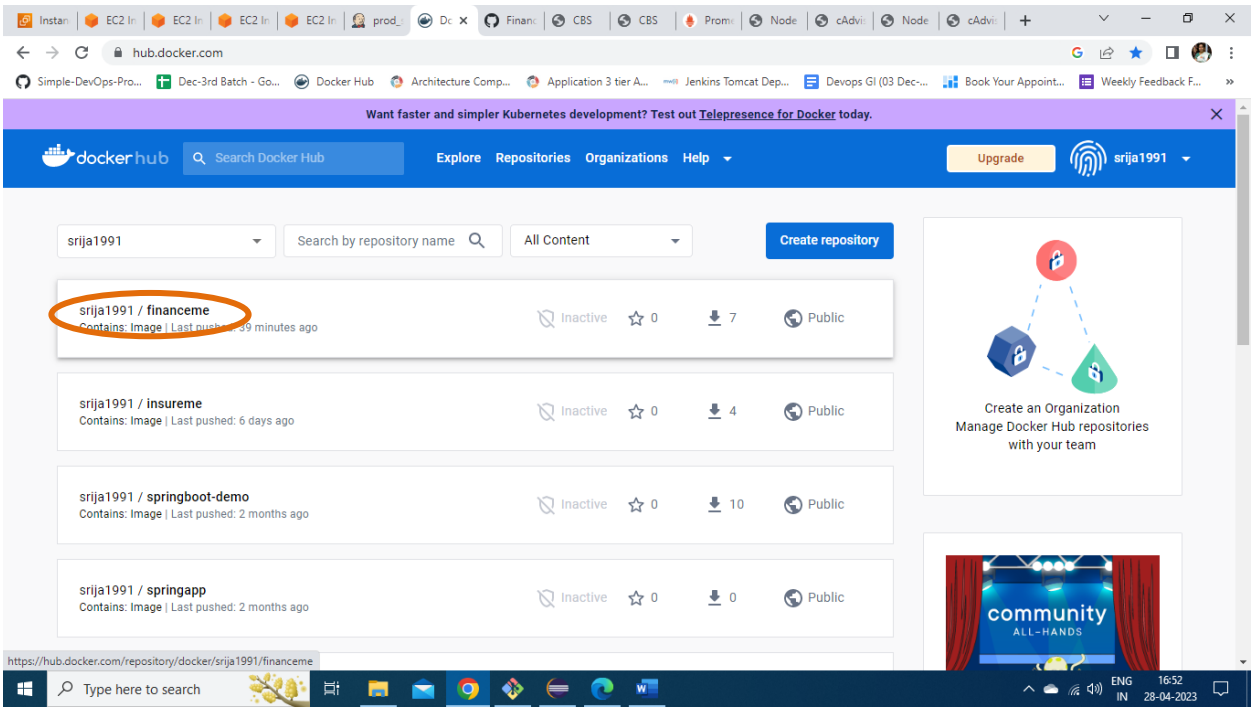
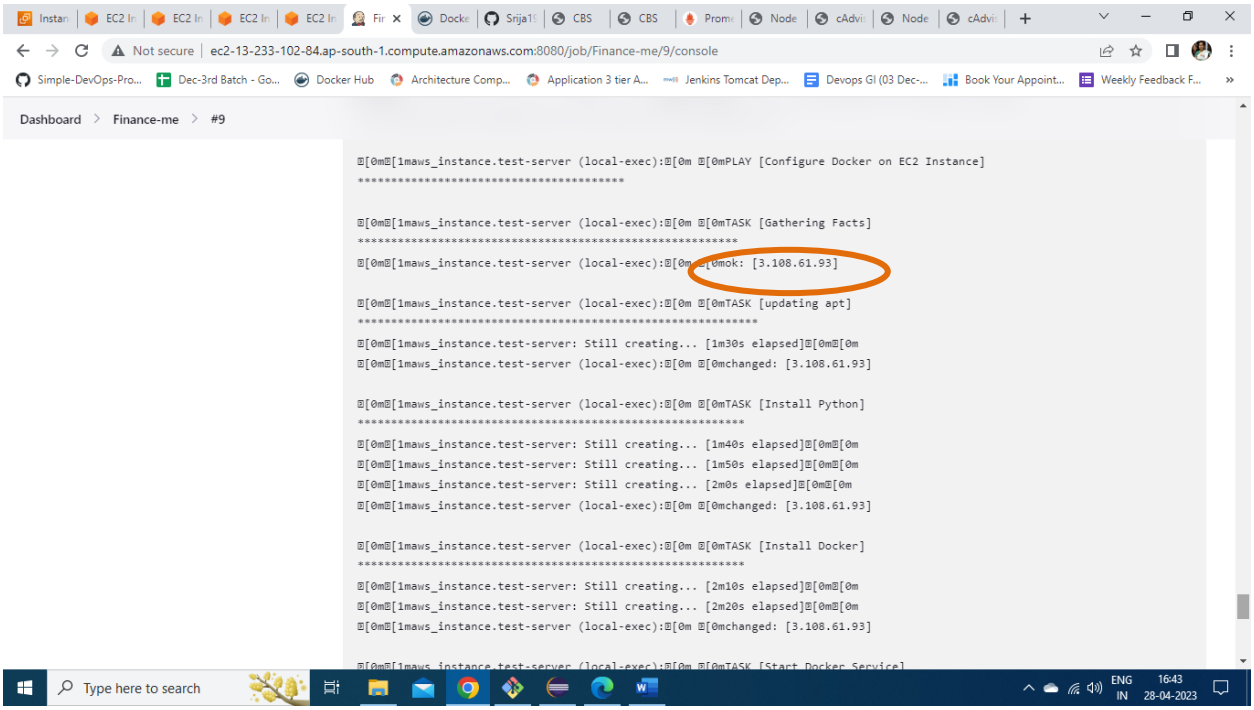
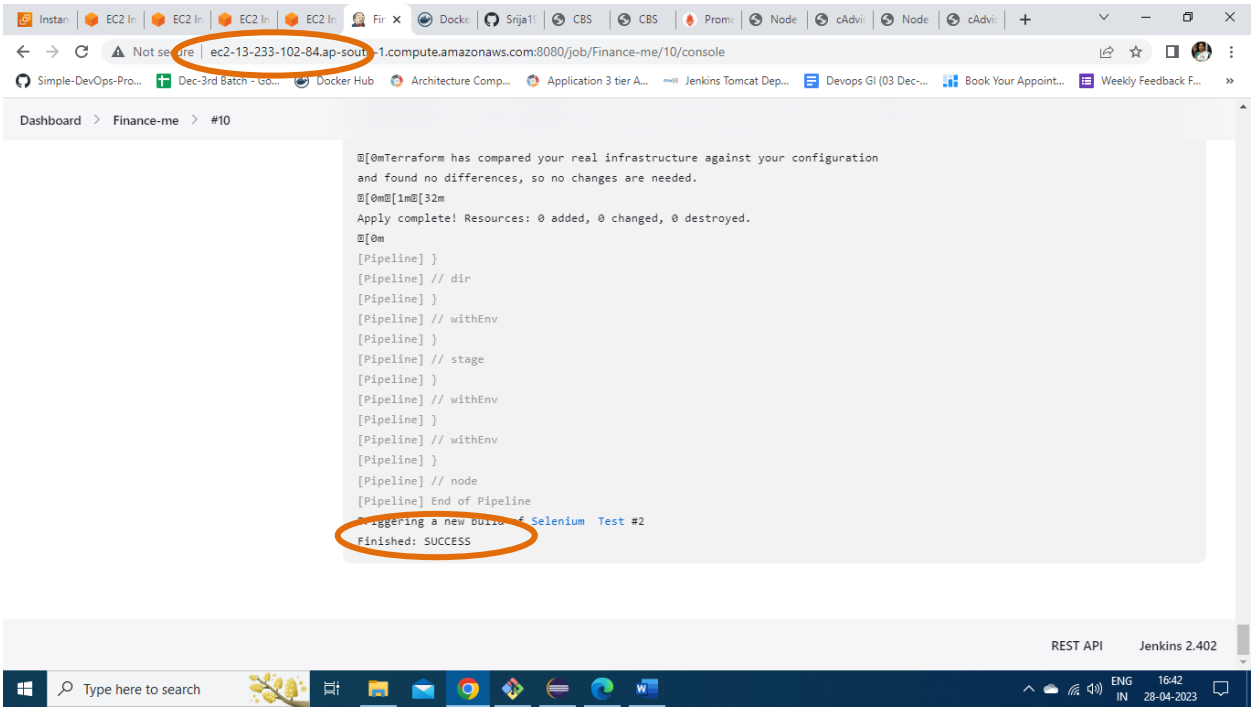


Image is pushed onto dockerhub with name *srija1991/financeme* which is used for further deployments.

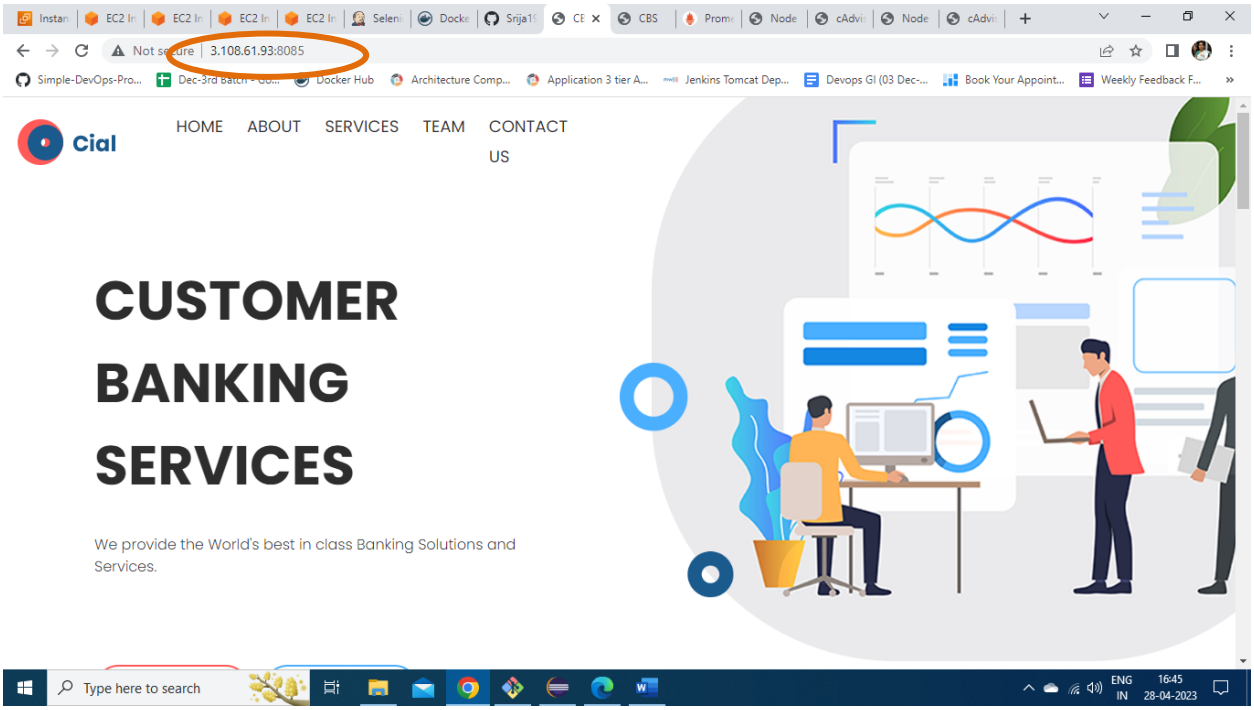


The Ansible-playbook is executed successfully and applicationis deployed onto test-server.

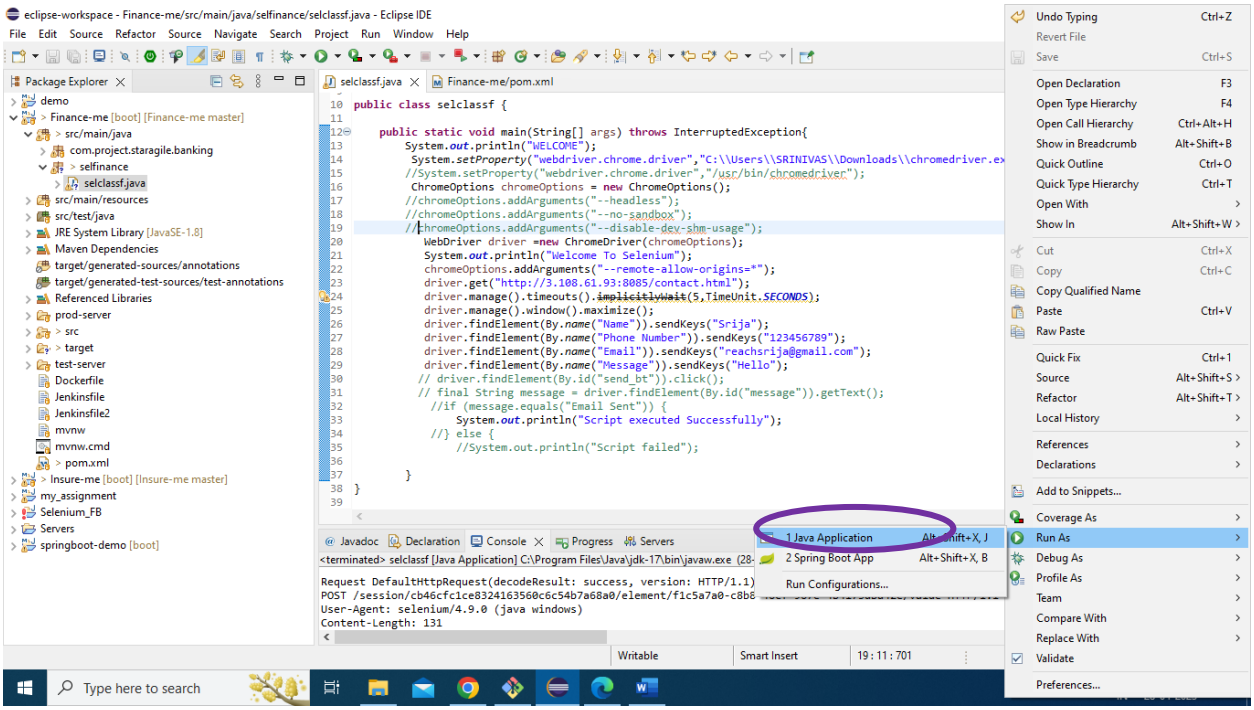




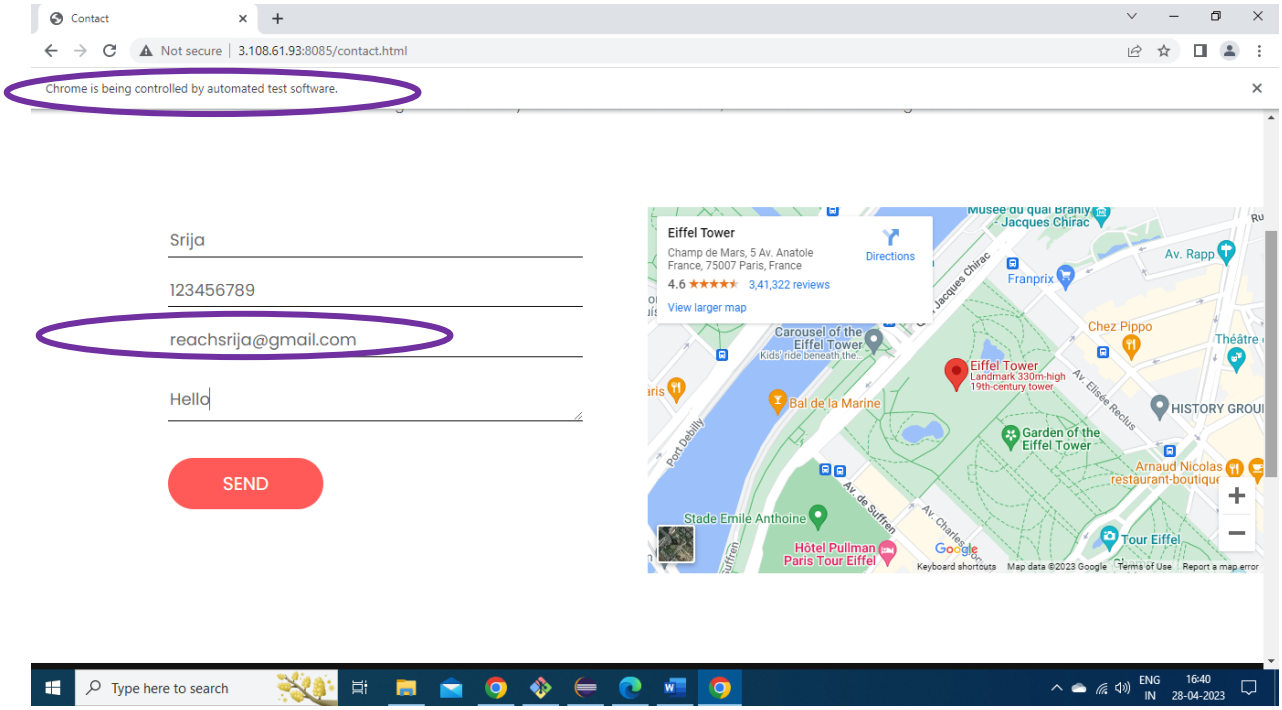
Test-server IP:3.108.61.93, and application is deployed on 8085 port.



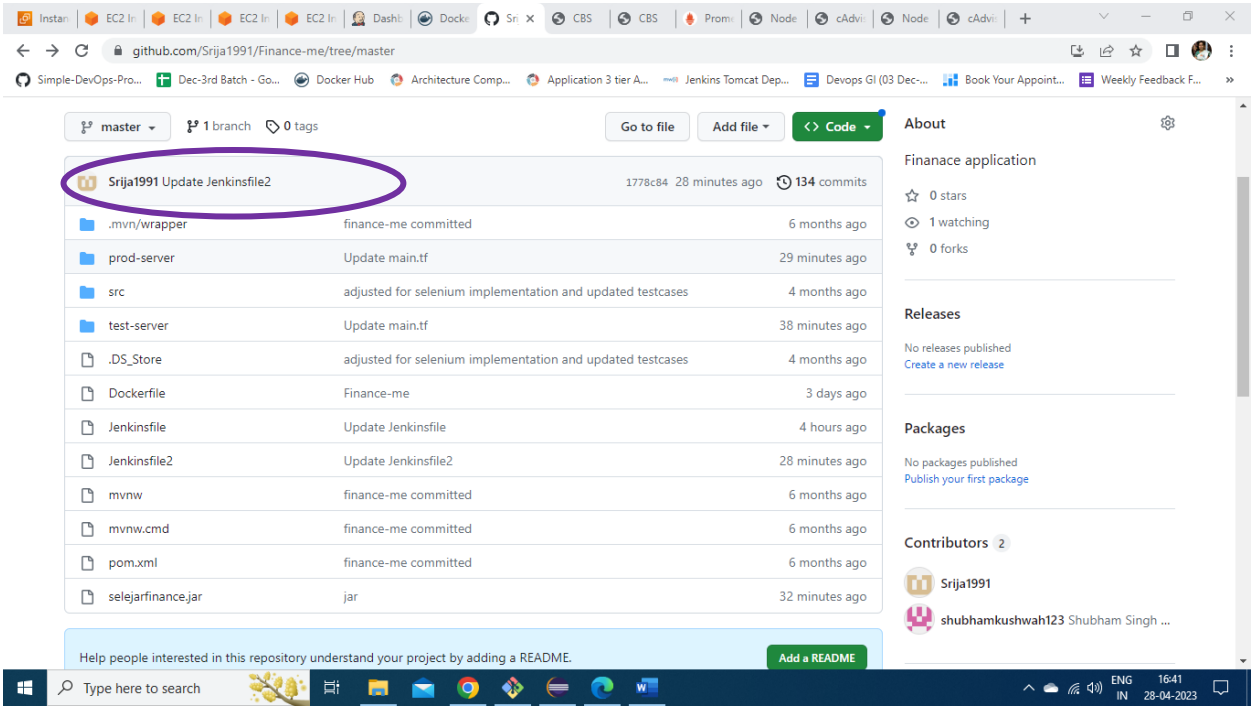
Then the selenium test code is written in eclipse and cross checked the automation test on windows platform.



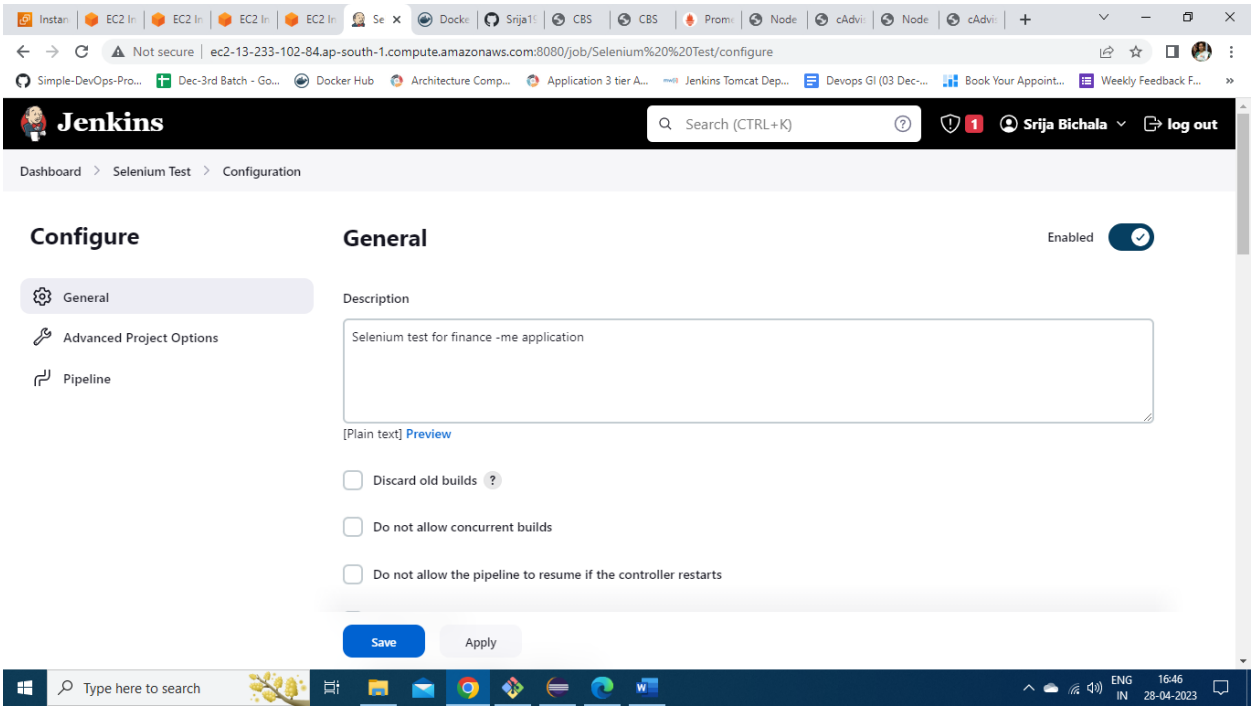
It is run as Java application to check the automation.

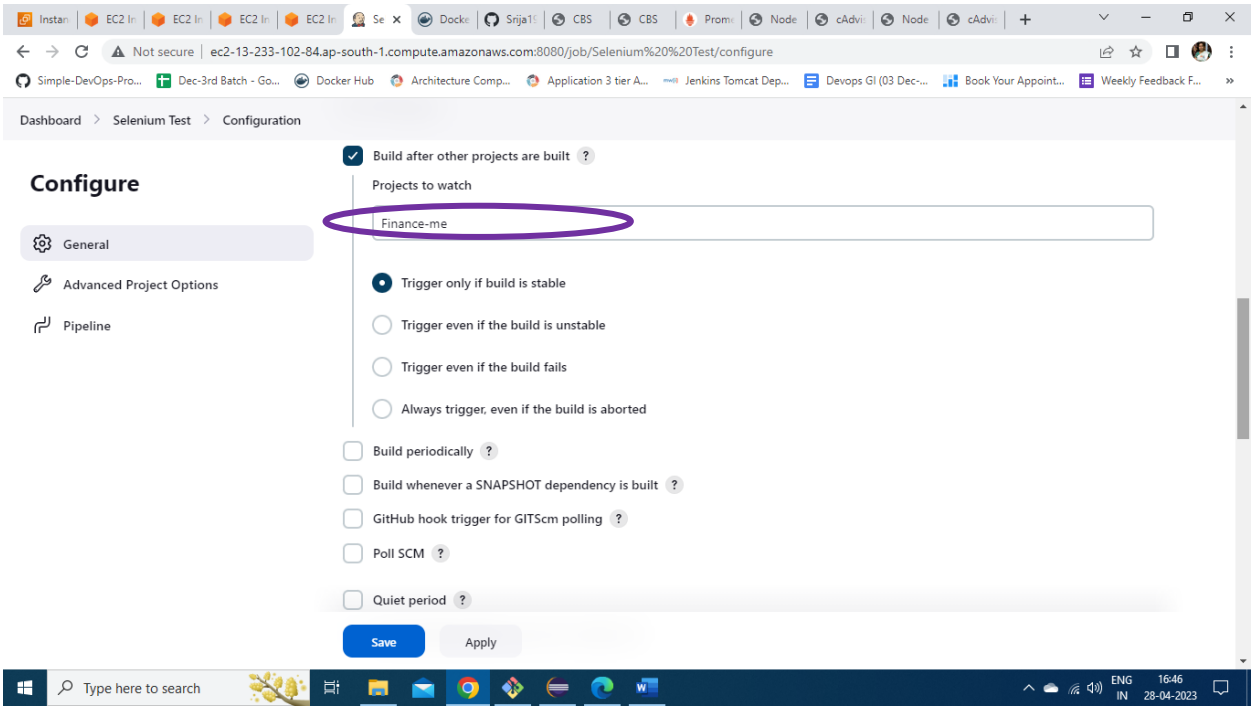


The jar file is exported as runnable jar and pushed onto github with name *selejarfinance.jar*.

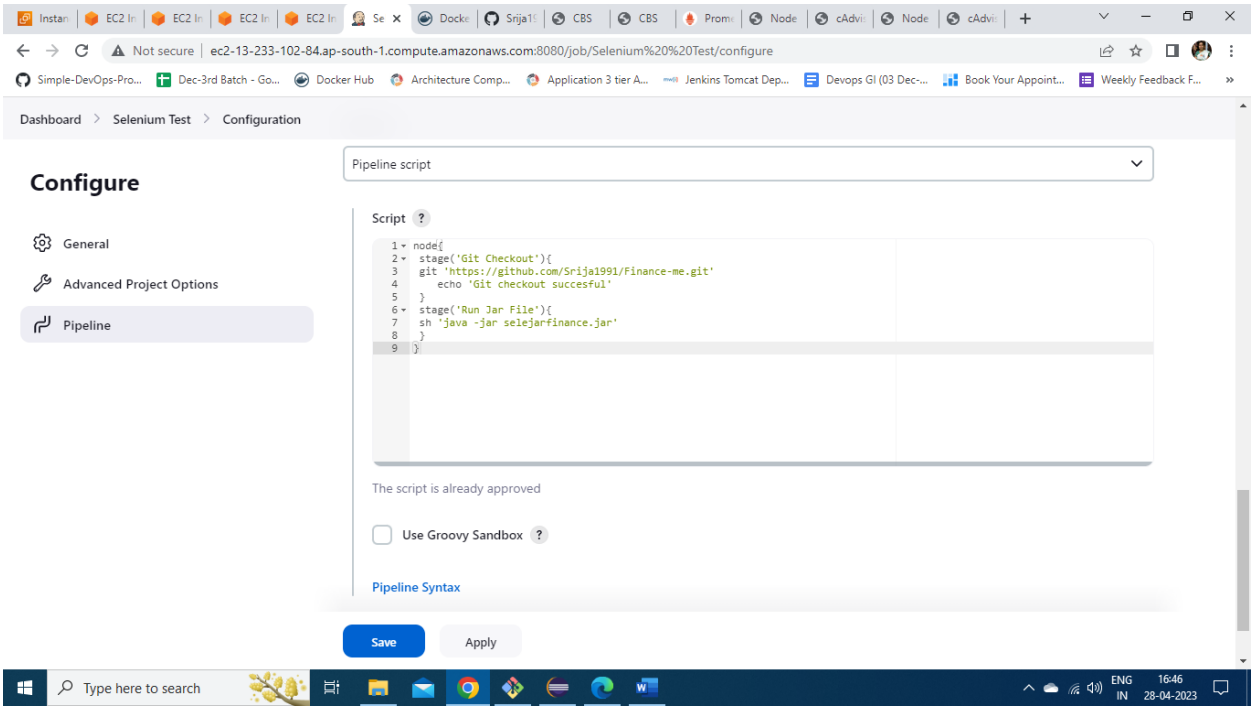


The Selenium Test is configured and triggered after the Finance-me job is successful and is shown here.

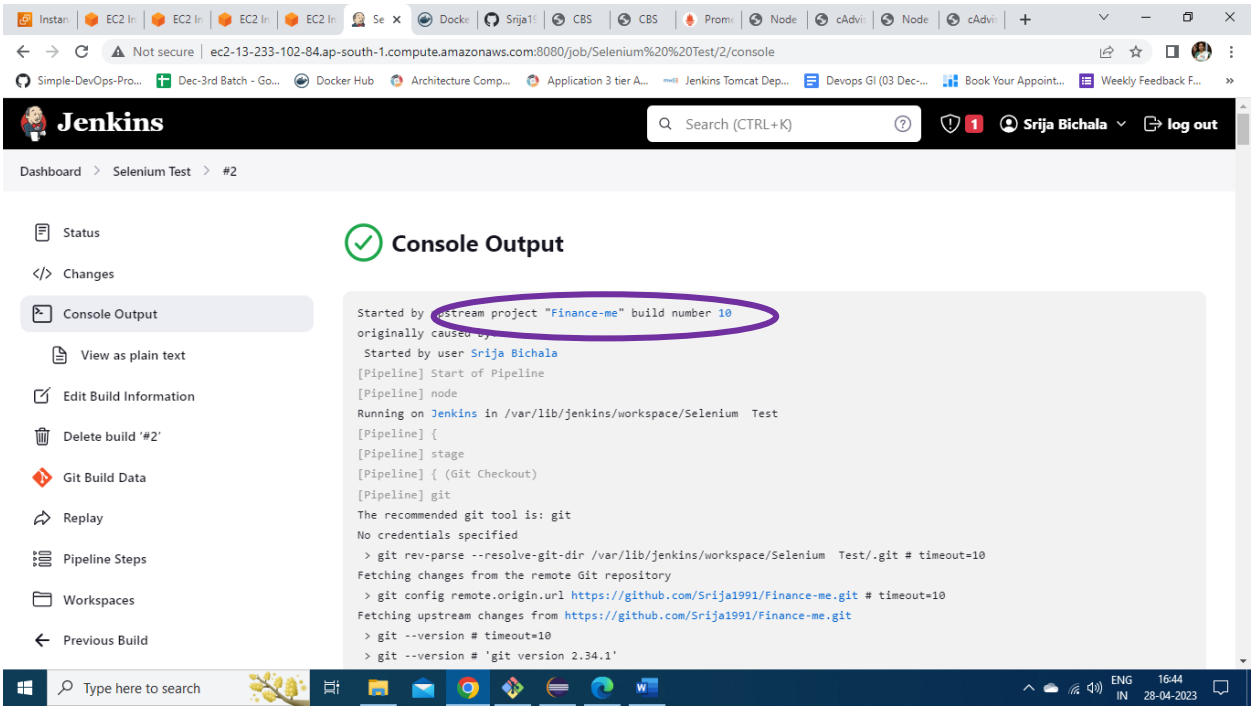


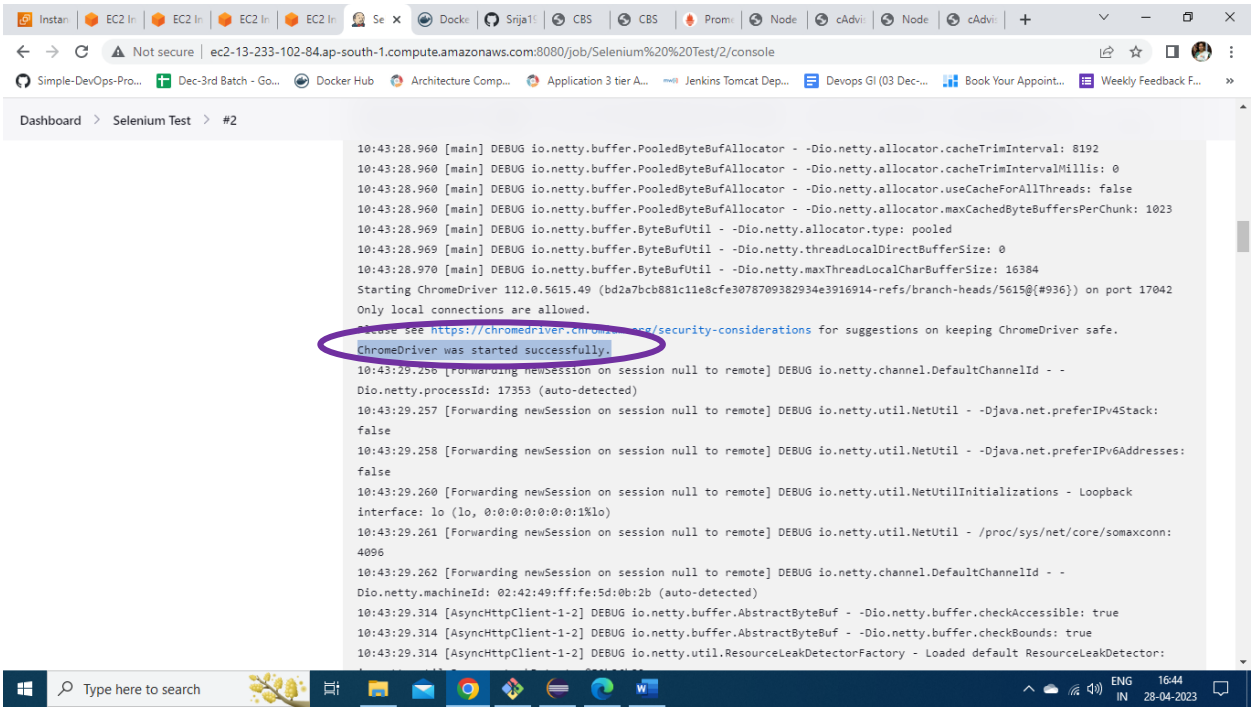


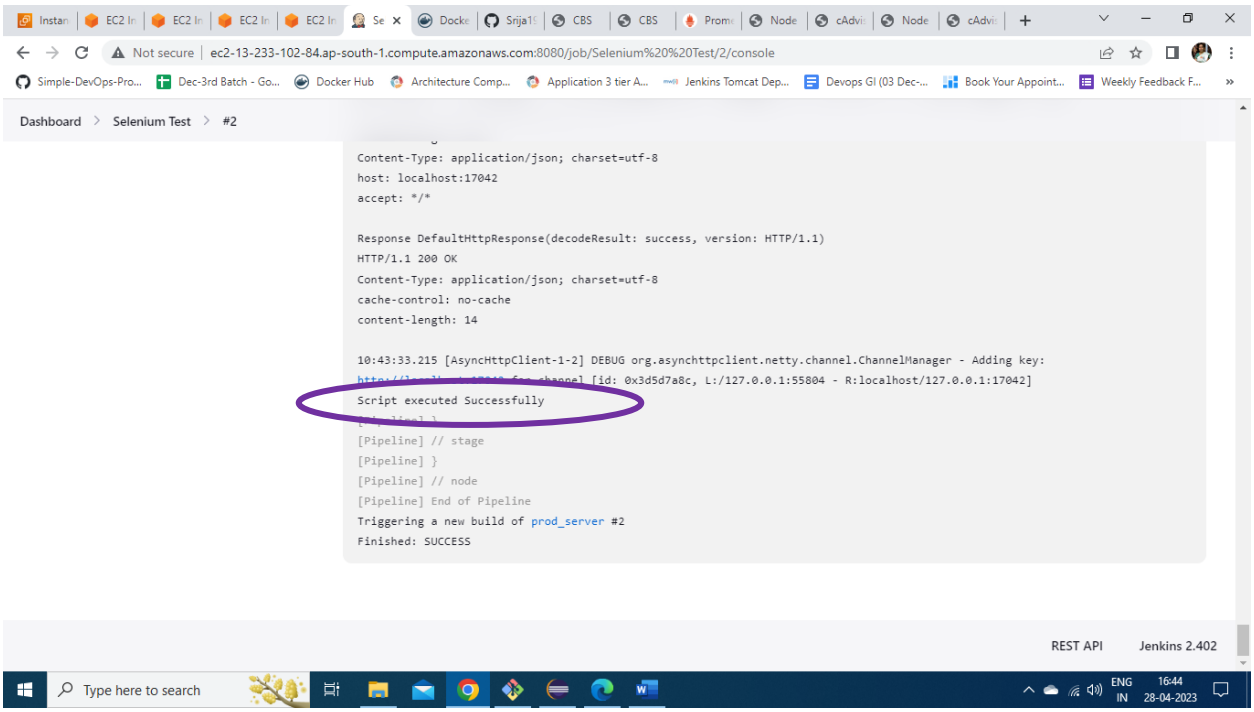
The pipeline script for selenium test case is shown here,



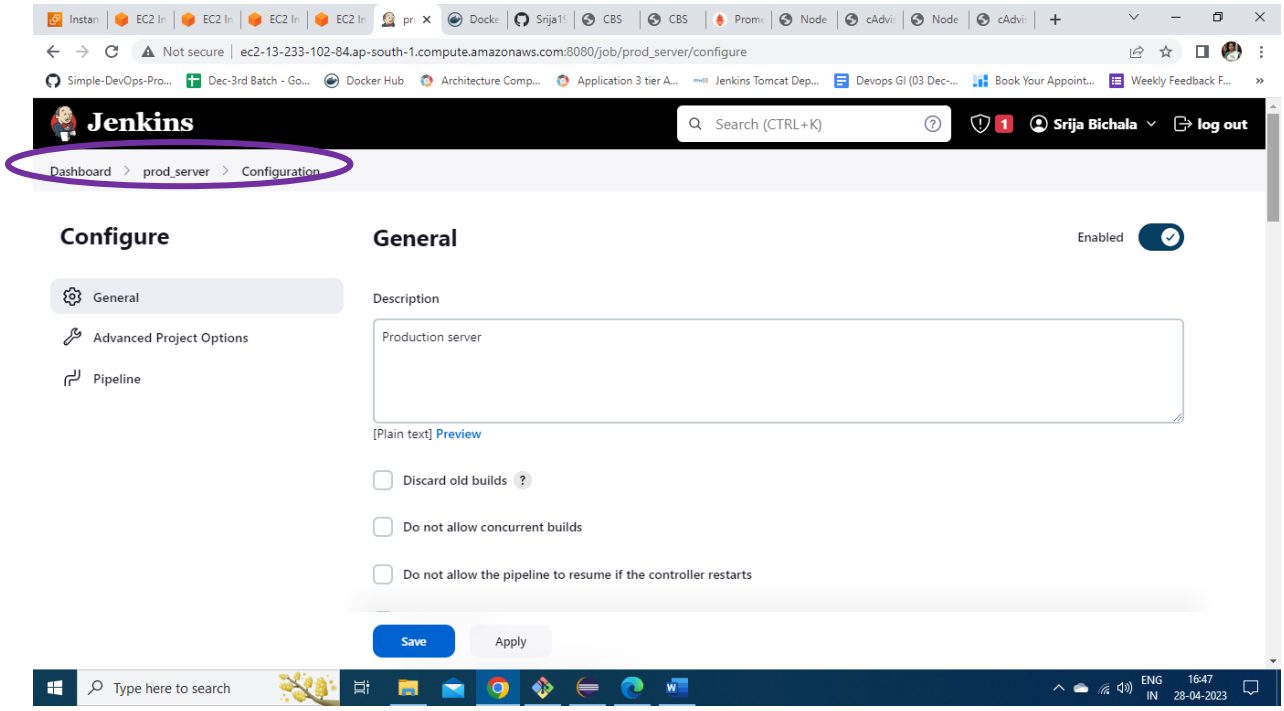
This is console output for selenium test which is triggered from Finance-me build 10. The build is success displaying the *google-chrome is started successfully* and *script is executed successfully.*

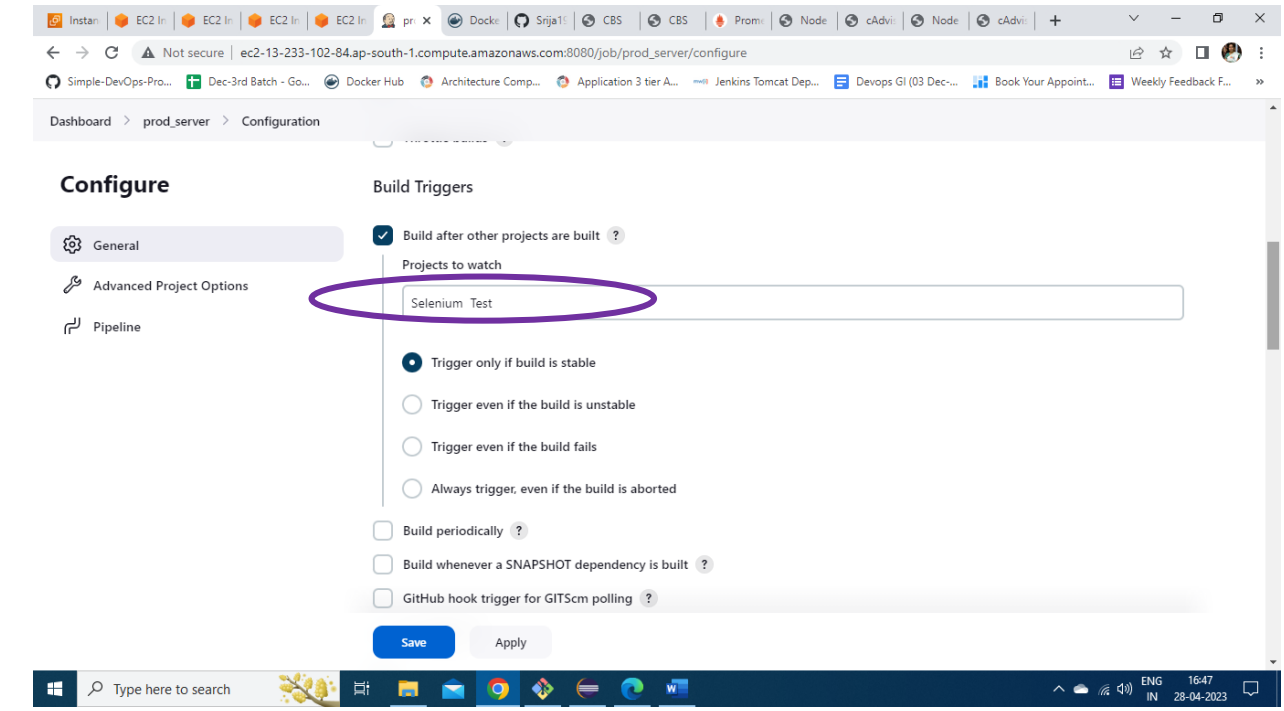


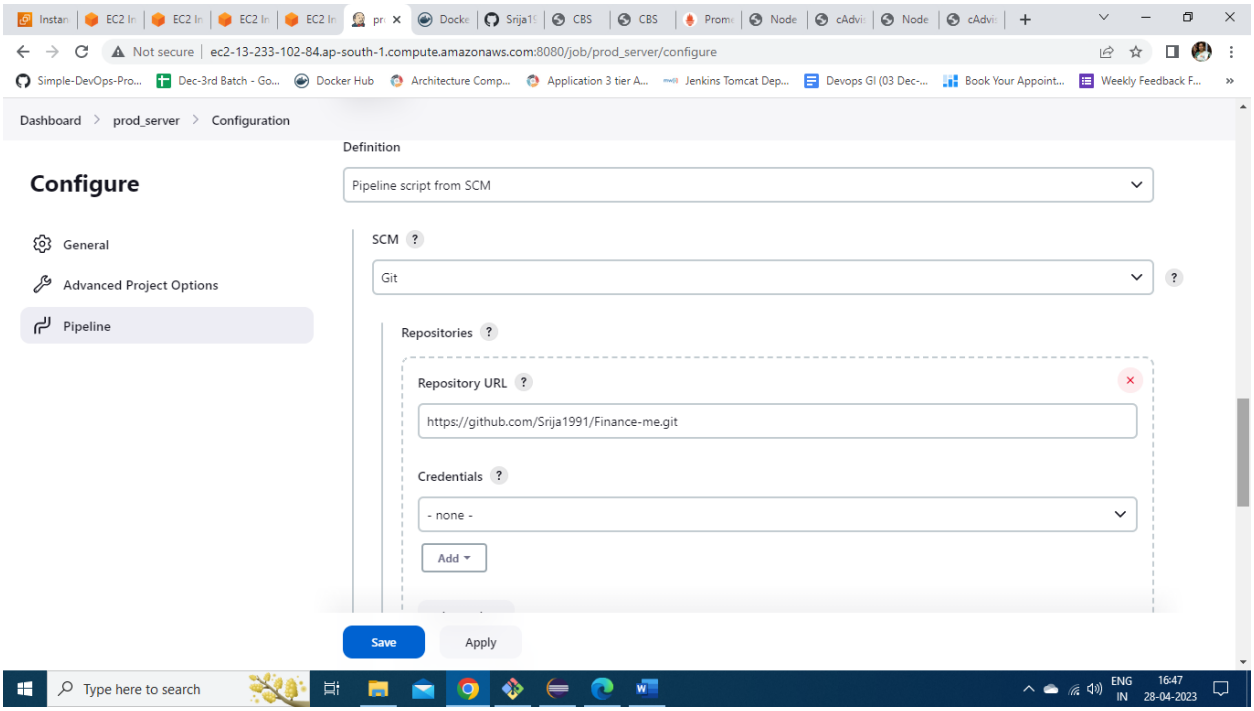

Not secure | ec2-13-233-102-84.ap-south-1.compute.amazonaws.com:8080/job/Selenium%20Test/2/console
Dashboard > Selenium Test > #2
10:43:28.960 [main] DEBUG io.netty.buffer.PooledByteBufAllocator - -Dio.netty.allocator.cacheTrimInterval: 8192
10:43:28.960 [main] DEBUG io.netty.buffer.PooledByteBufAllocator - -Dio.netty.allocator.cacheTrimIntervalMillis: 0
10:43:28.960 [main] DEBUG io.netty.buffer.PooledByteBufAllocator - -Dio.netty.allocator.useCacheForAllThreads: false
10:43:28.960 [main] DEBUG io.netty.buffer.PooledByteBufAllocator - -Dio.netty.allocator.maxCachedByteBuffersPerChunk: 1023
10:43:28.969 [main] DEBUG io.netty.buffer.ByteBufUtil - -Dio.netty.allocator.type: pooled
10:43:28.969 [main] DEBUG io.netty.buffer.ByteBufUtil - -Dio.netty.threadLocalDirectBufferSize: 0
10:43:28.970 [main] DEBUG io.netty.buffer.ByteBufUtil - -Dio.netty.maxThreadLocalCharBufferSize: 16384
Starting ChromeDriver 112.0.5615.49 (bd2a7bcb881c11e8cfe3078709382934e3916914-refs/branch-heads/5615@{#936}) on port 17042
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
10:43:29.256 [Forwarding newSession on session null to remote] DEBUG io.netty.channel.DefaultChannelId - -
Dio.netty.processId: 17353 (auto-detected)
10:43:29.257 [Forwarding newSession on session null to remote] DEBUG io.netty.util.NetUtil - -Djava.net.preferIPv4Stack:
false
10:43:29.258 [Forwarding newSession on session null to remote] DEBUG io.netty.util.NetUtil - -Djava.net.preferIPv6Addresses:
false
10:43:29.260 [Forwarding newSession on session null to remote] DEBUG io.netty.util.NetUtilInitializations - Loopback
interface: lo (lo, 0:0:0:0:0:0:0:1%lo)
10:43:29.261 [Forwarding newSession on session null to remote] DEBUG io.netty.util.NetUtil - /proc/sys/net/core/somaxconn:
4096
10:43:29.262 [Forwarding newSession on session null to remote] DEBUG io.netty.channel.DefaultChannelId - -
Dio.netty.machineId: 02:42:49:ff:fe:5d:0b:2b (auto-detected)
10:43:29.314 [AsyncHttpClient-1-2] DEBUG io.netty.buffer.AbstractByteBuf - -Dio.netty.buffer.checkAccessible: true
10:43:29.314 [AsyncHttpClient-1-2] DEBUG io.netty.buffer.AbstractByteBuf - -Dio.netty.buffer.checkBounds: true
10:43:29.314 [AsyncHttpClient-1-2] DEBUG io.netty.util.ResourceLeakDetectorFactory - Loaded default ResourceLeakDetector:
Type here to search
ENG IN 16:44 28-04-2023


Not secure | ec2-13-233-102-84.ap-south-1.compute.amazonaws.com:8080/job/Selenium%20Test/2/console
Dashboard > Selenium Test > #2
Content-Type: application/json; charset=utf-8
host: localhost:17042
accept: */*
Response DefaultHttpResponse(decodeResult: success, version: HTTP/1.1)
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
cache-control: no-cache
content-length: 14
10:43:33.215 [AsyncHttpClient-1-2] DEBUG org.asynchttpclient.netty.channel.ChannelManager - Adding key:
Script executed Successfully
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Triggering a new build of prod_server #2
Finished: SUCCESS
REST API
Jenkins 2.402
Type here to search
ENG IN 16:44 28-04-2023

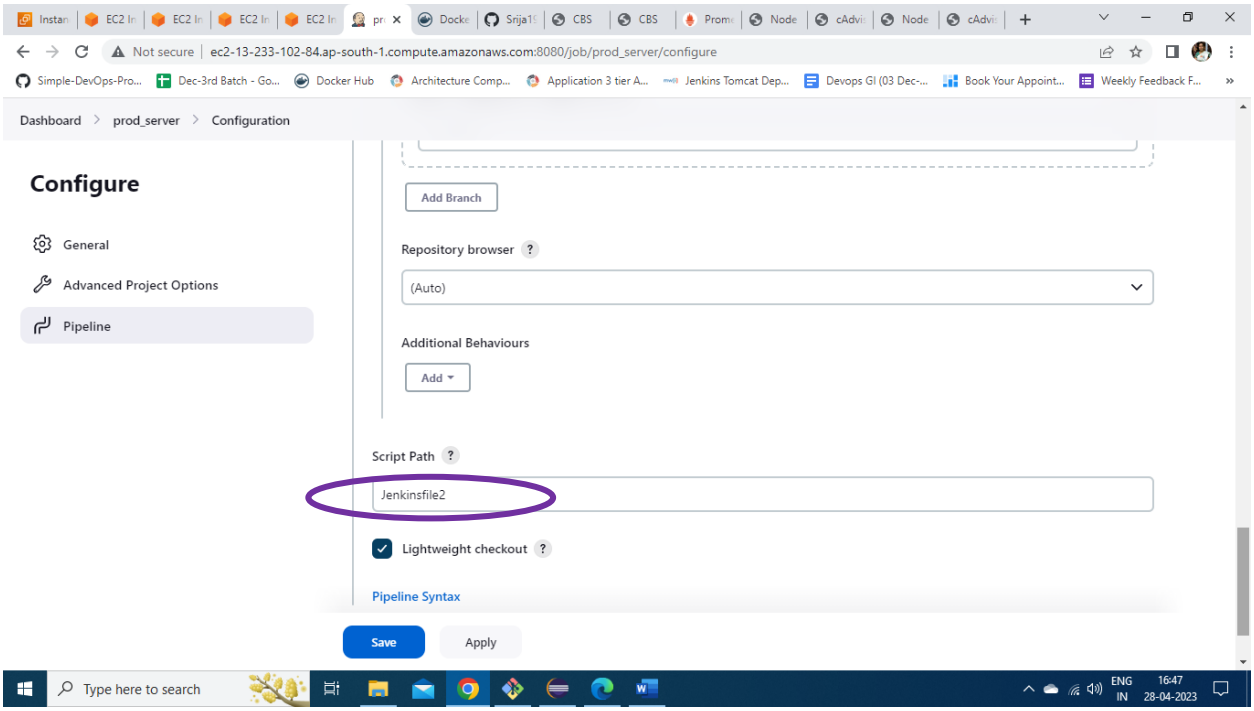
Once the test is successful, the application is deployed onto production server by triggering the job *prod_server*. The configuration for prod_server is shown here.



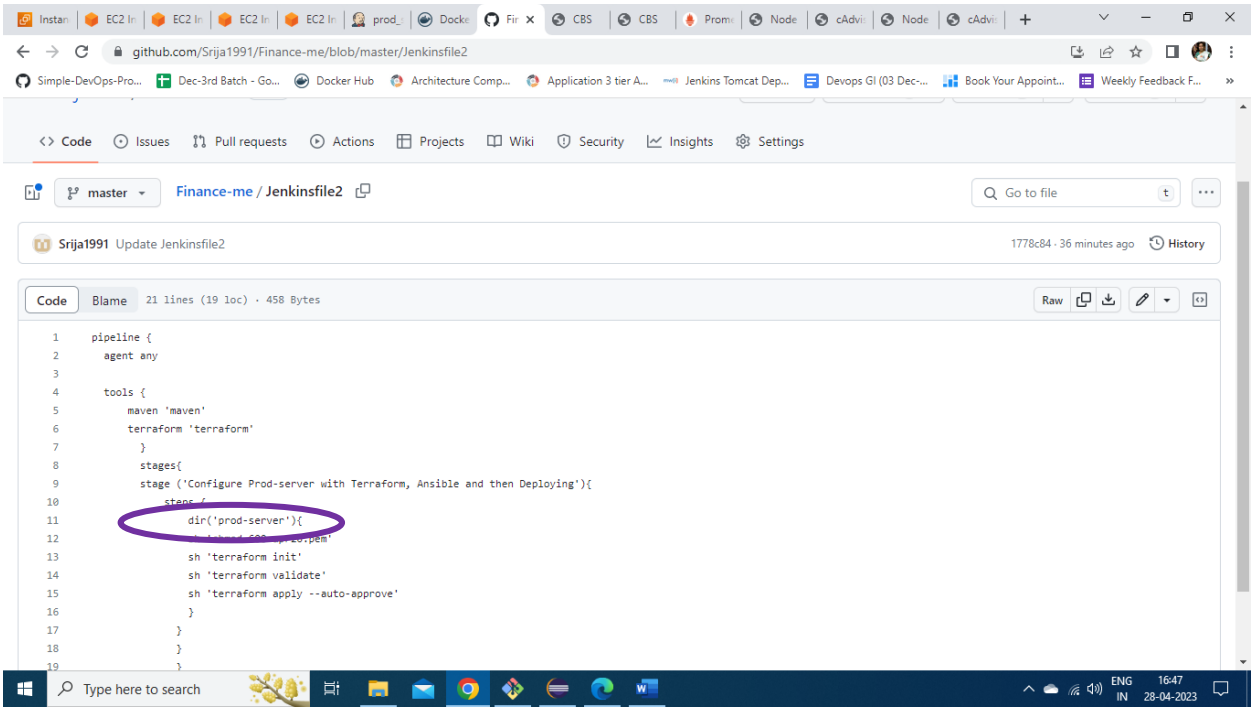




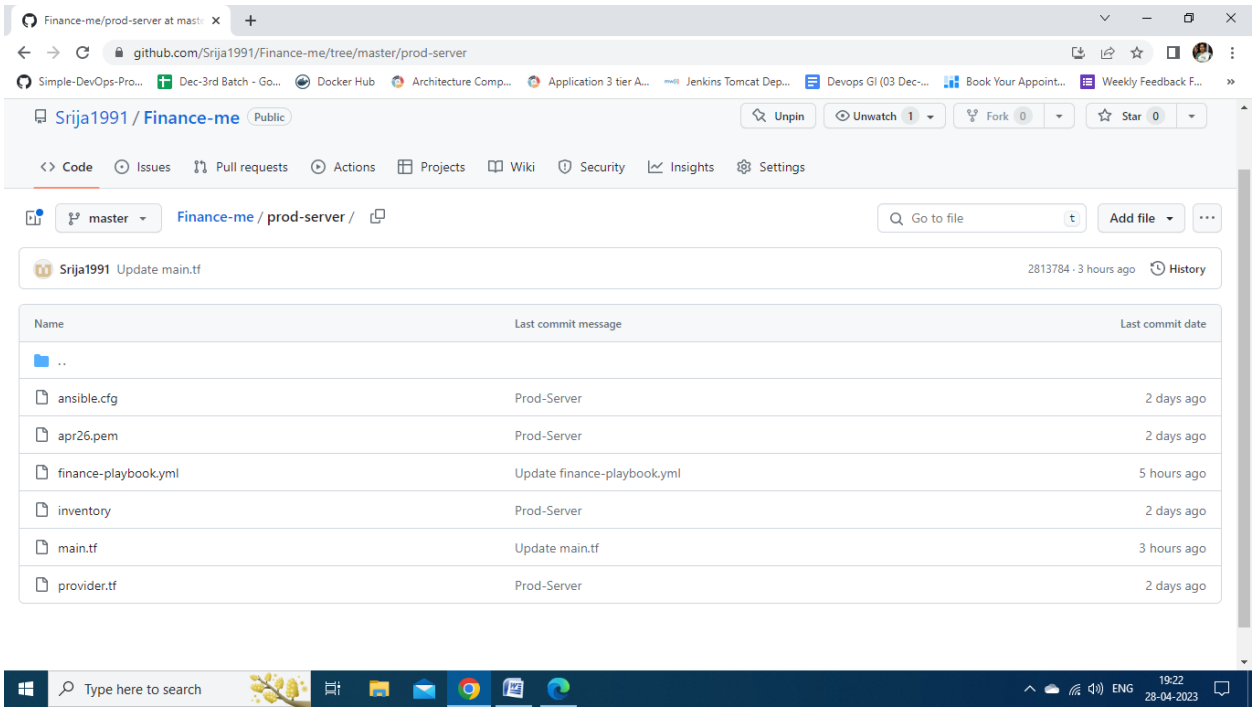
This job is triggered after successful build of Selenium Test job.

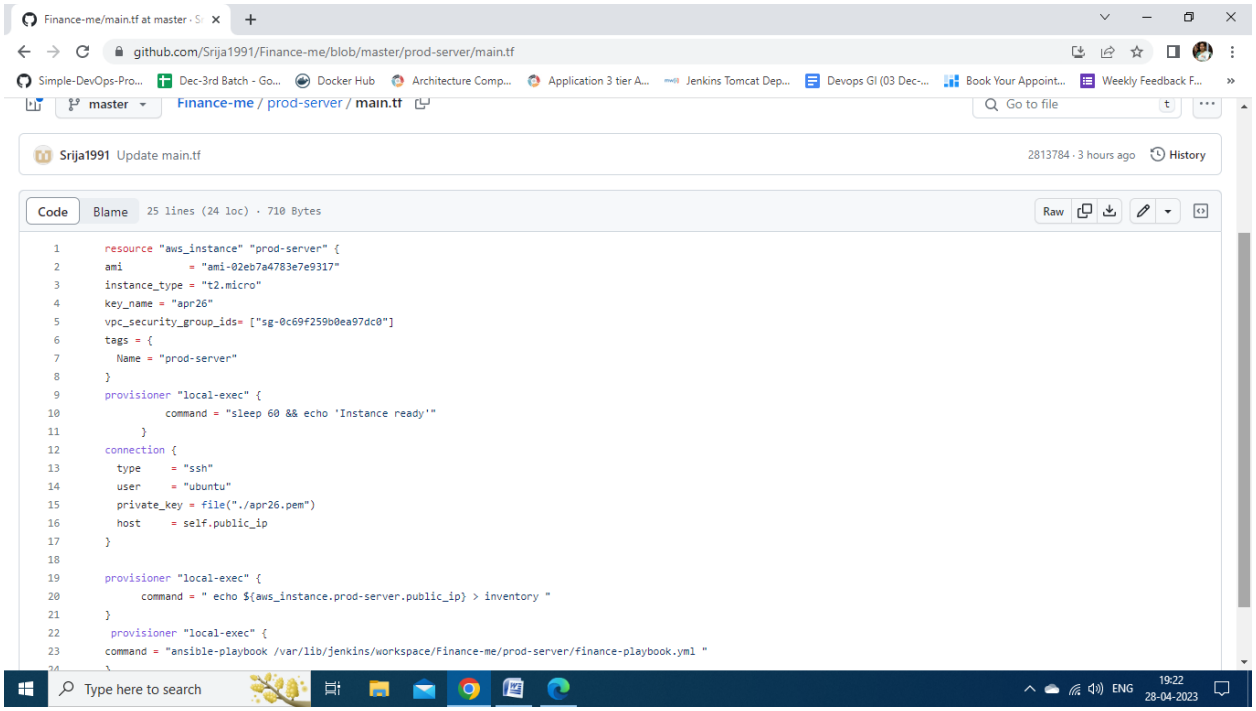


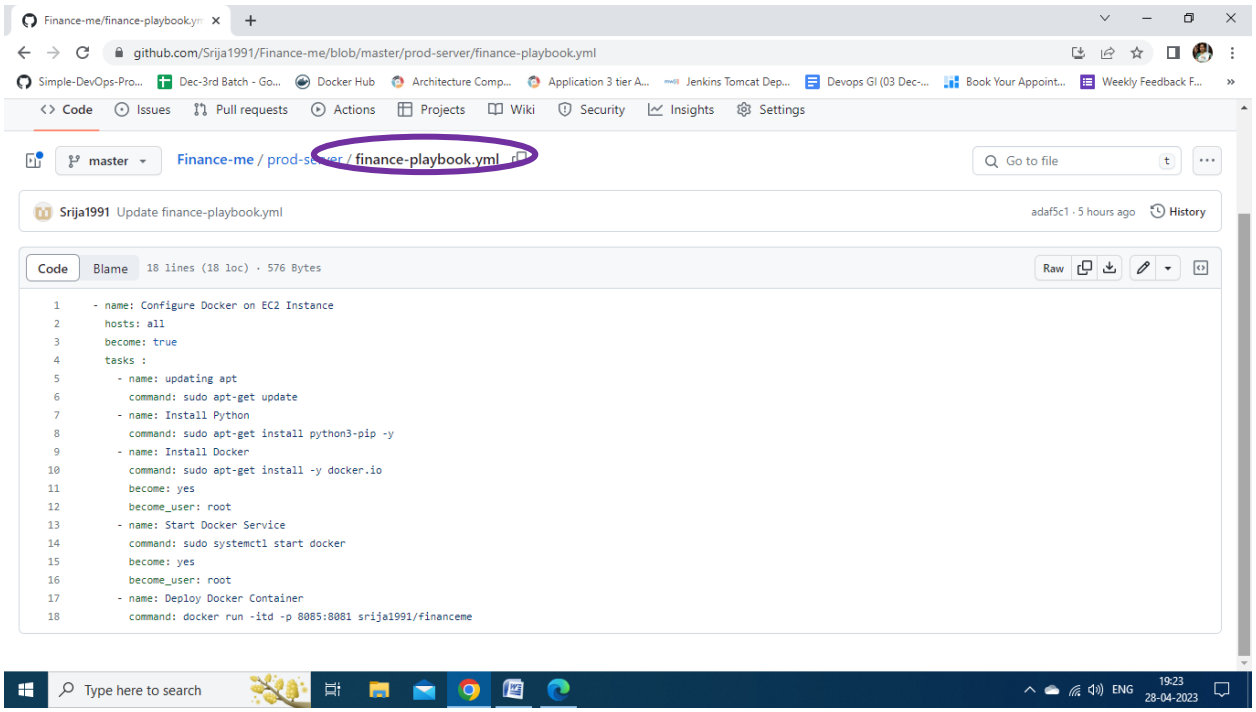
Here is the script for deployment onto production server.



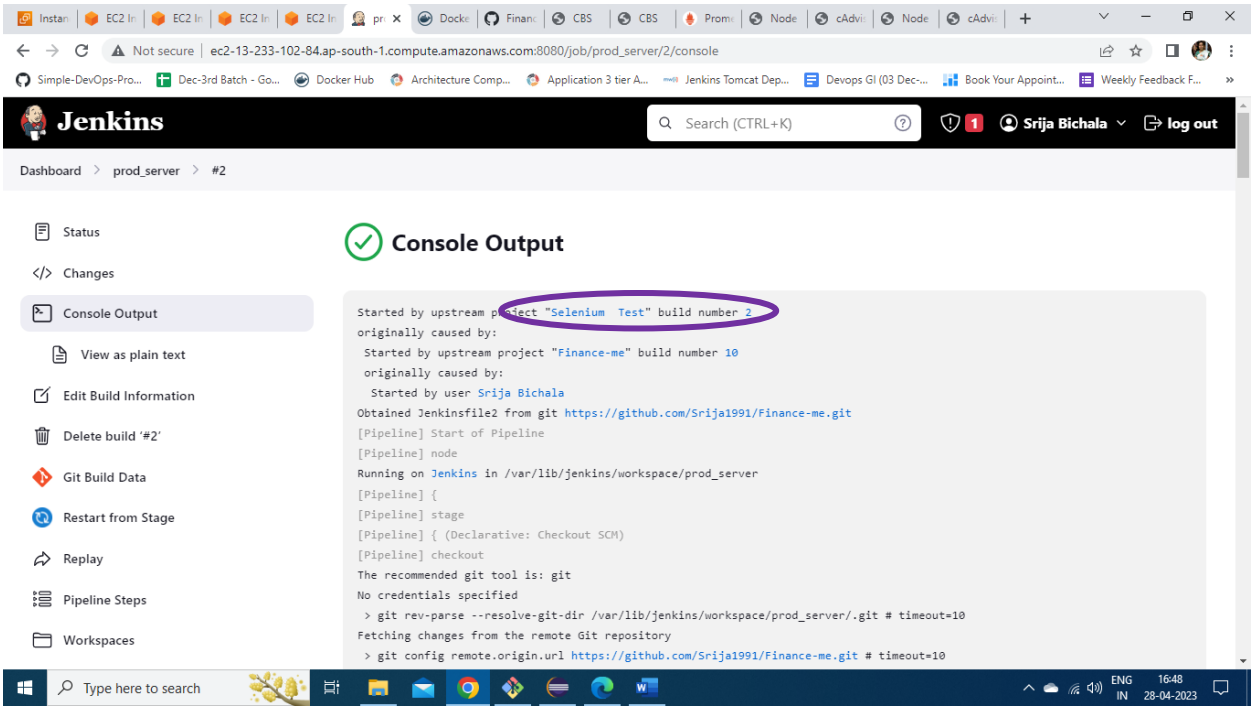
The prod-server folder on the repository is shown below. It has *main.tf* file used to create EC2 infrastructure, *finance-playbook*.yml which configures docker onto prod-server.

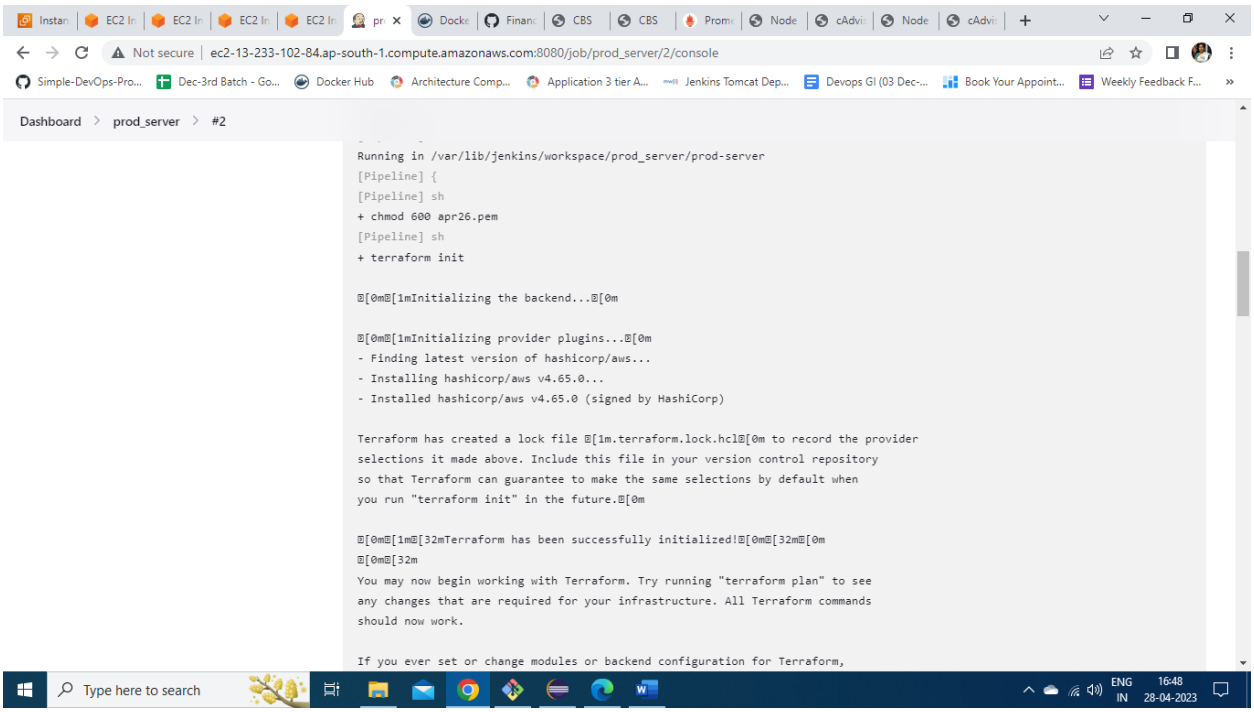


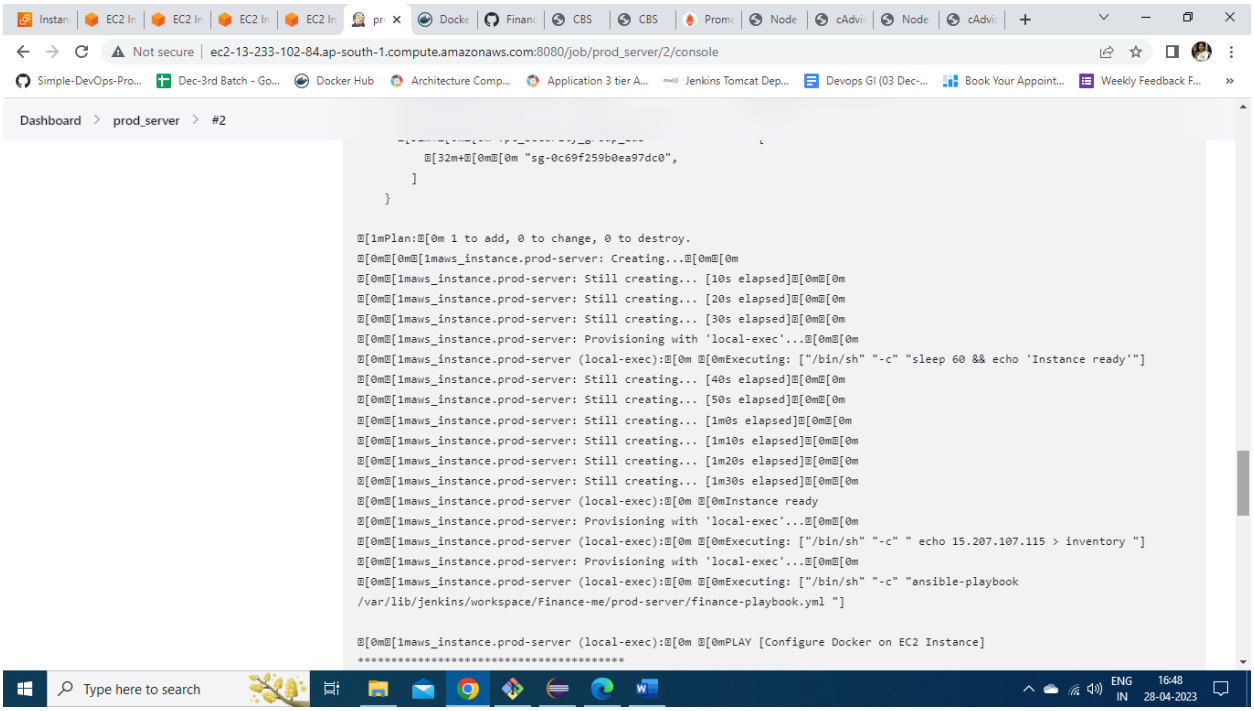

Finance-me/main.tf at master
github.com/Srija1991/Finance-me/blob/master/prod-server/main.tf
master
Finance-me / prod-server / main.tf
Srija1991 Update main.tf
2813784 · 3 hours ago
History
Code
Blame
25 lines (24 loc) · 710 Bytes
Raw
resource "aws_instance" "prod-server" {
ami = "ami-02eb7a4783e7e9317"
instance_type = "t2.micro"
key_name = "apr26"
vpc_security_group_ids= ["sg-0c69f259b0ea97dc0"]
tags = {
Name = "prod-server"
}
provisioner "local-exec" {
command = "sleep 60 && echo 'Instance ready'"
}
connection {
type = "ssh"
user = "ubuntu"
private_key = file("./apr26.pem")
host = self.public_ip
}
provisioner "local-exec" {
command = " echo ${aws_instance.prod-server.public_ip} > inventory "
}
provisioner "local-exec" {
command = "ansible-playbook /var/lib/jenkins/workspace/Finance-me/prod-server/finance-playbook.yml "
Type here to search
ENG
19:22
28-04-2023


Finance-me/finance-playbook.yml
github.com/Srija1991/Finance-me/blob/master/prod-server/finance-playbook.yml
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
master
Finance-me / prod-server / finance-playbook.yml
Go to file
Srija1991 Update finance-playbook.yml
adaf5c1 · 5 hours ago
History
Code
Blame
18 lines (18 loc) · 576 Bytes
Raw
- name: Configure Docker on EC2 Instance
hosts: all
become: true
tasks :
- name: updating apt
command: sudo apt-get update
- name: Install Python
command: sudo apt-get install python3-pip -y
- name: Install Docker
command: sudo apt-get install -y docker.io
become: yes
become_user: root
- name: Start Docker Service
command: sudo systemctl start docker
become: yes
become_user: root
- name: Deploy Docker Container
command: docker run -itd -p 8085:8081 srija1991/financeme
Type here to search
ENG
19:23
28-04-2023

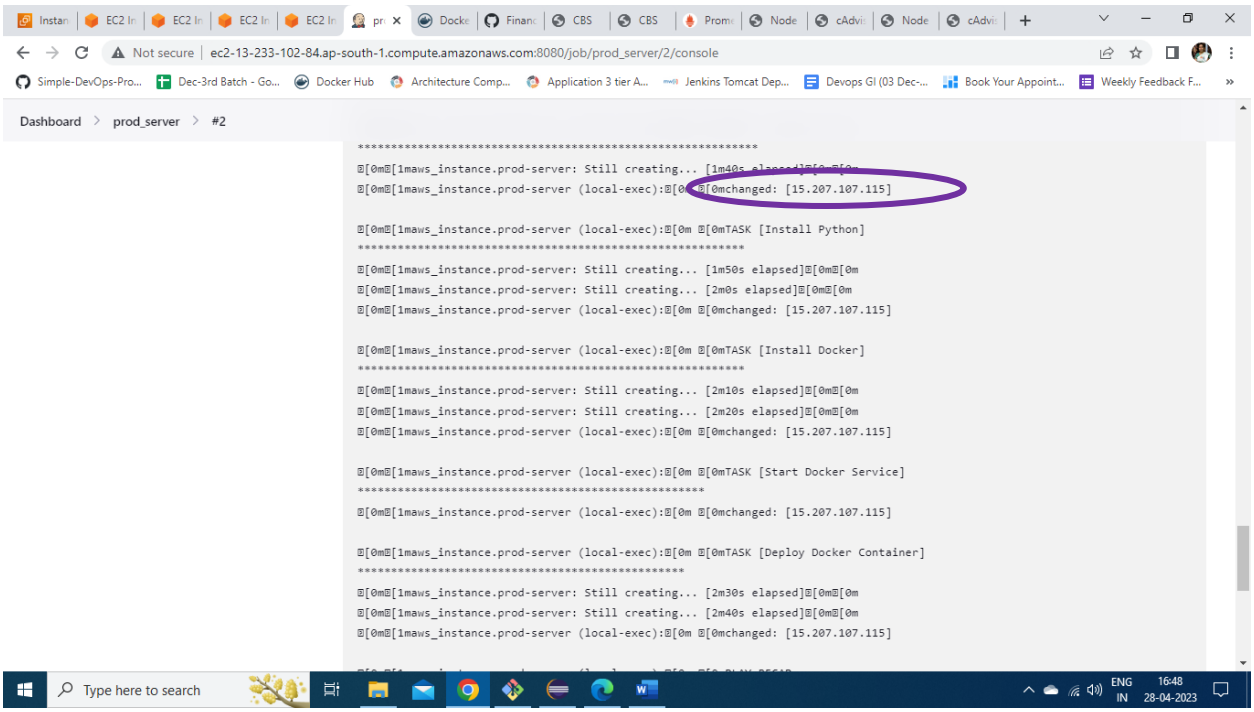
Here is the console output for prod-server and application successfully deployed.



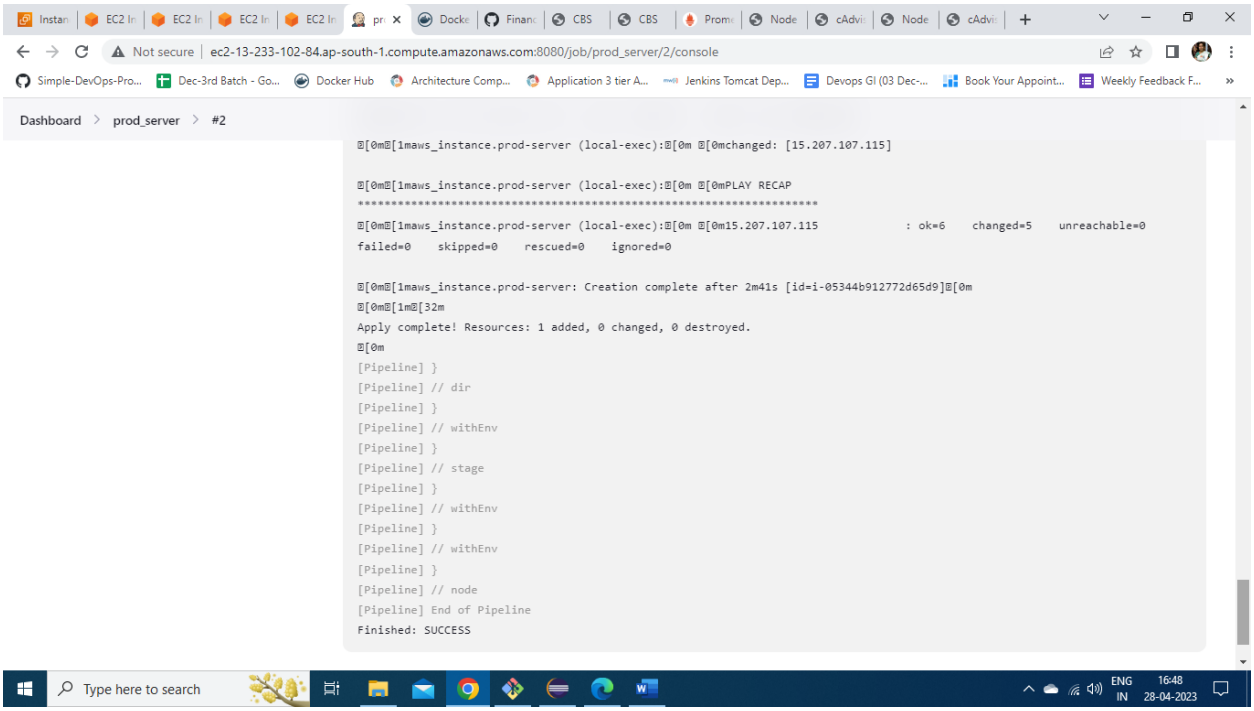




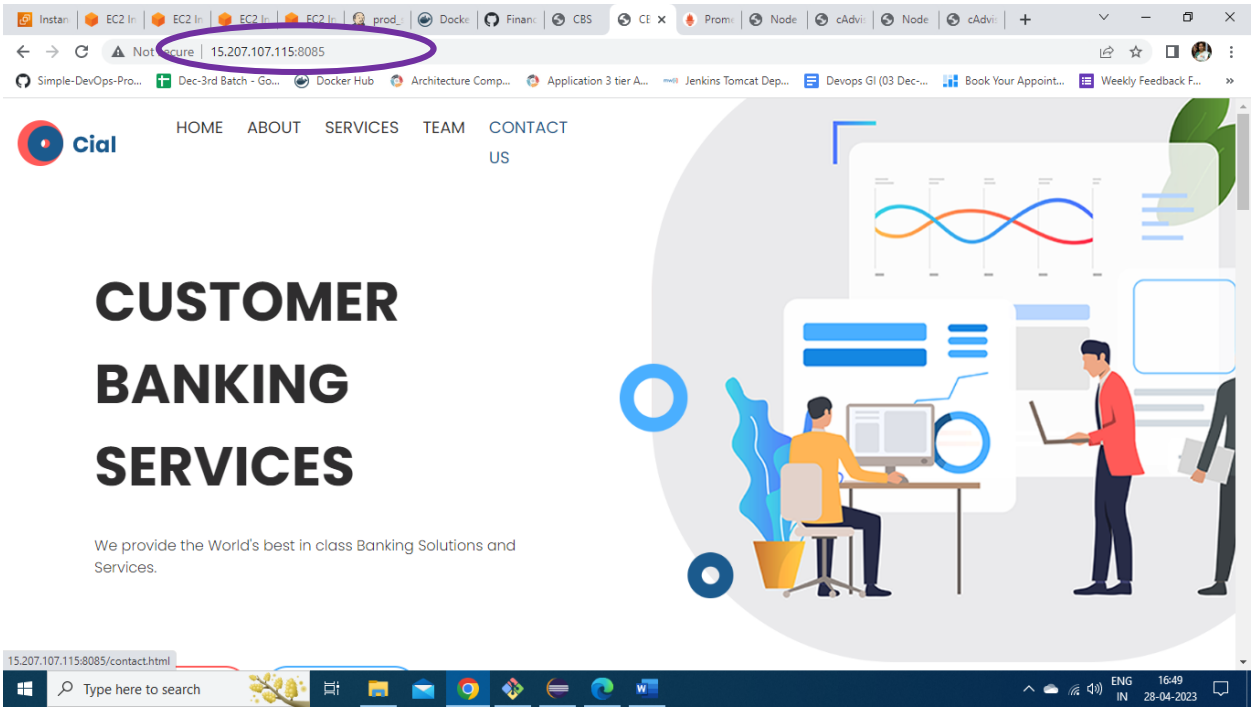

```
        ␛[32m+␛[0m␛[0m "sg-0c69f259b0ea97dc0",
      ]
  }

␛[1mPlan:␛[0m 1 to add, 0 to change, 0 to destroy.
␛[0m␛[0m␛[1maws_instance.prod-server: Creating...␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [10s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [20s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [30s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Provisioning with 'local-exec'...␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mExecuting: ["/bin/sh" "-c" "sleep 60 && echo 'Instance ready'"]
␛[0m␛[1maws_instance.prod-server: Still creating... [40s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [50s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [1m0s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [1m10s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [1m20s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [1m30s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mInstance ready
␛[0m␛[1maws_instance.prod-server: Provisioning with 'local-exec'...␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mExecuting: ["/bin/sh" "-c" " echo 15.207.107.115 > inventory "]
␛[0m␛[1maws_instance.prod-server: Provisioning with 'local-exec'...␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mExecuting: ["/bin/sh" "-c" "ansible-playbook
/var/lib/jenkins/workspace/Finance-me/prod-server/finance-playbook.yml "]

␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mPLAY [Configure Docker on EC2 Instance]
*************************************
```



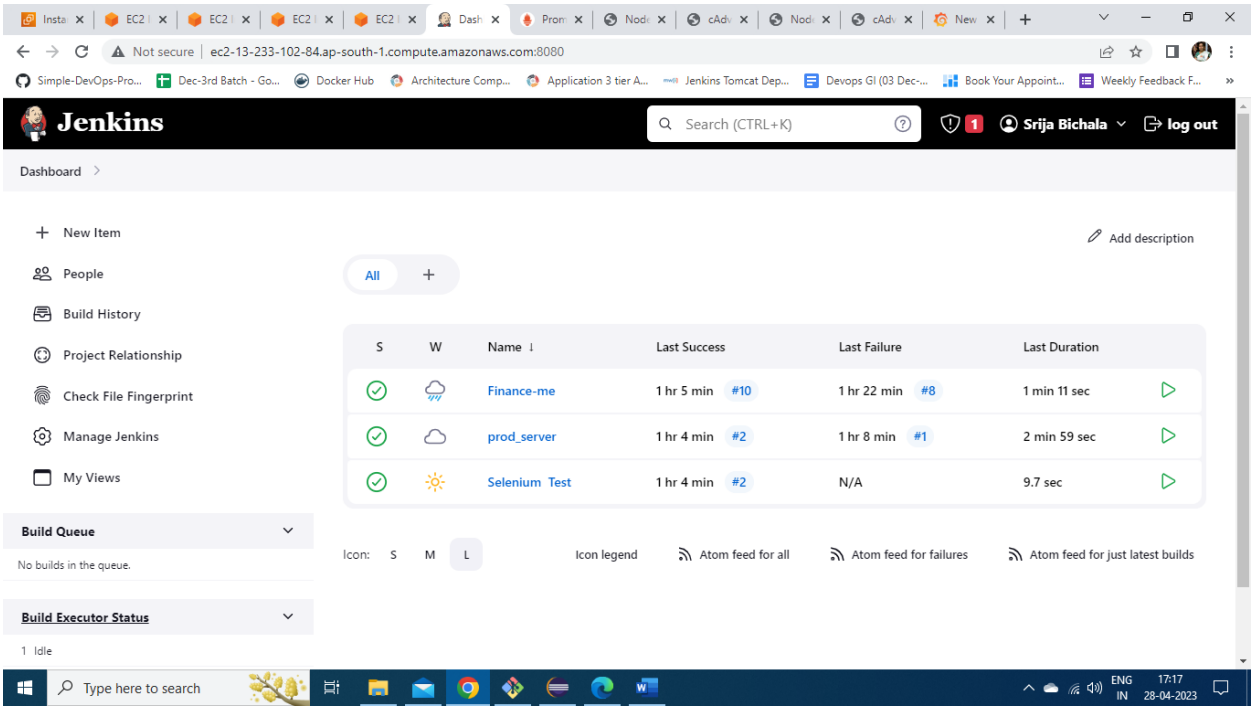

```
*********************************************************
␛[0m␛[1maws_instance.prod-server: Still creating... [1m40s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mchanged: [15.207.107.115]

␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mTASK [Install Python]
*********************************************************
␛[0m␛[1maws_instance.prod-server: Still creating... [1m50s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [2m0s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mchanged: [15.207.107.115]

␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mTASK [Install Docker]
*********************************************************
␛[0m␛[1maws_instance.prod-server: Still creating... [2m10s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [2m20s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mchanged: [15.207.107.115]

␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mTASK [Start Docker Service]
*********************************************************
␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mchanged: [15.207.107.115]

␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mTASK [Deploy Docker Container]
*********************************************************
␛[0m␛[1maws_instance.prod-server: Still creating... [2m30s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server: Still creating... [2m40s elapsed]␛[0m␛[0m
␛[0m␛[1maws_instance.prod-server (local-exec):␛[0m ␛[0mchanged: [15.207.107.115]
```



Production server IP: 15.207.107.115:8085.

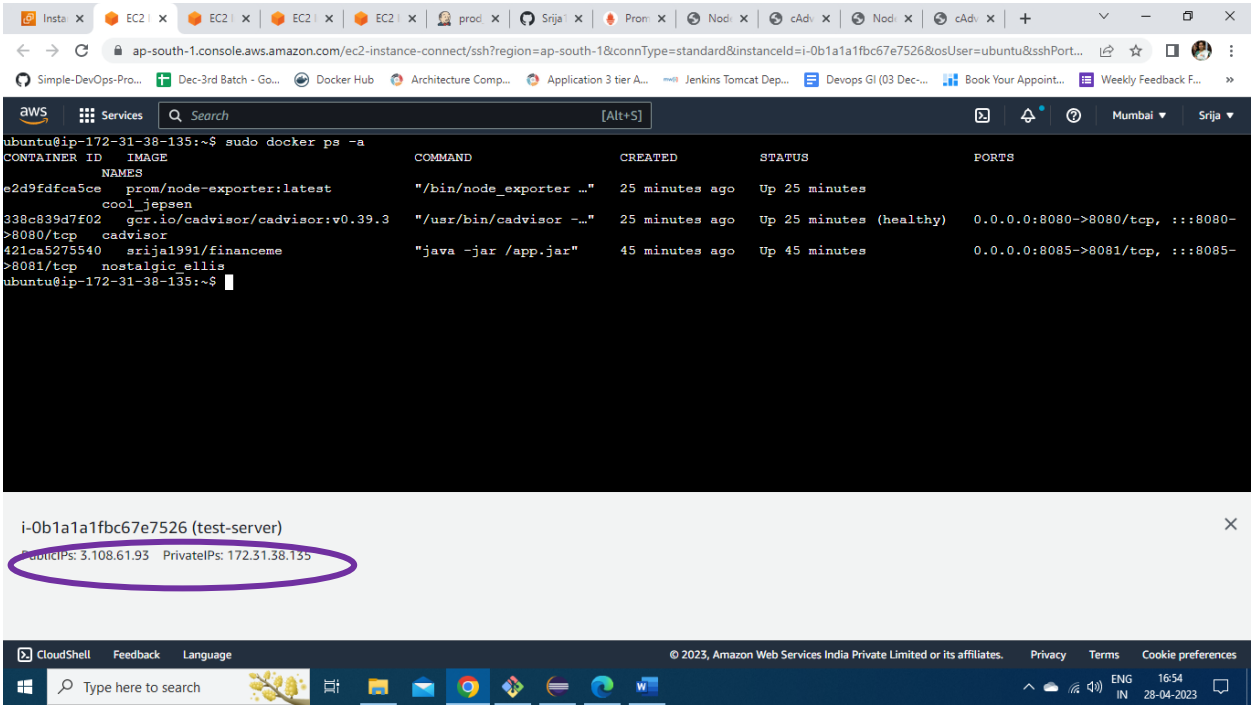


The three jobs are successfully build and application is first deployed on test-server after selenium test it is deployed onto production server.

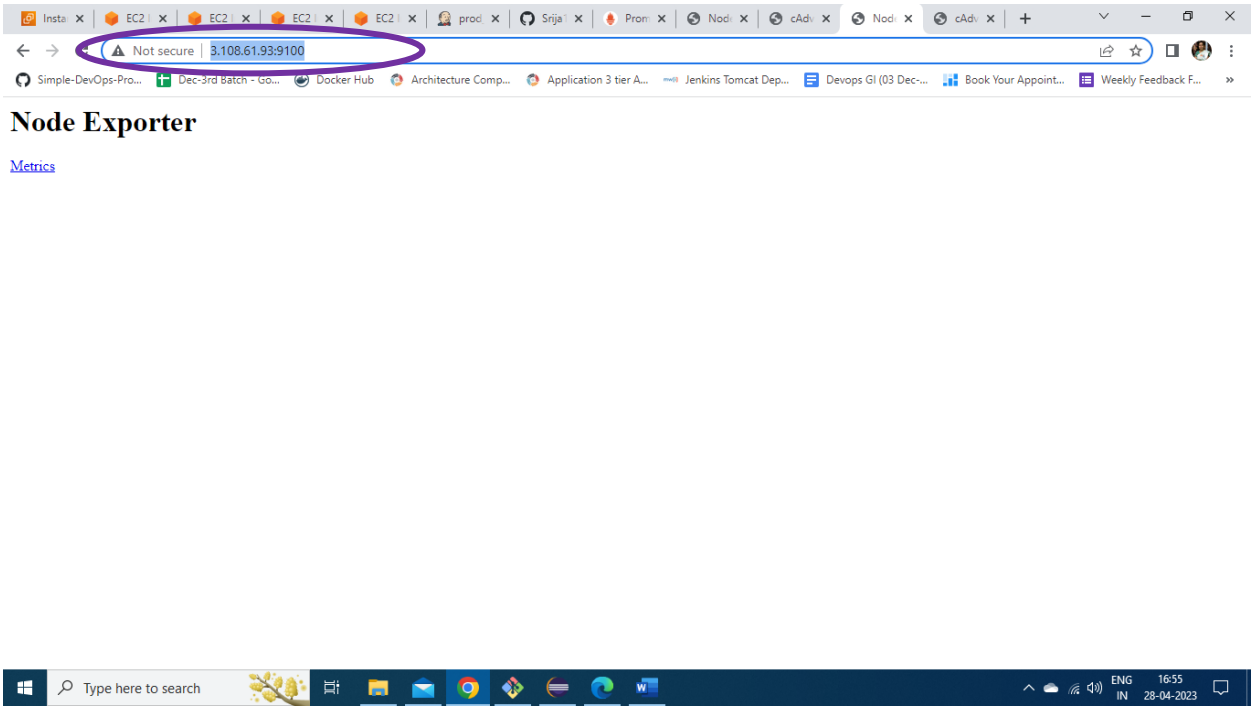


Now cAdvisor and Node-exporter are run on the test-server and prod-server to monitor using Prometheus and Grafana tools.

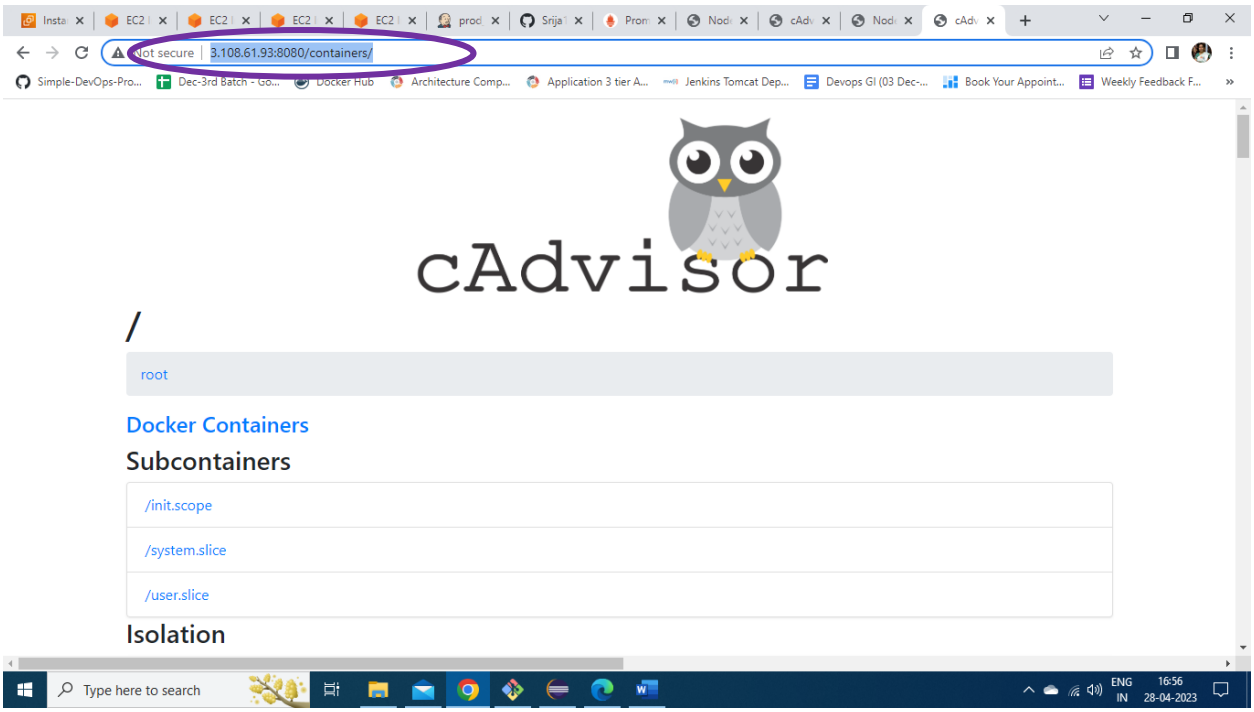
Test-server



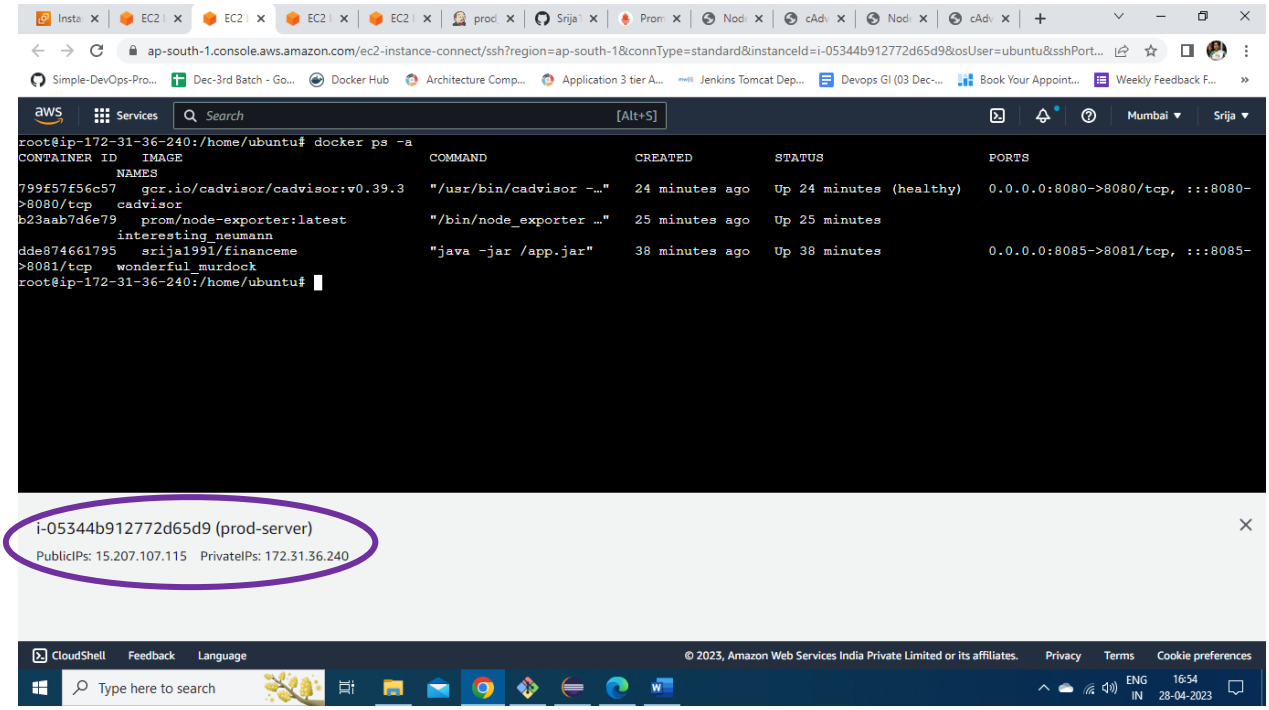
Node-exporter on test-server: port number : 3.108.61.93:9100



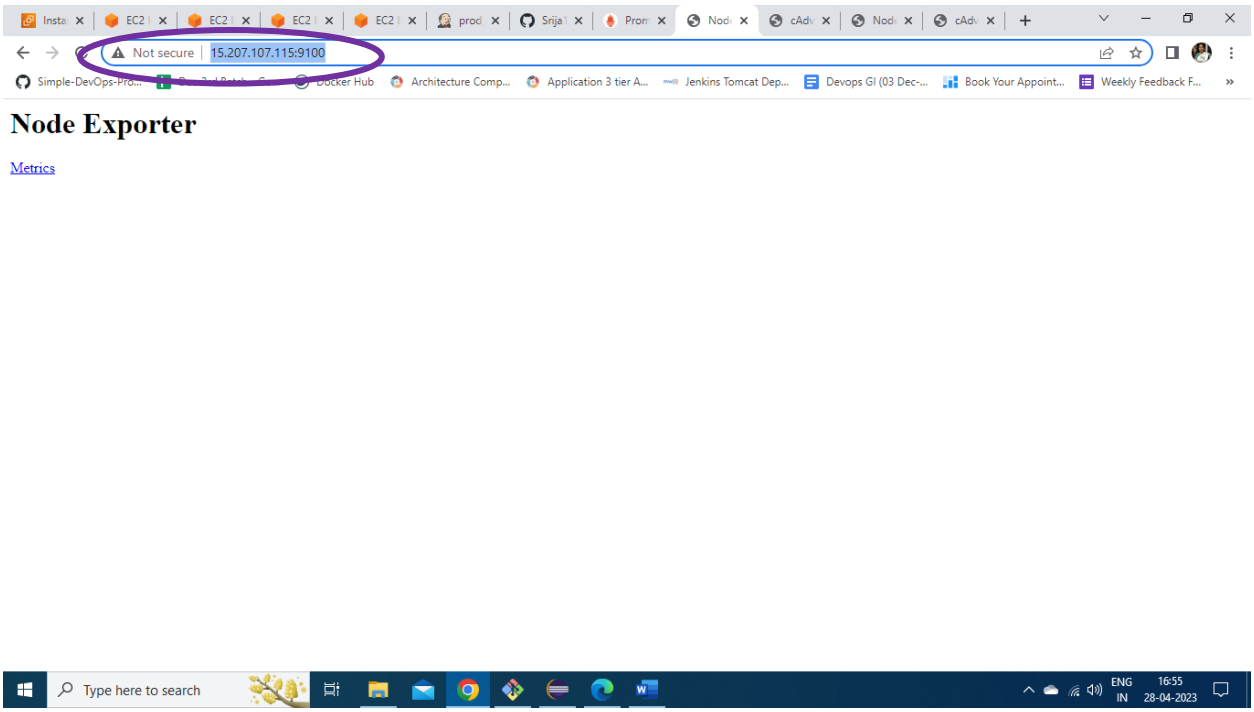
cAdvisor on test-server : port number: 3.108.61.93:8080



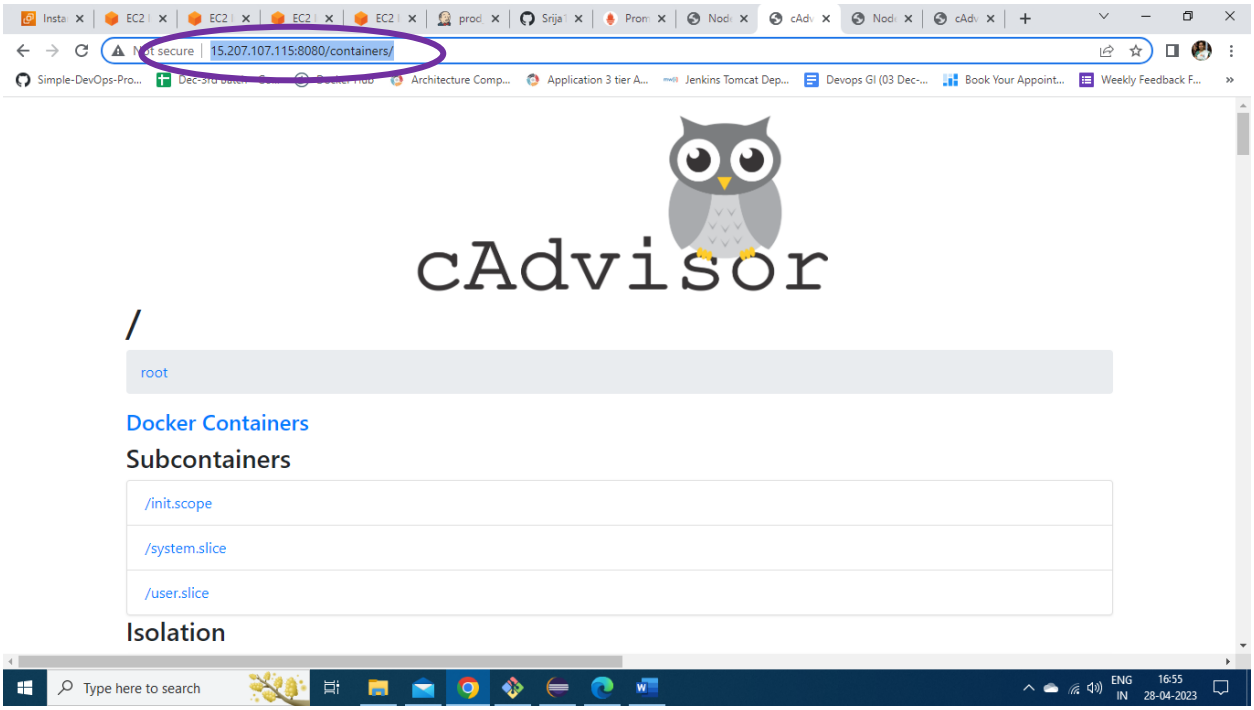
Cadvisor and Node-exporter on production server.



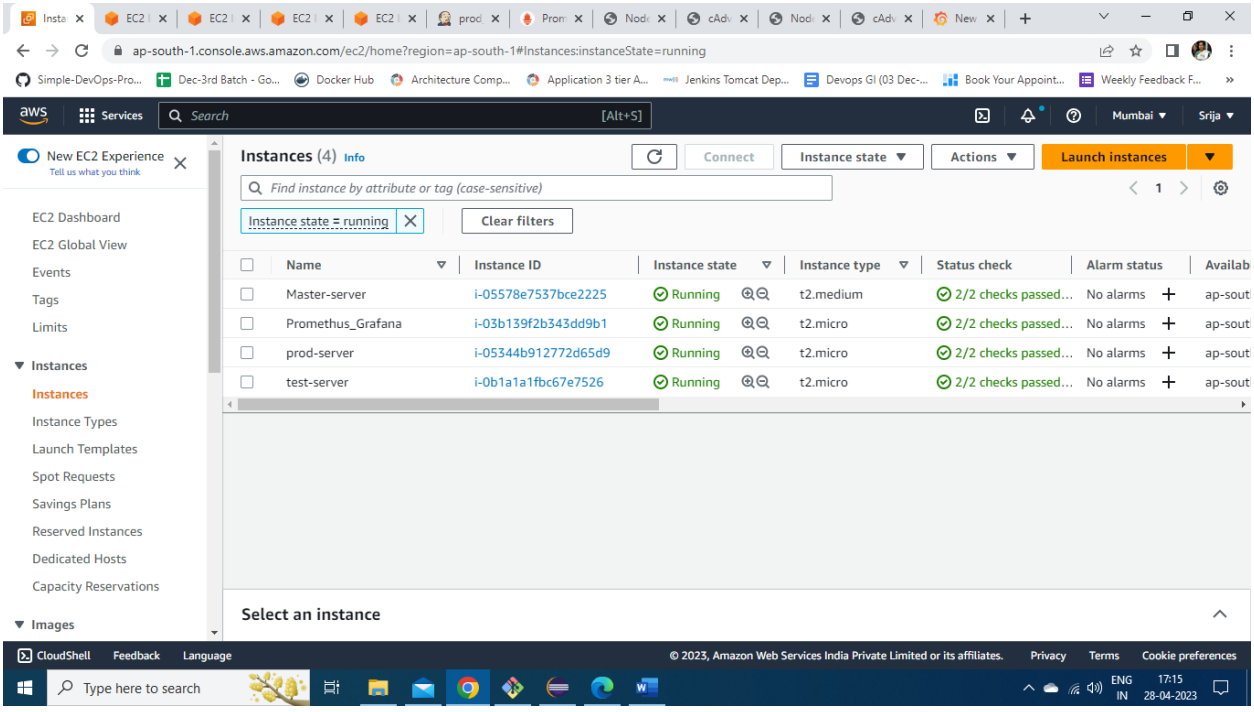
The Node-exporter on the production-server configured on URL: 15.207.107.115:9100.



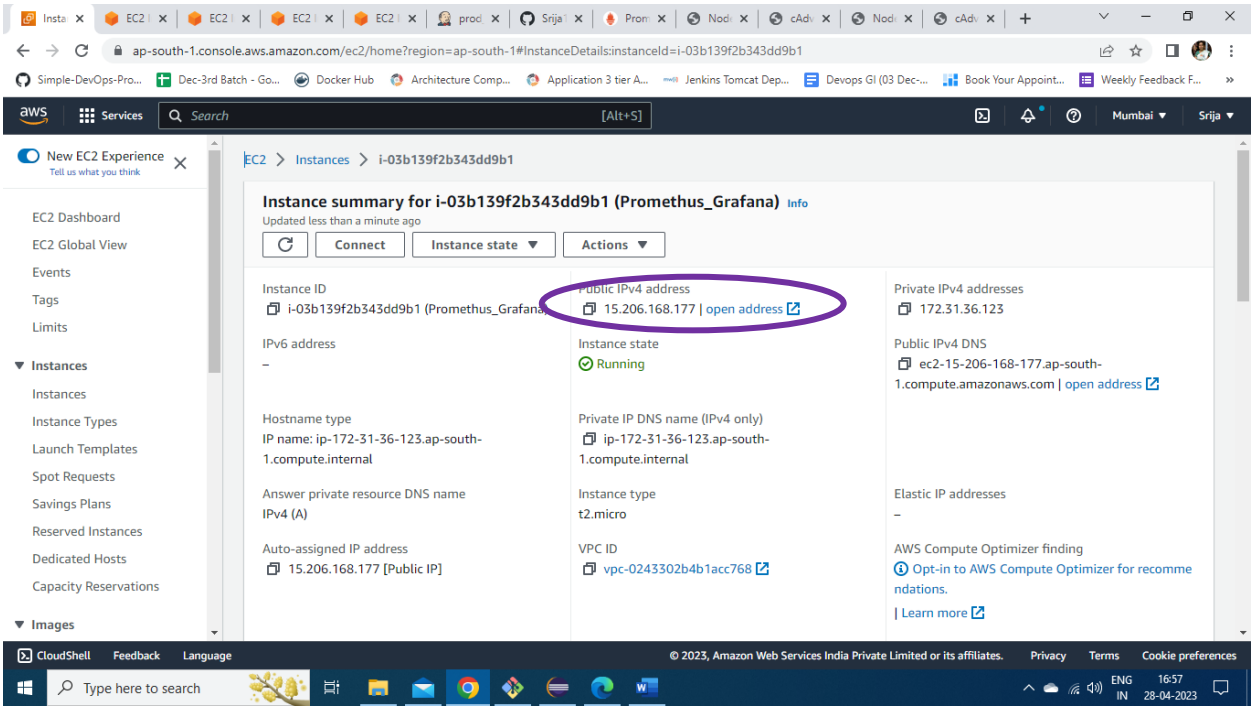
The cAdvisor on production server at URL: 15.207.107.115:8080.



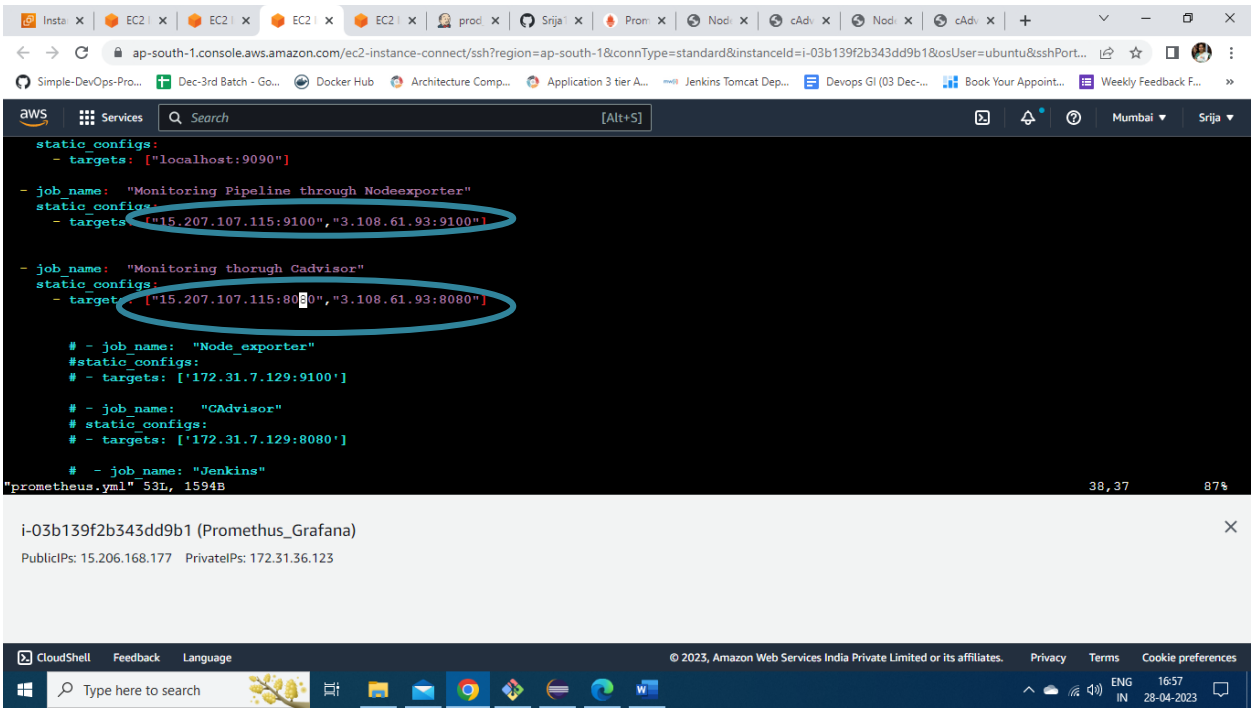
The snapshot of the servers .



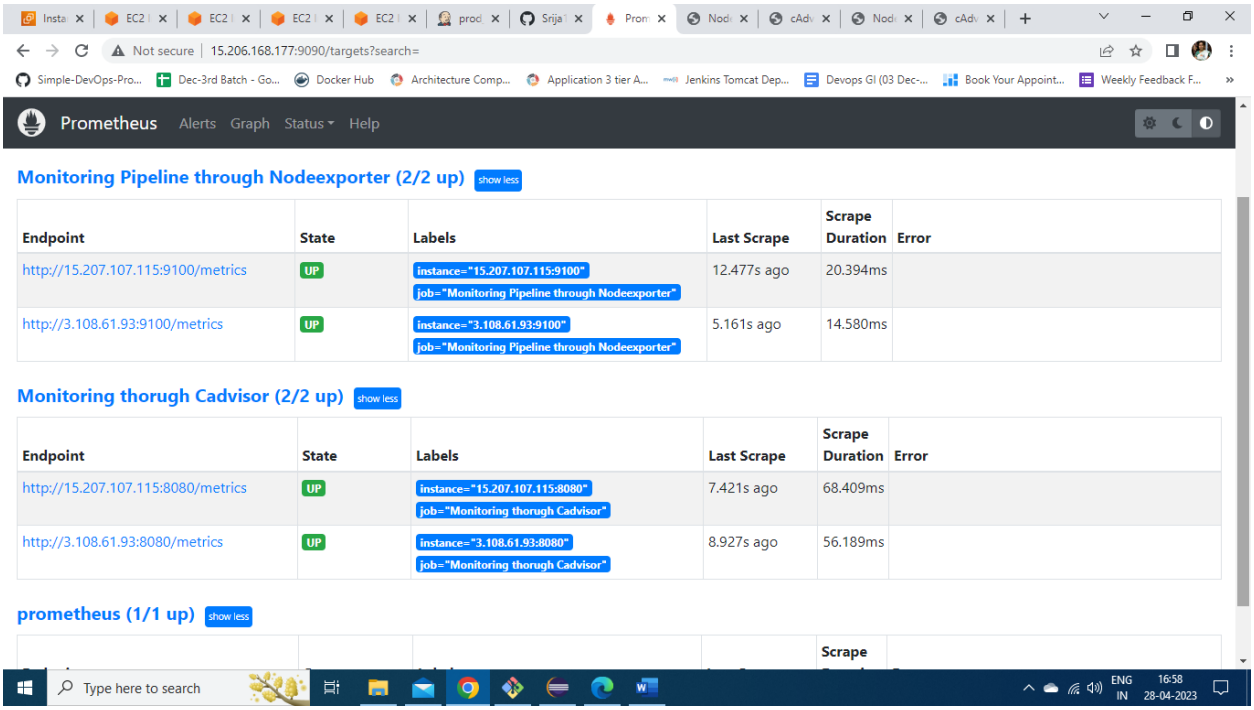
Here is the Monitoring server, public IP:15.206.168.177



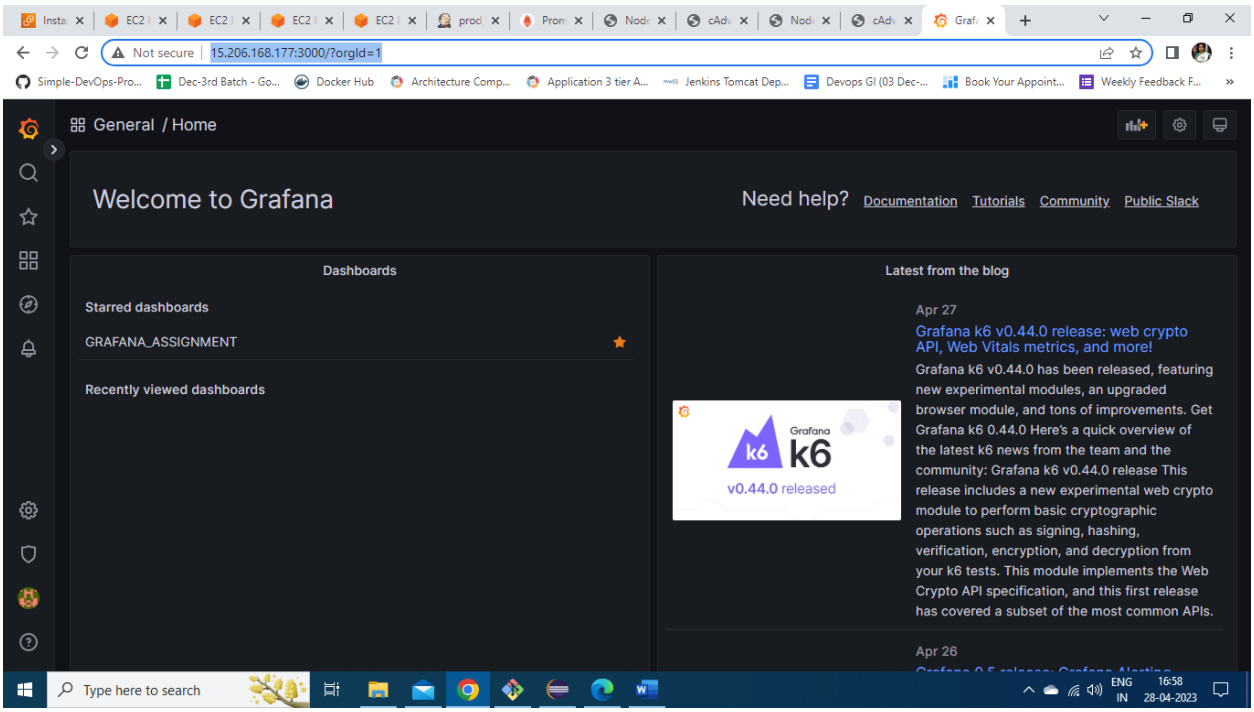
Prometheus is configured to add the metrics on to the monitoring server.



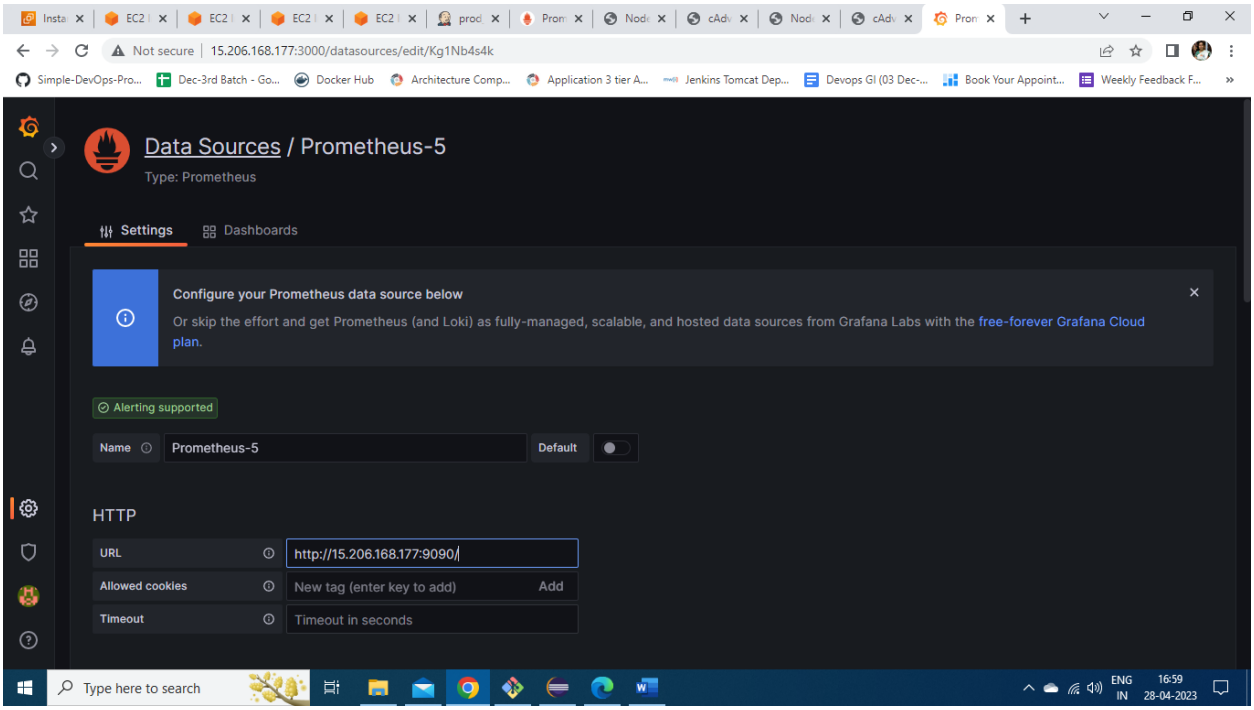
Prometheus GUI is accessed from URL: 15.206.168.177:9090 and targets are added.

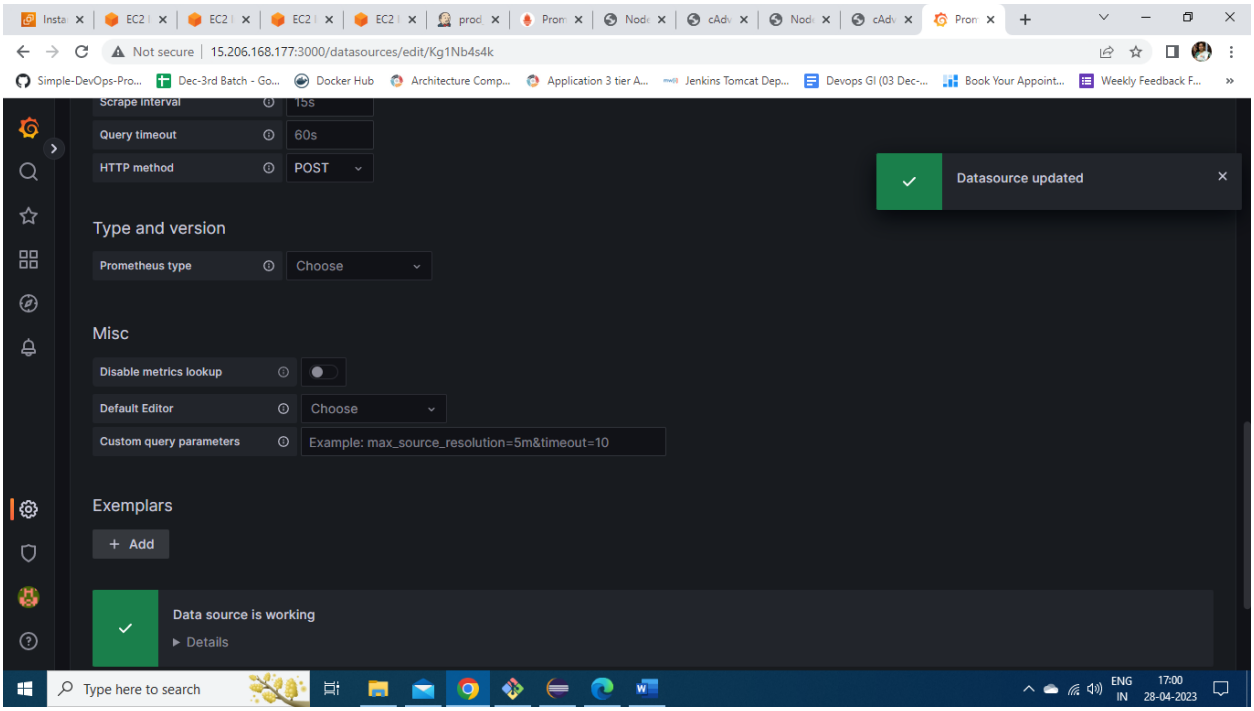


The Grafana is accessed on URL: 15.206.168.177:3000

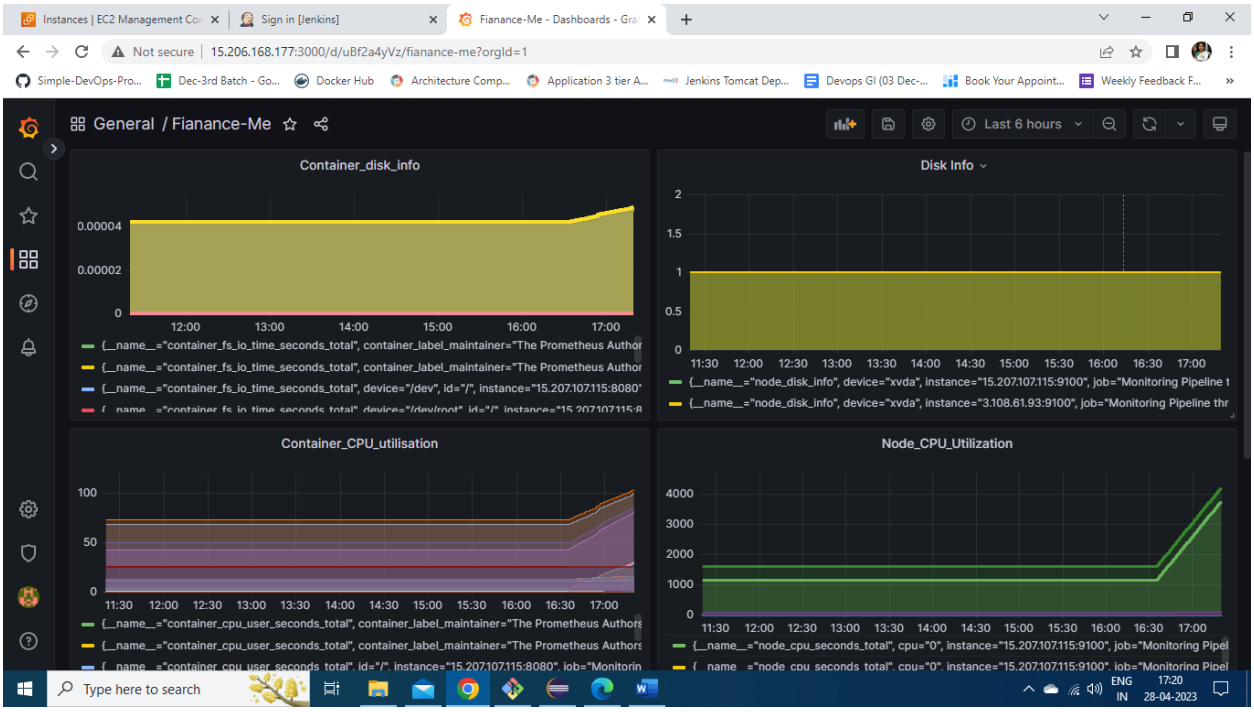


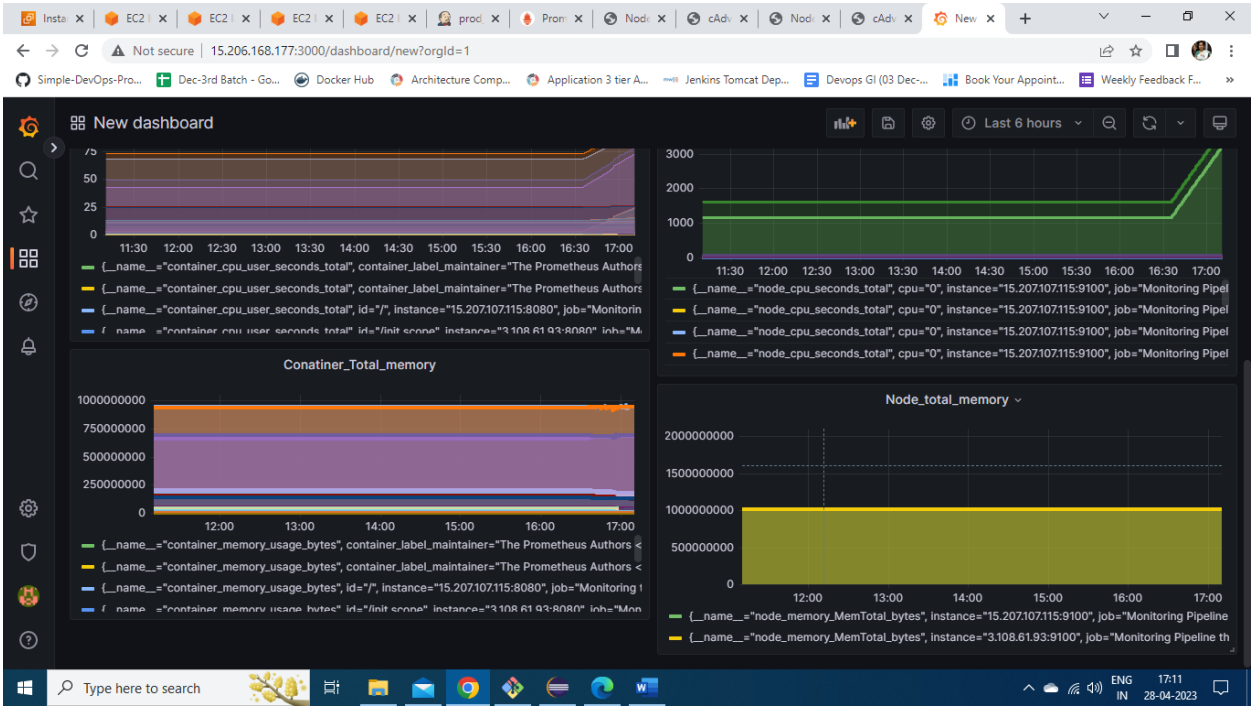
The new data source is added as Prometheus 5.





The metrics Disk utilization of node and container, CPU utilization of Node and Container and Container and node memory of both test server and production server are observed.





THANK YOU