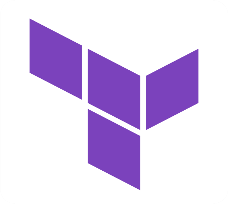
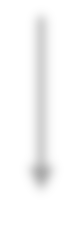
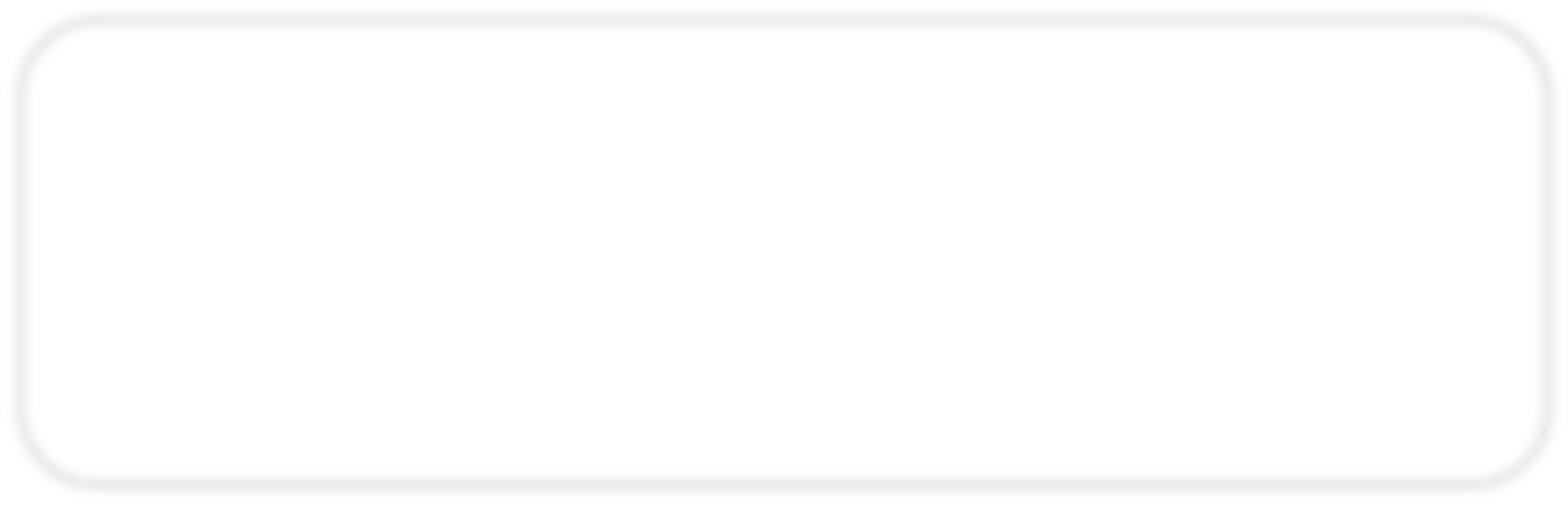
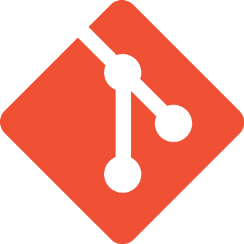
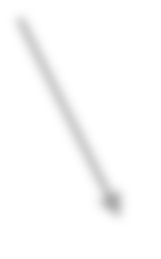
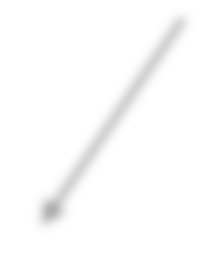
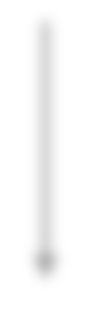
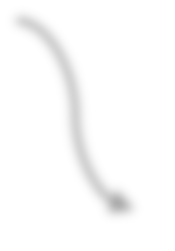
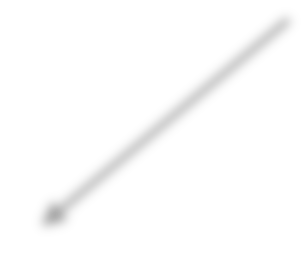
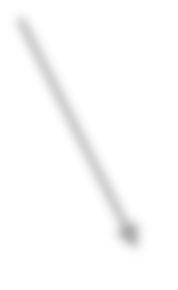
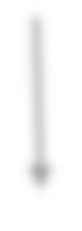
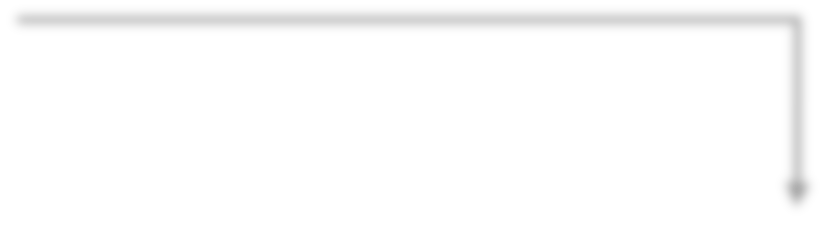
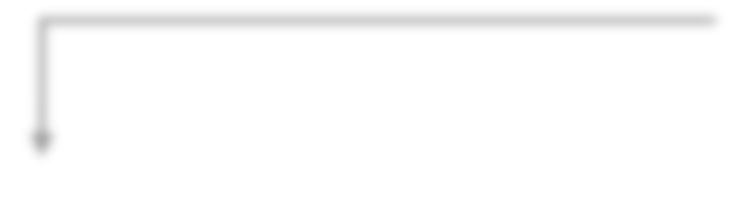
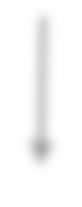
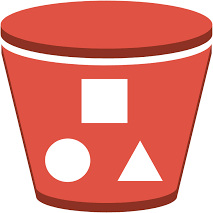
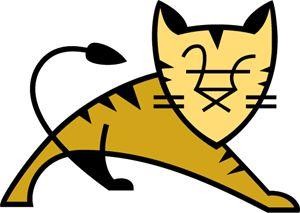
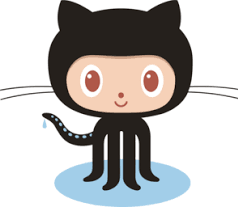
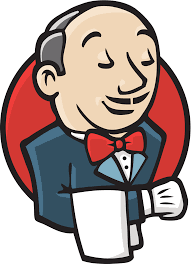
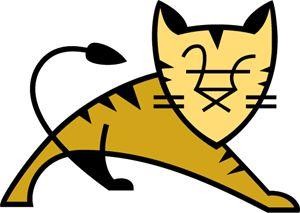
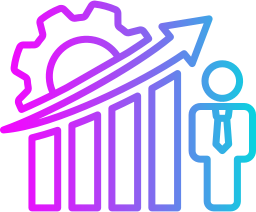
# PROJECT

Netflix Deployment Automation with Terraform, Jenkins and Ansible

**TOOL USE**

* GIT: A version control system for managing source code.
* GITHUB: A platform for hosting and managing Git repositories.
* ANSIBLE: A tool for automating configuration management and deployment.
* JENKINS: A CI/CD tool for integrating and automating workflows.
* MAVEN: A build automation tool for compiling and packaging source code.
* S3 BUCKET: A storage service for application artifacts and data.
* TOMCAT: A web server for deploying Java applications.
* TERRAFORM: Infrastructure as Code (IaC) tool for provisioning AWS resources
* PROMETHEUS & GRAFANA: Tools for monitoring and visualizing system metrics.





**Dev Test**

**Ansible**

**Monitoring**



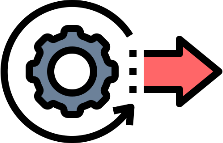
**Tomcat**

**GitHub**

**Jenkins**

**Prometheus Grafana**



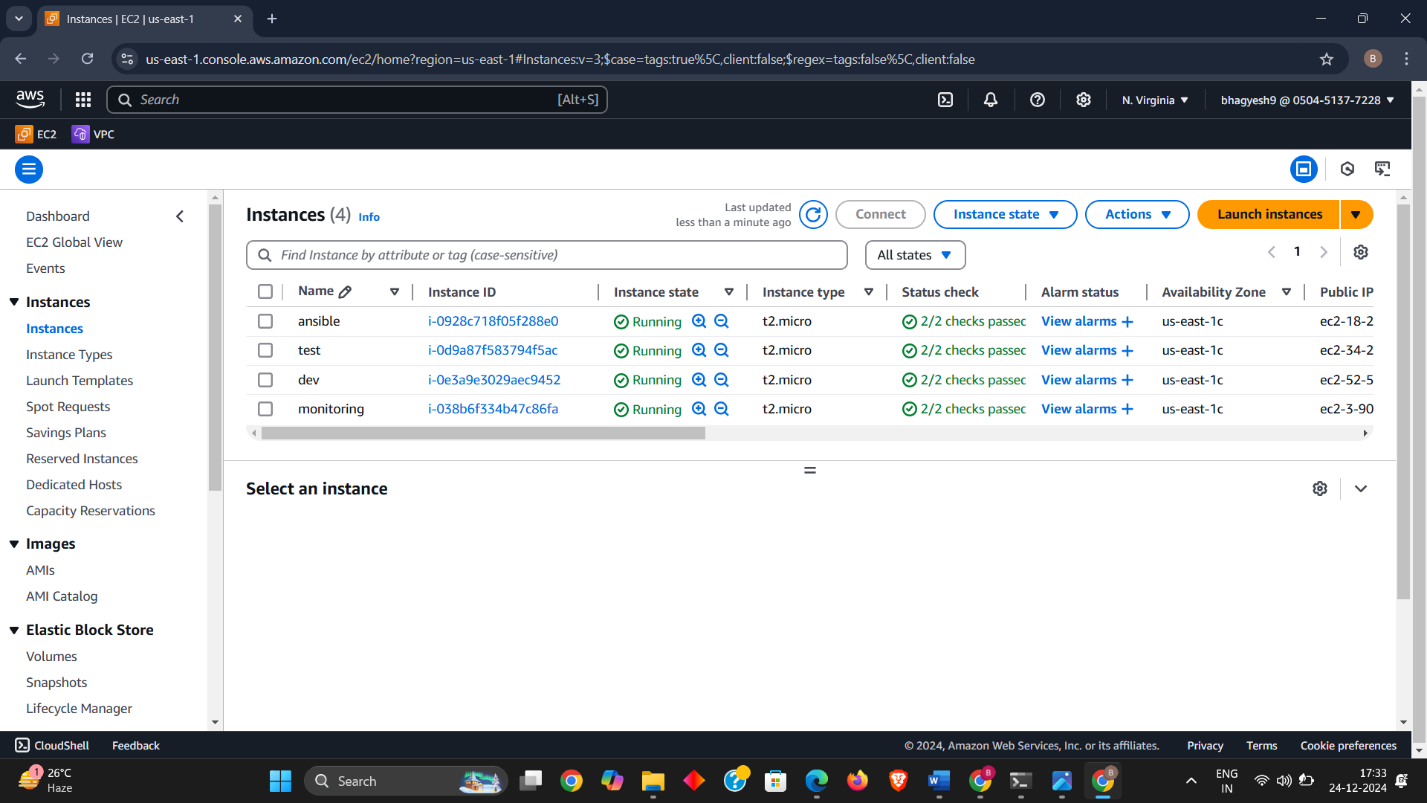
**Git Maven S3 Bucket Tomcat (Deploy)**

**OUTPUT**

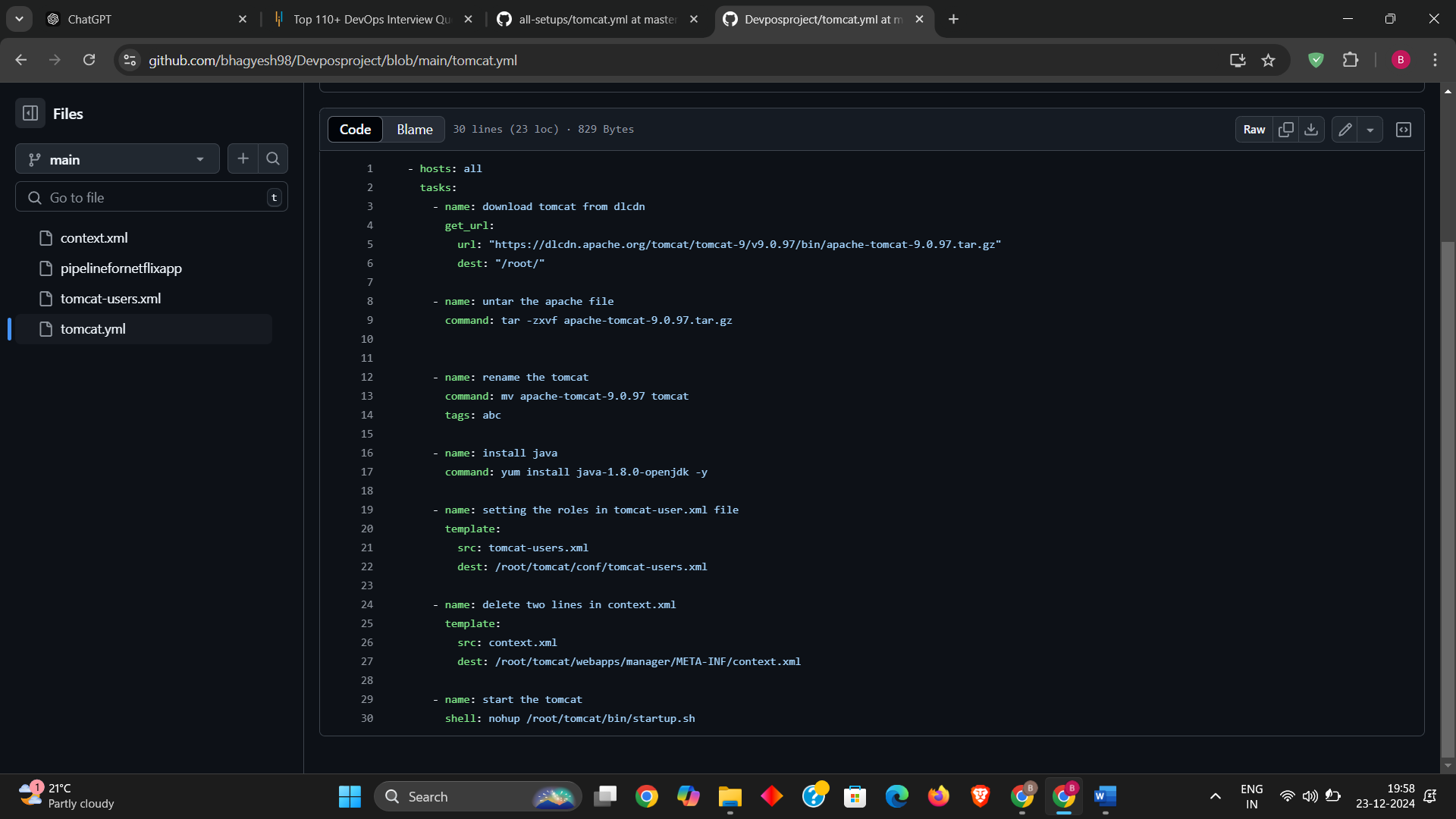
### STEP-1: Create four EC2 instances using Terraform, push the code to

### GitHub, and use HCP to run the code and create the infrastructure.

1. Ansible (also used for Jenkins)
2. Dev (worker node)
3. Test (worker node)
4. Monitoring

**Note** - Install ansible on ansible server and setup the configuration with worker nodes (dev and test) and install Jenkins on ansible server

**Install tomcat using ansible playbook on ansible server and move to all work nodes –**



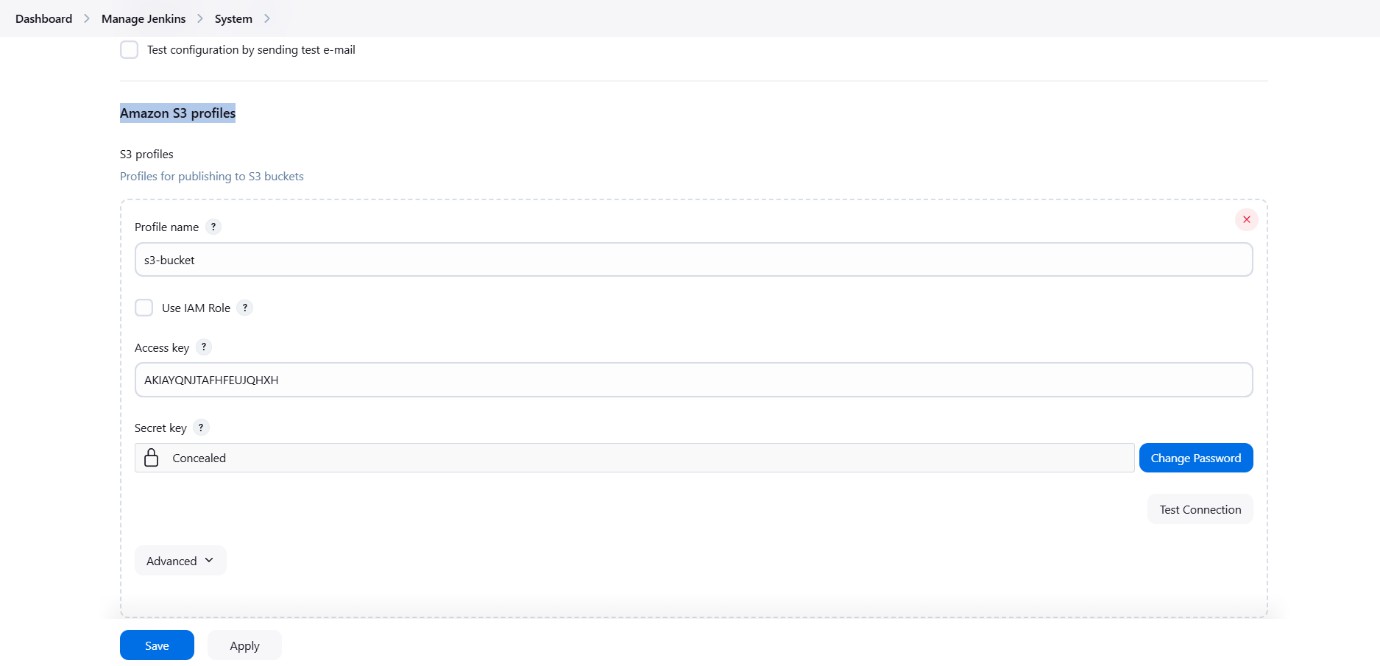
### STEP-2: PUSHING THE CODE FROM GIT TO GITHUB

The code which is used for our project it should be placed at github. Repository url: <https://github.com/bhagyesh98/jenkins-java-project.git>

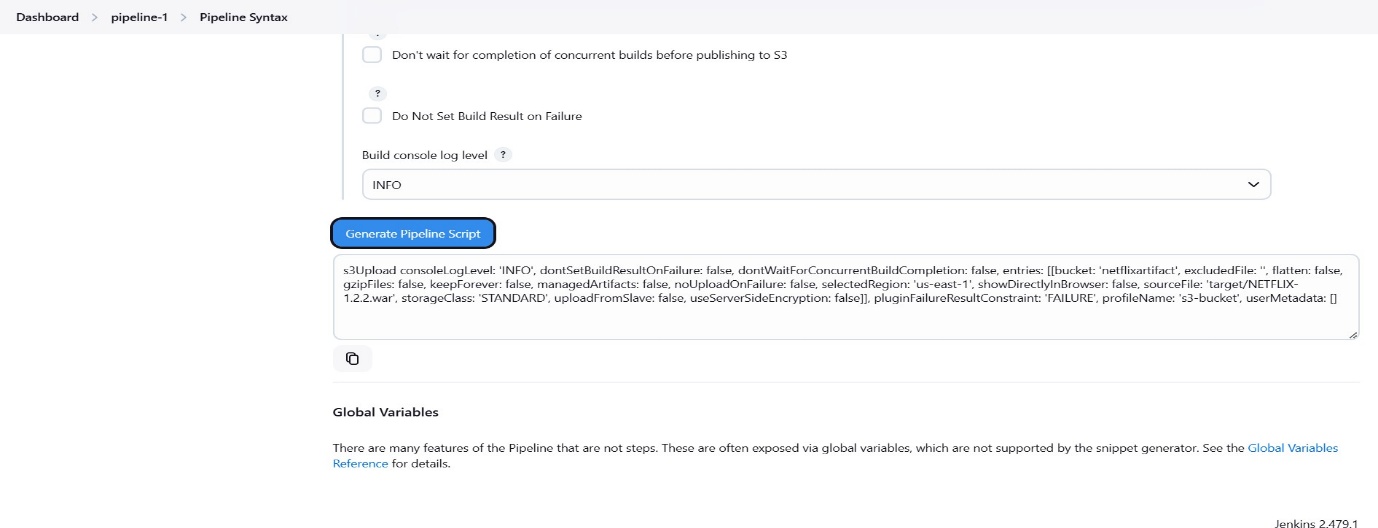
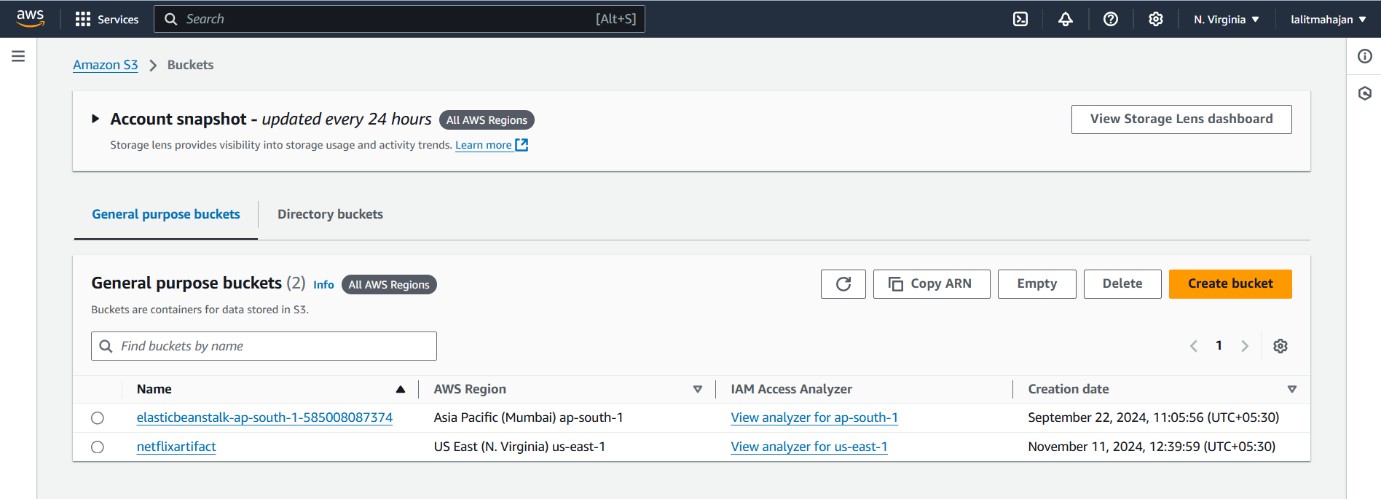
### STEP-3: STORE ARTIFACT ON S3

Create S3 bucket in AWS. INTEGRTAING S3 WITH JENKINS:

1. Installing s3 publish plugin.
2. Configure for s3profile

Manage Jenkins – System – S3 Profile – Access Key and Secret Key

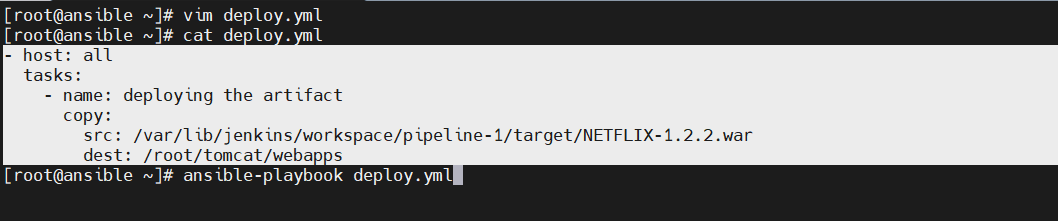
### 3. PERFORM CI ON JENKINS FOR S3 UPLOAD

Create a pipeline and write the script as stages code – build – test – artifact – s3 upload.

**STEP-5: PATH FOR WAR FILE**

We have to perform these at ansible server CD /var/lib/Jenkins/workspace/pipeline Cd /target ll (here we can see war fie) Give pwd (present working directory) /var/lib/Jenkins/workspace/pipeline/target

**STEP-7: INTEGRTAING ANSIBE WITH JENKINS**

1. install ansible plugin
2. manage Jenkins - tools - ansible - name: ansible & Path /usr/bin - save (NOTE: /usr/bin is a folder where all Linux commands will store)
3. Write Ansible Playbook
4. Move Playbook to ansible home path

-mv deploy.yml /etc/ansible

1. To generated pipeline syntax for deployment –

Name - ansible - Playbook file path in workspace: /etc/ansible/playbook.yml –

Inventory file path in workspace: /etc/ansible/hosts –

SSH CREDS: user name – root - password – type ansible password while you created for config worker to ansible Disable the host SSH key check

Complete declarative pipeline –

node {

stage('code') {

git ' https://github.com/bhagyesh98/jenkins-java-project.git '

}

stage('build') {

sh 'mvn compile'

}

stage('test') { sh 'mvn test'

}

stage('artifact') {

sh 'mvn package'

}

stage('s3') {

s3Upload consoleLogLevel: 'INFO', dontSetBuildResultOnFailure: false, dontWaitForConcurrentBuildCompletion: false, entries: [[bucket: 'netflixartifact', excludedFile: '', flatten: false, gzipFiles: false, keepForever: false, managedArtifacts: false, noUploadOnFailure: false, selectedRegion: 'us-east-1', showDirectlyInBrowser: false, sourceFile: 'target/NETFLIX-1.2.2.war', storageClass: 'STANDARD', uploadFromSlave: false, useServerSideEncryption: false]], pluginFailureResultConstraint: 'FAILURE', profileName: 's3-bucket', userMetadata: []

}

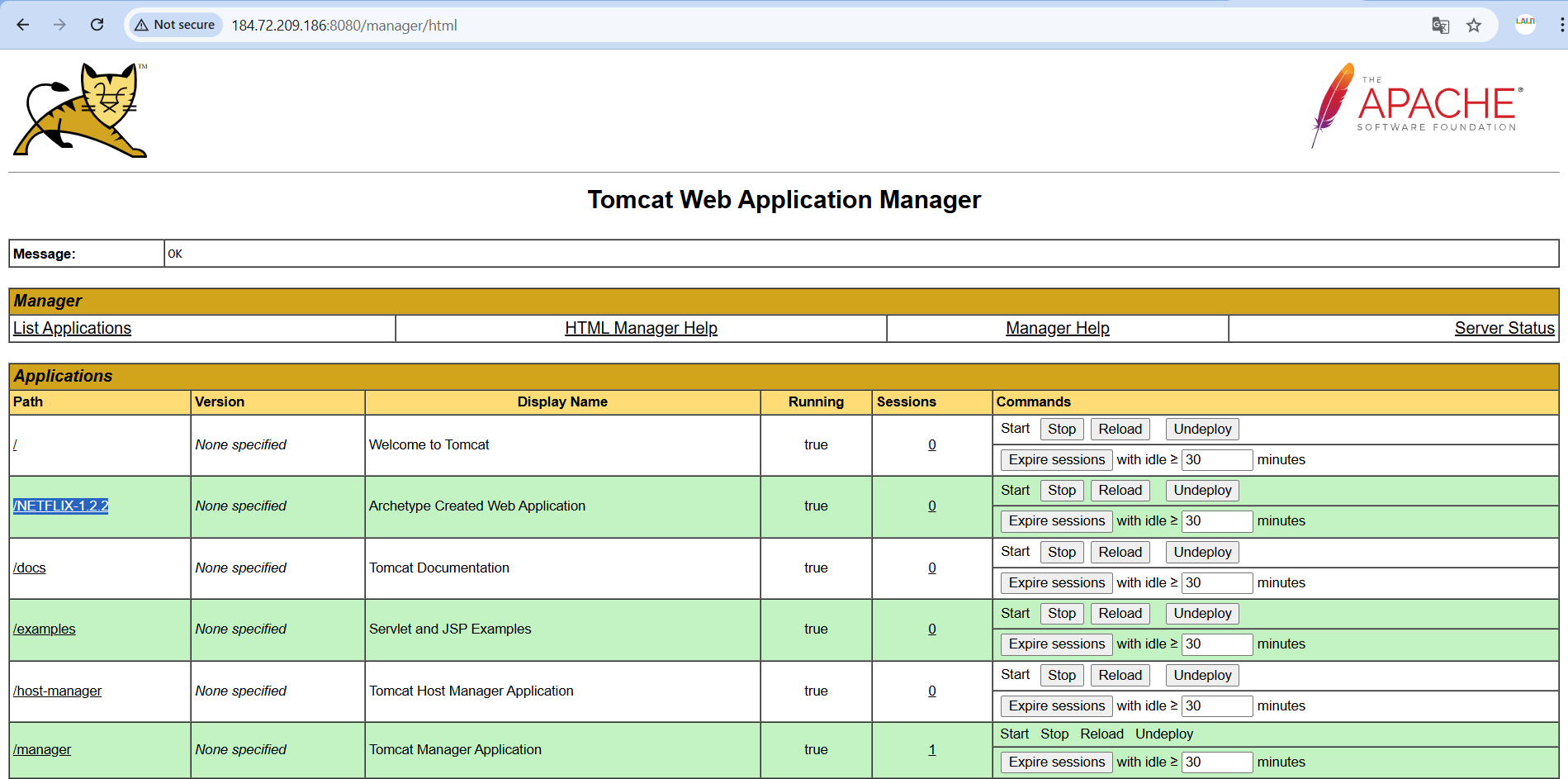
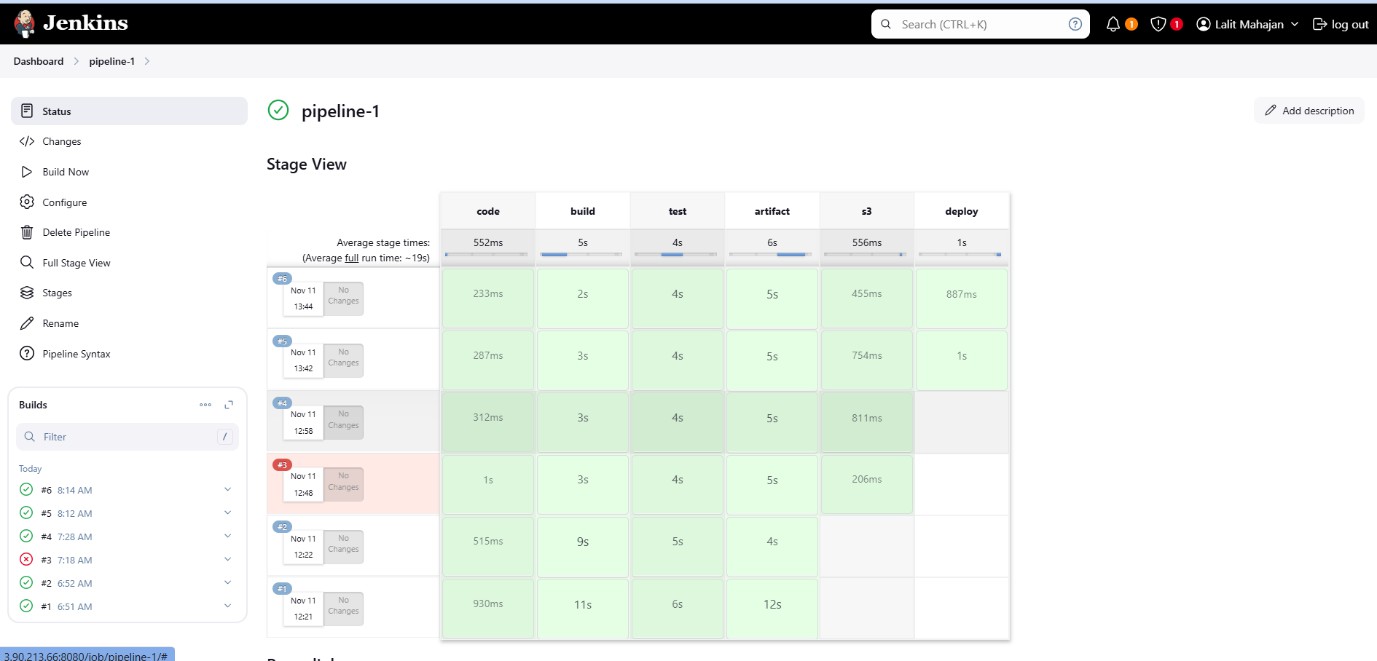
stage('deploy') {

ansiblePlaybook credentialsId: 'ansible', disableHostKeyChecking: true, installation: 'ansible', inventory: '/ect/ansible/hosts', limit: 'dev', playbook: '/etc/ansible/deploy.yml', vaultTmpPath: ''

}

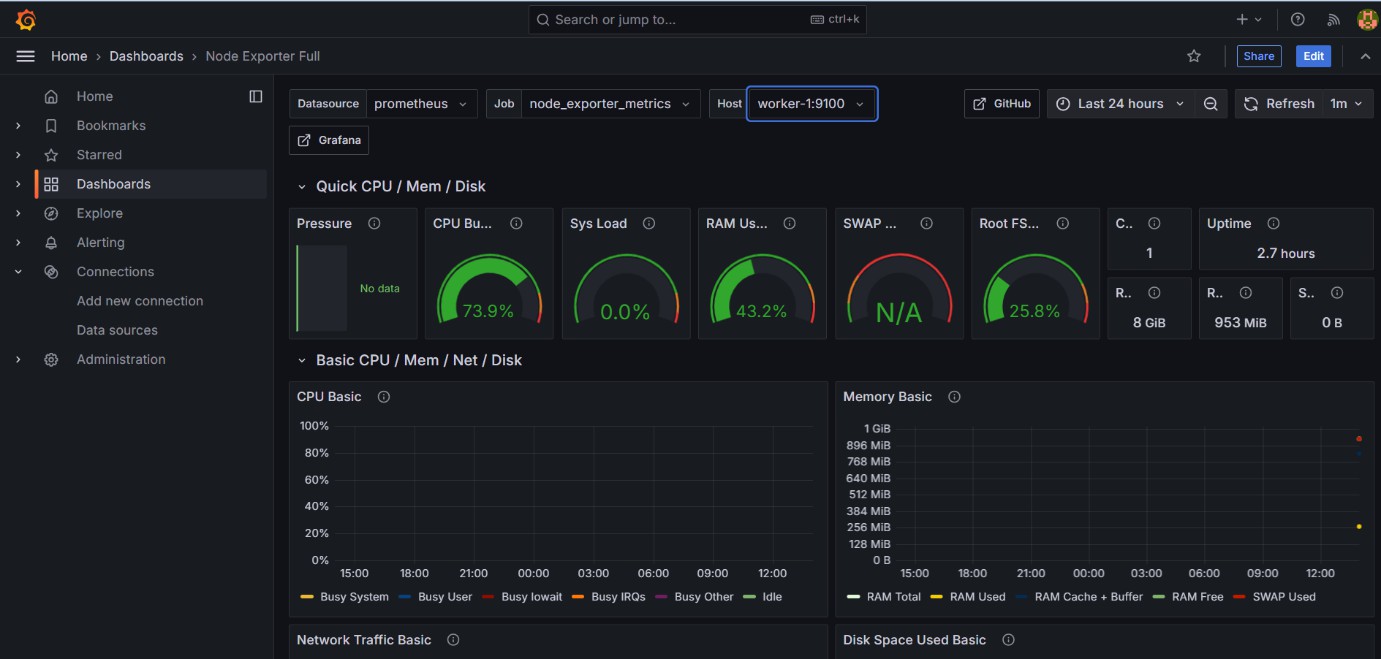
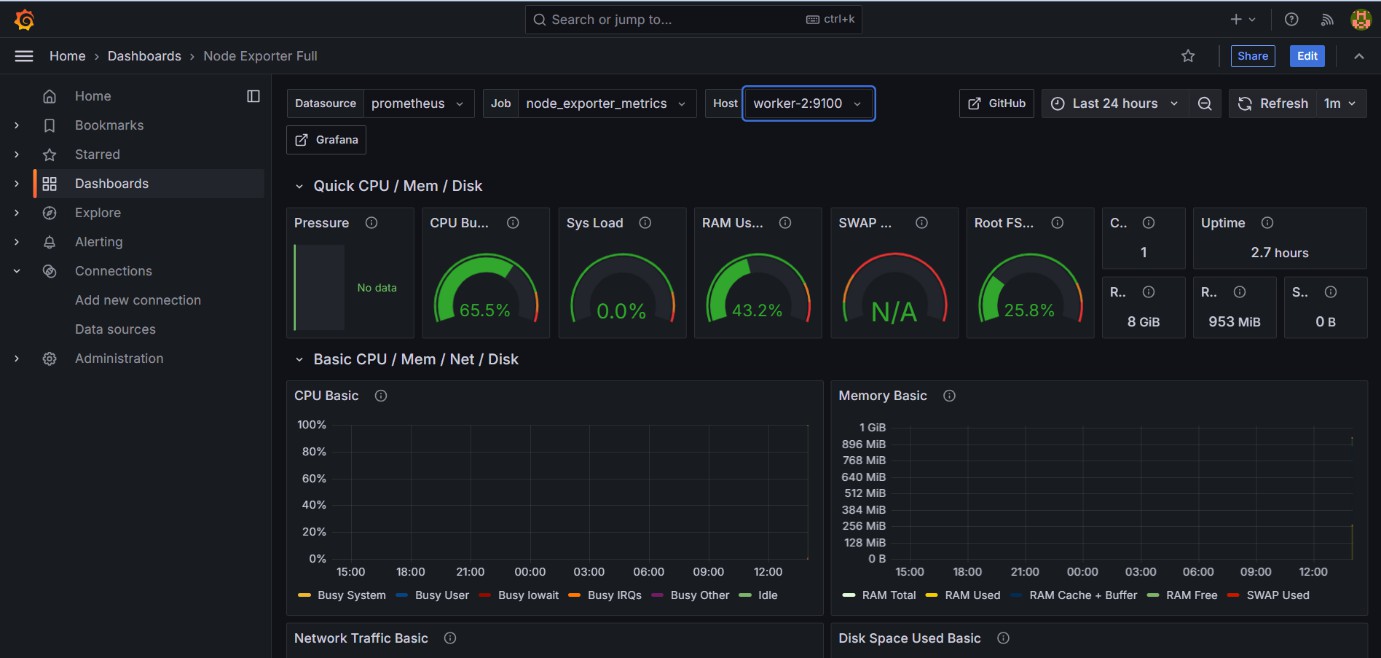
}

**After run the playbook our deployment is done, we can see through Jenkins’s pipeline and also at tomcat our application has deployed.**

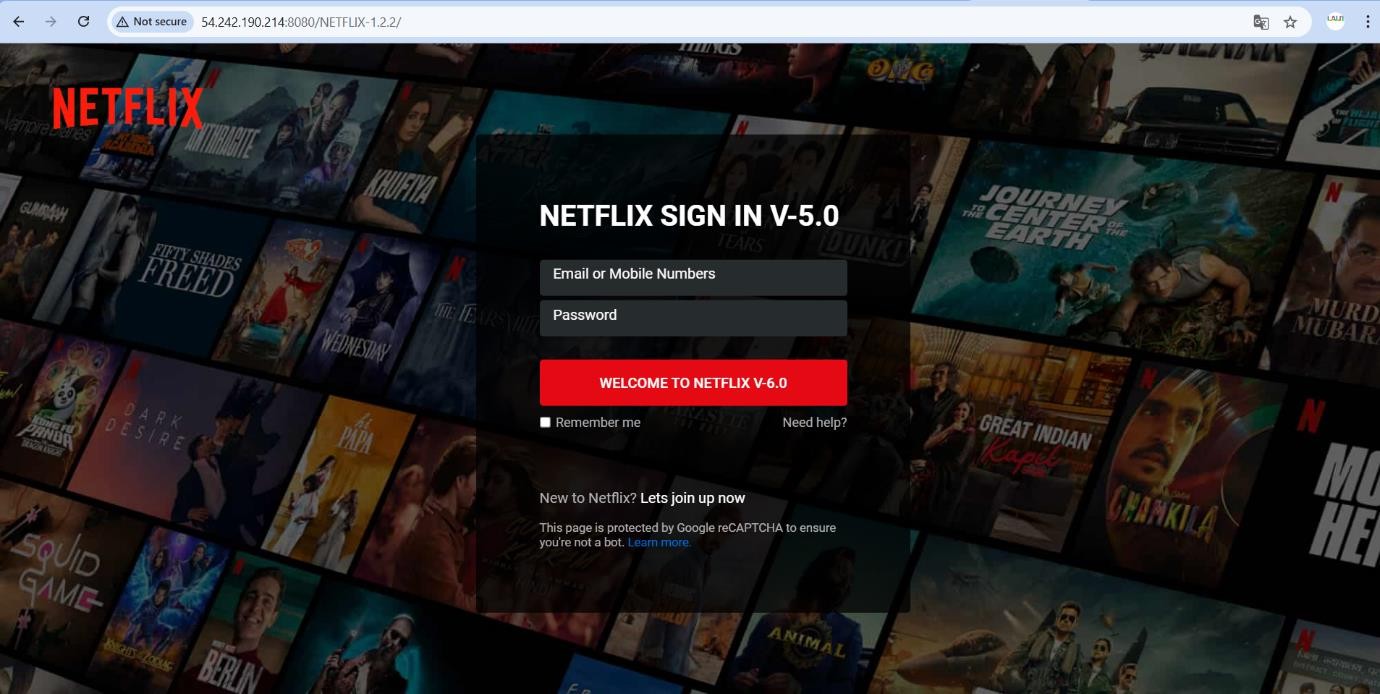
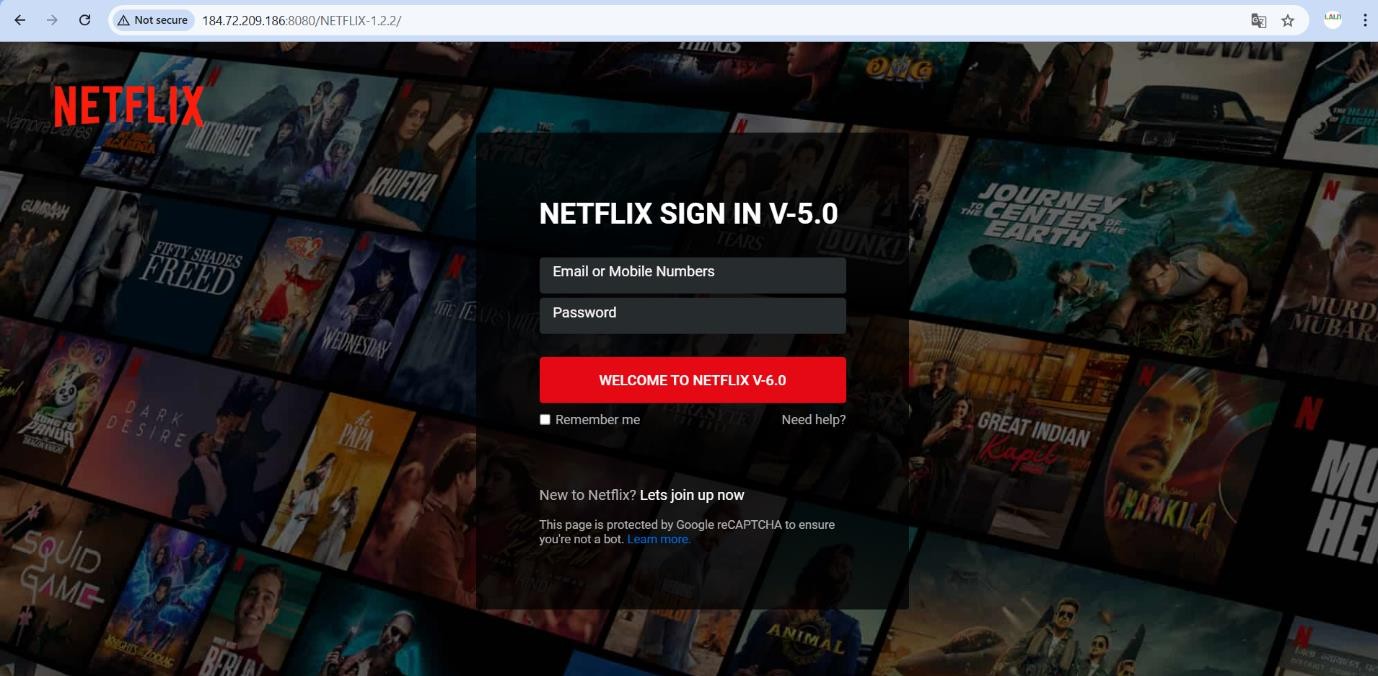
****

**Step 9: Monitoring Worker Nodes with Grafana**

1. Install Components:
   * Install Grafana, Prometheus, and Node Exporter on the monitoring server.
   * Install Node Exporter on worker nodes.
2. Access Grafana:
   * Port: 3000, Username/Password: admin/admin.
3. Connect Prometheus to Grafana:
   * Navigate to Data Sources → Add Prometheus → Enter Prometheus URL → Save & Test.
4. Import Dashboard:
   * Click + → Import → Enter Dashboard ID 1860 → Load → Select Prometheus → Import.
5. Connect to Tomcat:
   * Edit /etc/hosts to add:
   * <public-ip-of-node1> worker-1
   * <public-ip-of-node2> worker-2

* **Dev Server (Worker-1)-**
* **Test Server (Worker-2)-**

OUTPUT –

* **Dev Server (Worker-1)-**
* **Test Server (Worker-2)-**