PGP DevOps – Industry Grade Project

CI/CD PIPELINE FOR ABC TECHNOLOGIES

Divakar Kadupu

2023

Contents

[Foreword 2](#_Toc146908600)

[Github respository of the Code 3](#_Toc146908601)

[Github repository of the supporting files 3](#_Toc146908602)

[Validation of the project by Reviewer 3](#_Toc146908603)

[Business Challenges/Requirements 4](#_Toc146908604)

[Problem Statements/Tasks 5](#_Toc146908605)

[Approach to Solve 5](#_Toc146908606)

[Validation of lab setup 6](#_Toc146908607)

[Approach followed to execute the project. 6](#_Toc146908608)

[Actual Solution 7](#_Toc146908609)

[Task 1 7](#_Toc146908610)

[Task 2 9](#_Toc146908611)

[Task 3 19](#_Toc146908612)

# Foreword

My heartfelt thanks to Edureka and instructors for providing this course to enhancing my confidence to perform a DevOps project.

I have chosen project 1 out of two projects. Below is the solution and the approach I have followed to build the CI/CD for the given Java Project.

Name: Divakar Kadupu  
Email: [mail2divakar@gmail.com](mailto:mail2divakar@gmail.com)  
Mobile: +91-8886552227

Github respository of the Code

<https://github.com/mail2divakar/ABC-Technologies.git>

# Github repository of the supporting files

The following repository has Dockerfile, Jenkinsfile, Ansible playbook, K8s files and monitoring related files

<https://github.com/mail2divakar/devOpsProject.git>

# Validation of the project by Reviewer

Please run the pipeline job in Jenkins for task-3 deploying to tomcat and docker build and run

the container and uploading to Docker hub.

**project-abc-ci**

A screenshot of a computer

Description automatically generated

please run this pipeline job in Jenkins for task-4 deploying to kubernetes and docker using Jenkins

**project-abc-cd-k8s**

Business Challenges/Requirements

ABC Technologies is a leading online retail store, and it has recently acquired a large retail

offline business store. The business store has a large number of stores across the globe but

is following the conventional pattern of development and deployment. As a result, it has

landed at a great loss and is facing the following challenges.

* Low available
* Low scalable
* Low performance
* Hard to build and maintain.
* Developing and deploying are time-consuming.

ABC will acquire the data from all these storage systems and plans to use it for analytics and

prediction of the firm’s growth and sales prospects. In the first phase, ABC has to create the

servlets to add a product and display product details. Add servlet dependencies required to

compile the servlets. Create an HTML page that will be used to add a product. The team is

using Git to keep all the source code.

ABC has decided to use the DevOps model. Once source code is available in GitHub, we

need to integrate it with Jenkins and provide continuous build generation for continuous

delivery as well as integrate with Ansible and Kubernetes for deployment. Use Docker Hub

to pull and push images between Ansible and Kubernetes.

# Problem Statements/Tasks

We need to develop a CI/CD pipeline to automate the software development, testing,

packaging, and deployment, reducing the time to market the app and ensuring good quality

service is experienced by end users. In this project, we must—

* Push the code to our GitHub repository
* Create a continuous integration pipeline using Jenkins to compile, test, and
* Package the code present in GitHub
* Write Dockerfile to push the war file to the Tomcat server
* Integrate Docker with Ansible and write the playbook
* Deploy artifacts to the Kubernetes cluster
* Monitor resources using Grafana

# Approach to Solve

**Task 1:** Clone the project from the GitHub link shared in resources to your local machine.

Build the code using Maven commands.

**Task 2:** Set up the Git repository and push the source code. Then, log in to Jenkins.

1. Create a build pipeline containing a job for each
   1. One for compiling source code
   2. Second for testing source code
   3. Third for packing the code
2. Execute the CI/CD pipeline to execute the jobs created in step 1
3. Set up a master-slave node to distribute the tasks in the pipeline

**Task 3:** Write a Dockerfile. Create an Image and container on the Docker host. Integrate

docker host with Jenkins. Create CI/CD job on Jenkins to build and deploy on a container.

1. Enhance the package job created in step 1 of task 2 to create a docker image.
2. In the Docker image, add code to move the war file to the Tomcat server and build the image.

**Task 4:** Integrate the Docker host with Ansible. Write an Ansible playbook to create an

image and create a continuer. Integrate Ansible with Jenkins. Deploy Ansible-playbook.

CI/CD job to build code on ansible and deploy it on docker container

1. Deploy Artifacts on Kubernetes

2. Write pod, service, and deployment manifest file

3. Integrate Kubernetes with Ansible

4. Ansible playbook to create deployment and service

**Task 5:** Using Prometheus, monitor the resources like CPU utilization: Total Usage, Usage

per core, usage breakdown, memory, and network on the instance by providing the

endpoints on the local host. Install the node exporter and add the URL to the target in

Prometheus. Using this data, log in to Grafana and create a dashboard to show the metrics.

# Validation of lab setup

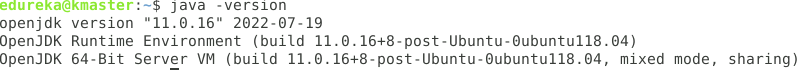
Edureka has given a lab setup with 2 VMs (Master and Worker Nodes) with pre-installed software on Master node

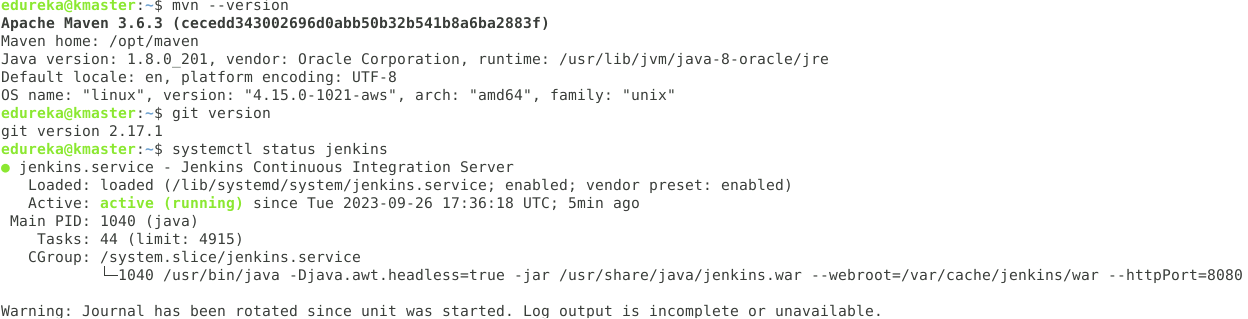
1. Git – 2.17.1
2. Maven – 3.6.3
3. Java – Open JDK 11
4. Jenkins
5. Docker
6. Ansible
7. Kubernetes

Worker node

1. Git – 2.17.1
2. Maven – 3.6.0
3. Java – Open JDK 8, need to upgrade it to 11
4. Docker – 19.03.6
5. Ansible – same as master node
6. Kubernetes – same as master node

Supporting snapshots



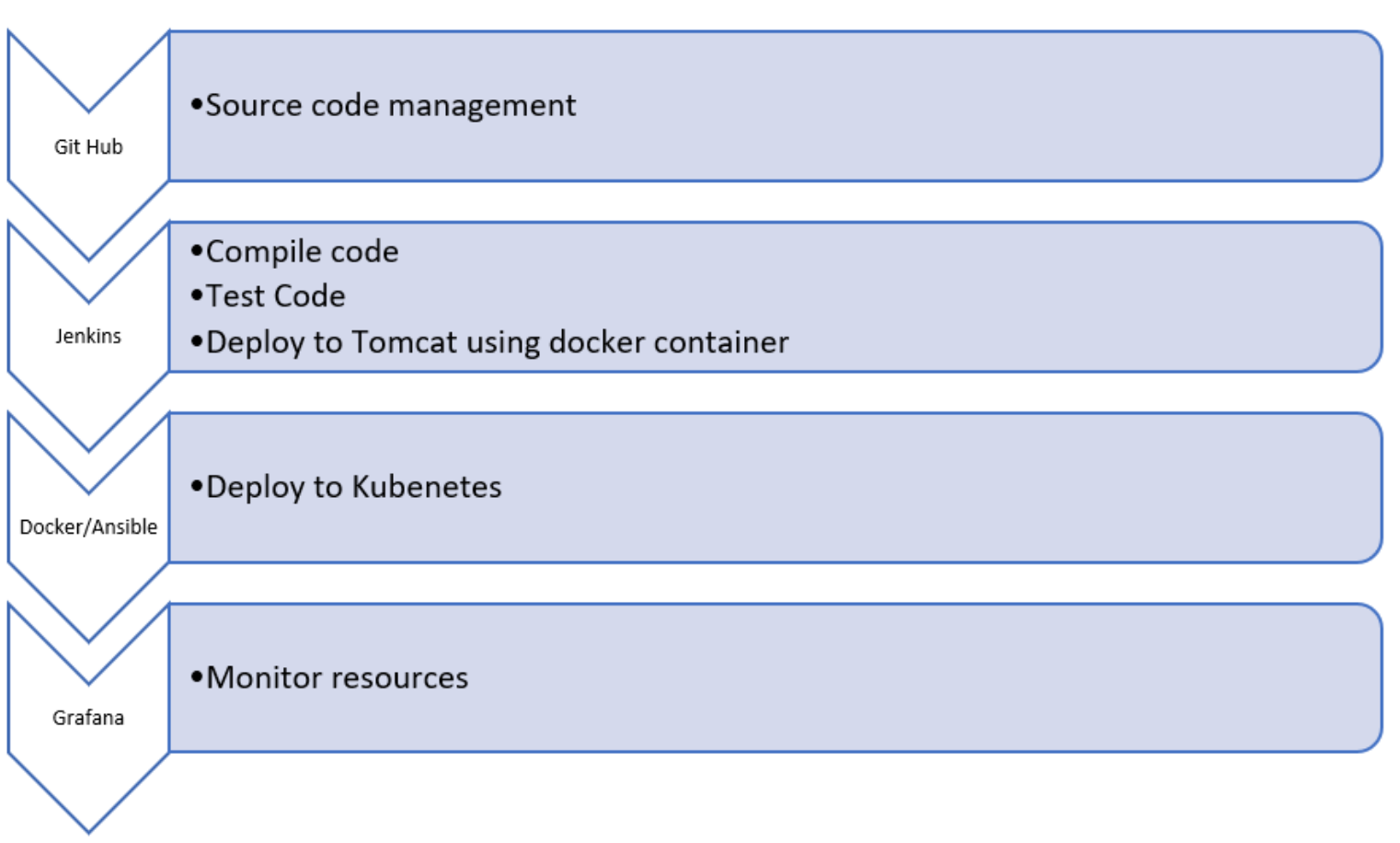






# Approach followed to execute the project.

1. Using Jenkins master and node combination, create a pipeline with steps each for
   1. Checkout of the project
   2. Compile
   3. Run tests
   4. Package
2. Using Ansible and Docker, first write a Dockerfile to take a base image of tomcat, copy the war file generated to the respective tomcat folder and finally build a docker image
3. Push the docker image to dockerhub repository
4. Using Ansible again, pull the docker image built previously and create pod/s using Kubernetes config files and run the pod/s.
5. Setup monitoring for the above pod/s using Prometheus and Grafana to monitor the metrics



# Actual Solution

## Task 1

Clone the project from the GitHub link shared in resources to your local machine. Build the code using Maven commands

Solution:

I have taken the project shared by edureka in LMS for this project and pushed the code to my own github repository @ <https://github.com/mail2divakar/ABC-Technologies.git>

First locally compiled the project and ran other maven targets for clean, compile, test, package, install directly from the terminal.

1. /opt/maven/bin/mvn compile
2. /opt/maven/bin/mvn test
3. /opt/maven/bin/mvn package
4. /opt/maven/bin/mvn clean install

/opt/maven/bin/mvn is the path of the maven home.

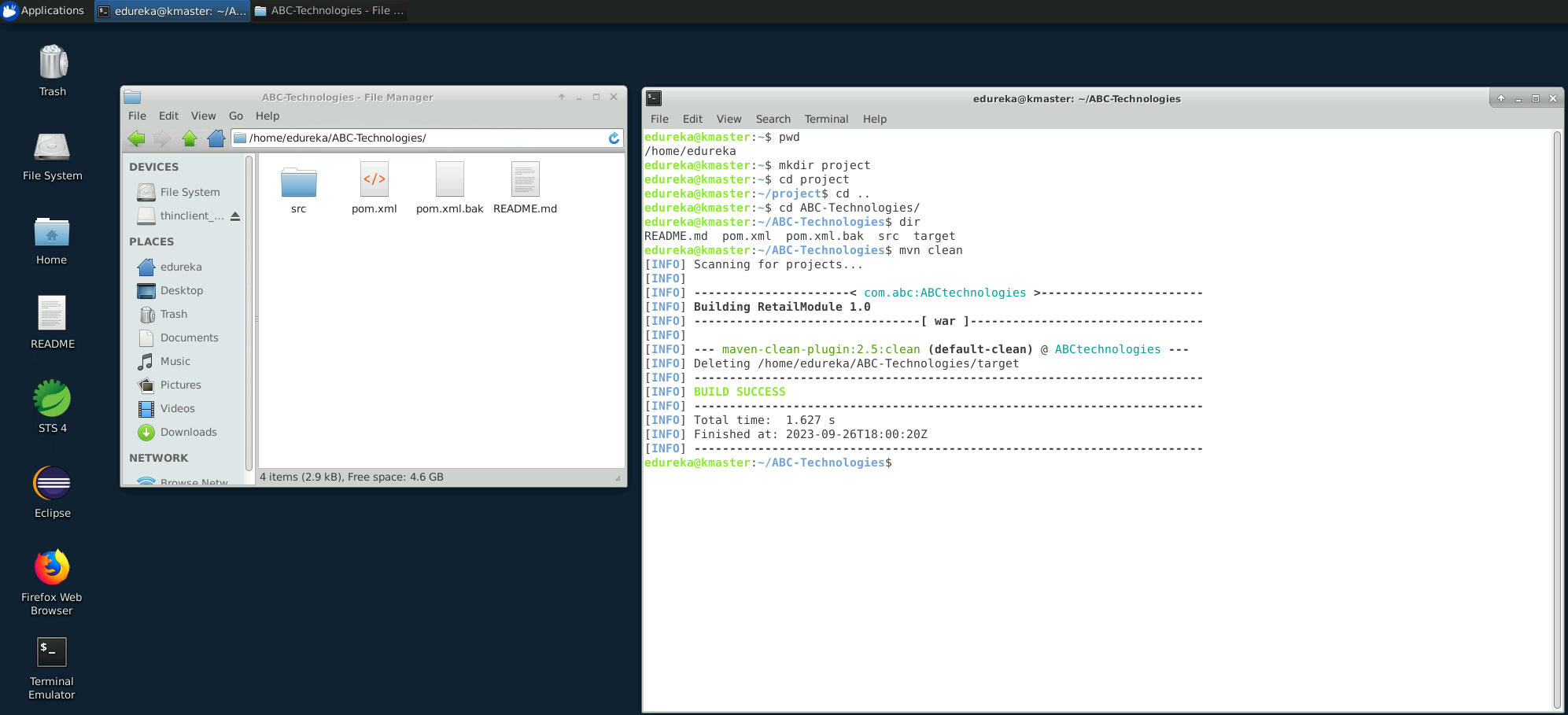
Below are the snapshots.

Git repo clone to local directory @ /home/edureka

A computer code with black text

Description automatically generated

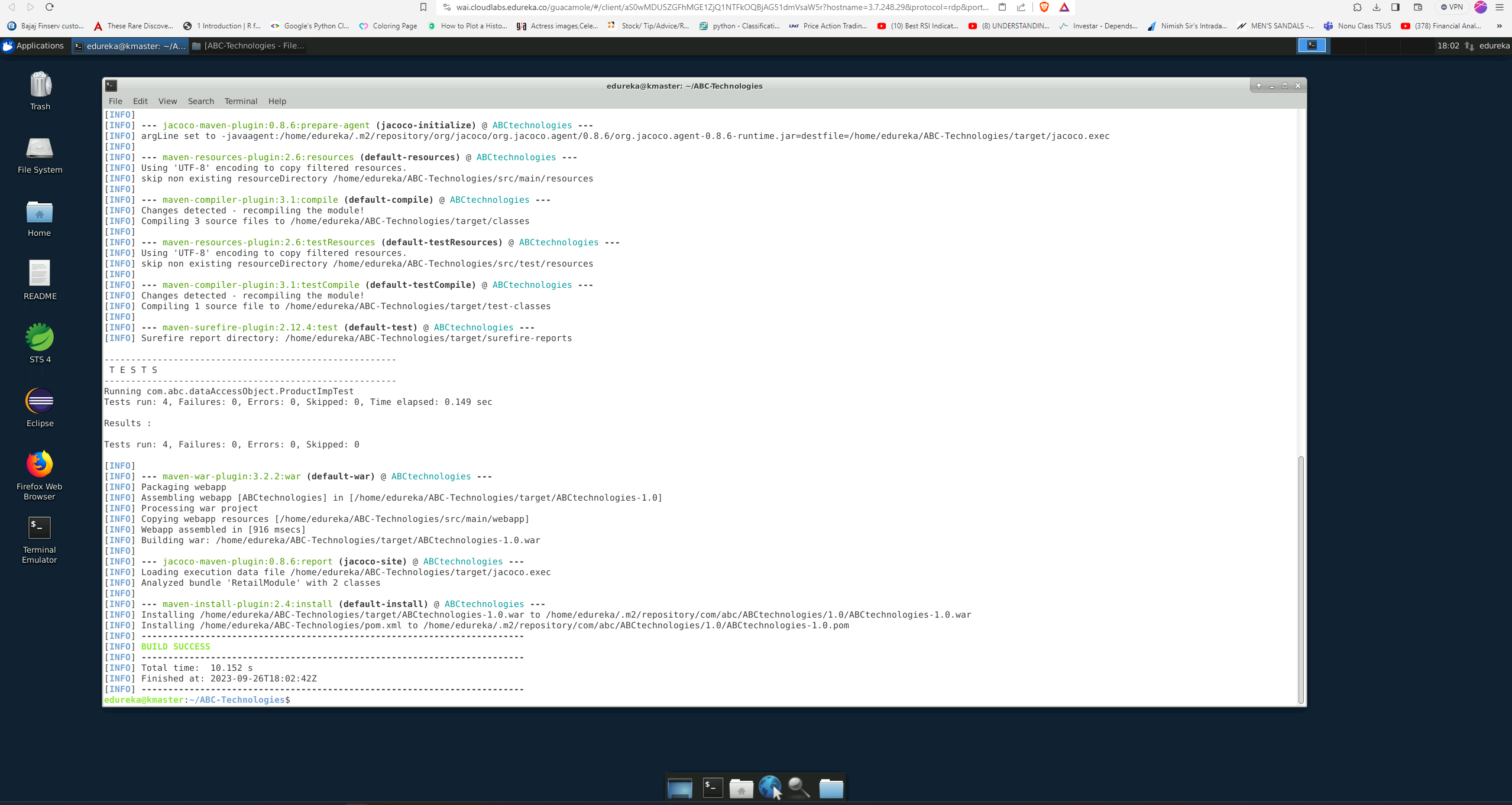
mvn clean



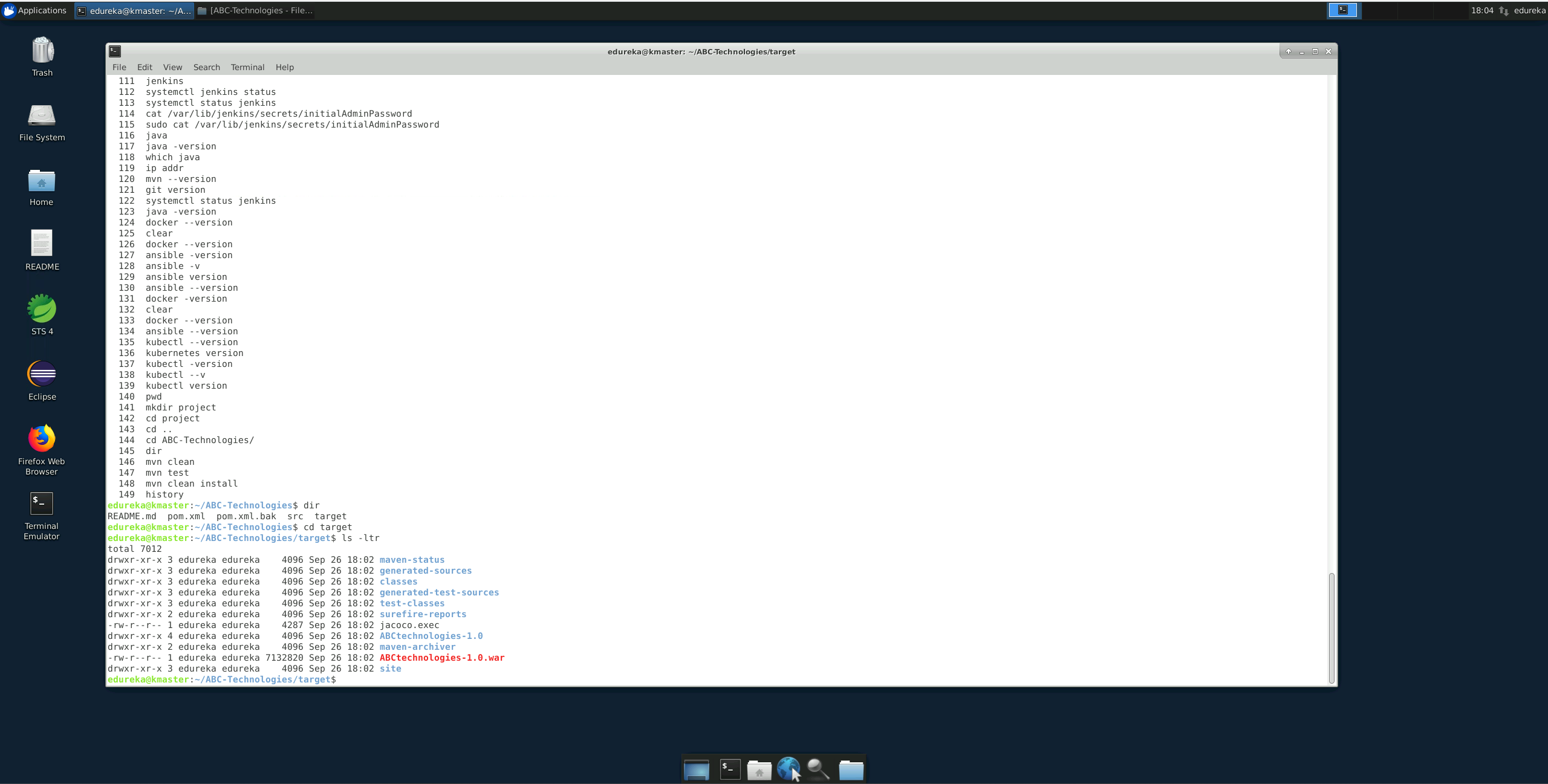
mvn test



mvn install



History of commands and the project’s target folder



target folder in above snapshot contains compilation data and package task created .war executable file to deploy.

Task 2

1. Set up the Git repository and push the source code. Then, log in to Jenkins.
2. Create a build pipeline containing a job for each
   1. One for compiling source code
   2. Second for testing source code
   3. Third for packing the code
3. Execute the CI/CD pipeline to execute the jobs created in step 1
4. Set up a master-slave node to distribute the tasks in the pipeline

**Approach I have followed**

- Jenkins has been installed in master server already so i have created the 3 jobs for compile, test and package in Jenkins and created pipeline with these 3 jobs and set up the agent machine (slave machine) and shared the load to agent as well.

- As a given project is based on java, I have used maven to build the code and Jenkins is a build automation server that helps to automate these things so i have set up the java, maven paths of master in global tool configuration in Jenkins and set up the Jenkins goals and left git path as default.

- Tools location:

* Maven - /opt/maven
* Jdk - /usr/lib/jvm/java-8-oracle

- Goals:

* clean
* compile
* test
* Package

Please find the screenshots of the above task-2:

Setting up Jenkins

A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

Setup is done

User: admin

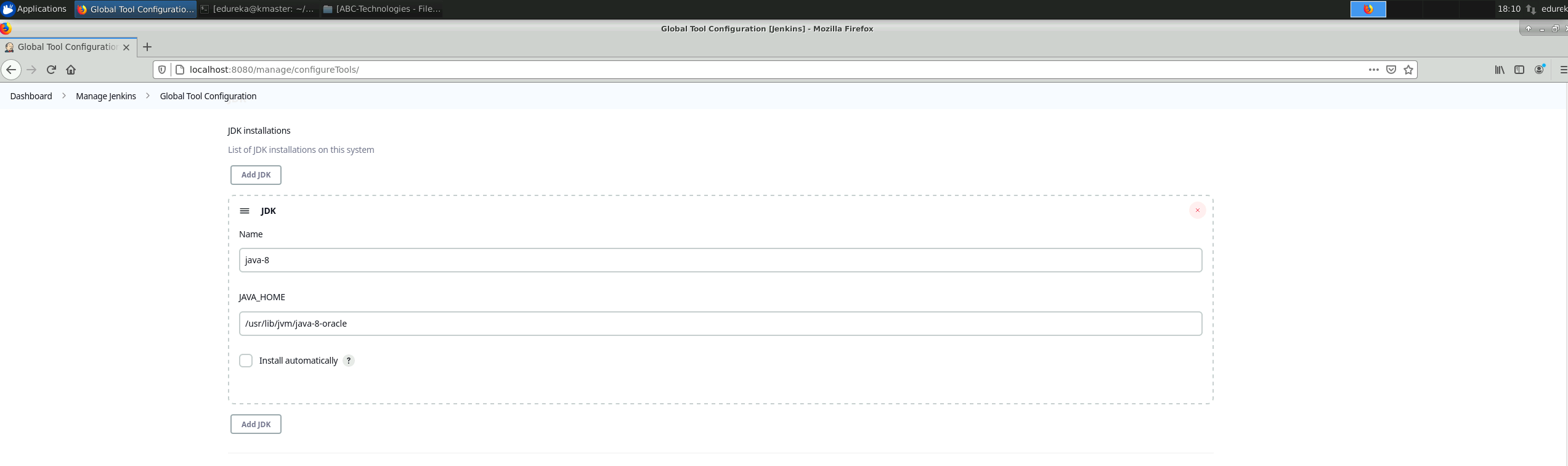
Password: Likhith#30

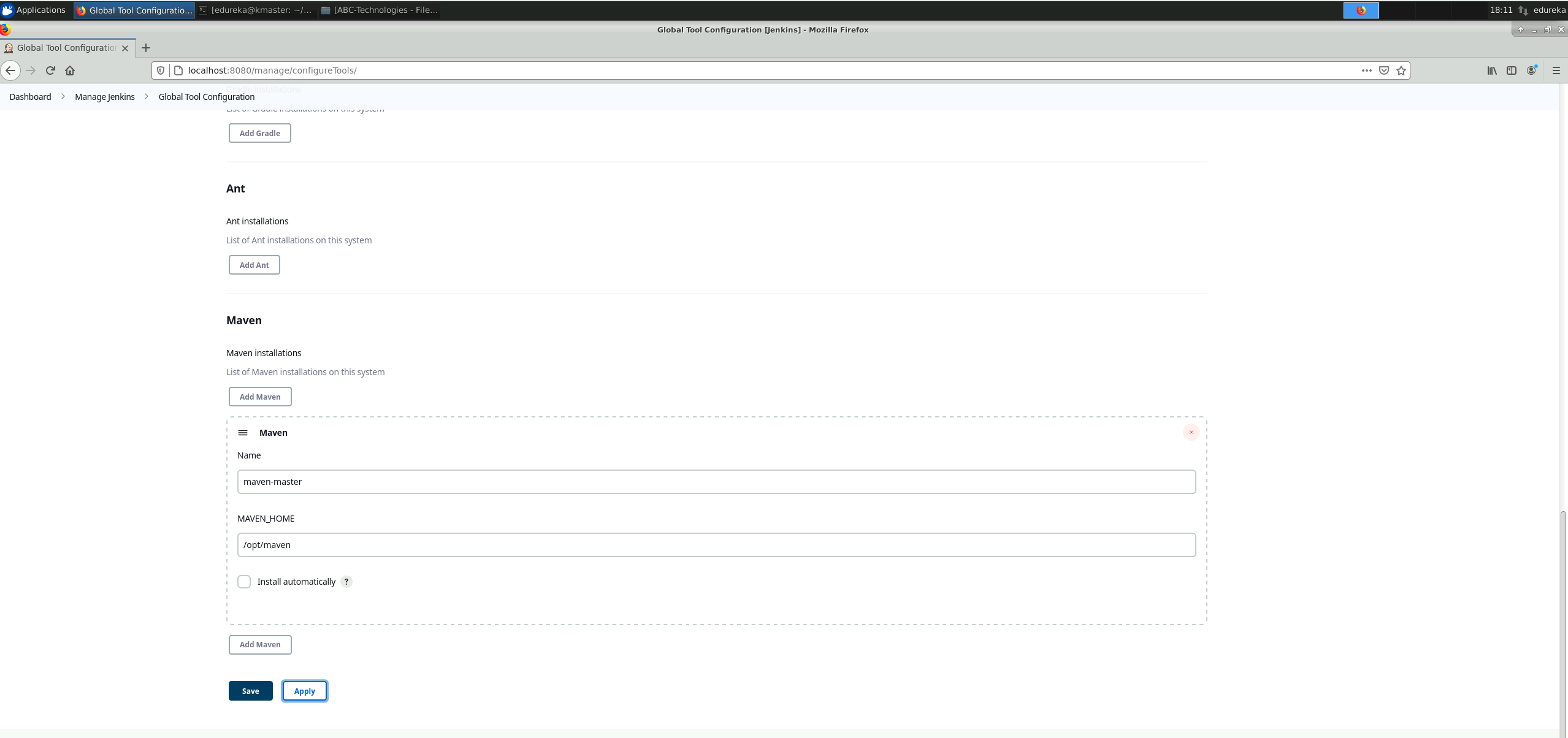
A screenshot of a computer

Description automatically generated

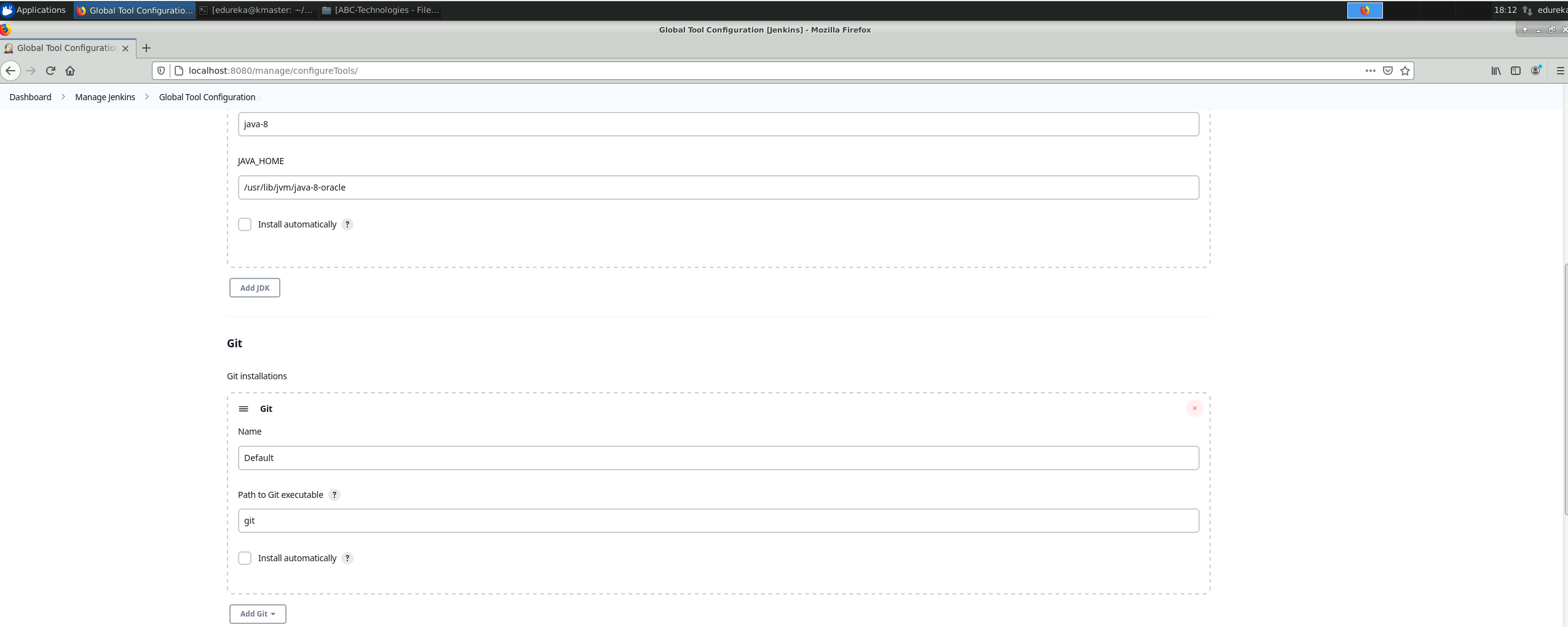
Tool configuration

Added Jdk and Maven as below





Option for Git is left as is (default)



Step 2: Execute the CI/CD pipeline to execute the jobs

I have used Pipeline job in Jenkins master with groovy script for checkout, clean, test, package stages of the given ABC-Technologies project.

Here is the pipeline job config

A screenshot of a computer

Description automatically generated

And I have configured the git polling as a cron job for the pipeline job that triggers for every 30 minutes for github code so that if whatever change happens then it will trigger and perform

the given task.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Here is the pipeline job status

A screenshot of a computer

Description automatically generated

Snippets of console log of this job.

Triggered by myself.

A screenshot of a computer

Description automatically generated

A close-up of a computer screen

Description automatically generated

Workspace target folder in Jenkins

A screenshot of a computer

Description automatically generated

**Set up a master-slave node to distribute the tasks in the pipeline**

**Created a ssh keys using root in slave65 machine. Generated ssh key for this user.**

A screenshot of a computer

Description automatically generated

**In Jenkins, configured global credentials to use ssh key based authentication**

A screenshot of a computer

Description automatically generated

The agent configurations:

- When I tried to configure the agent i was getting compatibility issue while running the agent

command in agent server(slave server) and got to know that this is due to java 8

compatibility issue beginning with Jenkins 2.357 and Jenkins 2.361.1, running the controller

on Java 11 and agents on Java 8 will result in the following error:

Error: A JNI error has occurred, please check your installation and try

again

Exception in thread "main" java.lang.UnsupportedClassVersionError:

hudson/remoting/Launcher has been compiled by a more recent version of the Java Runtime (class file version 55.0), this version of the Java Runtime only recognizes class file versions up to 52.0

at java.lang.ClassLoader.defineClass1(Native Method)

at java.lang.ClassLoader.defineClass(ClassLoader.java:756)

at java.security.SecureClassLoader.defineClass(SecureClassLoader.java:142)

at java.net.URLClassLoader.defineClass(URLClassLoader.java:473)

at java.net.URLClassLoader.access$100(URLClassLoader.java:74)

at java.net.URLClassLoader$1.run(URLClassLoader.java:369)

at java.net.URLClassLoader$1.run(URLClassLoader.java:363)

at java.security.AccessController.doPrivileged(Native Method)

at java.net.URLClassLoader.findClass(URLClassLoader.java:362)

at java.lang.ClassLoader.loadClass(ClassLoader.java:418)

at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:352)

at java.lang.ClassLoader.loadClass(ClassLoader.java:351)

at sun.launcher.LauncherHelper.checkAndLoadMain

Had to install the jdk version of 11

Agent node successfully linked to Jenkins master.

A screenshot of a computer

Description automatically generated

Re-triggered the job to run on worker node added above, below snapshot shows the job running on worker-1 node

A screenshot of a computer

Description automatically generated

Pipeline status

A screenshot of a computer

Description automatically generated

Build history of the node

A screenshot of a computer

Description automatically generated

Node information (additional)

A screenshot of a computer

Description automatically generated

Updated the ci job to run every day only once.

A screenshot of a computer

Description automatically generated

## Task 3

Write a Docker file. Create an Image and container on the Docker host. Integrate

docker host with Jenkins. Create CI/CD job on Jenkins to build and deploy on a container.

1. Enhance the package job created in step 1 of task 2 to create a docker image.

2. In the Docker image, add code to move the war file to the Tomcat server and build

the image.

**Approach I have followed:**

- Jenkins and Docker are already installed in master machine and slave machine.

- Using earlier Jenkins package job and enhancing with docker integration to do the below two things here

i) With the war file generated in package stage of the ci job, build the docker image

ii) Push/Upload the image to dockerhub

For this purpose, I have created a new dockerhub account and created a repository (public) to hold the image.

A screenshot of a computer

Description automatically generated

To successfully push the docker image, had to install the below plugins for the docker in Jenkins.

* Docker API Plugin
* Docker Commons Plugin
* Docker Pipeline
* Docker Plugin
* docker-build-step

A screenshot of a computer

Description automatically generated