**Project Plan**

Github Repo

EC2 instance

Ubuntu OS

Docker

Jenkins Server

**Jenkins tasks using Groovy script**

SCM Checkout

MAVEN

install

package

validate

compile

test

DOCKER

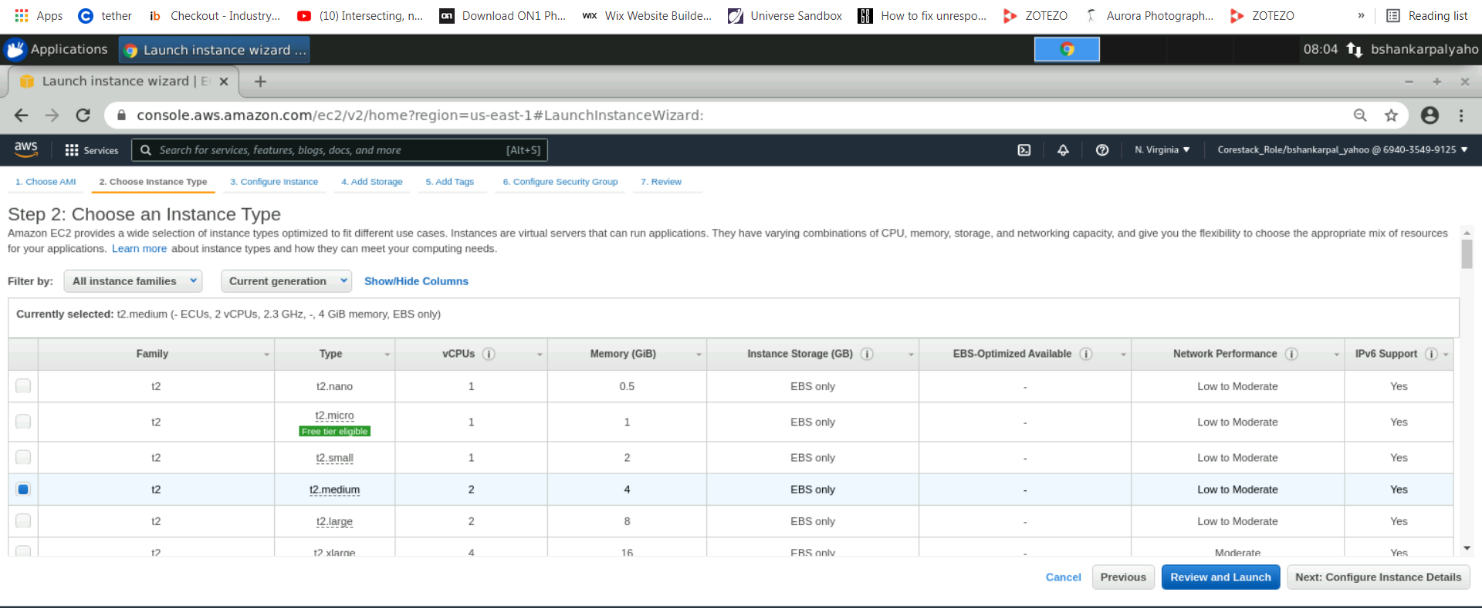
image

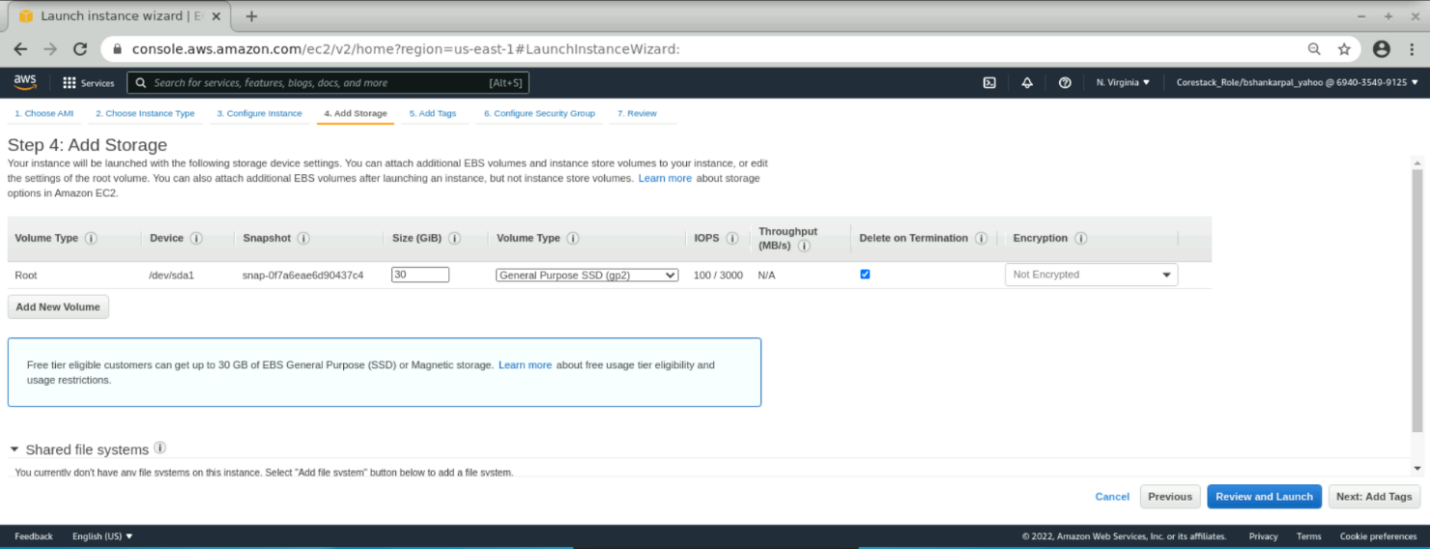
container

push

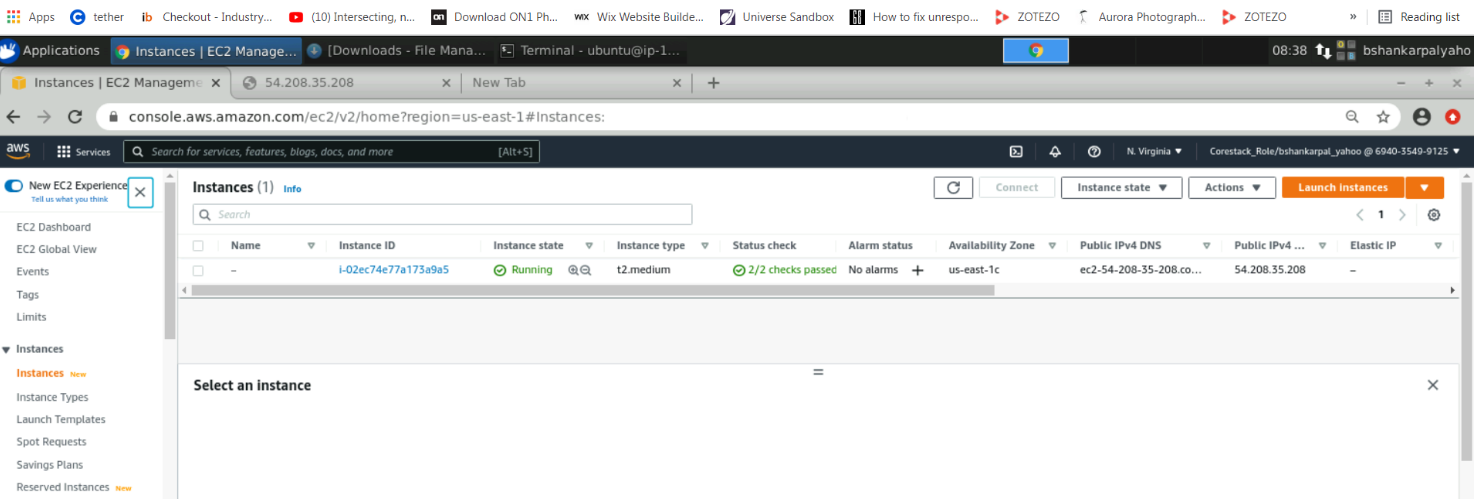
**Screen Shots in sequence**

1. A. Create EC2 instance [ choosing instance type t2 medium]

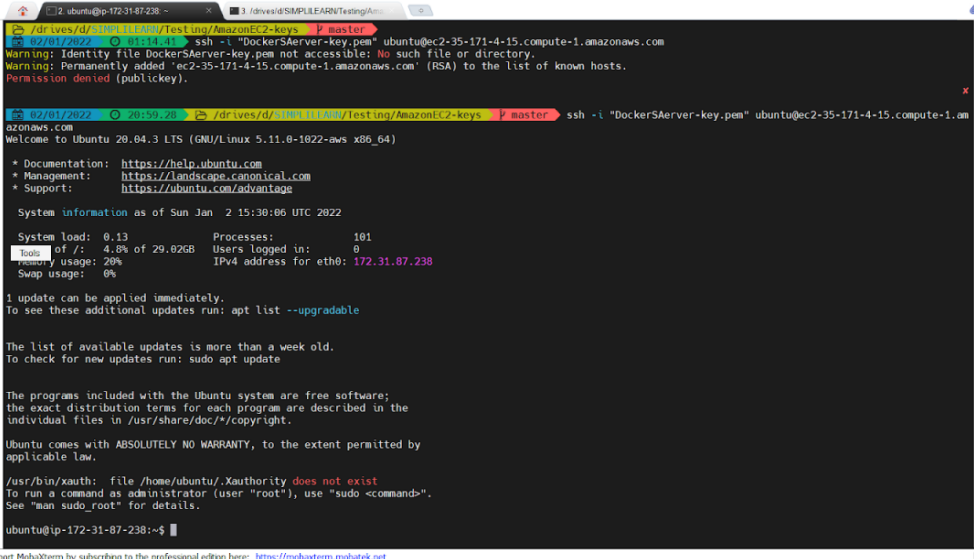


1. b. Create EC2 instance [ choosing 30GB Storage]  
   

1.c. Create EC2 instance [ running]



1. Connecting to EC2 instance, through ssh client using private key downloaed to local directory, ssh url copied from ”connect” button corresponding to instance.



1. Installed docker using following commands

sudo apt-get update

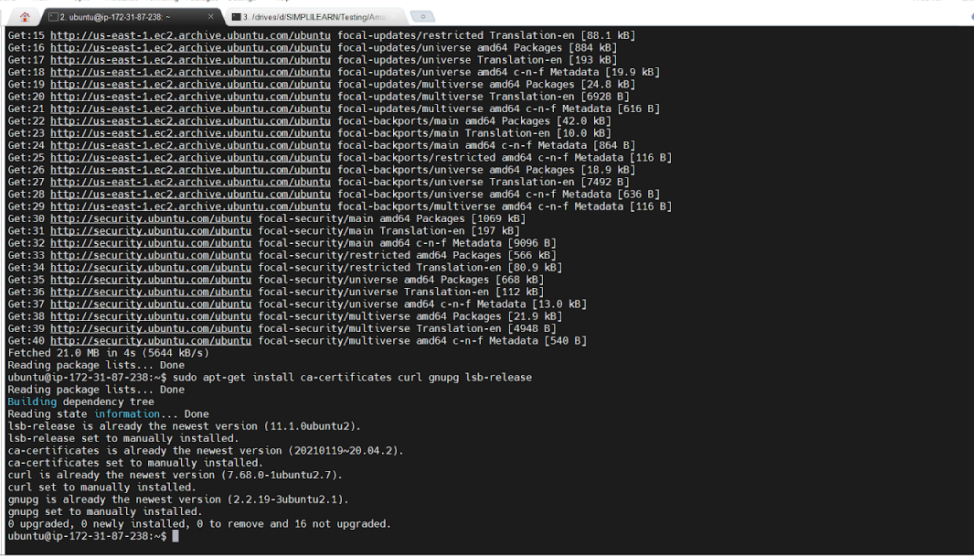
sudo apt-get install ca-certificates curl gnupg lsb-release

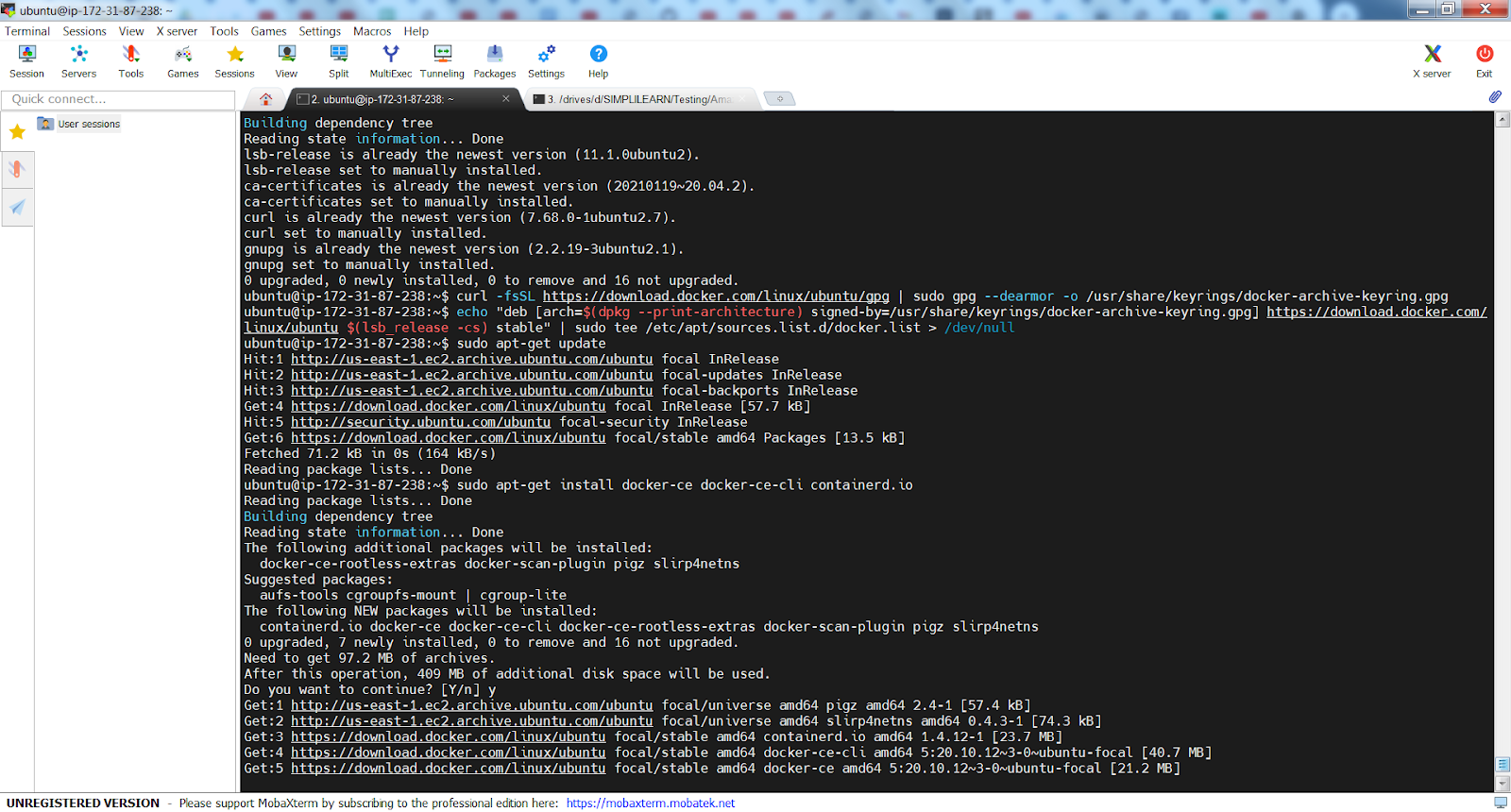
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io





1. Installed java and maven using following command

sudo apt-get install default-jdk maven -y

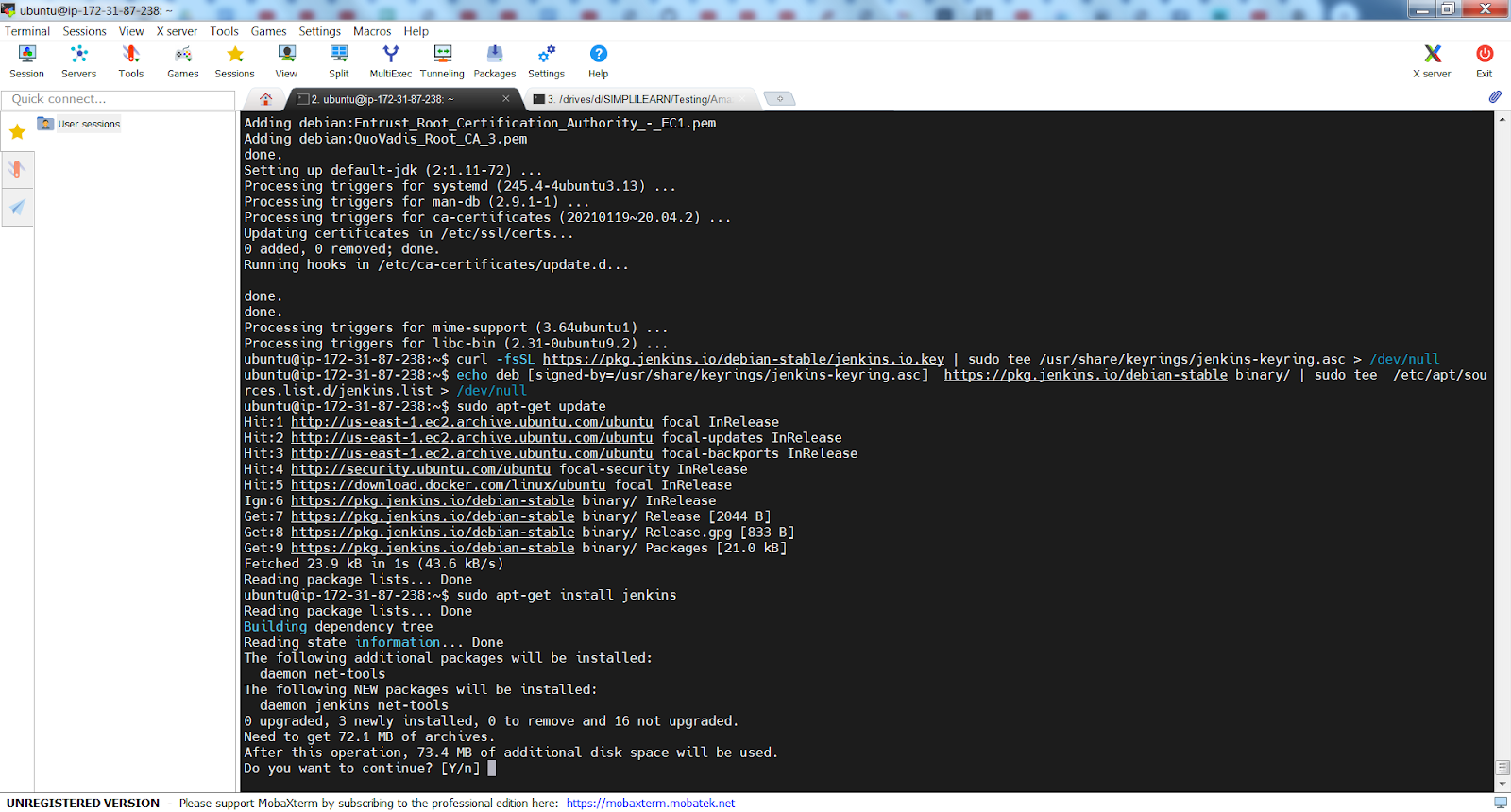
1. Installed jenkins server using following commands

curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null

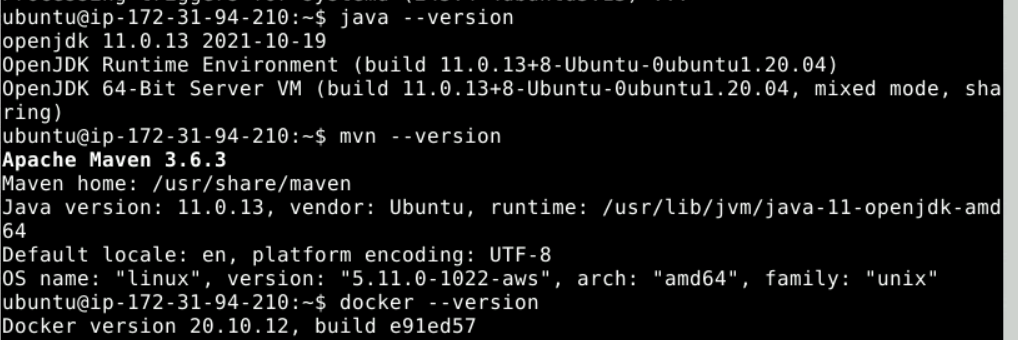
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt-get update

sudo apt-get install jenkins



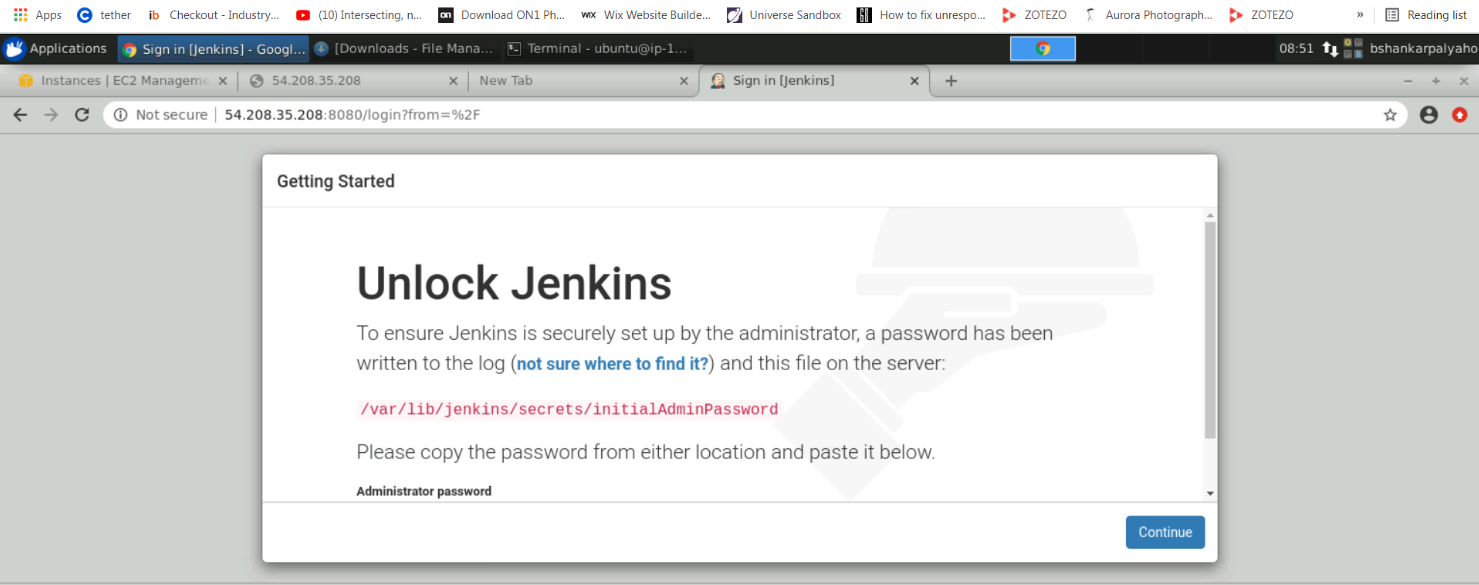
Following checks for the versions of docker, maven, java and jenkins





1. Check EC2 instance “security group” for port 8081 being added
2. Check Jenkins local webpage by typing following URL in browser

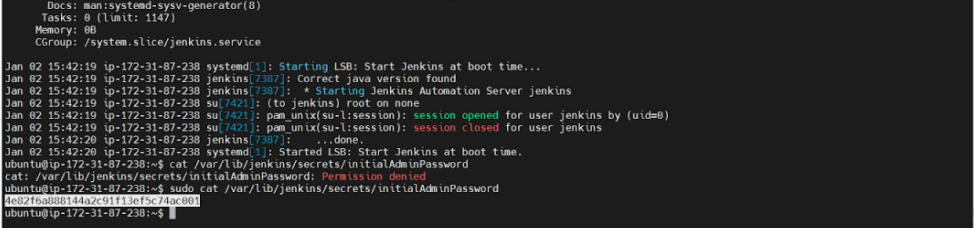
http://54.208.35.208:8080

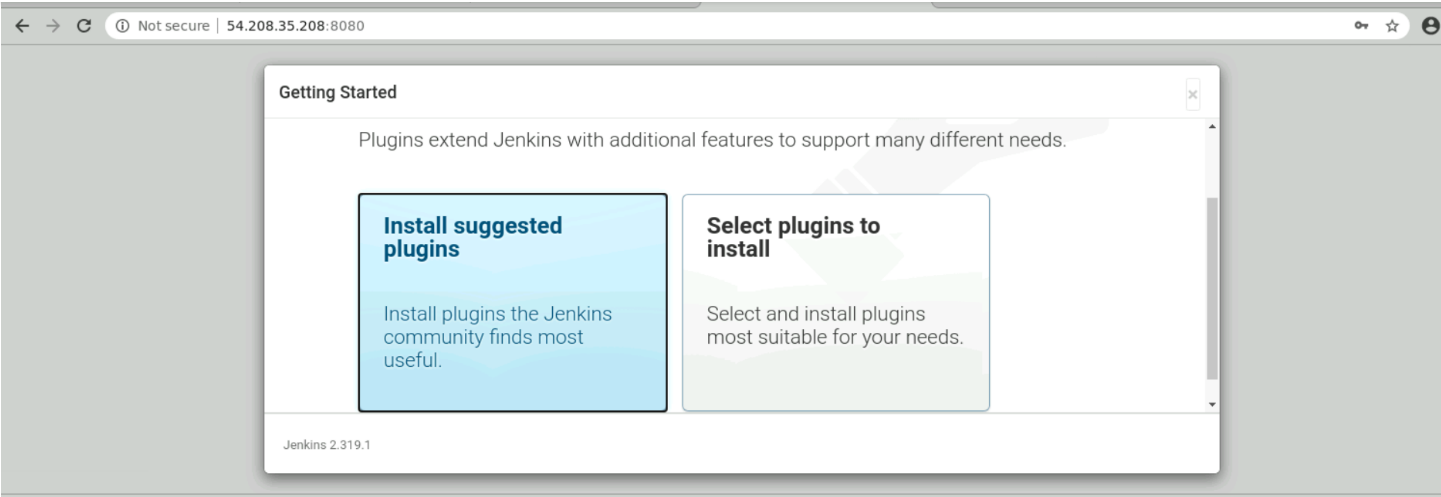


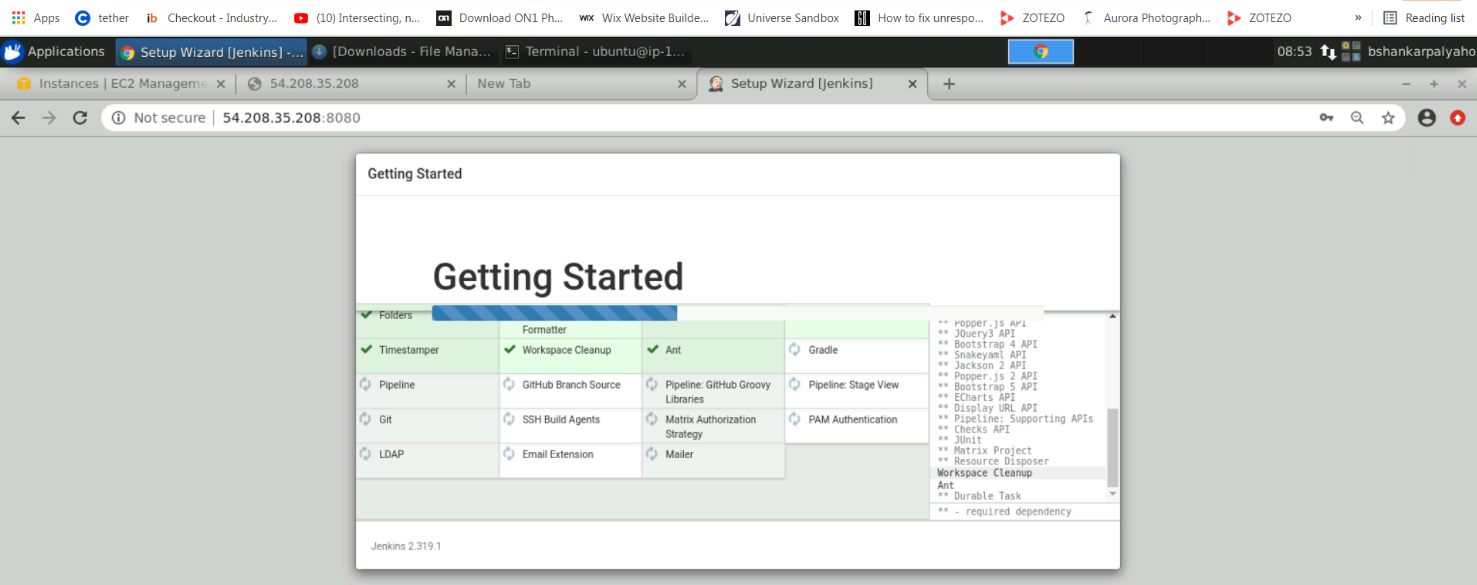
1. Copy above path and use following command

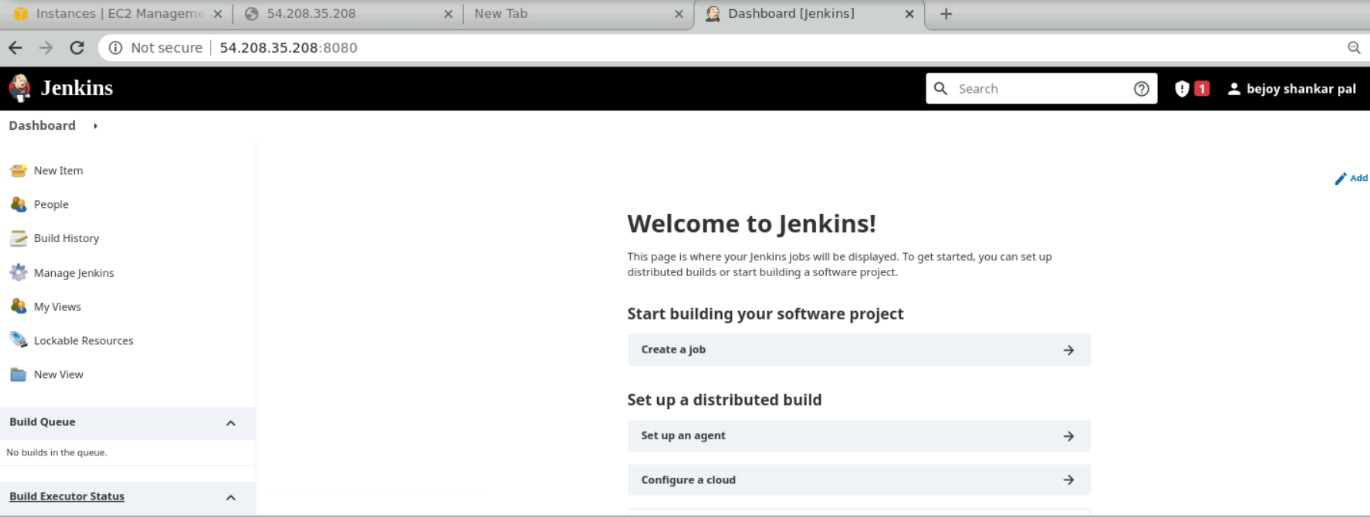
Sudo cat /var/lib/jenkins/secrets/initialAddminPassword

Copy the ouput and paste in password textbox for login

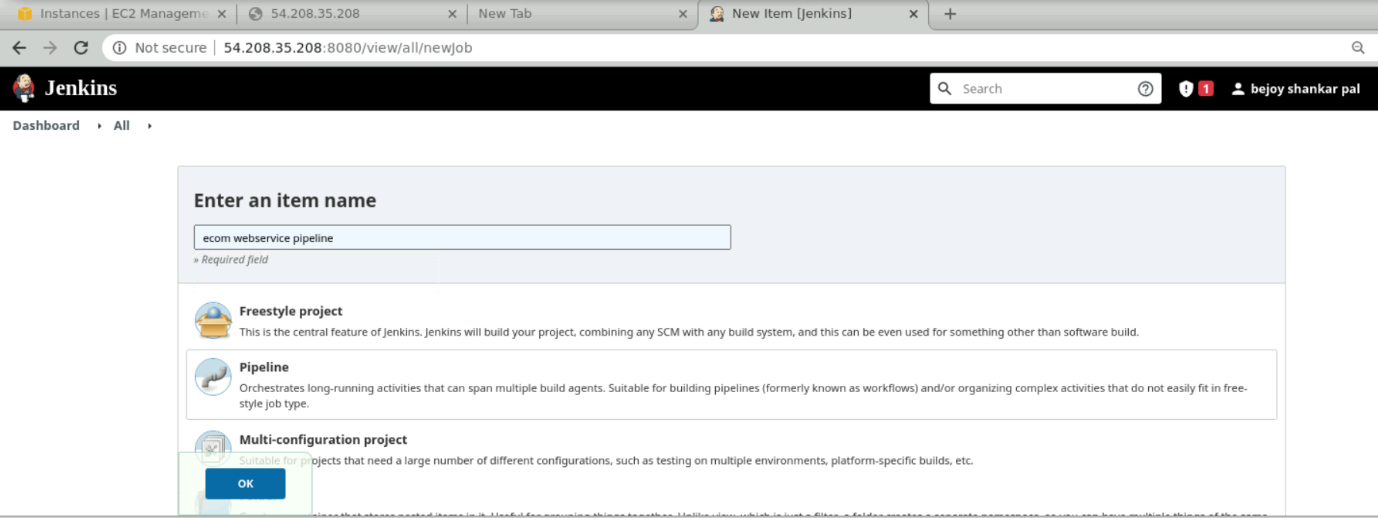




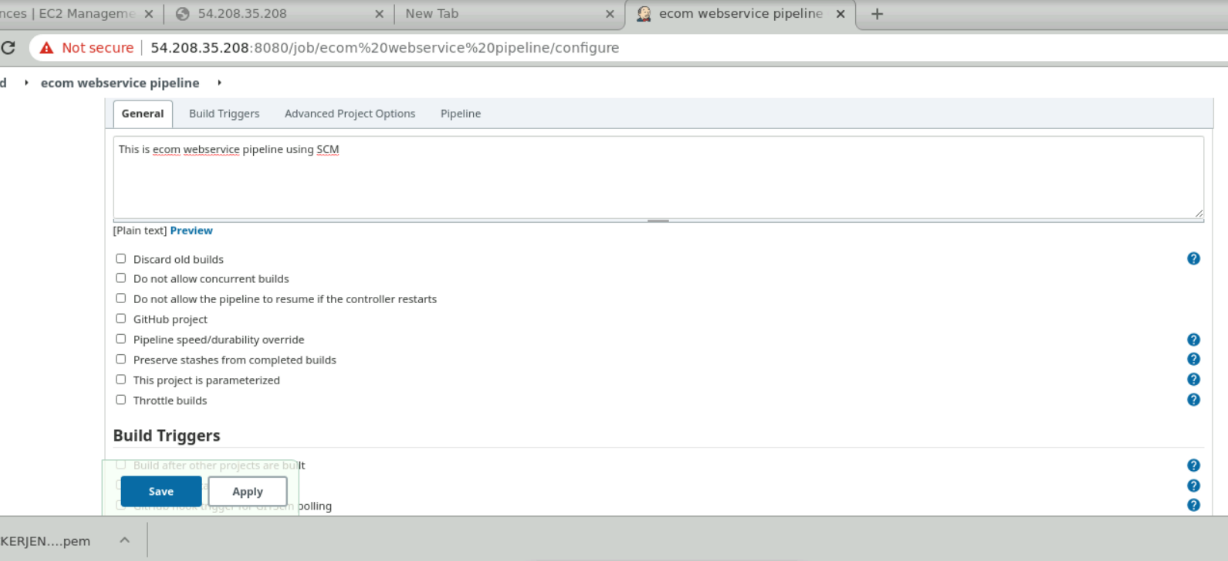




1. Select New Item, from left menu, enter item name as “ecom webservice pipeline” and select pipeline option, click OK

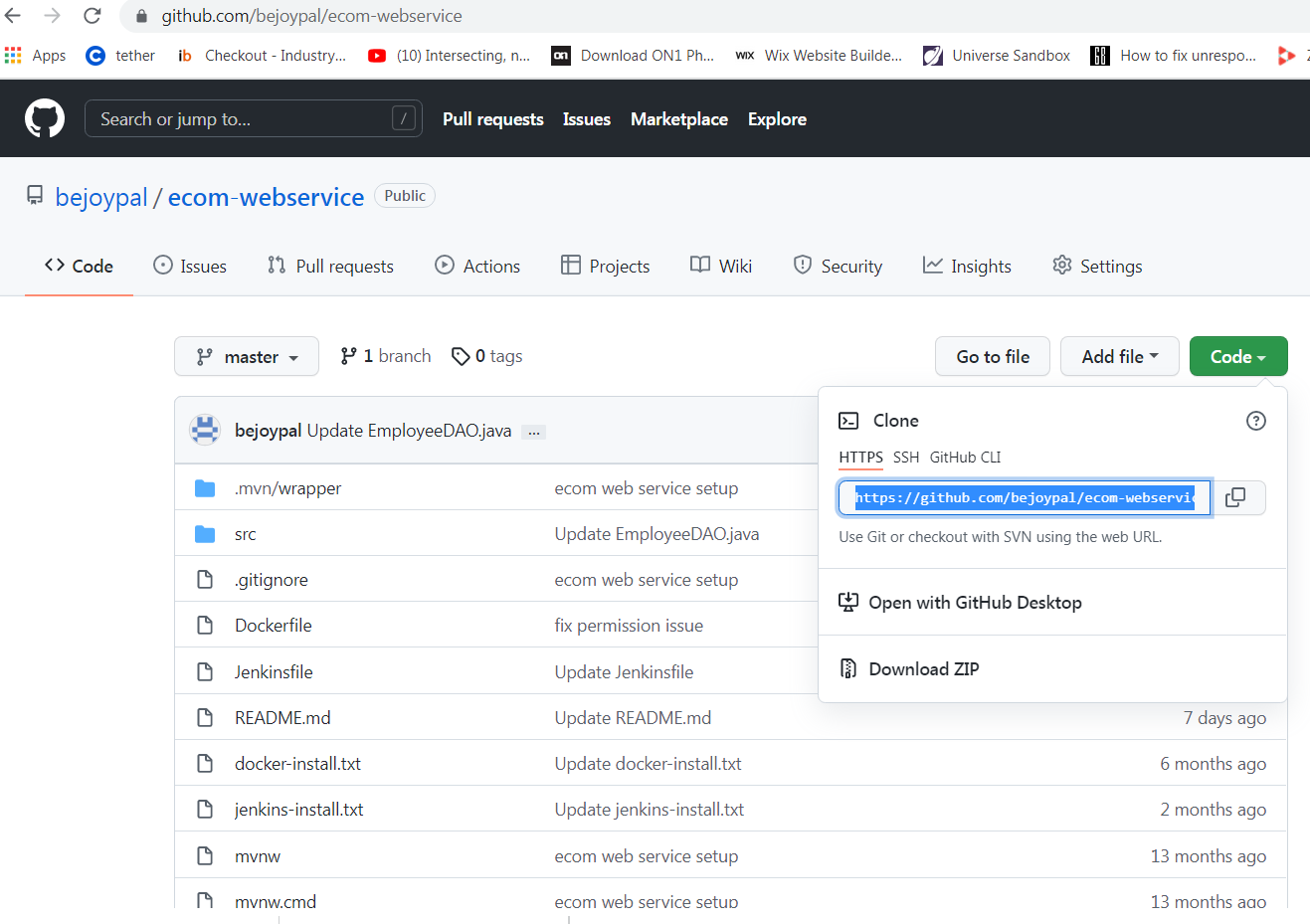


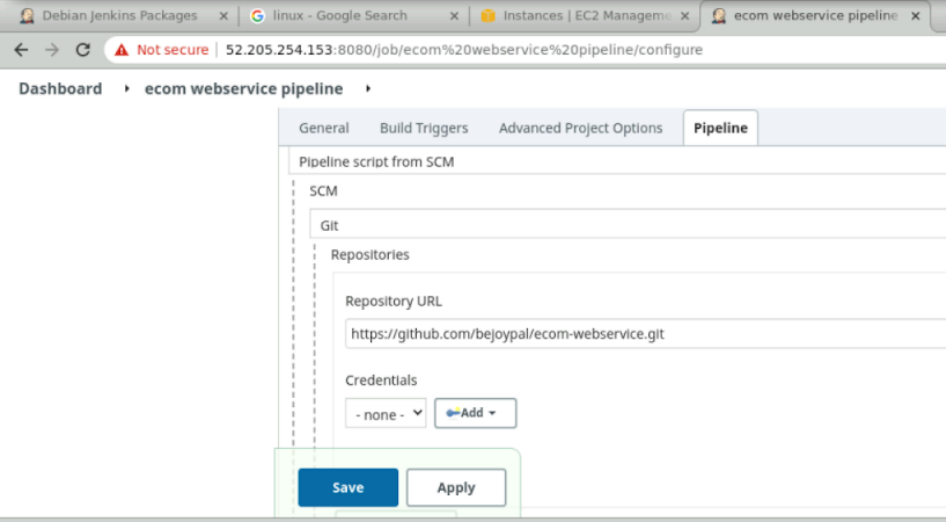
1. Type “This is ecom webservice pipeline using SCM” in desciption box.



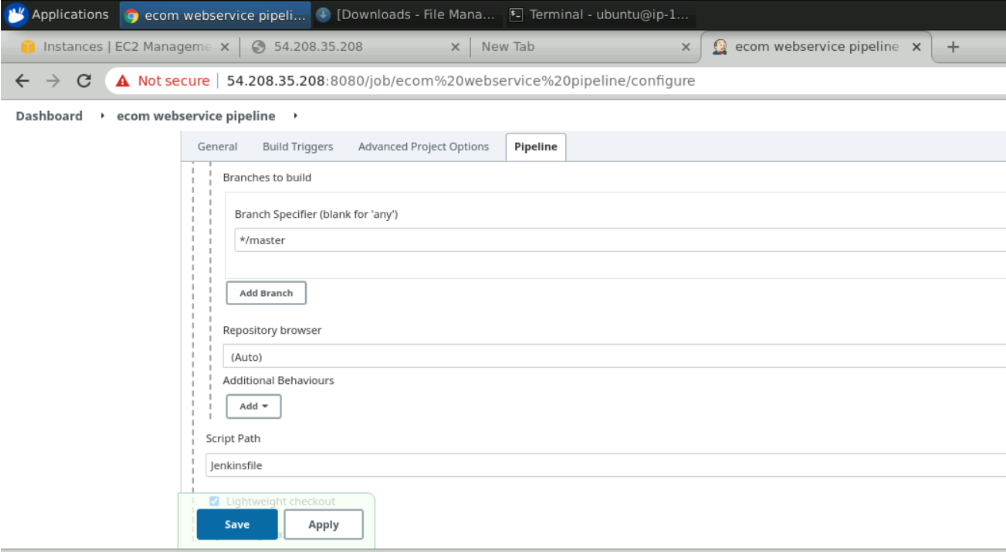
1. Go to pipeline tab, select Pipeline from SCM and enter following github repository

<https://github.com/bejoypal/ecom-webservice.git>





Which refers to Jenkins groovy script file



Click Save

Here is the content of Jenkins file

|  |
| --- |
| pipeline { |
|  | agent any |
|  |  |
|  | triggers { |
|  | pollSCM('\* \* \* \* \*') |
|  | } |
|  | // Got permission denied while trying to connect to the Docker daemon socket at unix. |
|  | // sudo usermod -a -G docker jenkins |
|  | // restart jenkins server -> sudo service jenkins restart |
|  | stages { |
|  |  |
|  | stage('Maven Compile') { |
|  | steps { |
|  | echo '----------------- This is a compile phase ----------' |
|  | sh 'mvn clean compile' |
|  | } |
|  | } |
|  |  |
|  | stage('Maven Test') { |
|  | steps { |
|  | echo '----------------- This is a compile phase ----------' |
|  | sh 'mvn clean test' |
|  | } |
|  | } |
|  |  |
|  | stage('Maven Build') { |
|  | steps { |
|  | echo '----------------- This is a build phase ----------' |
|  | sh 'mvn clean package -DskipTests' |
|  | } |
|  | } |
|  |  |
|  | stage('Docker Build') { |
|  | steps { |
|  | echo '----------------- This is a build docker image phase ----------' |
|  | sh ''' |
|  | docker image build -t ecom-webservice . |
|  | ''' |
|  | } |
|  | } |
|  |  |
|  | stage('Docker Deploy') { |
|  | steps { |
|  | echo '----------------- This is a docker deploment phase ----------' |
|  | sh ''' |
|  | (if [ $(docker ps -a | grep ecom-webservice | cut -d " " -f1) ]; then \ |
|  | echo $(docker rm -f ecom-webservice); \ |
|  | echo "---------------- successfully removed ecom-webservice ----------------" |
|  | else \ |
|  | echo OK; \ |
|  | fi;); |
|  | docker container run --restart always --name ecom-webservice -p 8081:8081 -d ecom-webservice |
|  | ''' |
|  | } |
|  | } |
|  | } |
|  | } |

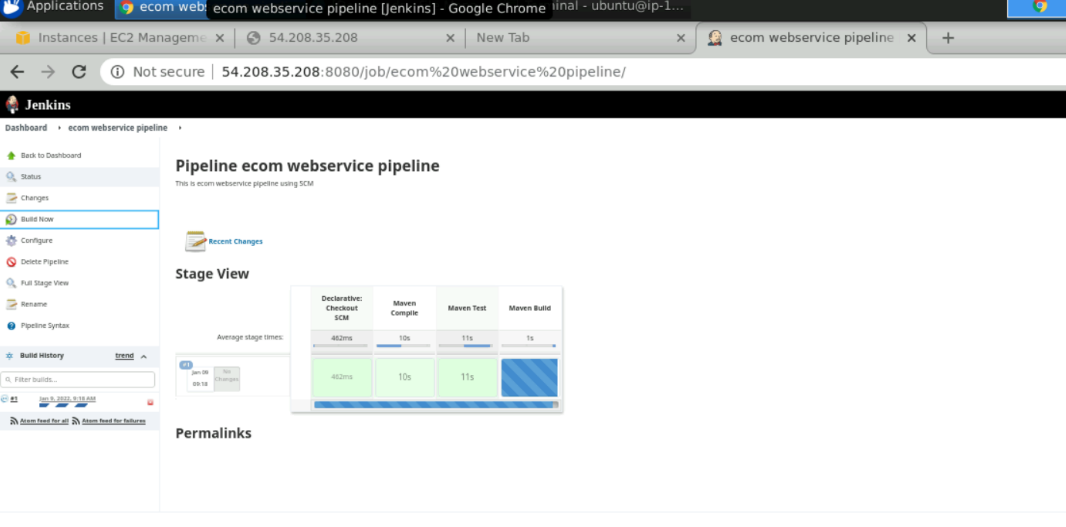
* © 2022 GitHub, Inc.

This Jenkins file uses docker to build image, for tis Docker file is there in ecom webservice repository

Content of Docker fle is

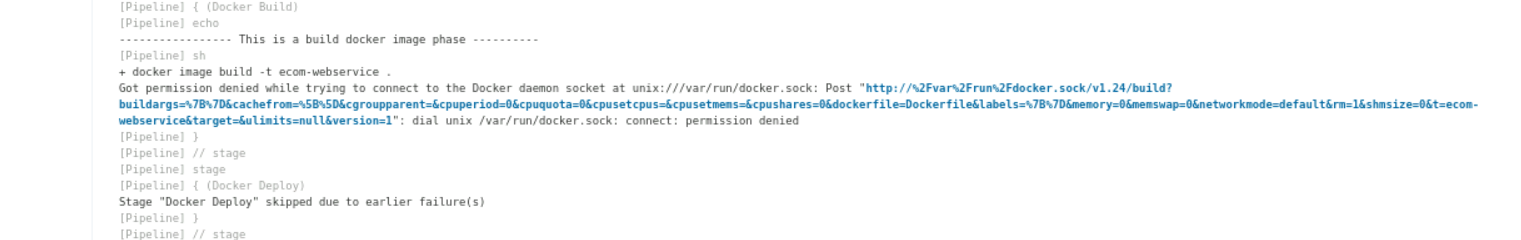
|  |
| --- |
|  |
| FROM openjdk:8-jdk |
|  | COPY ./target/ecom-webservice-0.0.1-SNAPSHOT.jar ecom-webservice-0.0.1-SNAPSHOT.jar |
|  | CMD ["java" ,"-jar","ecom-webservice-0.0.1-SNAPSHOT.jar"] |
|  | RUN echo "jenkins ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers |
|  |  |

1. Click “build now” option in left menu, following pipeline build will start using Jenkins script file in ecom webservice repository





1. Pipeline build failed because of permission denied using jenkins,

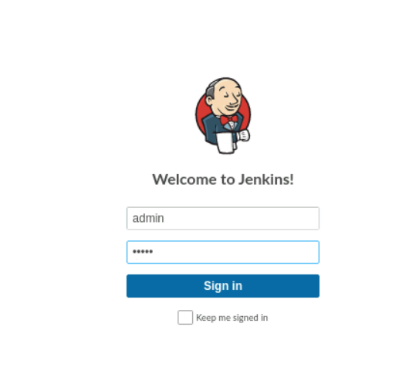


For this run following commands at prompt

Sudo usermod -a -G docker jenkins

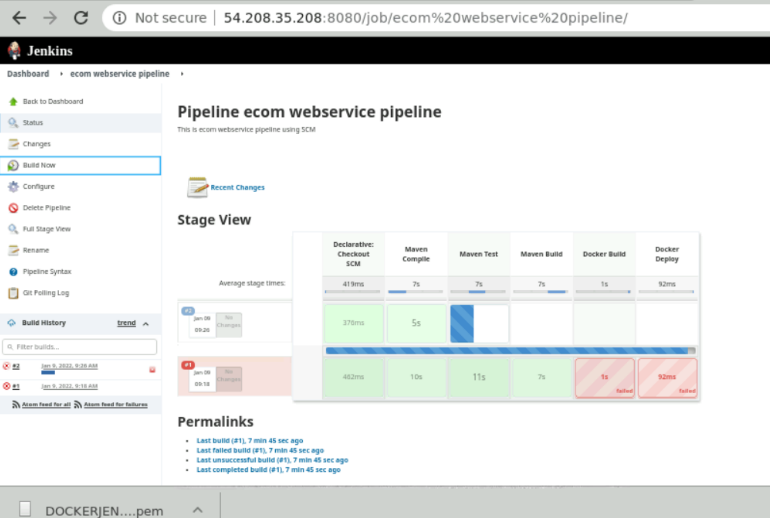
Sudo service jenkins restart

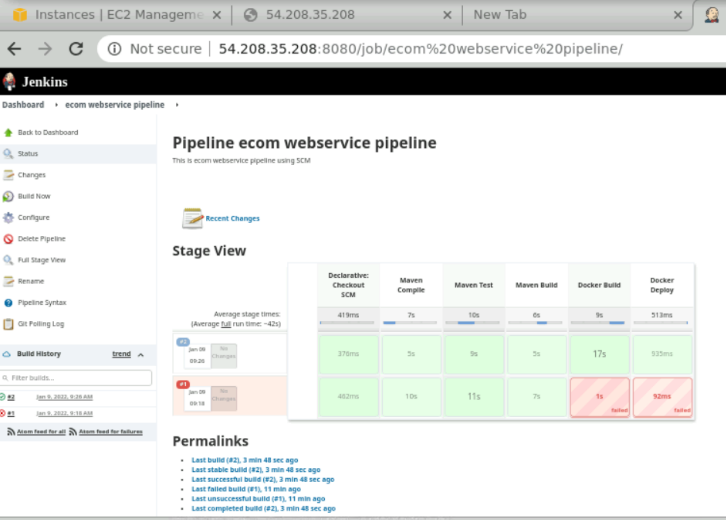




Login to jenkins website

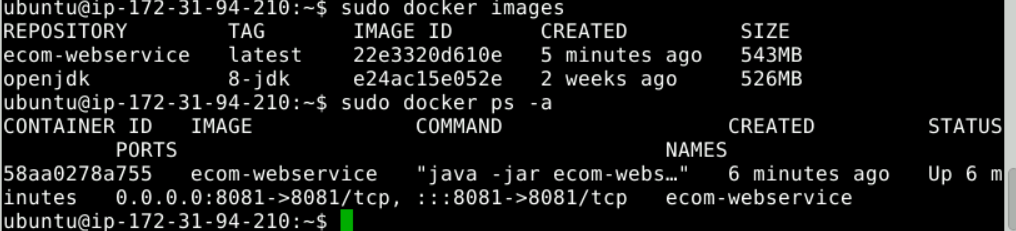
1. Click ”build now” again





Success.

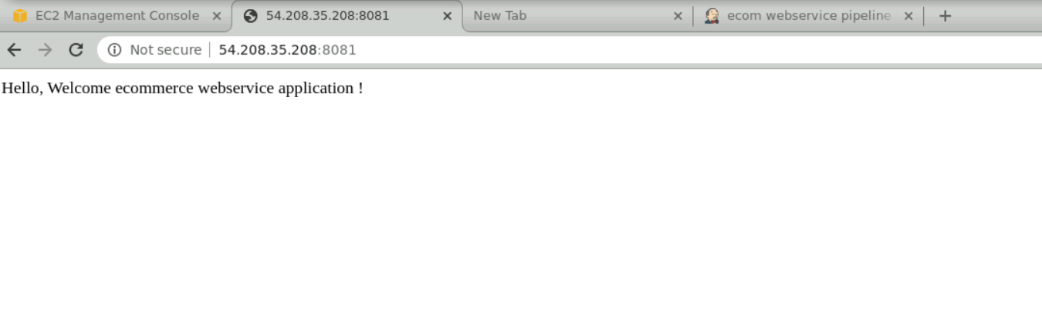
1. Check the images



Running at port 8081, must add in security group

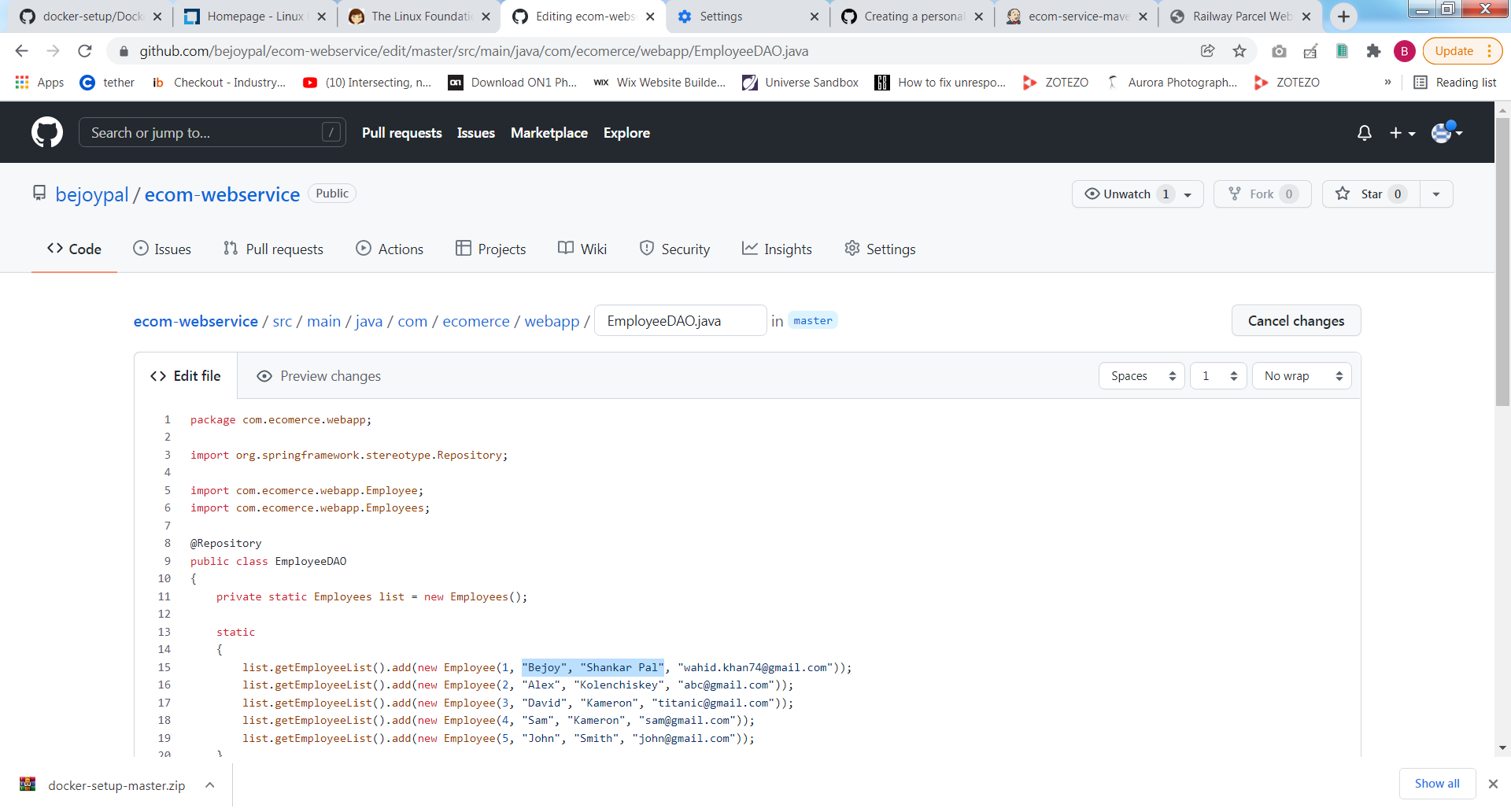


1. Check for ecom web service in browser using port 8081

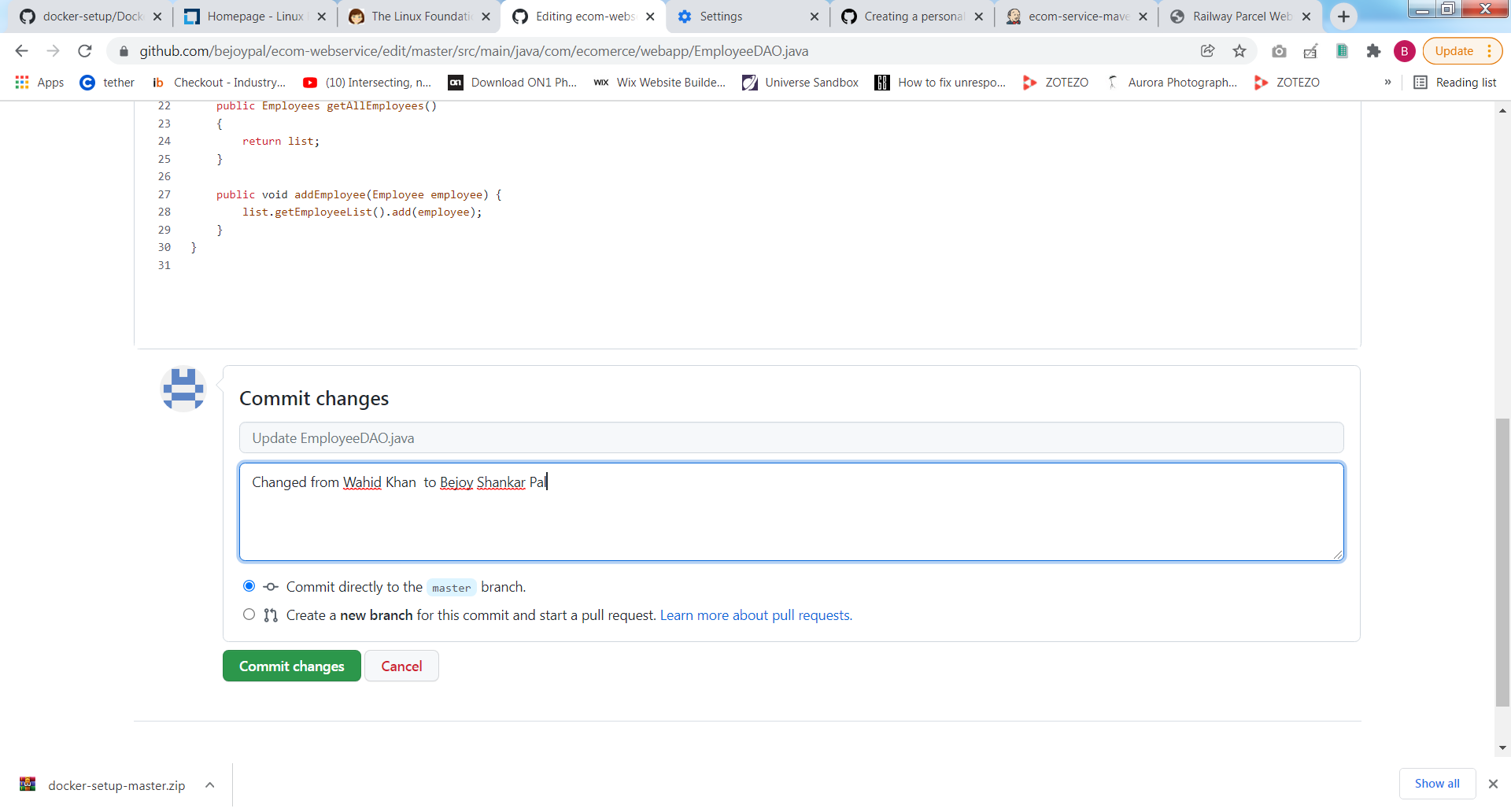




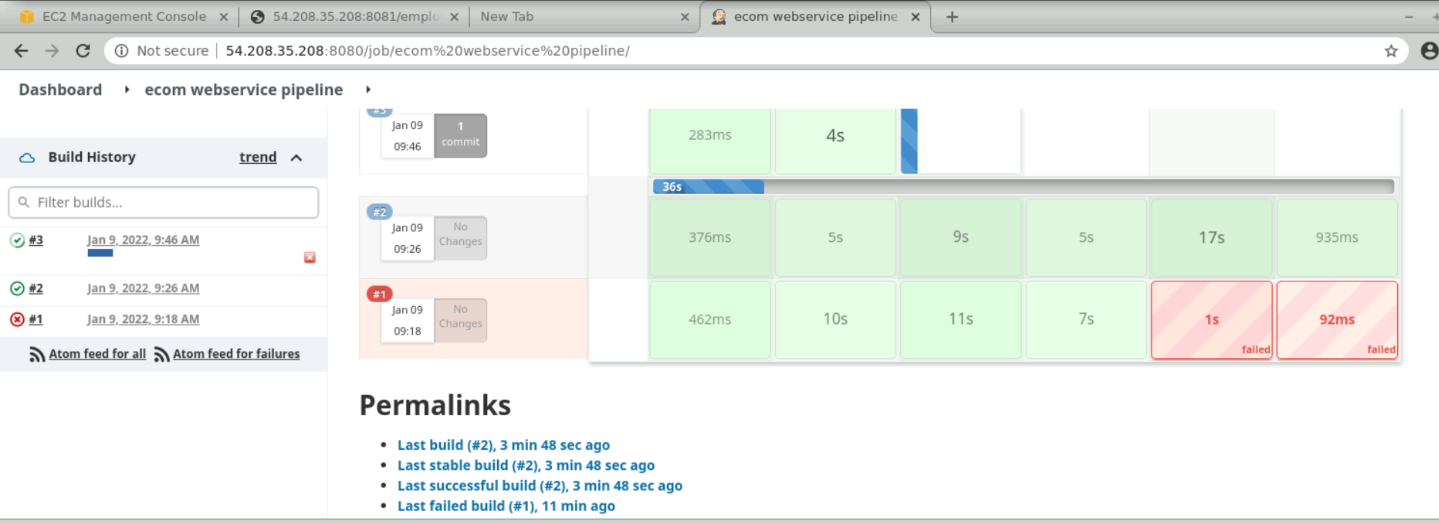
1. Now change the file EmployeeDAO.java in Github, just change the name “Wahid” to “Bejoy” and “Khan” to “Shankar Pal”

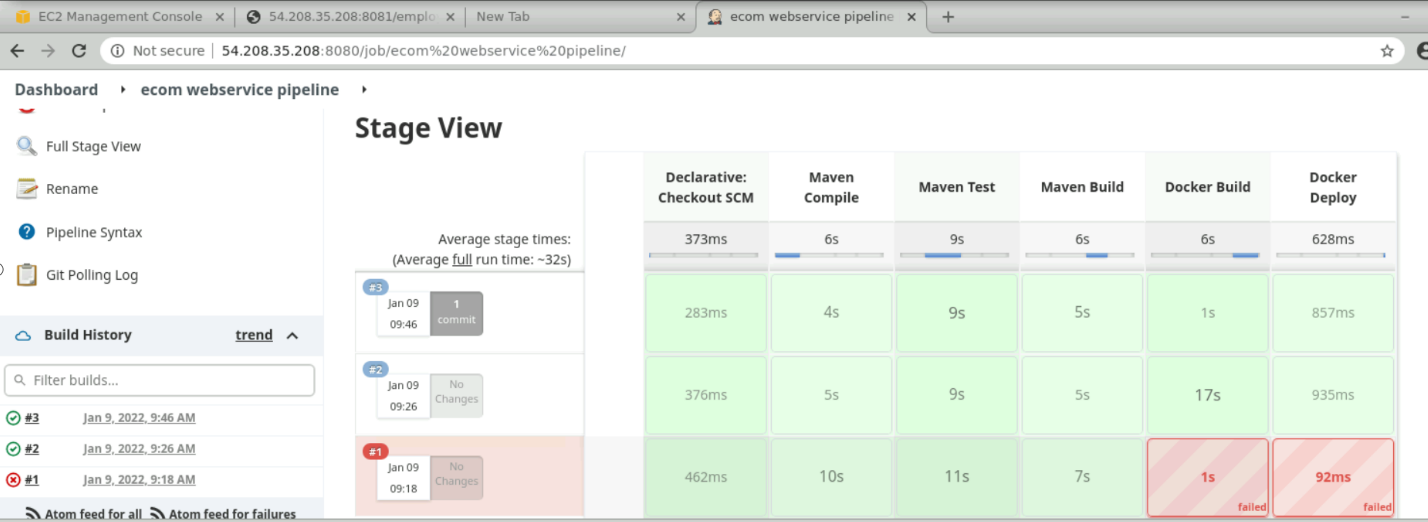


Save the file in Github



1. Now check the Jenkins website, it should start building again due to SCM schedule in Jenkins groovy script file which is \* \* \* \* \*, means running every minute , every hour





Success, next check webservice, using port 8081, to see changes relected or not.



Success, changes reflected through automatic build and deploy by Jenkins.

**Concepts used**

1. AWS EC2 instance creation with Ubuntu OS, its connection from client using MobaXterm using EC2 insance certificate
2. Installation docker, building of image, running of container using docker image and other operations
3. Installation of Jenkins server, and running it for automating SCM checkout, maven build and docker build/running container/push with further complete automation of above steps using groovy script