**DevOps Project**

**Objective:** You have been Hired Sr. Devops Engineer in Abode Software. They want to implement Devops Lifecycle in their company. You have been asked to implement this lifecycle as fast as possible. Abode Software is a product-based company, their product is available on this GitHub link.

<https://github.com/hshar/website.git>

Following are the specifications of the lifecycle:

1. Git Workflow has to be implemented

2. Code Build should automatically be triggered once commit is made to master branchor develop branch.

If commit is made to master branch, test and push to prod

If commit is made to develop branch, just test the product, do not push to prod

3. The Code should be containerized with the help of a Dockerfile. The Dockerfile should be built every time there is a push to Git-Hub. Use the following pre-built container for your application: hshar/webapp The code should reside in '/var/www/html'

4. Once the website is built, you have to design a test-case, which will basically check if the website can be opened or not. If yes, the test should pass. This test has to run in headless mode, on the test server.

5. The above tasks should be defined in a Jenkins Pipeline, with the following Job

1 - Building Website Job

2 - Testing Website Job

3 - Push to Production

6. Since you are setting up the server for the first time, ensure the following file exists on both Test and Prod server in /home/ubuntu/config-management/status.txt. This file will be used by a third-party tool. This should basically have the info whether apache is installed on the system or not The content of this file, should be based on whether git is installed or not.

If apache is installed => Apache is Installed on this System"

If apache is not installed => "Apache is not installed on this System"

7. Create a Monitoring Service for the website on the Production server

Architectural Advice:

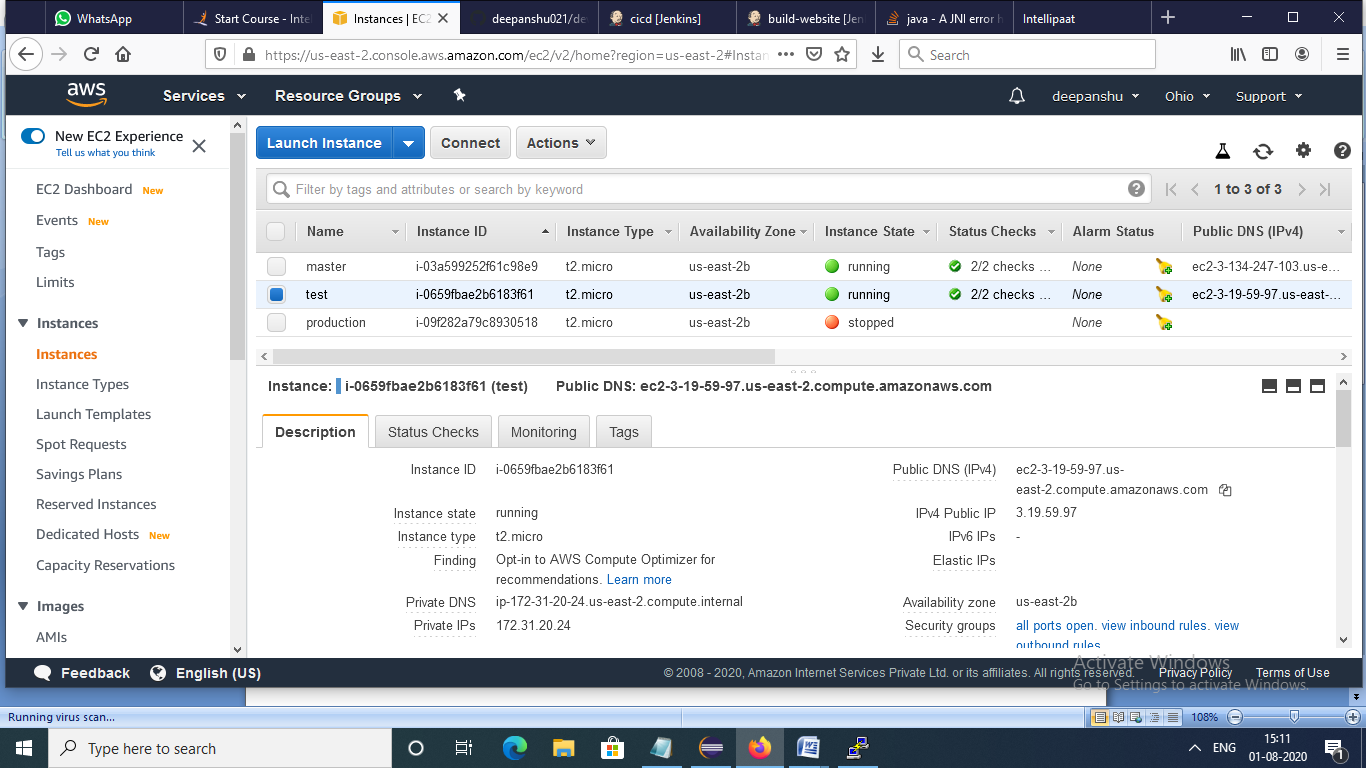
Create 3 servers on AWS "t2.micro"

Server1 - should have Jenkins Master, Puppet Master and Nagios Installed

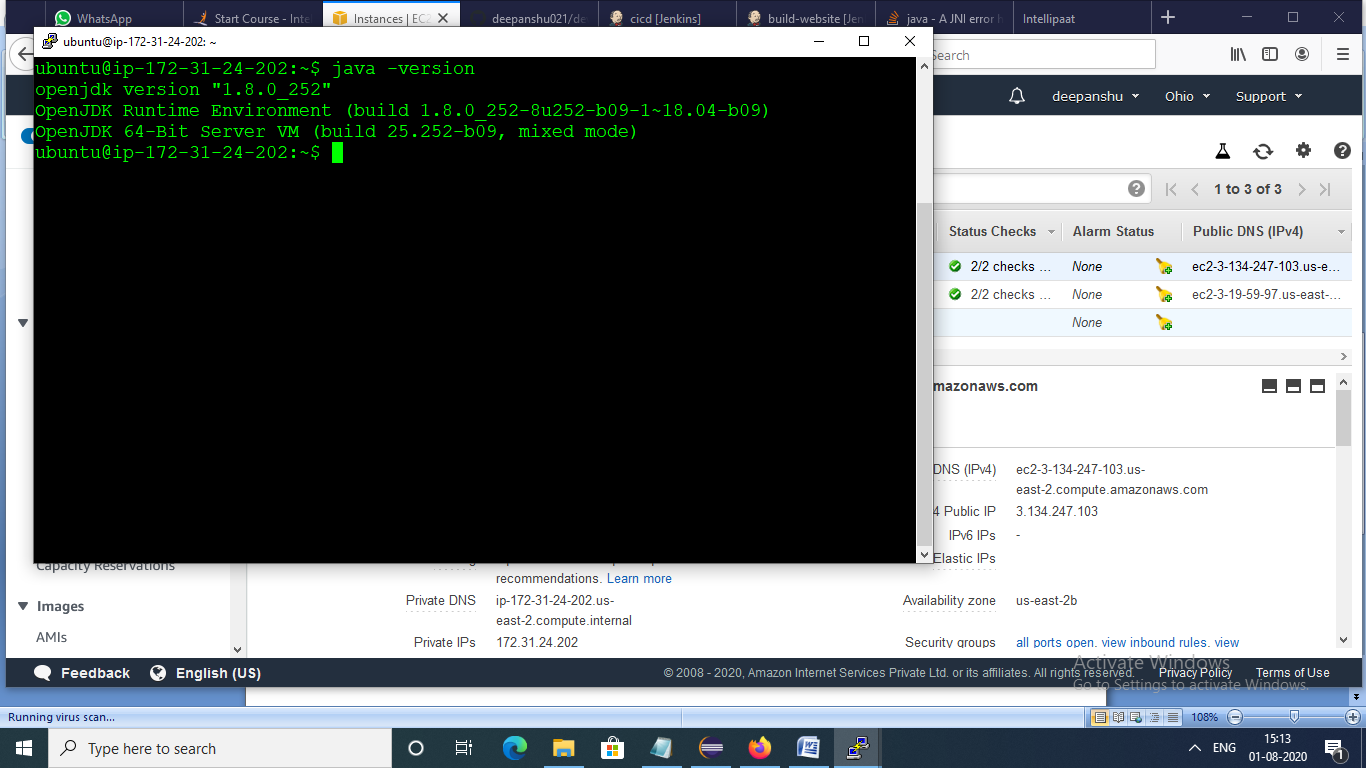
Server 2 - Testing Server, Jenkins Slave

Server 3 - Prod Server, Jenkins Slave

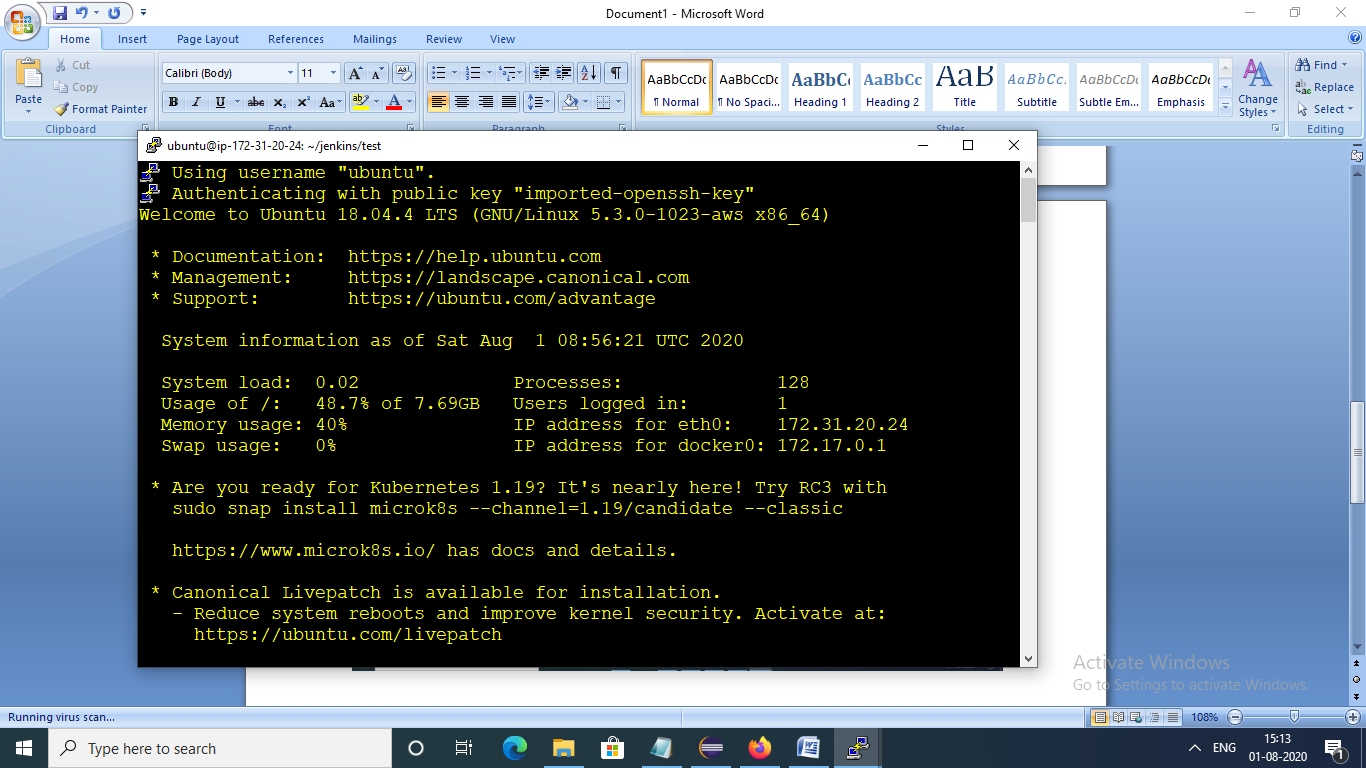
Creation of servers in AWS for master, test and production environment



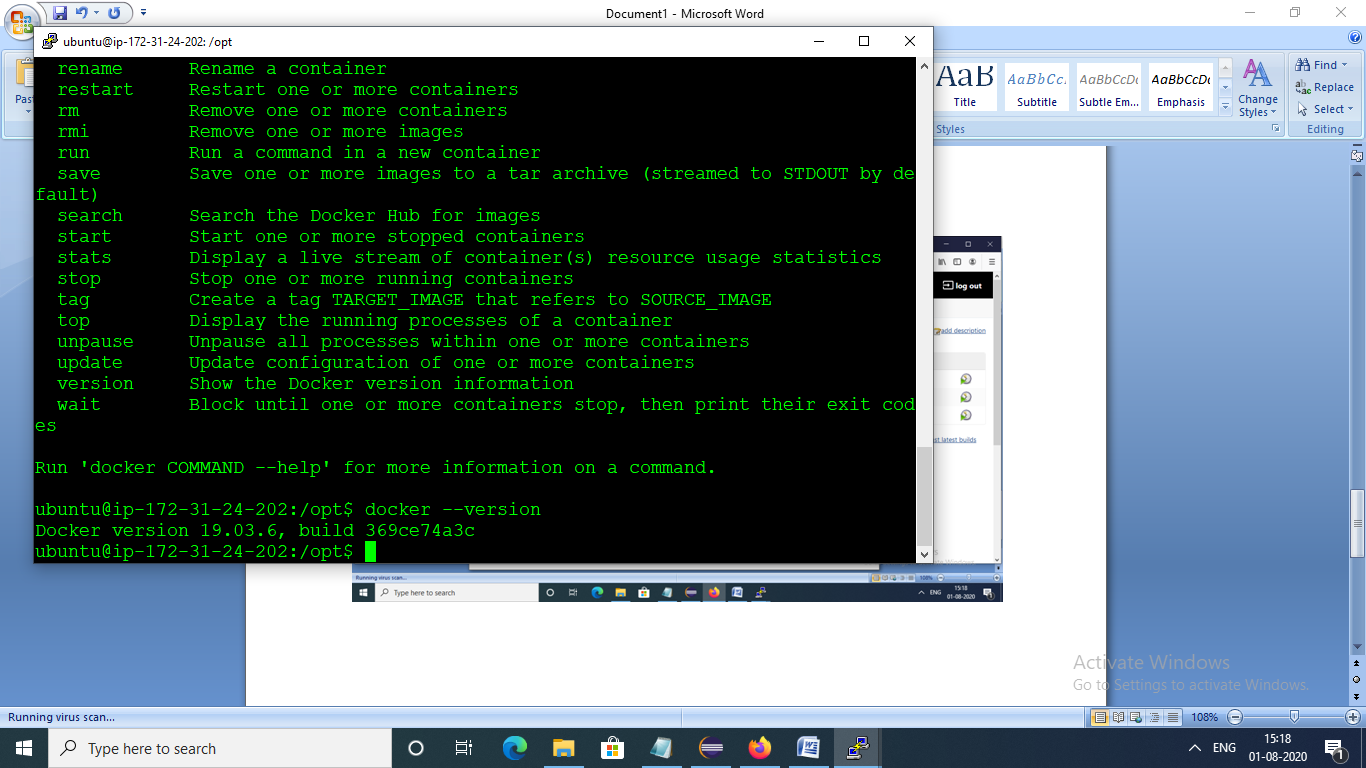
Master server



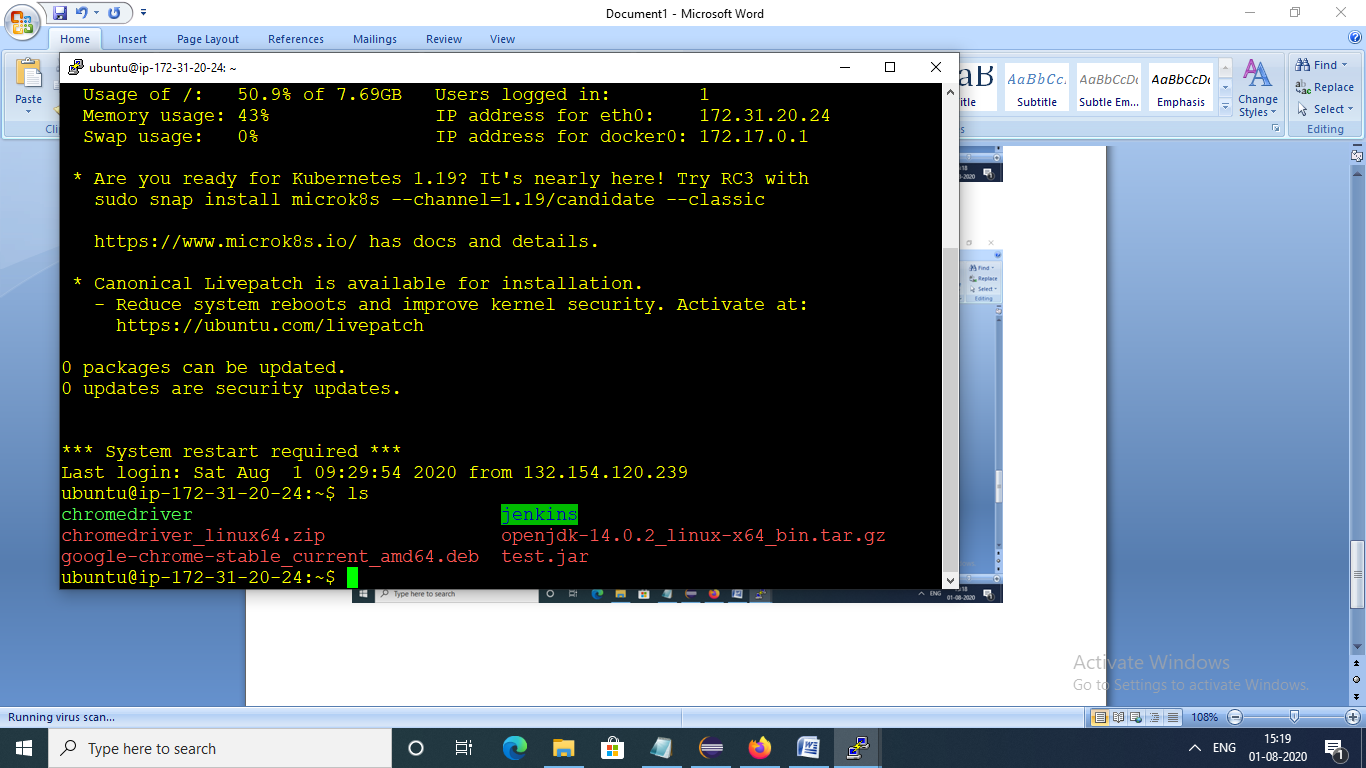
Testing server



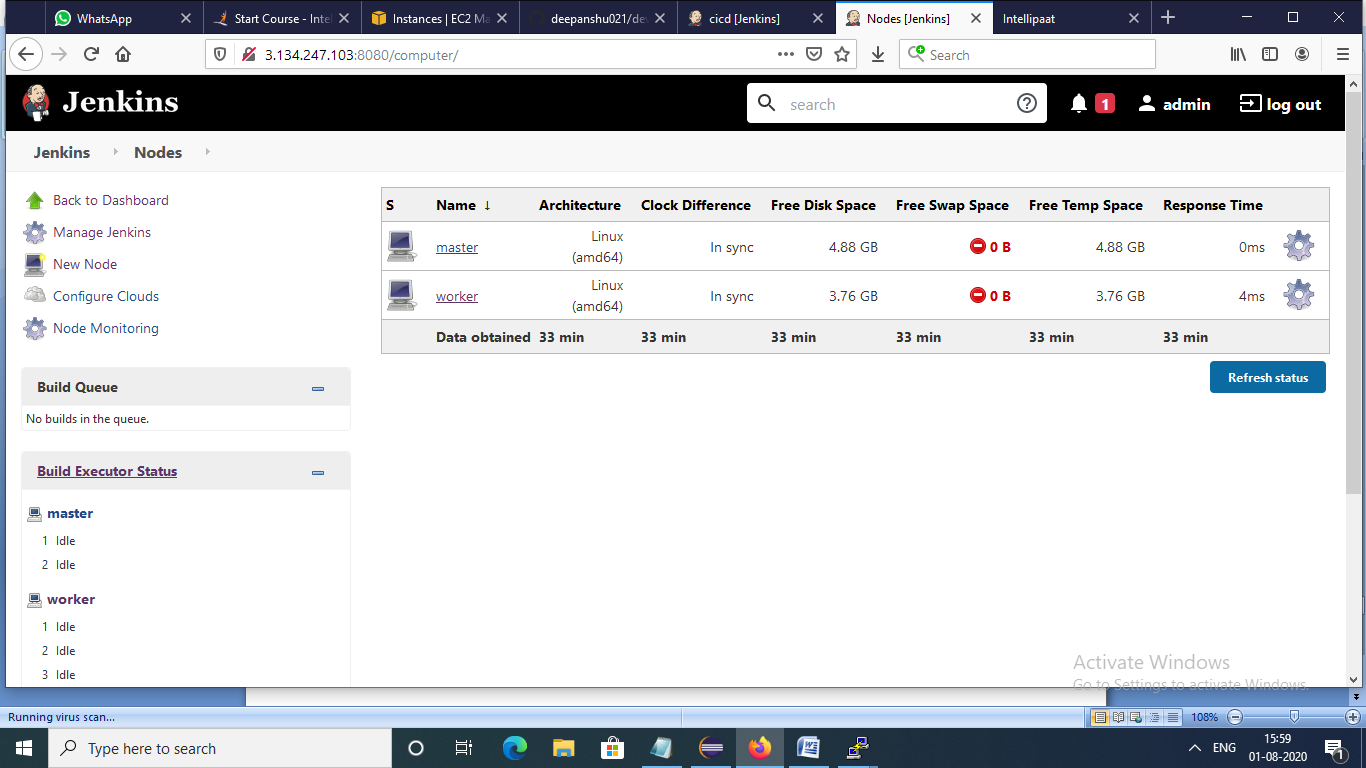
Installation of java Jenkins and docker on master as well as testing server



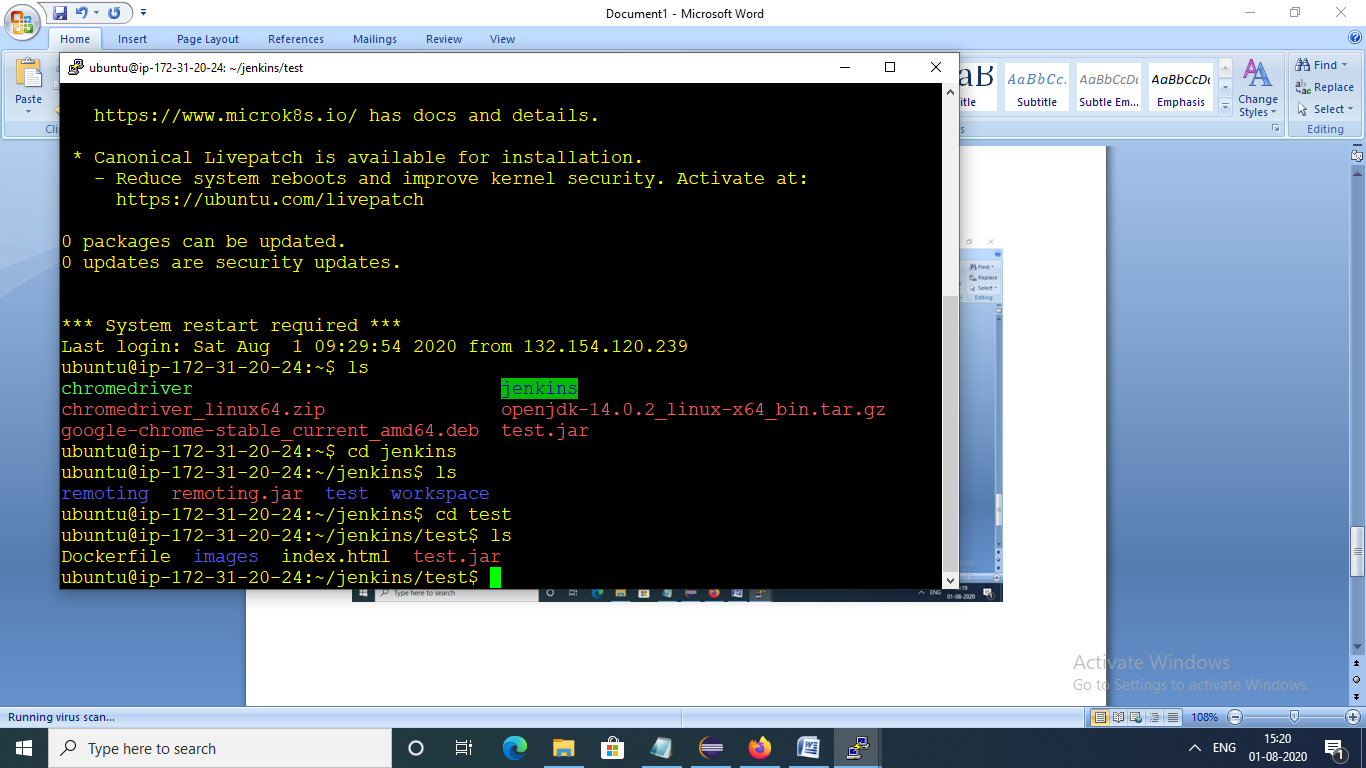
Testing server with connection with Jenkins and chrome driver installed and with git initialized with jar file



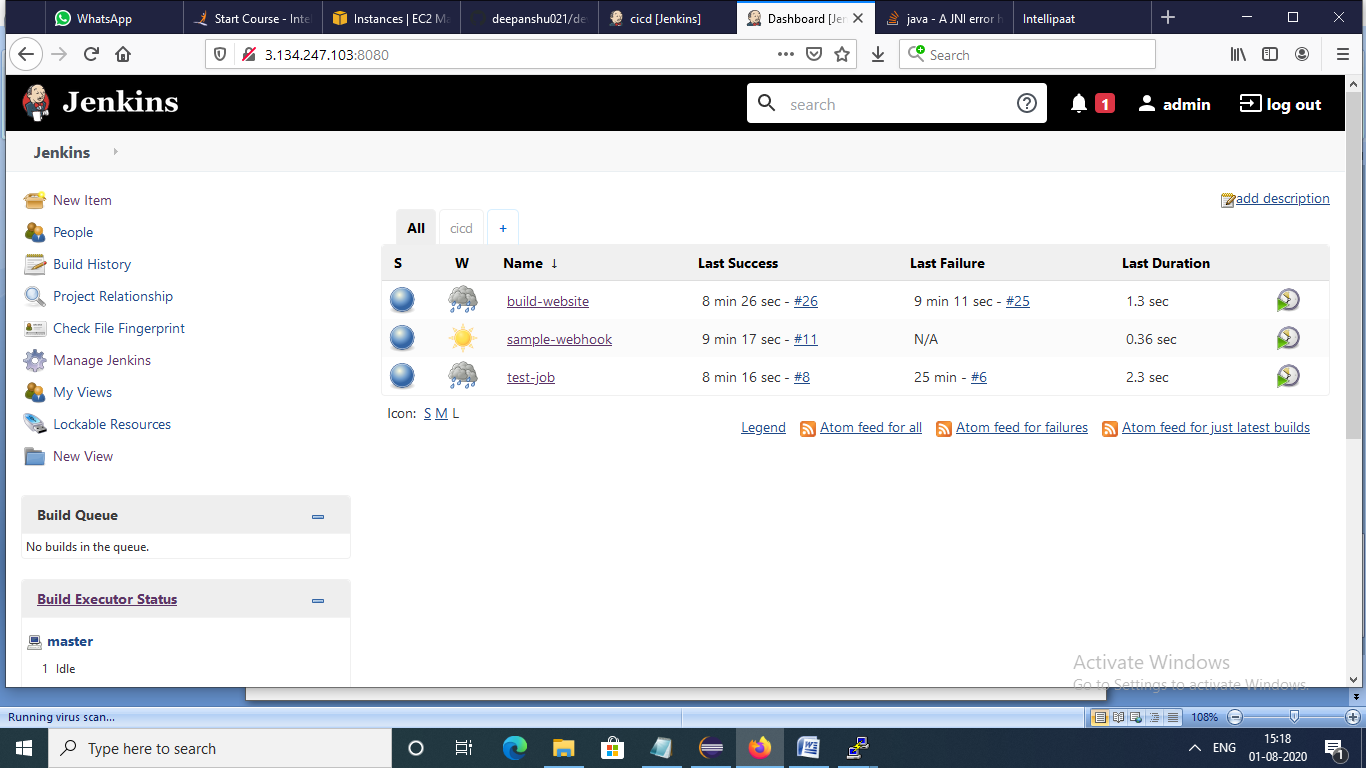
Node connection in jenkins



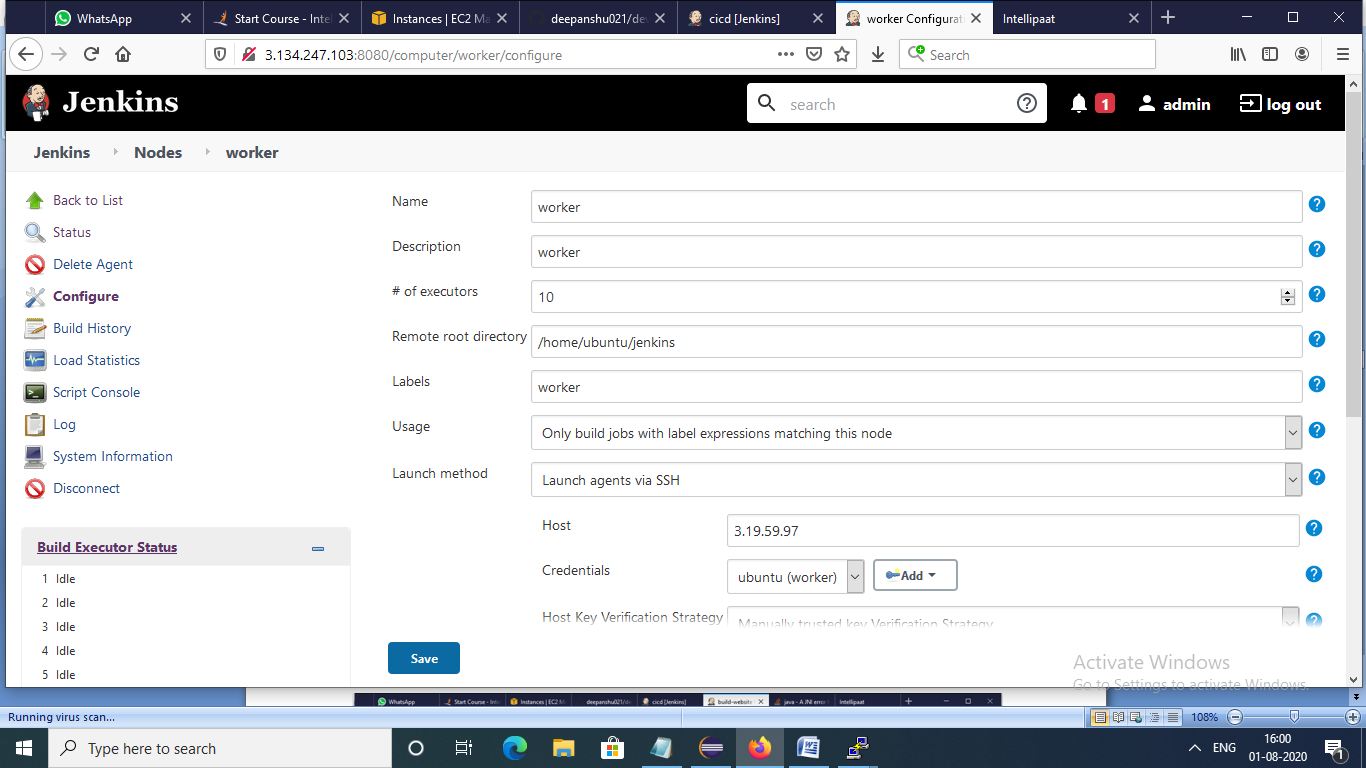
All files in testing server

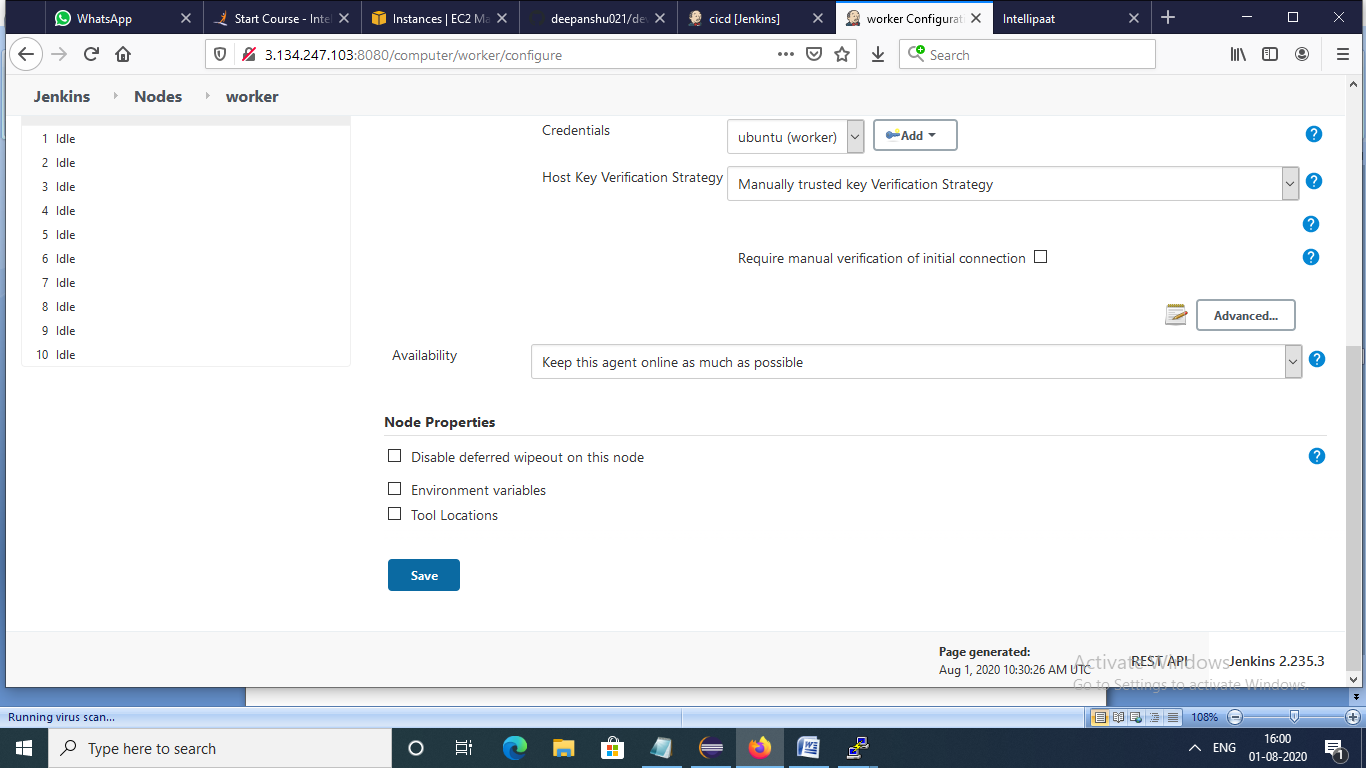


Successful builds in jenkins

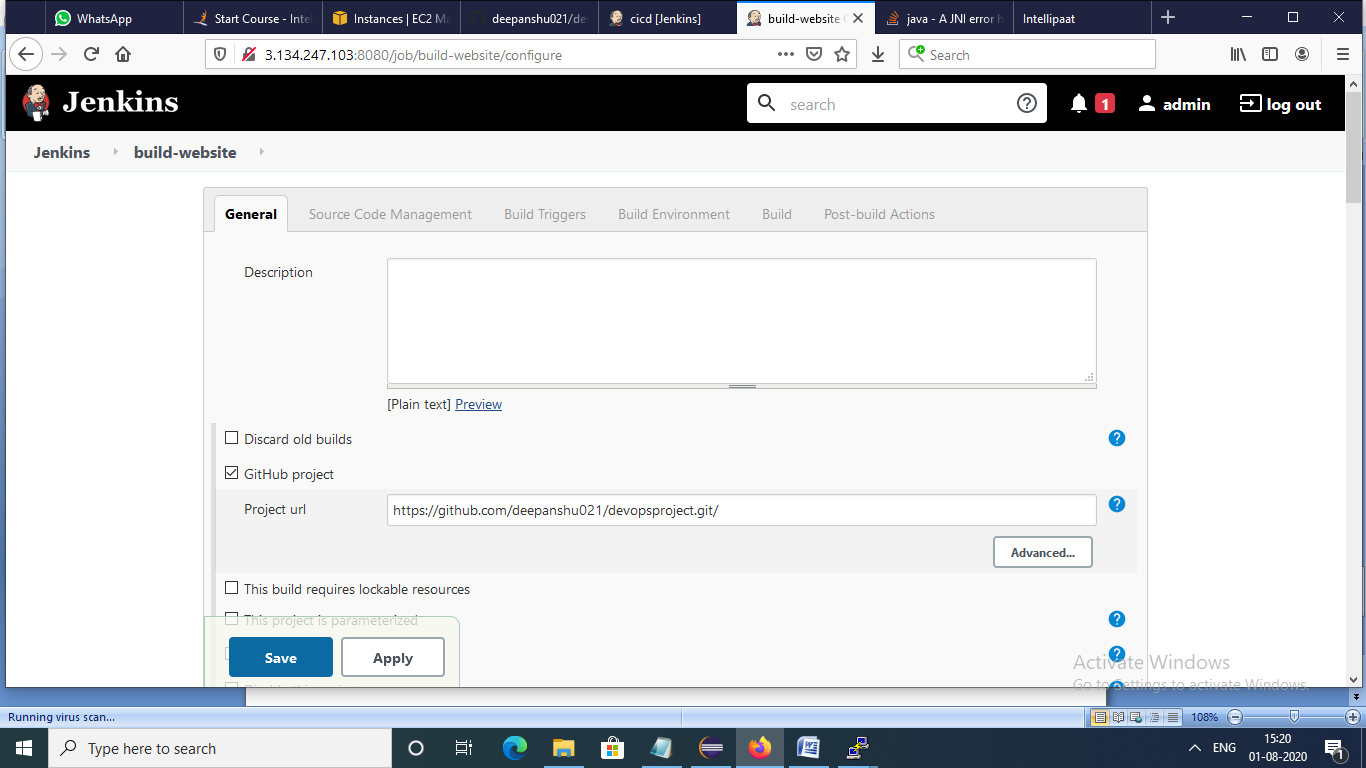


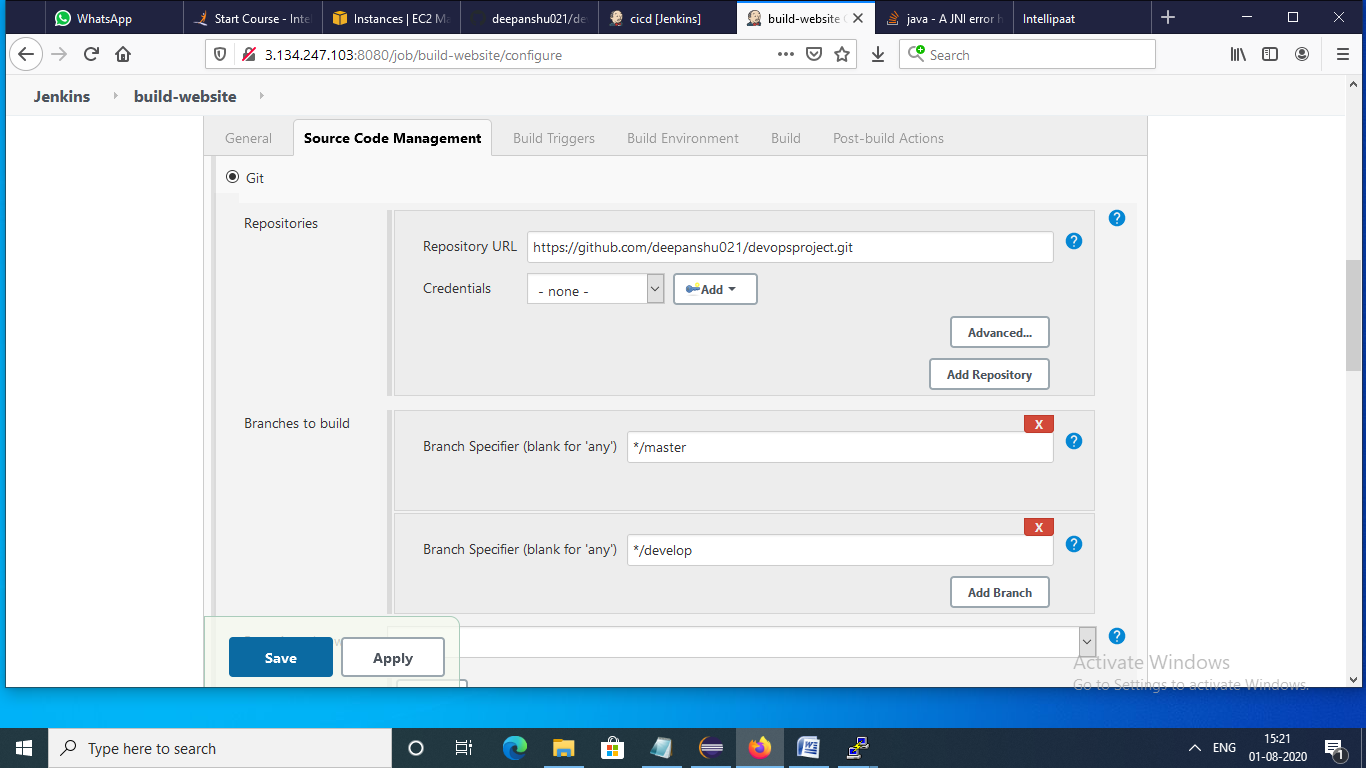
Configuration of node in Jenkins which connected using SSH

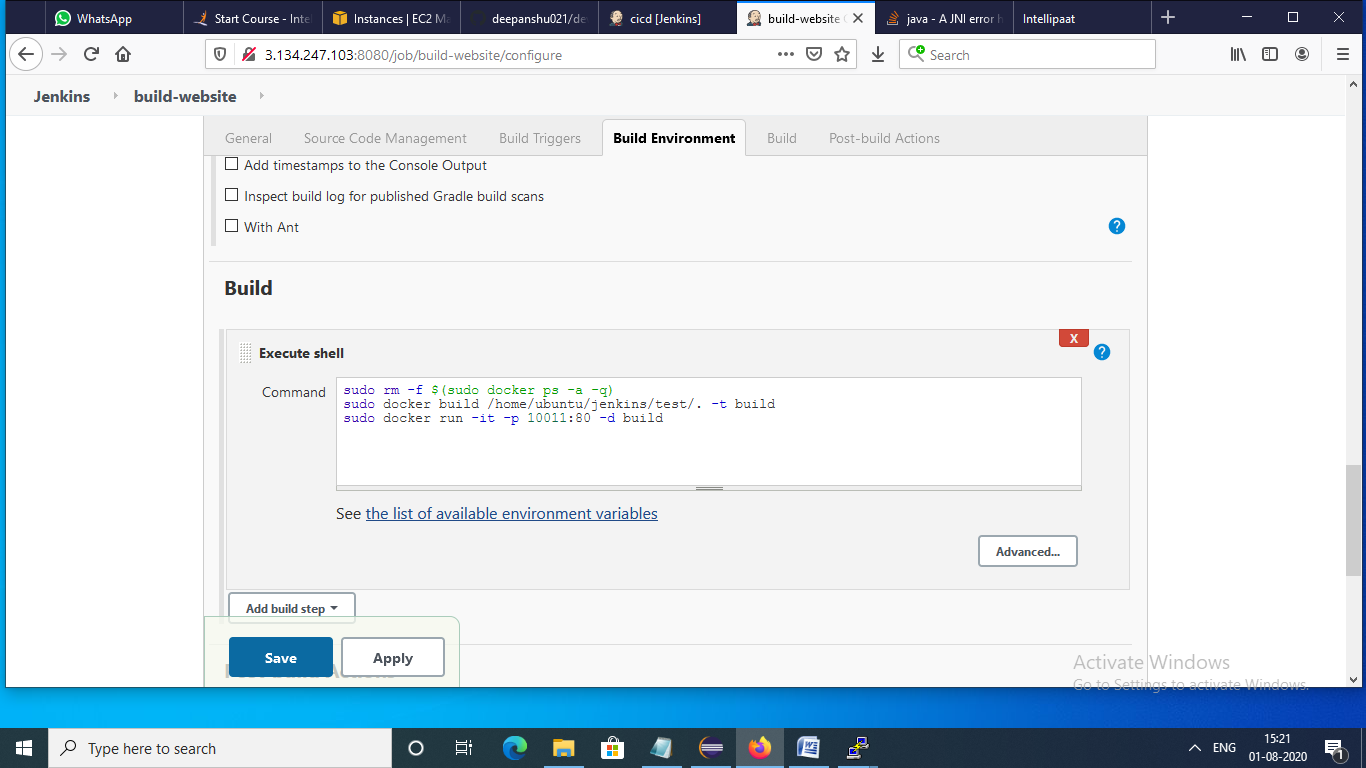




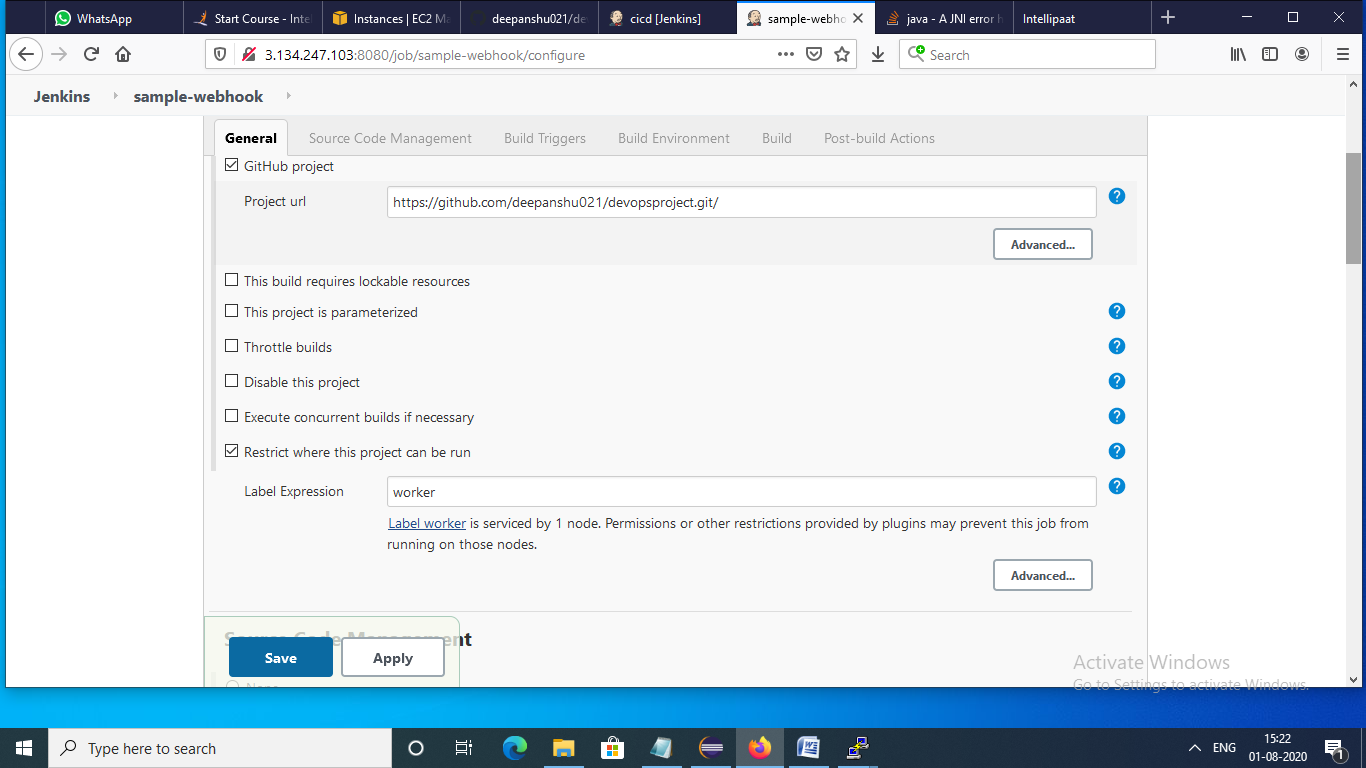
Configuration of build – build-website

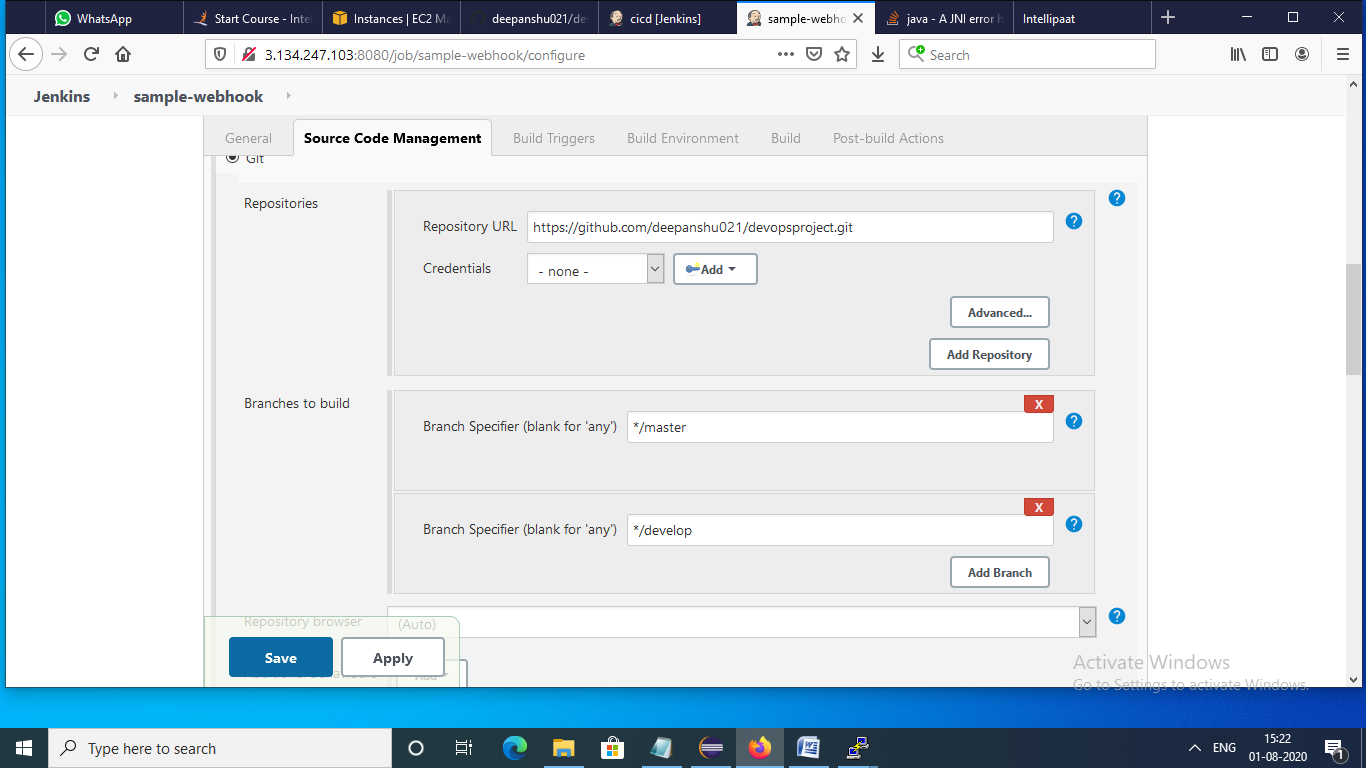


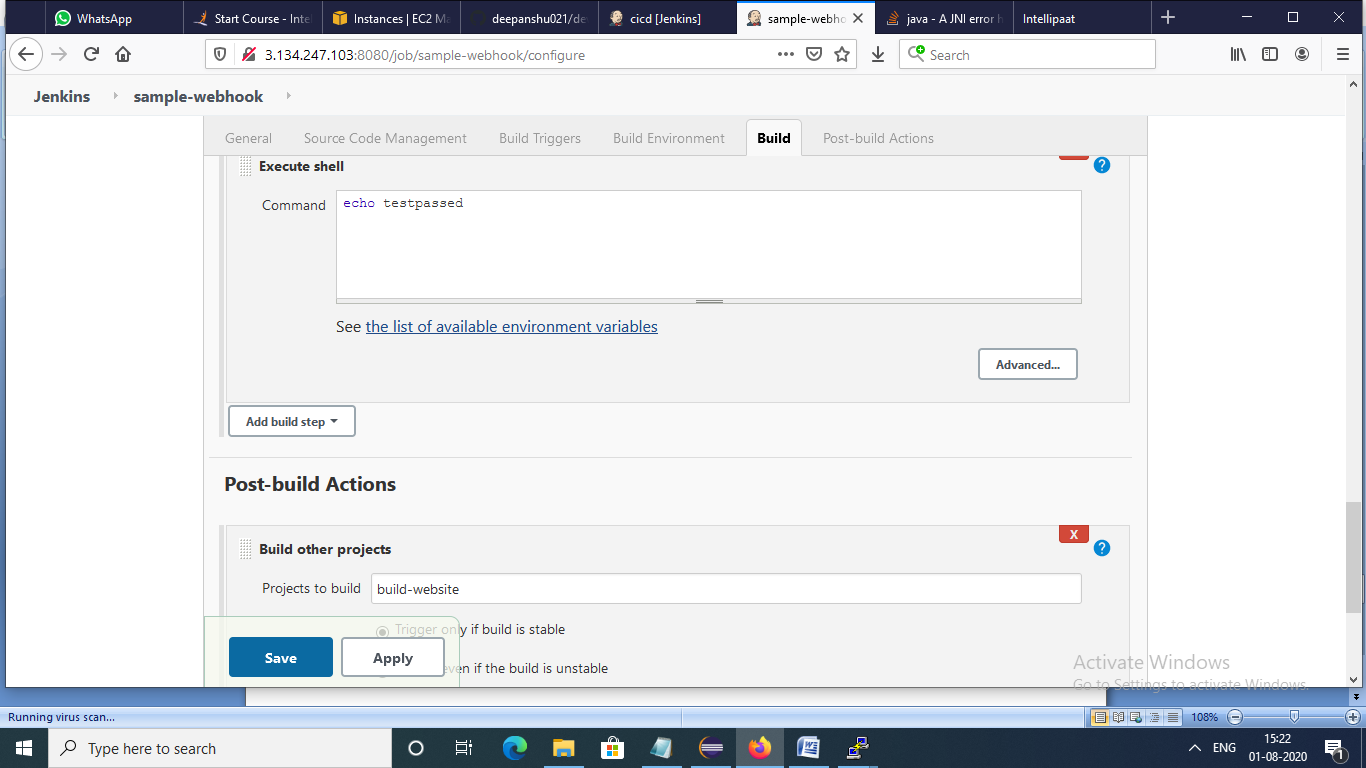




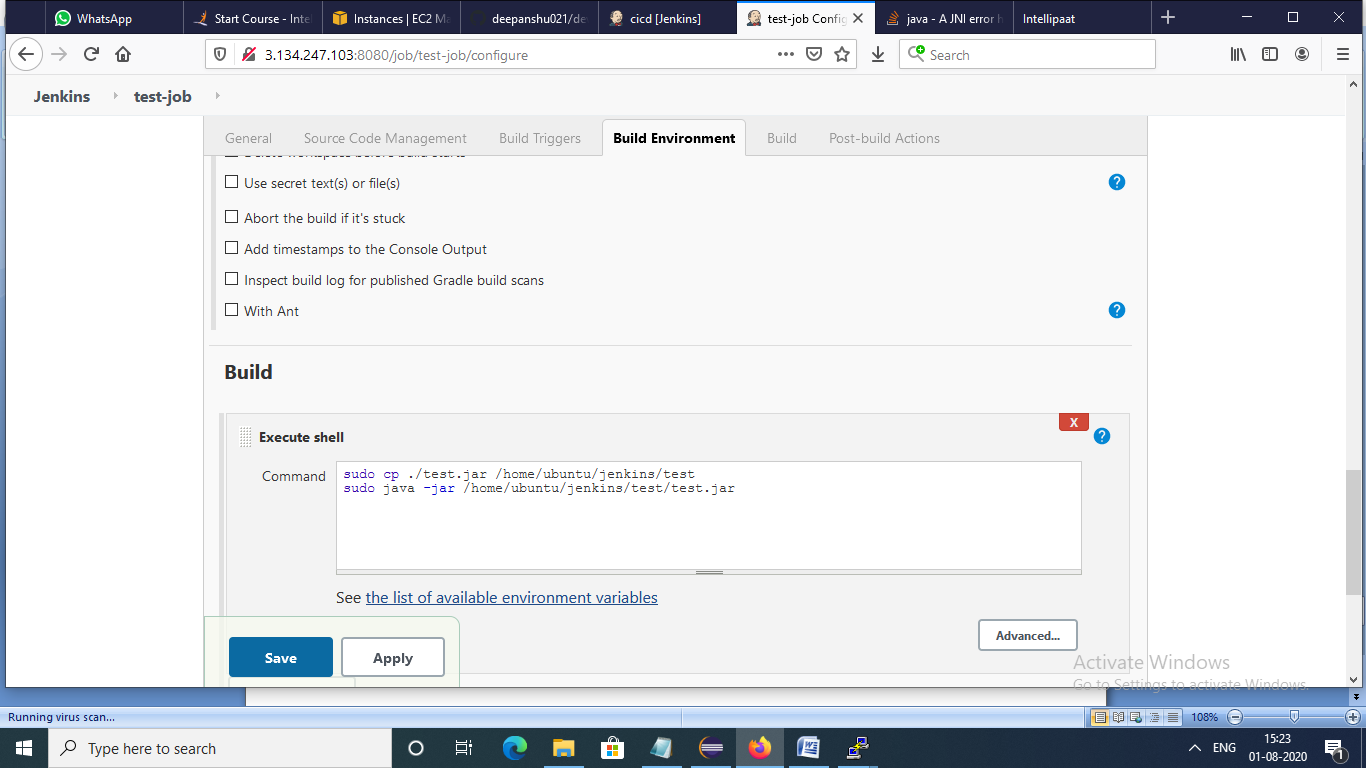
Configuration of build -sample webhook



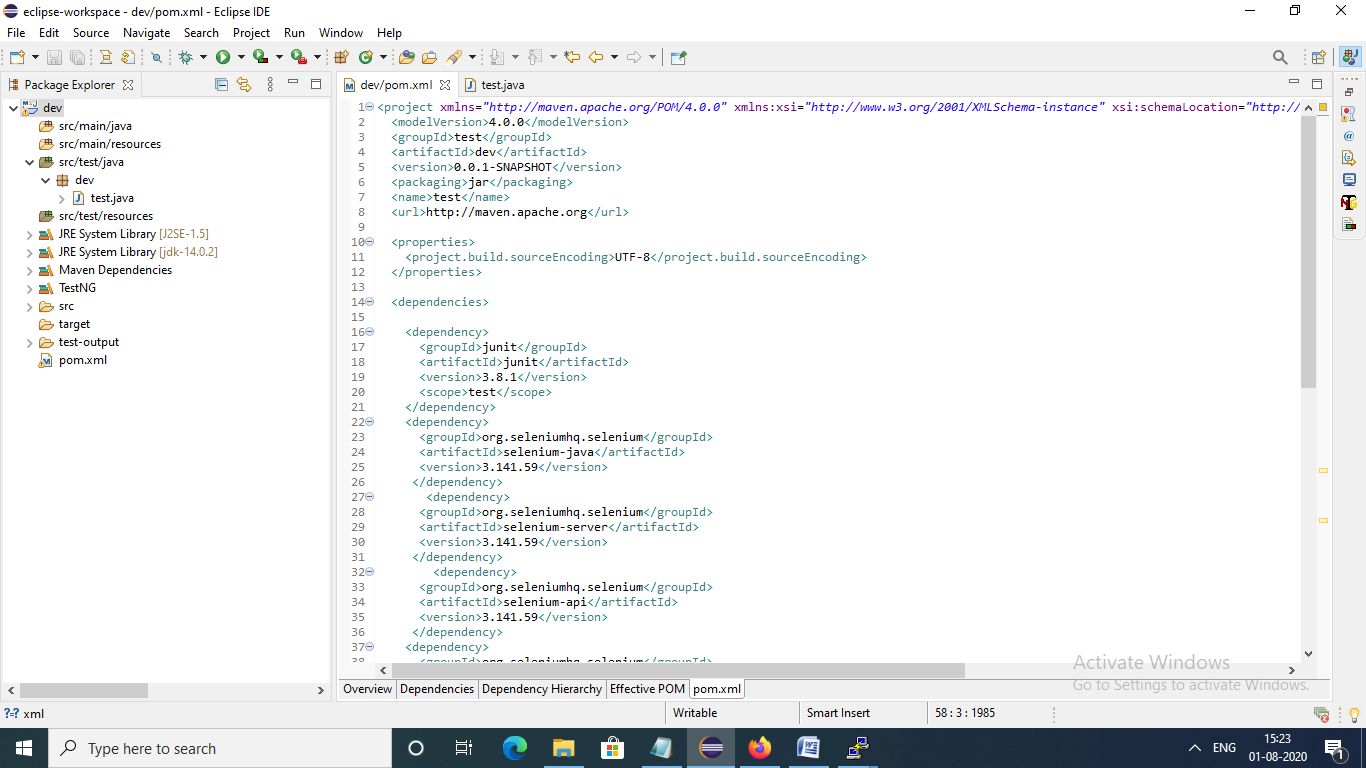




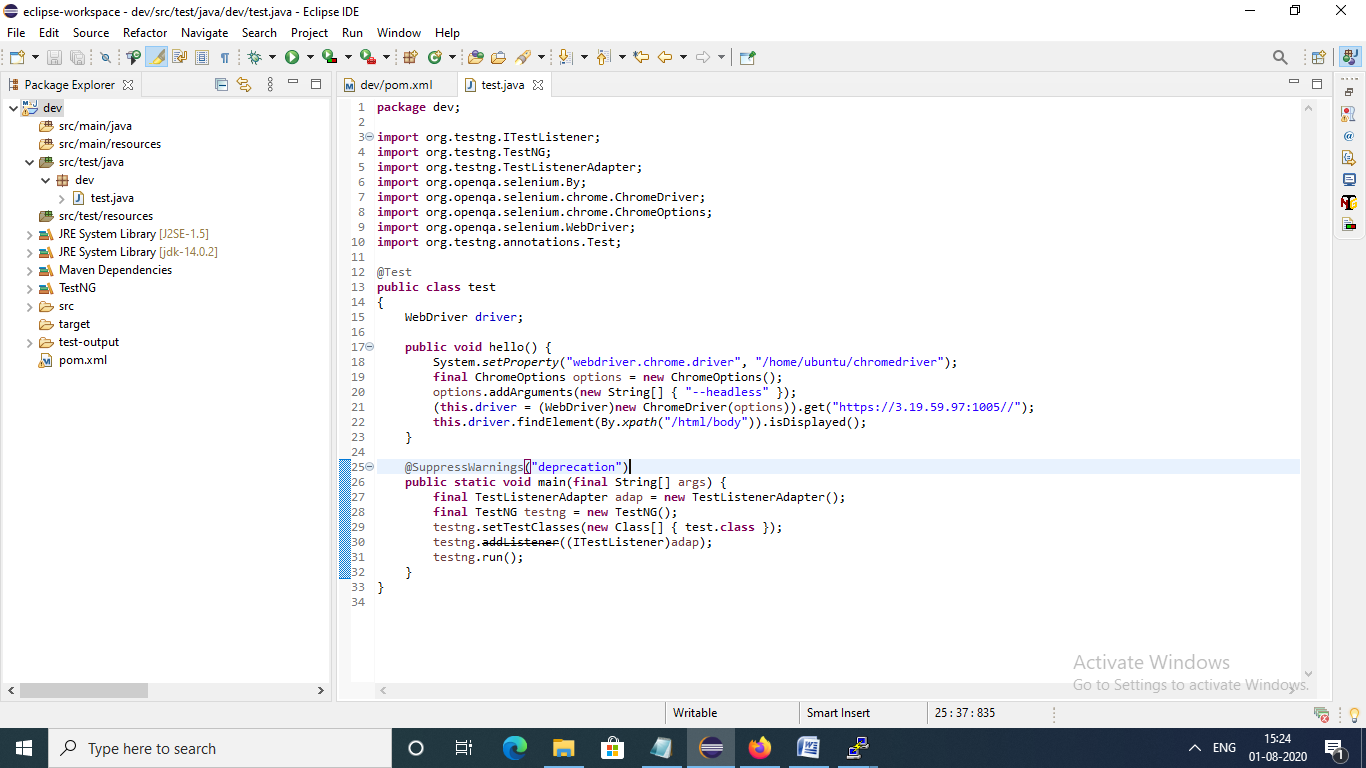
Configuration of build- test job



POM.XML file in eclipse



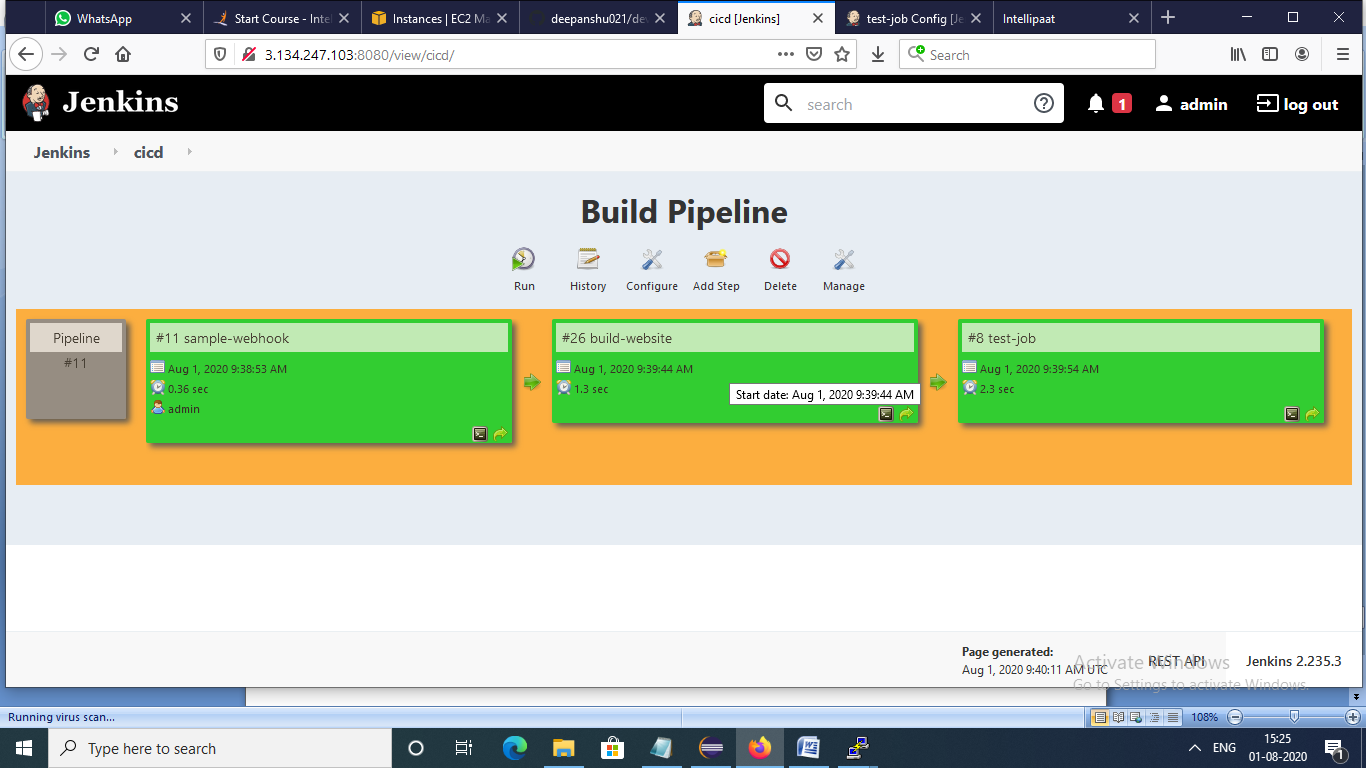
Test.java file in eclipse



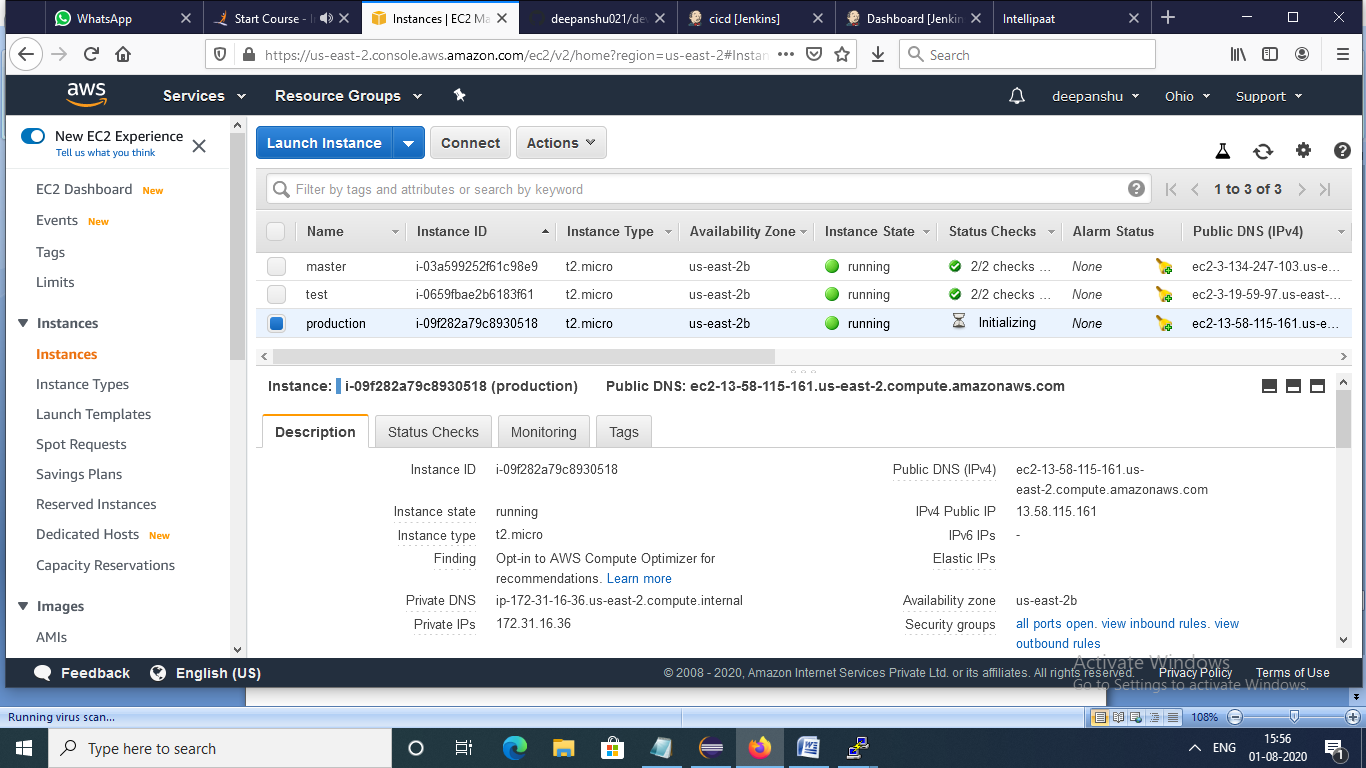
Successful build of website using dockerfile

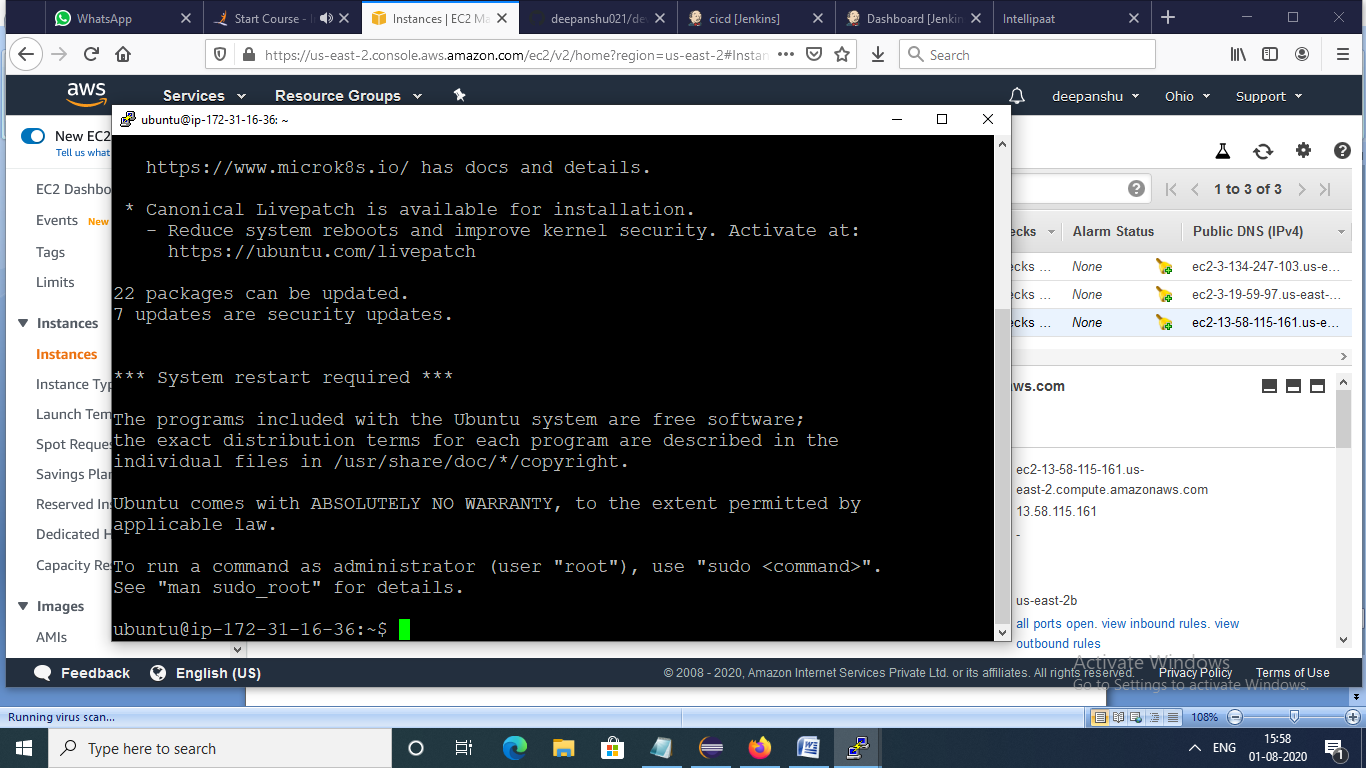


CICD pipeline

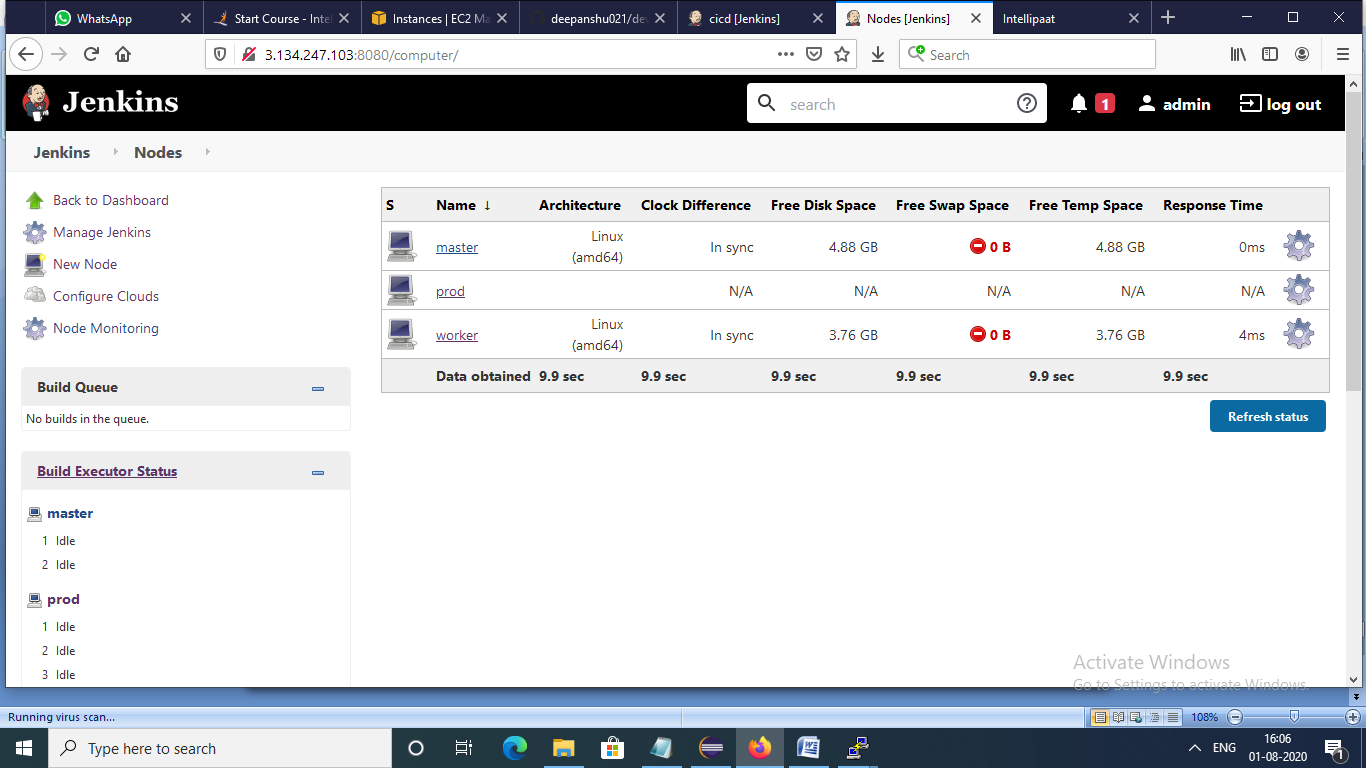


Production server starts

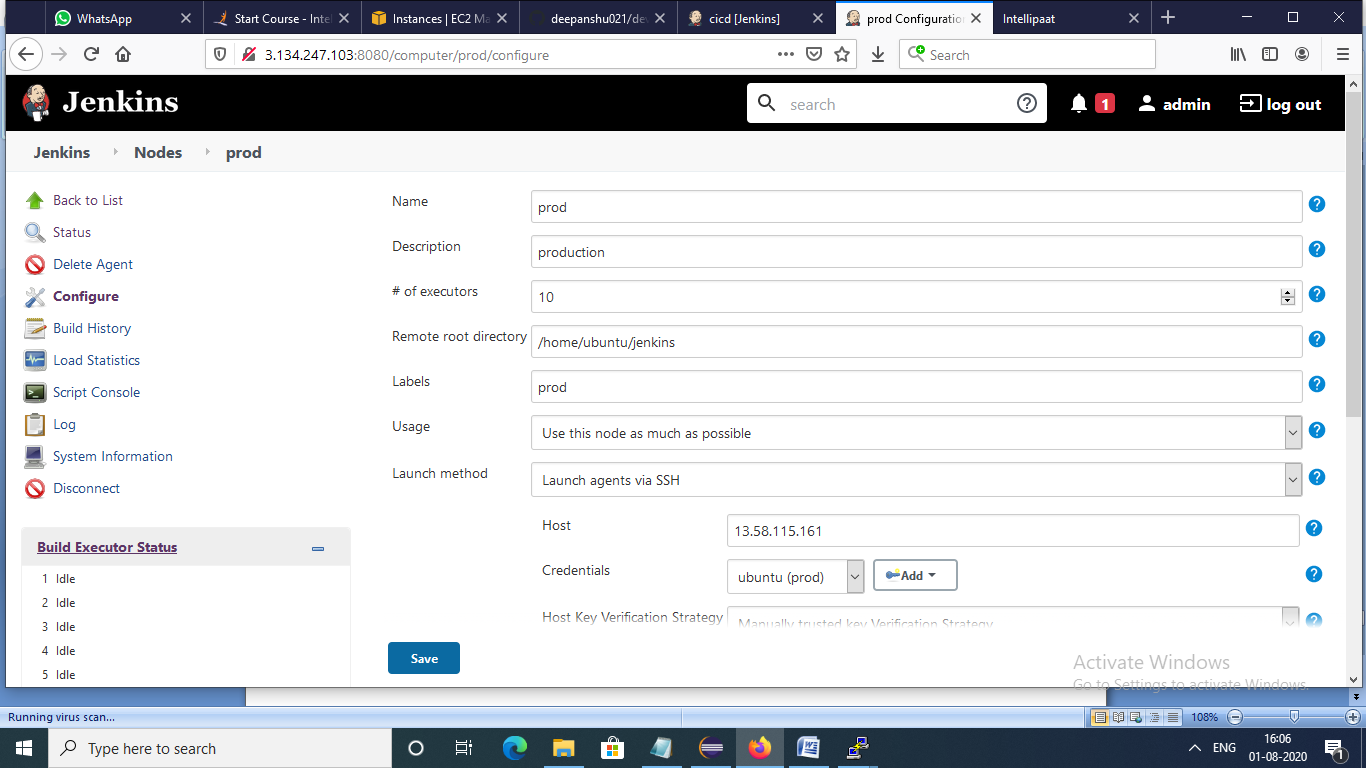


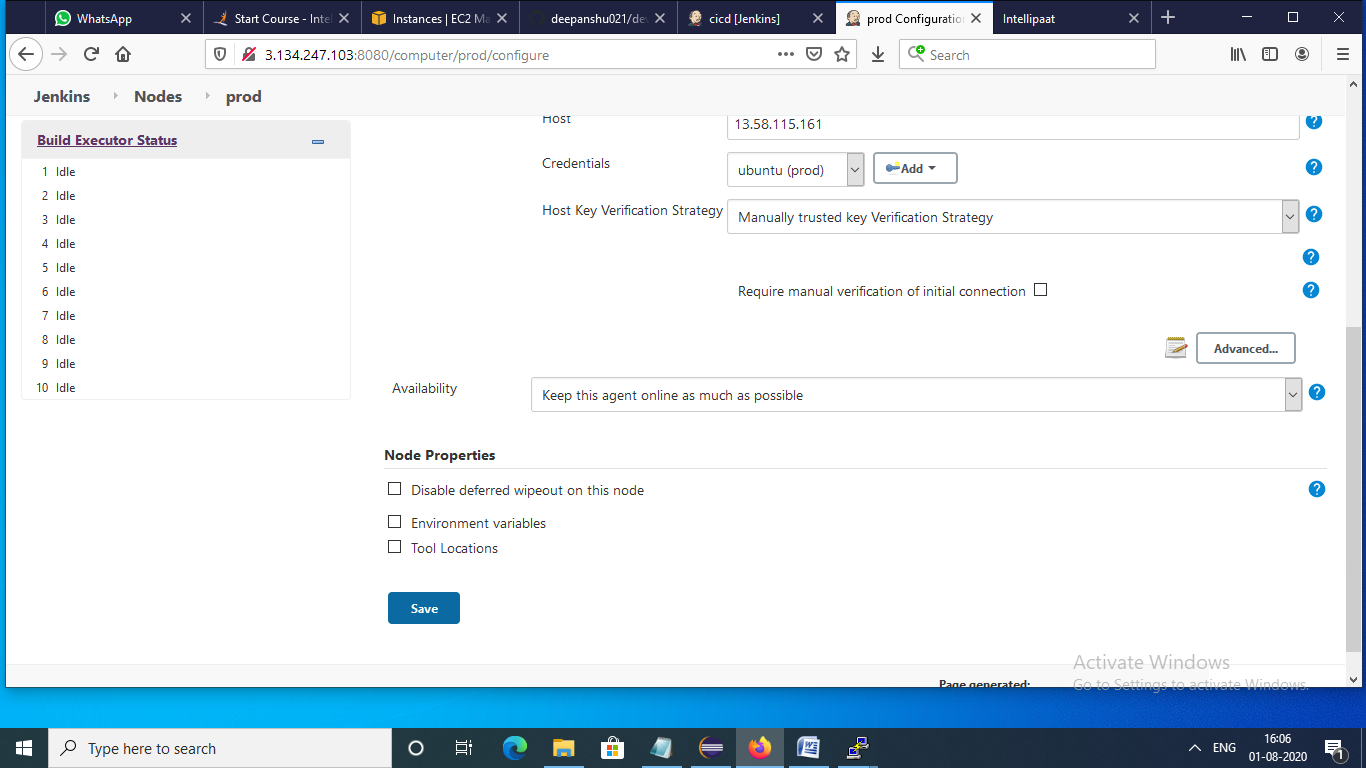


Production server connected with Jenkins as node using SSH

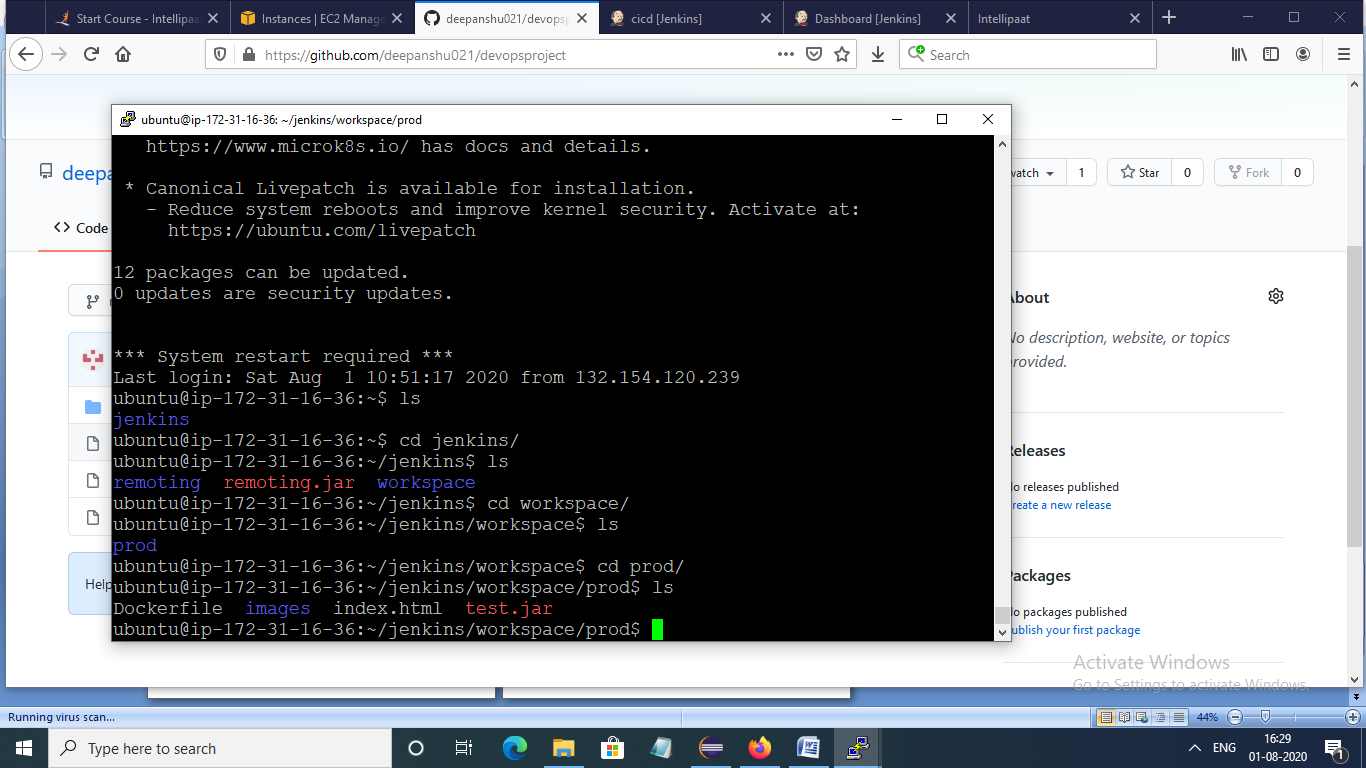


Configuration of production server



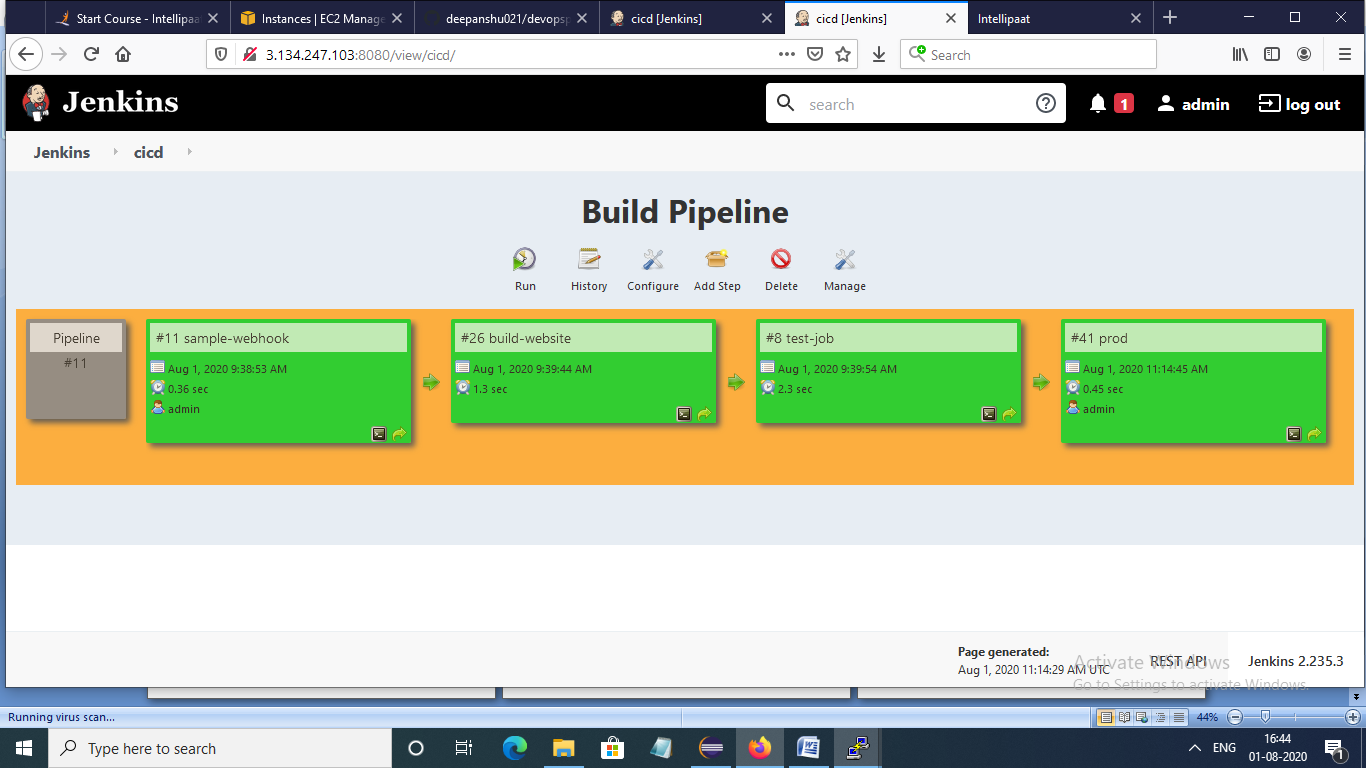


When production server connected with Jenkins all files automatically copied in production server

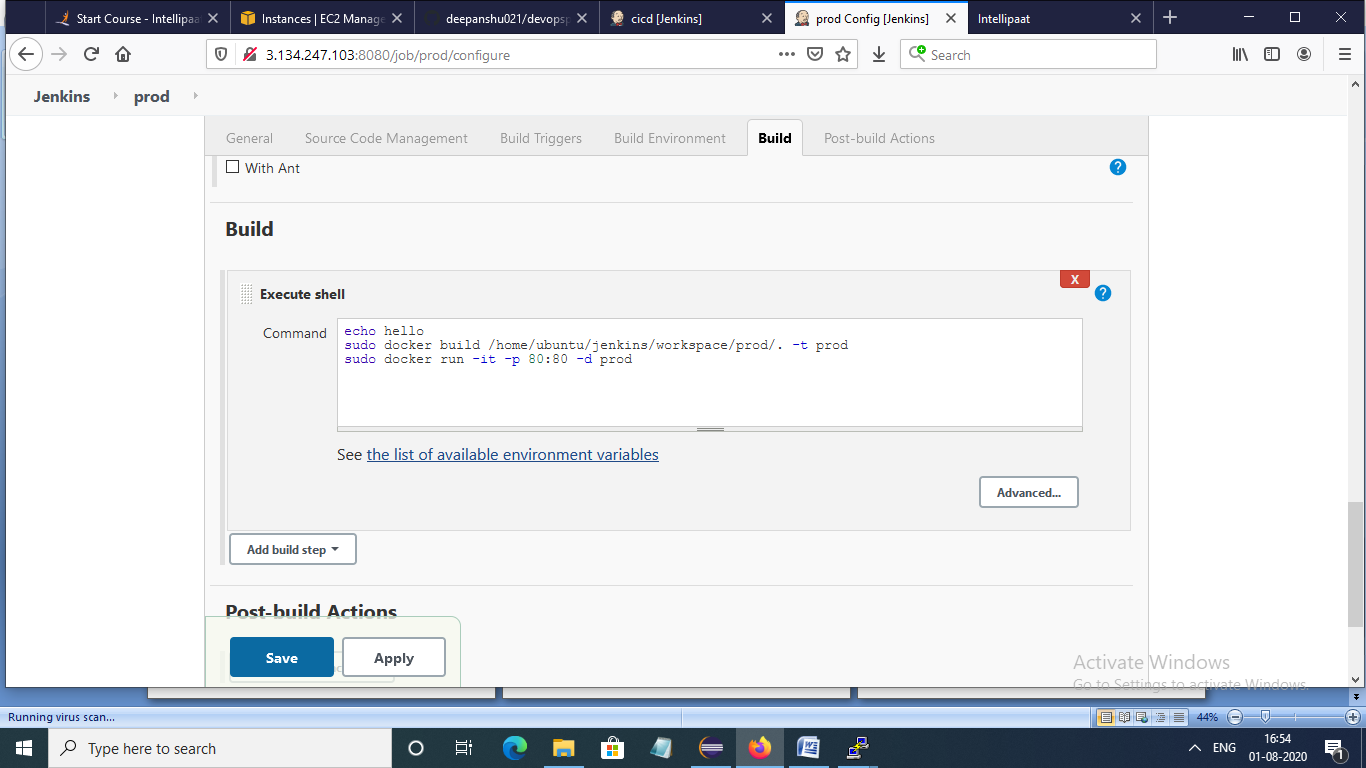




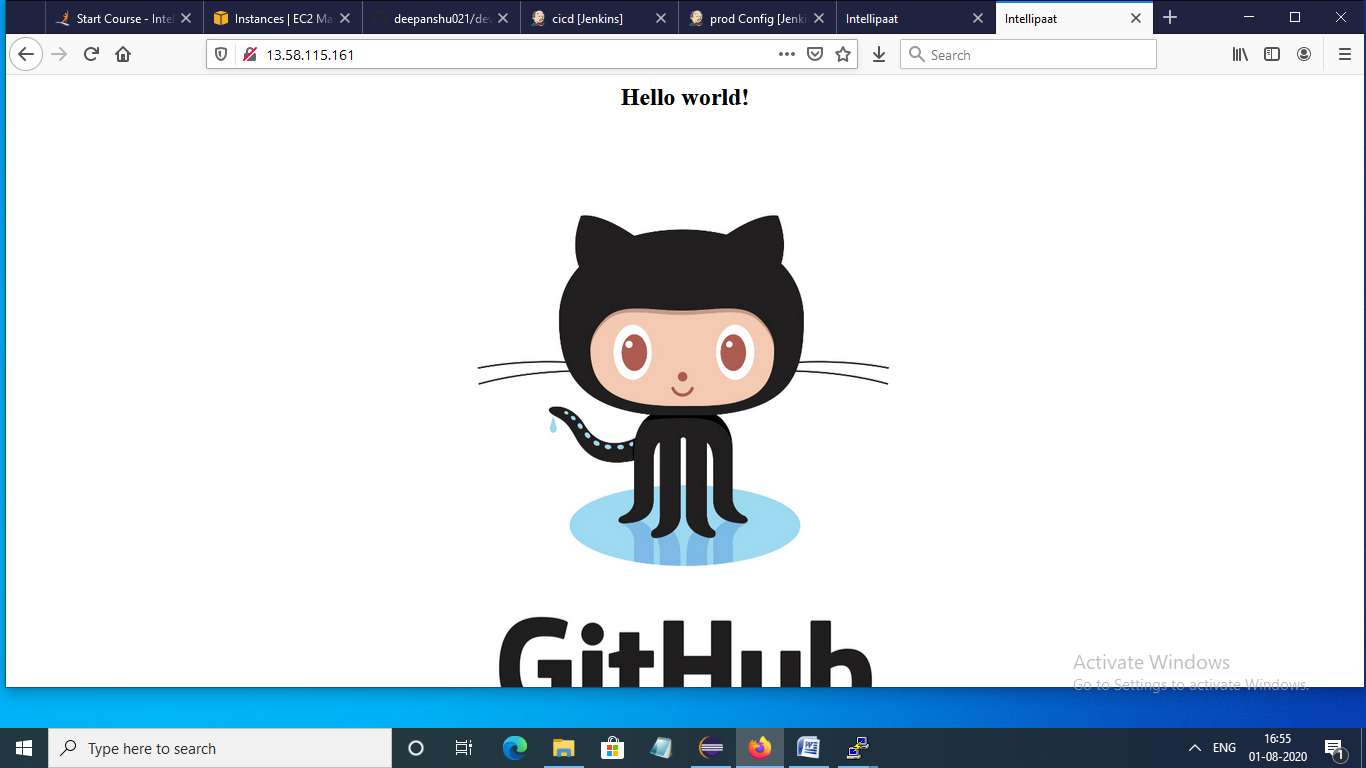
Successful build of CICD pipeline with production server



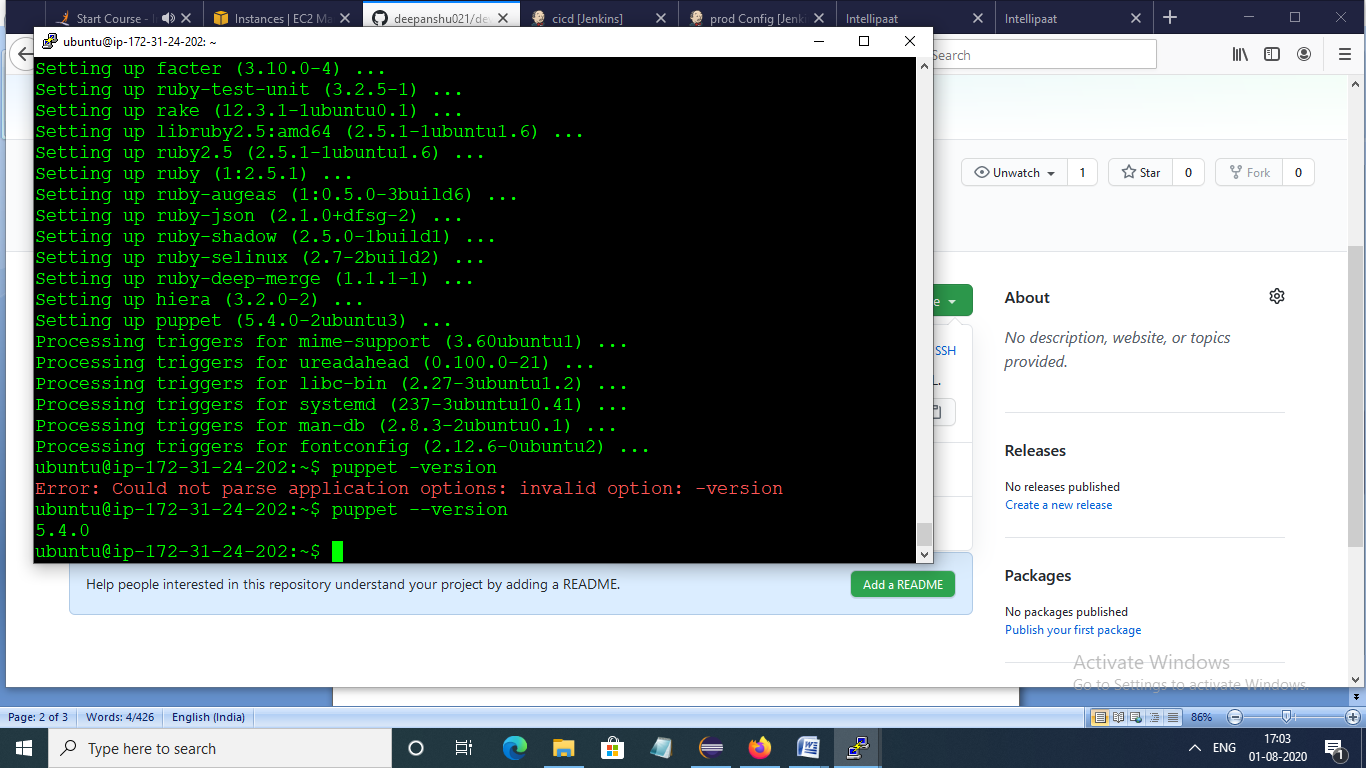
Configuration of production server



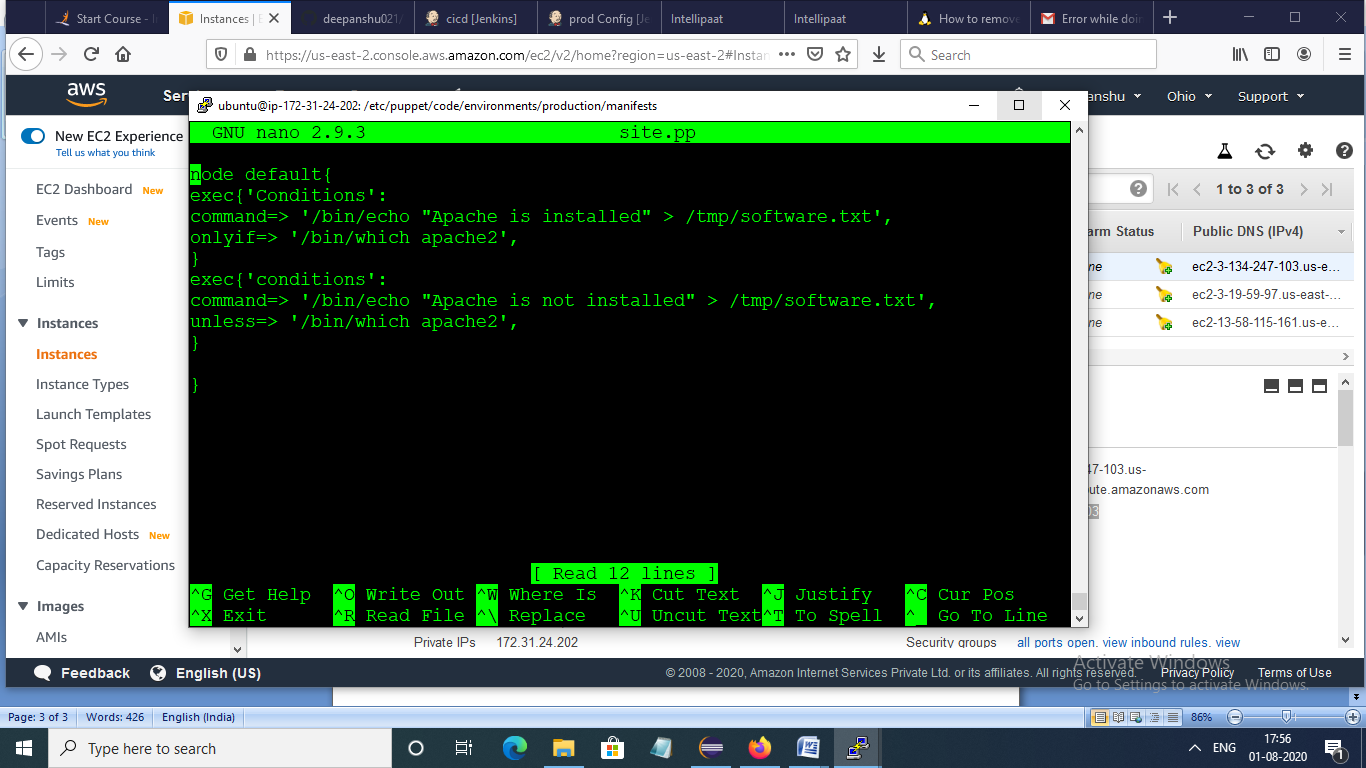
Website runs on port 80



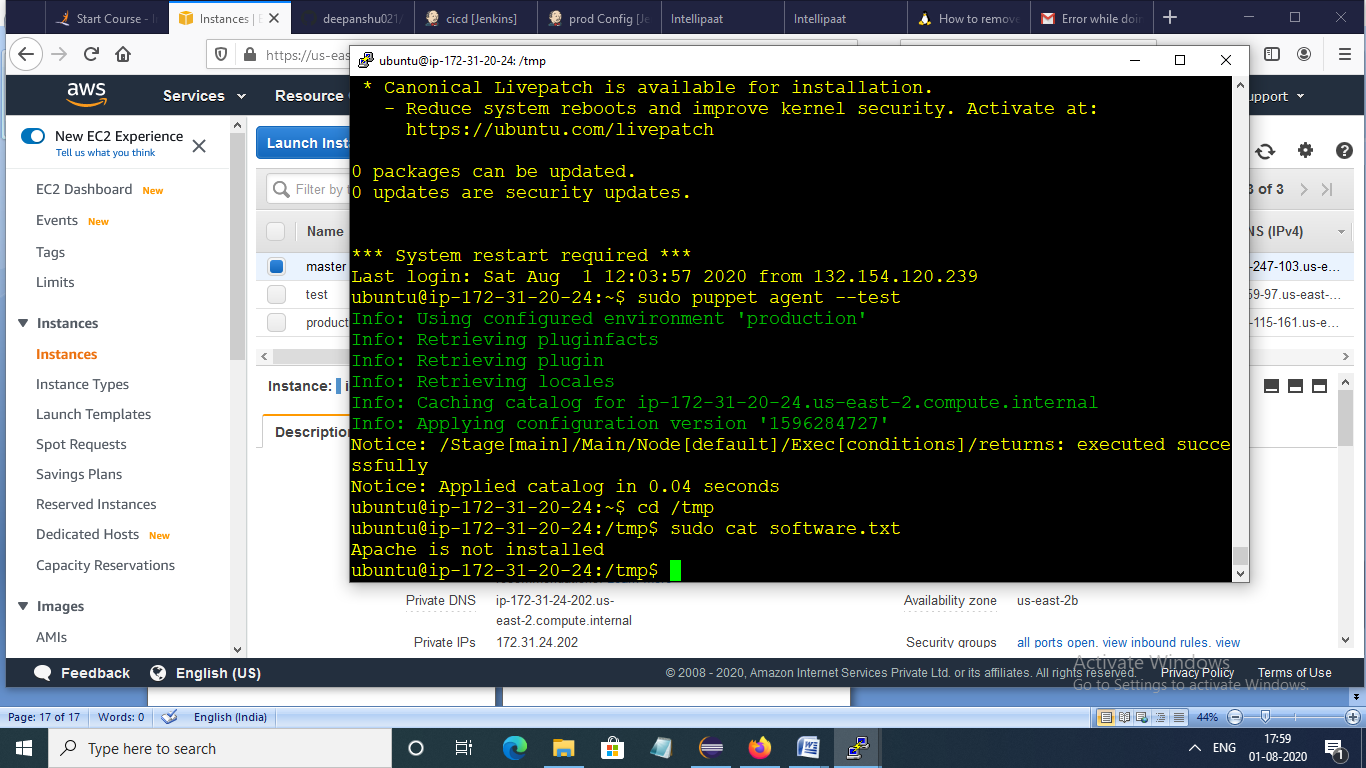
Installation of puppet master on master server



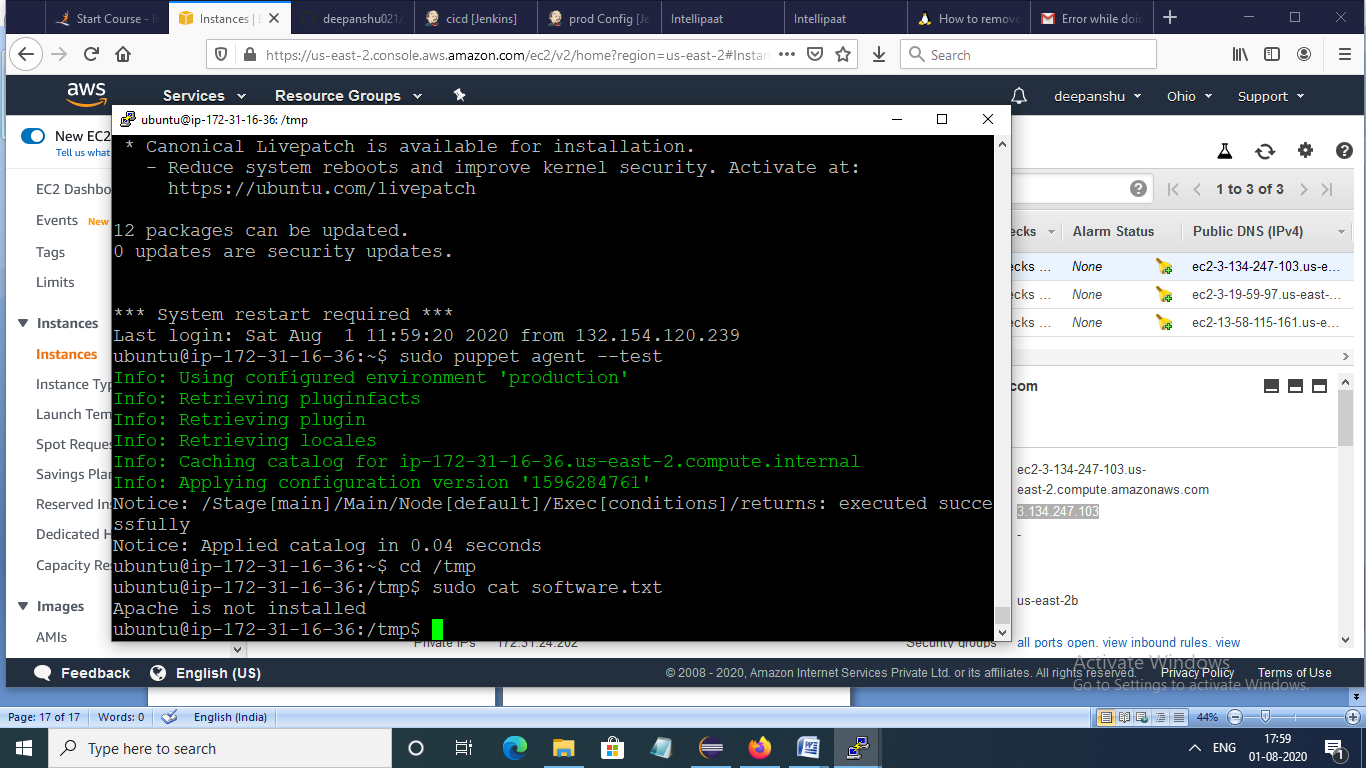
Configuration of master server fir apache2 using puppet



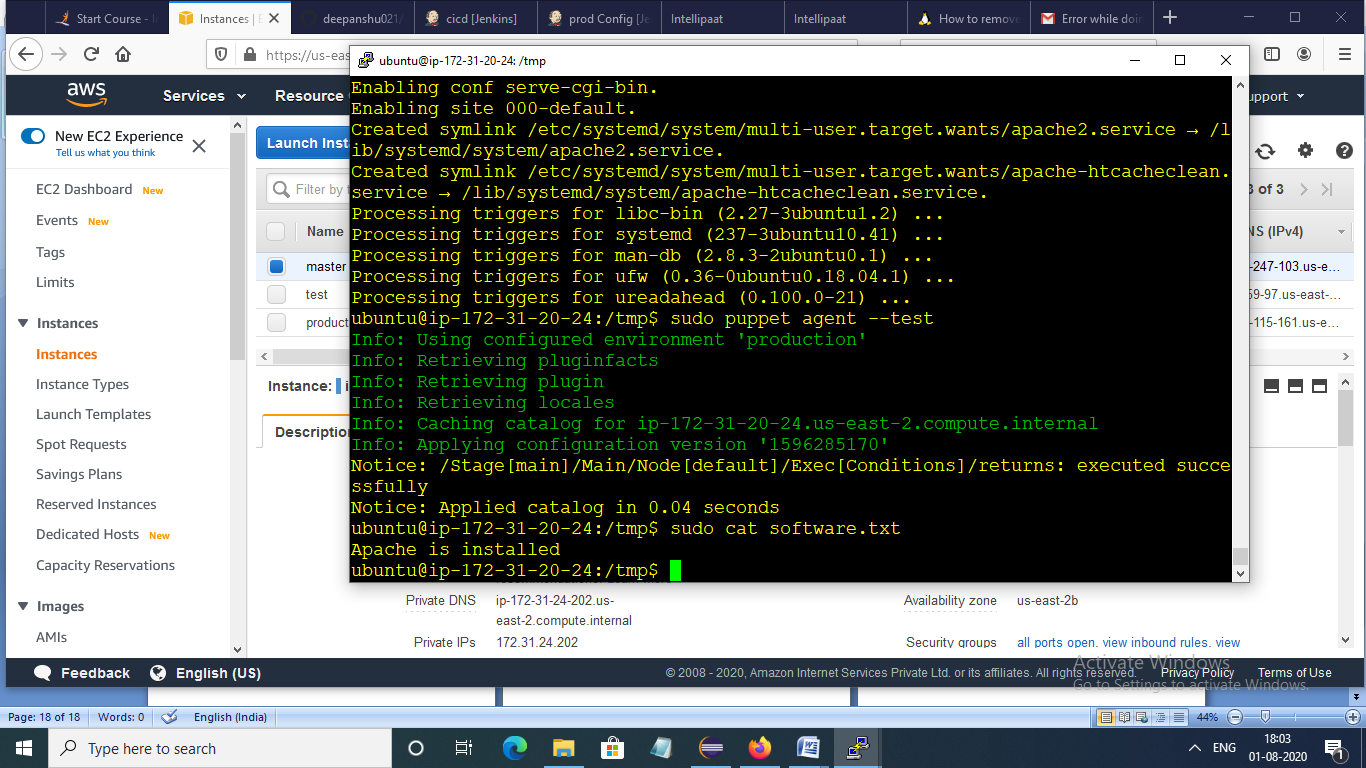
Test server showing puppet is installed or not



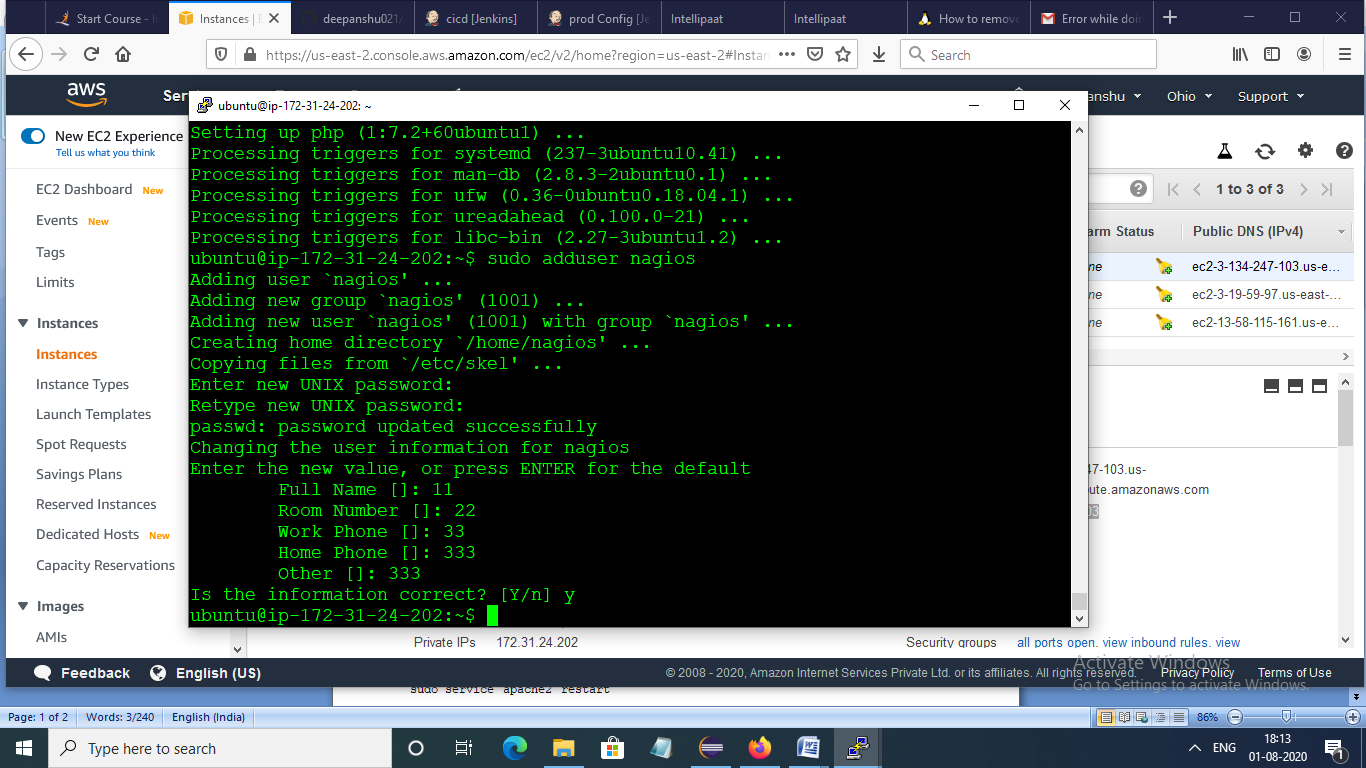
Production server with apache2 installation



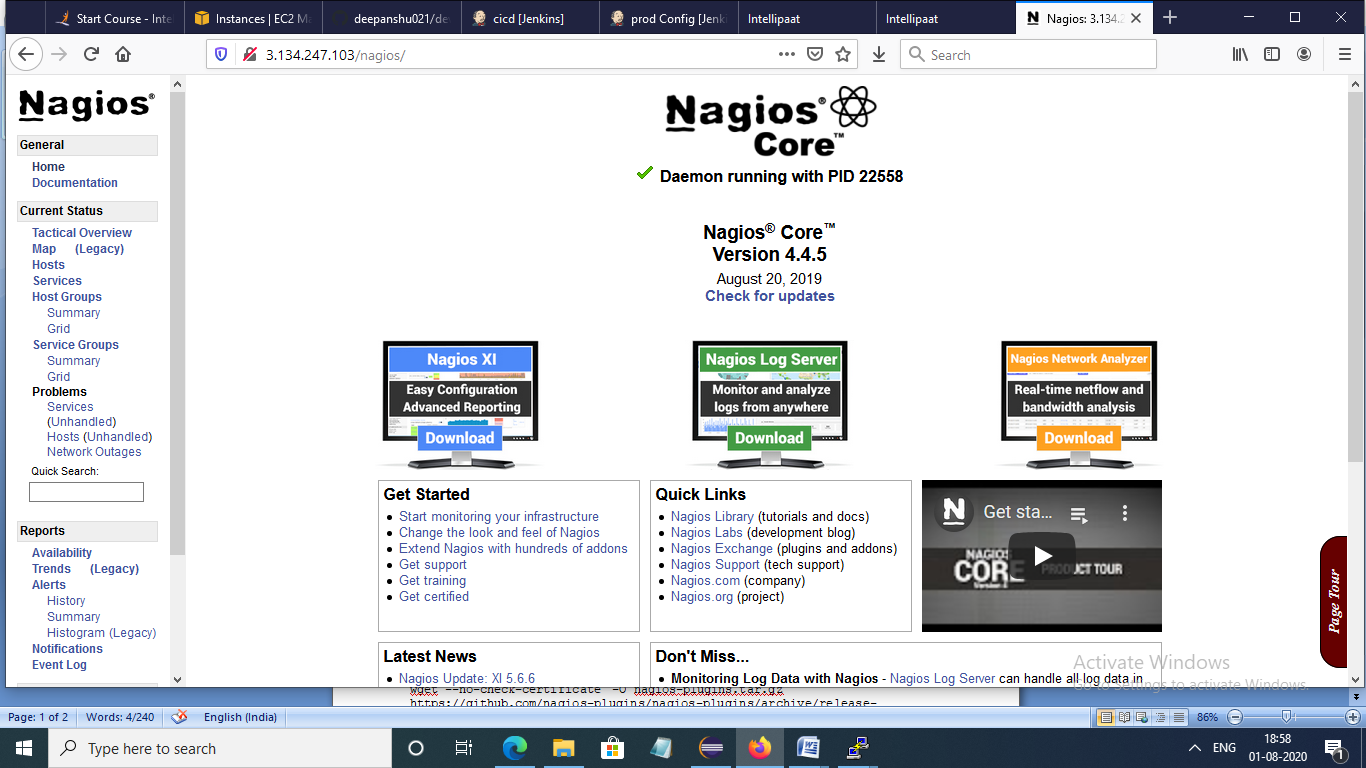
Installation of puppet on testing server

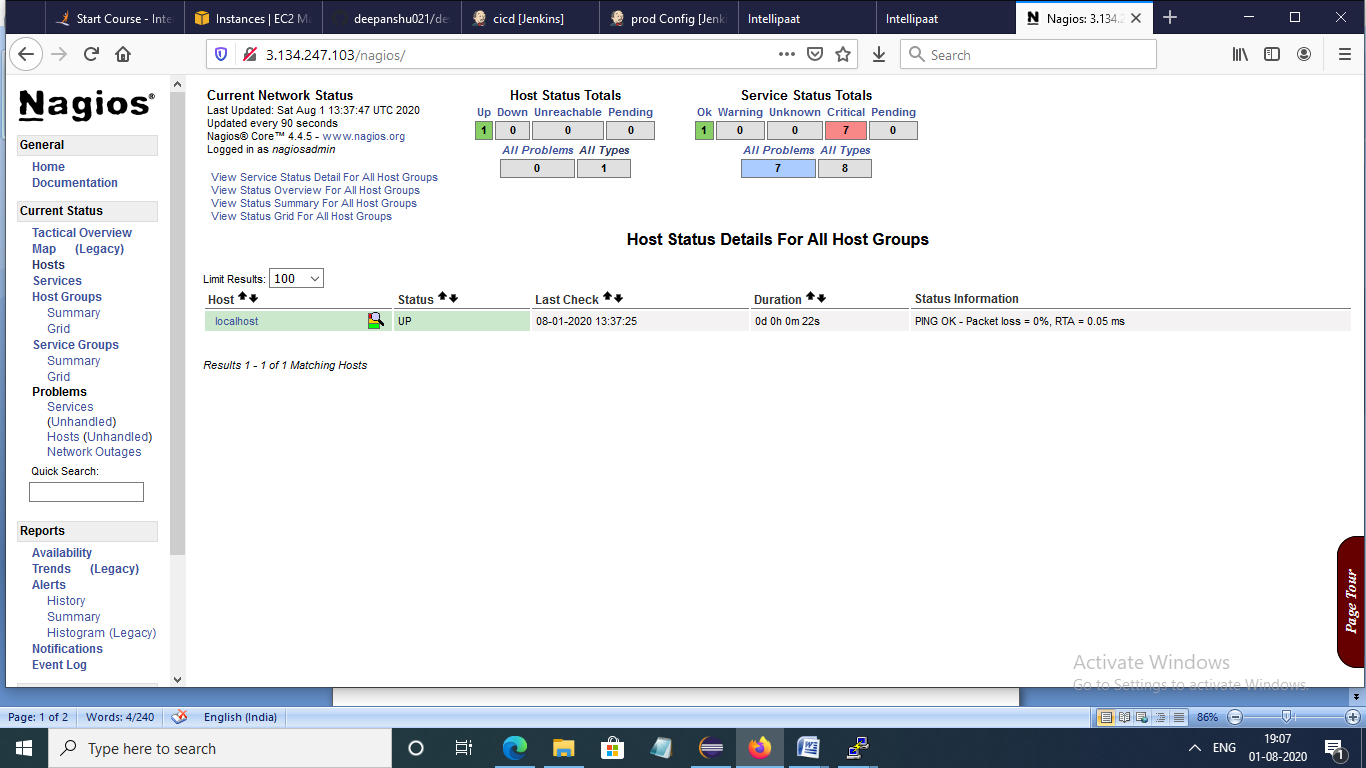


Installation of nagios on master server

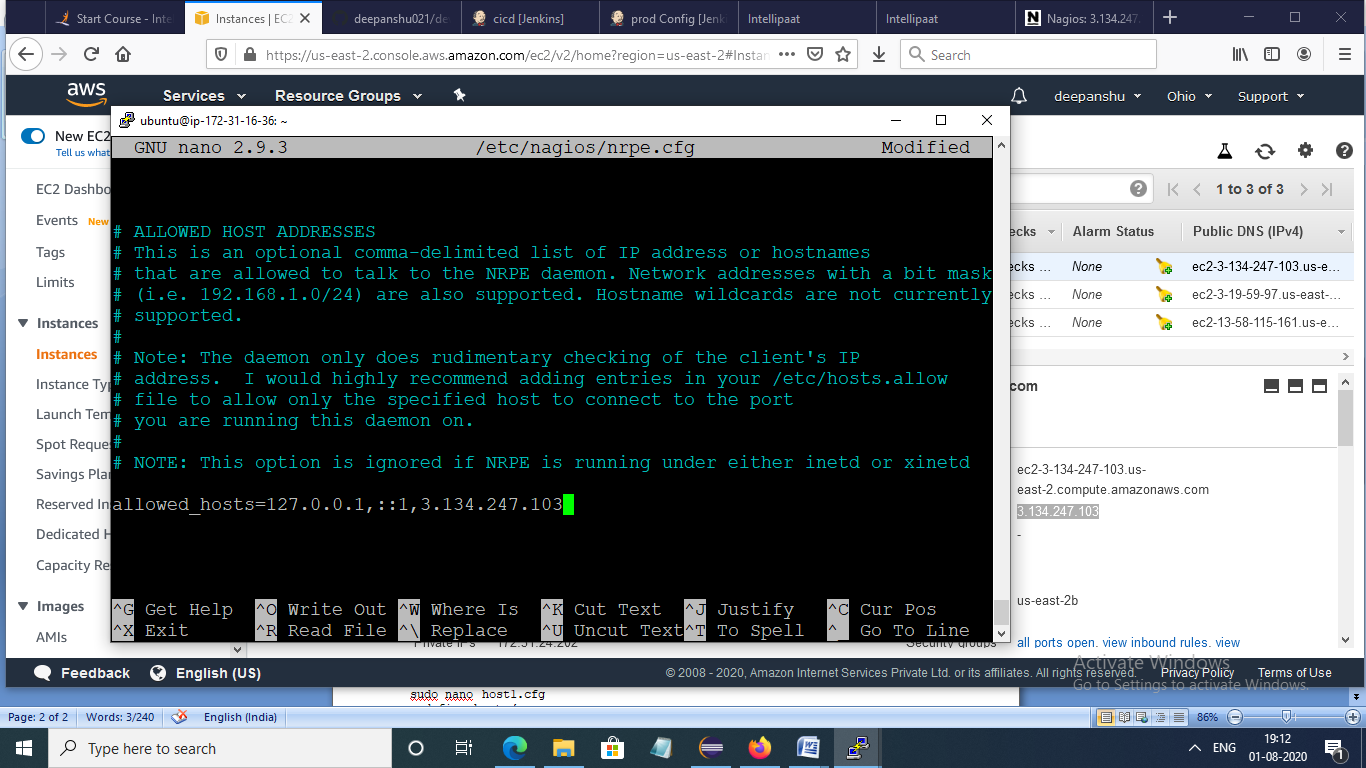


Nagios installed

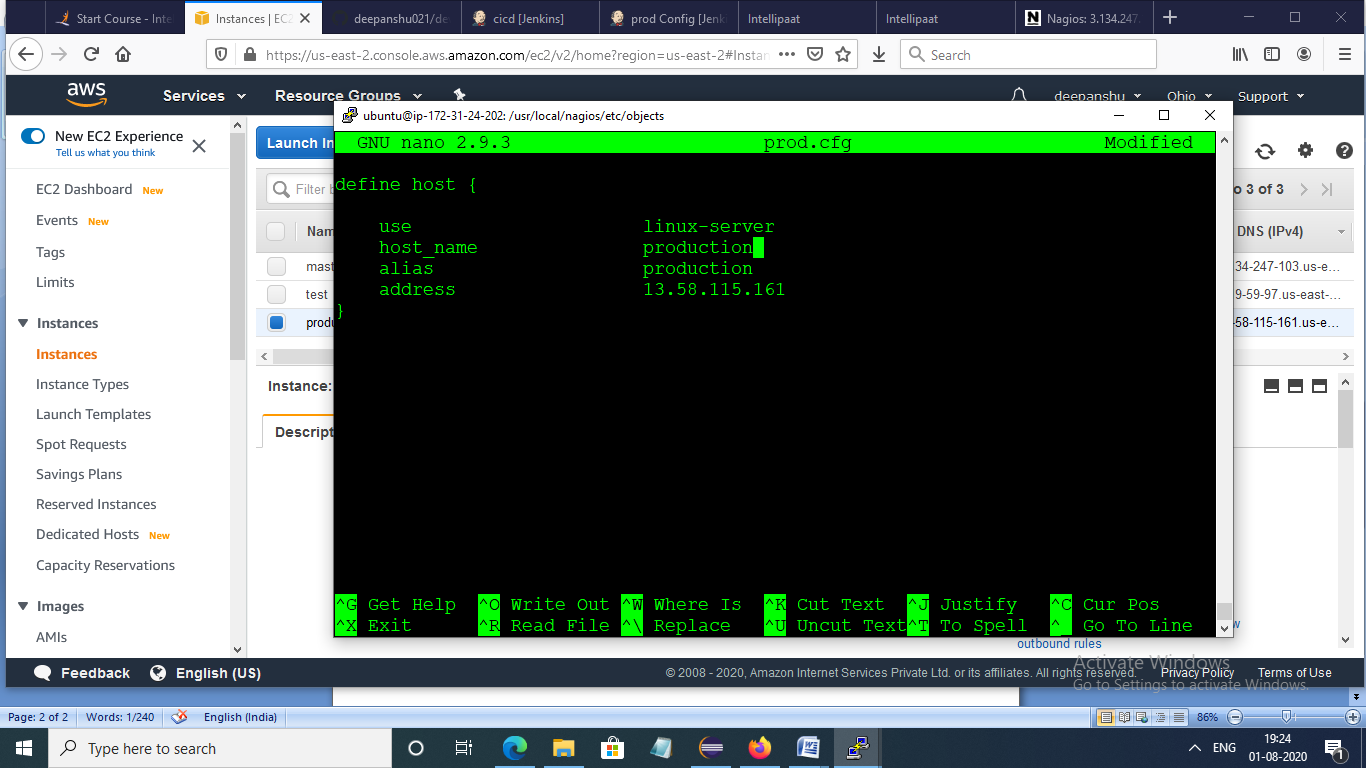


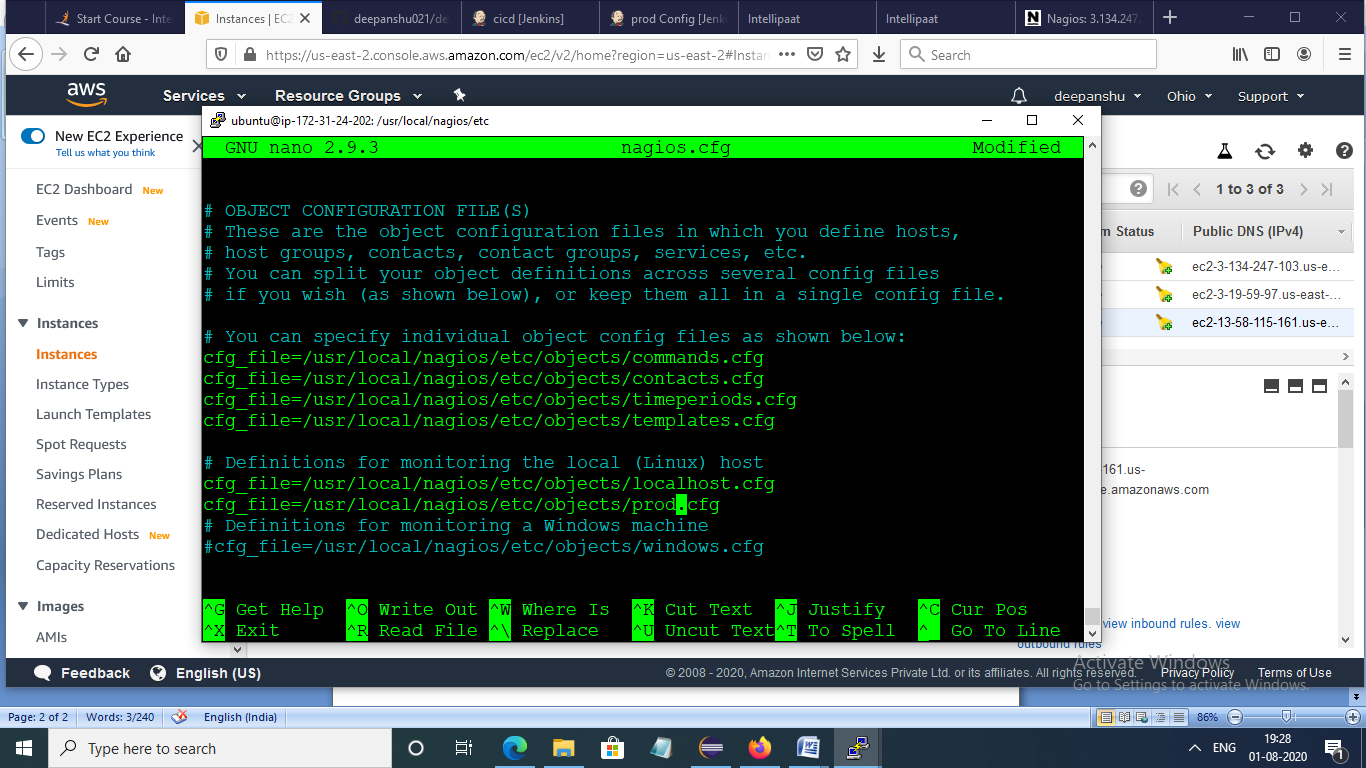


Configuration of nagios

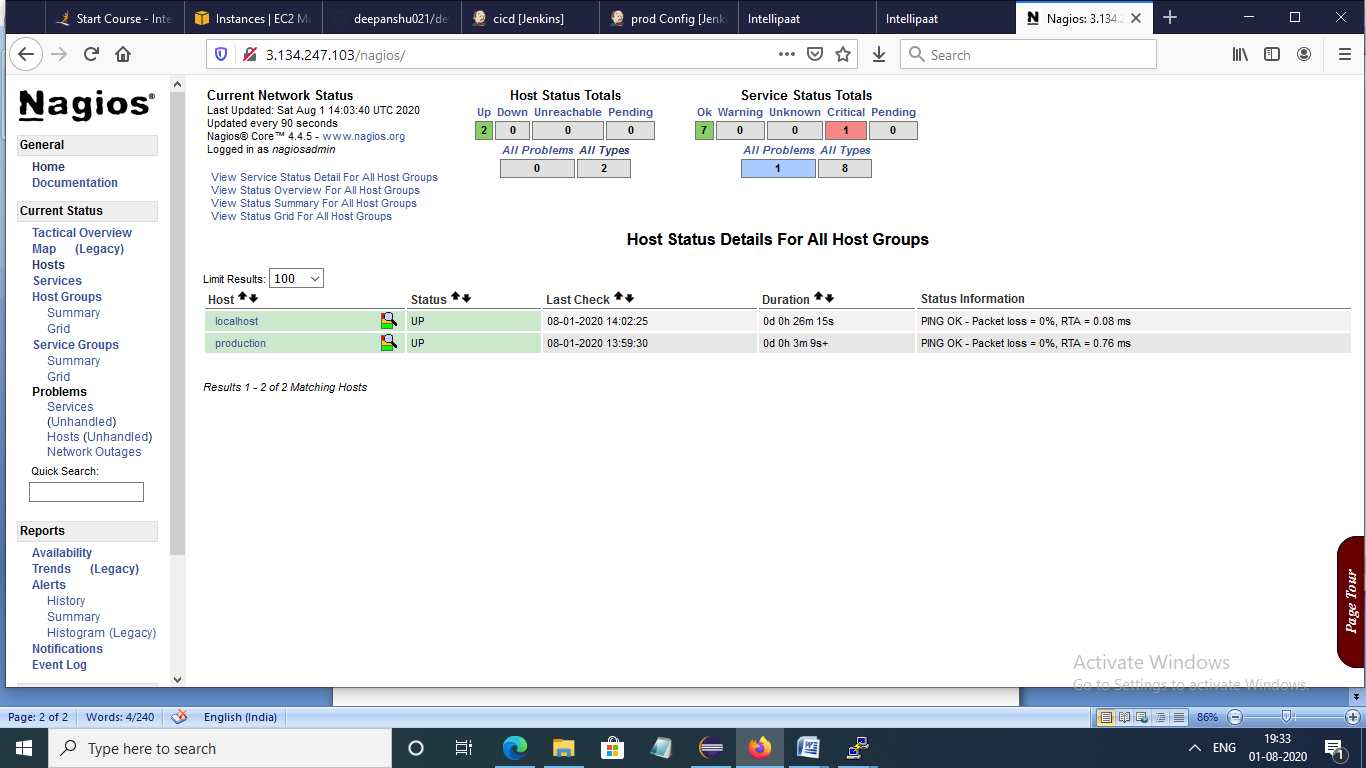


Connection of nagios with production server

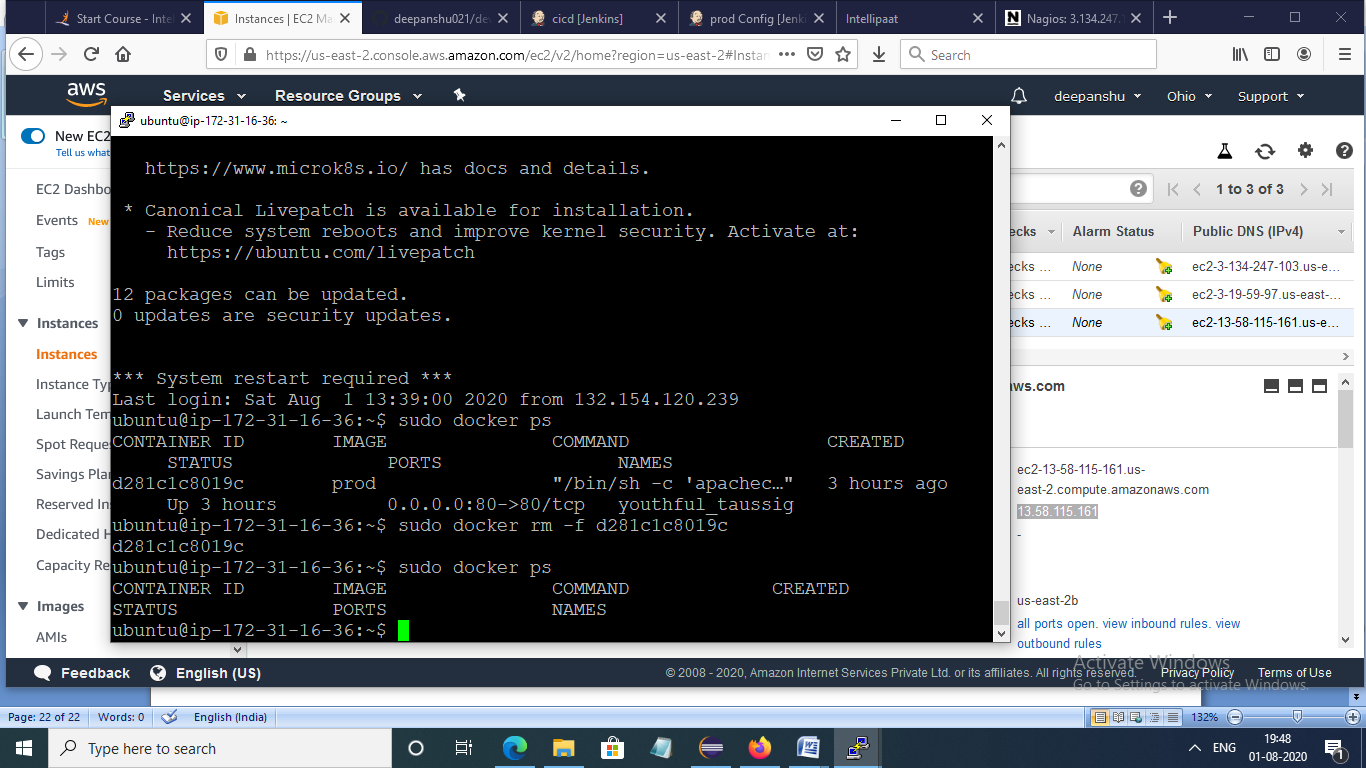




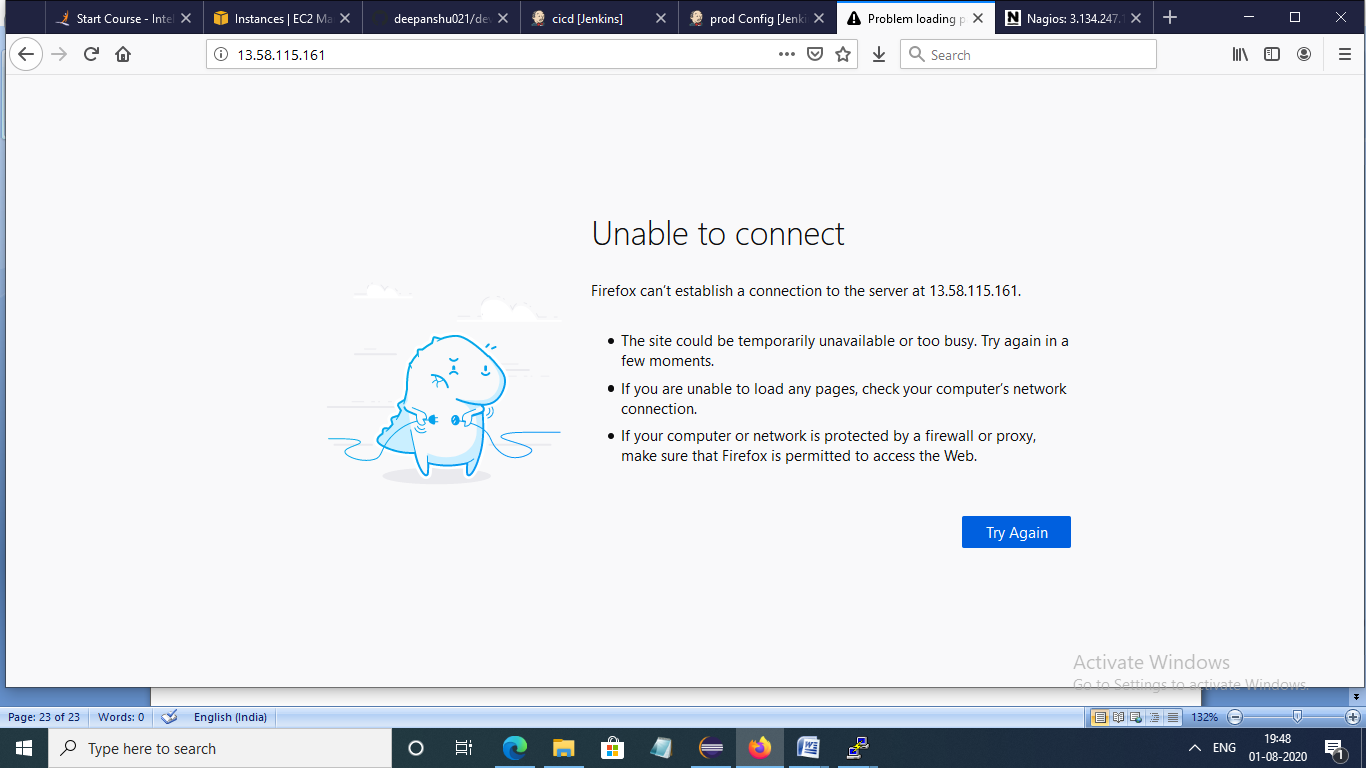
Production server up and running



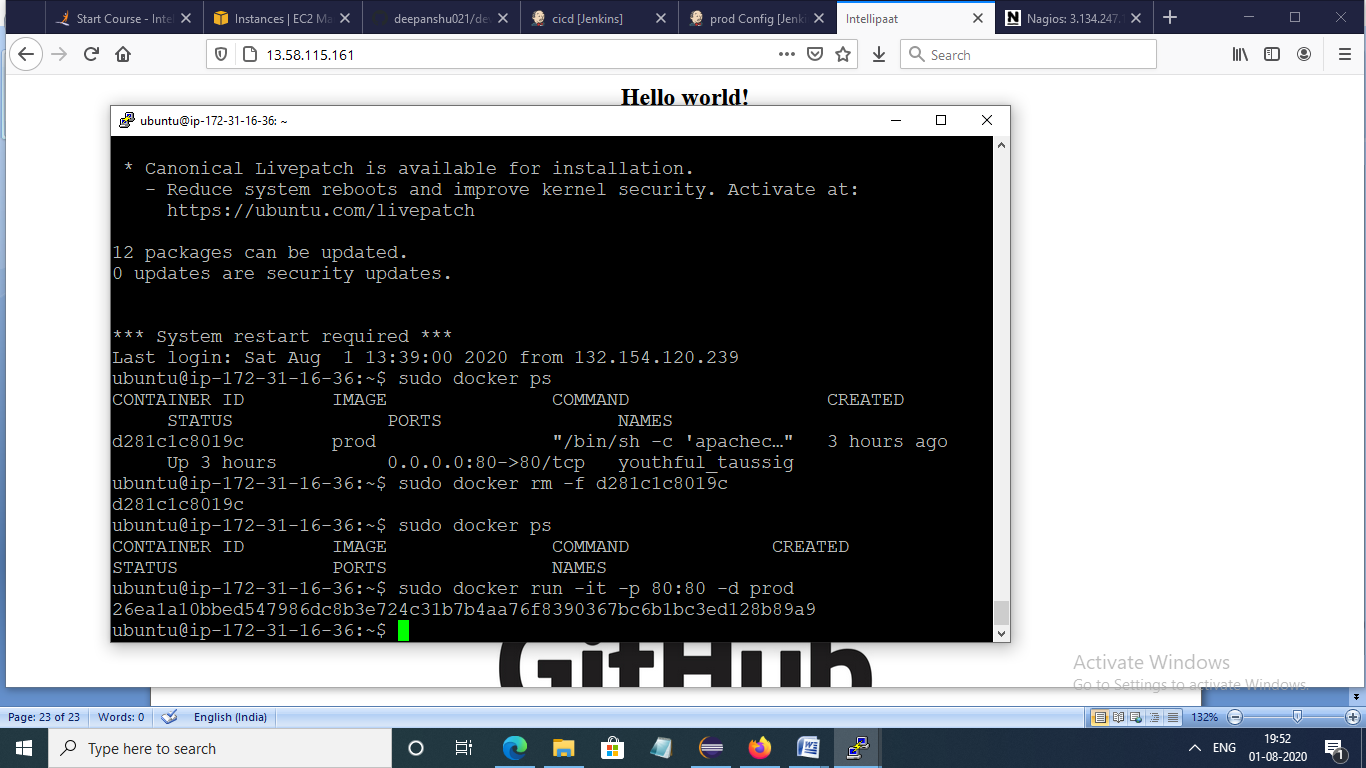
Removal of docker container port 80



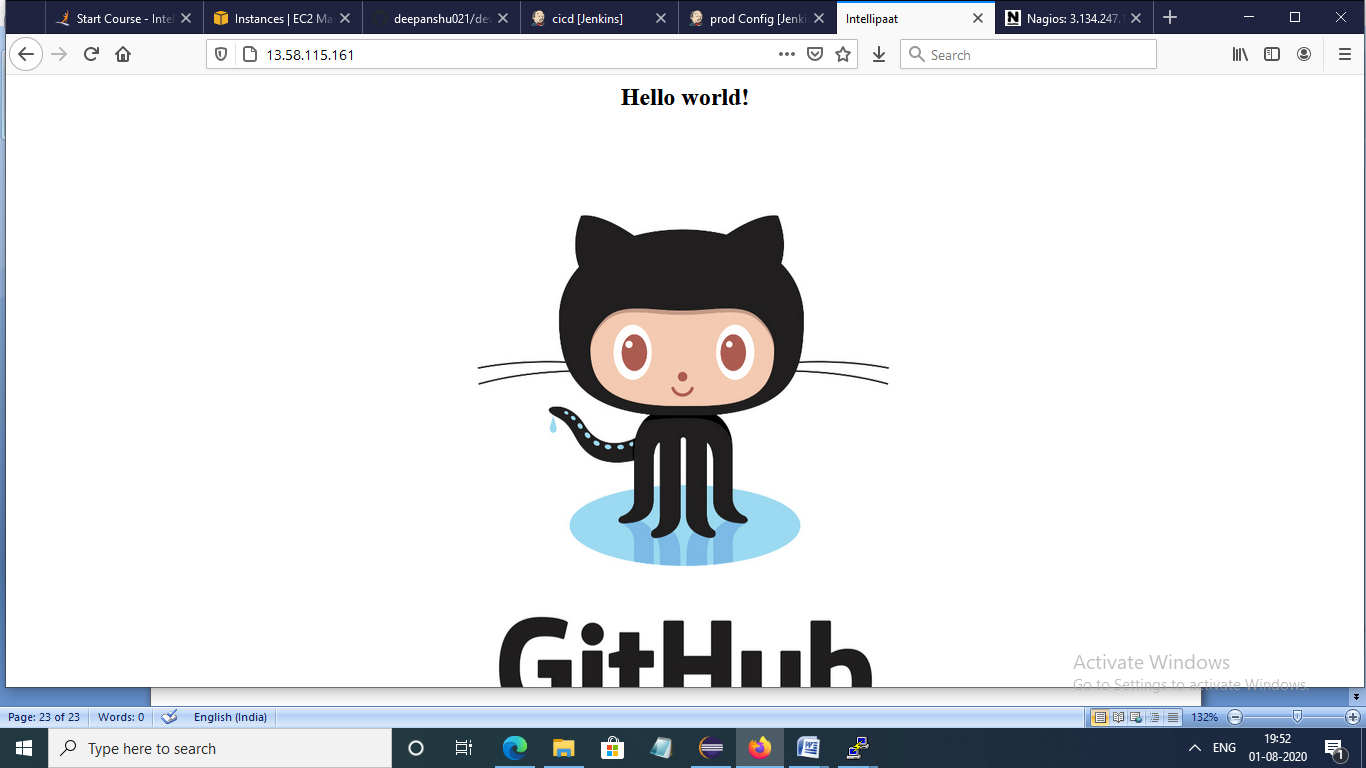
Website went down



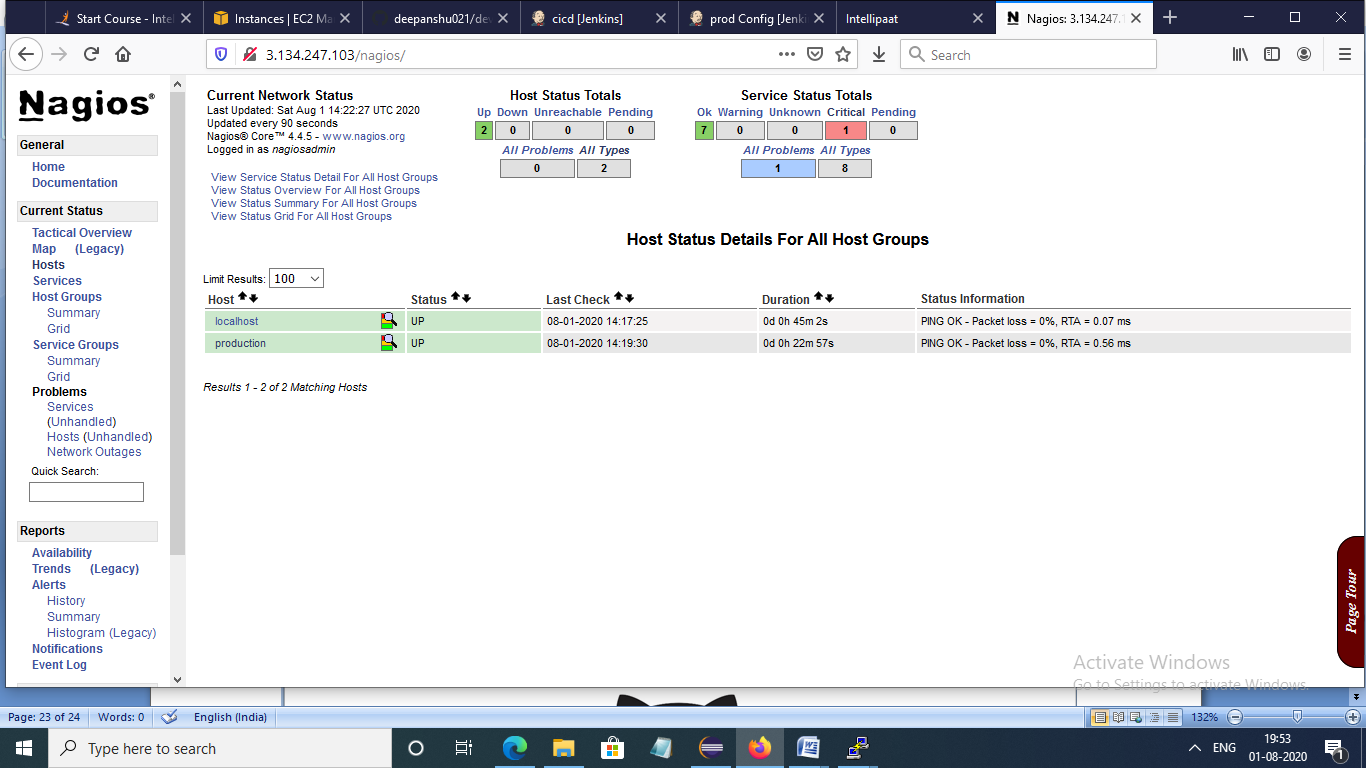
Creation of docker container on port 80



Website again pop up



And monitoring server again up and running



**THANK YOU**