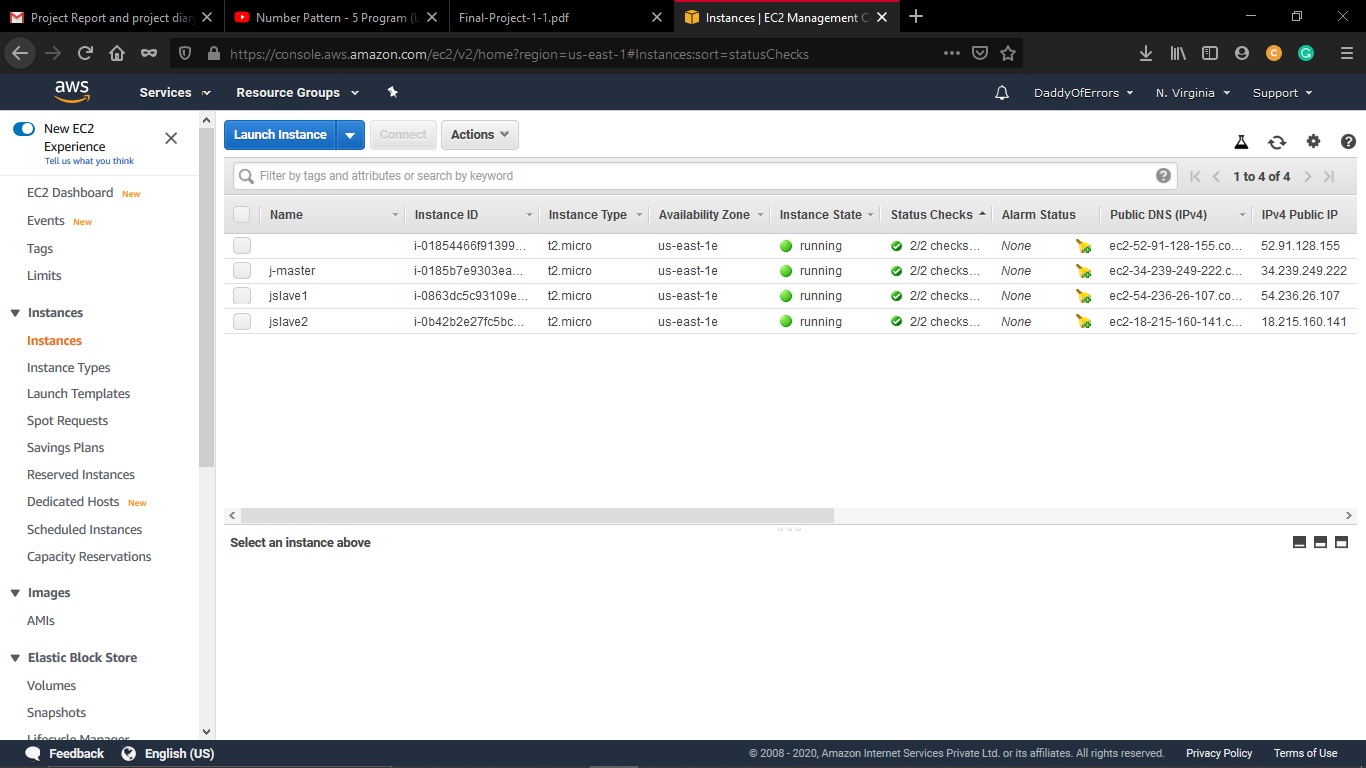
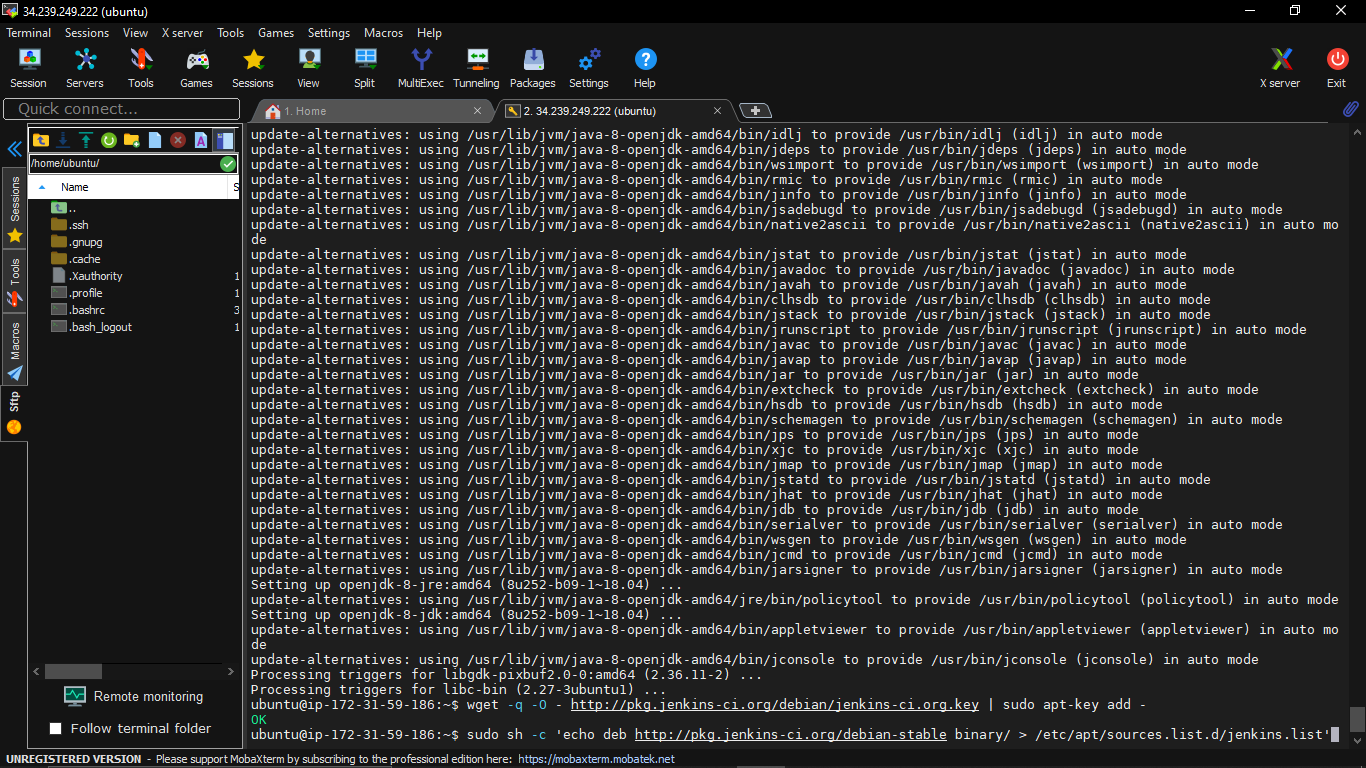
CAPSTONE PROJECT

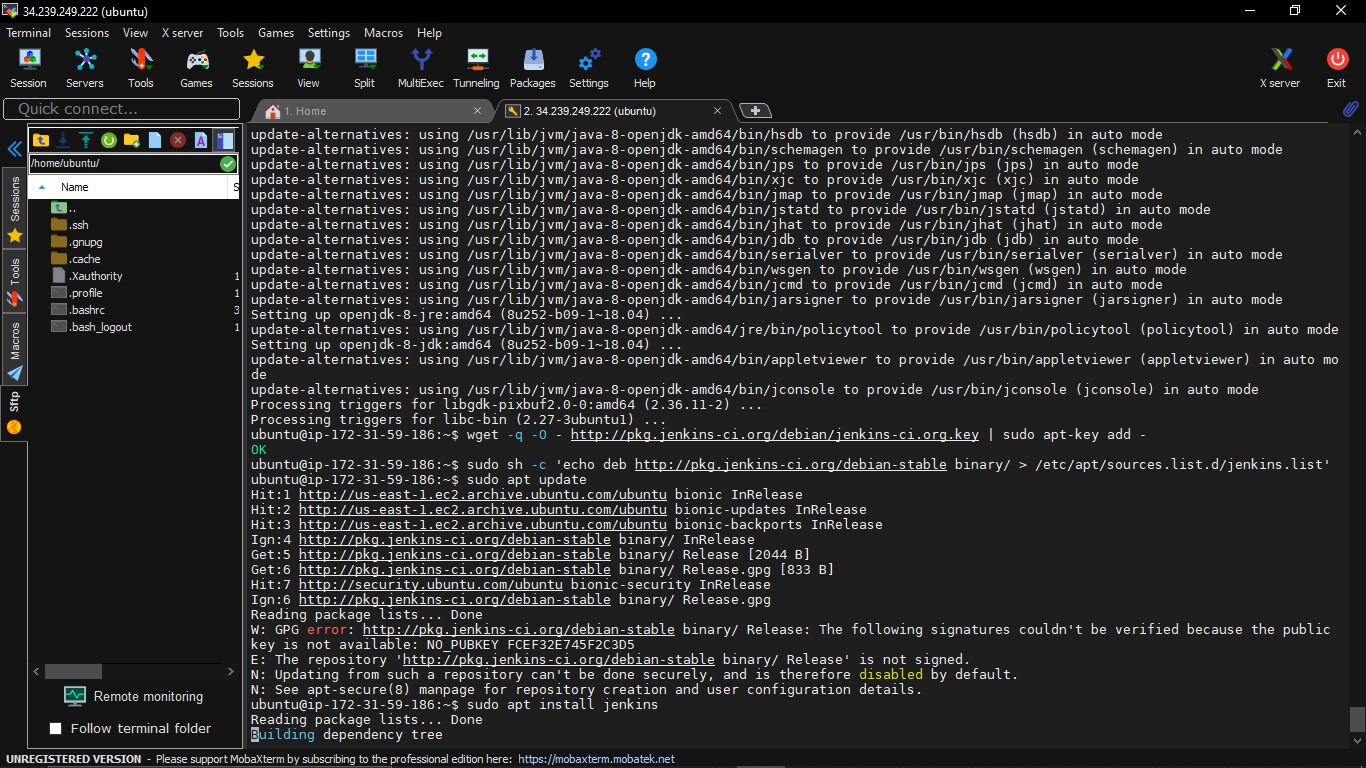
Made 3 instance 1 for jenkins master and other for slaves



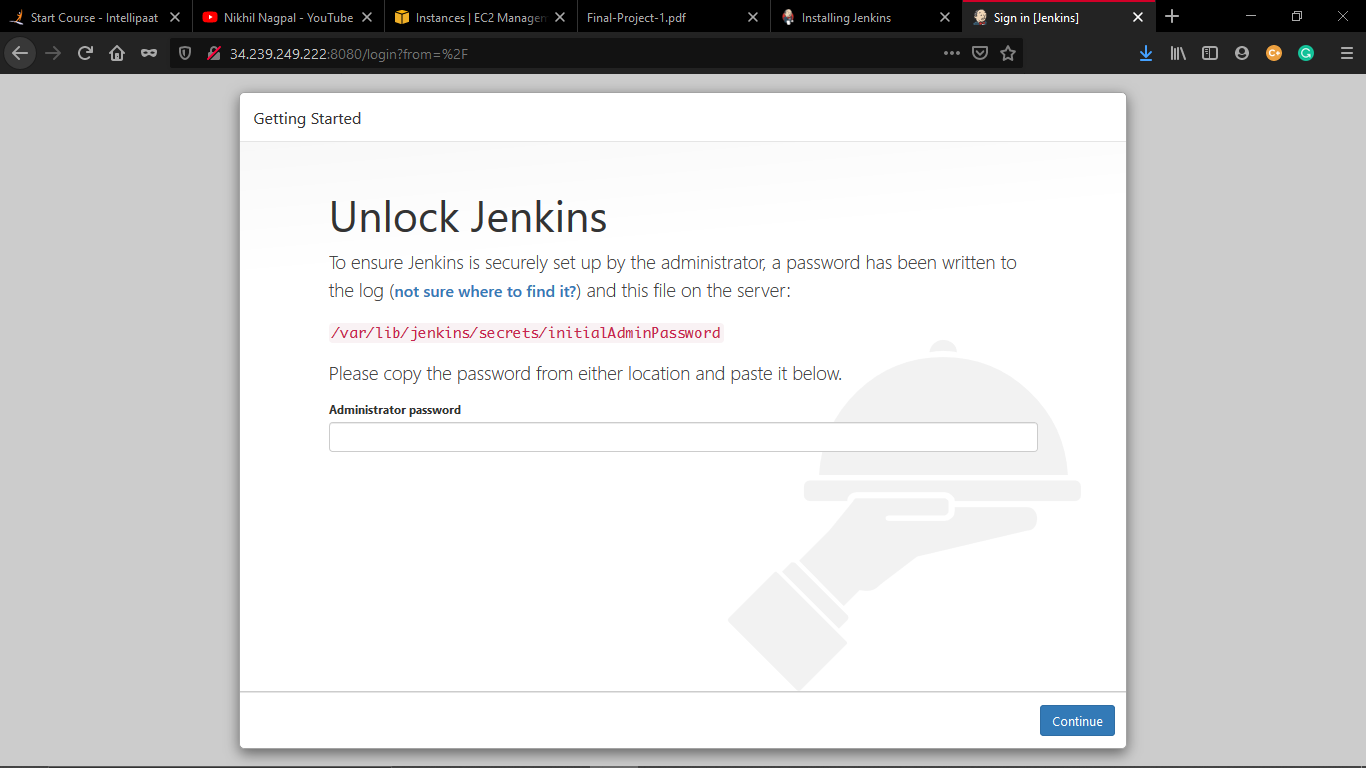
Setting op jenkins on master



Adding jenkins keys

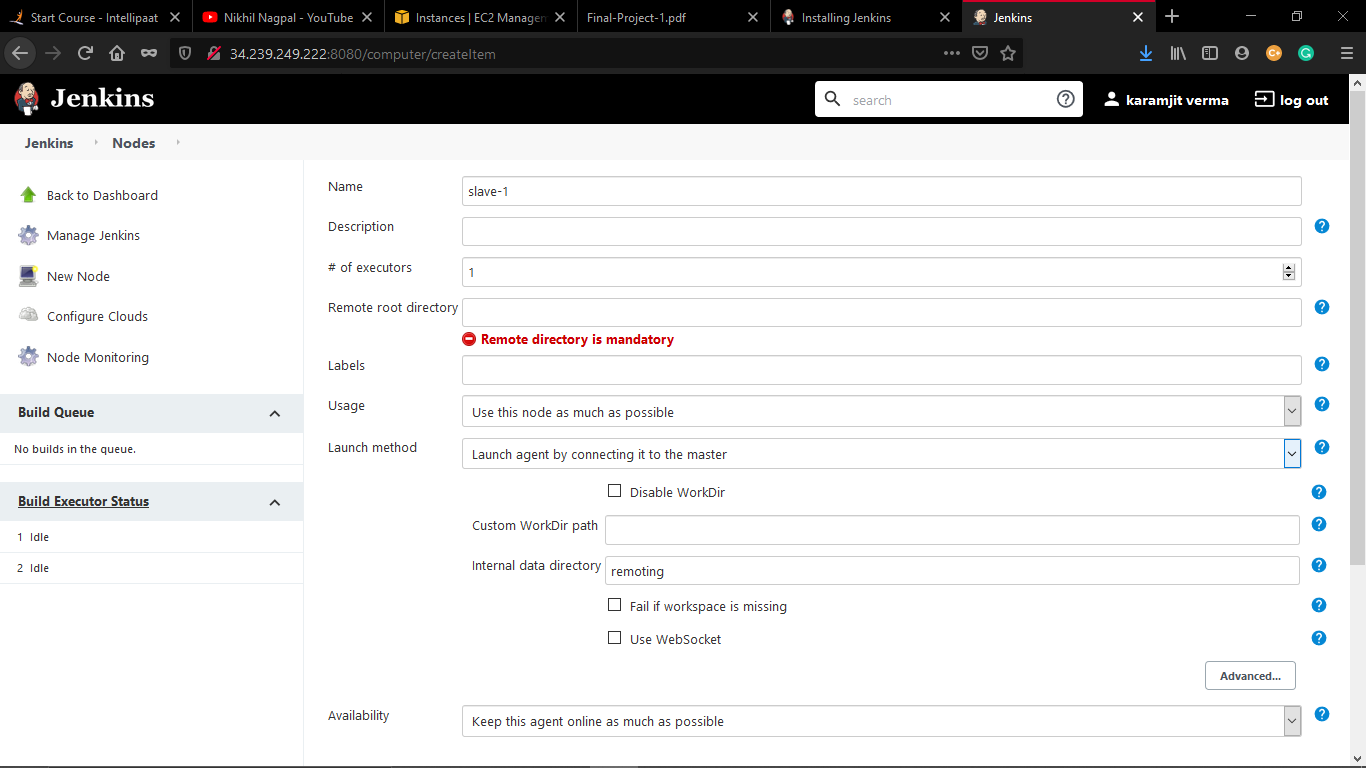


Logging into my jenkins dashboard

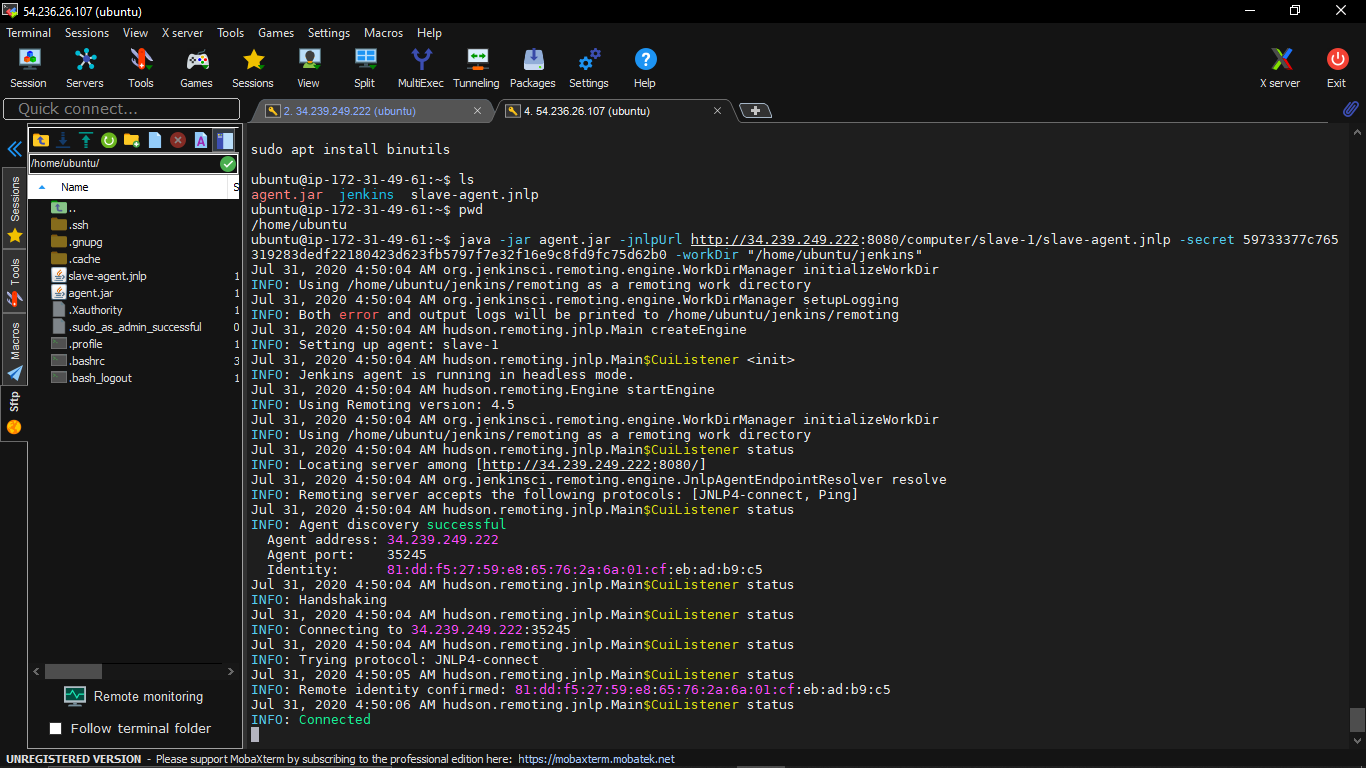


Now it’s the time for setting op slaves:

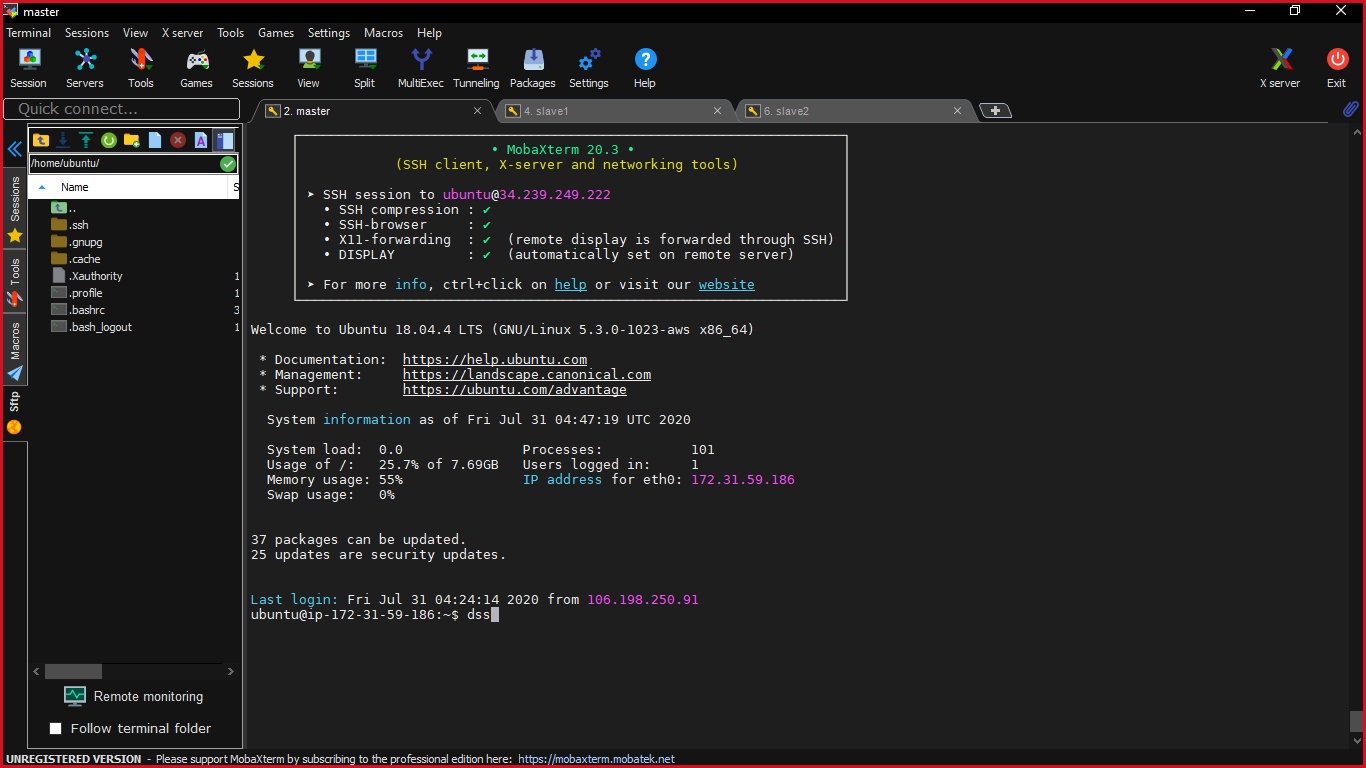
Setting slave 1:



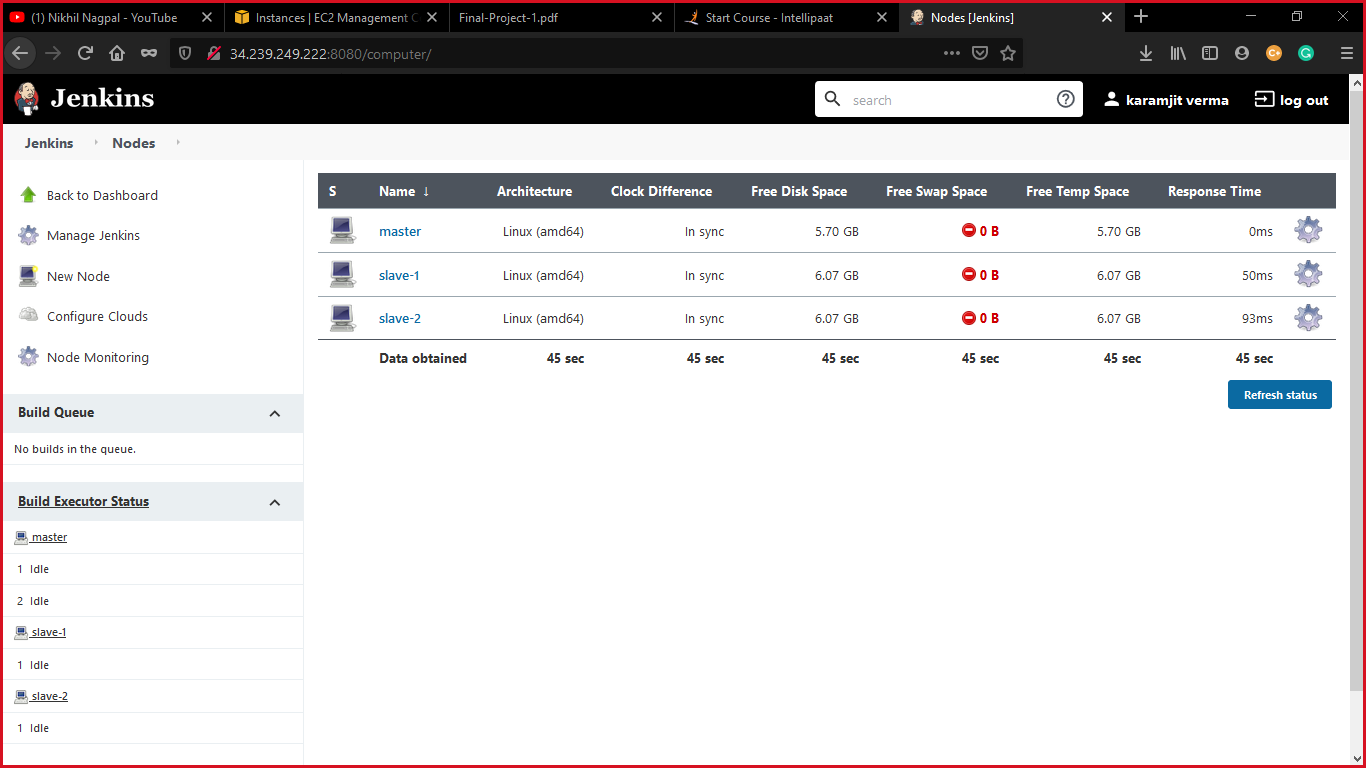
Installing agent jar on slave machine



Used Mobaxtreme software instead of putty because I can manage my terminals easily:



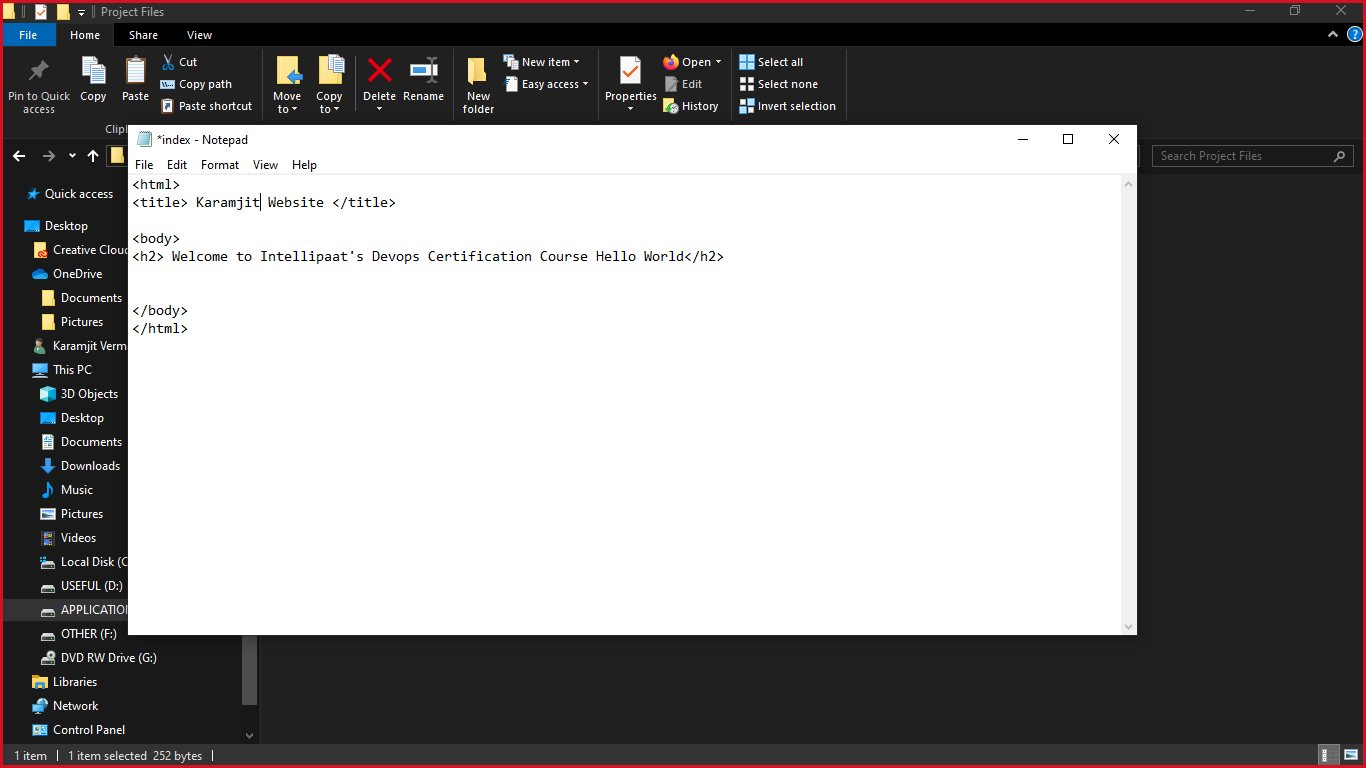
Finally done with the master and slave architecture:



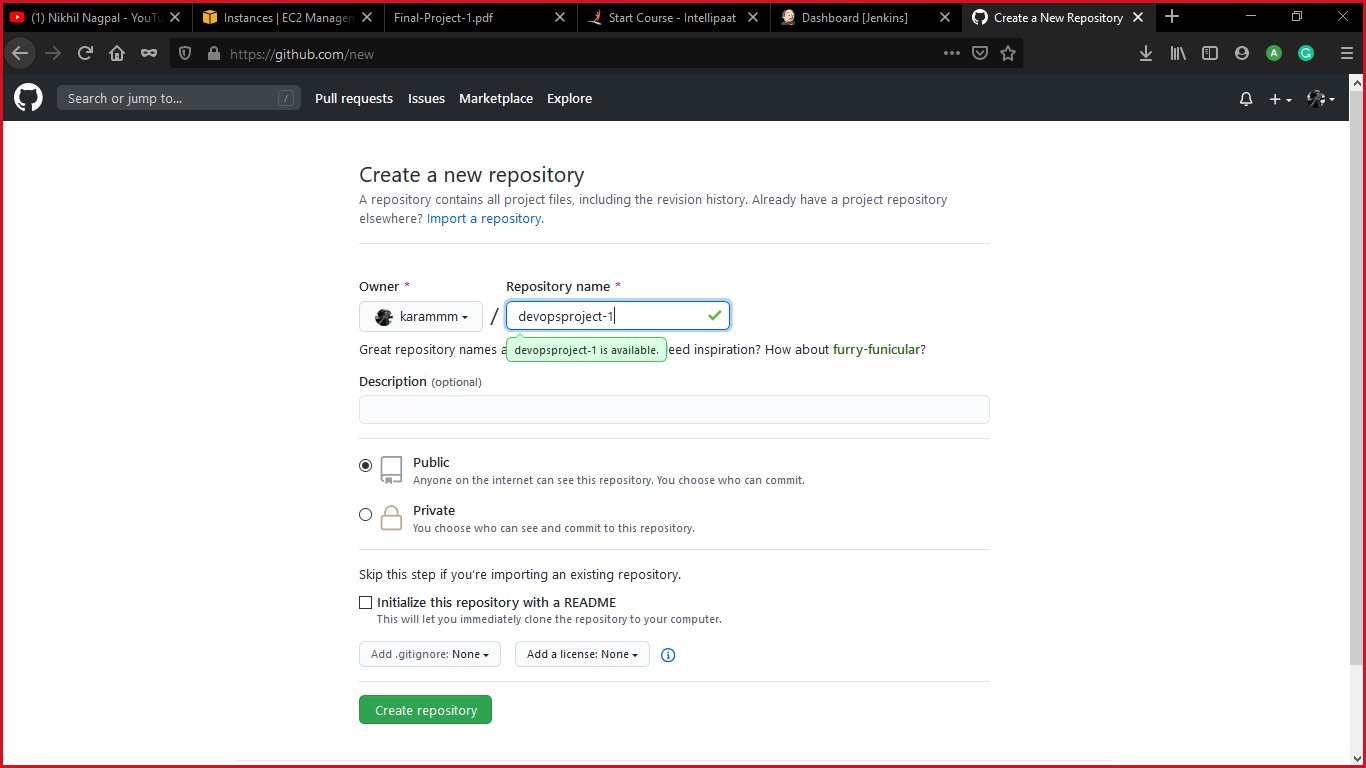
Slave1 is testing server.

Slave2 is production server.

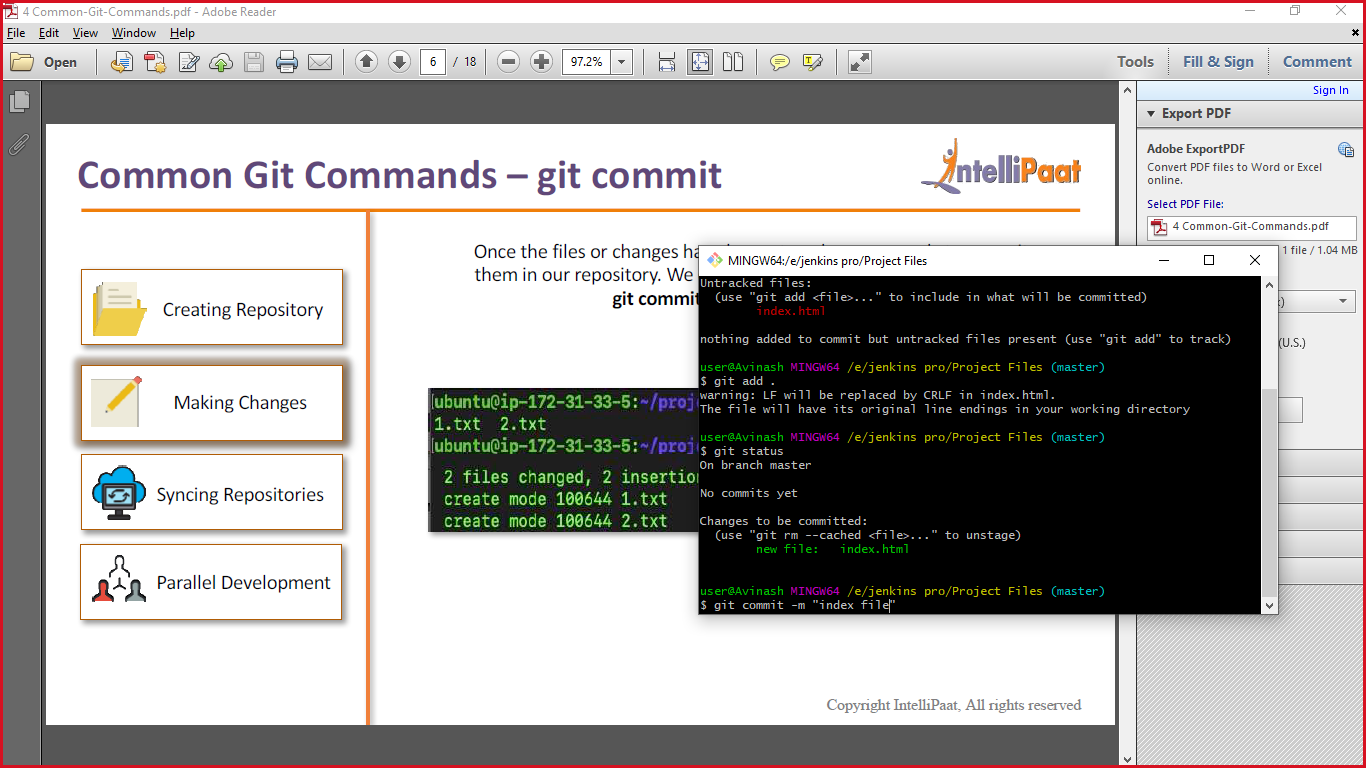
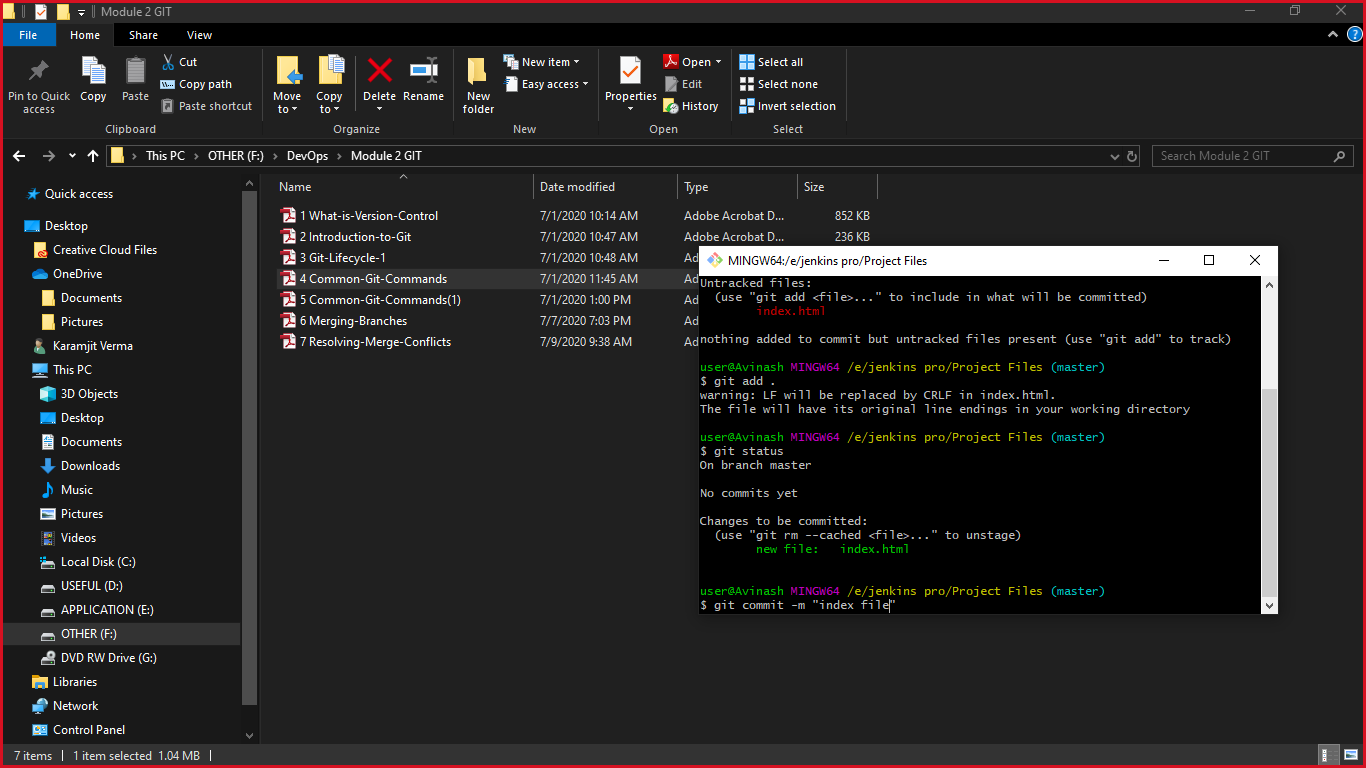
Creating a project website with title karam website as shown below:

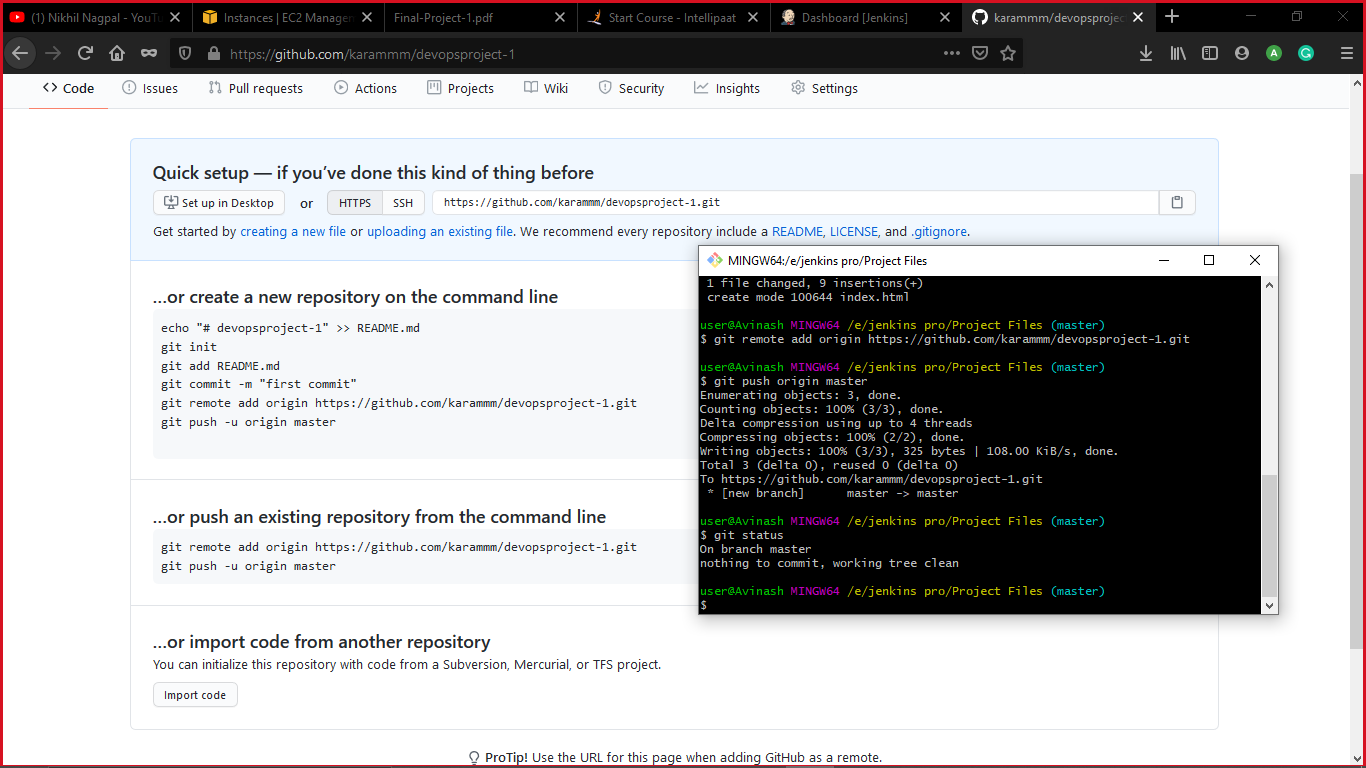


Creating git repository for version control

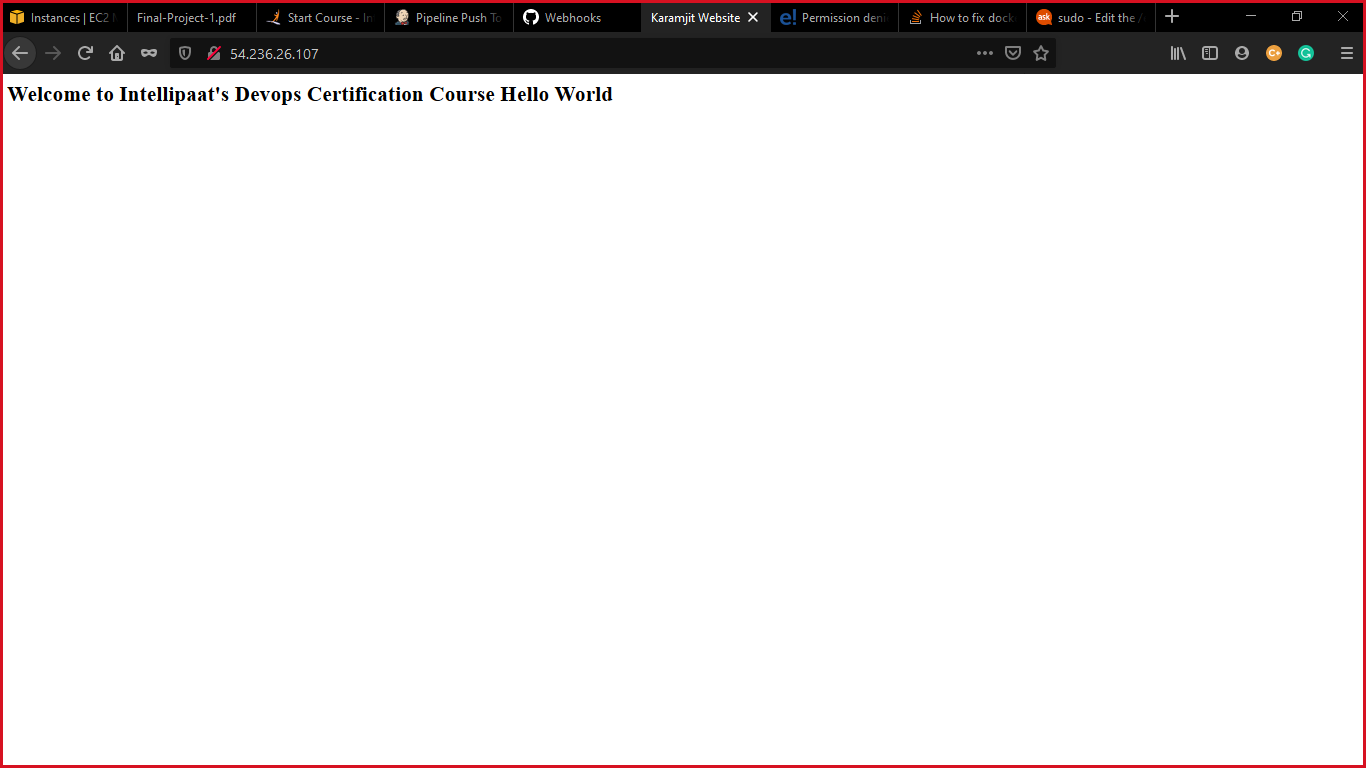


Initially pushed my files from the local system to the git hub



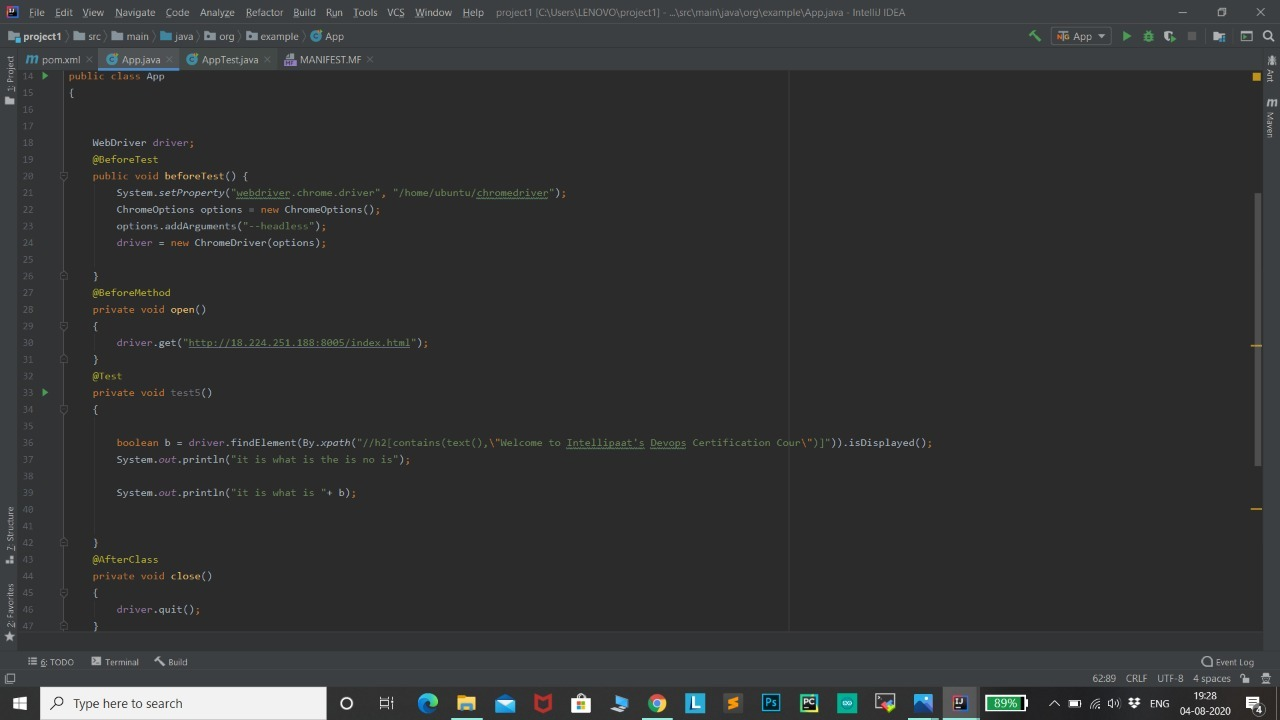
This is the docker file: 



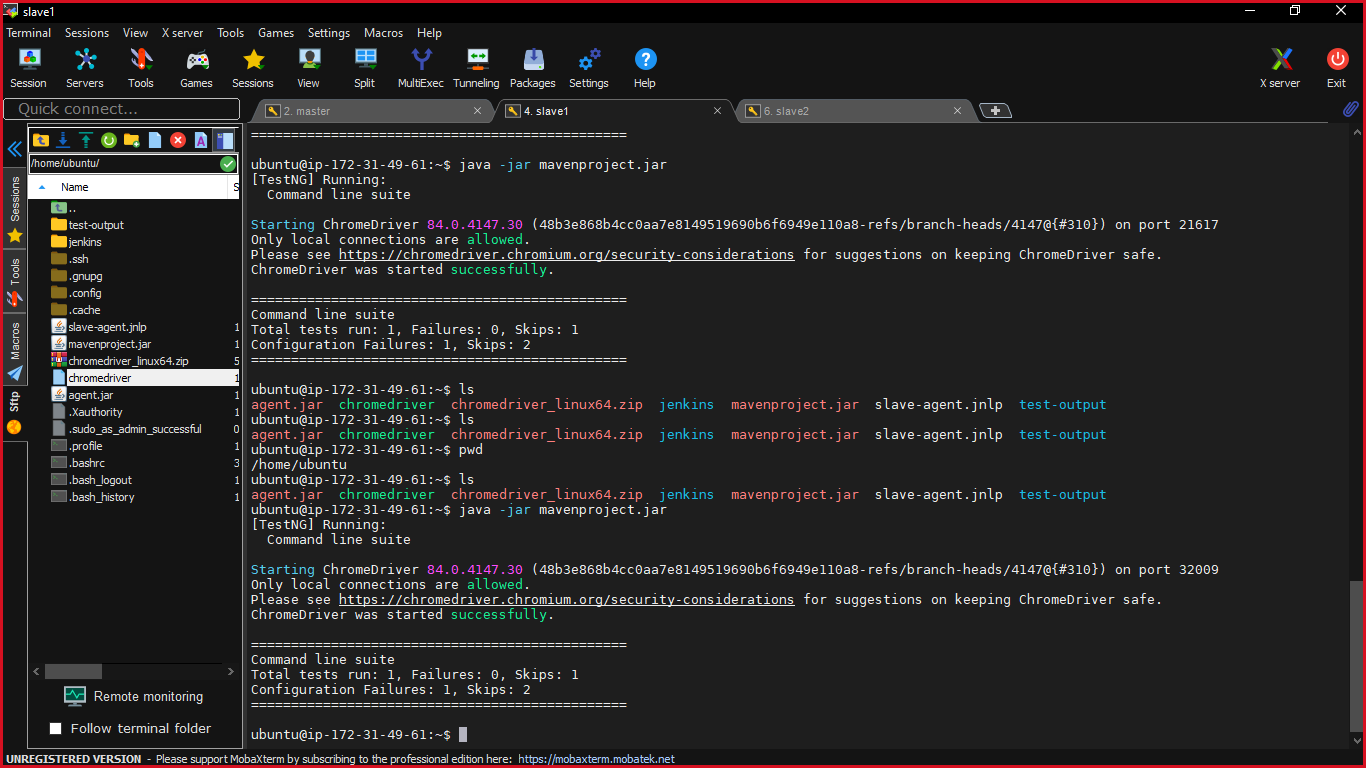
Till here I have completed the 3 tasks of project

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 testserver.

Code for selenium:

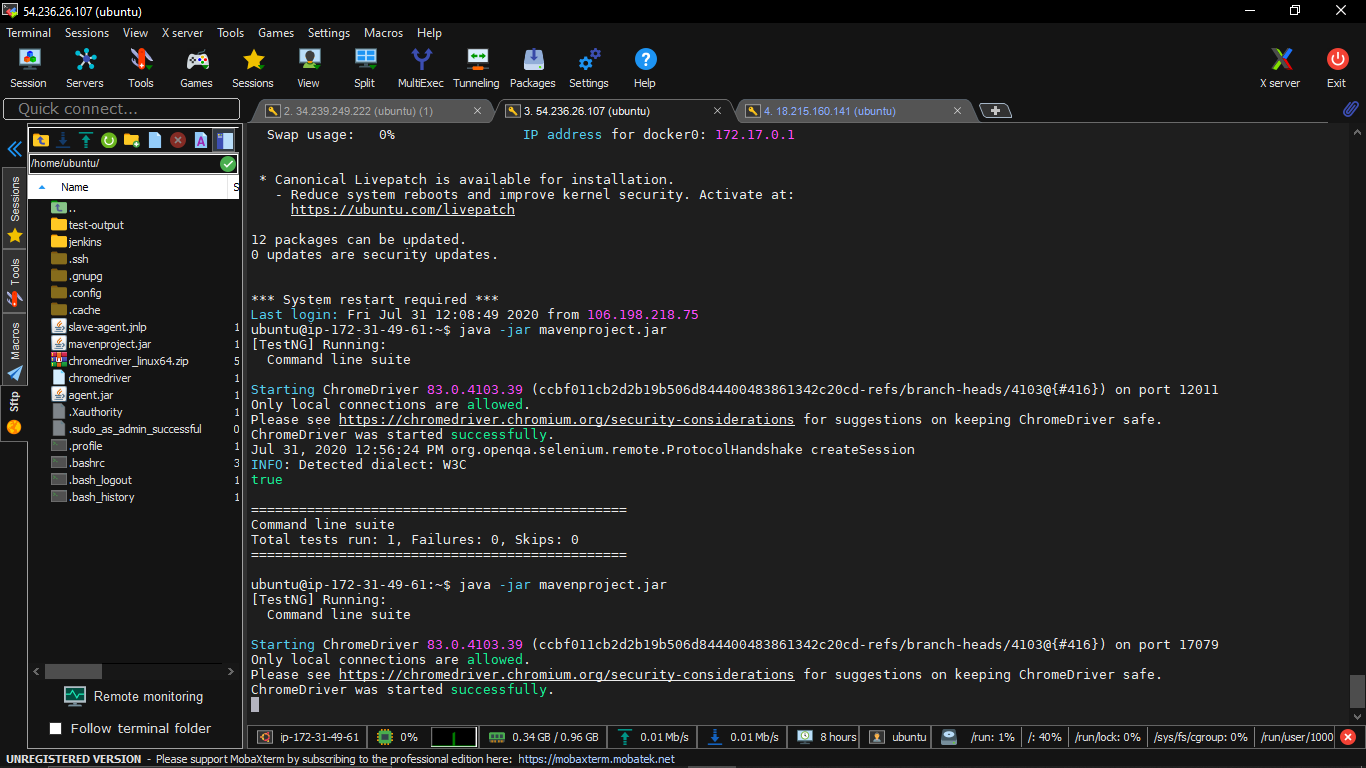


Now for this task I have used selenium :

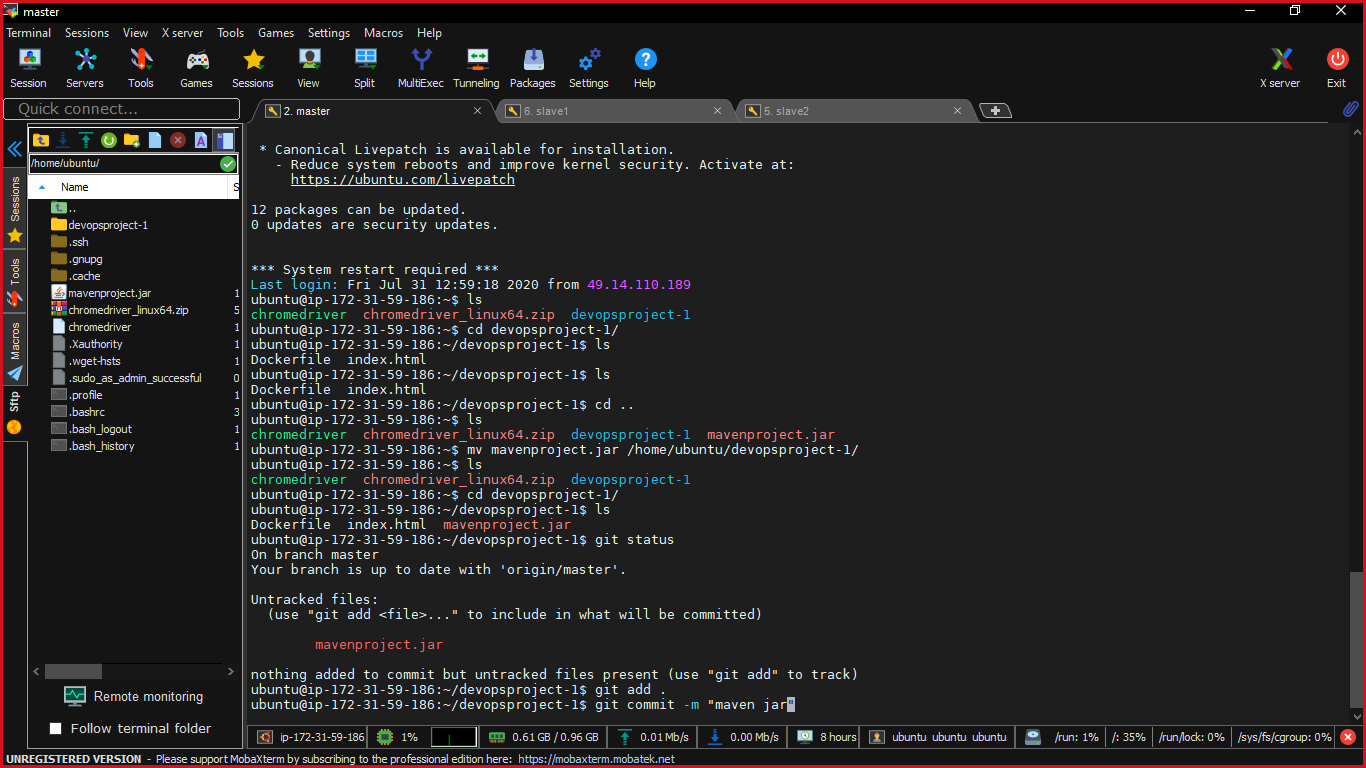


Got some configuration failures.

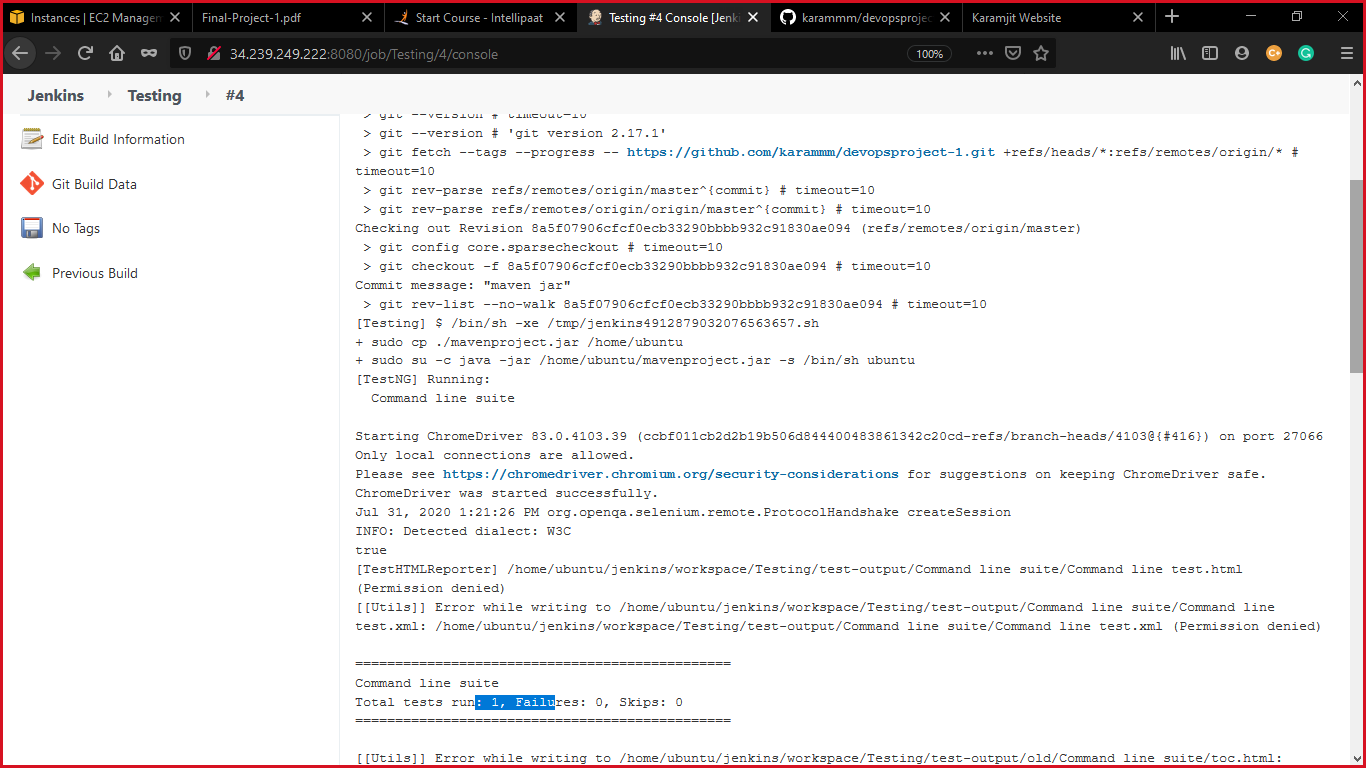
After resolving them finally the test case passed in test mode

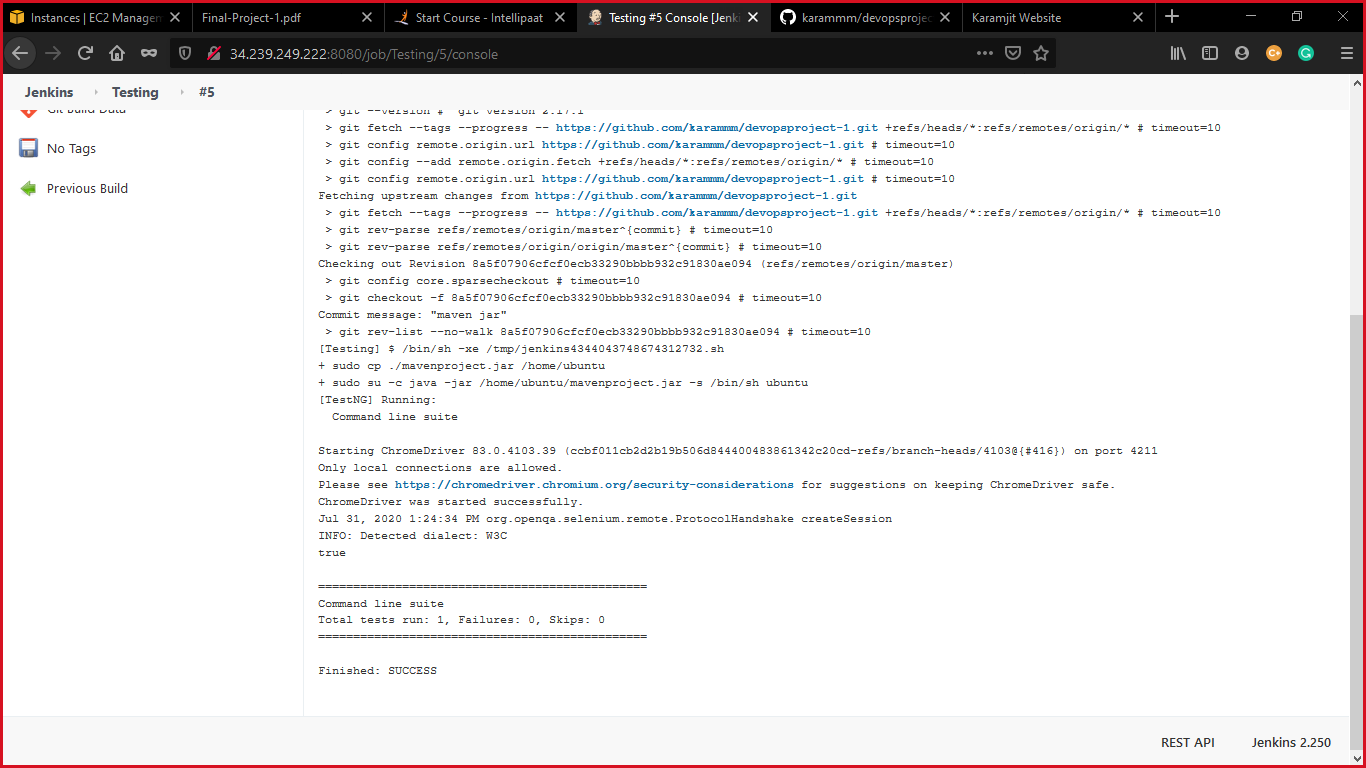


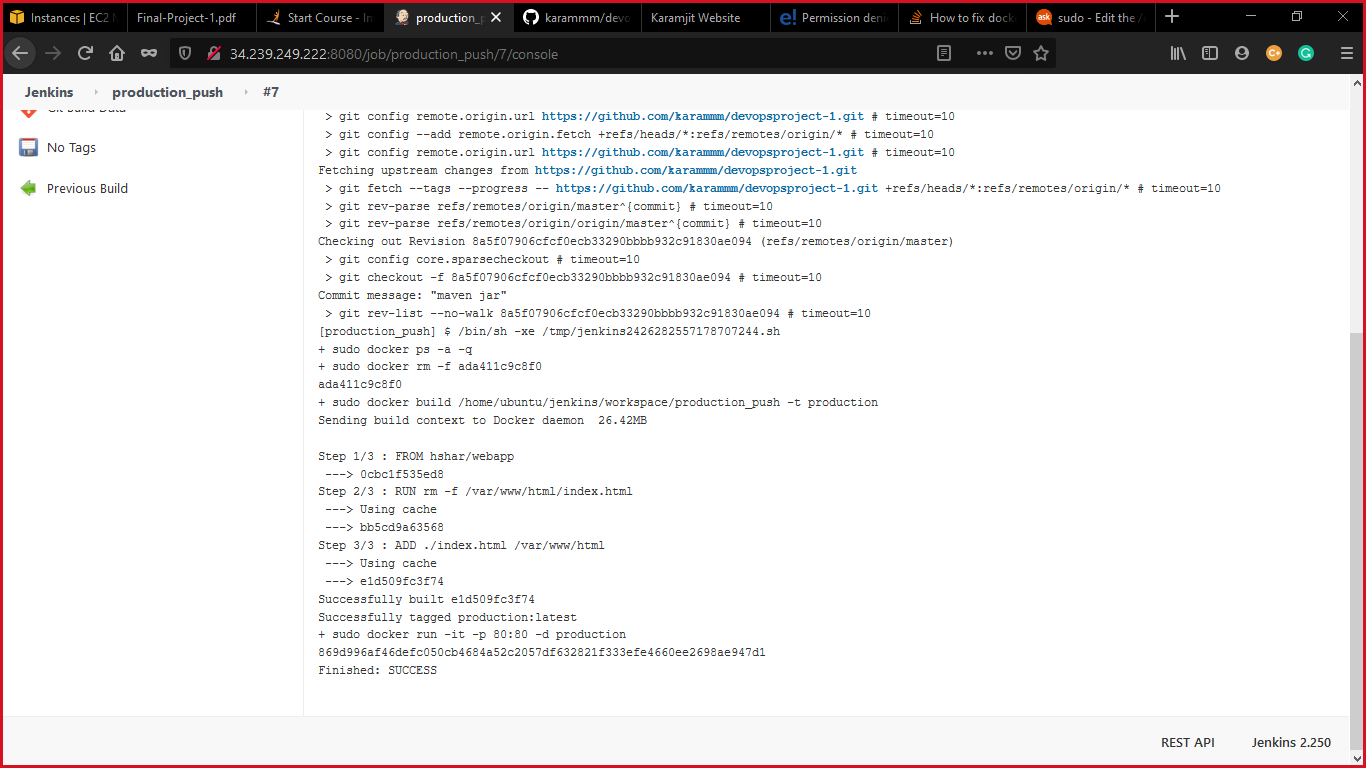
Now pushing maventest jar file to github



After running the test job I got no failure.





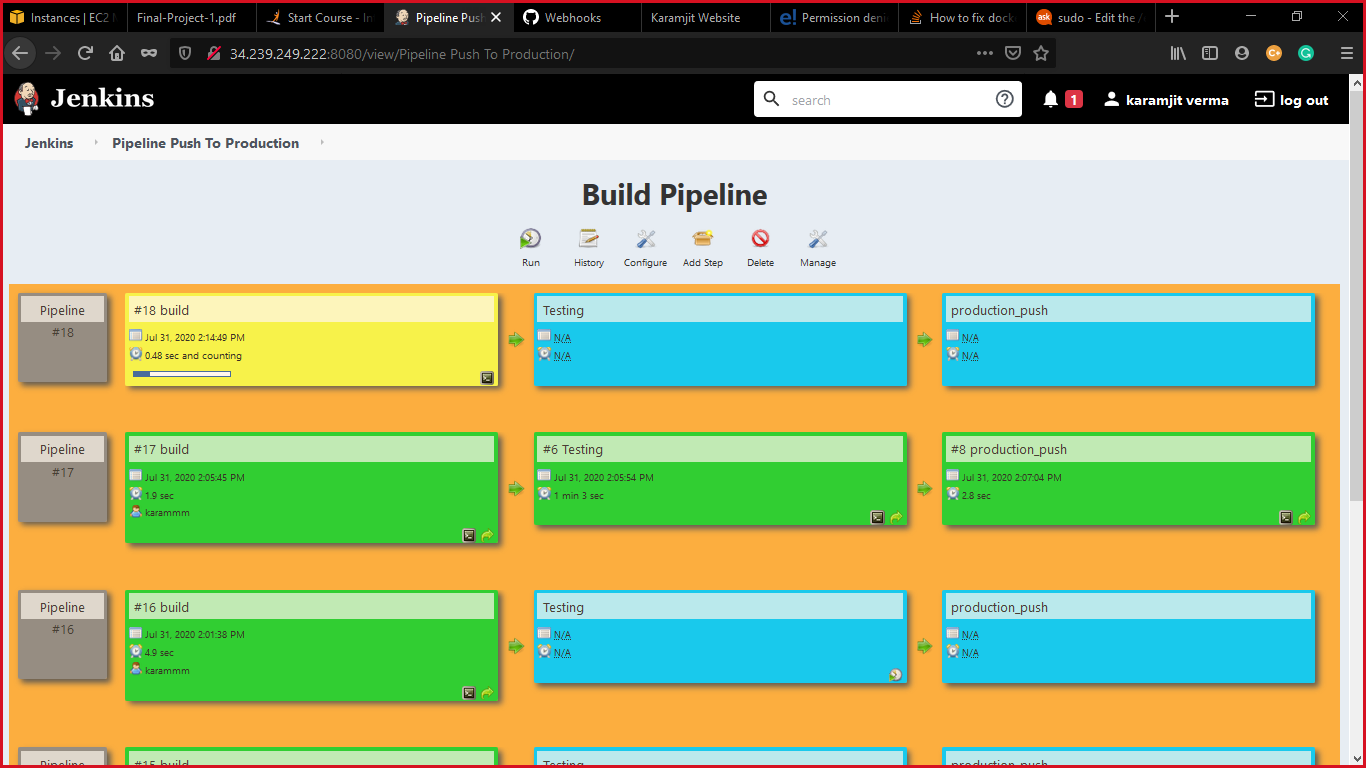


Task 5: This is the pipeline of tree separate freestyle task as shown

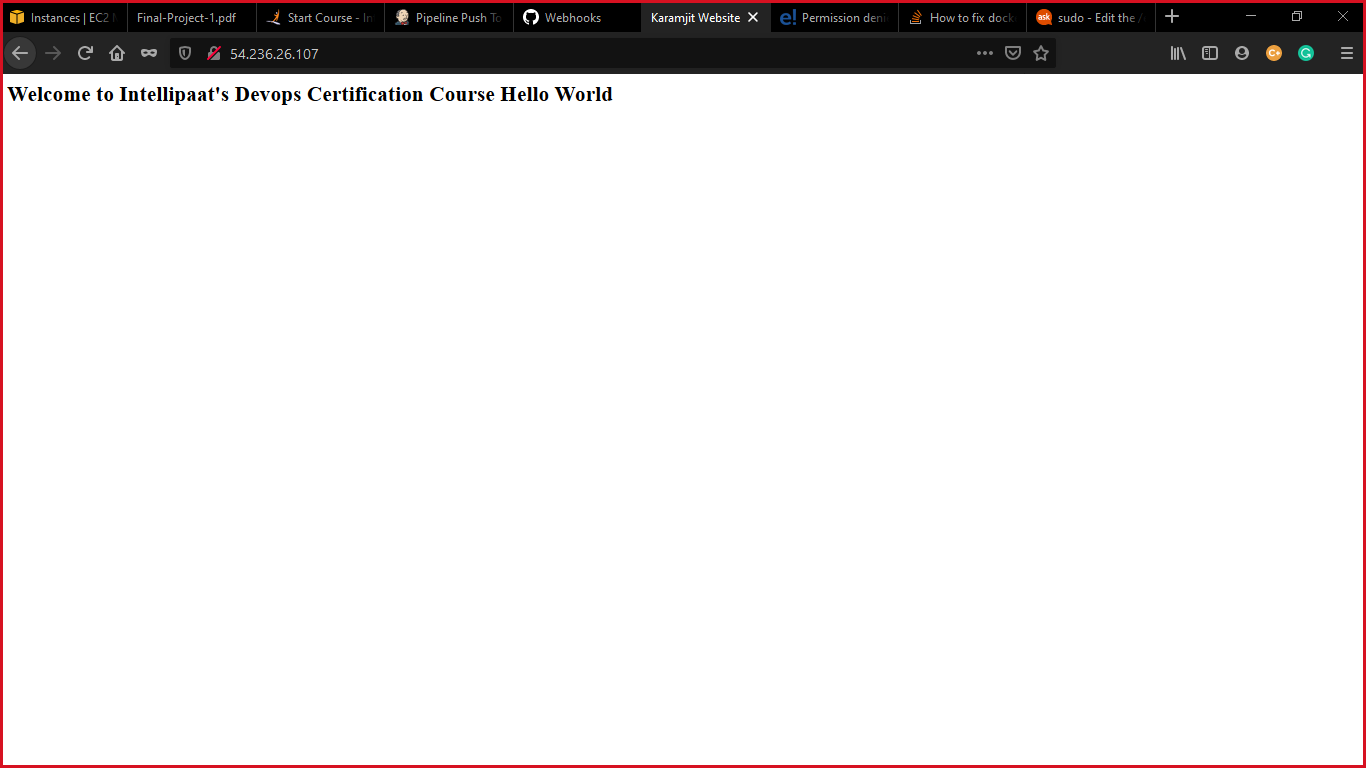
1st is the Build

2nd is the testing

3rd is the production push

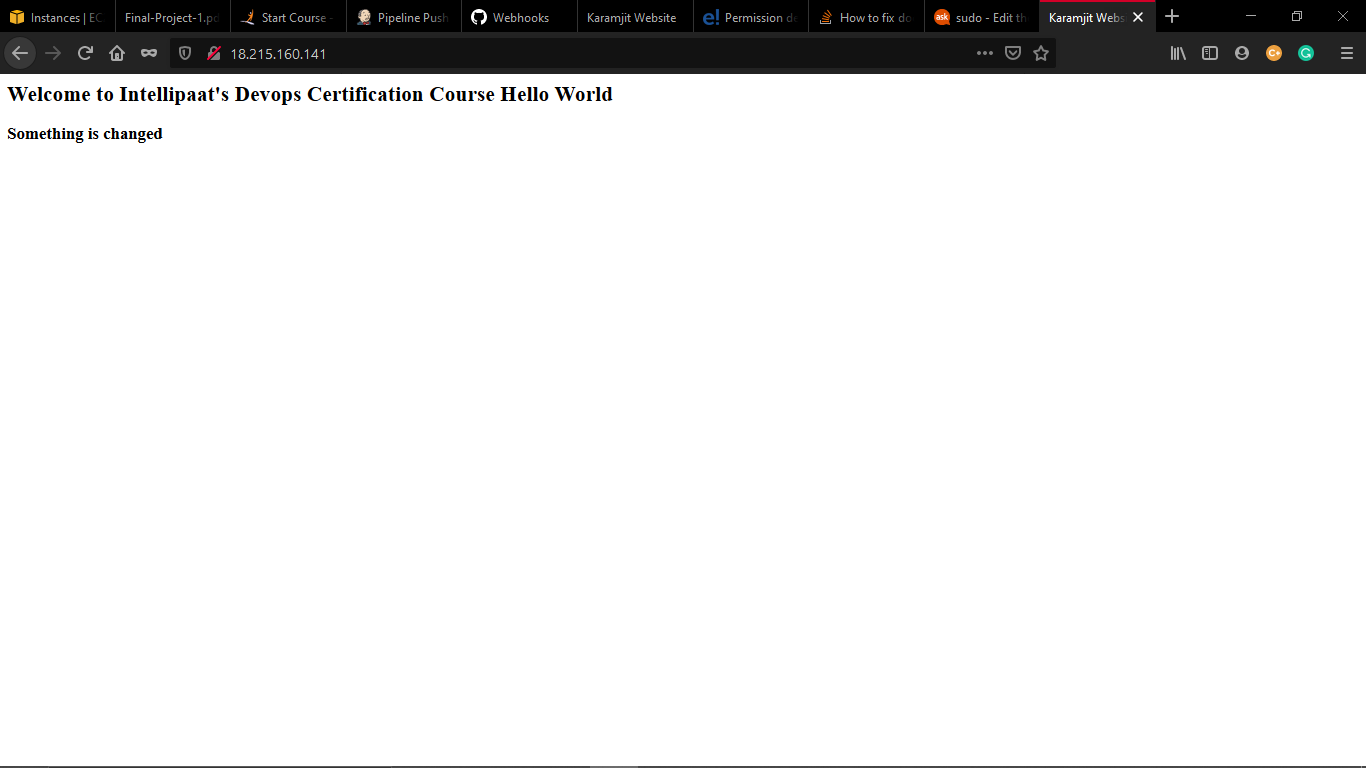


This is the website after success of testing by selenium:

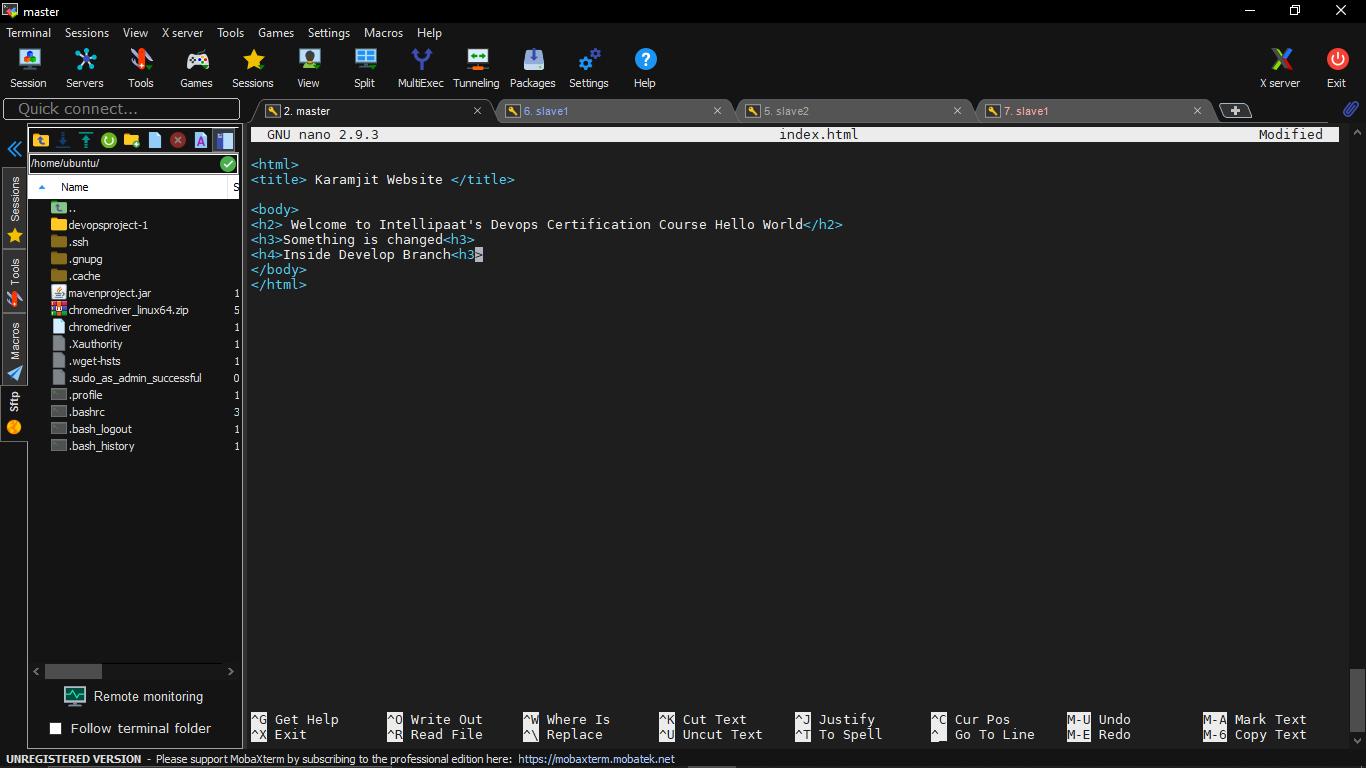


Testing is passed.

And if I have changed something on the repository the change is tested and reflected on the website through webhooks.



Some more testing:

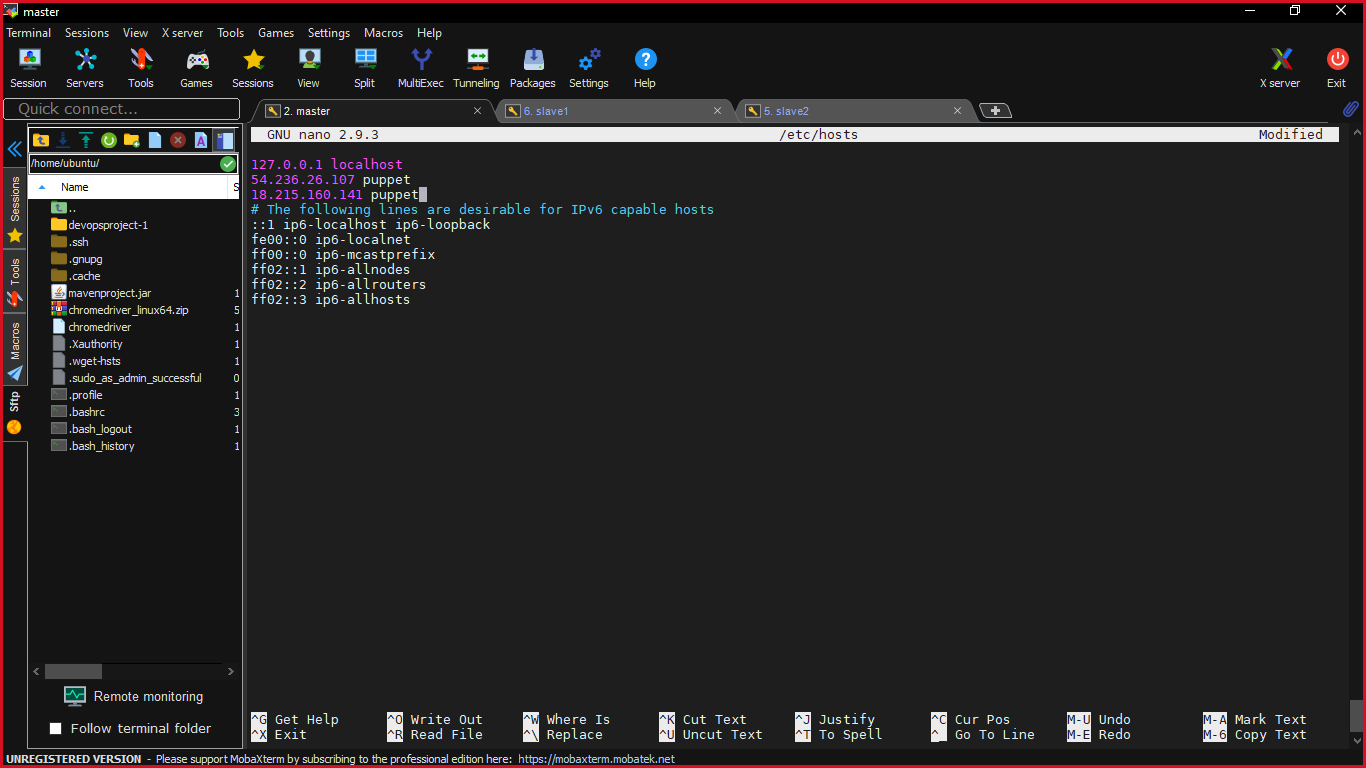


Task 6: Since you are setting up the server for the first time, ensure the following file exists on both Test and Prod server in /home/ubuntu/config-management/status.txt. This file will be used by a third-party tool. This should basically have the info whether apache is installed on the system or not

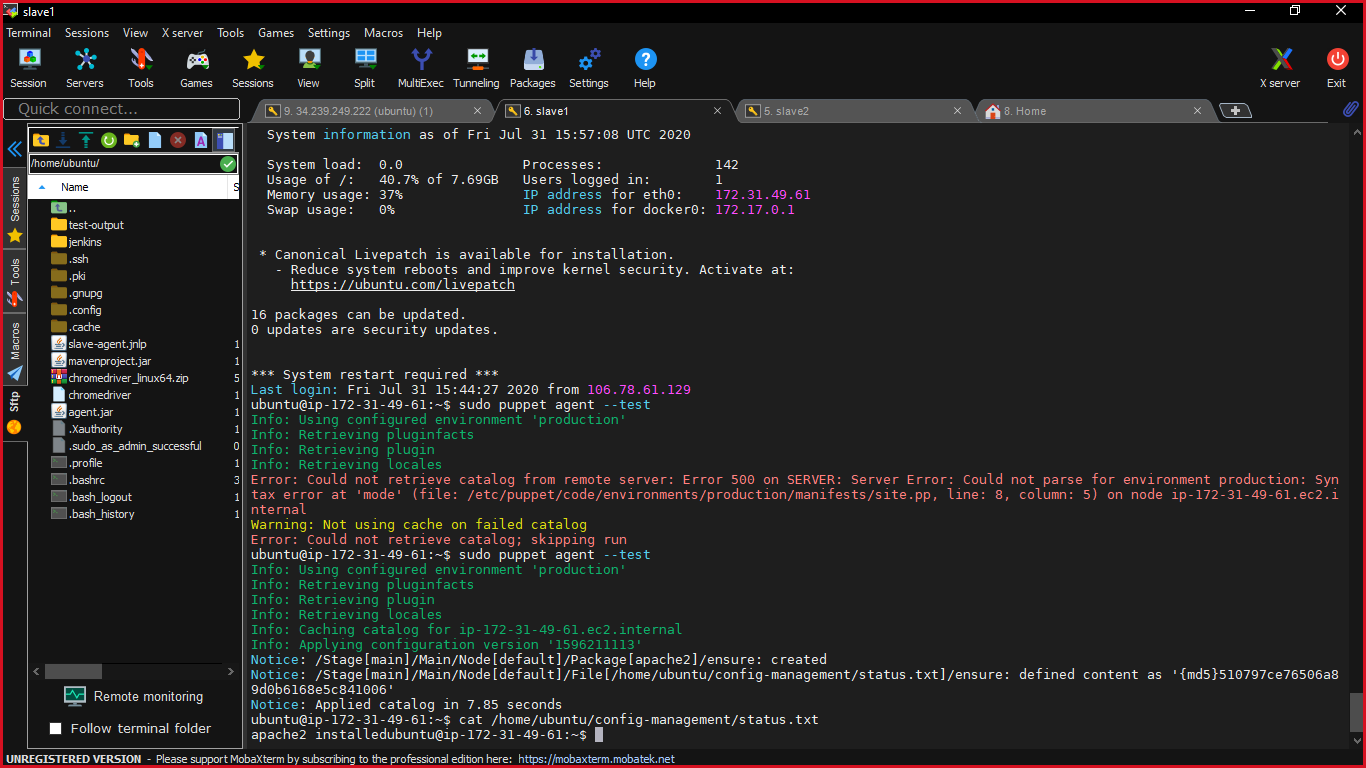
The content of this file, should be based on whether git is installed or not.If apacheis installed => Apacheis Installed on this System"If apache is not installed => "Apacheis not installed on this System

----------------------------------------------------------------------------------------------------------------------------------------------------

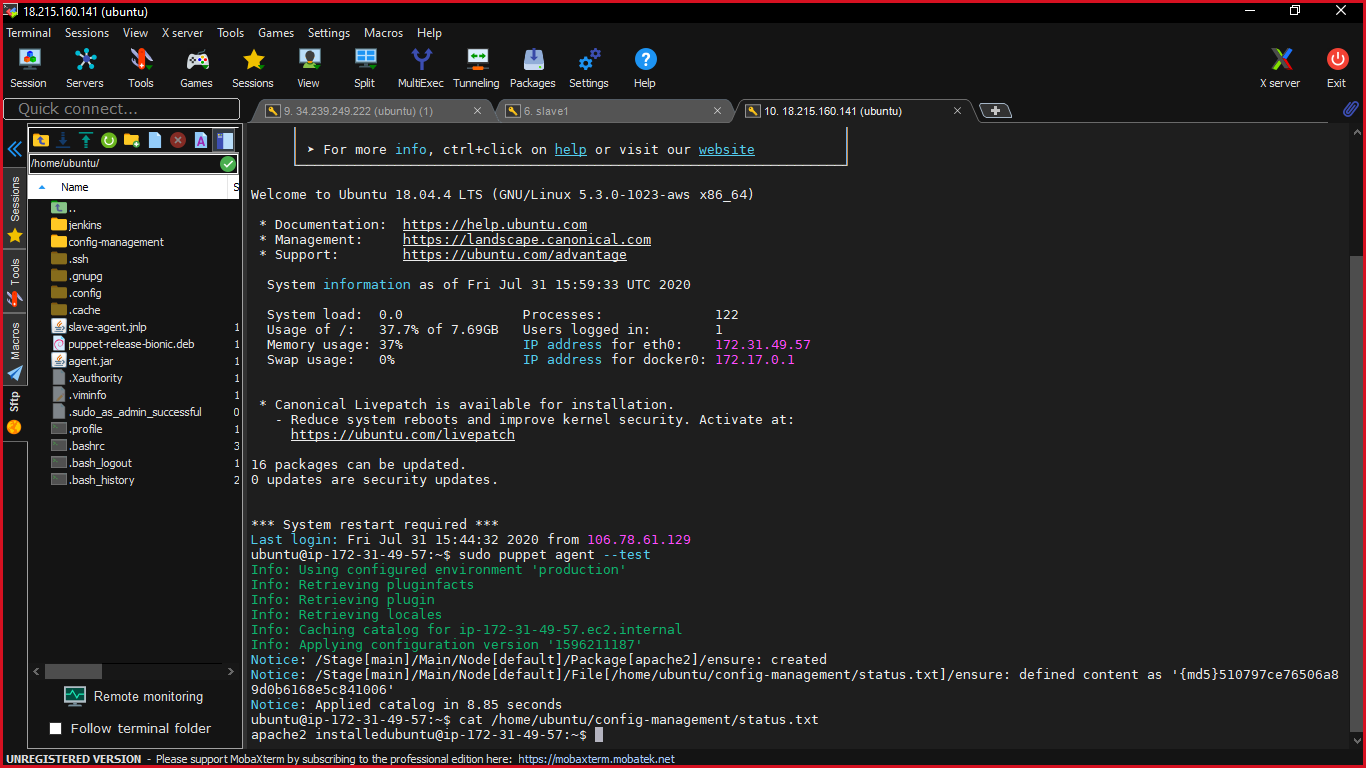
Let go with puppet installation after that installation on master I have added the host in the master both testing and production server:



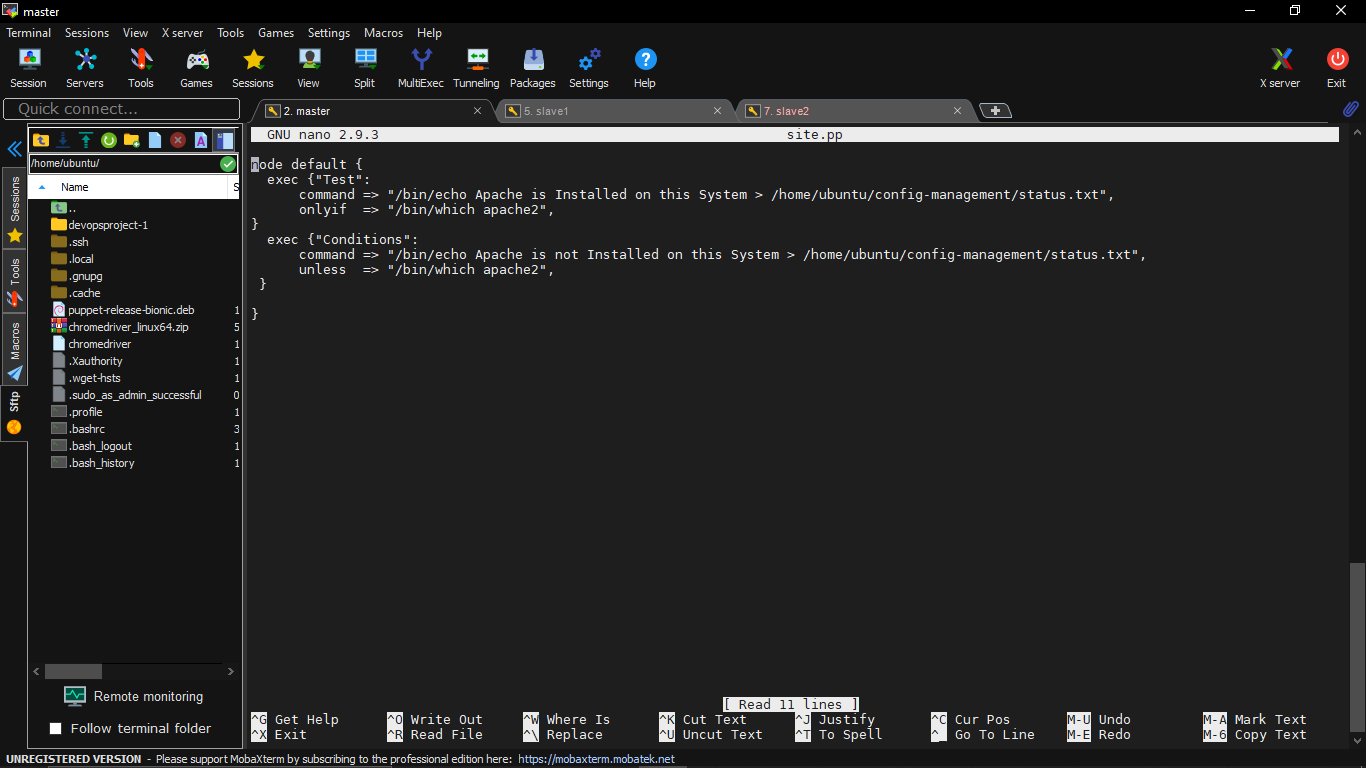
Testing the agent:

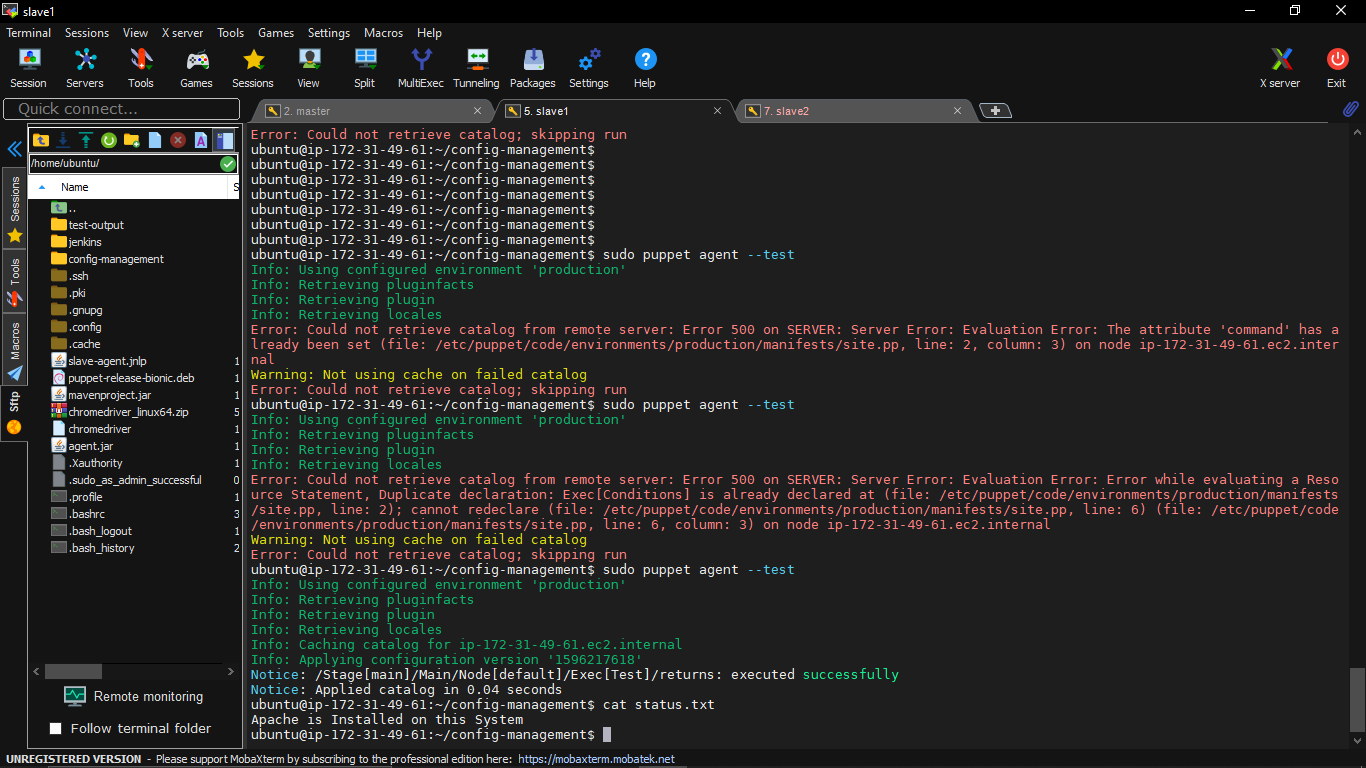


Production server:

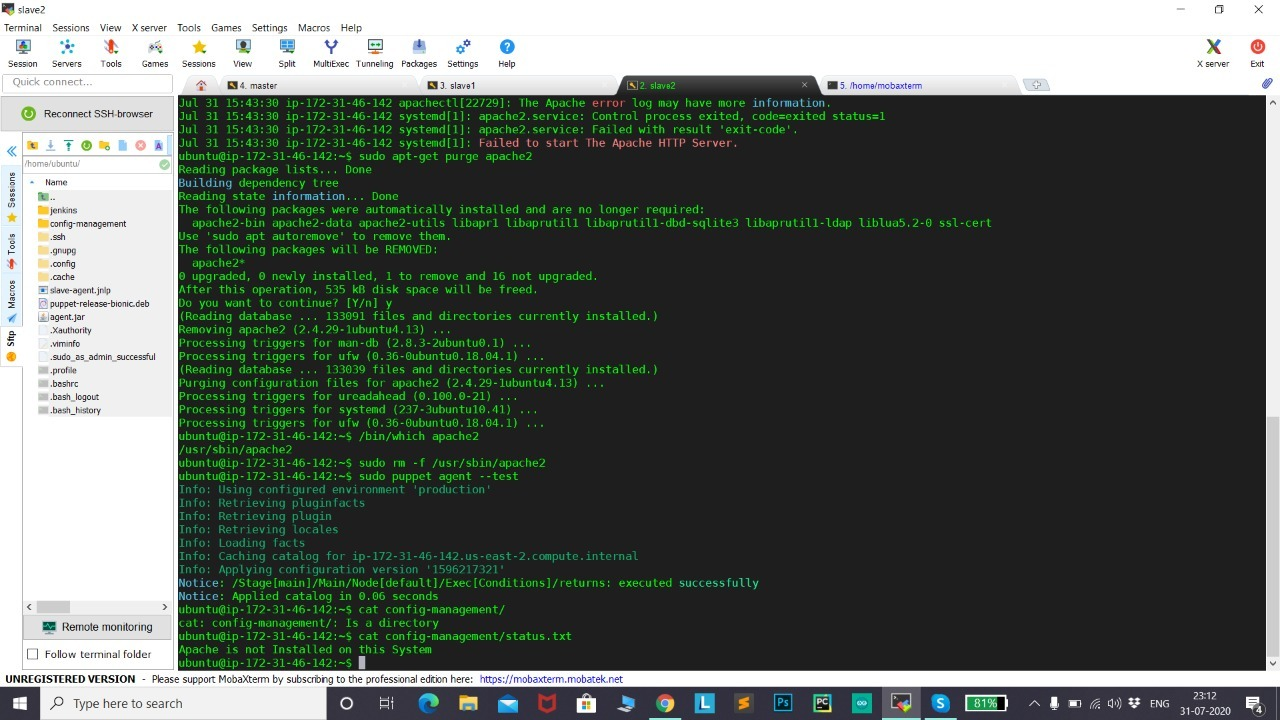


Making a site.pp file that stores a file on the path /home/ubuntu/config-management/status.txt containing the information about the apache is running or not with the help of puppet.



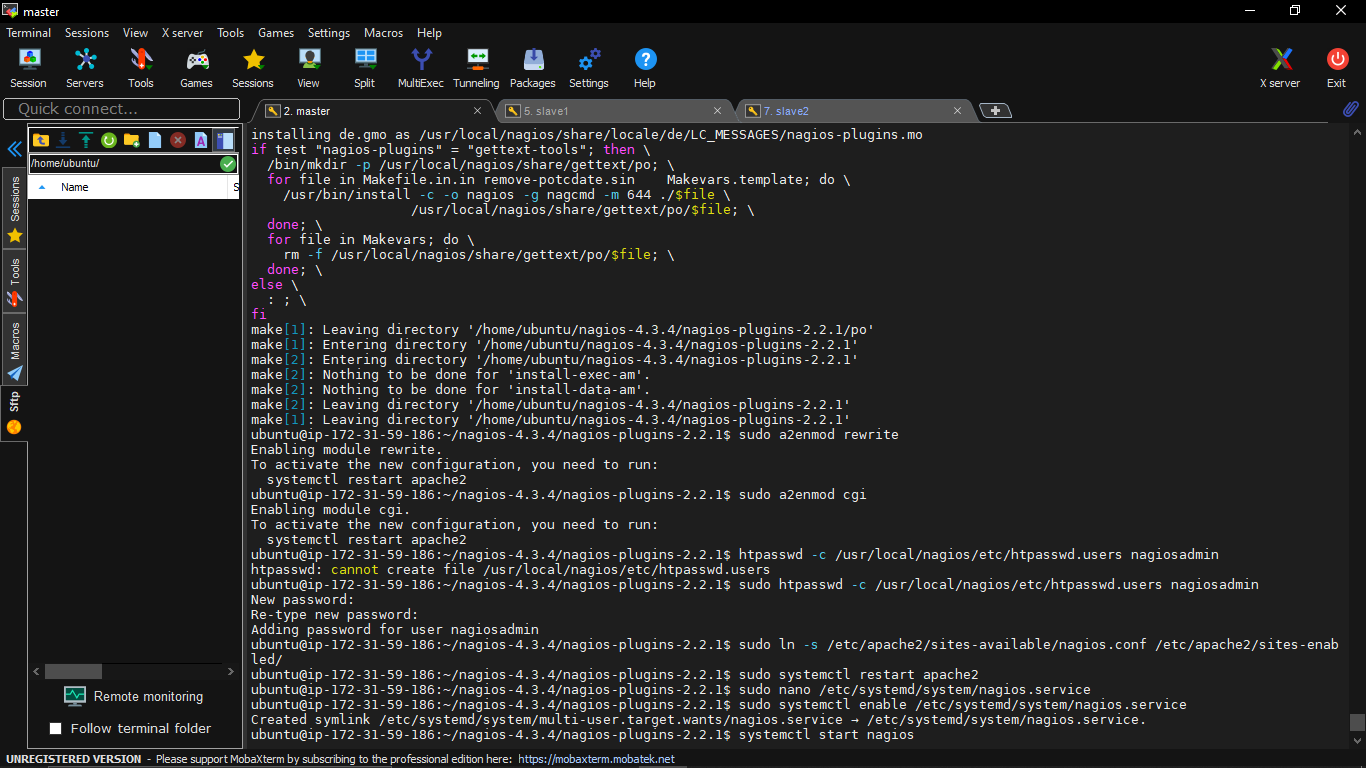
Now you can see that a file is created on the production server containing information about whether the apache is present on the system or not.

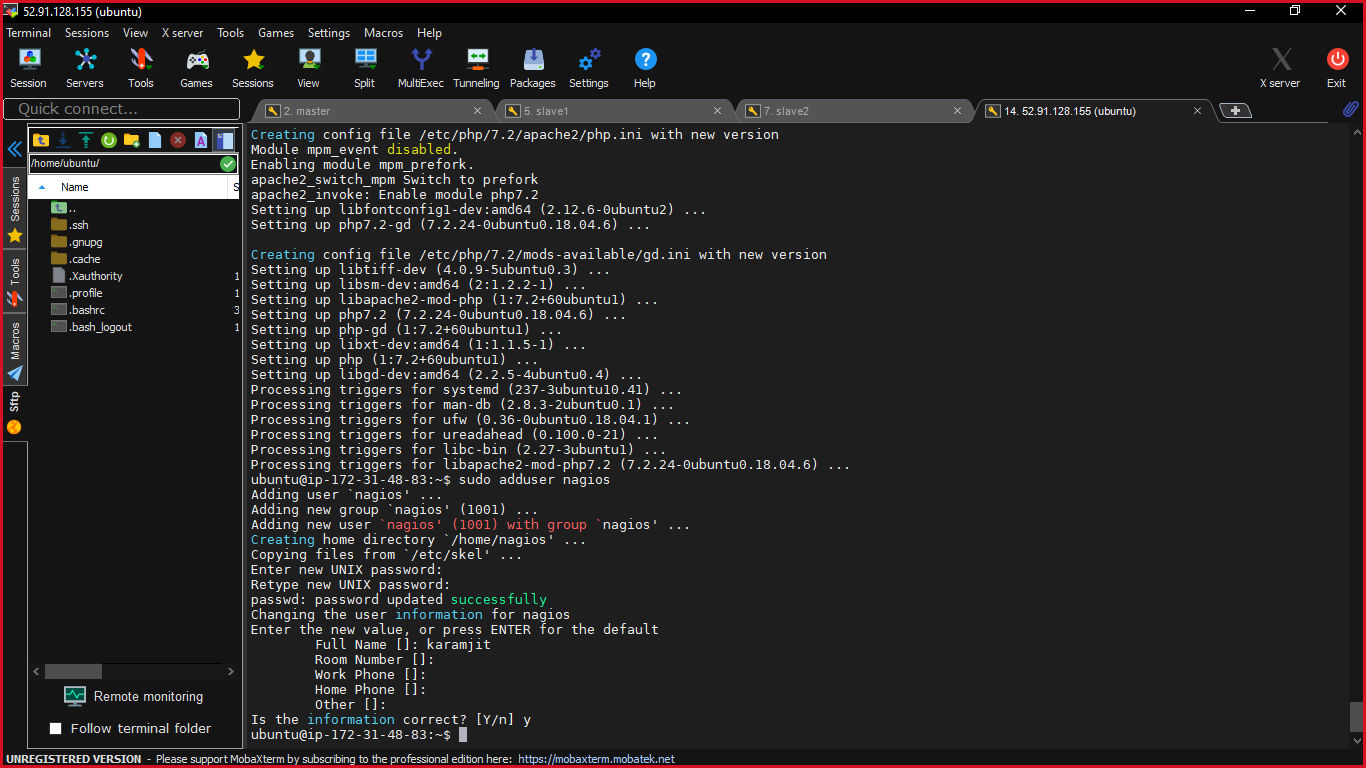
Now if I stop apache2 then I will get apache is not installed on the slave

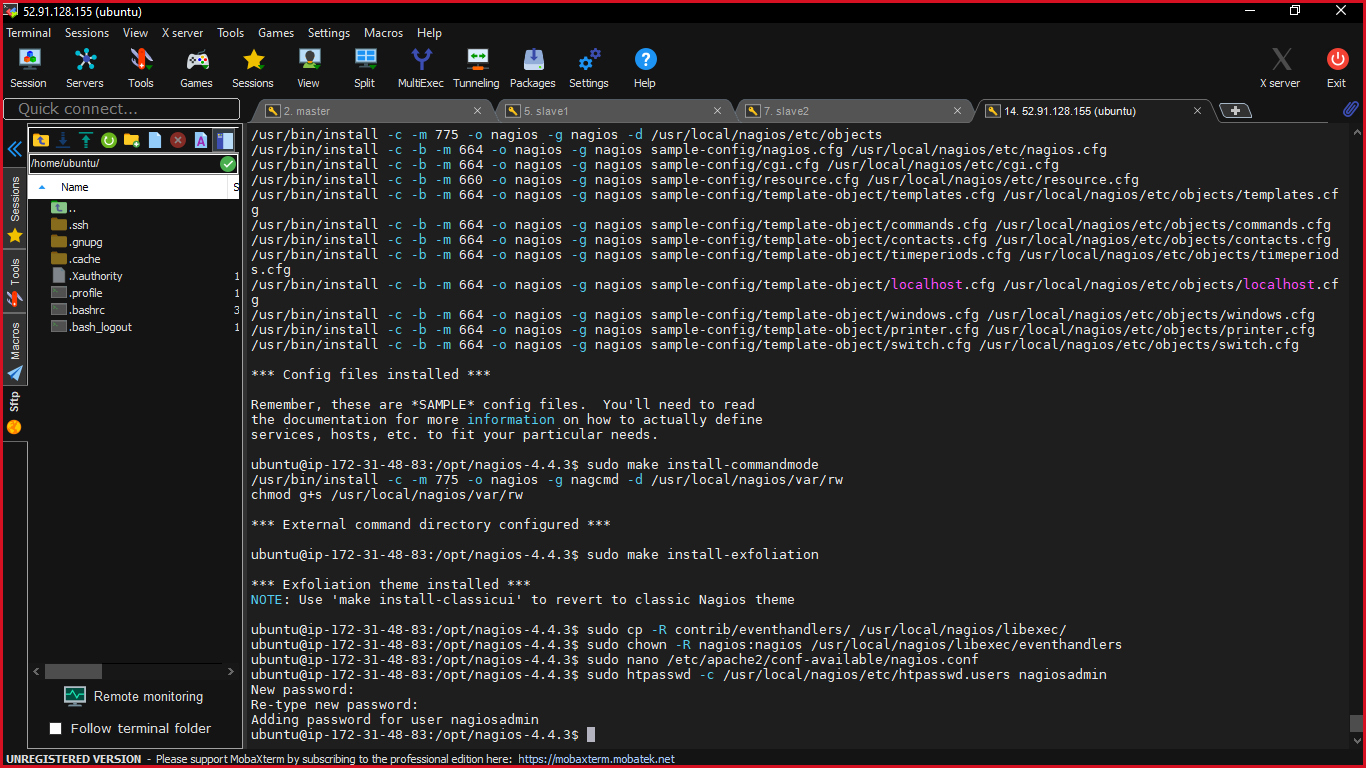


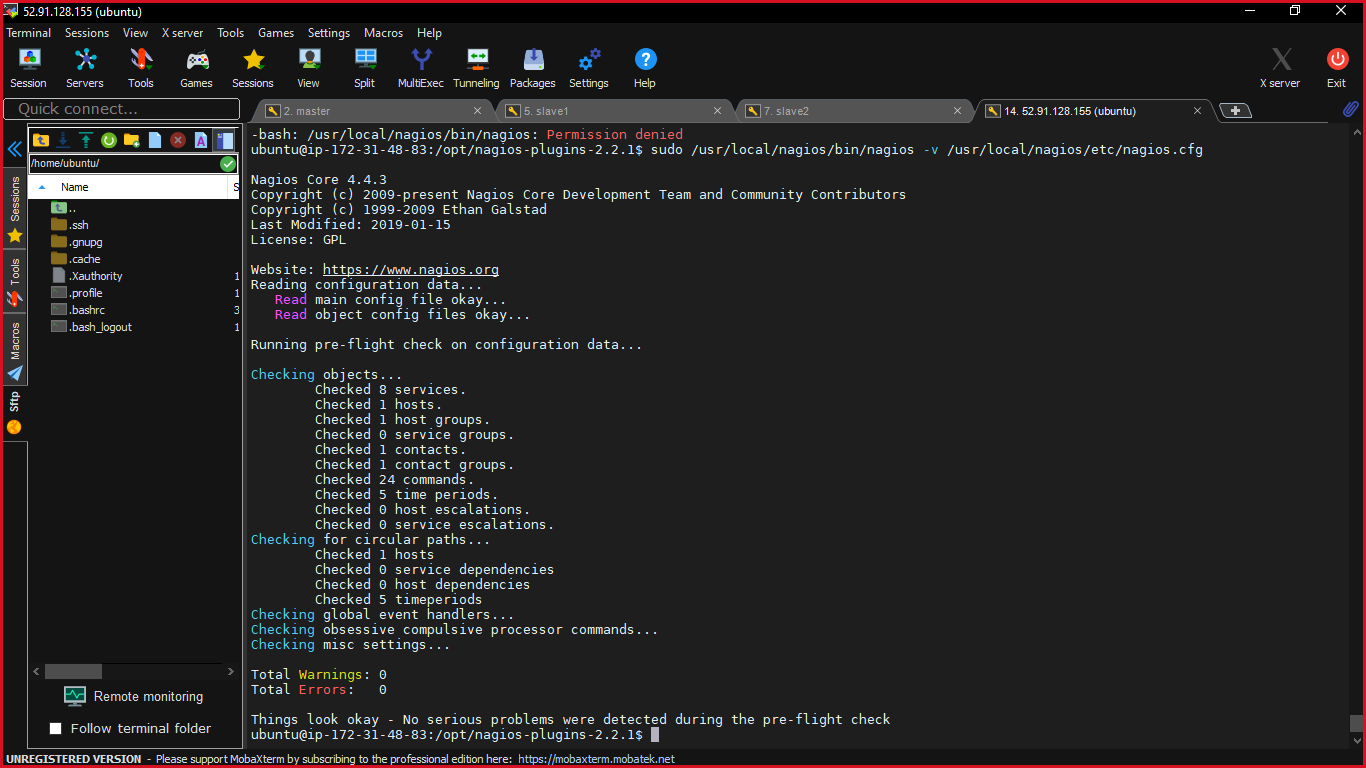
Now coming up to last task Task 7: 7.Create a Monitoring Service for the website on the Production server:

Installing nagios on master server:

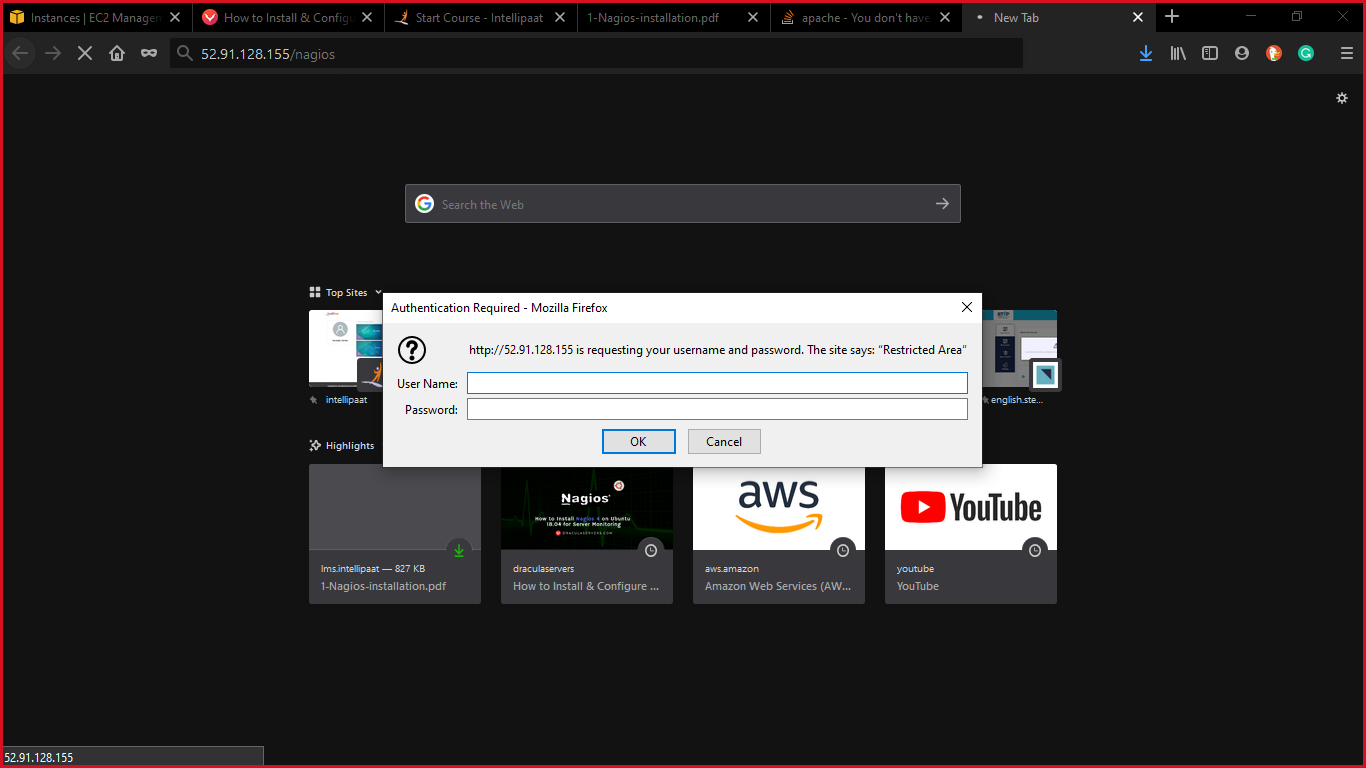


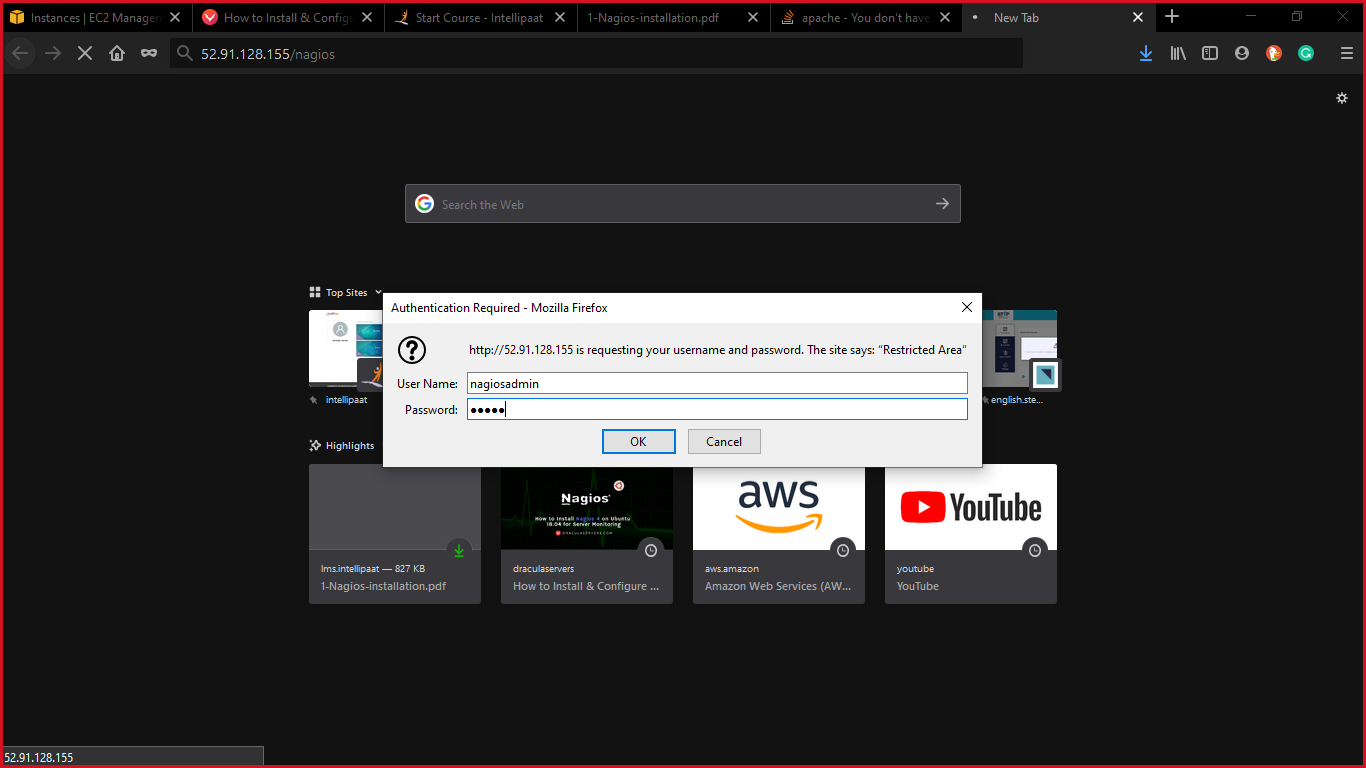




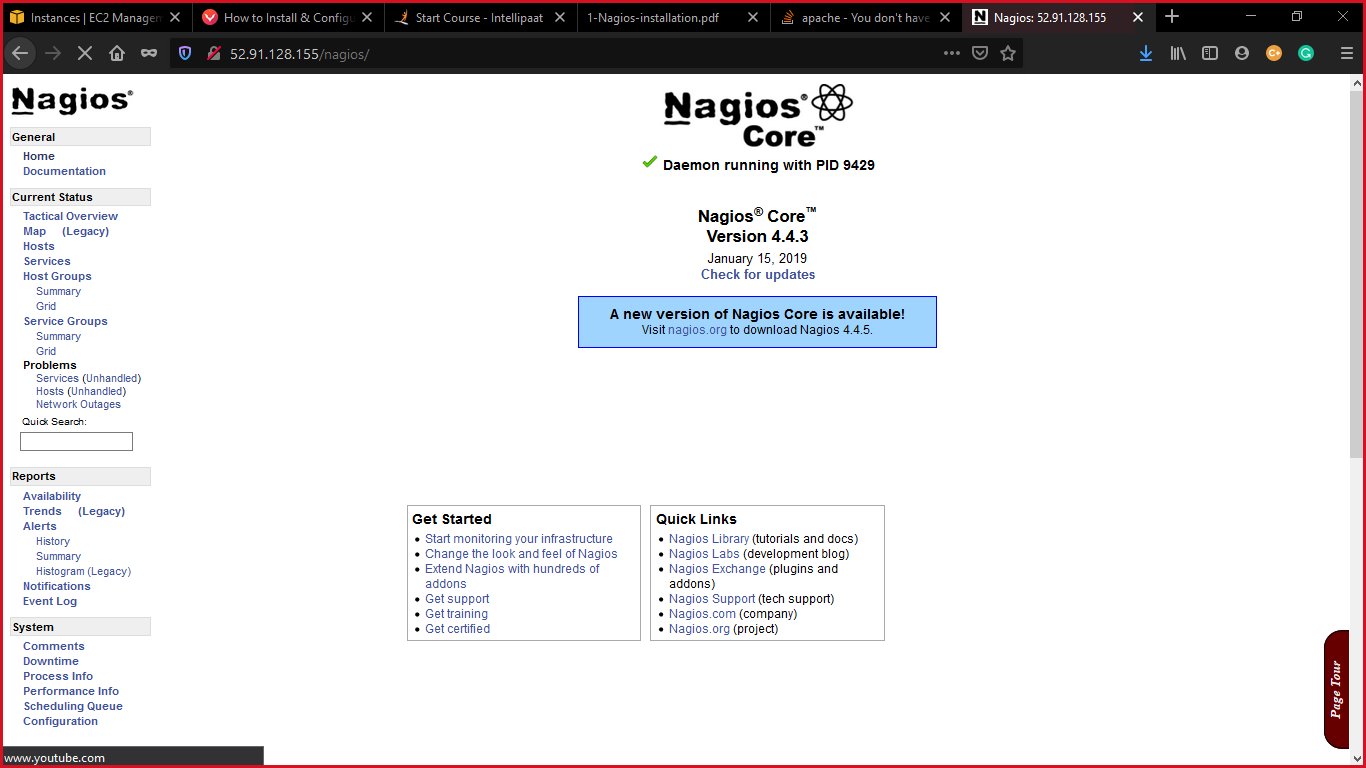


Entering user id and password which was setup while installation:



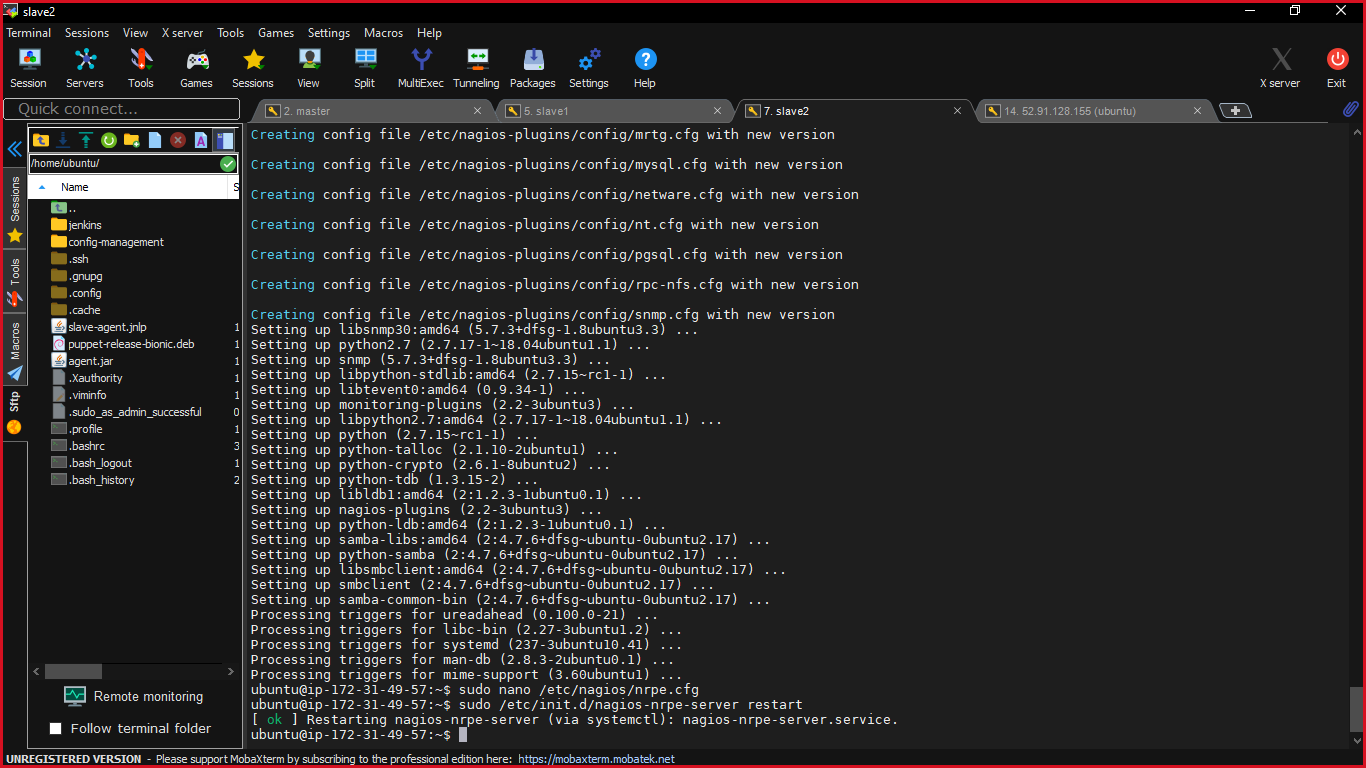


Finally we have the nagios dashboard

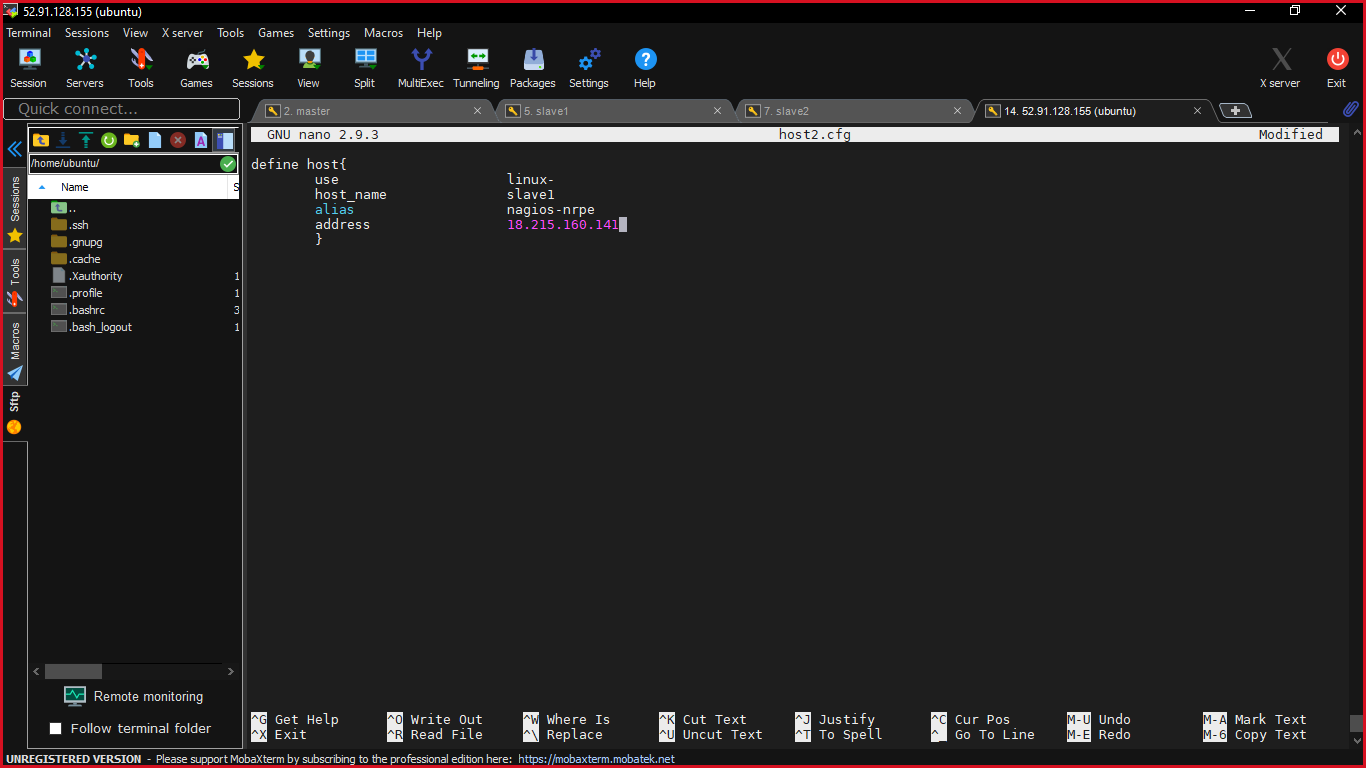


Now adding master ipaddress on slave’s nrpe.cfg file so that our production server can talk to master

The slave 2 is the production server:

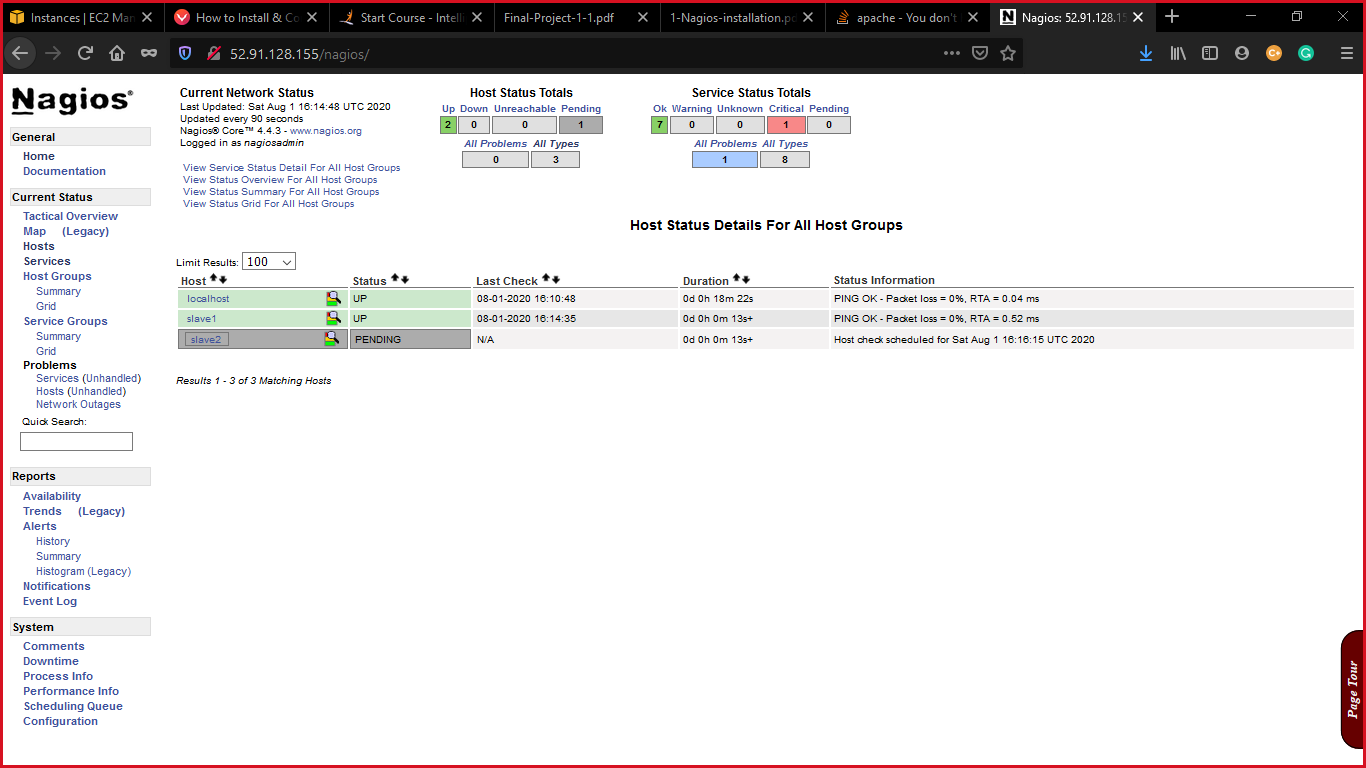


Adding host file on the master node

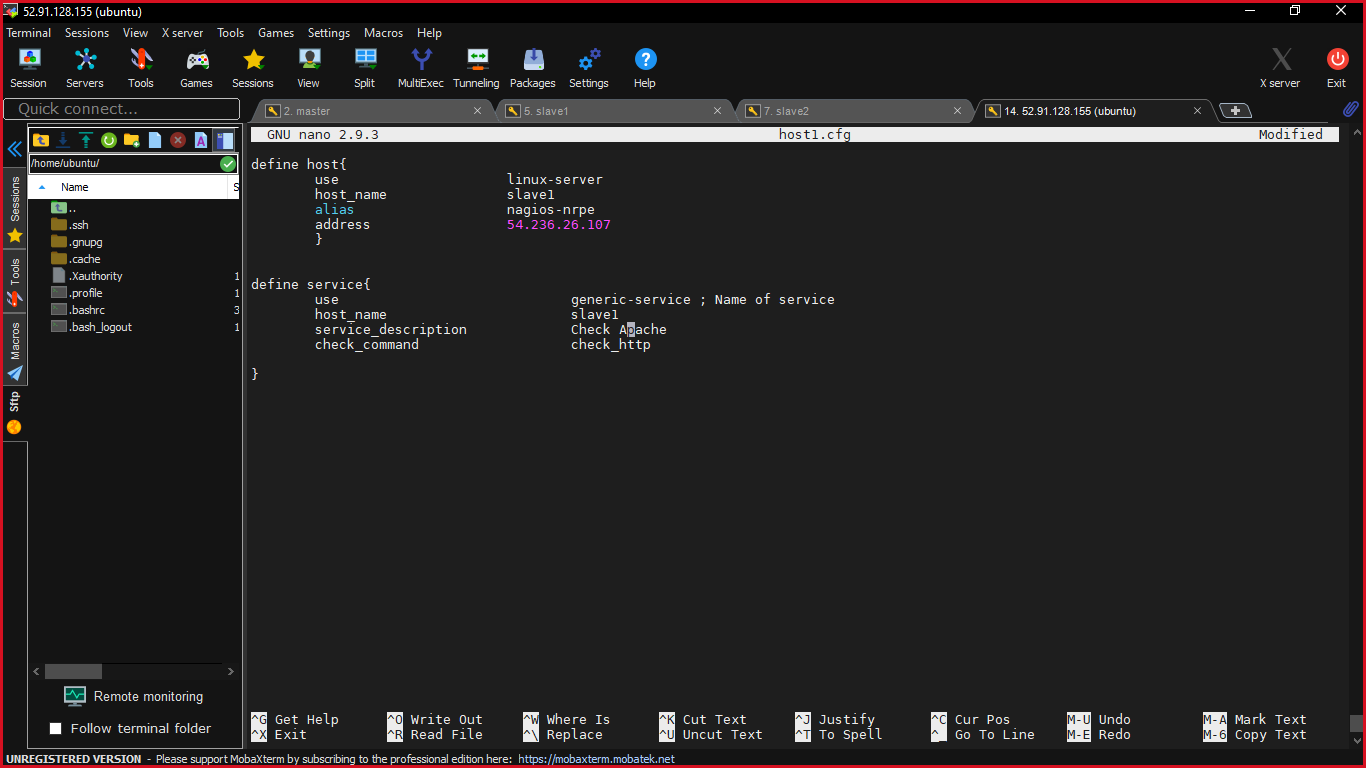


Now you can see the host are finally on the dashboard of nagios:

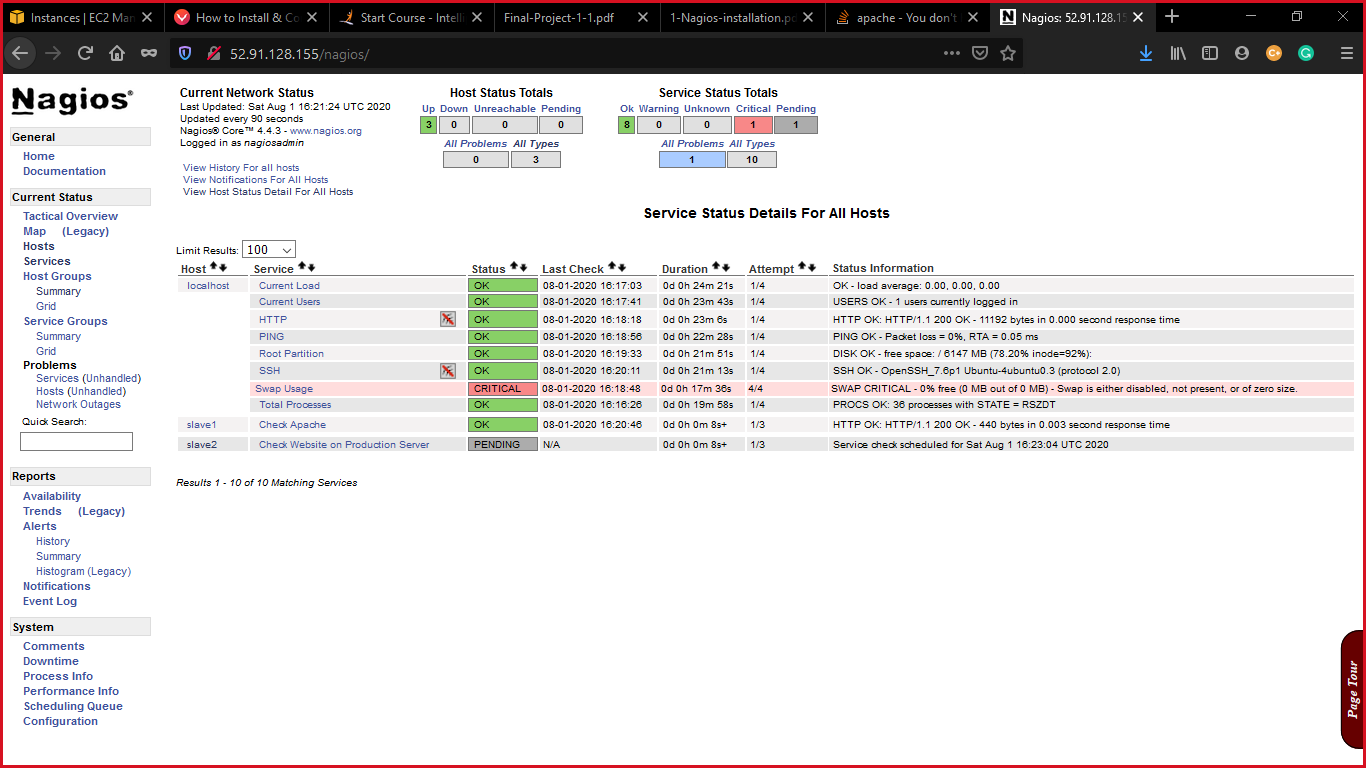
Slave 2 is the production server:



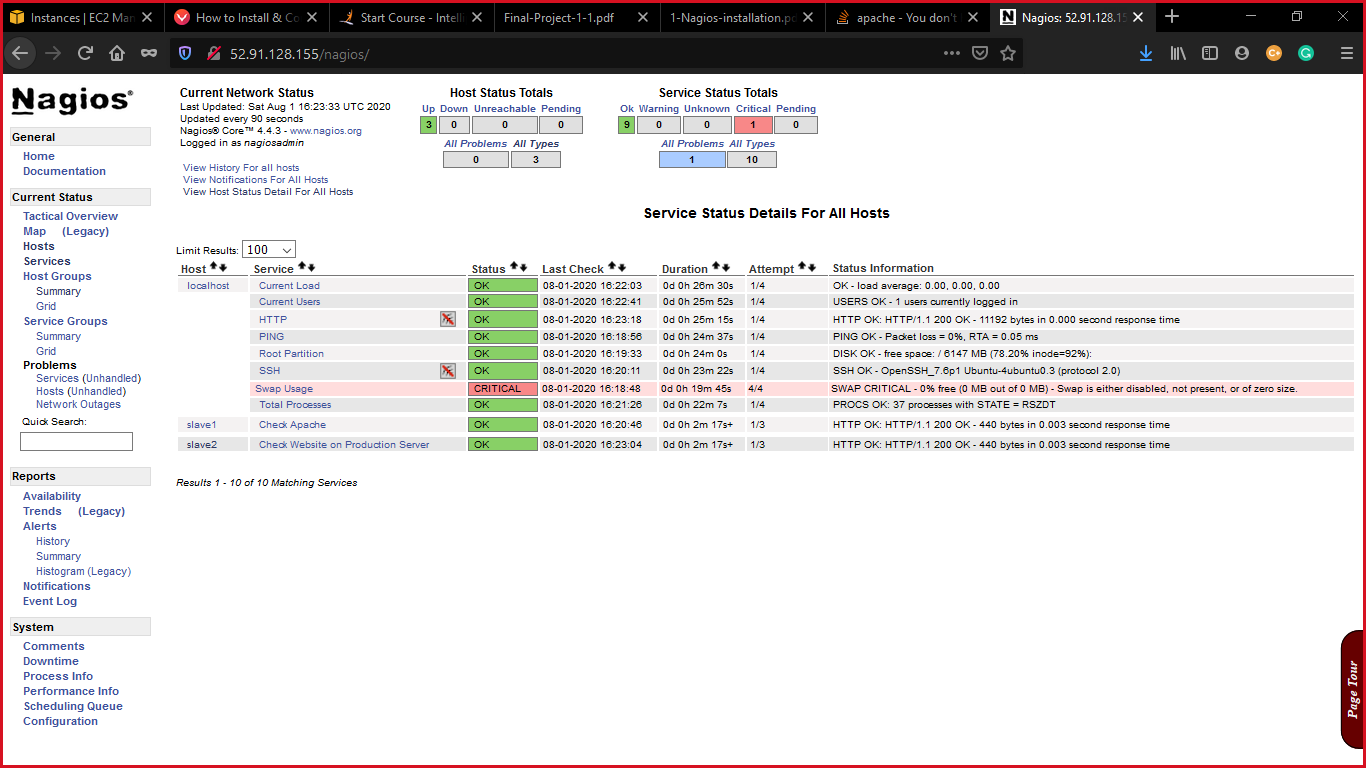
Now I have added service to check whether the website is up and running or not:



Checking apache on production server:



Now finally it is up and running as shown below:



Health of production server:

