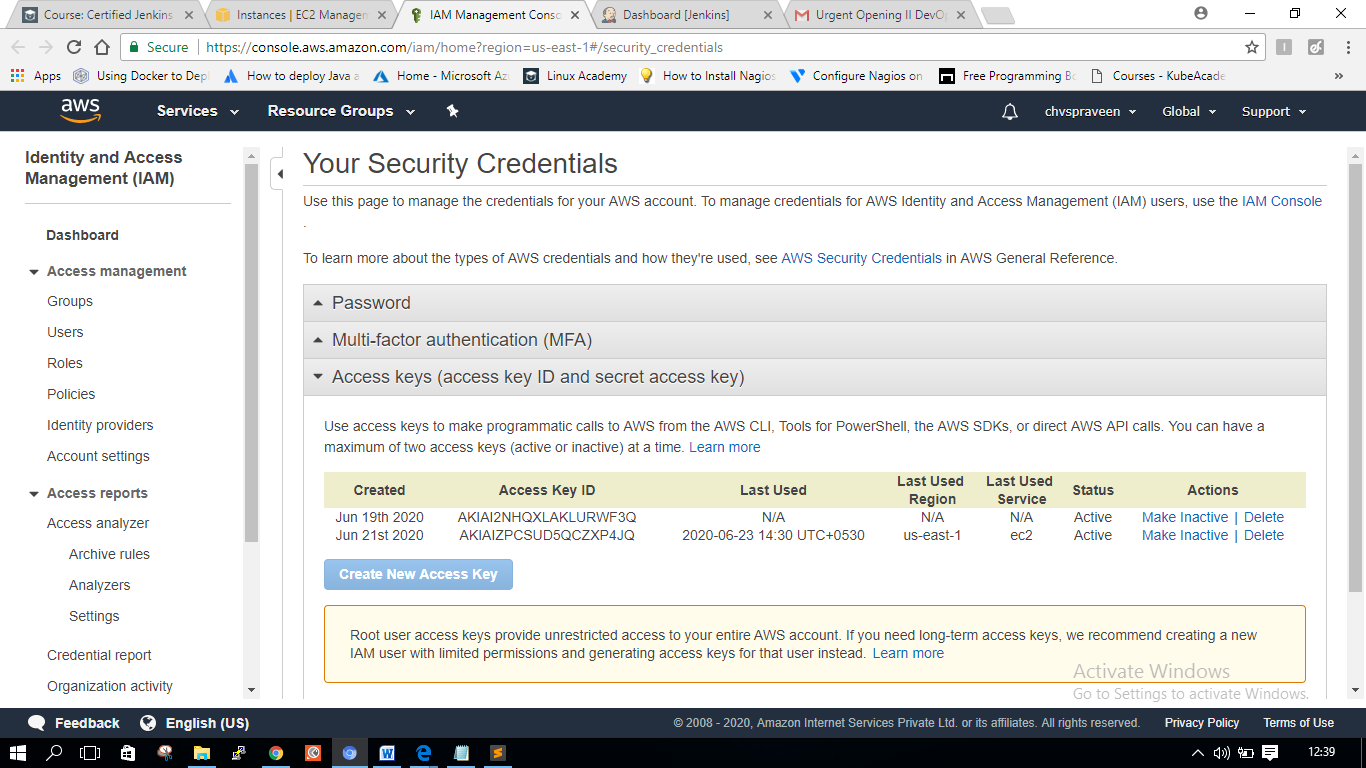
How to crete the aws instance using Jenkins

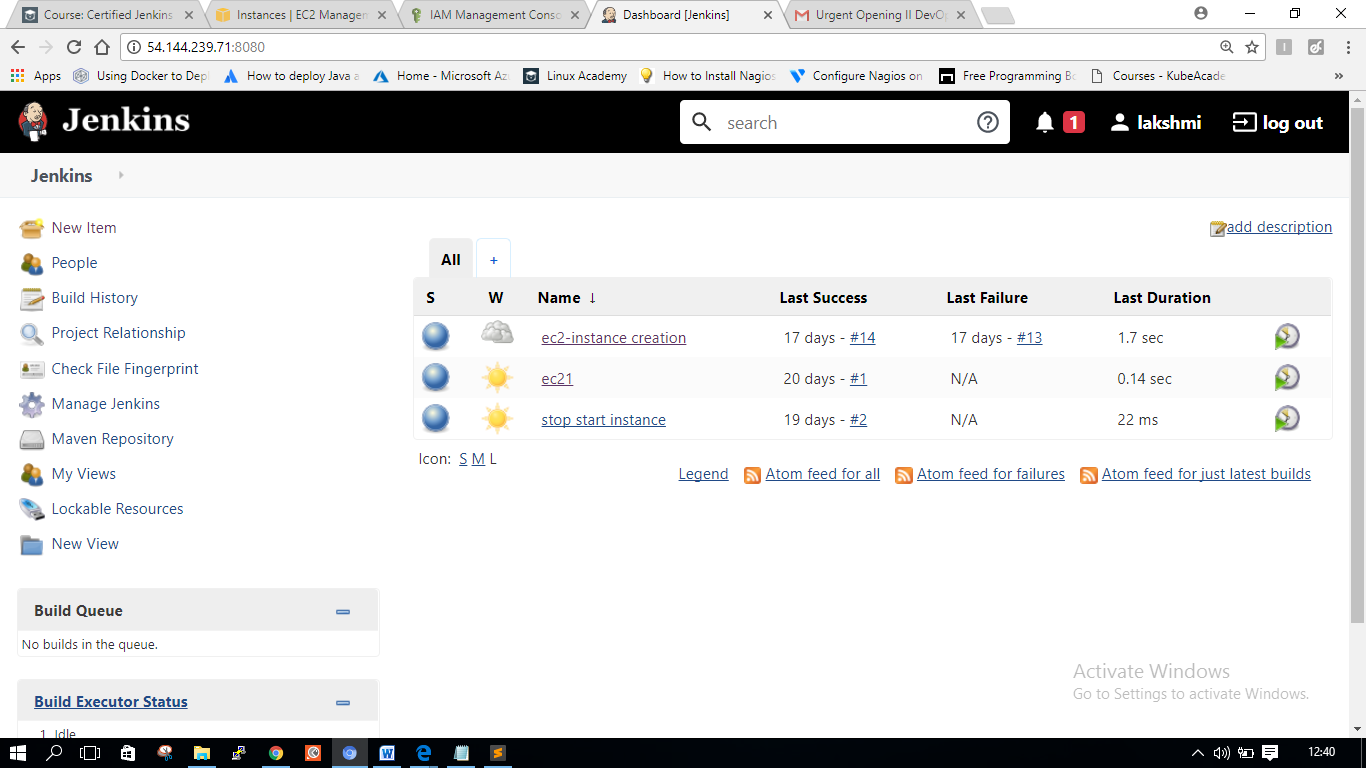
First take aws credintials



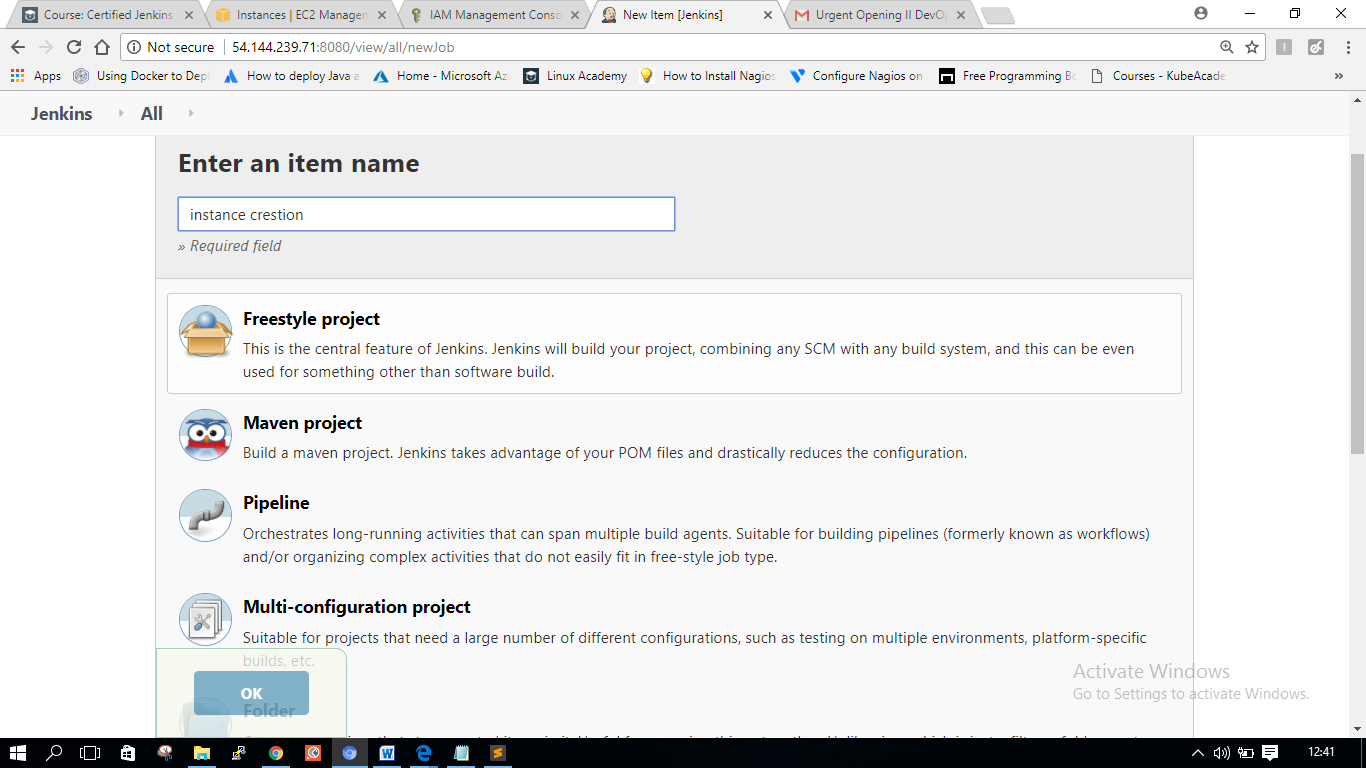
Creste the job

You go yo mange Jenkins -🡪plugins 🡪 install the cloudbee plugin

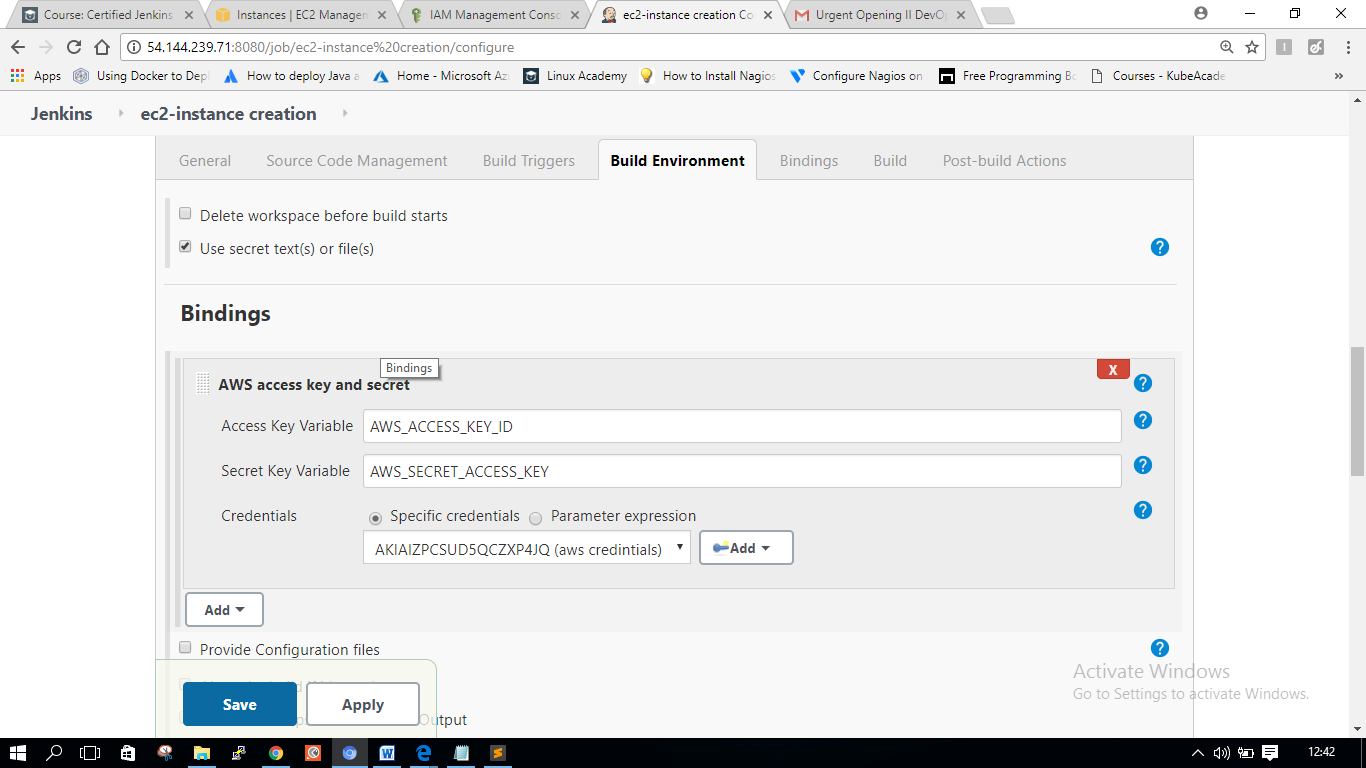
After the continue below procedure



New project 🡪select the free style project



Go to job configure 🡪 build environment -🡪select the secret ket text add our aws credintialsdintials



Go to build 🡪eecute shell

aws ec2 run-instances \

--image-id ami-085925f297f89fce1 \

--instance-type t2.micro \

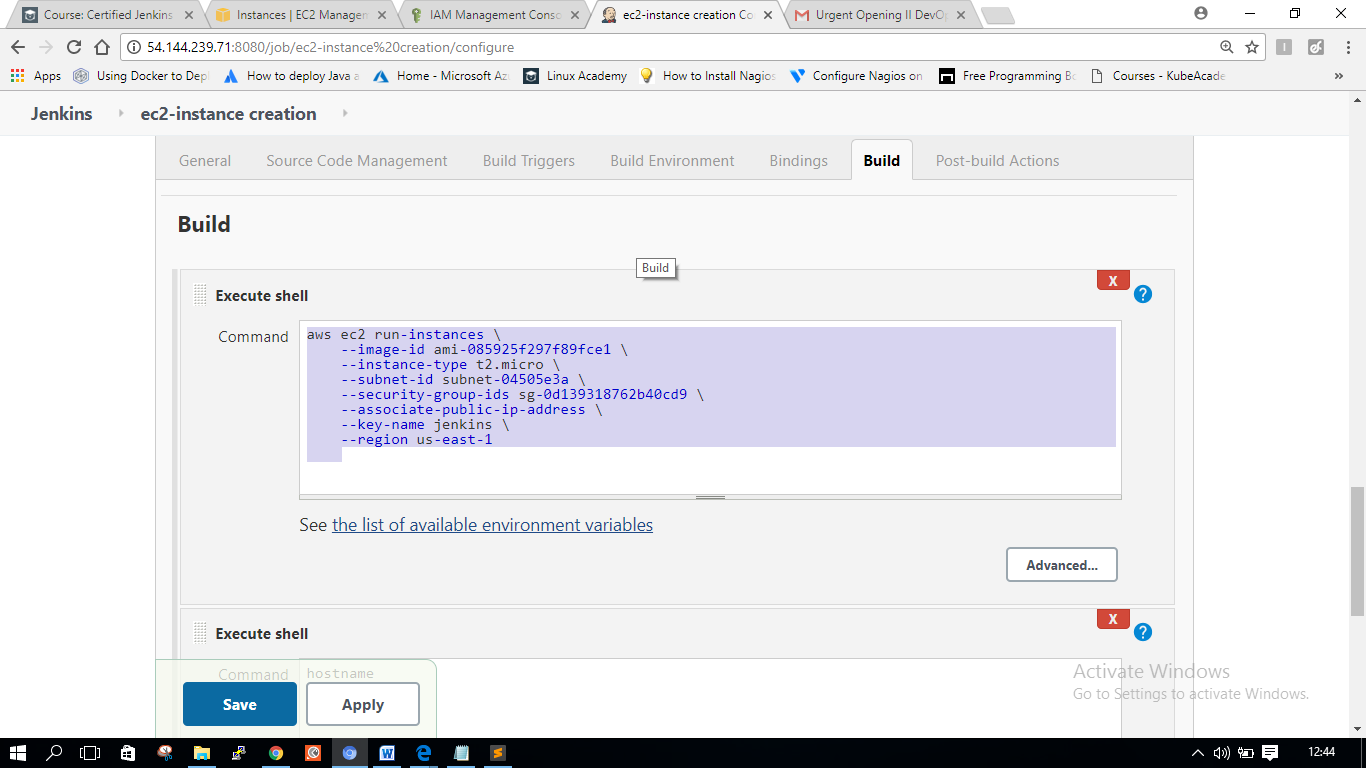
--subnet-id subnet-04505e3a \

--security-group-ids sg-0d139318762b40cd9 \

--associate-public-ip-address \

--key-name jenkins \

--region us-east-1



Sav and build now

You get output

You created the aws instance through Jenkins job

How to Launch a ec2 instance in aws from local jenkins machine

In this example iam running jenkins on my windows laptop

1. Download cloudbees plugin to local jenkins.

2. Store AWS IAM credentials in jenkins at Credentials.

3. Create a project by selecting the aws credentials at

Build Environment > "Use secret text(s) or file(s)" > Bindings > Add > select your KEY ID

4. In Build step add this command

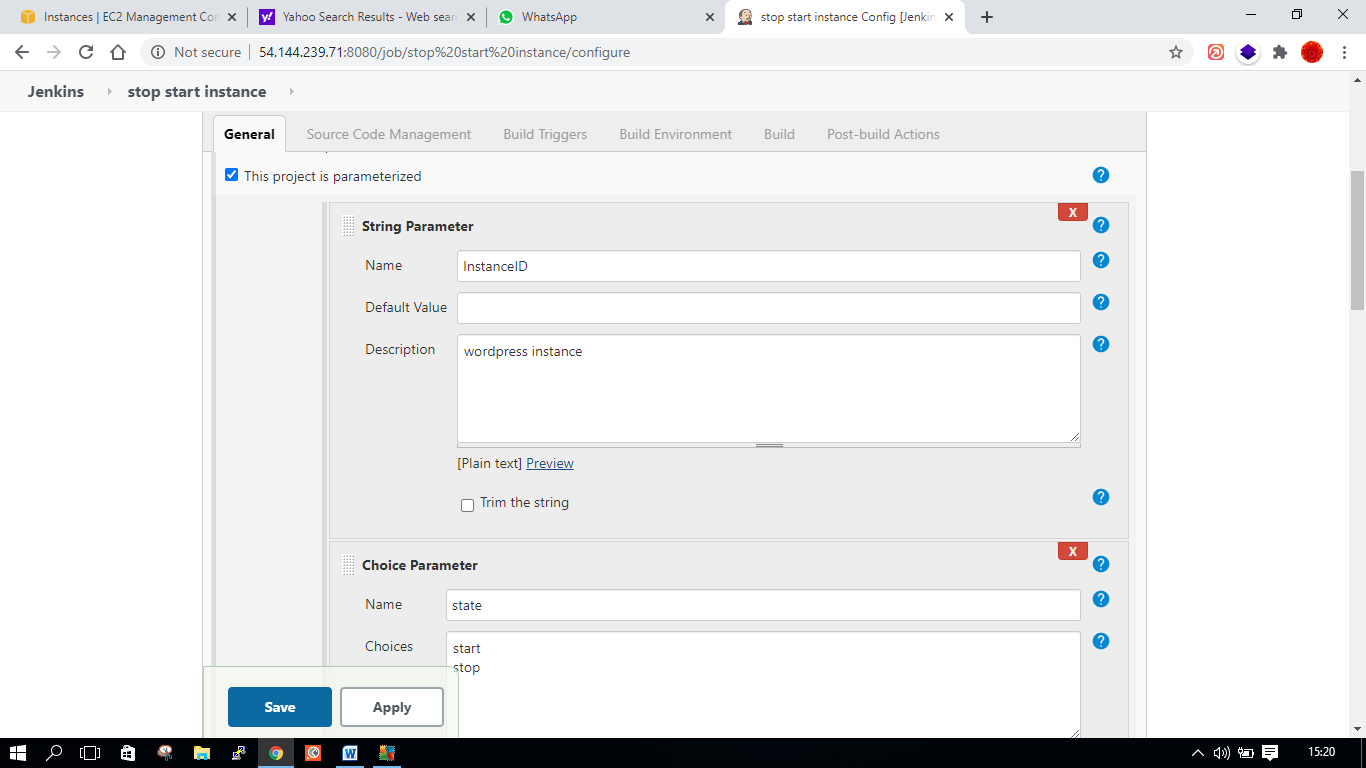
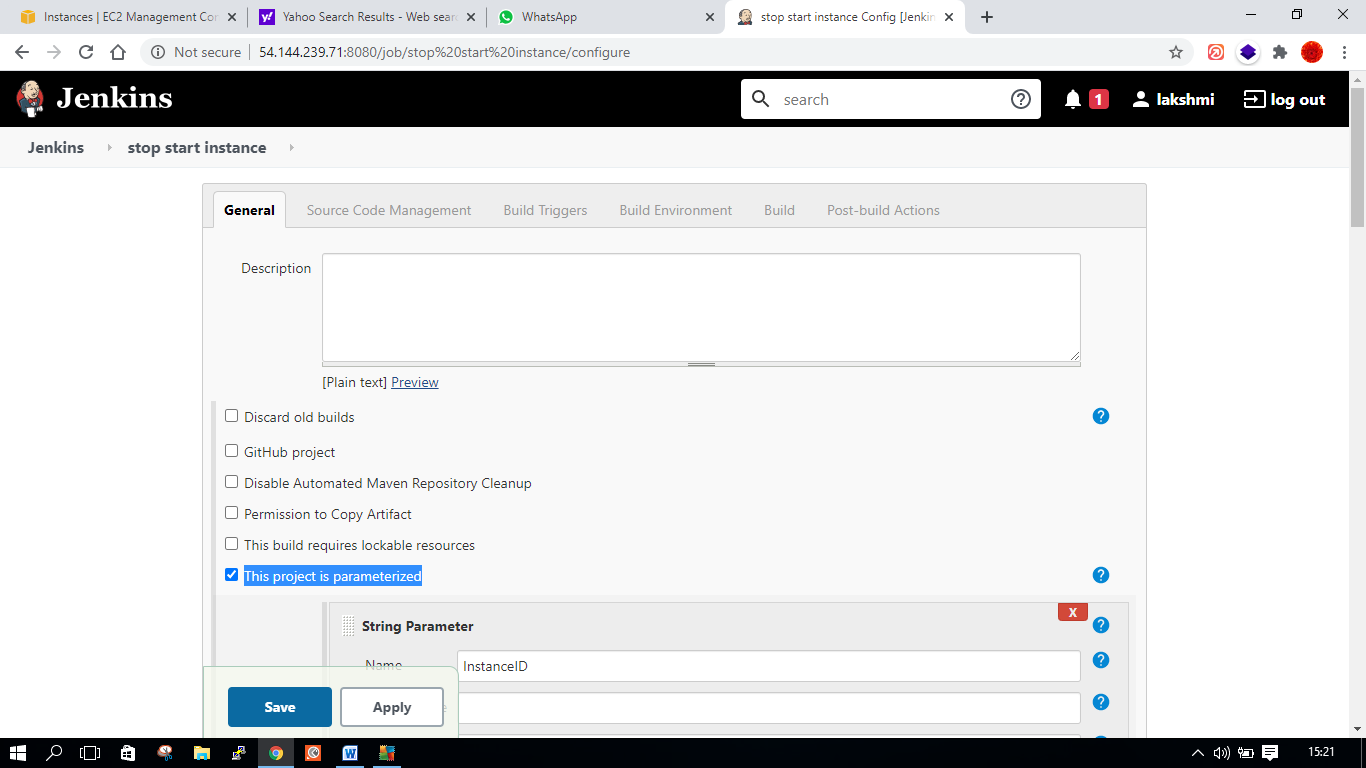
aws ec2 run-instances --image-id ami-0ad42f4f66f6c1cc9 --count 1 --instance-type t2.micro --key-name YourkeyPairName --security-group-ids sg-xxxxxx --subnet-id subnet-xxxxx --region xxxxxxxx

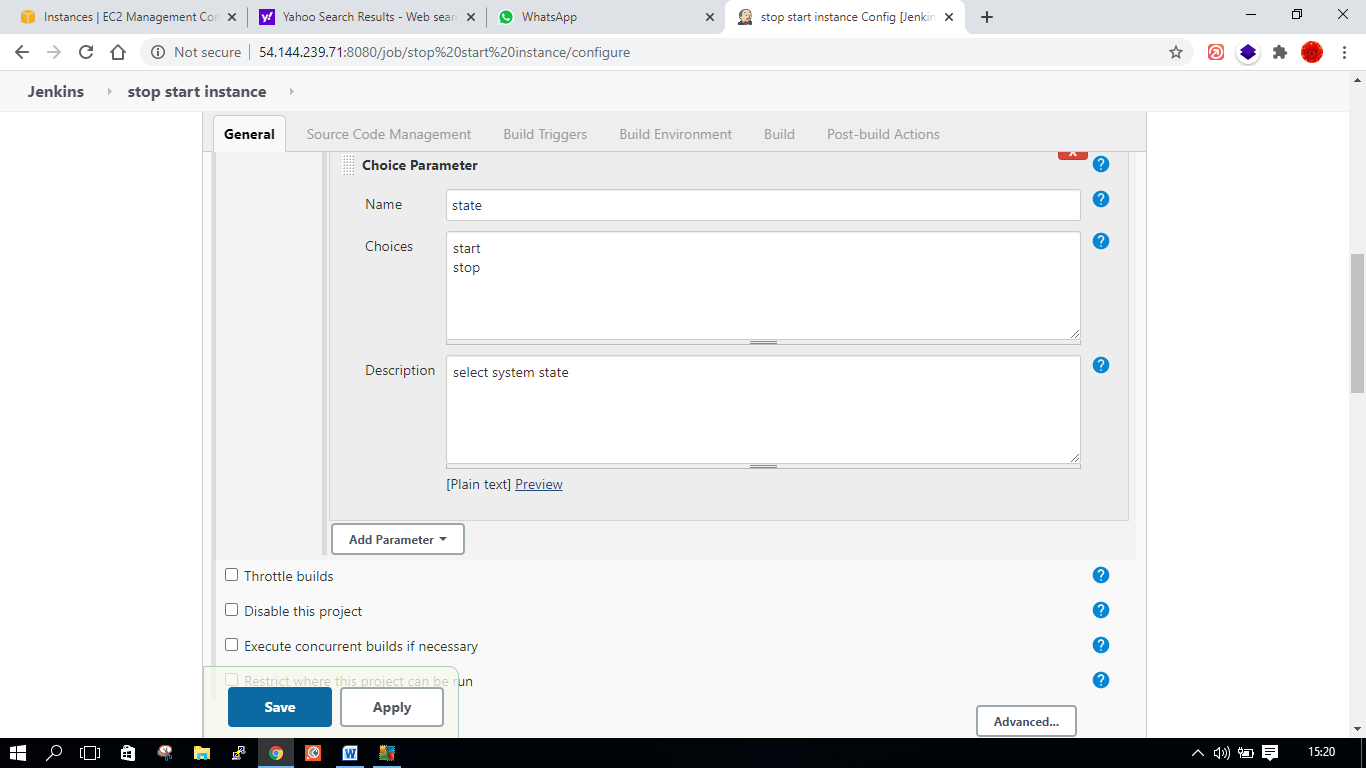
5. Save and Build the Job.

Note: change the parameters in the above command.

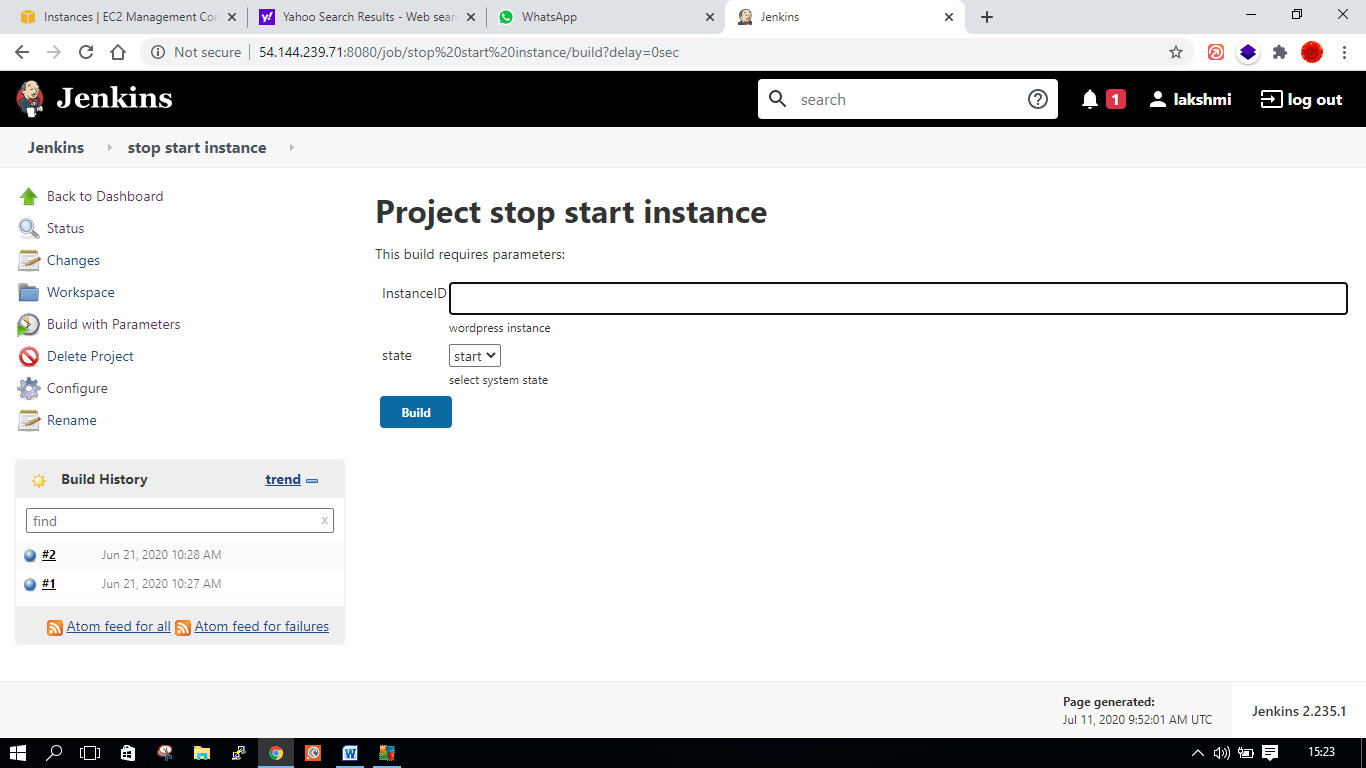
2--🡪same screnario using ec2 plug

**Project stop start instance In Jenkins**





**output like this when your are giving the instance id**



# Building from SCM

## Introduction

In this lab we configure Maven to perform a build. This includes pulling the source code for the build from SCM. At the end of the build process, we also create an artifact for the build.

## Solution

### Configure Maven Installer

**Distributing a Build**

Introduction

In this hands-on lab, we will configure Maven to build a project pulled from SCM — but we will configure a slave node to build the project instead of building the project on the master node.

Solution

Log in to the Jenkins master server using the credentials provided:

ssh cloud\_user@<MASTER\_PUBLIC\_IP\_ADDRESS>

Become root:

sudo su

Configure the Slave Machine for Use with the Jenkins Master

1. Open the /etc/passwd file:

[root@master]$ vim /etc/passwd

1. In the last line in the file (beginning with jenkins), change /bin/false to /bin/bash to allow the jenkins user a shell login.
2. Save and exit the file by pressing **Escape** followed by :x.
3. Change the password for the jenkins user:

[root@master]$ passwd jenkins

1. Enter a password of your choice that you'll easily remember.
2. Switch to jenkins:

[root@master]$ su jenkins

1. Change directory:

[jenkins@master]$ cd ~

1. Generate a public/private RSA key pair:

[jenkins@master]$ ssh-keygen

1. Log in to the slave server:

[jenkins@master]$ ssh cloud\_user@<SLAVE\_PUBLIC\_IP\_ADDRESS>

1. Become root:

[cloud\_user@slave]$ sudo su

1. Create a jenkins user:

[root@slave]$ useradd jenkins

1. Create a password:

[root@slave]$ passwd jenkins

1. Open the sudoers file:

[root@slave]$ visudo

1. In the Defaults section, beneath root, add:

jenkins ALL=(ALL) NOPASSWD: ALL

1. Save and exit the file by pressing **Escape** followed by :x.
2. Exit root:

[root@slave]$ exit

1. See who you're logged in as:

[cloud\_user@slave]$ whoami

You should see you're cloud\_user.

1. Switch to jenkins:

[cloud\_user@slave]$ su jenkins

Enter the password you created.

1. Change directory:

[jenkins@slave]$ cd ~

1. Enter exit *twice* to exit back to the master server.
2. See who you're signed in as:

whoami

You should see you're jenkins.

1. As the jenkins user on the master server, copy the jenkins user's ssh keys to the slave server:

[jenkins@master]$ ssh-copy-id jenkins@<SLAVE\_PUBLIC\_IP\_ADDRESS>

1. Run the following:

cat ./.ssh/id\_rsa

Keep the output listed, as we'll need it for a later step.

Run the Maven Build on the Remote Agent

1. In a new browser tab, navigate to http://<JENKINS\_MASTER\_SERVER\_PUBLIC\_IP>:8080.
2. Log in to Jenkins using the following credentials:
   * *User*: **student**
   * *Password*: **OmgPassword!**
3. Click **Manage Jenkins** in the left-hand menu.
4. Click **Manage Nodes and Clouds**.
5. Click **New Node**.
6. Give it a name of slave1.
7. Select **Permanent Agent**.
8. Click **OK**.
9. For *Remote root directory*, enter /home/jenkins.
10. For *Labels*, enter slave1.
11. For *Host*, enter the slave server's public IP address.
12. Next to *Credentials*, click **Add** > **Jenkins**.
13. Set the following values:
    * *Kind*: **SSH Username with private key**
    * *Username*: **jenkins**
    * *Private Key*: **Enter directly**
      + Copy the *entire* RSA key in the terminal (from dashes to dashes) and paste it into the *Key* window
    * *ID*: **jkey**
    * *Description*: **jenkinsuser**
14. Click **Add**.
15. Set *Credentials* to **jenkins (jenkinsuser)**.
16. Click **Save**.
17. In the upper-left corner, click **Jenkins** > **New Item**.
18. Enter an item name of mavenproject.
19. Select **Freestyle project**.
20. Click **OK**.