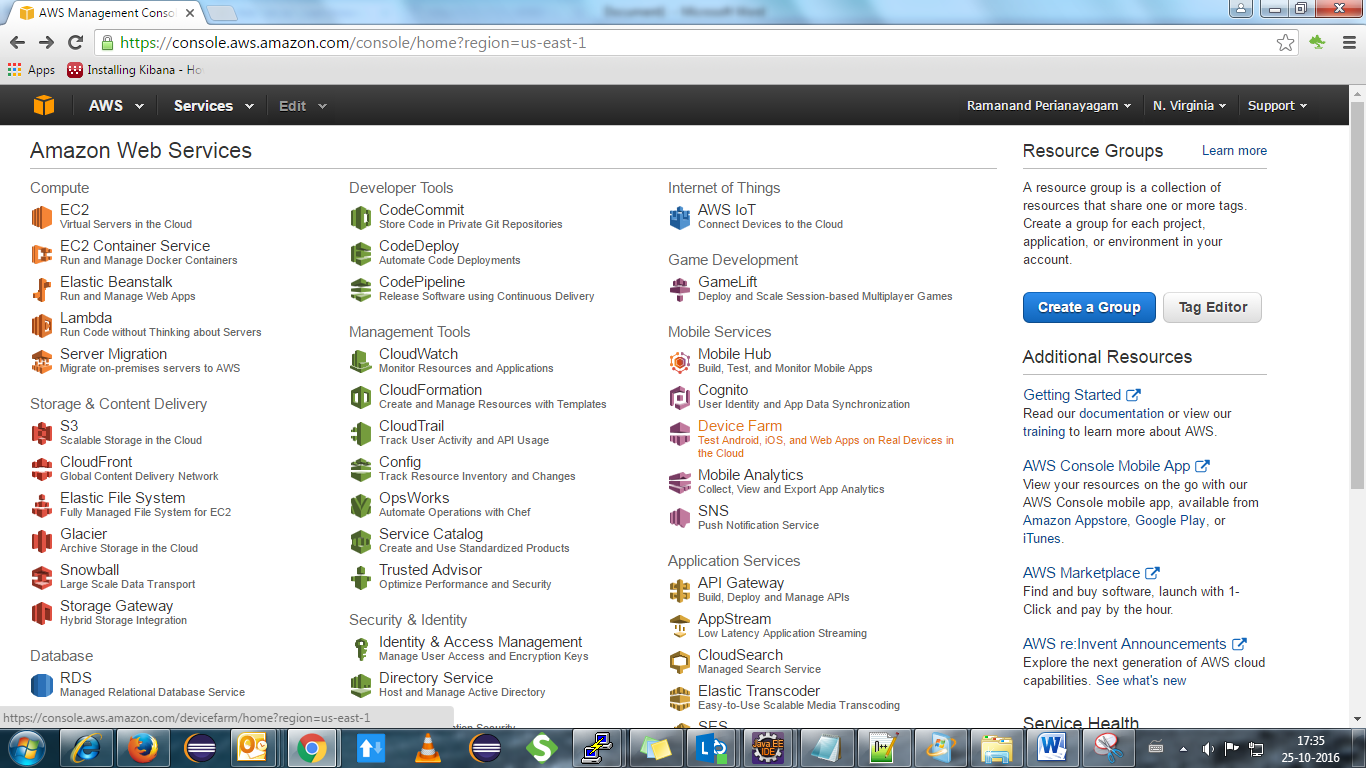
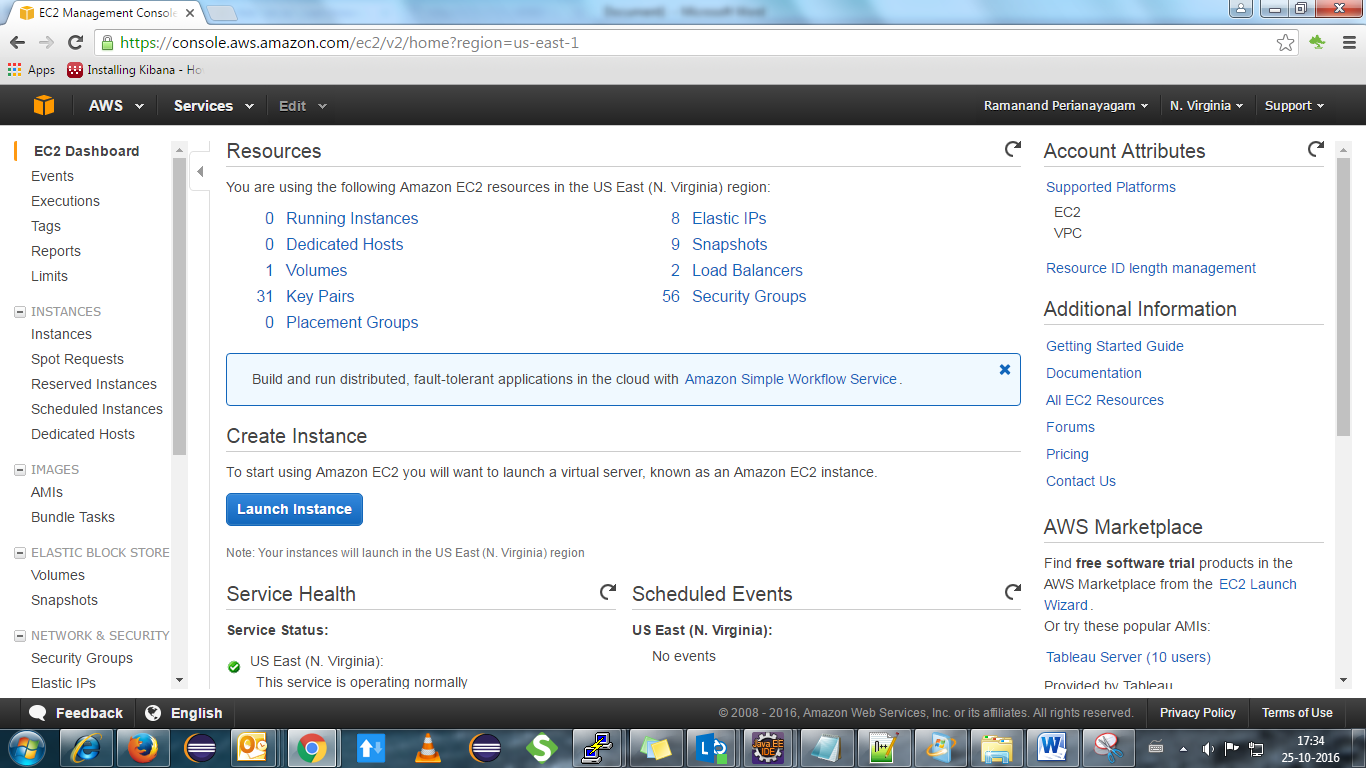
1. Launching EC2 Instance

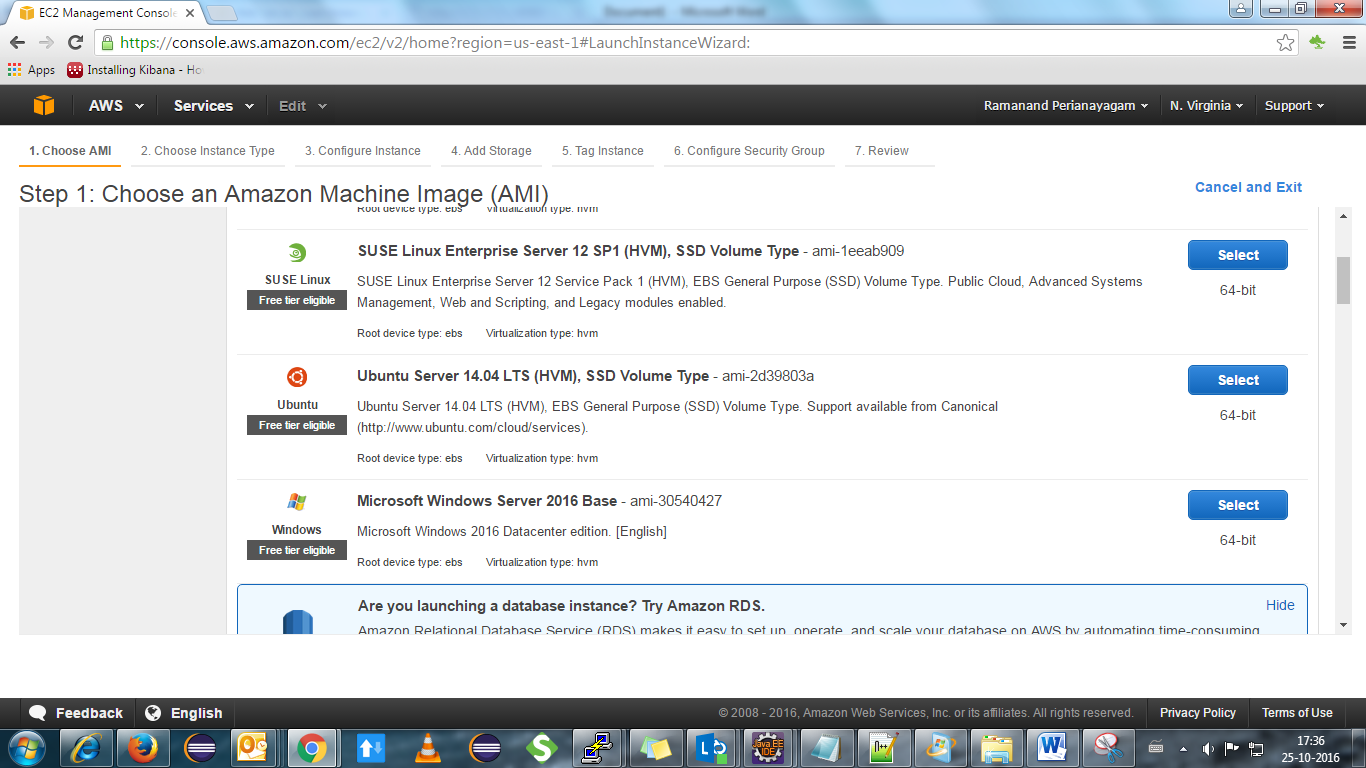
Step 1: Click on **EC2** tab



Step2: Click on blue tab “**Launch Instance**”



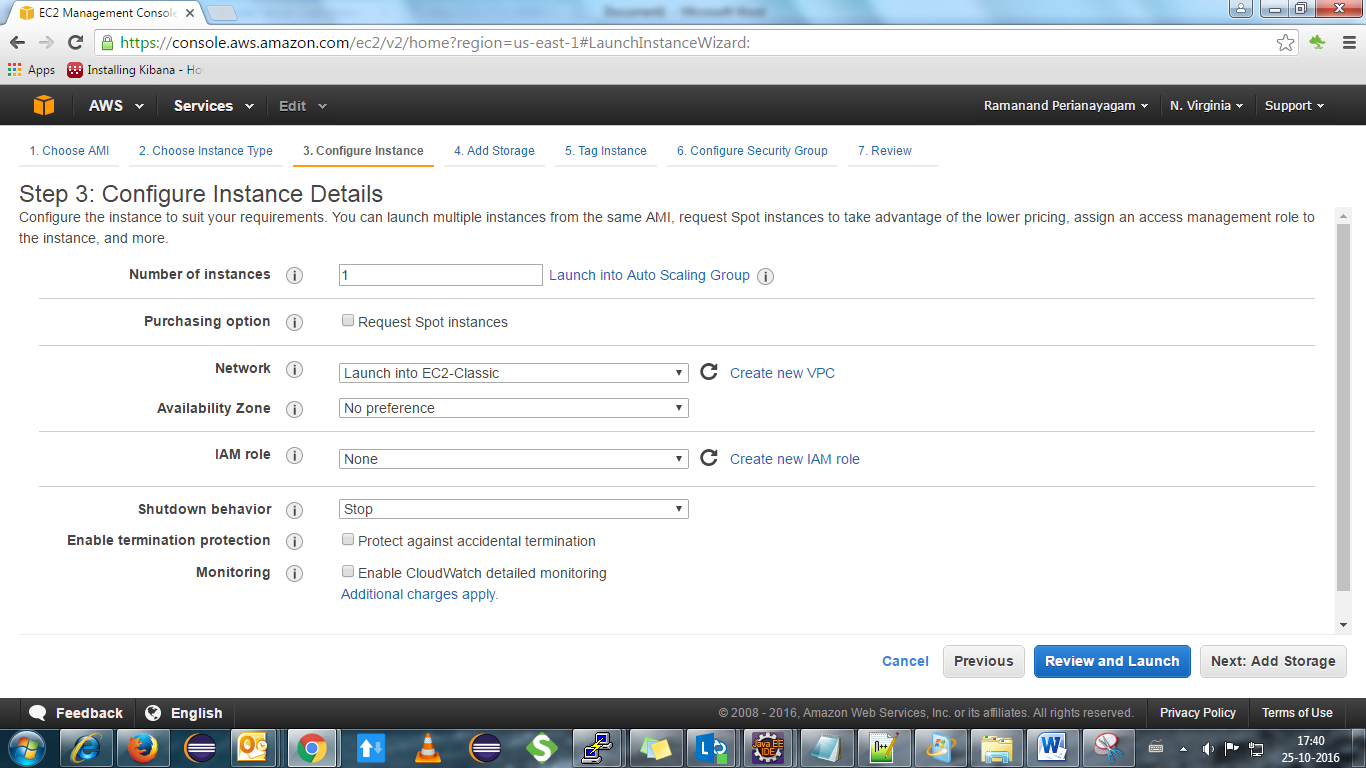
Step3: Select any Ubuntu Machine(eg:**Ubuntu server 14.04**)



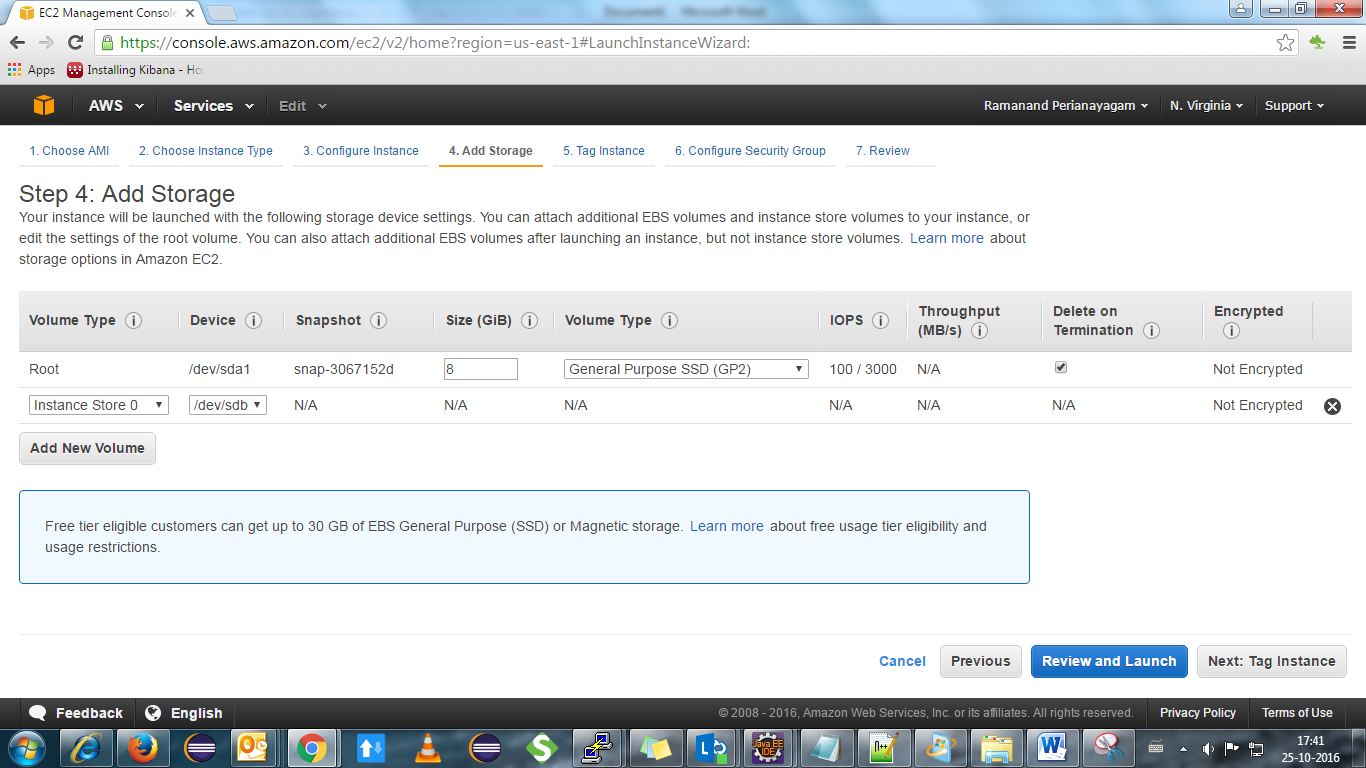
Step 3: Select Size(eg: m3.medium) and click **NEXT**



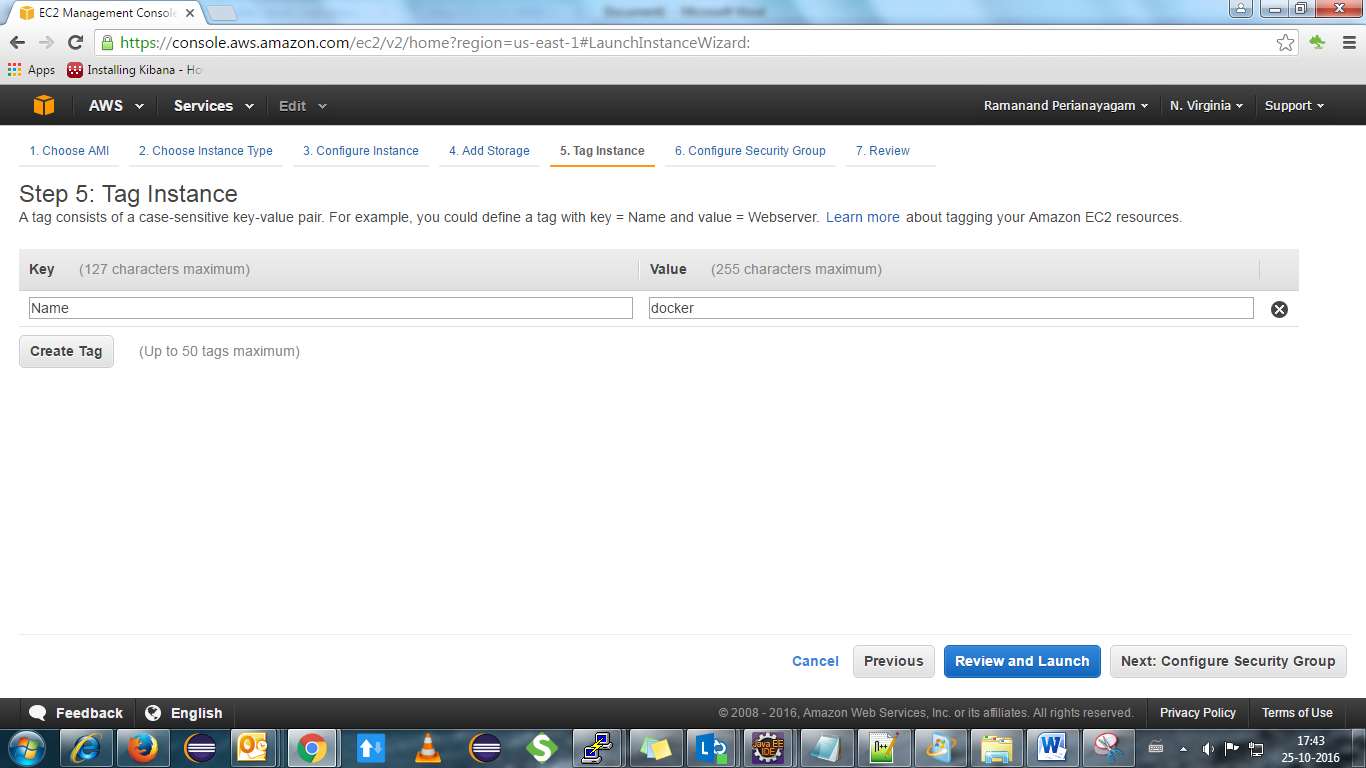
Step 4: Continue with default configuration and click NEXT(**add storage**)



Step 4: Continue with default configuration and click NEXT(**tag instance**)



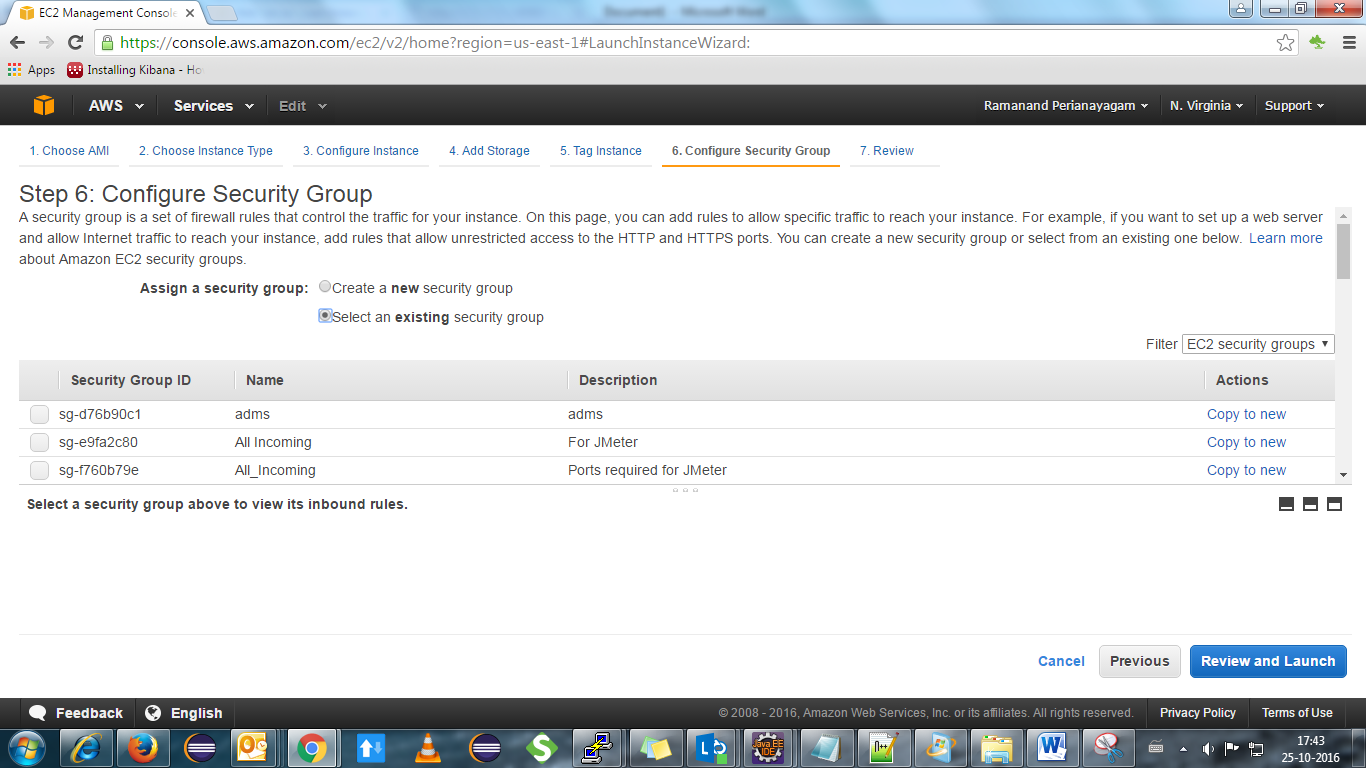
Step 5: Give any name for your instance(eg: docker) and click NEXT(**configure security Group**)



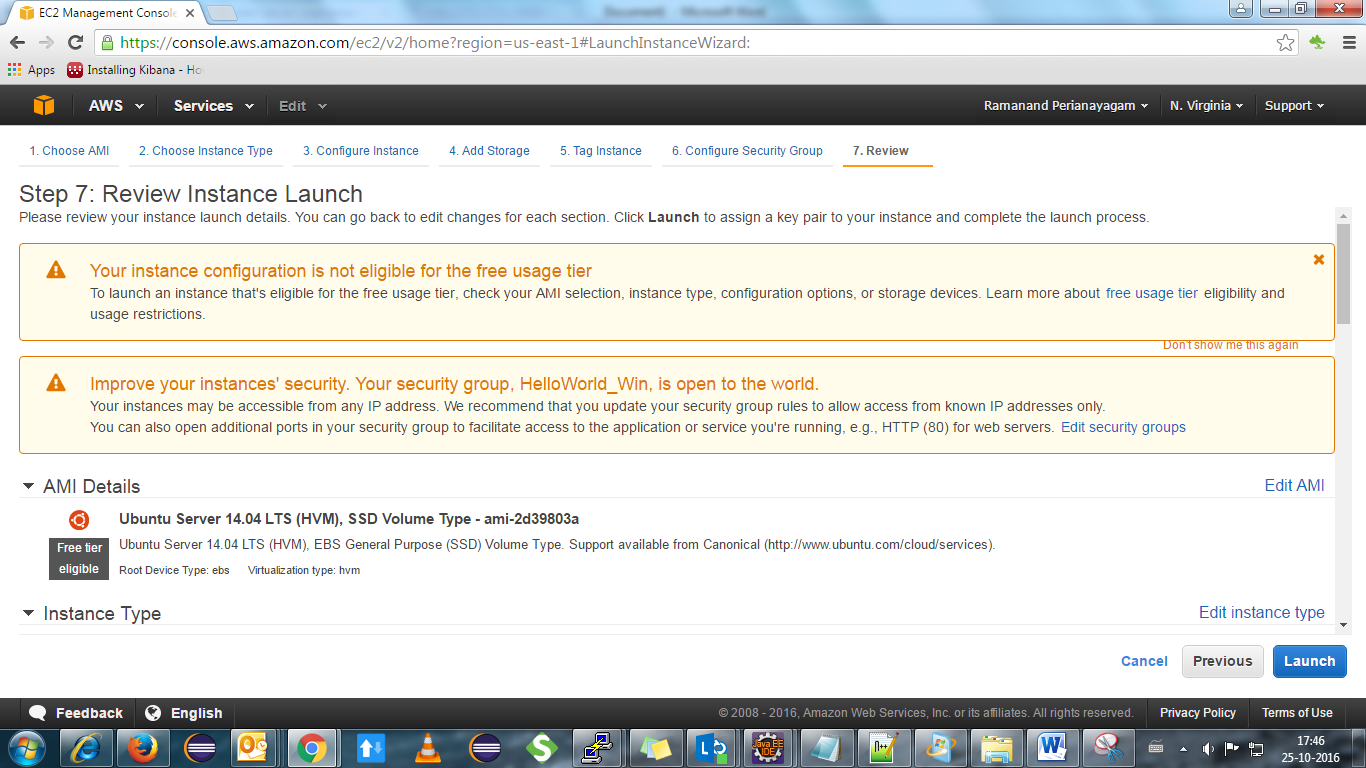
Step6: Click “**Select an existing security group**” and choose “**HelloWorld\_Win**” from the list

(note:our configured security group). Click “**Review and Launch”**

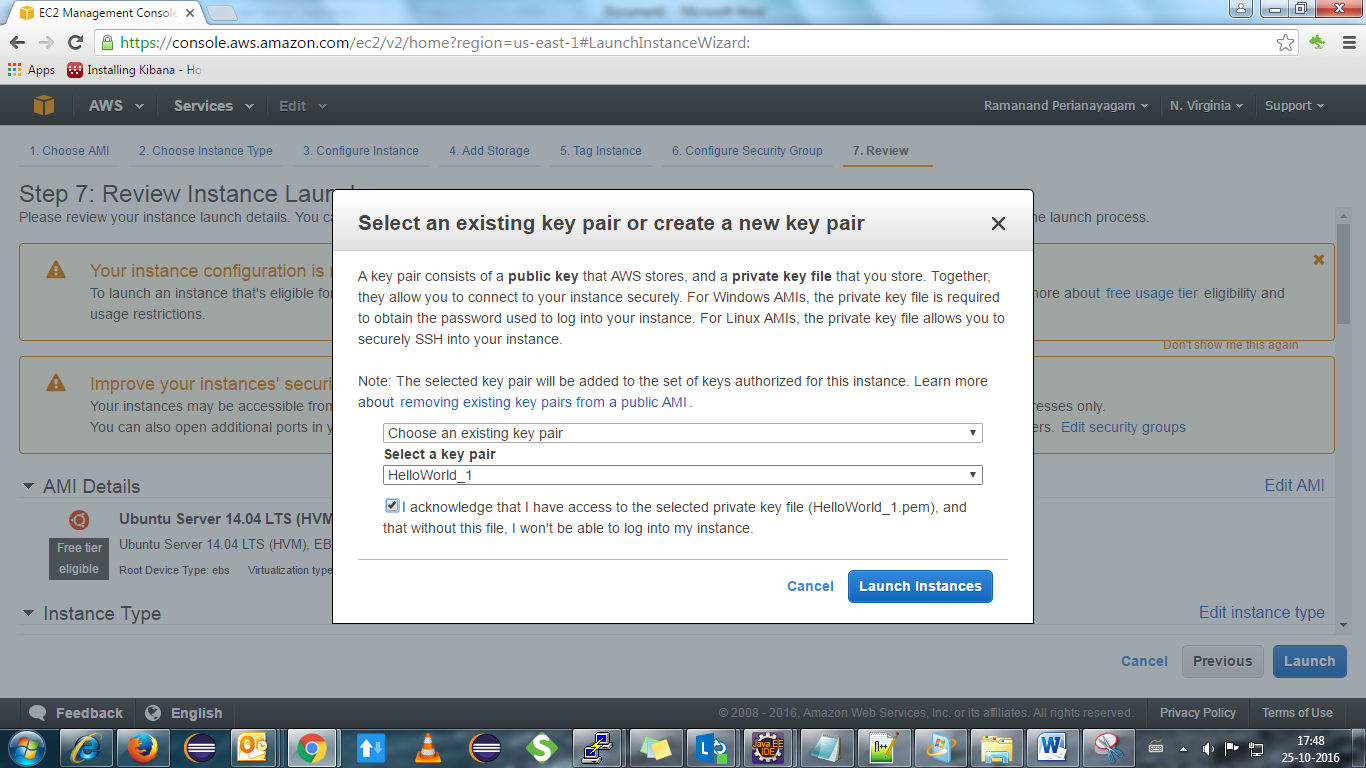
”



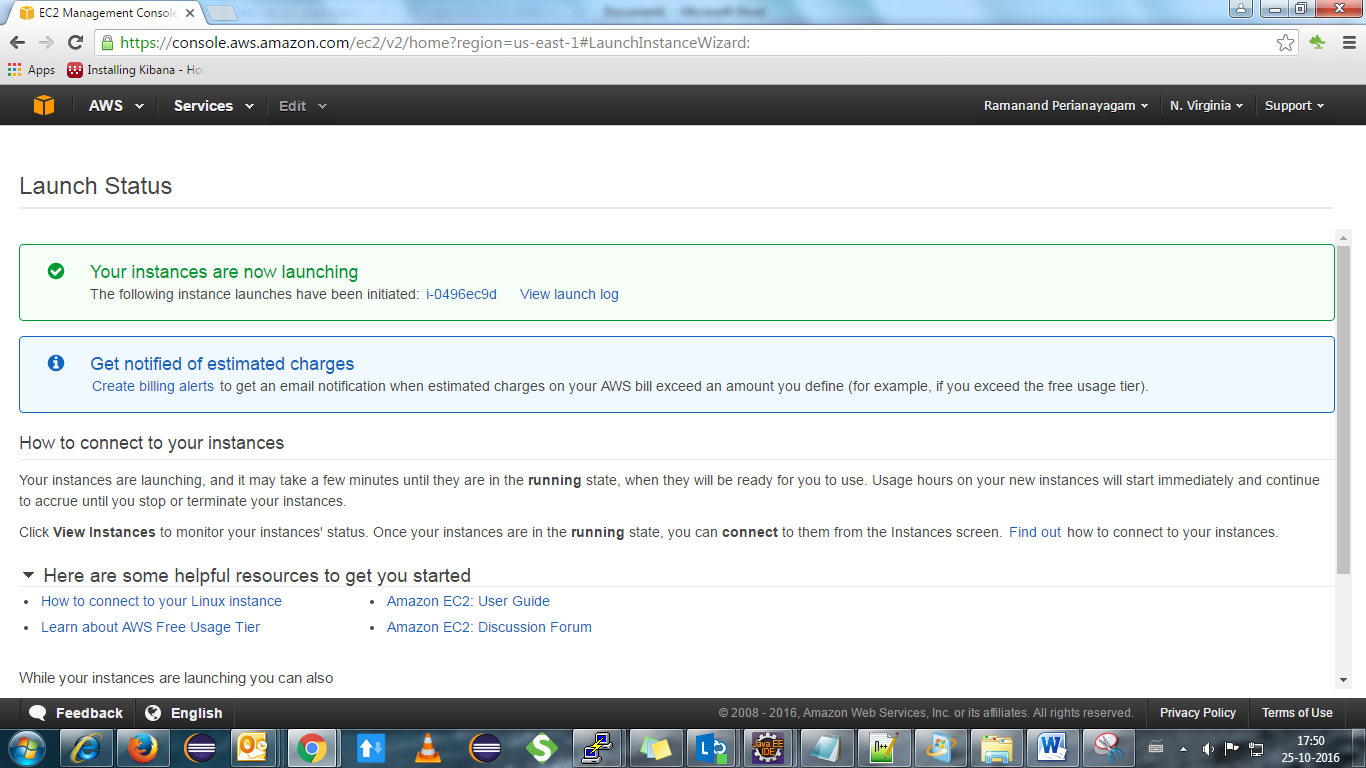
Step7: Click “**Launch**”



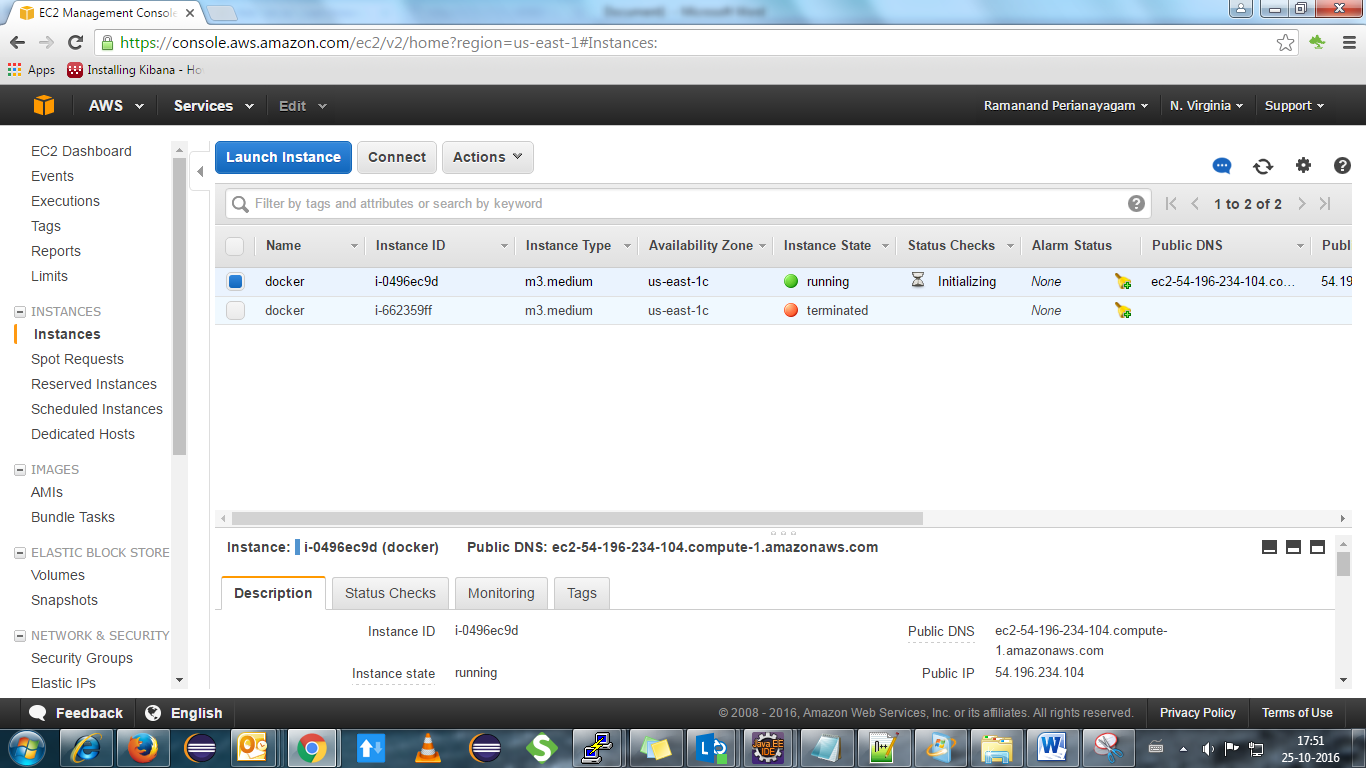
Step8: Choose “**Helloworld\_1**” key pair from the list and Acknowledge the conditions and click “**Launch Instance**”.



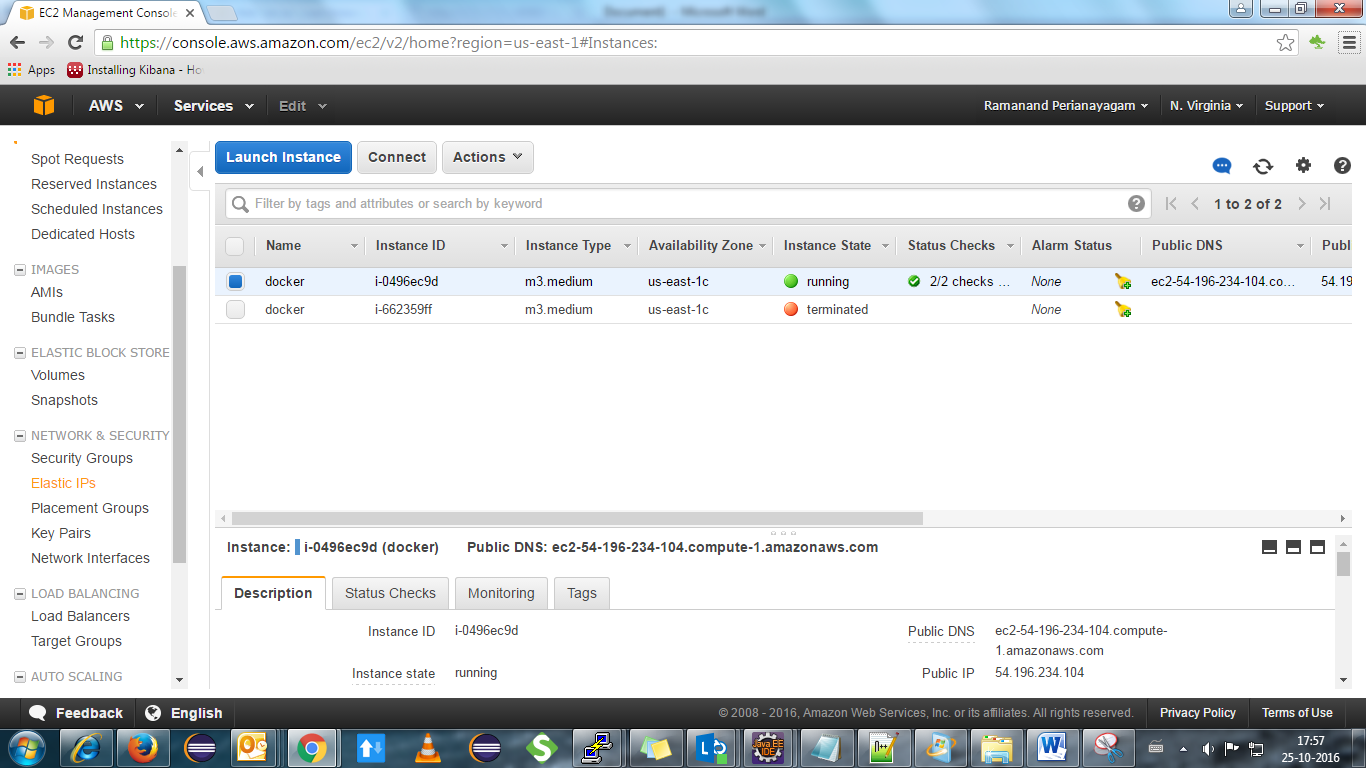
Step9:The instance status is shown and click “**View Instance**”



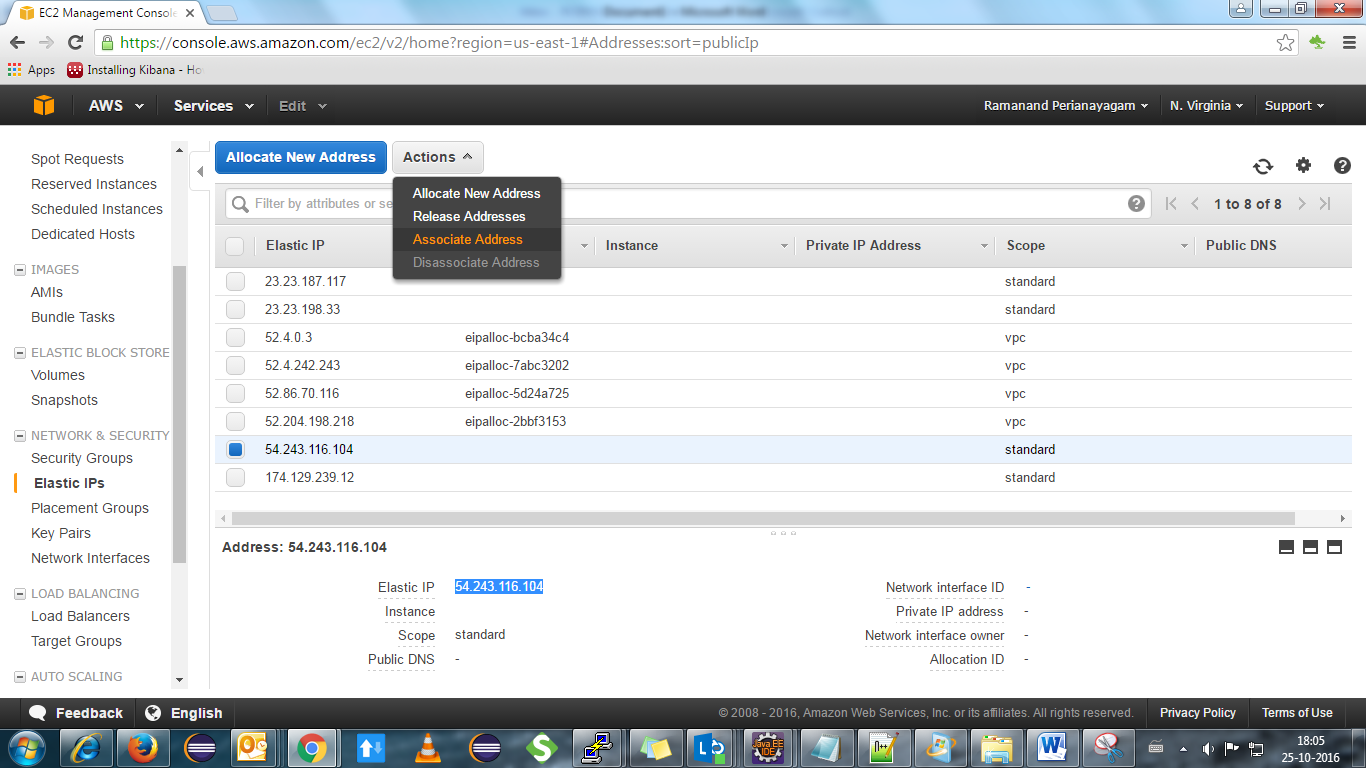
Step10: The Instance you launched is shown in the list



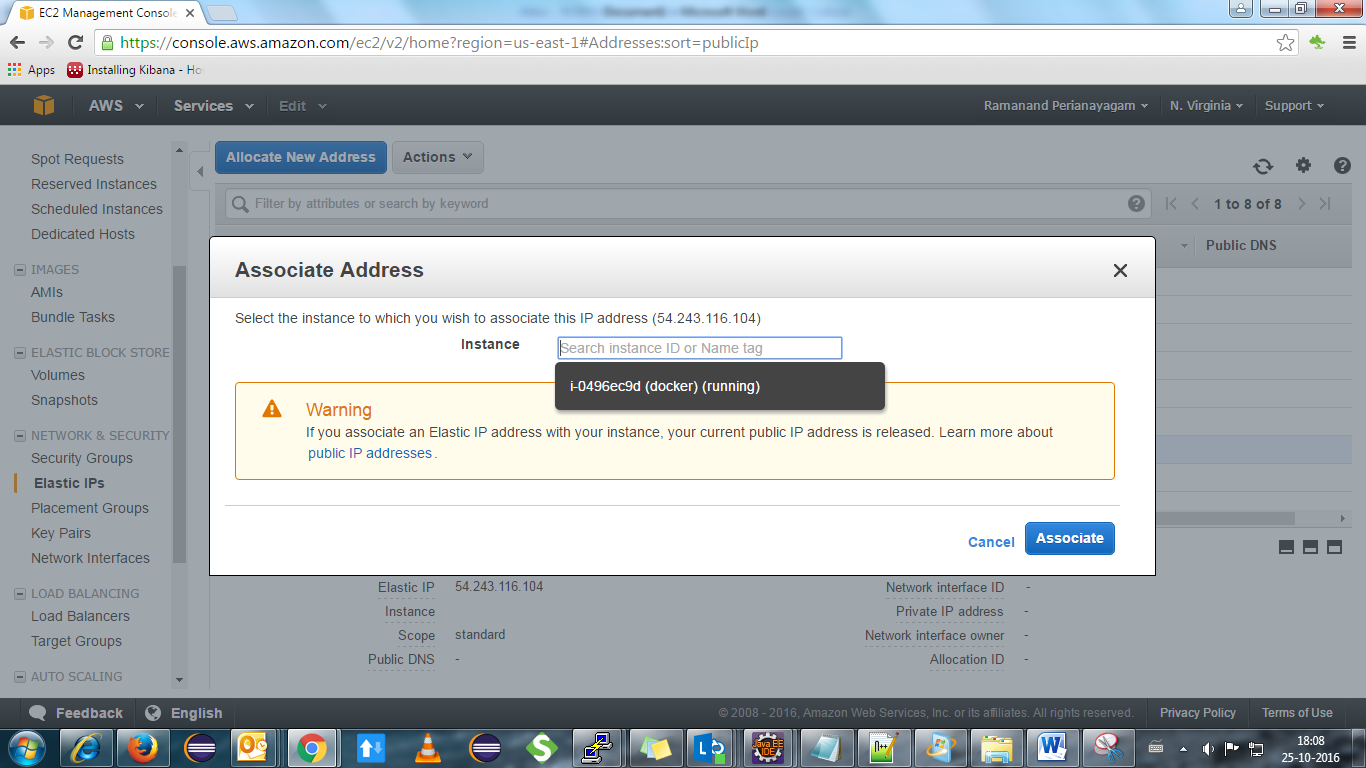
Step11:Assign an Elastic Ip to your instance(note:because of IP conflict). Click on “**Elastic Ip**” tab



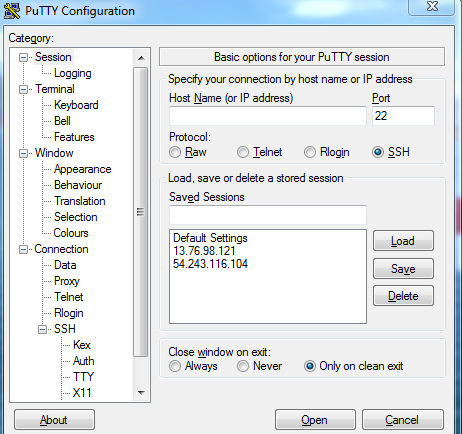
Step12: Choose “**54.243.116.104**” from the list(note: no ip conflict) and choose “**Associate Address**” from “Actions” tab



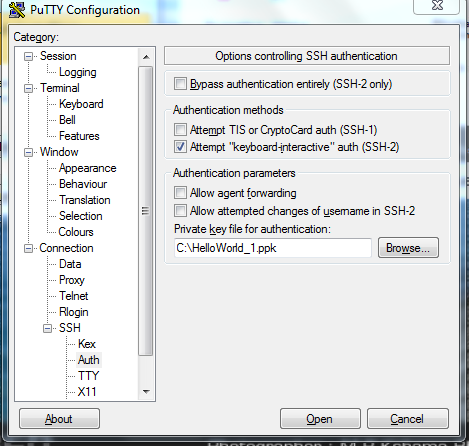
Step13: The running instance will be shown when clicked, choose your instance and click “**Associte**”.



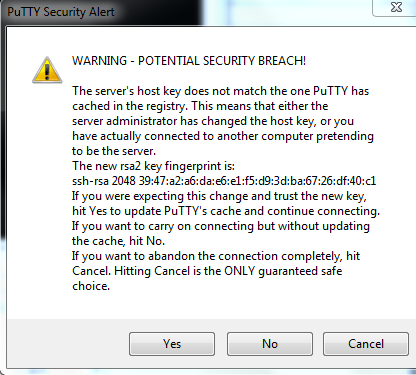
Step14: Now you can launch your Instance through PUTTY. Open putty(application)



Provide your **Elastic IP,** you associated in Host\_Name and under SSH, choose **Auth** and browse the Key-Pair(**HelloWorld.ppk**) and click “Open”



A warning prompts, choose “**Yes**” and login as: **Ubuntu**



**Now you can work on your Linux Machine**

**Installing Docker ,Docker-Compose and HAProxy**

Step 1: Login to your Ubuntu Machine and execute the following commands.

Update package information, ensure that APT works with the https method, and that CA certificates are installed.

1. **sudo apt-get update**
2. **sudo apt-get install apt-transport-https ca-certificates**

Add the new GPG key.

1. **sudo apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 --recv-keys 58118E89F3A912897C070ADBF76221572C52609D**

**echo "<REPO>" | sudo tee /etc/apt/sources.list.d/docker.list**

**(replace REPO with “deb https://apt.dockerproject.org/repo ubuntu-trusty main”)**

Update the APT package index.

1. **sudo apt-get update**

Verify that APT is pulling from the right repository.

1. **apt-cache policy docker-engine**

A similar output comes

$ apt-cache policy docker-engine

docker-engine:

Installed: 1.12.2-0~trusty

Candidate: 1.12.2-0~trusty

Version table:

\*\*\* 1.12.2-0~trusty 0

500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages

100 /var/lib/dpkg/status

1.12.1-0~trusty 0

500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages

1.12.0-0~trusty 0

500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages

1.11.2-0~trusty 0

500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages

1.11.1-0~trusty 0

500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages

Update your APT package index.

g)**sudo apt-get update**

Install Docker.

h)**sudo apt-get install docker-engine**

Start the docker daemon

**i)sudo service docker start**

Verify docker is installed correctly.

j)**sudo docker run hello-world**

**Installing Docker Compose**

**Run the following Commands**

First, install python-pip as prerequisite:

a)**sudo apt-get -y install python-pip**

Then you can install Docker Compose:

b)**sudo pip install docker-compose**

## Running a Container with Docker Compose

First, create a directory for our YAML file:

c)**mkdir hello-world**

Then change into the directory:

d)**cd hello-world**

Now create the YAML file using your favorite text editor

e)**vi docker-compose.yml**

Put the following commands there

**weba:**

**build: .**

**expose:**

**- 80**

**webb:**

**build: .**

**expose:**

**- 80**

**webc:**

**build: .**

**expose:**

**- 80**

**haproxy:**

**image: hypriot/rpi-haproxy**

**volumes:**

**- haproxy:/haproxy-override**

**links:**

**- weba**

**- webb**

**- webc**

**ports:**

**- "80:80"**

**- "70:70"**

**expose:**

**- "80"**

**- "70"**

**weba:**

**image:”your docker image”**

**webb:**

**image:”your docker image”**

**webc:**

**image:”your docker image”**

While still in the ~/hello-world directory, execute the following command to create the container:

f)**docker-compose up**

Similar Output comes

$ docker-compose up -d

Recreating rpinodehaproxyexample\_webb\_1...

Recreating rpinodehaproxyexample\_webc\_1...

Recreating rpinodehaproxyexample\_weba\_1...

Recreating rpinodehaproxyexample\_haproxy\_1...

**Install HAProxy**

**Run the commands**

1. ***apt-get install haproxy***

You can check the version by:

1. ***haproxy –v***

***Go to directory and edit haproxy.cnfg***

*“/etc/default/haproxy*.” And set *ENABLED=1*

To verify if this change is done properly, execute the init script of HAProxy without any parameters. You should see the following:

1. ***service haproxy <press\_tab\_key>  
   reload   restart  start    status   stop***

#### Configuring HAProxy

Backup the original file by renaming it:

*d)****mv /etc/haproxy/haproxy.cfg{,.original}***

We'll create our own *haproxy.cfg* file. Using your favorite text editor create the */etc/haproxy/haproxy.cfg* file as:

global

log 127.0.0.1 local0

log 127.0.0.1 local1 notice

defaults

log global

mode http

option httplog

option dontlognull

timeout connect 5000

timeout client 10000

timeout server 10000

listen stats :70

stats enable

stats uri /

frontend balancer

bind 0.0.0.0:80

mode http

default\_backend aj\_backends

backend aj\_backends

mode http

option forwardfor

# http-request set-header X-Forwarded-Port %[dst\_port]

balance roundrobin

server weba weba:80 check

server webb webb:80 check

server webc webc:80 check

# option httpchk OPTIONS \* HTTP/1.1\r\nHost:\ localhost

option httpchk GET /

http-check expect status 200

Save this and run this command

e)**docker-compose up –d**

Similar Output Comes

$ docker-compose up -d

Recreating rpinodehaproxyexample\_webb\_1...

Recreating rpinodehaproxyexample\_webc\_1...

Recreating rpinodehaproxyexample\_weba\_1...

Recreating rpinodehaproxyexample\_haproxy\_1...