**6. Deploying a Node.js Web Application on AWS**

**HARDWARE REQUIREMENTS**: Core I5 Processor, 4 GB RAM, 40GB HDD

**SOFTWARE REQUIREMENTS**: Amazon AWS, EC2, VS Code/Eclipse, Node, NPM, GIT, Putty, PuttyGen

**Description:**

Node.js is a JavaScript runtime environment that allows one to run JS on the server. It is built on the open-source V8 JavaScript engine used in Chrome and written in C++ which executes JS in a standalone environment. In this experiment, we clone a Nodejs application from GITHUB and deploy this application on to Amazon EC2 instance, make it available over Amazon AWS URI.

**Steps to configure Ec2 Instance :**

**1. Create Ec2 instance.**

Choose amazon Ec2 instance machine image as Ubuntu 18.04 64 bit with type of micro.

Next configure instance details.

Next Add storage

Next configure Security Group.Give security group name is **Lotus Airline**

In this step we need to allow http and https requests to access from any group.

Finally Review and Launch instance.

**2. Launch an EC2 instance:**

When it is launched you will be asked to download key pair for one time store it carefully. you can choose an existing key pair or a new key pair.

When the instance state is running means that your instance was created successfully.

Copy the public DNS of your Instance. You can access different app running on your instance at a different port.

**3. Connect to your Instance:**

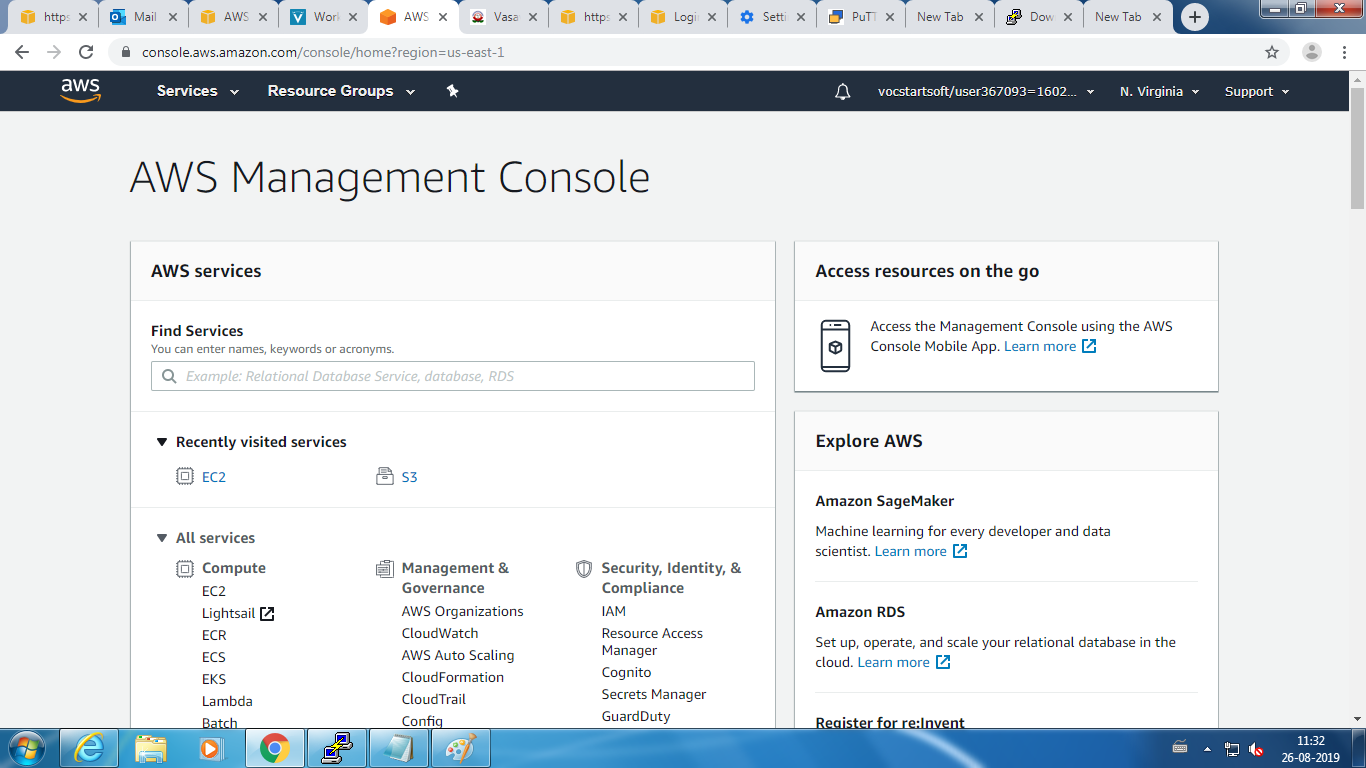
Click on launch instance then it shows popup window giving details how to connect to your instance.

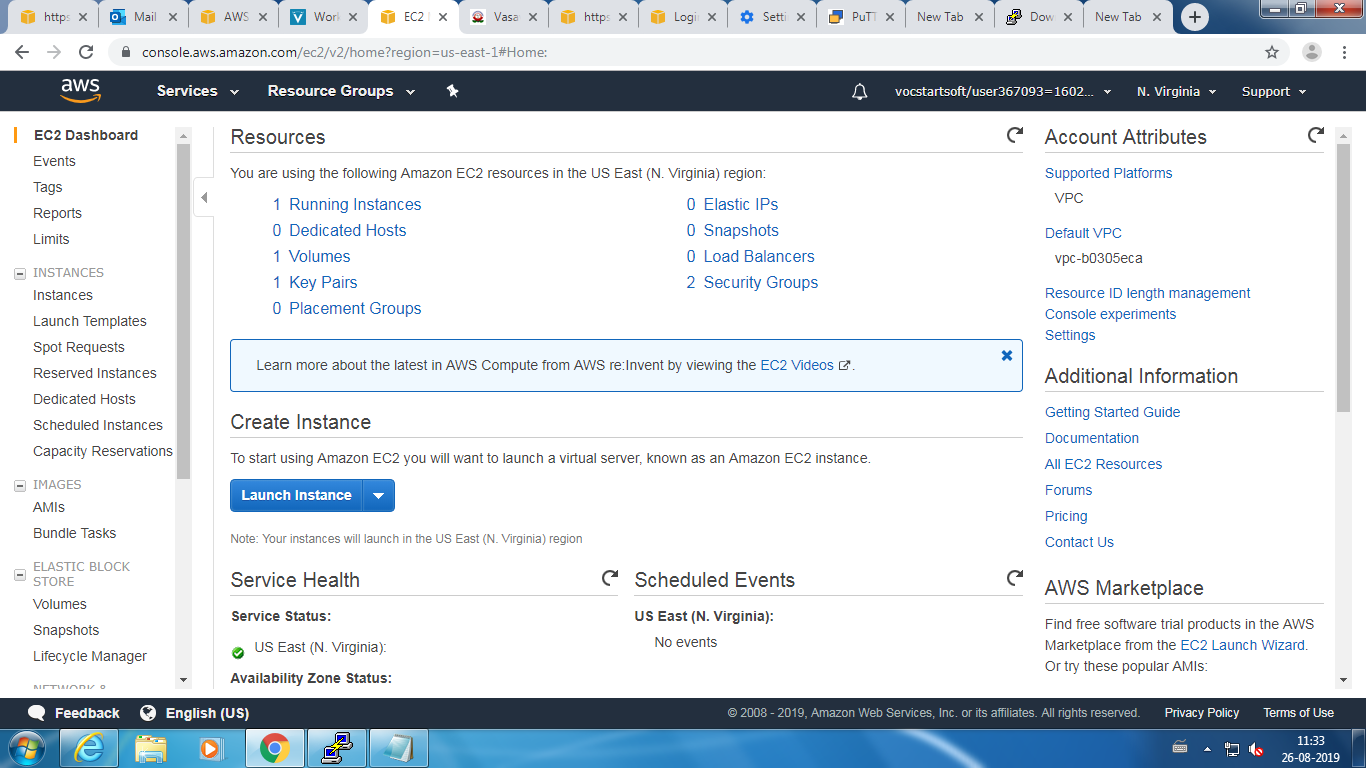
To open SSH client and lf we are in windows platform we need to launch the instance with the help of putty soft.

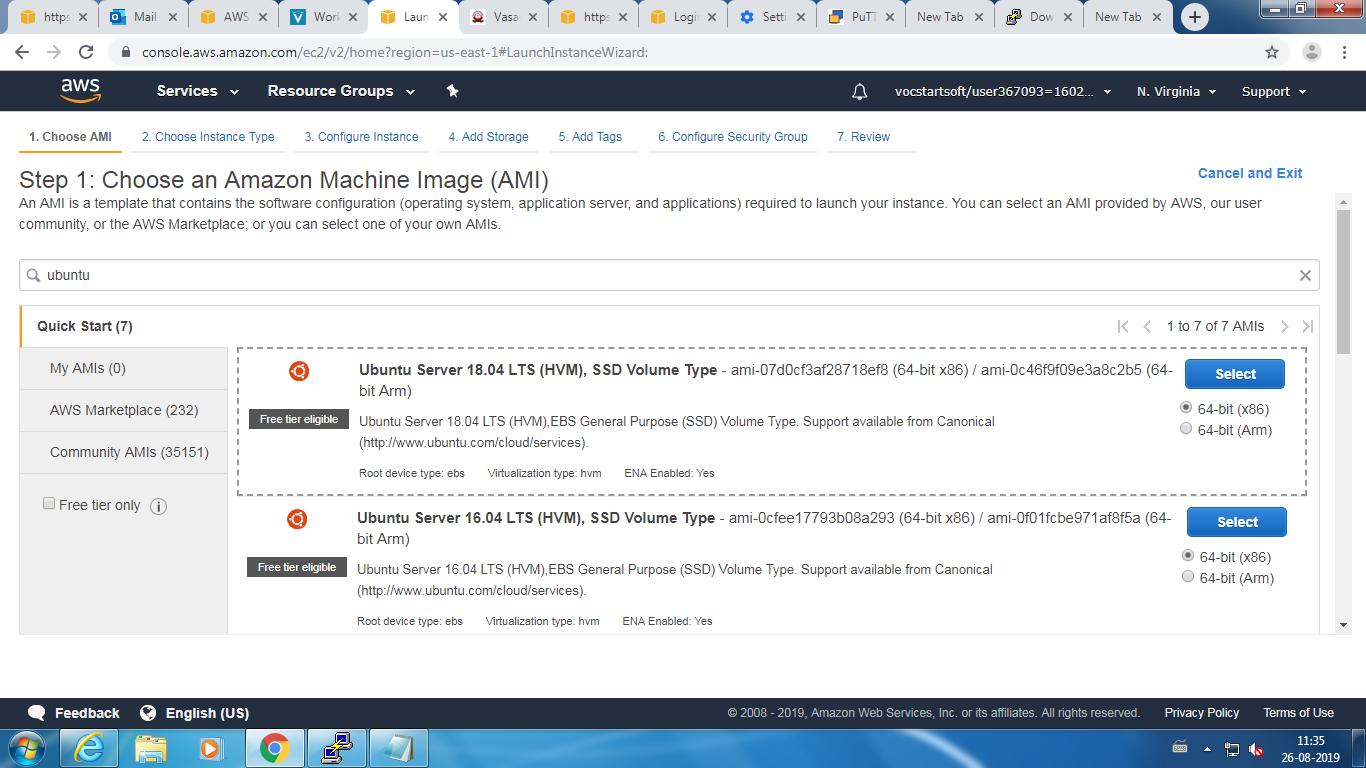
Once it is opened login as ubuntu user.

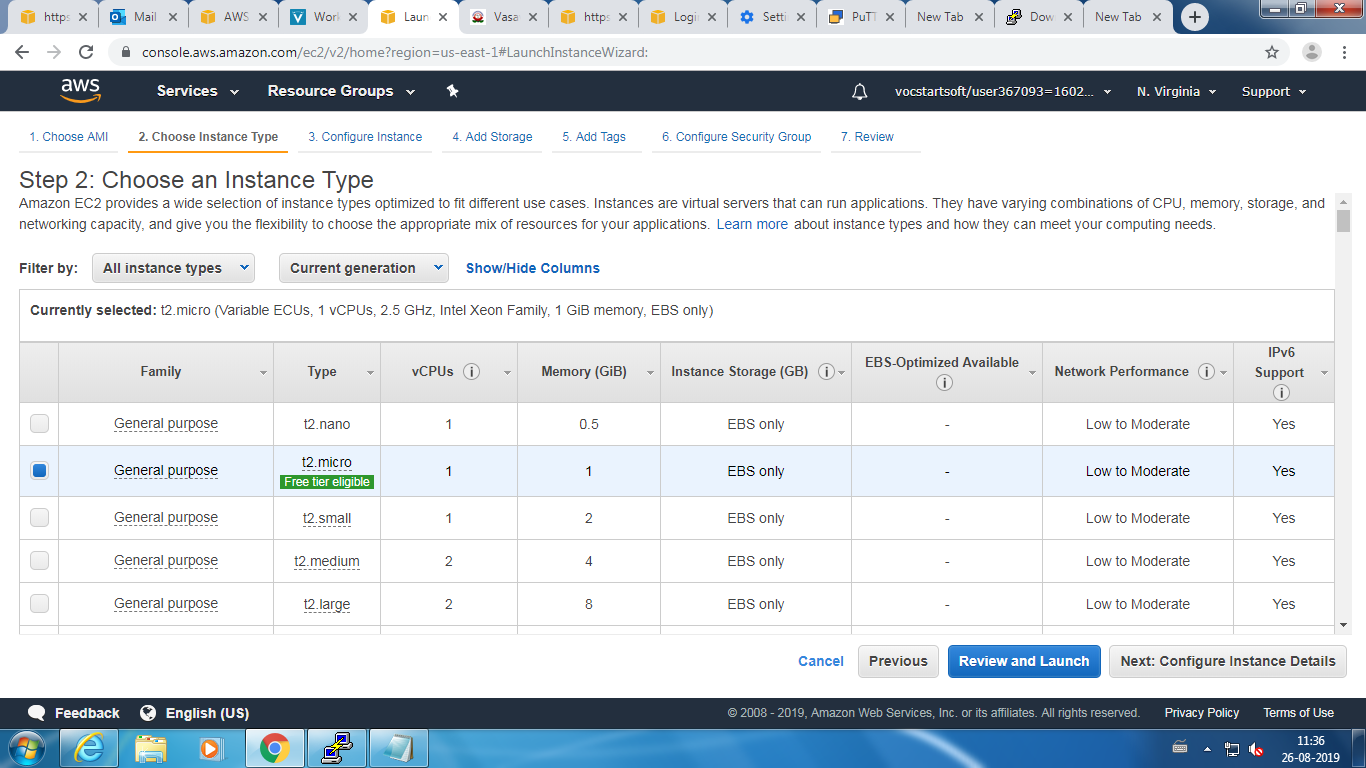
Finally it will take you to Amazon EC2 ubuntu instance SSH window.

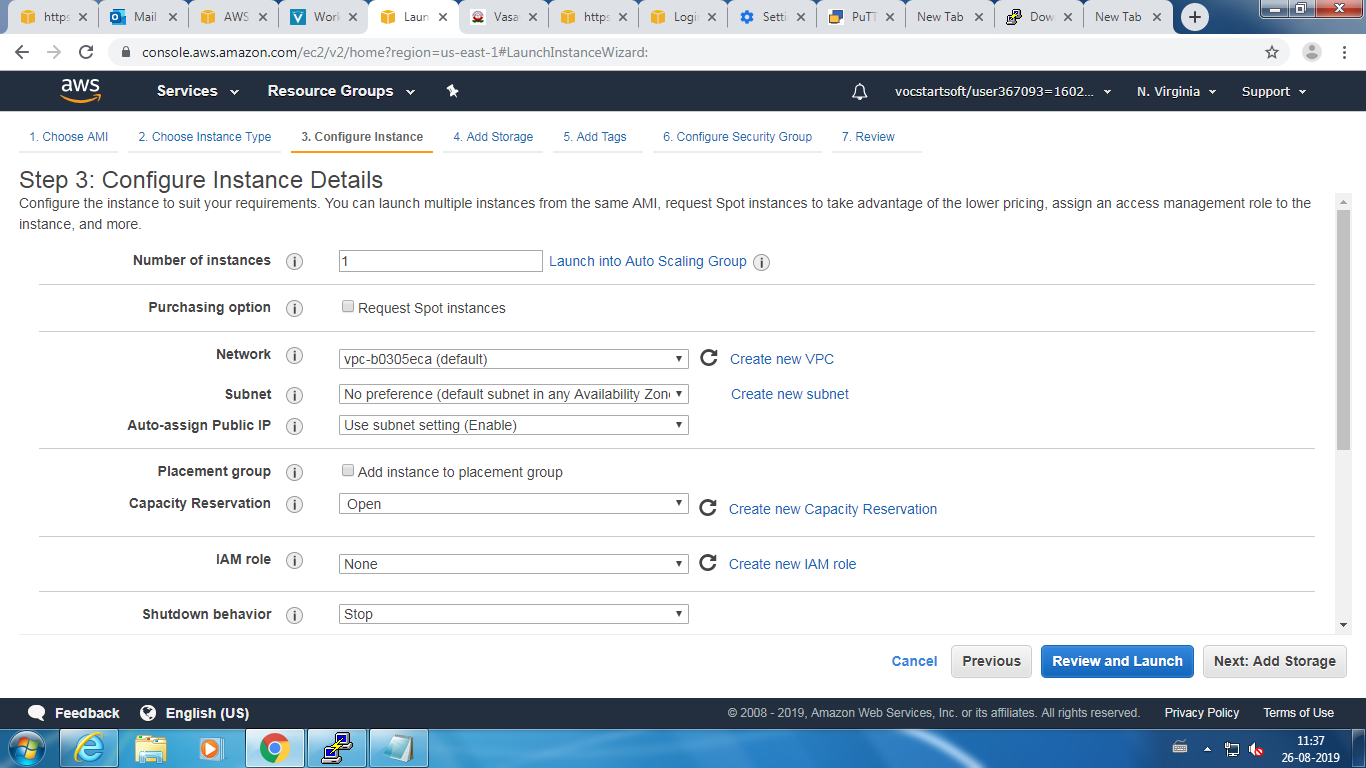
Below are the screen shots.

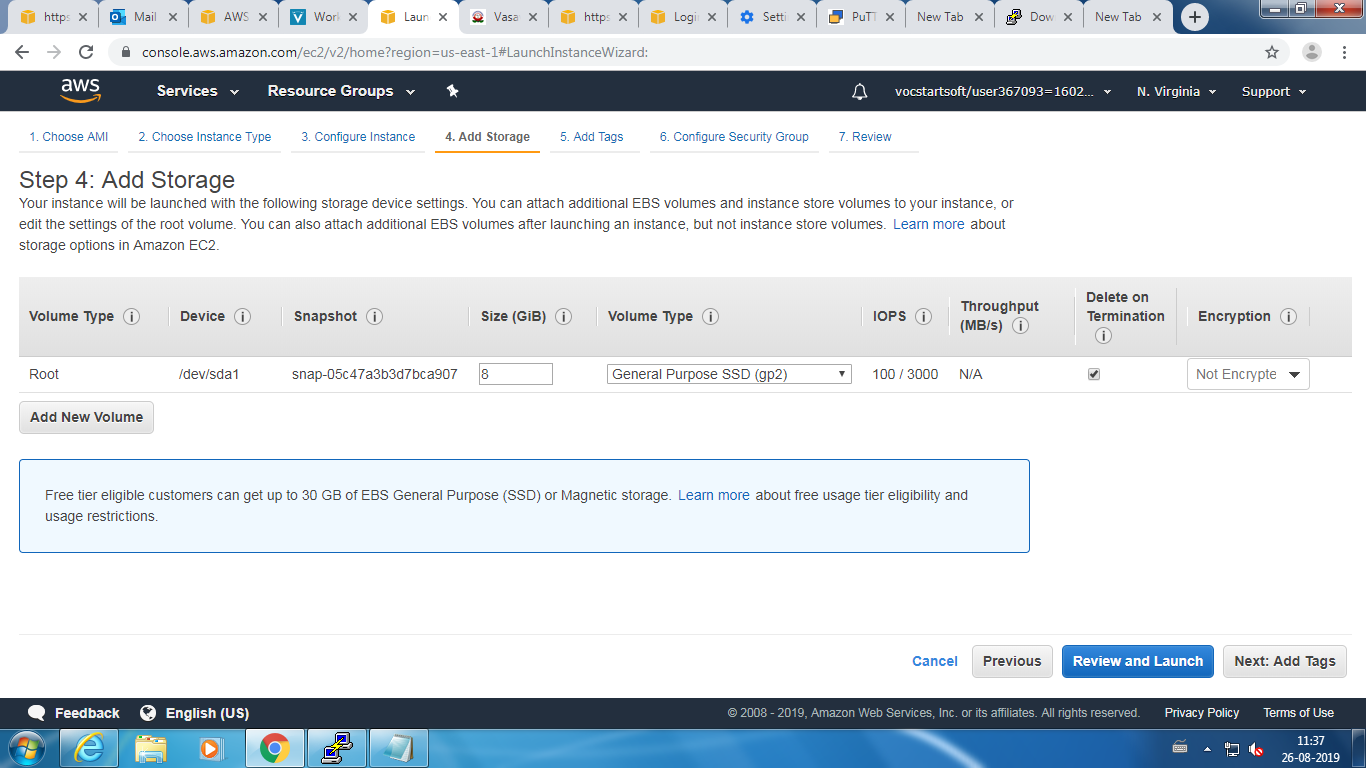


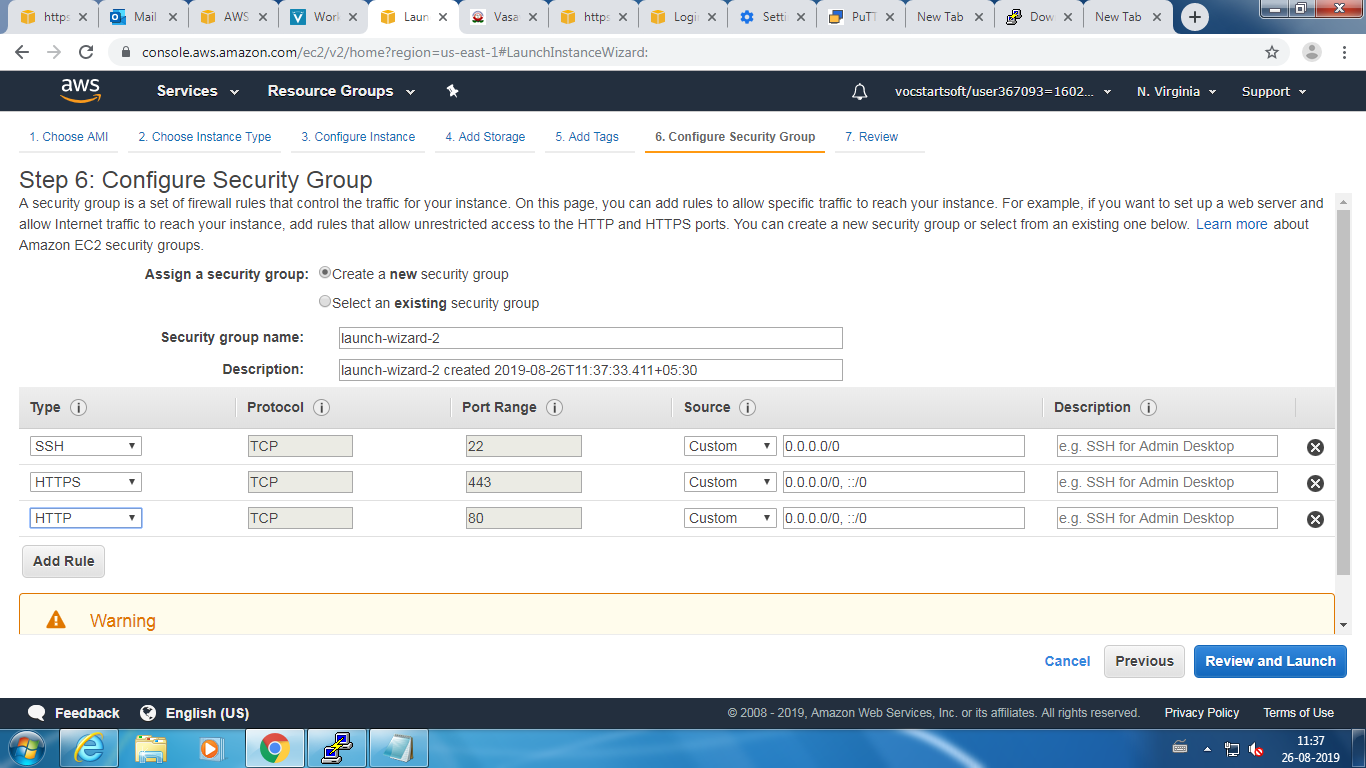


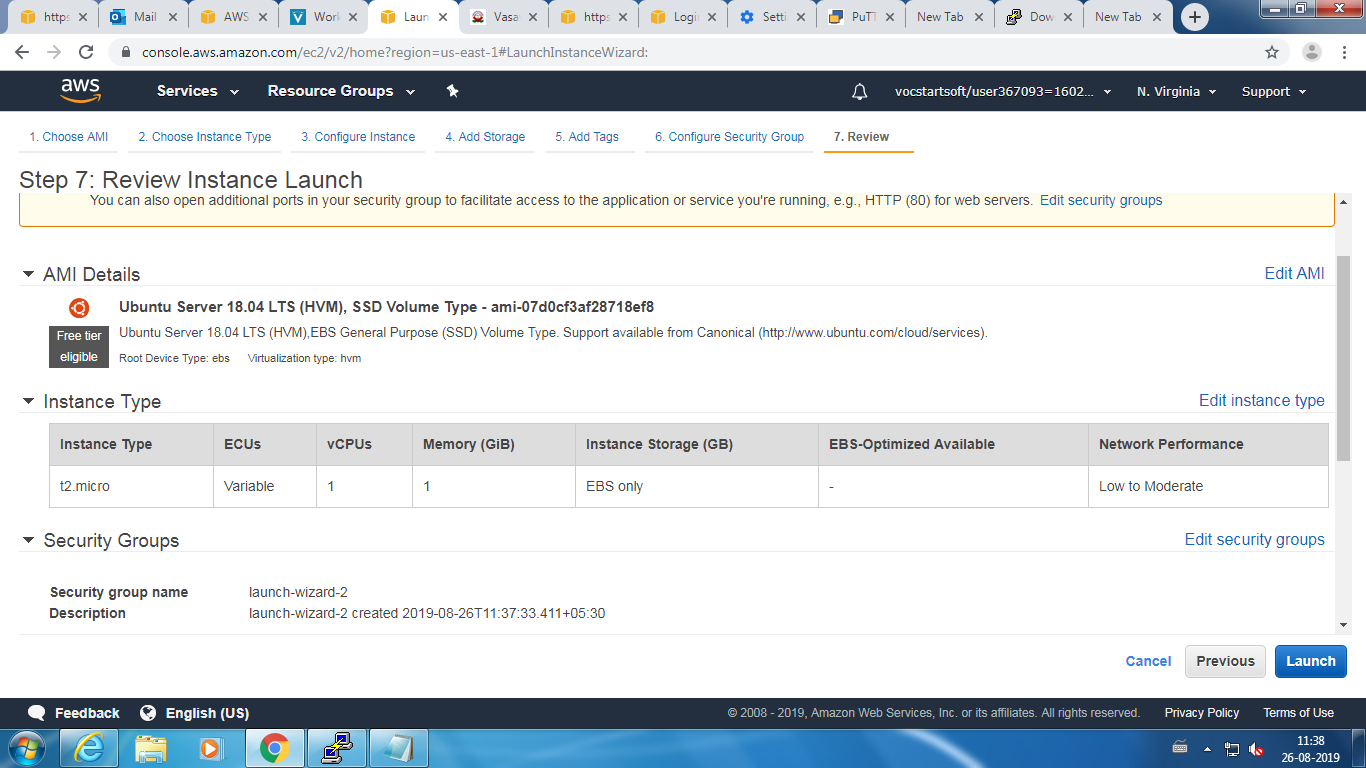


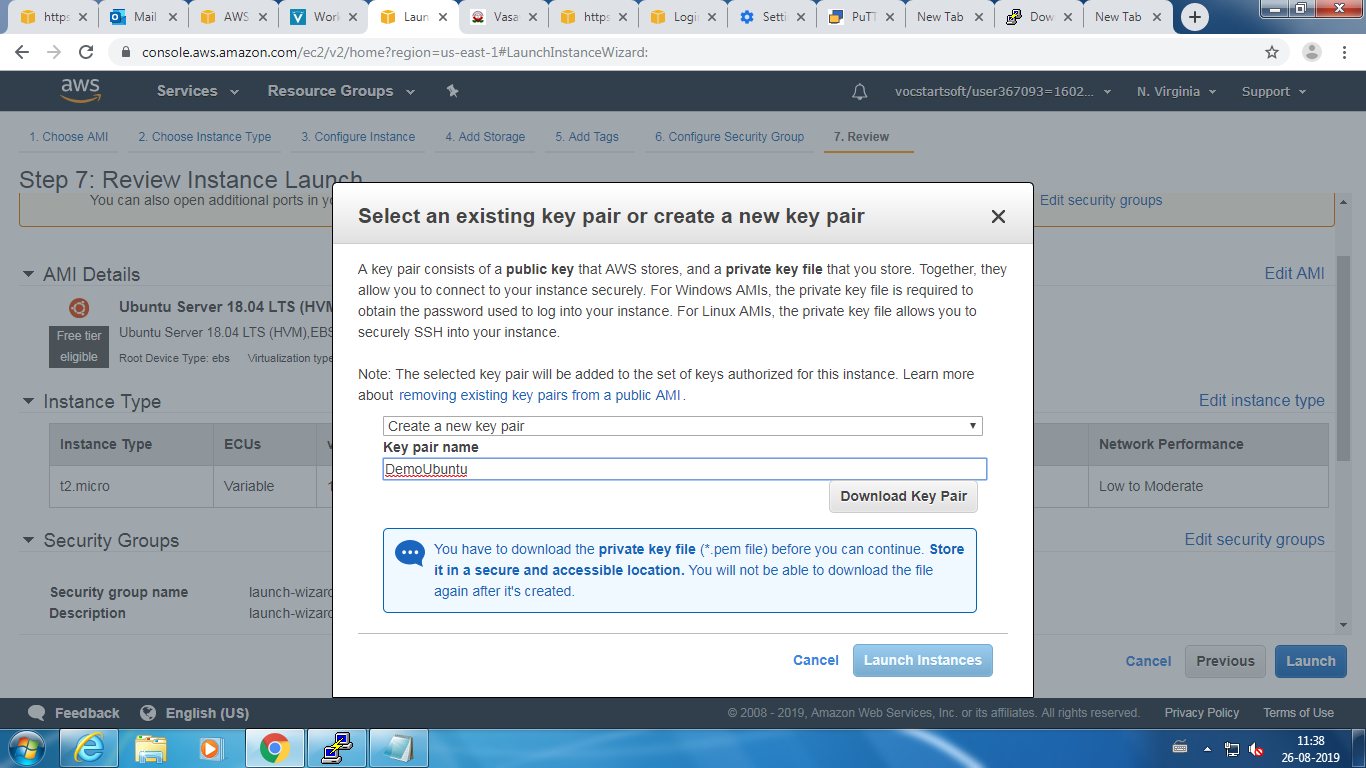


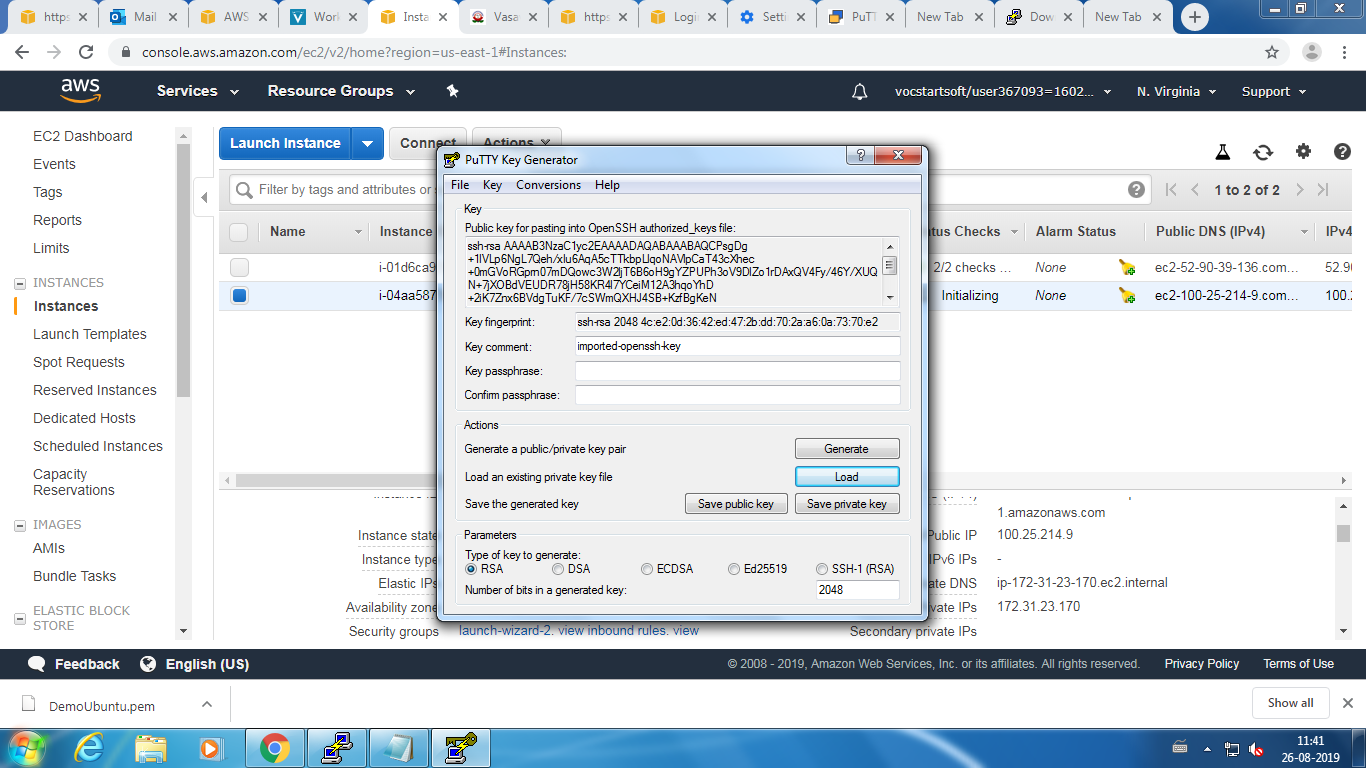


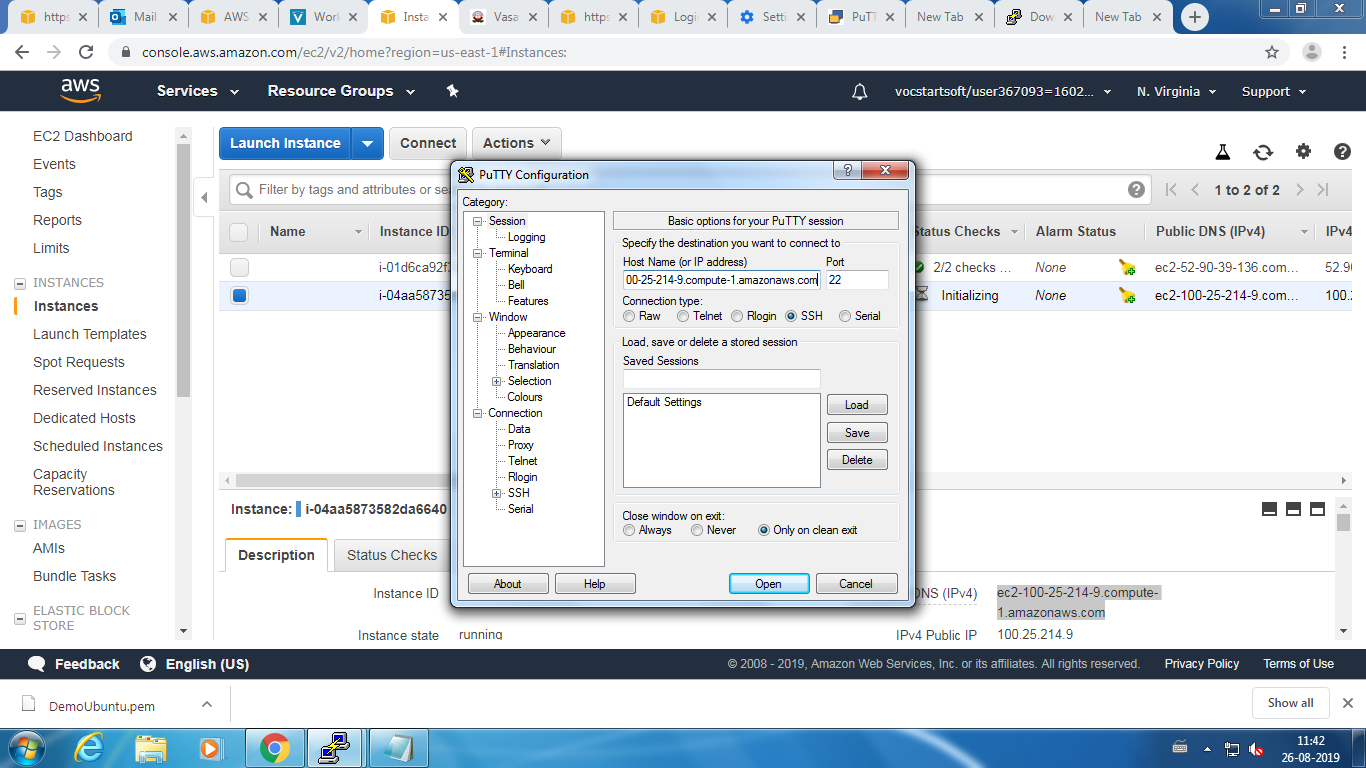


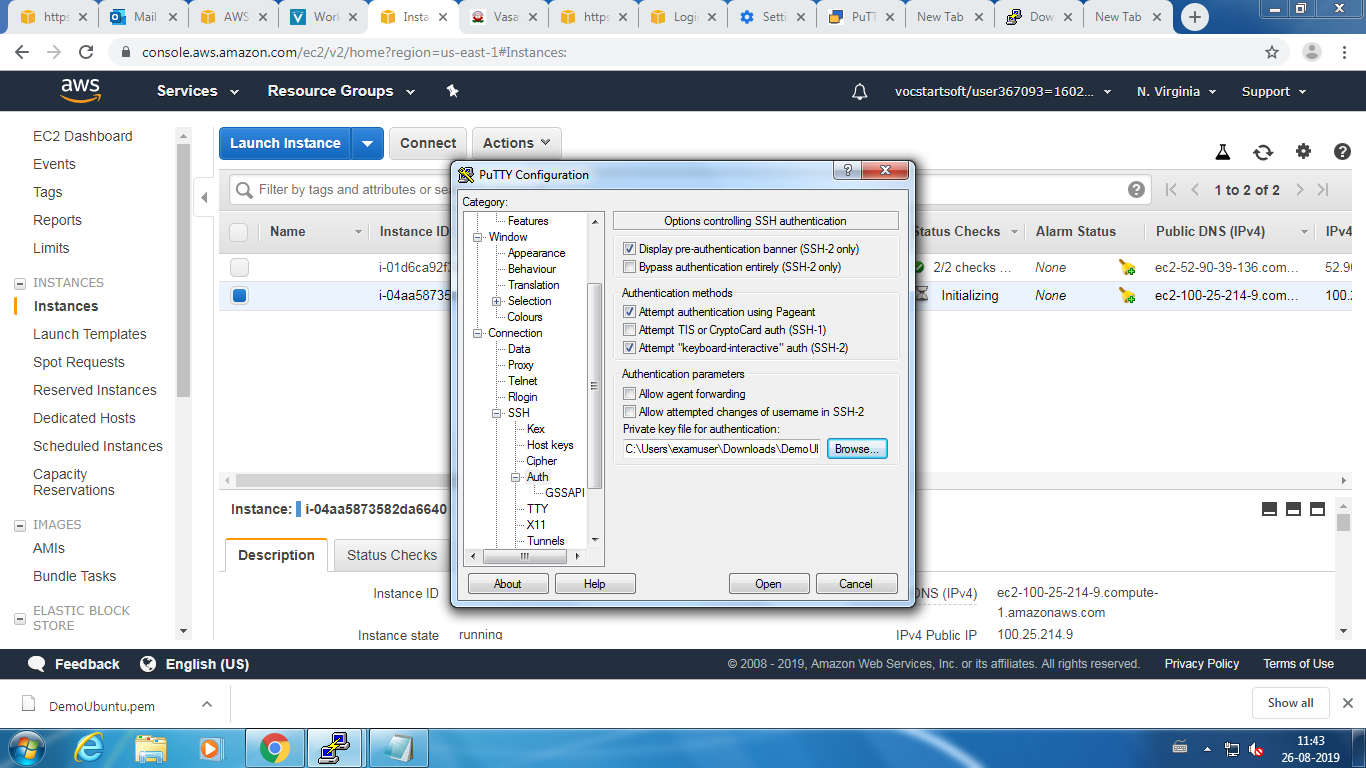












**SSH into your instance:**

Use the following commands to deploy Airlines Reservation Application

cd /home/ubuntu

mkdir demo

cd demo

git clone https://github.com/hoanghuynh1995/AirlineReservation

cd AirlineReservation

sudo apt-get update

sudo apt-get install npm

npm install

sudo apt-get install nodejs -legacy

Install Node.js.

**Source Code:**

**// server.js**

#!/usr/bin/env node

/\*\*

\* Module dependencies.

\*/

**var** app = require(‘./app/config/express’);

**var** debug = require(‘debug’)(‘AirlineReservation:server.js’);

**var** http = require(‘http’);

/\*\*

\* Get port from environment and store in Express.

\*/

**var** port = normalizePort(process.env.PORT || ‘80’);

app.set(‘port’, port);

/\*\*

\* Create HTTP server.js.

\*/

**var** server = http.createServer(app);

/\*\*

\* Listen on provided port, on all network interfaces.

\*/

server.listen(port, **function** () {

console.log(‘Server running at [http://localhost](http://localhost/):’ + port);

});

server.on(‘error’, onError);

server.on(‘listening’, onListening);

/\*\*

\* Normalize a port into a number, string, or false.

\*/

**function** normalizePort(val) {

**var** port = parseInt(val, 10);

**if** (isNaN(port)) {

// named pipe

**return** val;

}

**if** (port >= 0) {

// port number

**return** port;

}

**returnfalse**;

}

/\*\*

\* Event listener for HTTP server.js “error” event.

\*/

**function** onError(error) {

**if** (error.syscall !== ‘listen’) {

**throw** error;

}

**var** bind = **typeof** port === ‘string’

? ‘Pipe ‘ + port

: ‘Port ‘ + port;

// handle specific listen errors with friendly messages

**switch** (error.code) {

**case**‘EACCES’:

console.error(bind + ‘ requires elevated privileges’);

process.exit(1);

**break**;

**case**‘EADDRINUSE’:

console.error(bind + ‘ is already in use’);

process.exit(1);

**break**;

**default**:

**throw** error;

}

}

/\*\*

\* Event listener for HTTP server.js “listening” event.

\*/

**function** onListening() {

**var** addr = server.address();

**var** bind = **typeof** addr === ‘string’

? ‘pipe ‘ + addr

: ‘port ‘ + addr.port;

debug(‘Listening on ‘ + bind);

}

**Input / Output:**

