# **Guide to Set up Ubuntu 18.04 on AWS EC2 Instance** Reference: <https://medium.com/@jeevananandanne/guide-to-set-up-ubuntu-16-04-on-aws-ec2-instance-745f3433f16>

In this tutorial, We will learn how to configure the Ubuntu 16.04 OS on AWS (Amazon Web Service) EC2 (Elastic Cloud Compute) Instance from scratch.

#### **Sign up for AWS**

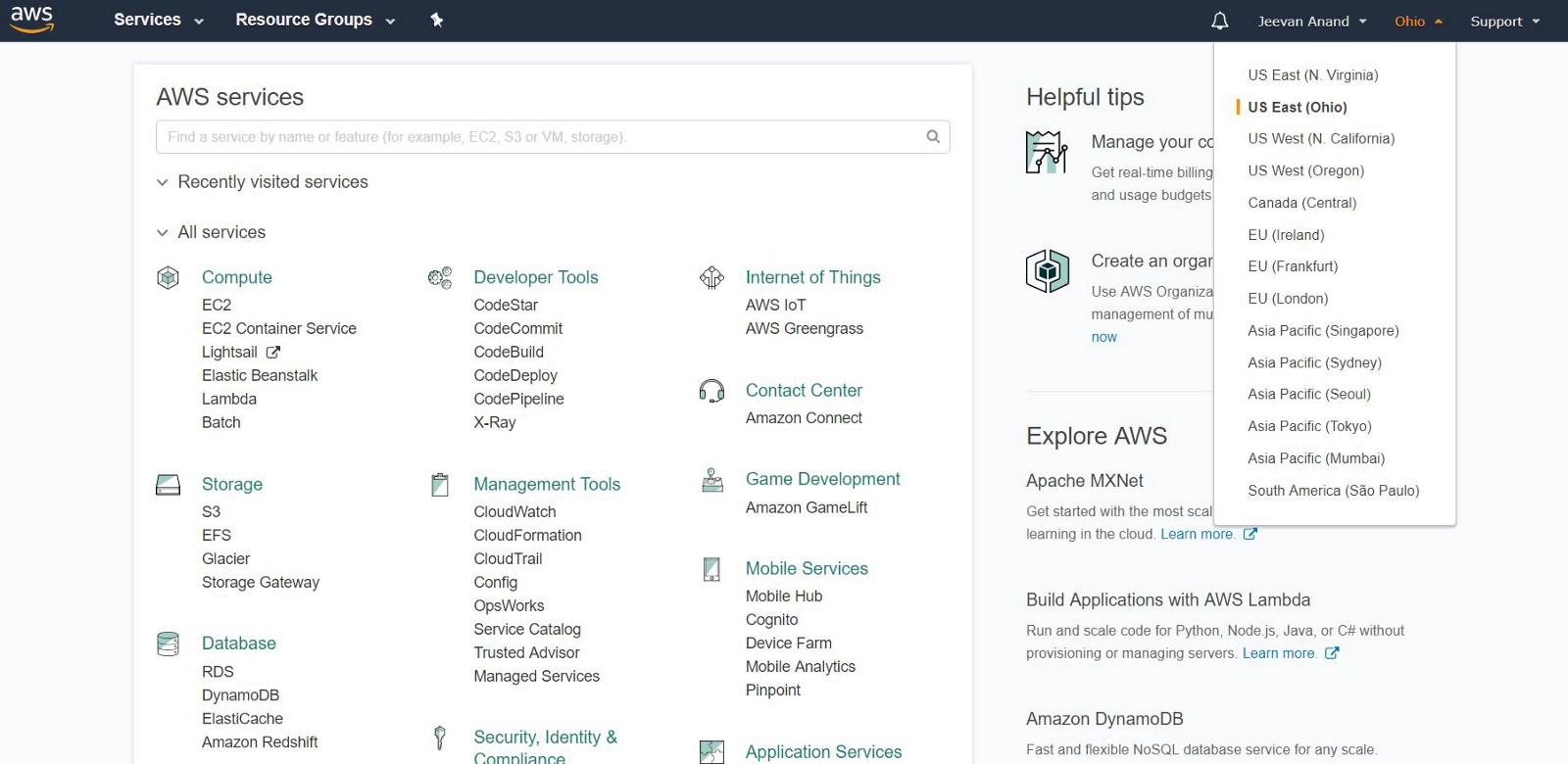
* Go to [AWS signup page](https://portal.aws.amazon.com/billing/signup?redirect_url=https%3A%2F%2Faws.amazon.com%2Fregistration-confirmation#/start), if you don’t have an account already and continue with the process. For this you would need your Credit Card details (because instances will be charged based on the usage, we are just using free tier) and a valid phone number.



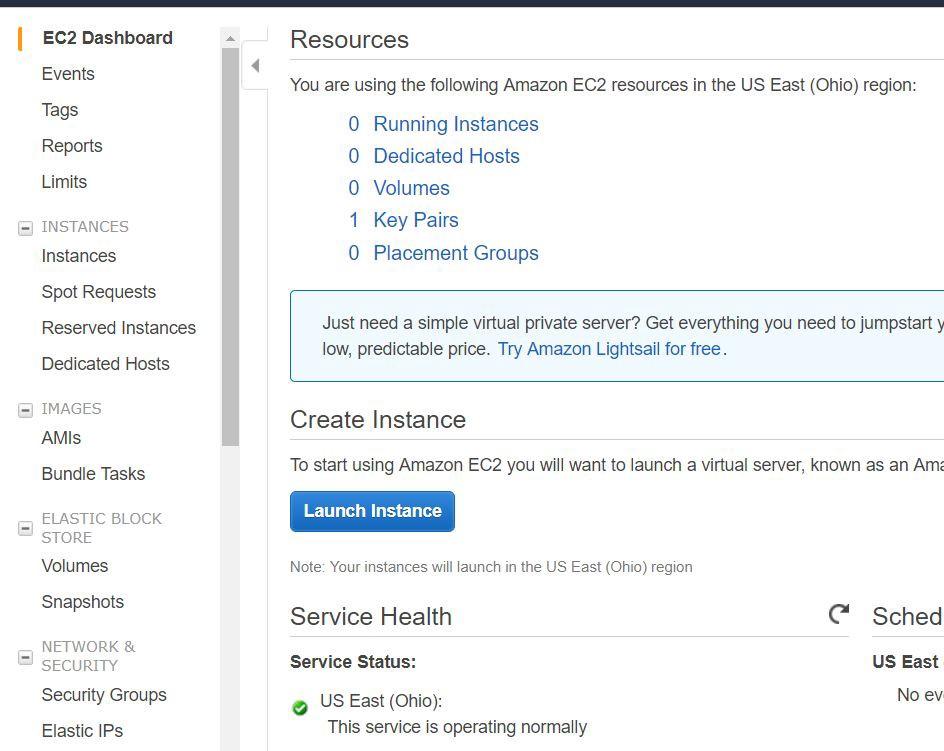
* These are virtual machines which we are going to create can be requested and run from anywhere in the world, typically at some cost per hour (recently Amazon announced [per second billing](https://aws.amazon.com/blogs/aws/new-per-second-billing-for-ec2-instances-and-ebs-volumes/)), per machine. Each virtual machine has a configurable number of CPUs and amount of memory and storage. We’re going to be using a free offering, but most organizations will need more resources to store and analyze their data.

#### **Login and access to AWS services**

* After you log in, you should choose a region that is closest to you shown here. You could select the services you need. We will be select EC2 service under Compute.

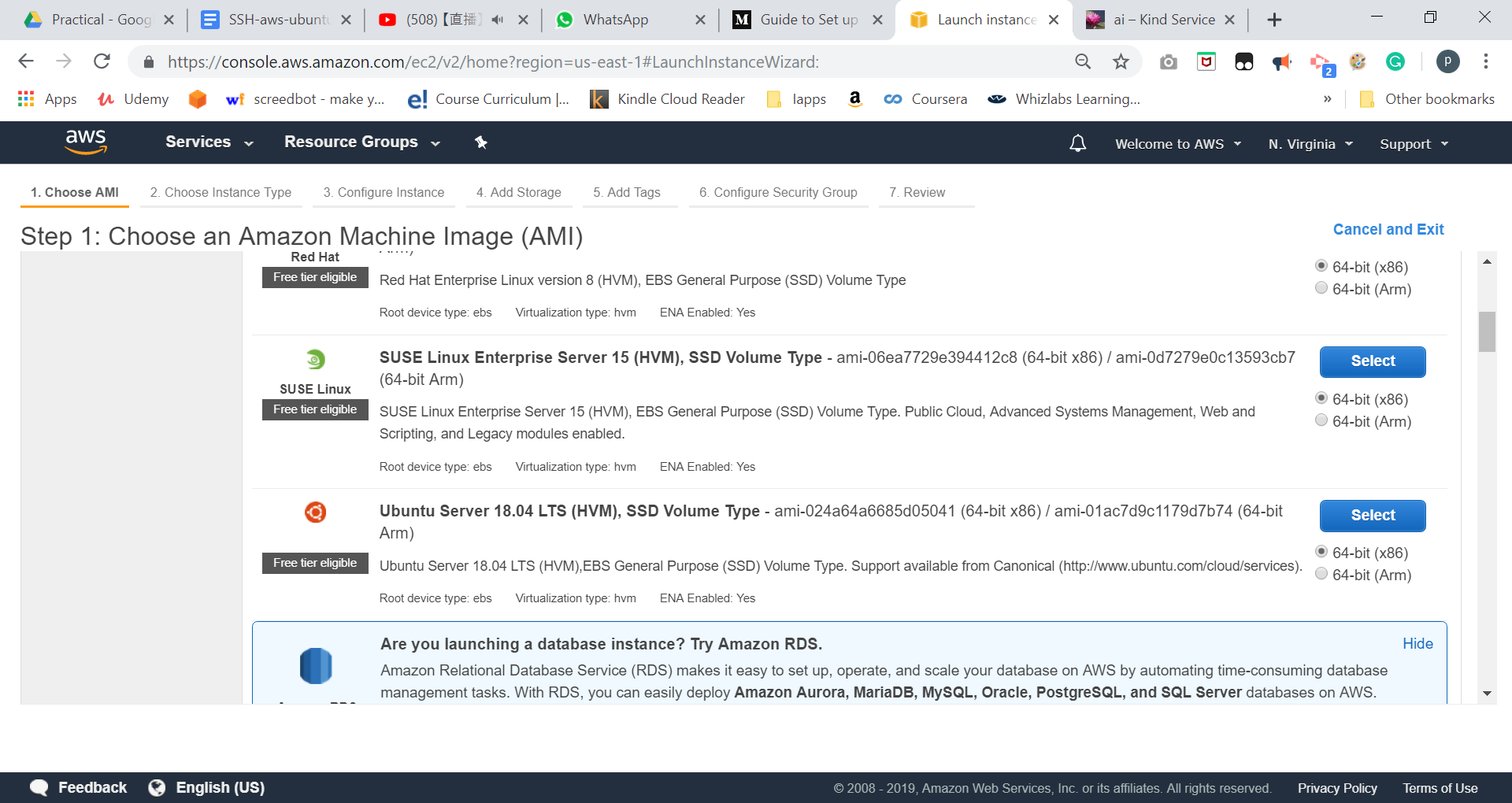


* On the EC2 dashboard, Click on ‘Launch Instance’ button in the section of Create Instance (as shown below).

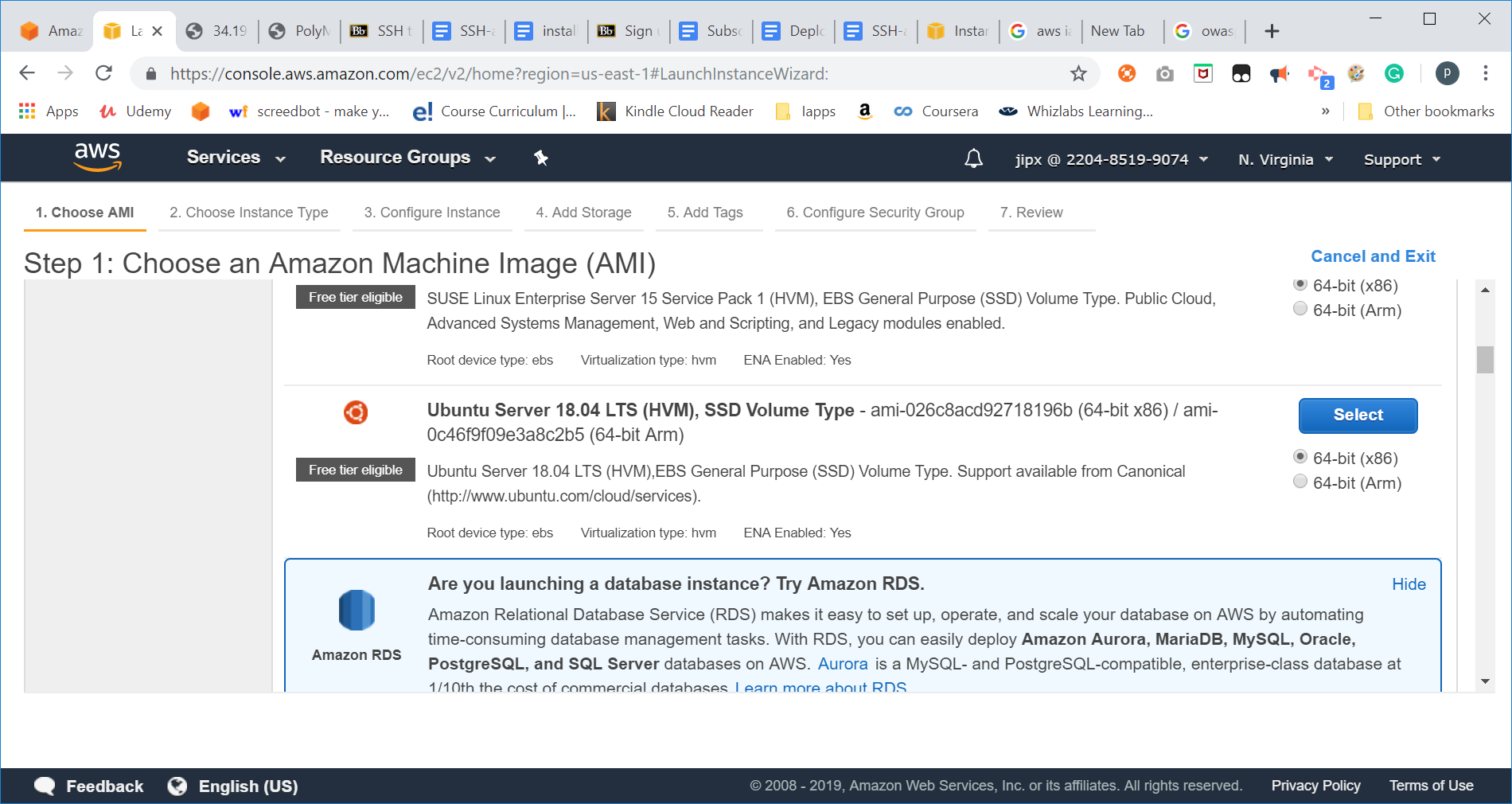


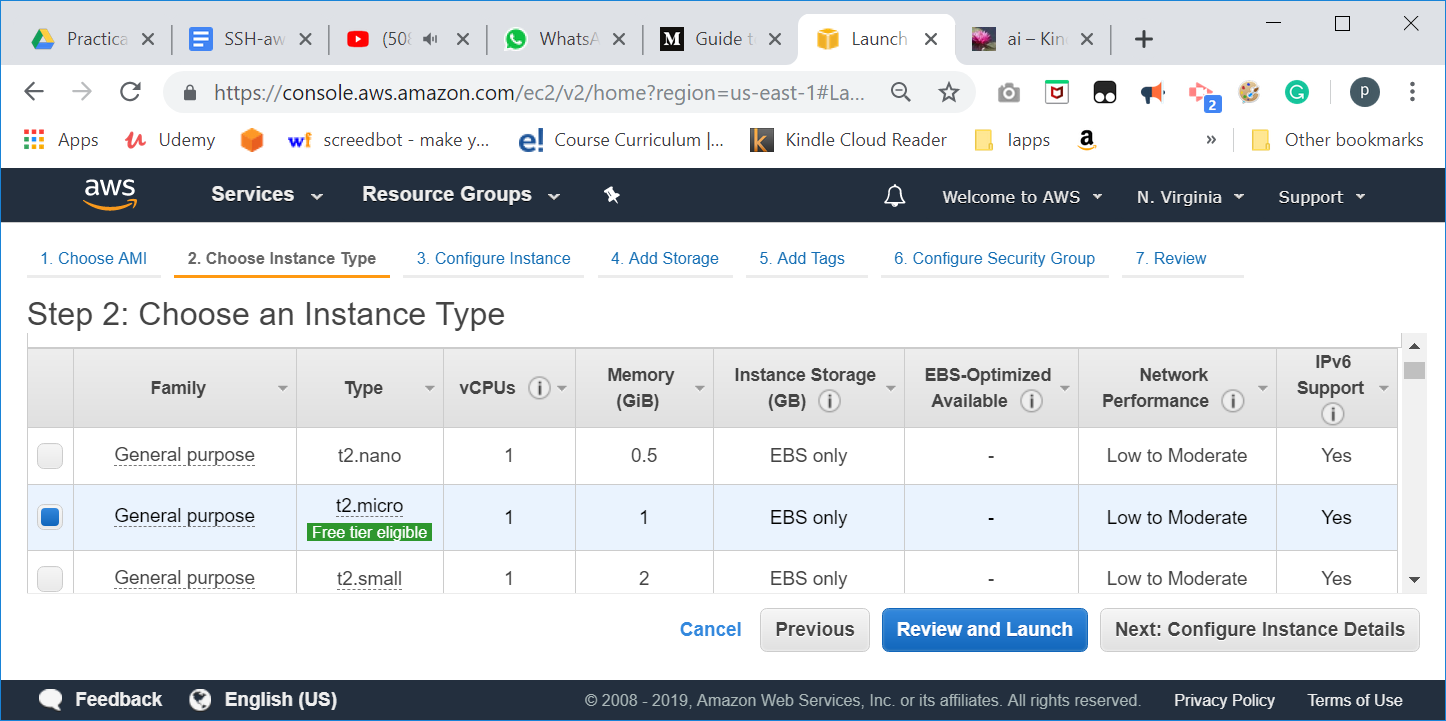
#### **Choose AMI and Instance Type**

* You will be asked to choose an AMI of your choice. (An AMI is an Amazon Machine Image. It is a template basically of an Operating System platform which you can use as a base to create your instance). Once you launch an EC2 instance from your preferred AMI, the instance will automatically be booted with the desired OS.
* Here we are choosing **Ubuntu Server 18.04 LTS (64-bit)** AMI.



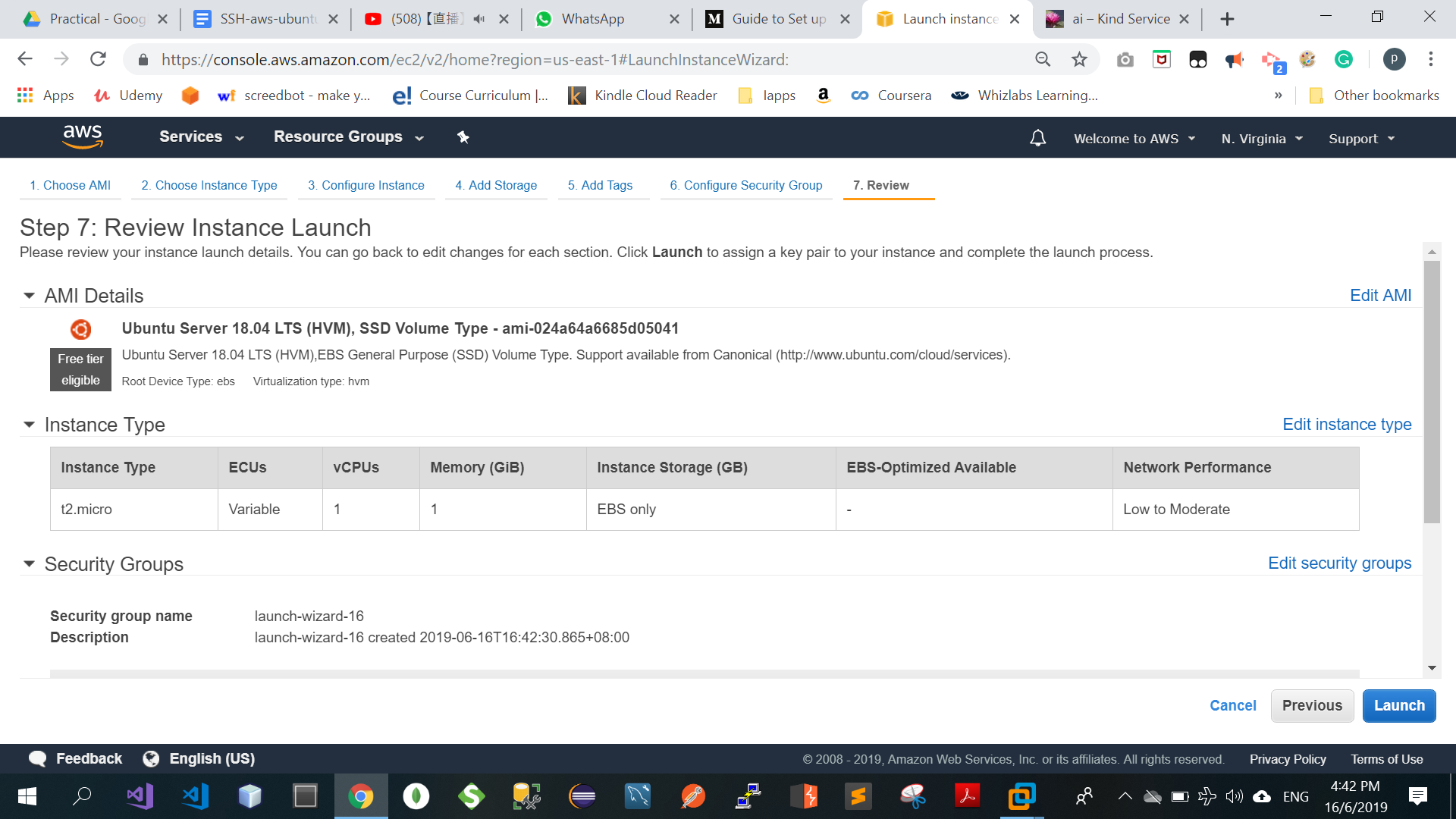
* We will choose t2.micro instance type, which is a 1vCPU and 1GB memory server offered by AWS as a free tier. Click on “Next: Configure Instance Details” for further configurations





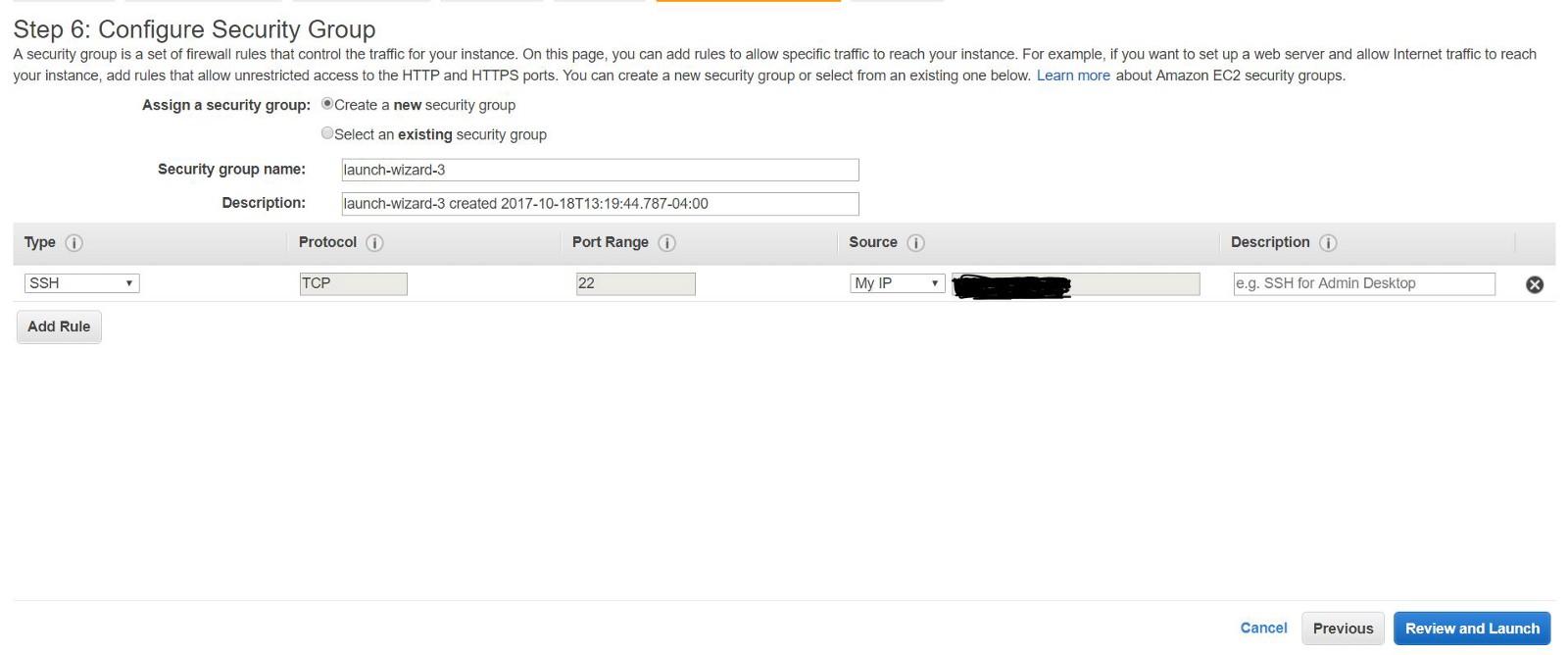
#### **Configure Instance**

* Here are you can mention the number of instances needed, also we could create new VPC, which we will not be exploring here. We will keep it as is and proceed by clicking “Next: Add Storage”



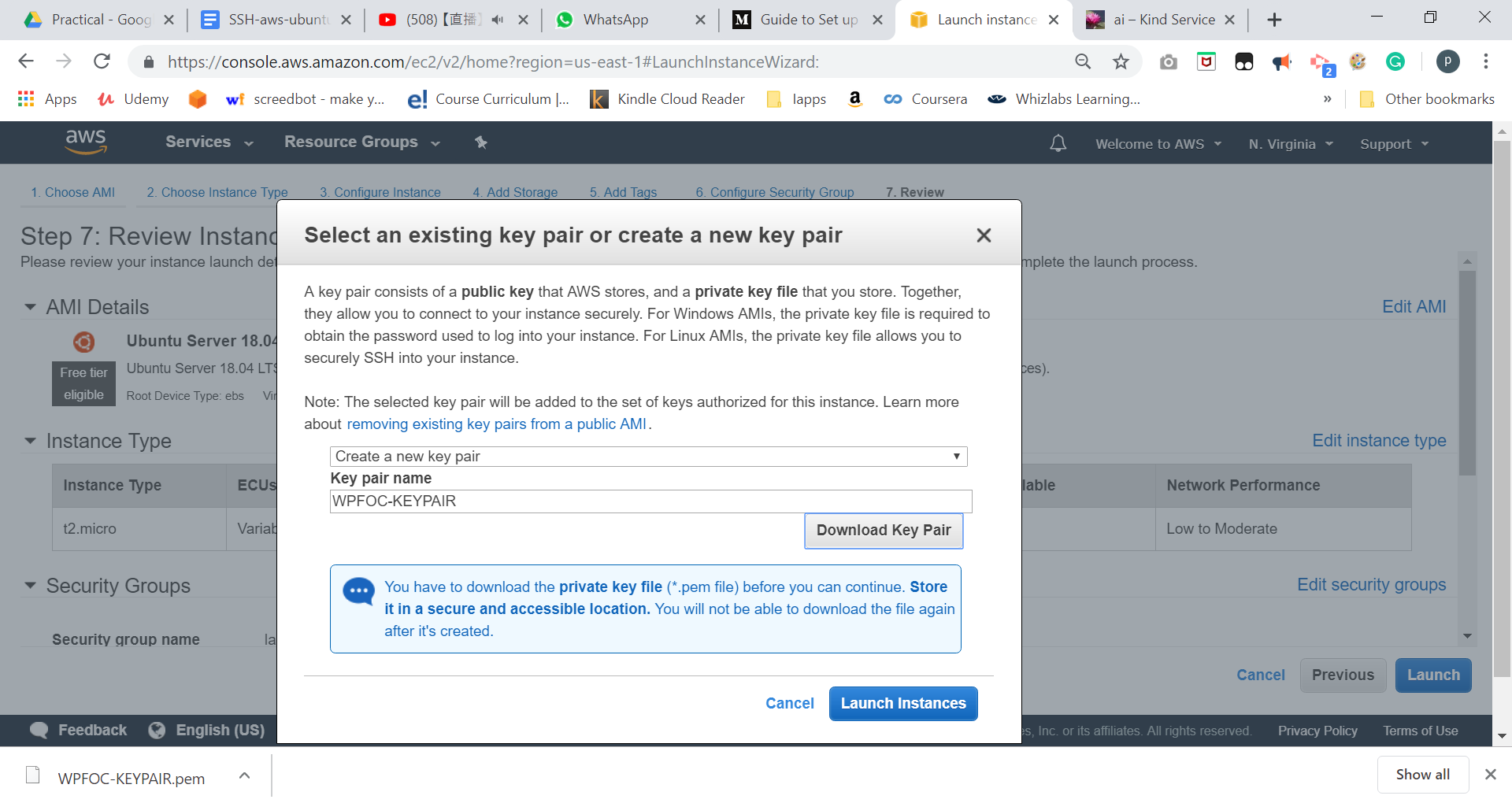
#### **Configure Security Group**

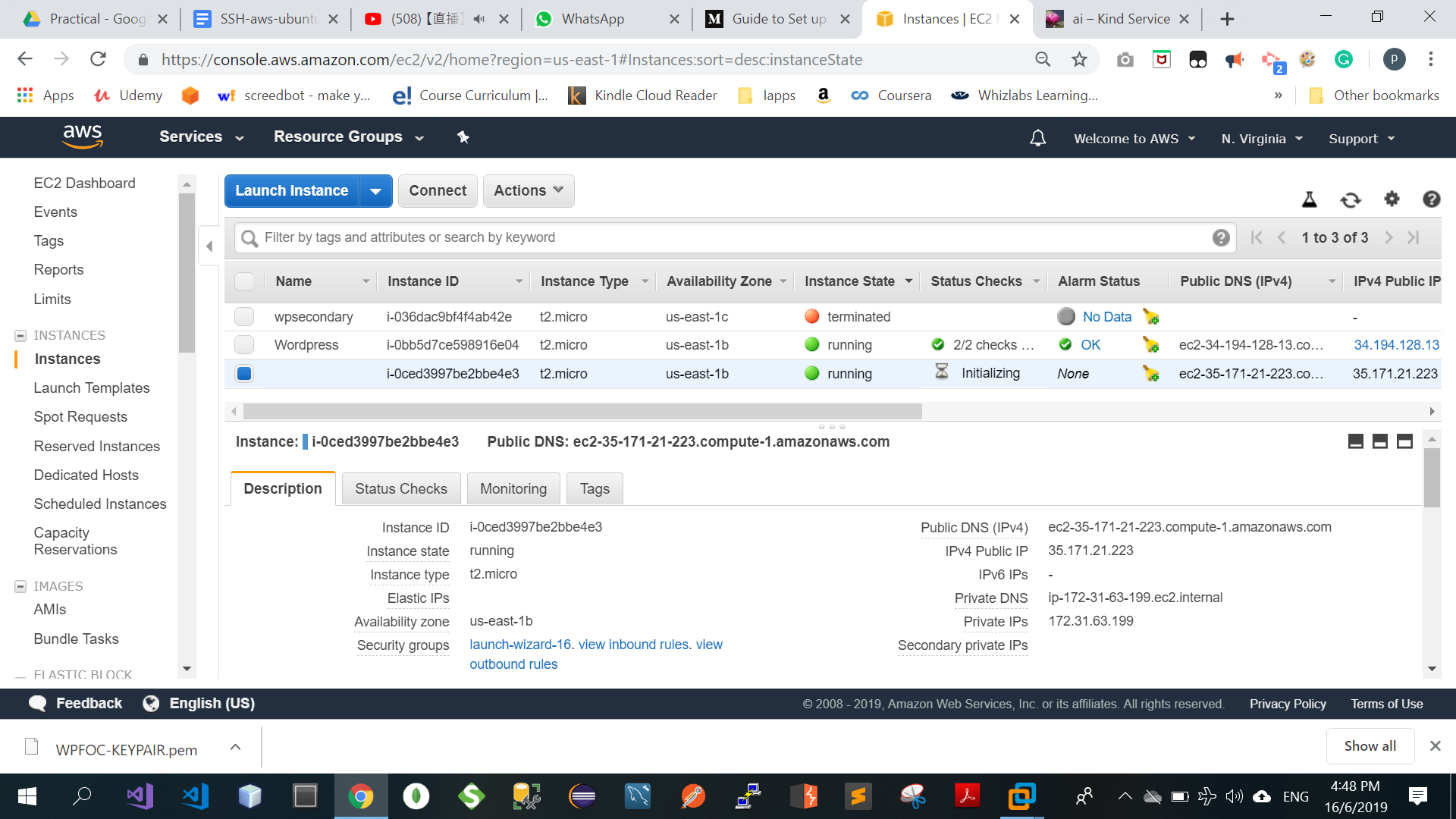
* The security group determines who can connect to your instances. The default is “SSH” and 0.0.0.0/0. This means that any computer can connect to your instance over ssh. Generally, you would create different rules to restrict list of IP addresses that can access your cluster on port range. However, for ease, you should leave it open. You’ll also need to change the Source to “My IP” (it automatically pops up the ip address) so that we can access the instances over SSH. Then, click “Review and Launch” to launch your instances.



Launch the instance by clicking on “Launch”. You will be prompted to create and download a private key. This key will allow you to connect to your instances with SSH. If you don’t download this, or delete it somehow, you won’t be able to connect to your cluster. If you lose it, don’t panic, but you’ll have to shut down the instances and start up new ones. Name it whatever you like, I just used “WPF

OC-KEYPAIR.” Click “View Instances” to see your instance booting up in the EC2 dashboard. You’ll want to write down the public hostname, called “Public DNS” on the instance panel.





#### Record down your public DNS for AWS Ubuntu instance:

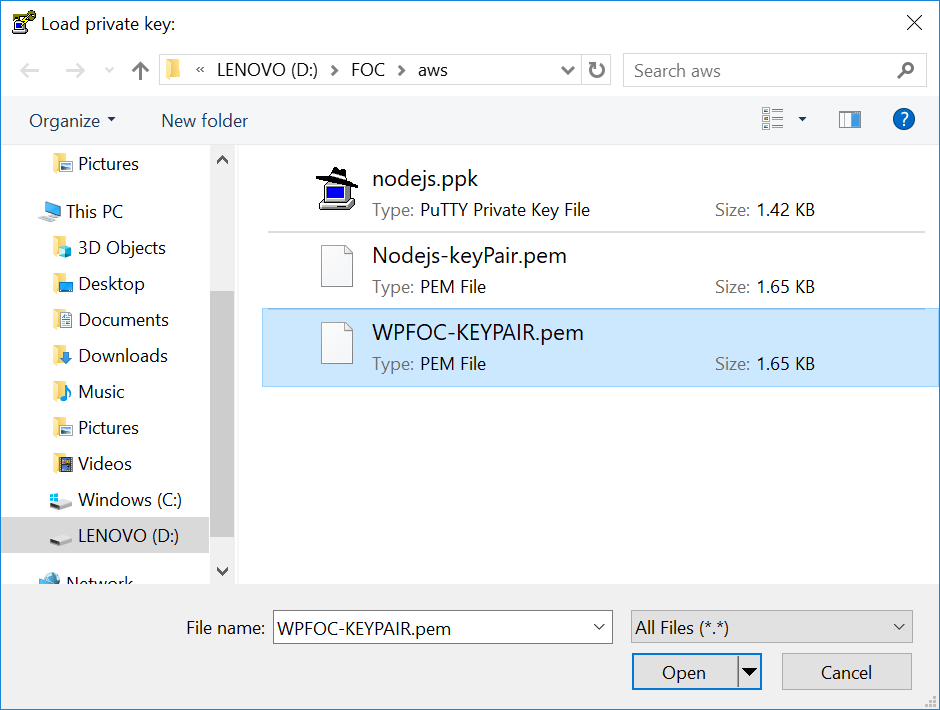
|  |  |
| --- | --- |
| Public DNS(for example: **ec2-35-171-21-223.compute-1.amazonaws.com)** |  |

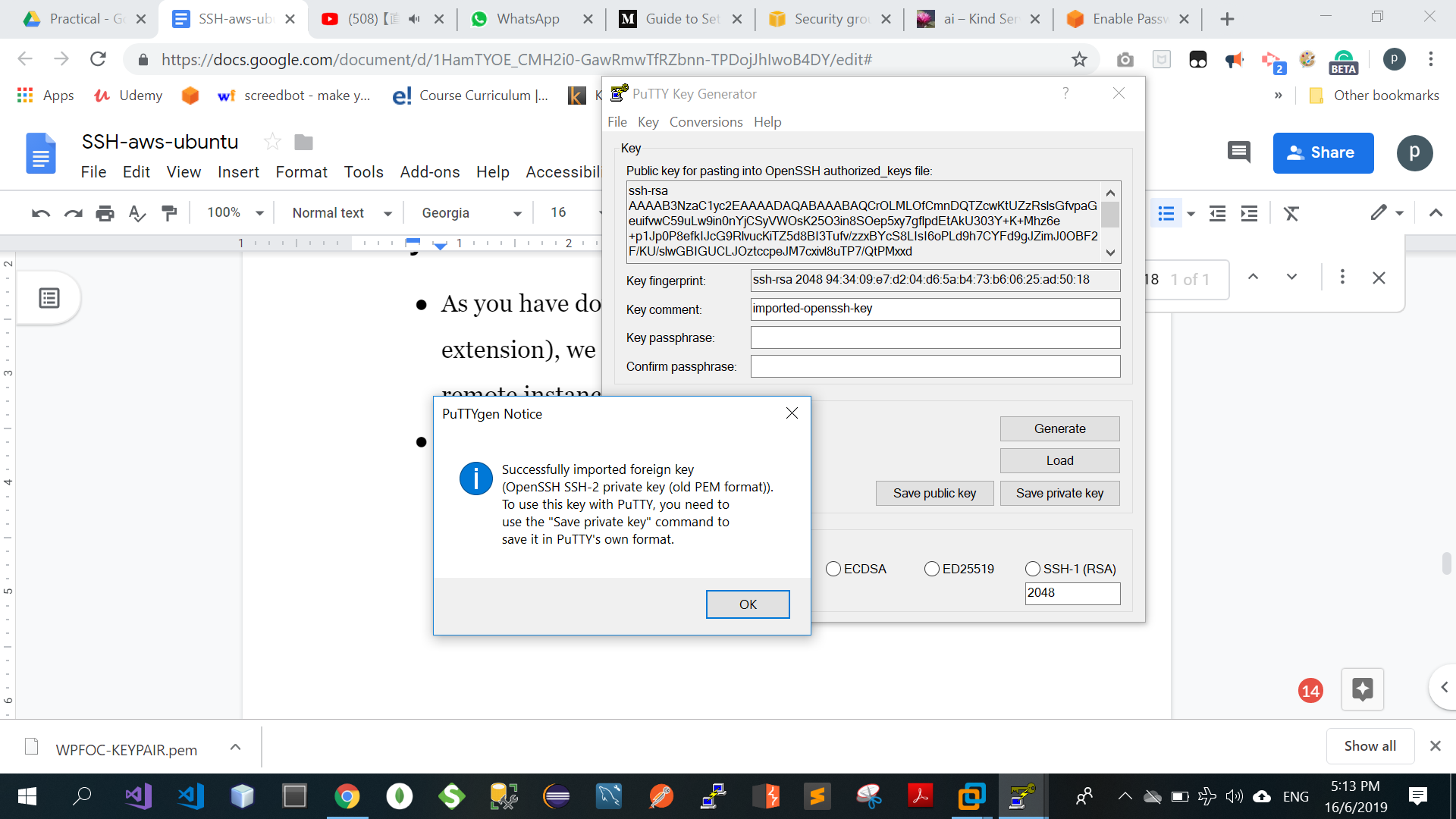
#### 

#### **Connecting to the Created Remote Instance using Putty**

|  |
| --- |
| **Convert Your Private Key Using PuTTYgen** **PuTTY does not natively support the private key format (.pem) generated by Amazon EC2. PuTTY has a tool named PuTTYgen, which can convert keys to the required PuTTY format (.ppk). You must convert your private key into this format (.ppk) before attempting to connect to your instance using PuTTY.** |

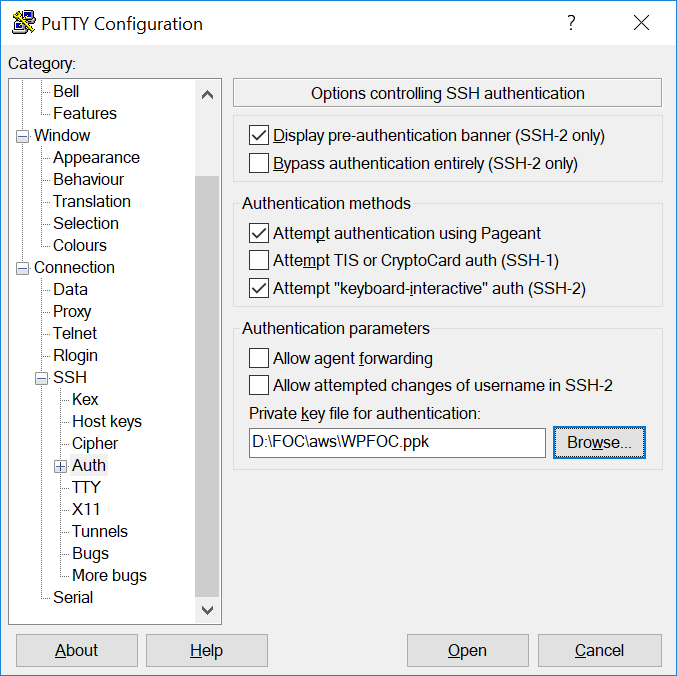
* As you have downloaded the key pair (which is with .pem extension), we need to generate private key to login to the remote instance using putty.
* By installing [Putty](https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html), PuttyGen will also be installed. Open the PuttyGen, select “File” -> “Load private key”.
* Under **Type of key to generate**, choose **RSA**.  
  
         RSA key in PuTTYgen
          
  If you're using an older version of PuTTYgen, choose **SSH-2 RSA**.
* Choose **Load**. By default, PuTTYgen displays only files with the extension .ppk. To locate your .pem file, select the option to display files of all types.  
  
         Select all file types
        
* Select your .pem file for the key pair that you specified when you launched your instance and choose **Open**. Choose **OK**.



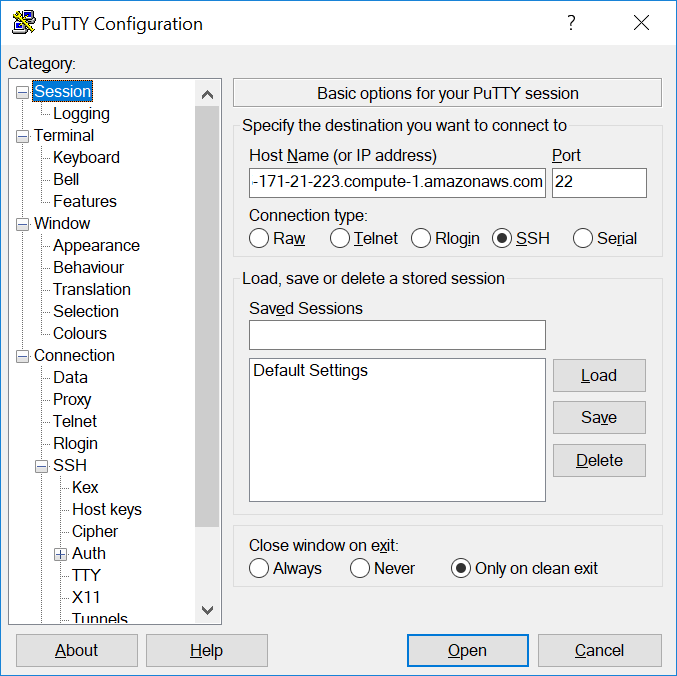
* It will generate the private key. Save private key to a file by selecting “Save private key”, name it as “WPFOC.ppk”   
  To save the key in the format that PuTTY can use, choose **Save private key**. PuTTYgen displays a warning about saving the key without a passphrase. Choose **Yes**.  
  

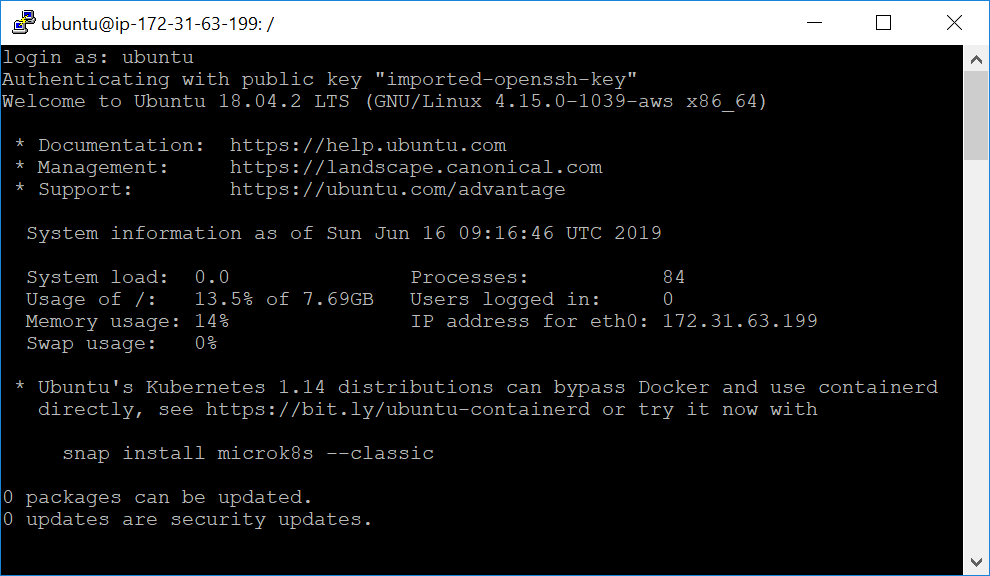
**Note**A passphrase on a private key is an extra layer of protection. Even if your private key is discovered, it can't be used without the passphrase. The downside to using a passphrase is that it makes automation harder because human intervention is needed to log on to an instance, or to copy files to an instance.

* Now, open Putty, Select the “Connection” -->SSH -->Auth and then load the .ppk file as shown below.



* Fill the Hostname field with **<your public dns>** . Also you can save the session by giving it a name and clicking on “Save”.



* Click on Open, and “yes” to login as “ubuntu” to the server.  
  If this is the first time you have connected to this instance, PuTTY displays a security alert dialog box that asks whether you trust the host to which you are connecting. (reference: [Connecting to Your Linux Instance from Windows Using PuTTY)](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html)

You are logged in now.

**Note: If you forget to stop/terminate the instance when not in use, you could be hit with real sticker shock when you get your next invoice.**

# [**how to make putty ssh connection never to timeout when user is idle?**](https://askubuntu.com/questions/254750/how-to-make-putty-ssh-connection-never-to-timeout-when-user-is-idle) If you go to your putty settings -> Connection and set the value of "Seconds between keepalives" to 30 seconds this should solve your problem.PuTTY Configuration

<https://askubuntu.com/questions/254750/how-to-make-putty-ssh-connection-never-to-timeout-when-user-is-idle>

|  |
| --- |
| Beware of the SECURITY implication…. |