



Presents

# Machine Learning Bootcamp 2019

## Technical Documentation

---

# Table of Content

Python Libraries Installation Procedure.....	Pg. 02
How to set up AWS Instance for this bootcamp?.....	Pg. 05
How to connect to EC2 Instance using Putty & Puttygen ?.....	Pg. 10
How to set up Filezilla for File Transfer.....	Pg. 15
How to configure Apache server on EC2.....	Pg. 16
How to set up the Flask app on EC2 Instance.....	Pg. 17

# How to install python libraries required in the bootcamp

## # What do I need to install all python libraries?

To install the mentioned libraries you would need to **make sure that your python installation also has pip installed** and setup correctly.

## # What is pip?

pip is a package-management system used to install and manage software packages written in Python.

## # How to check if pip is installed?

To check if pip is correctly installed on your system, run the following commands:

- For Windows & Ubuntu both

- pip

If you get the output similar to what is shown below in the screenshot, it means your pip is working fine and we can move forward with the libraries installation.

```
C:\Users\Rishabh Malhotra>pip
Usage:
  pip <command> [options]

Commands:
  install           Install packages.
  download          Download packages.
  uninstall         Uninstall packages.
  freeze           Output installed packages in requirements format.
  list             List installed packages.
  show             Show information about installed packages.
  check            Verify installed packages have compatible dependencies.
  config           Manage local and global configuration.
  search           Search PyPI for packages.
  wheel            Build wheels from your requirements.
  hash            Compute hashes of package archives.
  completion       A helper command used for command completion.
  help            Show help for commands.

General Options:
  -h, --help            Show help.
  --isolated            Run pip in an isolated mode, ignoring environment variables and user configuration.
  -v, --verbose         Give more output. Option is additive, and can be used up to 3 times.
  -V, --version         Show version and exit.
  -q, --quiet           Give less output. Option is additive, and can be used up to 3 times (corresponding to WARNING, ERROR, and CRITICAL logging levels).
  --log <path>         Path to a verbose appending log.
  --proxy <proxy>       Specify a proxy in the form [user:passwd@]proxy.server:port.
  --retries <retries>   Maximum number of retries each connection should attempt (default 5 times).
  --timeout <sec>      Set the socket timeout (default 15 seconds).
  --exists-action <action> Default action when a path already exists: (s)witch, (i)gnore, (w)ipe, (b)ackup, (a)abort.
  --trusted-host <hostname> Mark this host as trusted, even though it does not have valid or any HTTPS.
  --cert <path>         Path to alternate CA bundle.
  --client-cert <path>  Path to SSL client certificate, a single file containing the private key and the certificate in PEM format.
  --cache-dir <dir>     Store the cache data in <dir>.
  --no-cache-dir        Disable the cache.
  --disable-pip-version-check Don't periodically check PyPI to determine whether a new version of pip is available for download. Implied with --no-index.
  --no-color            Suppress colored output
```

## # How to install pip?

If you don't get the above output, you need to install pip. To do that follow the below-given instructions:

1. Save the pip installation script from the url <https://bootstrap.pypa.io/get-pip.py> on your system.
2. Open the command prompt (or terminal), navigate to the location where you saved the pip installation script & run the following command  
`python get-pip.py`

## # Installing numpy

- On Windows
  - `pip install numpy`
- On Ubuntu
  - `sudo pip install numpy`

## # Installing pandas

- On Windows
  - `pip install pandas`
- On Ubuntu
  - `sudo pip install pandas`

## # Installing sklearn

- On Windows
  - `pip install sklearn`
- On Ubuntu
  - `sudo pip install sklearn`

## # Installing opencv

- On Windows
  - `pip install opencv-python`
- On Ubuntu
  - `sudo pip install opencv-python`

### # Installing PIL

- On Windows
  - `pip install pillow`
- On Ubuntu
  - `sudo pip install pillow`

### # Installing flask

- On Windows
  - `pip install flask`
- On Ubuntu
  - `sudo pip install flask`

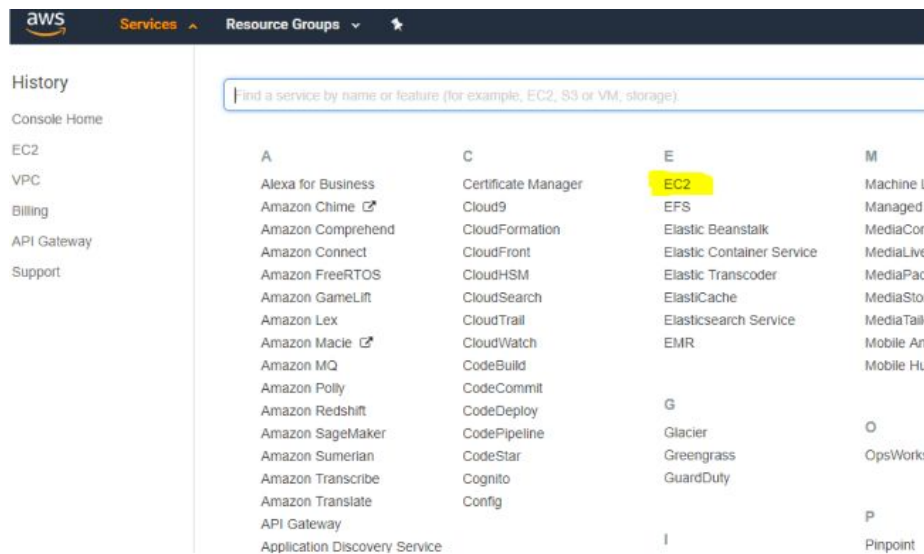
### # Installing matplotlib

- On Windows
  - `pip install matplotlib`
- On Ubuntu
  - `sudo pip install matplotlib`

# How to set up AWS Instance for this bootcamp?

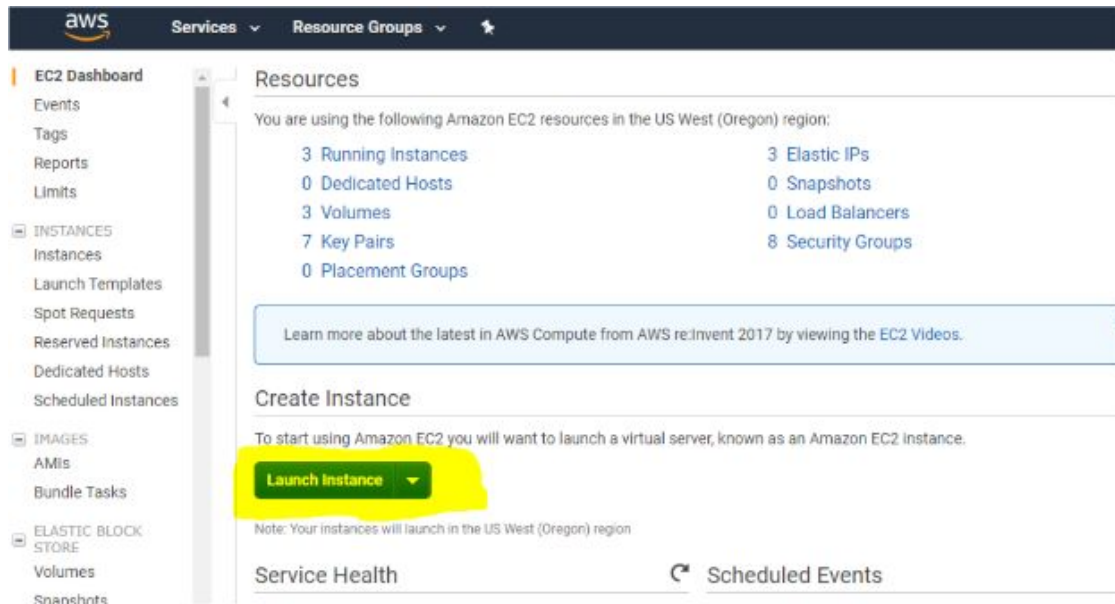
## # Creating an EC2 Instance on AWS for Model Training & API Deployment

**Step1.** Select the EC2 option from the services dropdown on your aws dashboard.

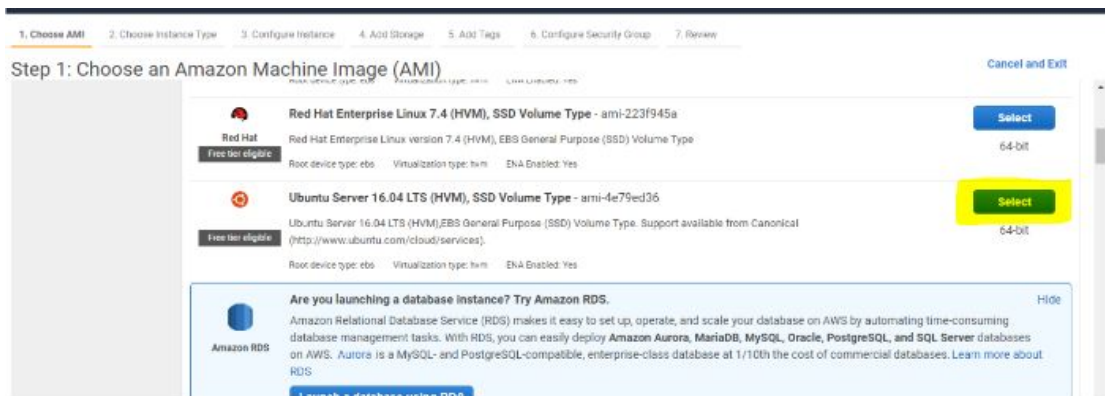


## Section 2 - AWS EC2 Instance Setup

**Step 2.** On the next screen of EC2 Dashboard. Select the Launch Instance Option



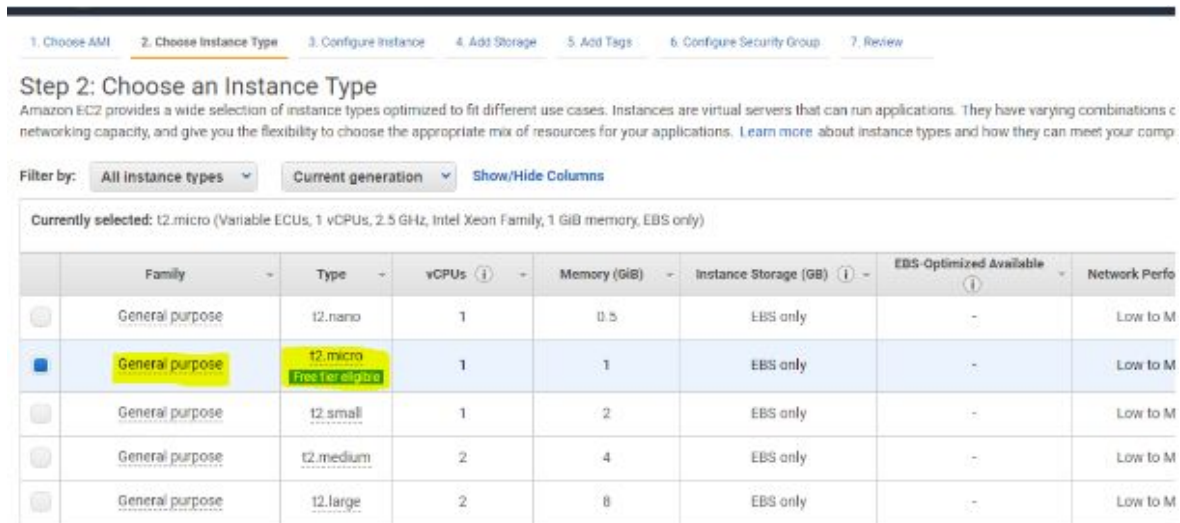
**Step 3.** In this bootcamp we are using Ubuntu Server 16.04 based machine, so select that from the list of choices for machine image.



## Section 2 - AWS EC2 Instance Setup

**Step 4.** Now select the instance type - t2.micro (It comes under the Free Tier Label).

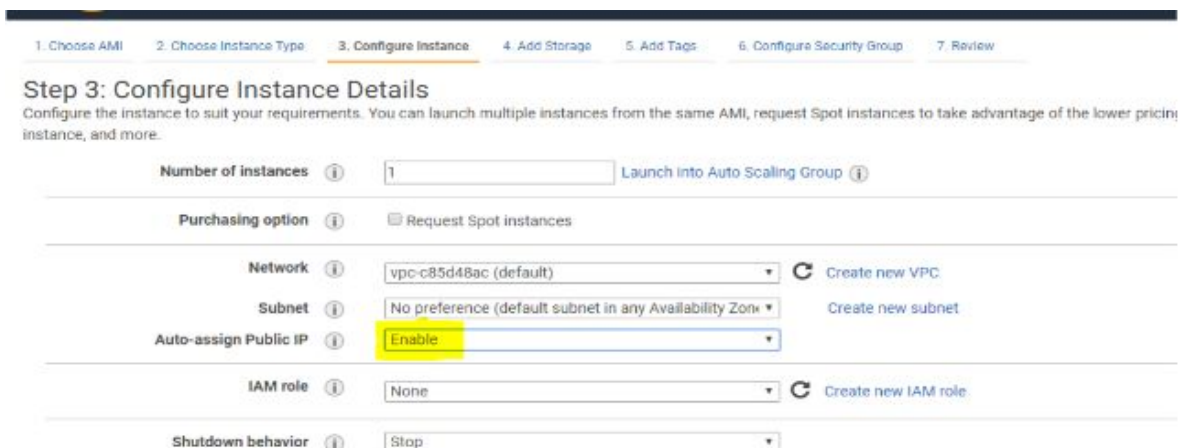
**NOTE:** One thing to note here is that you might have to change (upgrade) your machine at a later stage when we would be training the model as t2.micro would not be able to handle the load for that. The good news is you can perfectly do that by just stopping the machine through EC2 Dashboard and upgrading it to your specific configuration and then restarting the machine. Don't worry I will tell you how to do this when you would require it.



The screenshot shows the 'Step 2: Choose an Instance Type' page in the AWS Management Console. The navigation bar at the top includes: 1. Choose AMI, 2. Choose Instance Type (active), 3. Configure Instance, 4. Add Storage, 5. Add Tags, 6. Configure Security Group, and 7. Review. Below the navigation bar, the title 'Step 2: Choose an Instance Type' is followed by a descriptive paragraph. A filter bar shows 'All instance types' selected, with 'Current generation' and 'Show/Hide Columns' options. The 'Currently selected' text indicates 't2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)'. A table lists various instance types with columns for Family, Type, vCPUs, Memory (GiB), Instance Storage (GB), EBS-Optimized Available, and Network Performance. The 't2.micro' instance type is highlighted in yellow, and its 'Free tier eligible' status is also highlighted.

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Perfo
<input type="radio"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to M
<input checked="" type="radio"/>	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to M
<input type="radio"/>	General purpose	t2.small	1	2	EBS only	-	Low to M
<input type="radio"/>	General purpose	t2.medium	2	4	EBS only	-	Low to M
<input type="radio"/>	General purpose	t2.large	2	8	EBS only	-	Low to M

**Step 5.** Enable Auto Assign IP - to open access to the machine through the internet.

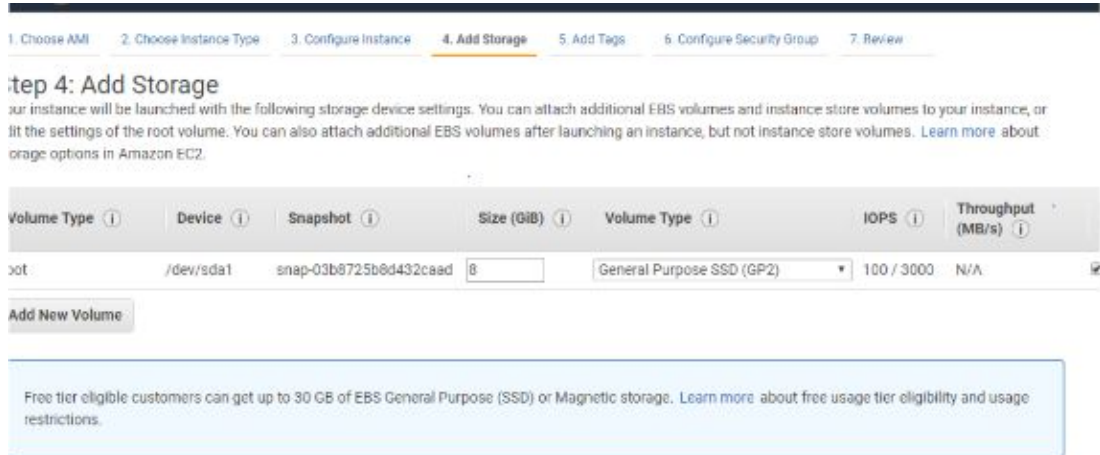


The screenshot shows the 'Step 3: Configure Instance Details' page in the AWS Management Console. The navigation bar at the top includes: 1. Choose AMI, 2. Choose Instance Type, 3. Configure Instance (active), 4. Add Storage, 5. Add Tags, 6. Configure Security Group, and 7. Review. Below the navigation bar, the title 'Step 3: Configure Instance Details' is followed by a descriptive paragraph. The configuration options are as follows: 'Number of instances' is set to 1, with a 'Launch into Auto Scaling Group' link; 'Purchasing option' is set to 'Request Spot instances'; 'Network' is set to 'vpc-c85d48ac (default)', with a 'Create new VPC' link; 'Subnet' is set to 'No preference (default subnet in any Availability Zone)', with a 'Create new subnet' link; 'Auto-assign Public IP' is set to 'Enable'; 'IAM role' is set to 'None', with a 'Create new IAM role' link; and 'Shutdown behavior' is set to 'Stop'.



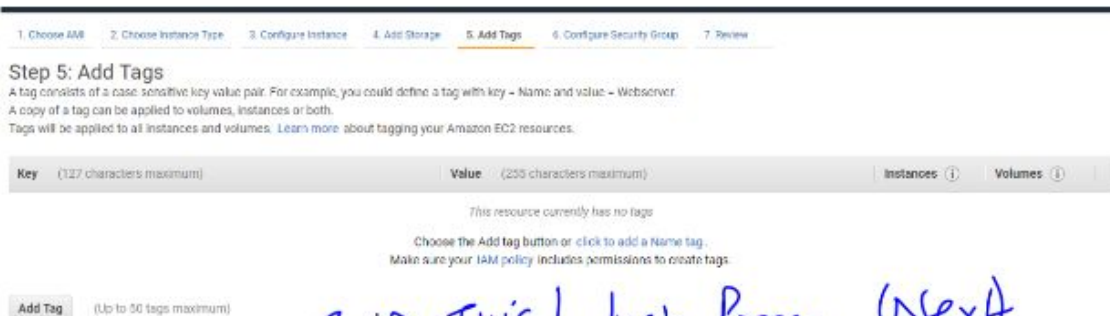
## Section 2 - AWS EC2 Instance Setup

**Step 6.** For this step of adding storage, just set the Default Settings and continue.



⇒ (Use Default settings for this)

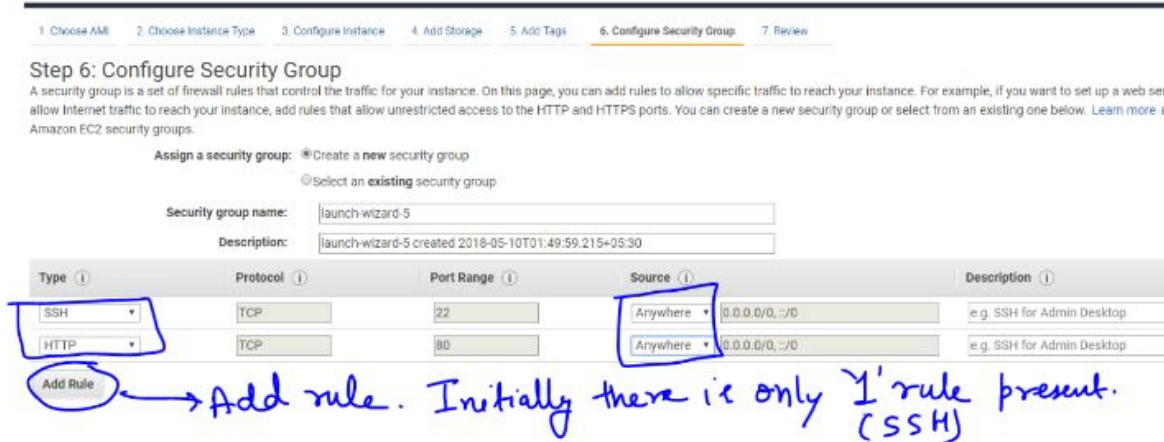
**Step 7.** In the next step of adding tags, again set the default settings and continue.



SKIP THIS! Just Press Next

## Section 2 - AWS EC2 Instance Setup

**Step 8.** Configure the security groups as shown below to have 2 rules - SSH & HTTP



**Step 6: Configure Security Group**

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server, allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more about Amazon EC2 security groups.](#)

Assign a security group: ☒ Create a new security group ☐ Select an existing security group

Security group name:

Description:

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Anywhere	e.g. SSH for Admin Desktop
HTTP	TCP	80	Anywhere	e.g. SSH for Admin Desktop

**Add Rule** → Add rule. Initially there is only 1 rule present. (SSH)

**Step 9.** Save the .pem key to access the EC2 Machine and then launch.



**Select an existing key pair or create a new key pair**

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.

Create a new key pair

Key pair name:

**Download Key Pair**

You have to download the **private key file** (\*.pem file) before you can continue. Store it in a **secure and accessible location**. You will not be able to download the file again after it's created.

**Make Sure you download**

**Press Launch to open**

**Now press this**

Cancel Launch Instances

**Step 10.** Set up an elastic IP for the instance. Elastic IP - remains the same (whereas the ip which comes associated by default with the instance changes) when you restart the instance, thus giving you better control over the instance.

Watch the [Elastic-IP-CREATION.gif](#) at the following url to see how it's done - <https://bit.ly/Elastic-IP-CREATION>

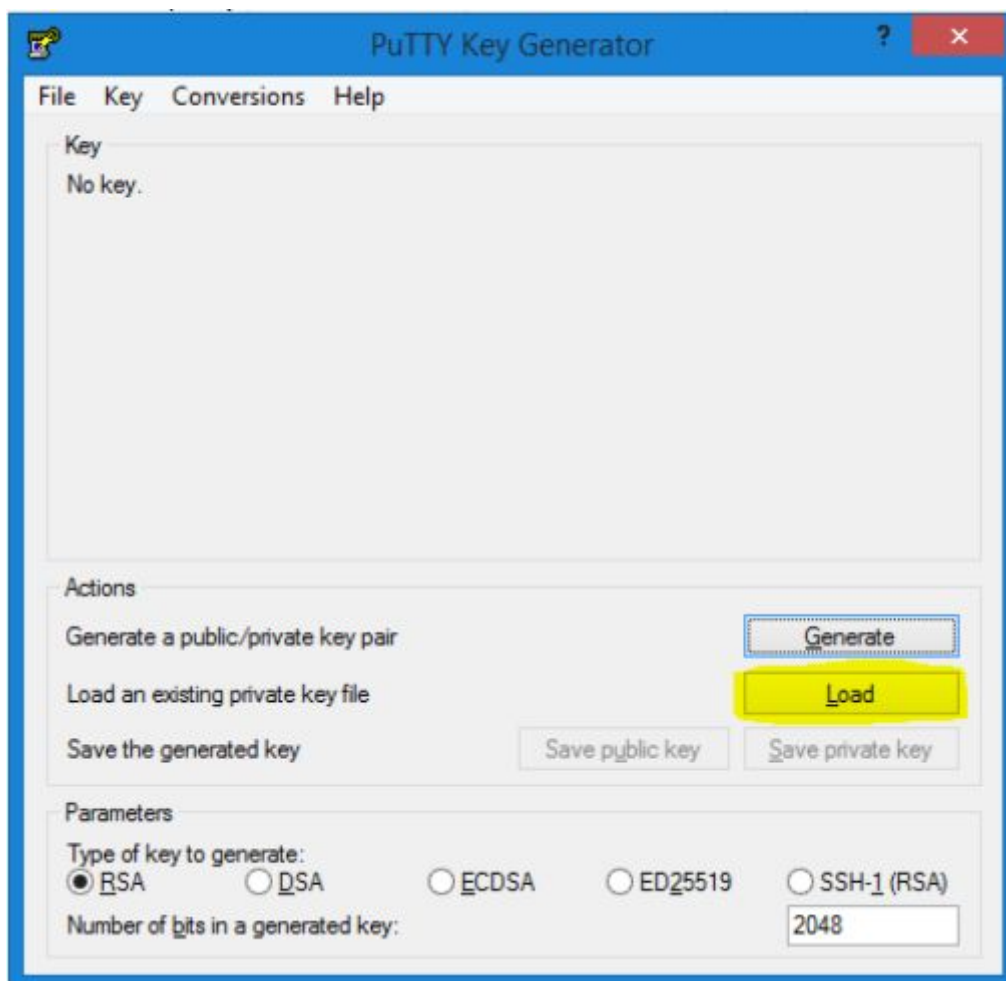
**Step 11.** Associate the elastic IP Address created in the last step with your ec2 instance.

Watch the [Elastic-IP-ASSOCIATION.gif](#) at the following url to see how it's done - <https://bit.ly/Elastic-IP-ASSOCIATION>

# How to connect to EC2 Instance using Putty & Puttygen ?

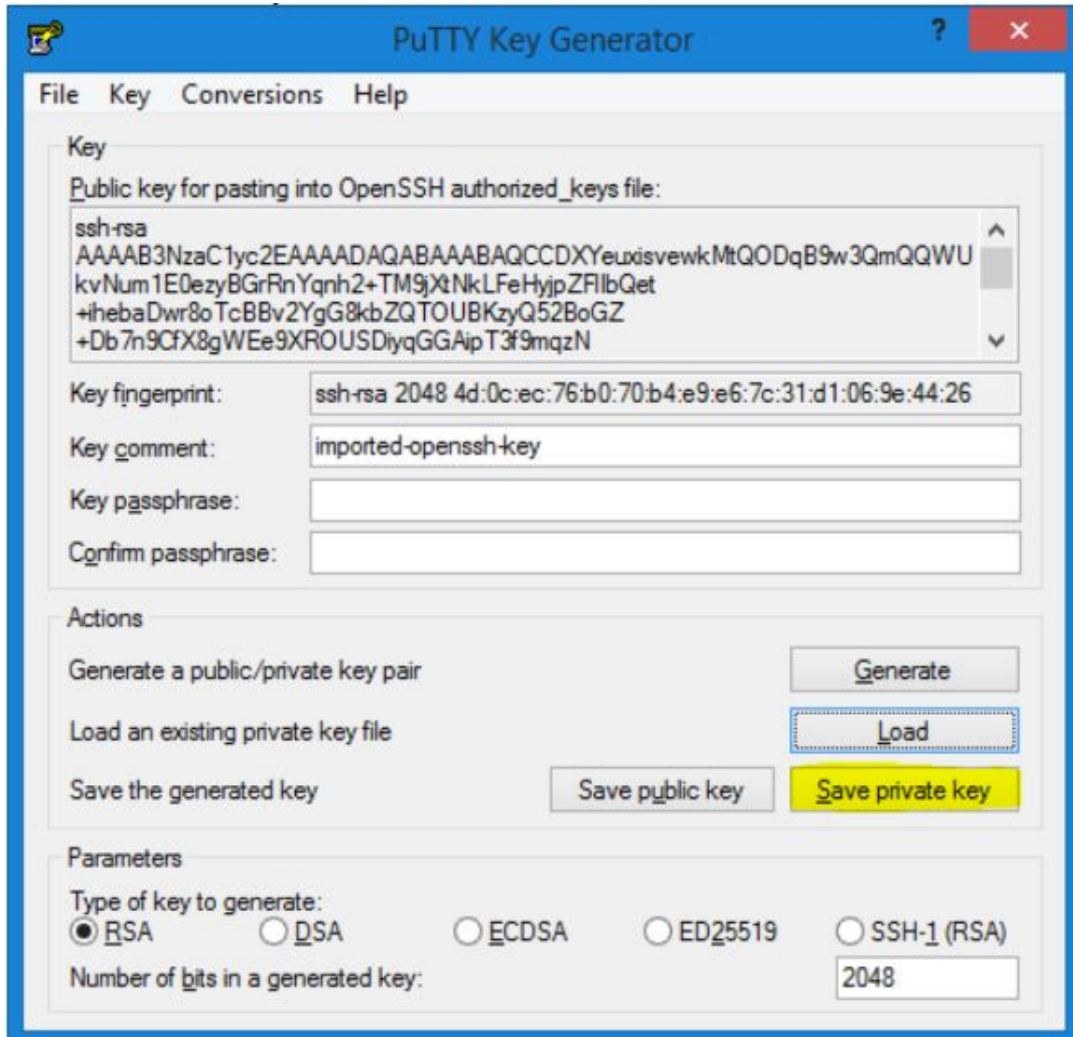
---

**Step 1.** Convert key from .pem to .ppk. To do this you need to start puttygen and press the highlighted load button in puttygen. You need to load your .pem key file which you downloaded from aws while creating the EC2 Instance.

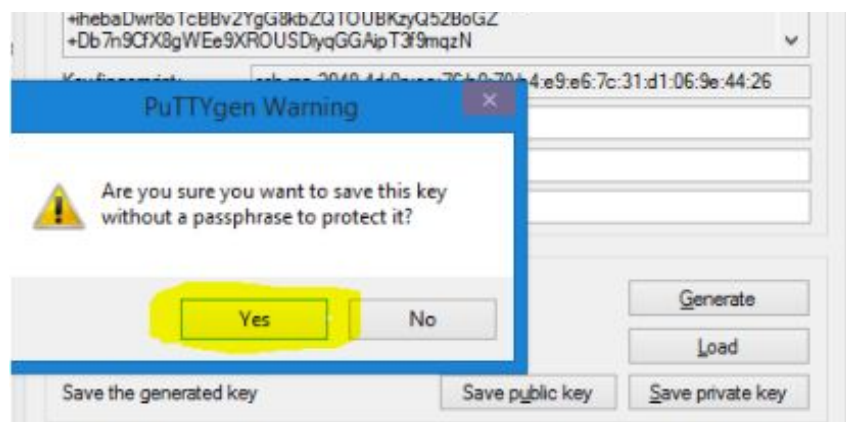


## Section 3 - Connecting with EC2 Instance using Putty & Puttygen

**Step 2.** Then you need to save the private key by pressing the “Save Private Key” Button.



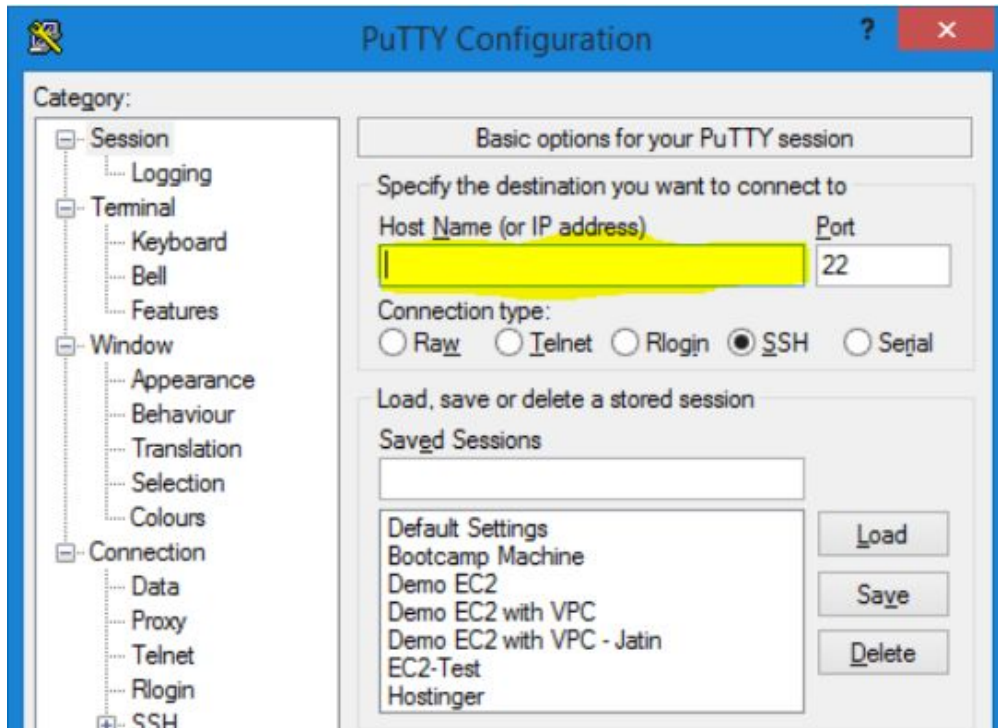
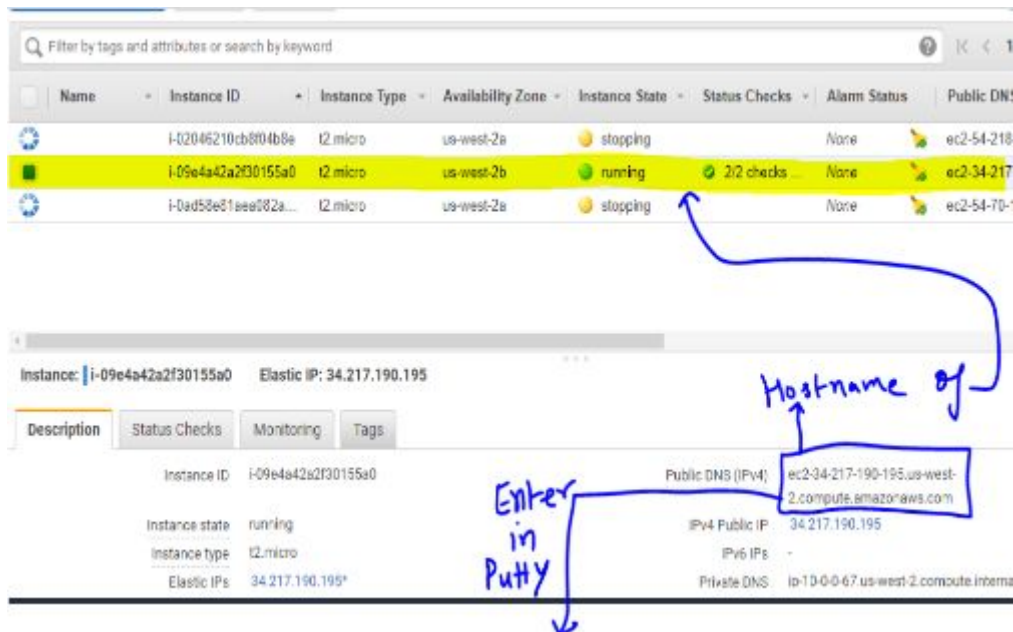
**Step 3.** You need to save it without the paraphrase. So ignore the warning.





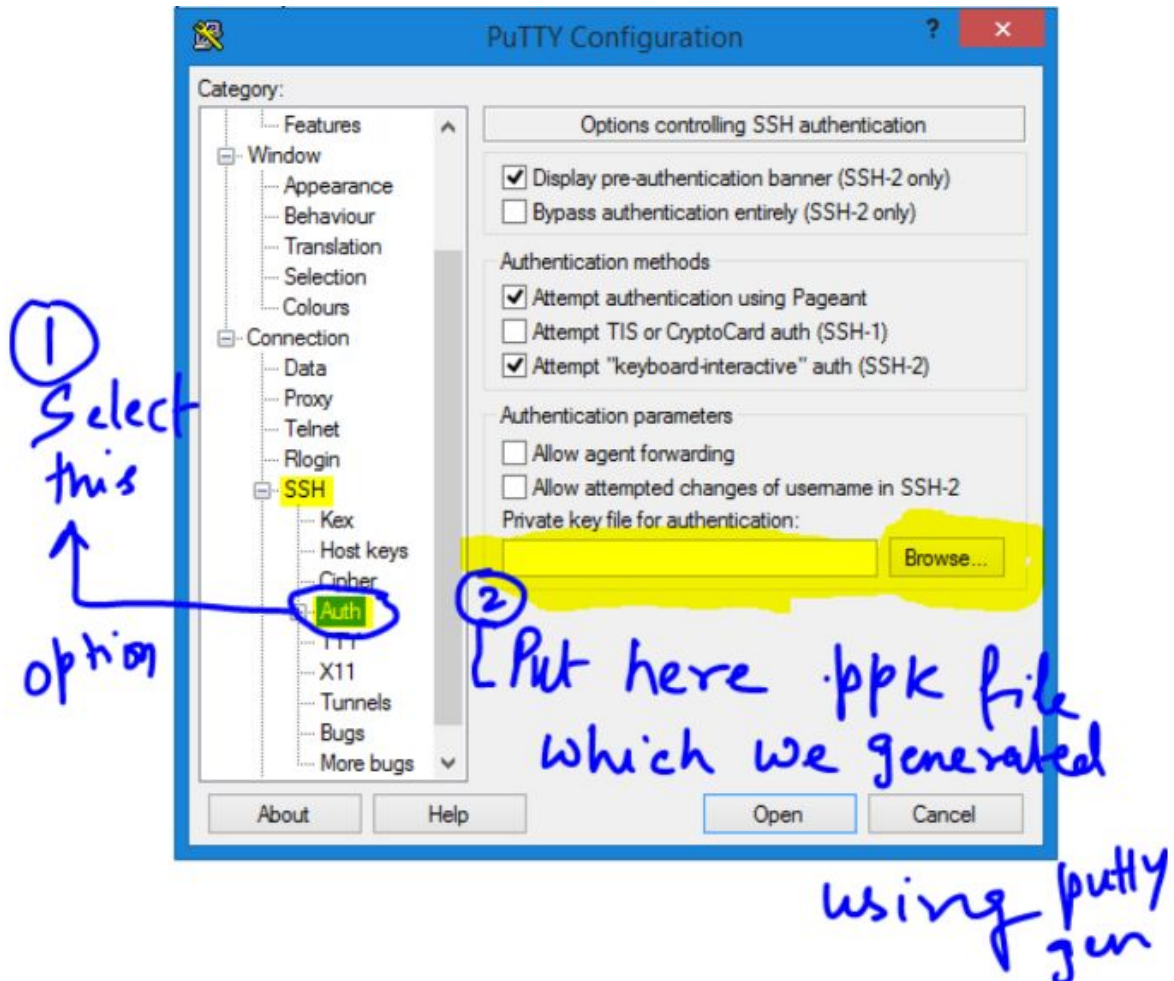
### Section 3 - Connecting with EC2 Instance using Putty & Puttygen

**Step 4.** Setup Putty by entering the hostname (i.e the url of your instance). You can take a look at the following image and see how you can extract the hostname (or url) of the instance.



### Section 3 - Connecting with EC2 Instance using Putty & Puttygen

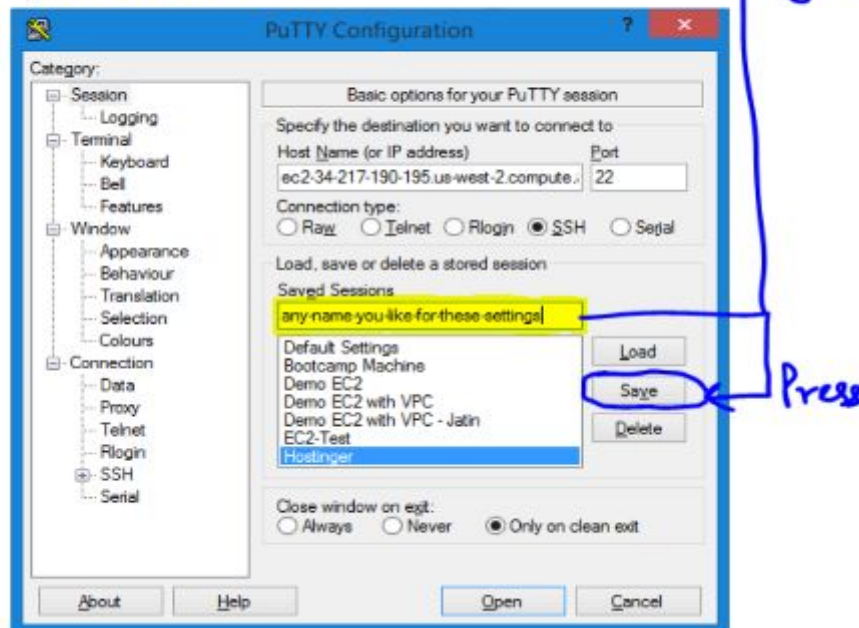
**Step 5.** Set the .ppk key in putty in the field shown below.



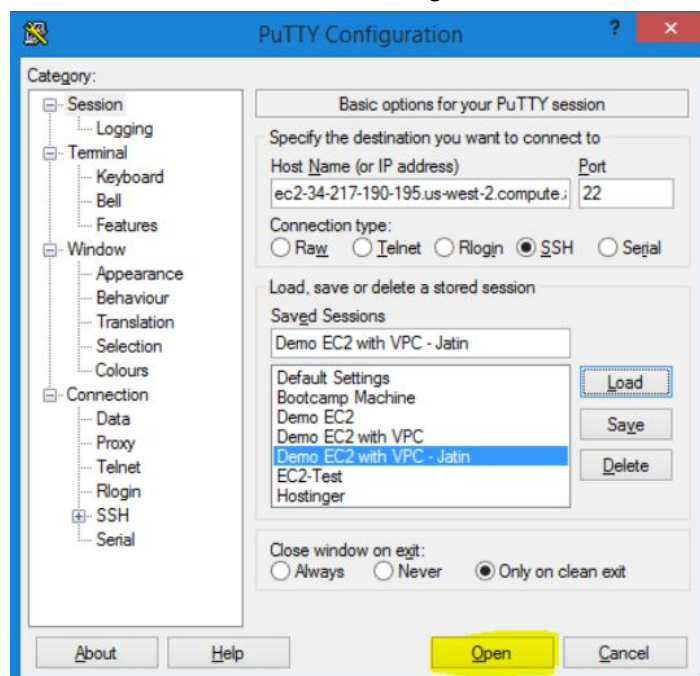
### Section 3 - Connecting with EC2 Instance using Putty & Puttygen

**Step 6.** Save these settings in putty, so as to avoid this procedure again and again. (Note: Once you save your settings they will be available under saved sessions and you can select and load them)

Save settings with a name  
so you don't have to do this again



**Step 7.** Open the Connection to check if it's working fine or not.



# How to set up Filezilla for File Transfer

---

**Step 1.** Add new site in site manager

**Step 2.** Add 'Amazon AWS key (.pem)' in Edit->Settings->SFTP

**Step 3.** Connect to the new site added using the site manager

Please watch the Filezilla-Setup.gif from the following url for step by step approach on how to connect to an ec2 machine in filezilla for file transfer

<https://bit.ly/dg-bootcamp-filezilla-setup>





## Section 5 - Apache Server Configuration on EC2 Machine

---

# How to configure Apache server on EC2

---

**Step 1.** Install apache server using the following command

```
sudo apt-get install apache2
```

**Step 2.** Install apache webserver gateway interface using the following command

```
Sudo apt-get install libapache2-mod-wsgi
```

Visit <https://<your-ec2-hostname-url>.com> to see if everything is working fine. If you installed it correctly you will see a apache welcome page.



## Section 6 - Setting up the flask app on EC2 Instance

---

# How to set up the Flask app on EC2 Instance

---

**Step 1.** Open the terminal & create the following directory

```
> /var/www/FlaskApplications
```

**Step 2.** Now create the following directory

```
> /var/www/FlaskApplications/SampleApp
```

**Step 3.** Change the permissions of both the above directories using the following commands

1. `sudo chown -R ubuntu:ubuntu /path/to/directory`
2. `sudo chmod -R 777 /path/to/directory`

NOTE: `/path/to/directory` (obviously ;) ) refers to the path of directory/file of which you want to change the permissions. We change the permissions so that we and apache (and in return the internet) are able to write at that location.

**Step 4.** Change the hostname in `SampleApp.conf` to match your EC2 Instance Hostname (or url)

**Step 5.** Place the `SampleApp.conf` file at

```
> /etc/apache2/sites-available/SampleApp.conf
```

**Step 6.** Place the `.wsgi` file at

```
> /var/www/FlaskApplications/
```

**Step 7.** Place the `demo.py` file at

```
> /var/www/FlaskApplications/SampleApp/api
```

**Step 8.** Execute the following command to initialize the apache server and the api

1. `sudo a2enmod wsgi`
2. `sudo apachectl restart`
3. `sudo a2ensite sampleApp`



## Section 6 - Setting up the flask app on EC2 Instance

---

**Step 9.** Execute the following command to start the server

1. `sudo service apache2 reload`
2. `sudo /etc/init.d/apache2 reload`