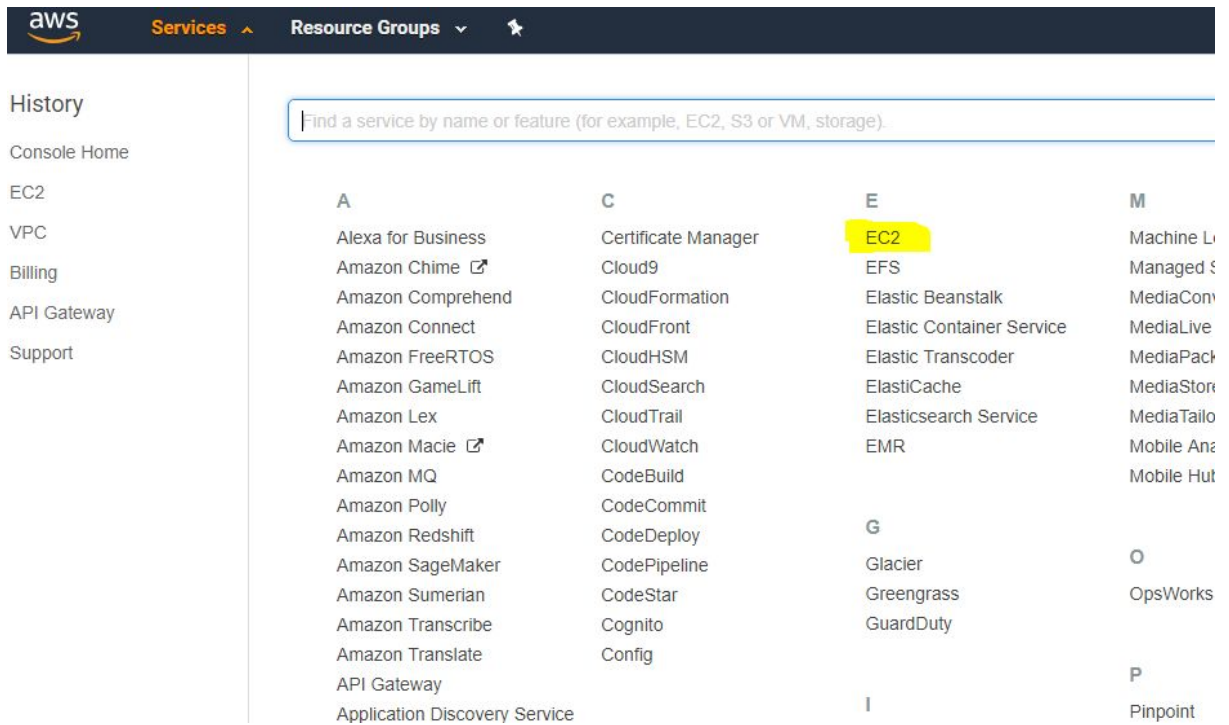


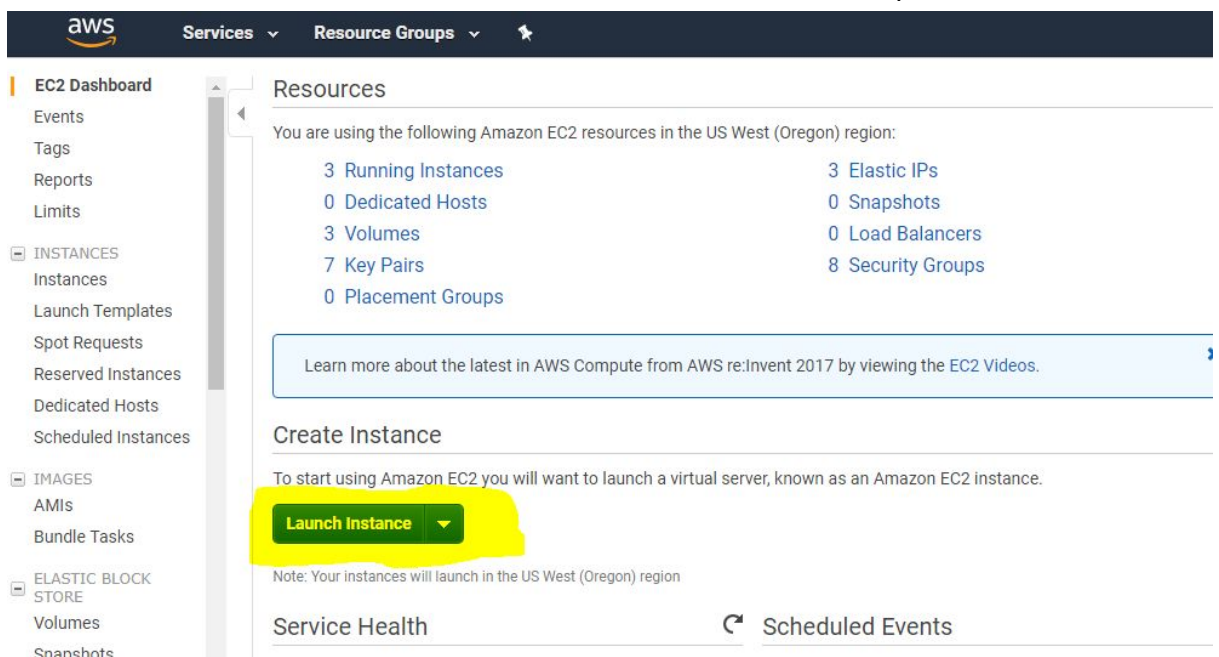
AWS and API Setup

Step 1 Create EC2 Machine

1. Select the EC2 Option from the services dropdown on your aws dashboard



2. On the next screen of EC2 Dashboard, Select the Launch Instance option




3. In this bootcamp we are using UBUNTU Server 16.04 Machine.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review


Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

**Red Hat**
Free tier eligible


Red Hat Enterprise Linux 7.4 (HVM), SSD Volume Type - ami-223f945a
Red Hat Enterprise Linux version 7.4 (HVM), EBS General Purpose (SSD) Volume Type
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Select
64-bit

**Ubuntu**
Free tier eligible

Ubuntu Server 16.04 LTS (HVM), SSD Volume Type - ami-4e79ed36
Ubuntu Server 16.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Select
64-bit

**Amazon RDS**

Are you launching a database instance? Try Amazon RDS.
Amazon Relational Database Service (RDS) makes it easy to set up, operate, and scale your database on AWS by automating time-consuming database management tasks. With RDS, you can easily deploy **Amazon Aurora, MariaDB, MySQL, Oracle, PostgreSQL, and SQL Server** databases on AWS. Aurora is a MySQL- and PostgreSQL-compatible, enterprise-class database at 1/10th the cost of commercial databases. [Learn more about RDS](#)
[Launch a database using RDS](#)

Hide

4. Now select the instance type - T2.Micro (It comes under the free tier label)

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance types Current generation Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GiB)	EBS-Optimized Available	Network Performance	IPv6 Support
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.large	2	8	EBS only	-	Low to Moderate	Yes

5. Enable auto assign IP – to open access from Internet

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing instance, and more.

Number of instances [Launch into Auto Scaling Group](#)

Purchasing option ☐ Request Spot instances

Network [Create new VPC](#)

Subnet [Create new subnet](#)

Auto-assign Public IP

IAM role [Create new IAM role](#)

Shutdown behavior

Enable termination protection ☐ Protect against accidental termination

Monitoring ☐ Enable CloudWatch detailed monitoring
[Additional charges apply.](#)

6. Set the default settings

7. Set the default settings

8. Configure the security groups as shown below

9. Save the .pem key to access the ec2 machine and then launch.

Step 2 Connect to instance using Putty

1. Convert key to .ppk from .pem
 - a. Load in puttyGen

b. Save Private Key

c. Save Without Paraphrase

2. Setup Putty

- a. Enter the Hostname to the specified field. You will get the hostname from AWS EC2 Dashboard as shown below in the figure.

b. Set the key in Putty in the field shown below

c. Save Settings for this, 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.)

- d. Open Connection

Step 3 Install Python and its dependencies

1) Install Python

- a. `sudo apt-get install python 2.7`

2) Install development python dependencies

- a. `sudo apt-get install python-setuptools`

Note : Run `sudo apt-get update` if you face any error like "Failed to fetch the resource"

- b. `sudo apt-get install python-dev`
- c. `sudo apt-get install build essential`
- d. `sudo apt-get install python-pip`

3) Install development related libraries

- a. `pip install numpy scipy sklearn`
- b. `sudo apt-get install python-pil`

- c. `sudo apt-get install python-joblib`
- d. `sudo apt-get install python-flask`
- e. To install opencv use this command
`curl -s "https://raw.githubusercontent.com/arthurbeggs/scripts/master/install_apps/install_opencv2.sh" | bash`

Step 4 Setup File Zilla for File Transfer

- 1) Add new site in site manager
- 2) Add 'Amazon AWS key(.pem)' in Edit->Settings->SFTP

Step 5 Configure Apache server on EC2

- 1) `sudo apt-get install apache2`
- 2) `sudo apt-get install libapache2-mod-wsgi`

Run "curl" => to => TEST => Everything is working

Step 6 Setting up the Flask app

- 1) Create Directory:
 - a. `/var/www/FlaskApplications`
- 2) Create another directory:
 - a. `/var/www/Flaskapplications/SampleApp`
- 3) Change Permissions:
 - a. `sudo chown -R ubuntu:ubuntu /path/to`
 - b. `sudo chown -R 755 /path/to`
- 4) Place the .conf file at
 - a. `/etc/apache2/sites-available/SampleApp.conf`
 - b. Change – hostname
- 5) Place the .wsgi file at
 - a. `/var/www/FlaskApplications/`
- 6) To test the setup, Place 'demo.py' file in `/var/www/Flaskapplications/sampleApp/api/`

7) Run

- a. `sudo a2enmod wsgi`
- b. `sudo apachectl restart`
- c. `sudo a2ensite sampleApp`

8) Run

- a. `sudo service apache2 reload`
- b. `sudo /etc/init.d/apache2 reload`
- c. `sudo service apache2 restart`
- d. `sudo /etc/init.d/apache2 reload`