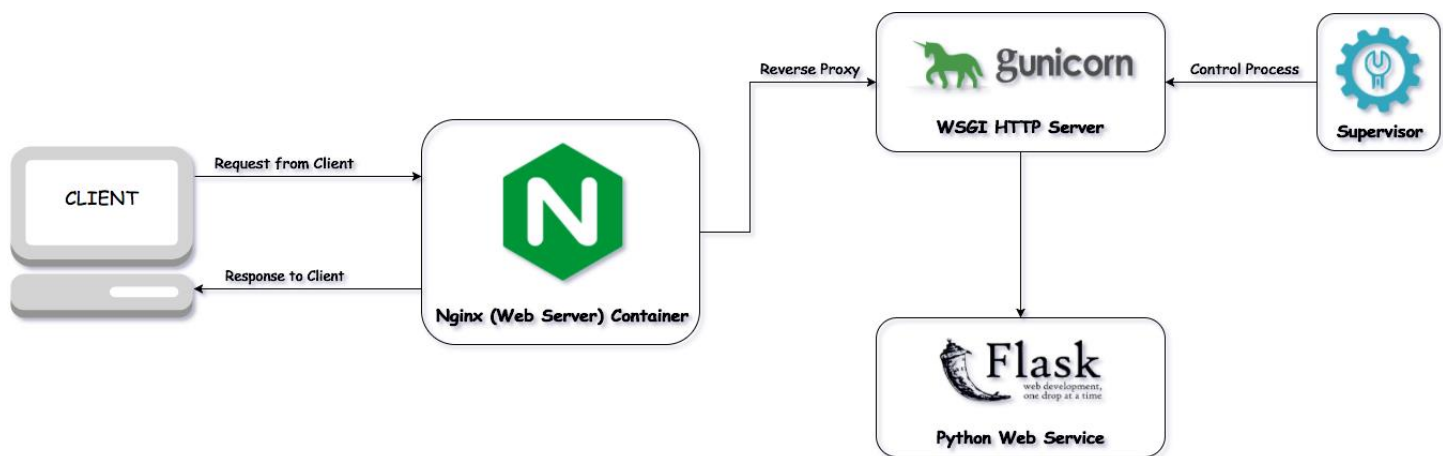


# End to End ML Project 1 - P6 - Deployment in AWS EC2 with NGINX, Guinicorn, Supervisor

## Architecture of Deployment

We will be using the following technologies:

- ➔ Nginx: Reverse proxy, web server
- ➔ Flask: Server backend
- ➔ Gunicorn: To run flask app
- ➔ Supervisor: Monitor and control gunicorn process



Here is the architecture of this deployment, where client could be web browser or mobile device etc. NGINX as the web server and reverse proxy. This means that NGINX will sit between your Flask application and external clients and forward all client requests to your running Flask application. Gunicorn (Green Unicorn), is a Python web server gateway interface (WSGI) HTTP Server for UNIX. It will be used to forward requests from your NGINX web server to your Flask application and finally Supervisor is a client/server system that allows its users to monitor and control a number of processes on UNIX-like operating systems. Supervisor can handle auto-reloading Gunicorn if it crashes or if your server is rebooted unexpectedly

## Starting up an EC2 instance

1. Login to the AWS console here  
<https://aws.amazon.com/console/>
  - ➔ Select EC2 from AWS services.
  - ➔ Click on Launch Instance.
  - ➔ Choose Amazon Machine Image
    - Select the Ubuntu Server 18.04 LTS (HVM), SSD Volume Type.
    - The exact versions may change with time.

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

**SUSE Linux**  
Free tier eligible

Ubuntu1365 / 63d34253 (64-bit Arm)

SUSE Linux Enterprise Server 15 Service Pack 1 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

**Ubuntu Server 18.04 LTS (HVM), SSD Volume Type**  
Free tier eligible

- ami-0b44050b2d893d5f7 (64-bit x86) / ami-0dc231c09aef801ea (64-bit Arm)

Ubuntu Server 18.04 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 (x86)  
☐ 64-bit (Arm)

## → Choose the instance type

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 (GB)	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

Cancel Previous **Review and Launch** Next: Configure Instance Details

## → Configure the Instance

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, assign an access management role to the 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

Placement group ☐ Add instance to placement group

Capacity Reservation  [Create new Capacity Reservation](#)

IAM role  [Create new IAM role](#)

Cancel Previous **Review and Launch** Next: Add Storage

## → Add Storage

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

## Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

Volume Type <sup>i</sup>	Device <sup>i</sup>	Snapshot <sup>i</sup>	Size (GiB) <sup>i</sup>	Volume Type <sup>i</sup>	IOPS <sup>i</sup>	Throughput (MB/s) <sup>i</sup>	Delete on Termination <sup>i</sup>	Encryption <sup>i</sup>
Root	/dev/sda1	snap-0ae07ad9c9401fae6	8	General Purpose SSD (gp2)	100 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypt <sup>▼</sup>
Add New Volume								

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more](#) about free usage tier eligibility and usage restrictions.

Cancel Previous **Review and Launch** Next: Add Tags

## → Add Tags

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

## Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.

A copy of a tag can be applied to volumes, instances or both.

Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

Key <sup>(128 characters maximum)</sup>	Value <sup>(256 characters maximum)</sup>	Instances <sup>i</sup>	Volumes <sup>i</sup>
This resource currently has no tags			
Choose the Add tag button or <a href="#">click to add a Name tag</a> .			
Make sure your <a href="#">IAM policy</a> includes permissions to create tags.			
Add Tag <sup>(Up to 50 tags maximum)</sup>			

Cancel Previous **Review and Launch** Next: Configure Security Group

## → Configure the Security Group-> add HTTP and HTTPS

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

## 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 and 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: launch-wizard-2

Description: launch-wizard-2 created 2020-06-12T13:29:05.448+03:00

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Custom 0.0.0.0/0, :::0	e.g. SSH for Admin Desktop
HTTPS	TCP	443	Custom 0.0.0.0/0, :::0	e.g. SSH for Admin Desktop

Add Rule

Cancel Previous Review and Launch

## ➔ Review Instance Launch

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

## Step 7: Review Instance Launch

### AMI Details

Edit AMI



Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0b44050b2d893d5f7

Free tier  
eligible

Ubuntu Server 18.04 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

### Instance Type

Edit instance type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

### Security Groups

Edit security groups

Security group name launch-wizard-2  
Description launch-wizard-2 created 2020-06-12T13:29:05.448+03:00

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	0.0.0.0/0	
HTTP	TCP	80	0.0.0.0/0	

Cancel Previous Launch

## ➔ Create new key pair

## 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

bcheekati\_mlkey

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.

Cancel

Launch Instances

➔ Download the Key Pair

Local Disk (E:) <

File name: bcheekati\_mlkey

Save as type: PEM File

Folders

Save Cancel

bcheekati\_mlkey

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.

Cancel

Launch Instances

➔ Finally click on Launch Instances button

EC2 Dashboard **New**

Events **New**

Tags

Reports

**Launch Instance** **Connect** **Actions**

Filter by tags and attributes or search by keyword

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
	i-0949b839b765996...	t2.micro	ap-south-1b	running	Initializing	None	ec2-13-232-126-85.ap-...

➔ Update the name of instance.

aws Services Resource Groups

SCANCOST Mumbai Support

EC2 Dashboard **New**

Events **New**

Tags

Reports

**Launch Instance** **Connect** **Actions**

Filter by tags and attributes or search by keyword

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
BCHEEKATI_ML_VM2	i-041b7bbb9dc64a9e2	t2.micro	ap-south-1a	running	Initializing	None	ec2-13-233-245-6.ap-...

➔ Note down the instance public ip, public DNS name and User

Public IP: 13.233.245.6

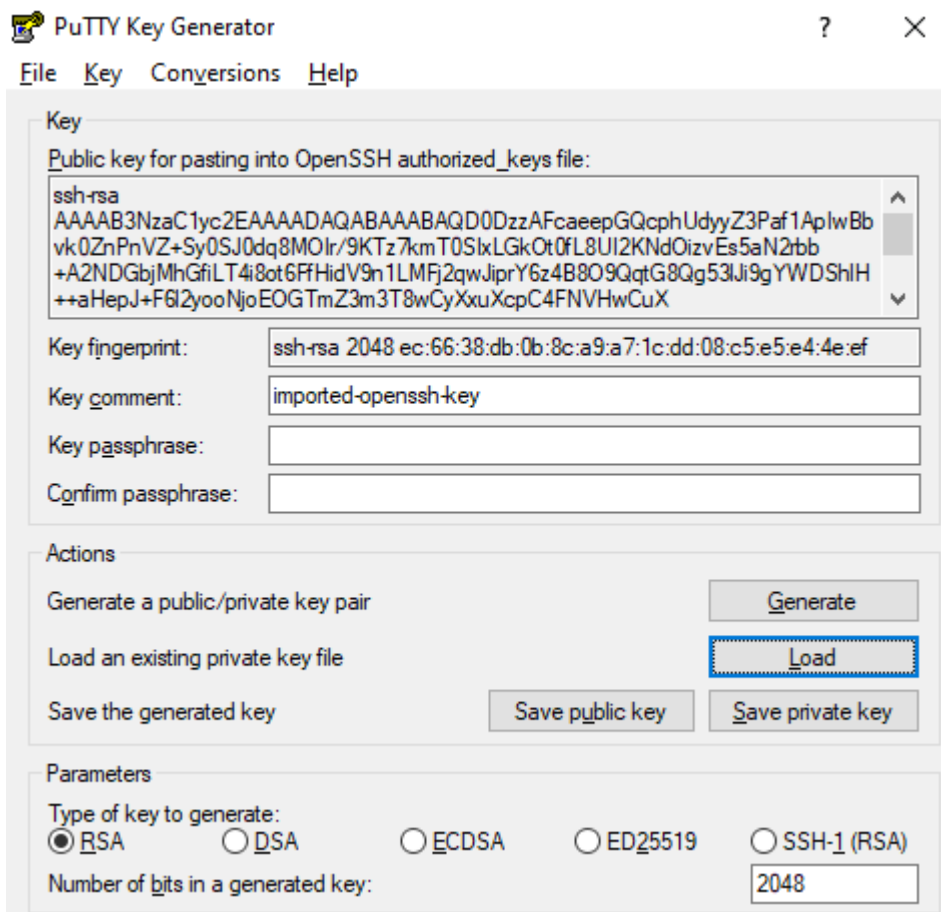
Public DNS name: ec2-13-233-245-6.ap-south-1.compute.amazonaws.com

User: ubuntu

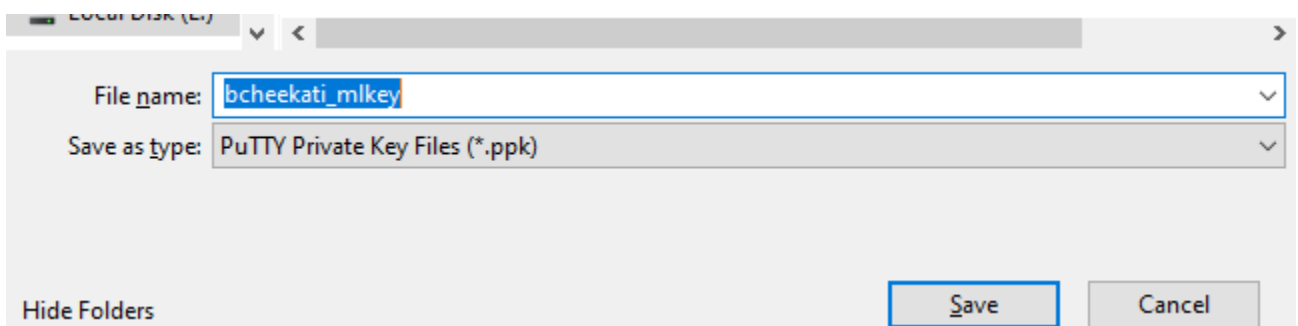
```
ssh -i "bcheekati_mlkey.pem" ubuntu@ec2-13-233-245-6.ap-south-1.compute.amazonaws.com
```

➔ Download putty, puttygen and winscp

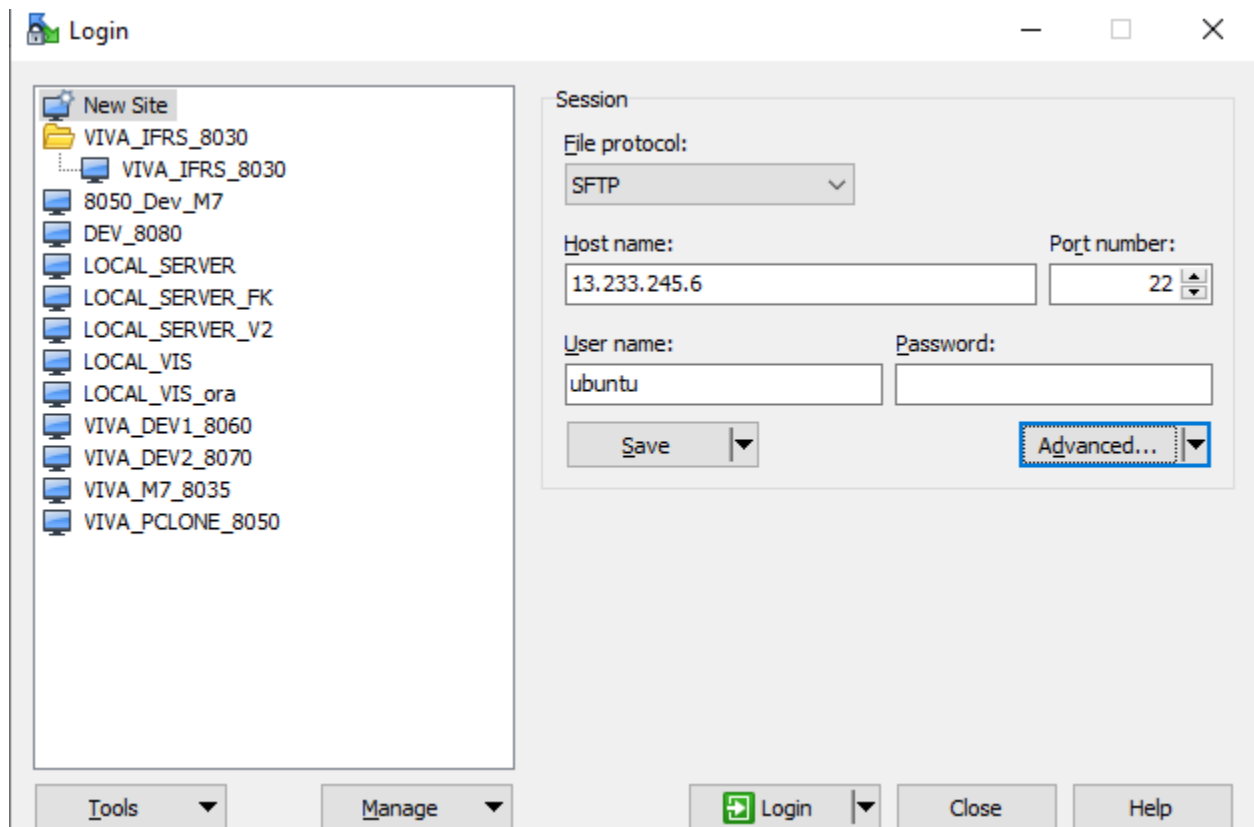
➔ Load PEM private key file (bcheekati\_mlkey.pem) using puttygen to generate private key in ppk format



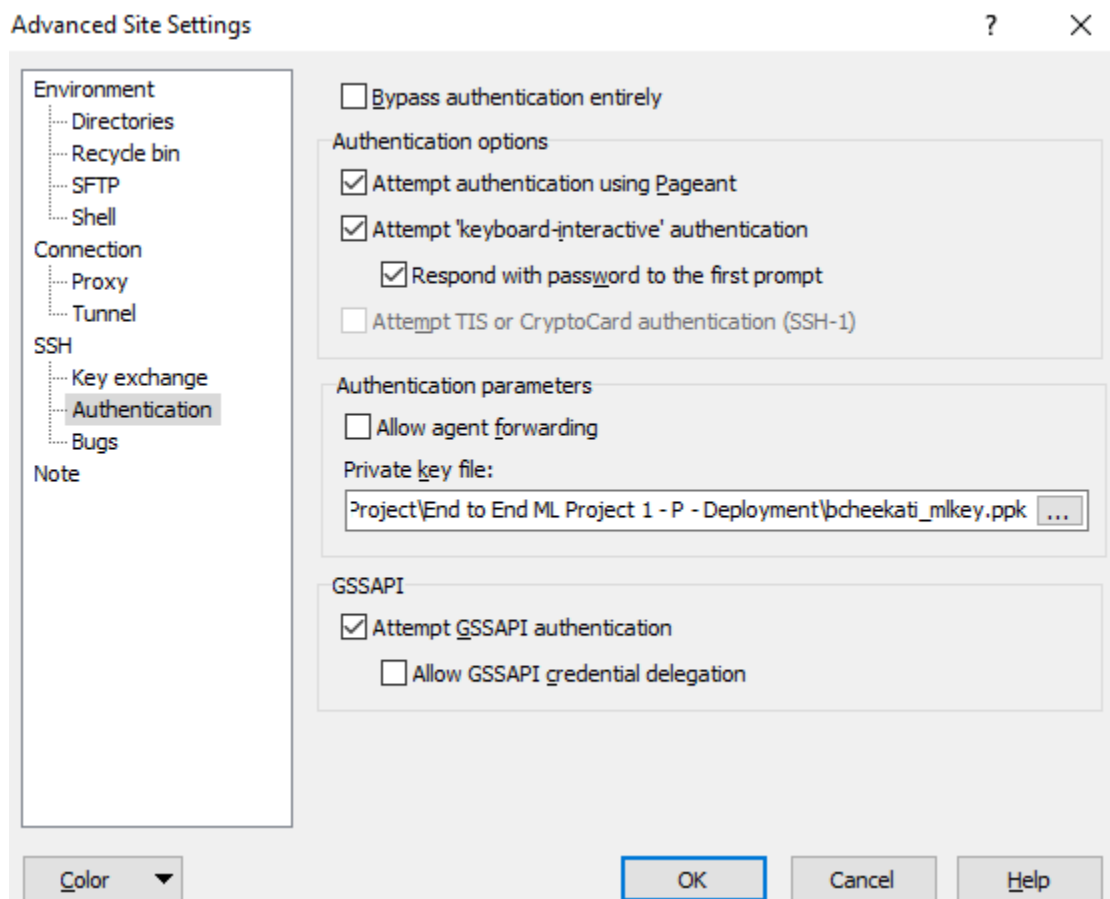
- ➔ click on Save private key to save the private key file in ppk format. Private key will help to connect server from third party tools like putty and winscp without using password



2. Connect WINSCP tool to upload the code files into server  
Enter host name and port and user name.



➔ Click on Advanced -> SSH -> Authentication. and browse private key file (bcheekati\_mlkey.ppk)





➔ Finally click on Login Button to connect server

C:\Users\Admin\PycharmProjects\EndtoEndML_v11\				/home/ubuntu/			
Name	Size	Type	Changed	Name	Size	Changed	Rights
..		Parent directory	6/14/2020 11:39:30 P	..		6/16/2020 6:35:51 PM	rw-r--r--
.idea		File folder	6/14/2020 11:49:35 P				
templates		File folder	6/14/2020 11:28:13 P				
static		File folder	6/14/2020 10:56:33 P				
apps		File folder	6/3/2020 2:43:22 AM				
data		File folder	6/3/2020 12:53:42 AM				
logs		File folder	6/3/2020 12:53:41 AM				
venv		File folder	6/2/2020 10:50:04 PM				
flask_monitoringdashboard.db	44 KB	Data Base File	6/14/2020 11:39:29 P				
main.py	6 KB	Python File	6/14/2020 11:36:27 P				
requirements.txt	2 KB	Text Document	6/2/2020 10:57:58 PM				

➔ Create new folder in /home/ubuntu/ directory to upload all code files

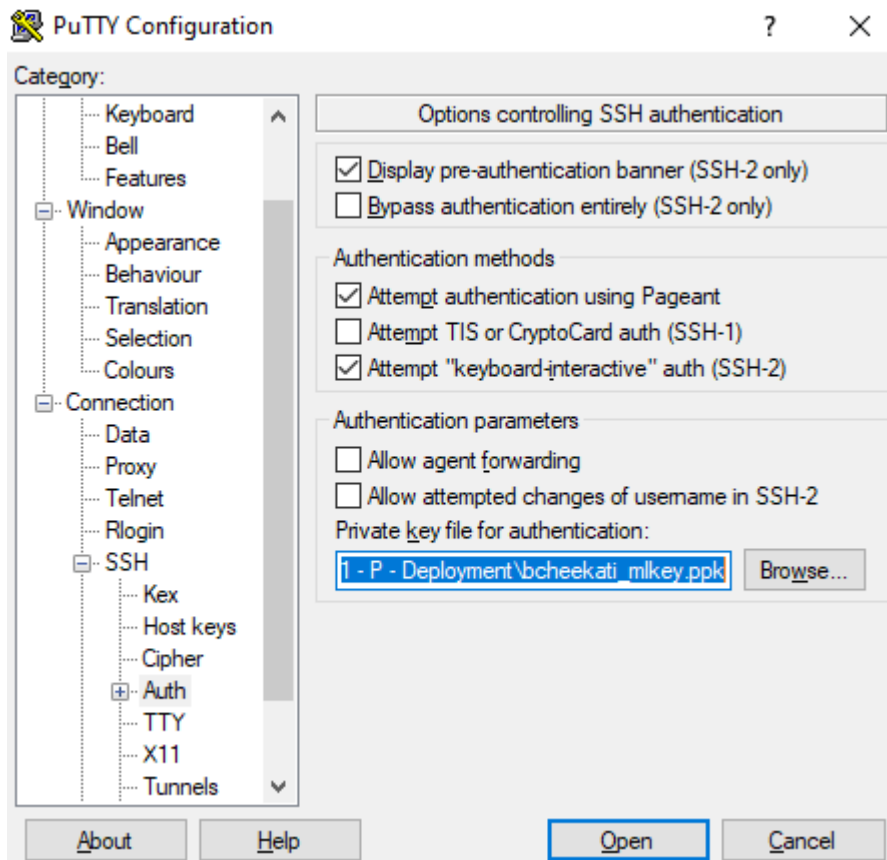
C:\Users\Admin\PycharmProjects\EndtoEndML_v11\				/home/ubuntu/			
Name	Size	Type	Changed	Name	Size	Changed	Rights
..		Parent directory	6/17/2020 3:18:16 PM	..		6/17/2020 1:28:36 PM	rw-r--r--
.idea		File folder	6/14/2020 11:49:35 P	endtoend_ml_v1		6/17/2020 3:07:00 PM	rw-r--r--
templates		File folder	6/14/2020 11:28:13 P				
static		File folder	6/14/2020 10:56:33 P				
apps		File folder	6/3/2020 2:43:22 AM				
data		File folder	6/3/2020 12:53:42 AM				
logs		File folder	6/3/2020 12:53:41 AM				
venv		File folder	6/2/2020 10:50:04 PM				
main.py	6 KB	Python File	6/17/2020 1:27:19 AM				
endtoend_v1.err.log	1 KB	Text Document	6/16/2020 9:19:21 PM				
flask_monitoringdashboard.db	44 KB	Data Base File	6/14/2020 11:39:29 P				
requirements.txt	2 KB	Text Document	6/2/2020 10:57:58 PM				

➔ Upload required code files into server

C:\Users\Admin\PycharmProjects\EndtoEndML_v11\*.py				/home/ubuntu/endtoend_ml_v1/			
Name	Size	Type	Changed	Name	Size	Changed	Rights
..		Parent directory	6/17/2020 3:18:59 PM	..		6/17/2020 2:30:18 PM	rw-r--r--
.idea		File folder	6/14/2020 11:49:35 P	apps		6/17/2020 1:46:26 PM	rw-r--r--
templates		File folder	6/14/2020 11:28:13 P	data		6/17/2020 1:46:35 PM	rw-r--r--
static		File folder	6/14/2020 10:56:33 P	logs		6/17/2020 2:30:52 PM	rw-r--r--
apps		File folder	6/3/2020 2:43:22 AM	static		6/17/2020 1:46:12 PM	rw-r--r--
data		File folder	6/3/2020 12:53:42 AM	templates		6/17/2020 1:46:10 PM	rw-r--r--
logs		File folder	6/3/2020 12:53:41 AM	main.py	6 KB	6/17/2020 1:27:19 AM	rw-r--r--
venv		File folder	6/2/2020 10:50:04 PM	requirements.txt	2 KB	6/2/2020 10:57:58 PM	rw-r--r--
main.py	6 KB	Python File	6/17/2020 1:27:19 AM				
flask_monitoringdashboard.db	44 KB	Data Base File	6/14/2020 11:39:29 P				
requirements.txt	2 KB	Text Document	6/2/2020 10:57:58 PM				

3. Connect putty to install webserver called NGINX, Flask Web framework and all the required libraries

➔ Upload private to connect server without password. Go to connection -> SSH -> Auth



```

ubuntu@ip-172-31-40-150: ~
Using username "ubuntu".
Authenticating with public key "imported-openssh-key"
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-1065-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue Jun 16 15:49:44 UTC 2020

System load:  0.0          Processes:    89
Usage of /:   14.0% of 7.69GB Users logged in:  0
Memory usage: 15%         IP address for eth0: 172.31.40.150
Swap usage:   0%

0 packages can be updated.
0 updates are security updates.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-40-150:~$

```

4. Run below command to upgrade pip install utility in Ubuntu server before installing all required libraries

➔ In your home directory, install Python 3:

`sudo apt install python3`

➔ Install pip3, the standard package manager for Python

`sudo apt-get update && sudo apt-get install python3-pip`

## 5. Install nginx web server by running below command

```
sudo apt-get install nginx
```

```
ubuntu@ip-172-31-40-150:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-40-150:~$ sudo apt-get install nginx
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  fontconfig-config fonts-dejavu-core libfontconfig1 libgd3 libjpeg-turbo8 libjpeg8 libnginx-mod-http-geoip libnginx-mod-http-image-filter
  libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream libtiff5 libwebp6 libxpm4 nginx-common nginx-core
Suggested packages:
  libgd-tools fcgiwrap nginx-doc ssl-cert
The following NEW packages will be installed:
  fontconfig-config fonts-dejavu-core libfontconfig1 libgd3 libjpeg-turbo8 libjpeg8 libnginx-mod-http-geoip libnginx-mod-http-image-filter
  libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream libtiff5 libwebp6 libxpm4 nginx nginx-common nginx-core
0 upgraded, 18 newly installed, 0 to remove and 64 not upgraded.
Need to get 2462 kB of archives.
After this operation, 8210 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libjpeg-turbo8 amd64 1.5.2-0ubuntu5.18.04.4 [110 kB]
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 fonts-dejavu-core all 2.37-1 [1041 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 fontconfig-config all 2.12.6-0ubuntu2 [55.8 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libfontconfig1 amd64 2.12.6-0ubuntu2 [127 kB]
```

Above will install nginx as well as run it.

Check status of nginx using

```
sudo service nginx status
```

Here are the commands to start/stop/restart nginx

```
sudo service nginx start
```

```
sudo service nginx stop
```

```
sudo service nginx restart
```

➔ Check the website by hitting public ip or with public DNS. It displays default page of nginx sever

<http://13.233.245.6/>



➔ remove the default page by deleting the default file to redirect our custom index.html page

```
sudo rm /etc/nginx/sites-enabled/default
```

➔ create a new config file in the sites-available folder and create a symbolic link to it in the sites-enabled folder.

```
sudo vim /etc/nginx/sites-available/endtoend_v1.conf
```

```
server {
    listen 80;
```

```

server_name 13.233.245.6;

root /home/ubuntu/endtoend_v1;

access_log /home/ubuntu/endtoend_v1/logs/nginx_log/access.log;
error_log /home/ubuntu/endtoend_v1/logs/nginx_log/error.log;

location / {
    proxy_set_header X-Forward-For $proxy_add_x_forwarded_for;
    proxy_set_header Host $http_host;
    proxy_redirect off;
    if (!-f $request_filename) {
        proxy_pass http://127.0.0.1:8000;
        break;
    }
}

location /static {
    alias /home/ubuntu/endtoend_v1/static/;
    autoindex on;
}
}

```

:wq to save and exit

- ➔ We have to create the directory for our nginx logs  
`mkdir -p ~endtoend_v1/logs/nginx_log`

- ➔ Create symlink for this file in /etc/nginx/sites-enabled by running this command,

```
sudo ln -s /etc/nginx/sites-available/endtoend_v1.conf /etc/nginx/sites-enabled/endtoend_v1.conf
```

- ➔ Restart nginx server  
`sudo service nginx restart`

## 6. Create virtual environment

```
sudo apt install virtualenv
```

- ➔ create a virtual environment and activate

```

cd /home/ubuntu/endtoend_v1
mkdir /home/ubuntu/endtoend_v1/.virtualenvs && cd /home/ubuntu/endtoend_v1/.virtualenvs
virtualenv -p python3 endtoend_v1_venv

```

```
ubuntu@ip-172-31-40-150:~$ cd /home/ubuntu/endtoend_v1
ubuntu@ip-172-31-40-150:~/endtoend_v1$ mkdir /home/ubuntu/endtoend_v1/.virtualenvs
ubuntu@ip-172-31-40-150:~/endtoend_v1$ ls
__pycache__  apps  data  flask_monitoringdashboard.db  logs  main.py  requirements.txt  static  templates
ubuntu@ip-172-31-40-150:~/endtoend_v1$ ls -aa
.  ..  .virtualenvs  __pycache__  apps  data  flask_monitoringdashboard.db  logs  main.py  requirements.txt  static  templates
ubuntu@ip-172-31-40-150:~/endtoend_v1$ cd /home/ubuntu/endtoend_v1/.virtualenvs
ubuntu@ip-172-31-40-150:~/endtoend_v1/.virtualenvs$ virtualenv -p python3 endtoend_v1_venv
Already using interpreter /usr/bin/python3
Using base prefix '/usr'
New python executable in /home/ubuntu/endtoend_v1/.virtualenvs/endtoend_v1_venv/bin/python3
Also creating executable in /home/ubuntu/endtoend_v1/.virtualenvs/endtoend_v1_venv/bin/python
Installing setuptools, pkg_resources, pip, wheel...done.
```

➔ Activate the virtual env

source /home/ubuntu/endtoend\_v1/.virtualenvs/endtoend\_v1\_venv/bin/activate

```
ubuntu@ip-172-31-40-150:~/endtoend_v1/.virtualenvs$ source /home/ubuntu/endtoend_v1/.virtualenvs/endtoend_v1_venv/bin/activate
(endtoend_v1_venv) ubuntu@ip-172-31-40-150:~/endtoend_v1/.virtualenvs$
```

7. Install dependencies using requirement.txt. run below command

pip3 install -r /home/ubuntu/endtoend\_v1/requirements.txt

```
(endtoend_v1_venv) ubuntu@ip-172-31-40-150:~/endtoend_v1/.virtualenvs$ pwd
/home/ubuntu/endtoend_v1/.virtualenvs
(endtoend_v1_venv) ubuntu@ip-172-31-40-150:~/endtoend_v1/.virtualenvs$ pip3 install -r /home/ubuntu/endtoend_v1/requirements.txt
Collecting Flask==1.1.1
  Using cached Flask-1.1.1-py2.py3-none-any.whl (94 kB)
Collecting Flask-Cors==3.0.8
  Using cached Flask-Cors-3.0.8-py2.py3-none-any.whl (14 kB)
Collecting matplotlib==3.1.2
  Using cached matplotlib-3.1.2-cp36-cp36m-manylinux1_x86_64.whl (13.1 MB)
Collecting numpy==1.18.1
  Using cached numpy-1.18.1-cp36-cp36m-manylinux1_x86_64.whl (20.1 MB)
Collecting pandas==0.25.3
  Using cached pandas-0.25.3-cp36-cp36m-manylinux1_x86_64.whl (10.4 MB)
Collecting scikit-learn==0.22.1
  Using cached scikit_learn-0.22.1-cp36-cp36m-manylinux1_x86_64.whl (7.0 MB)
Collecting kneed==0.5.1
  Using cached kneed-0.5.1-py2.py3-none-any.whl (9.9 kB)
Collecting xgboost==1.0.2
  Using cached xgboost-1.0.2-py3-none-manylinux1_x86_64.whl (109.7 MB)
```

8. Install Gunicorn. It act as python WSGI HTTP server for Unix

pip3 install gunicorn

```
(endtoend_v1_venv) ubuntu@ip-172-31-40-150:~/endtoend_v1/.virtualenvs$ pwd
/home/ubuntu/endtoend_v1/.virtualenvs
(endtoend_v1_venv) ubuntu@ip-172-31-40-150:~/endtoend_v1/.virtualenvs$ pip3 install gunicorn
Collecting gunicorn
  Using cached gunicorn-20.0.4-py2.py3-none-any.whl (77 kB)
Requirement already satisfied: setuptools>=3.0 in ./endtoend_v1_venv/lib/python3.6/site-packages (from gunicorn) (47.3.0)
Installing collected packages: gunicorn
Successfully installed gunicorn-20.0.4
(endtoend_v1_venv) ubuntu@ip-172-31-40-150:~/endtoend_v1/.virtualenvs$
```

➔ Let's start a Gunicorn process to serve your Flask app.

cd /home/ubuntu/endtoend\_v1  
gunicorn main:app -w 3

```
(endtoend_v1_venv) ubuntu@ip-172-31-40-150:~/endtoend_v1/.virtualenvs$ cd /home/ubuntu/endtoend_v1
(endtoend_v1_venv) ubuntu@ip-172-31-40-150:~/endtoend_v1$ gunicorn main:app -w 3
[2020-06-16 18:17:21 +0000] [9274] [INFO] Starting gunicorn 20.0.4
[2020-06-16 18:17:21 +0000] [9274] [INFO] Listening at: http://127.0.0.1:8000 (9274)
[2020-06-16 18:17:21 +0000] [9274] [INFO] Using worker: sync
[2020-06-16 18:17:21 +0000] [9277] [INFO] Booting worker with pid: 9277
[2020-06-16 18:17:21 +0000] [9278] [INFO] Booting worker with pid: 9278
[2020-06-16 18:17:21 +0000] [9279] [INFO] Booting worker with pid: 9279
Scheduler started
Scheduler started
Scheduler started
```

This will set your Gunicorn process off running in the background, which will work fine for your purposes here. An improvement that can be made here is to run Gunicorn via Supervisor.

9. Install supervisor lib which Supervisor allows to monitor and control a number of processes on UNIX-like operating systems. Supervisor will look after the Gunicorn process and make sure that they are restarted if anything goes wrong, or to ensure the processes are started at boot time.

```
sudo apt install supervisor
```

```
[2020-06-16 17:31:27 +0000] [8466] [INFO] Shutting down: Master
(endtoend_v1_venv) ubuntu@ip-172-31-40-150:~/endtoend_v1$ sudo apt install supervisor
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  python-meld3
Suggested packages:
  supervisor-doc
The following NEW packages will be installed:
  python-meld3 supervisor
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 287 kB of archives.
After this operation, 1580 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic/universe amd64 python-meld3 all 1.0.2-2 [30.9 kB]
```

- ➔ Create a supervisor file in `/etc/supervisor/conf.d/` and configure it according to your requirements.

```
sudo vim /etc/supervisor/conf.d/endtoend_v1.conf
```

```
[program:endtoend_v1]
directory=/home/ubuntu/endtoend_v1/
command=/home/ubuntu/endtoend_v1/.virtualenvs/endtoend_v1_venv/bin/gunicorn main:app
autostart=true
autorestart=true
stopasgroup=true
killasgroup=true
stderr_logfile=/home/ubuntu/endtoend_v1/logs/supervisor_log/endtoend_v1.err.log
stdout_logfile=/home/ubuntu/endtoend_v1/logs/supervisor_log/endtoend_v1.out.log
```

- ➔ Create the log directories and files listed in the `endtoend_v1.conf` file. Make sure to replace `endtoend_v1` if it was modified in the Supervisor script above:

```
sudo mkdir /home/ubuntu/endtoend_v1/logs/supervisor_log
sudo touch /home/ubuntu/endtoend_v1/logs/supervisor_log/endtoend_v1.err.log
sudo touch /home/ubuntu/endtoend_v1/logs/supervisor_log/endtoend_v1.out.log
```

- ➔ To enable the configuration, run the following commands:

```
sudo supervisorctl reread
sudo supervisorctl update
sudo supervisorctl reload
```

```
additional command
sudo service supervisor restart
```

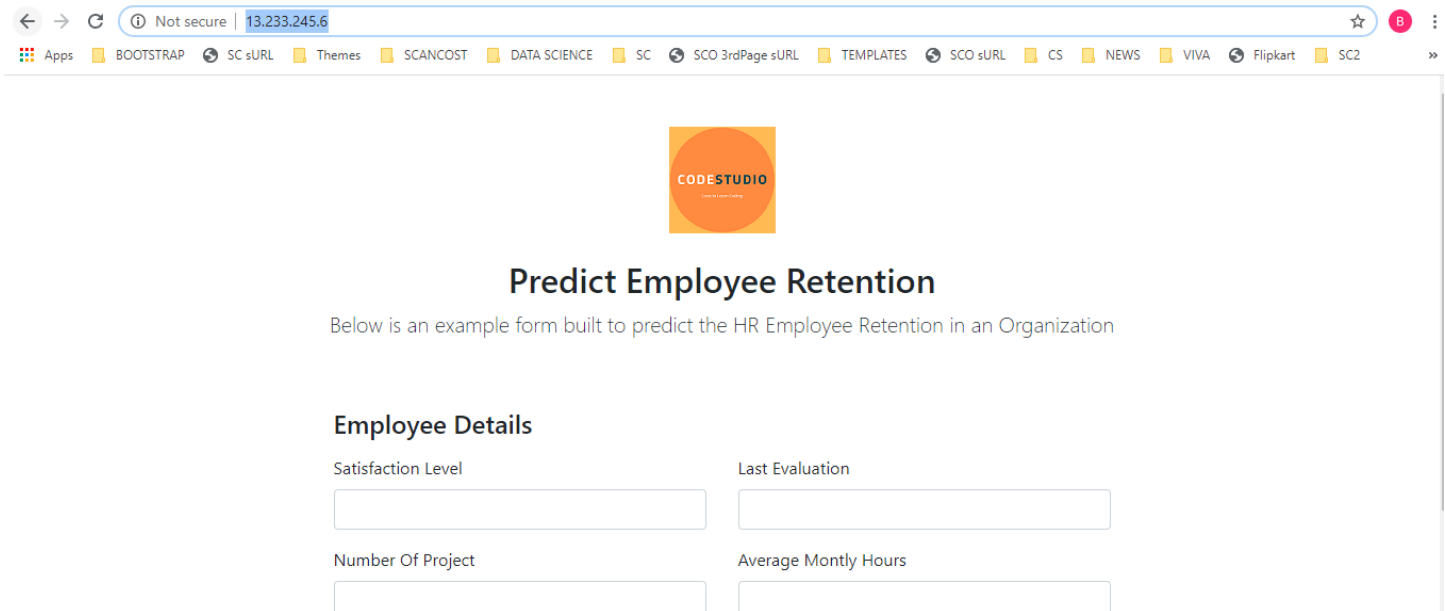
```
sudo service supervisor stop
```

➔ This should start a new process. To check the status of all monitored apps, use the following command:

```
sudo supervisorctl status
```


10. run below URL into browser to see our custom index.html page

<http://13.233.245.6/>



Not secure | 13.233.245.6

Apps BOOTSTRAP SC sURL Themes SCANCOST DATA SCIENCE SC SCO 3rdPage sURL TEMPLATES SCO sURL CS NEWS VIVA Flipkart SC2



## Predict Employee Retention

Below is an example form built to predict the HR Employee Retention in an Organization

### Employee Details

Satisfaction Level	Last Evaluation
<input type="text"/>	<input type="text"/>
Number Of Project	Average Montly Hours
<input type="text"/>	<input type="text"/>