

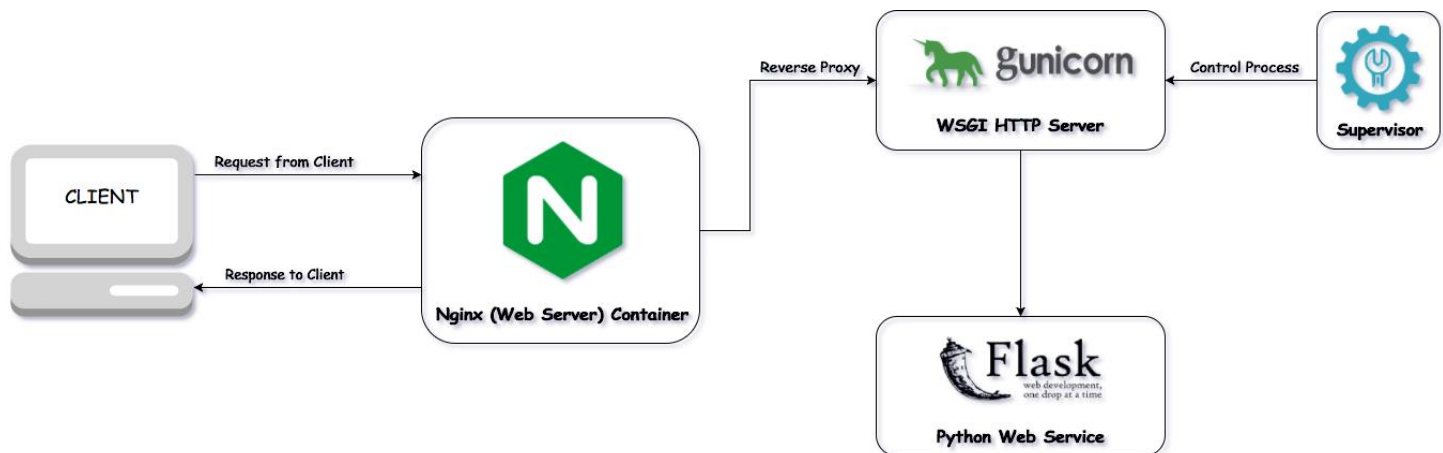
Model Deployment Tutorial 5 - ML Model in Microsoft Azure Cloud (IAAS)

ML Deployment in Azure with NGINX, Guinicorn, Supervisor

Architecture of Deployment

We will be using the following technologies:

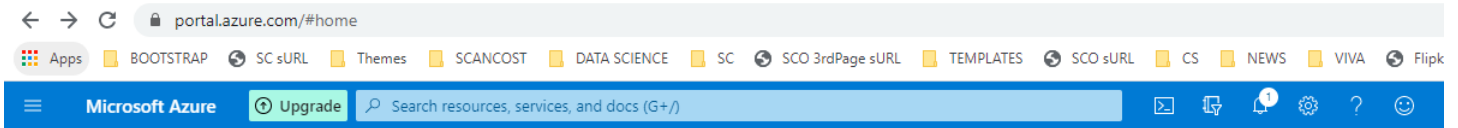
- ➔ Nginx: Reverse proxy, web server
- ➔ Flask: Server backend
- ➔ Gunicorn: To run flask app
- ➔ Supervisor: Monitor and control gunicorn process



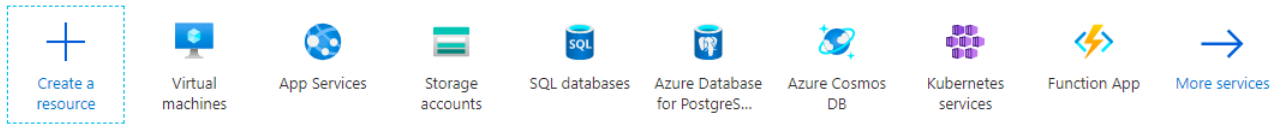
Here is the architecture of this deployment, where client could be web browser or mobile device etc. NGINX as the web server and reverse proxy. This means that NGINX will sit between your Flask application and external clients and forward all client requests to your running Flask application. Gunicorn (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 instance in Microsoft Azure

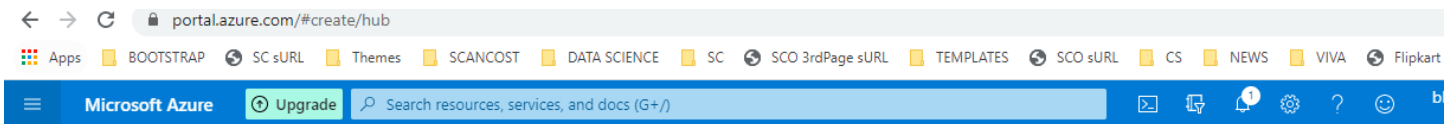
1. Login to the azure console here
<https://portal.azure.com/#home>
- ➔ Create virtual machine by clicking on create a resource



Azure services



➔ Select the Ubuntu server



Home >

New

Azure Marketplace [See all](#)

Popular

Get started

Recently created

AI + Machine Learning

Analytics

**Windows Server 2016 Datacenter**
[Quickstarts + tutorials](#)**Ubuntu Server 18.04 LTS**
[Learn more](#)

➔ Enter required fields

Microsoft Azure

Search resources, services, and docs (G+)

Home > New >

Create a virtual machine

Subscription *

Free Trial

Resource group *

(New) ml_resource_group

Create new

Instance details

Virtual machine name *

mlvm-v1

Region *

(US) East US

Availability options

No infrastructure redundancy required

Image *

Ubuntu Server 18.04 LTS

Browse all public and private images

Azure Spot instance

☐ Yes ☒ No

Size *

Standard_B1s - 1 vcpu, 1 GiB memory (₹501.80/month)

Select size

Review + create

< Previous

Next : Disks >

Microsoft Azure Search resources, services, and docs (G+)

Home > New >

Create a virtual machine

Username * ⓘ mladmin ✓

Password * ⓘ ✓

Confirm password * ⓘ ✓

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports * ⓘ ☐ None ☒ Allow selected ports

Select inbound ports * HTTP (80), HTTPS (443), SSH (22) ✓

⚠ This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

[Review + create](#) < Previous Next : Disks >

Please note down user name and password which to be used to connect server from third party tool such as putty, WinScp.

➔ Click on next button to fill other required fields. And finally review and create the virtual machine

Microsoft Azure Upgrade Search resources, services, and docs (G+)

Home > Virtual machines >

mlvm-v1 Virtual machine

Search (Ctrl+/) << Connect Start Restart Stop Capture Delete Refresh

Overview

- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

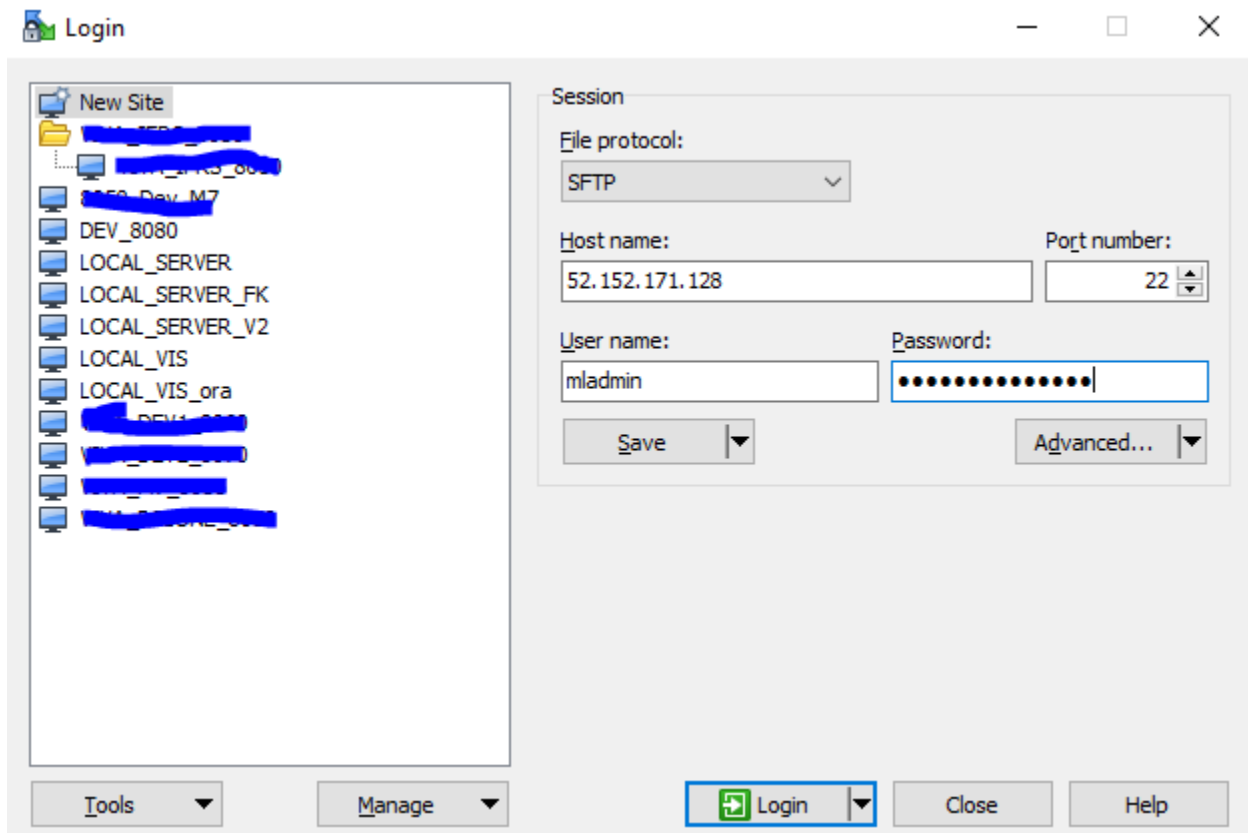
Settings

- Networking
- Connect

| | | | |
|-------------------------|--|---------------------------|----------------------------------|
| Resource group (change) | : ml_resource_group | Azure Spot | : N/A |
| Status | : Running | Public IP address | : 52.152.171.128 |
| Location | : East US | Private IP address | : 10.0.1.4 |
| Subscription (change) | : Free Trial | Public IP address (IPv6) | : - |
| Subscription ID | : 63c91c58-09e2-4fc2-8cda-4c586807cd2d | Private IP address (IPv6) | : - |
| Computer name | : mlvm-v1 | Virtual network/subnet | : ml_resource_group-vnet/default |
| Operating system | : Linux (ubuntu 18.04) | DNS name | : Configure |
| Size | : Standard B1s (1 vcpus, 1 GiB memory) | | |
| Tags (change) | : Click here to add tags | | |

➔ Note down the instance public IP, public DNS name and User
 Public IP: 52.152.171.128
 User: mladmin

2. Connect WINSCP to upload code files



➔ Finally click on Login Button to connect server

Please note that home directory may be different for your deployment.

➔ Create new folder in home 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-xr-x |
| .idea | | File folder | 6/14/2020 11:49:35 P | endtoend_ml_v1 | | 6/17/2020 3:07:00 PM | rw-r-xr-x |
| 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\ | | | | /home/ubuntu/ | | | |
|--|------|------------------|----------------------|------------------|------|----------------------|-----------|
| Name | Size | Type | Changed | Name | Size | Changed | Rights |
| .. | | Parent directory | 7/10/2020 12:44:47 P | .. | | 7/10/2020 2:44:05 PM | rw-r-xr-x |
| .idea | | File folder | 6/28/2020 6:13:30 PM | models | | 7/10/2020 2:57:31 PM | rw-rw-r-x |
| models | | File folder | 6/25/2020 1:57:01 AM | static | | 7/10/2020 2:00:29 PM | rw-rw-r-x |
| templates | | File folder | 6/25/2020 1:50:27 AM | templates | | 7/10/2020 2:00:24 PM | rw-rw-r-x |
| static | | File folder | 6/24/2020 10:01:38 P | requirements.txt | 1 KB | 6/28/2020 8:40:19 PM | rw-rw-r-- |
| venv | | File folder | 6/21/2020 12:43:03 A | main.py | 4 KB | 6/25/2020 2:01:40 AM | rw-rw-r-- |
| requirements.txt | 1 KB | Text Document | 6/28/2020 8:40:19 PM | | | | |
| main.py | 4 KB | Python File | 6/25/2020 2:01:40 AM | | | | |

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



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
```

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://52.152.171.128/
```

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

➔ 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 52.152.171.128;
```

```
    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
```

➔ Activate the virtual env

```
source /home/ubuntu/endtoend_v1/.virtualenvs/endtoend_v1_venv/bin/activate
```

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

```
pip3 install -r /home/ubuntu/endtoend_v1/requirements.txt
```

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

```
pip3 install gunicorn
```

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

```
cd /home/ubuntu/endtoend_v1
gunicorn main:app -w 3
```

This will set your Gunicorn process off running in the background, which will work fine for your purposes here. An improvement that can 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
```

- ➔ 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 URL into browser to see our custom index.html page



Startup Profit Prediction

Below is an example form built to predict the Startup Profits

Startup Expenses Details

Startup Profit must be \$ 97423.08

R&D Expenses

Administration Expenses

Marketing Expenses

State

Predict