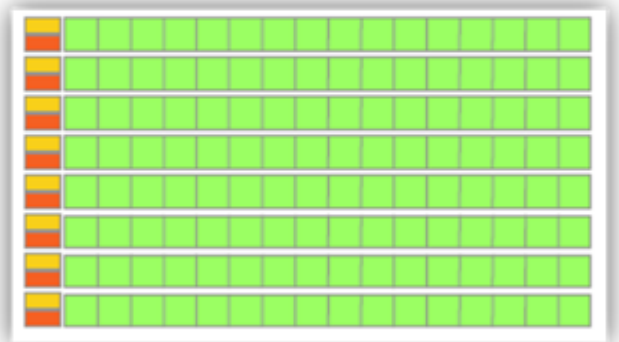
Setting up Deep Learning Environment with GPU in AWS

Institute of Analytics USA ™
Chennai, India
New Jersey, USA
Dallas, USA

CPU

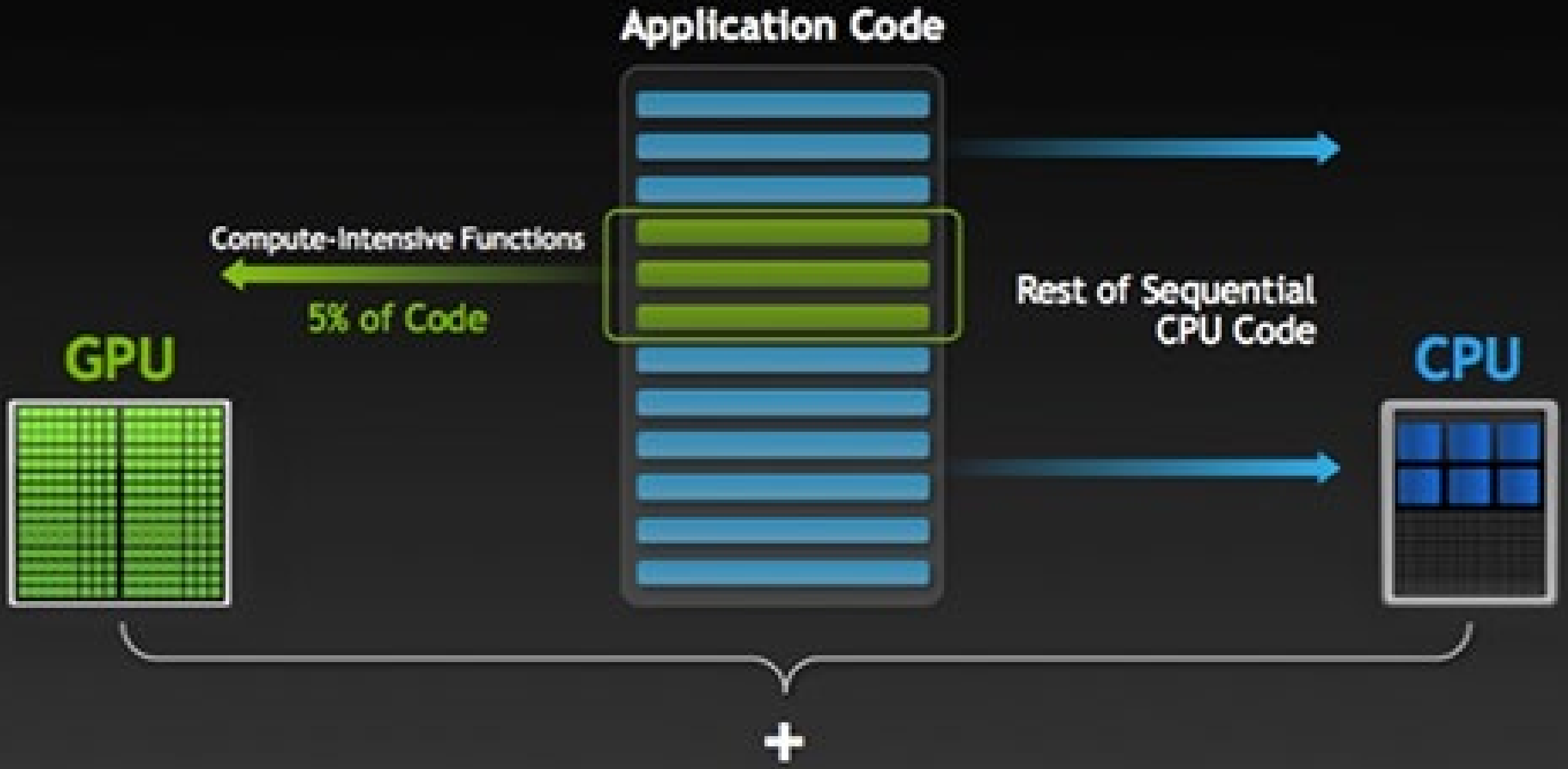
* Low compute density
* Complex control logic
* Large caches (L1$/L2$, etc.)
* Optimized for serial operations
  - Fewer execution units (ALUs)
  - Higher clock speeds
* Shallow pipelines (<30 stages)
* Low Latency Tolerance
* Newer CPUs have more parallelism

GPU



* High compute density
* High Computations per Memory Access
* Built for parallel operations
  - Many parallel execution units (ALUs)
  - Graphics is the best known case of parallelism
* Deep pipelines (hundreds of stages)
* High Throughput
* High Latency Tolerance
* Newer GPUs:
  - Better flow control logic (becoming more CPU-like)
  - Scatter/Gather Memory Access
  - Don’t have one-way pipelines anymore

How GPU Acceleration Works



SOME COMMON QUESIONS

Why GPU when we can do the work with CPU chips? -
https://colab.research.google.com/notebooks/gpu.ipynb#scrollTo=tMce8muBqXQP

It is so easy to setup a GPU machine in Google colab.

Why do we have to go through a difficult process of setting it up in AWS or even in Google?

- Google colab is useful to get an exposure on learning principles of machine learning as a methodology; However, to apply it in real life projects you need to setup GPU machine, whether it is in Google GCP or Microsoft Azure or Amazon AWS.

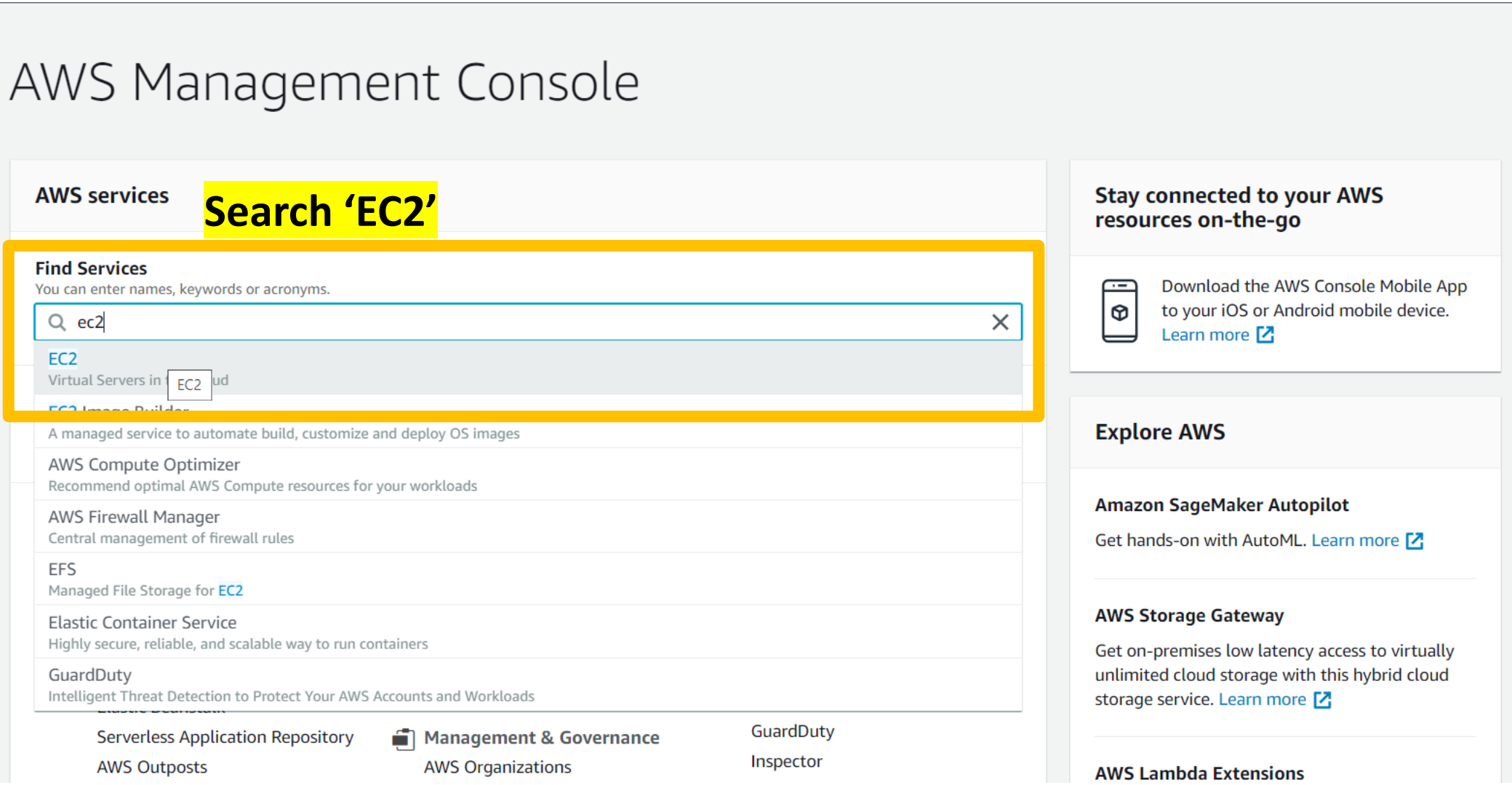
- In other words, if you are looking at full stack development and application implementation, you need to work with your computational (GPU) platform

- Today we are showing this how to create instances in AWS

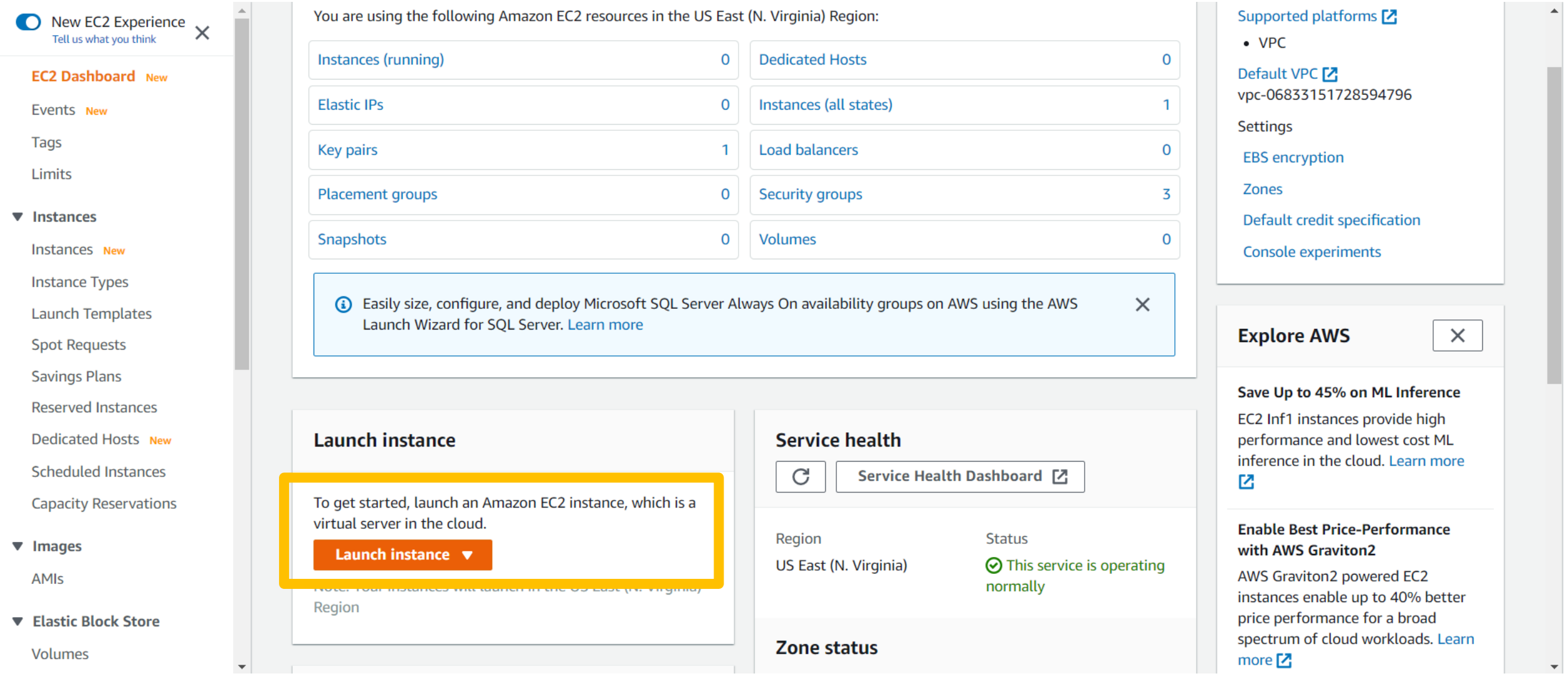
- We will use this for vision engineering, NLP, Generative models, and Recommendation Engine applications

- If you want to work with GCP, use the ref: https://medium.com/@senthilnathangautham/colab-gcp-compute-how-to-link-them-together-98747e8d940e

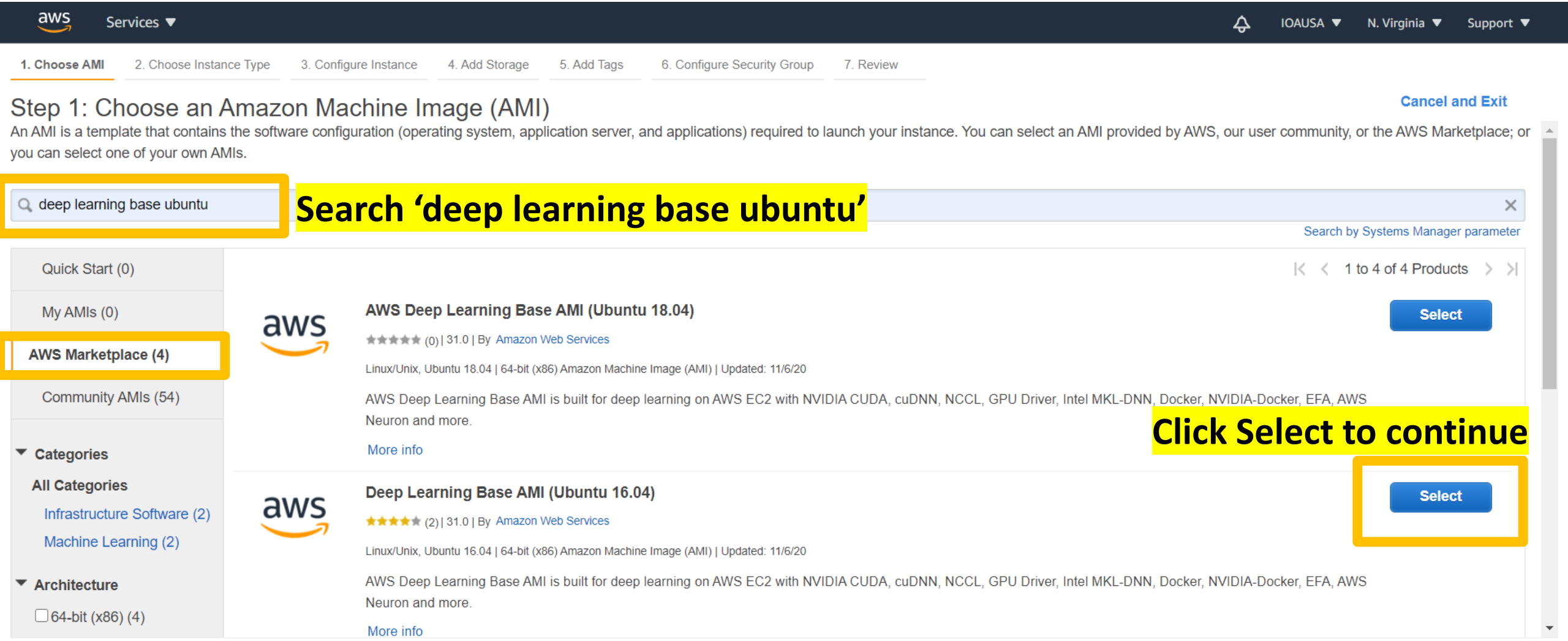
Log in to your AWS Management Console

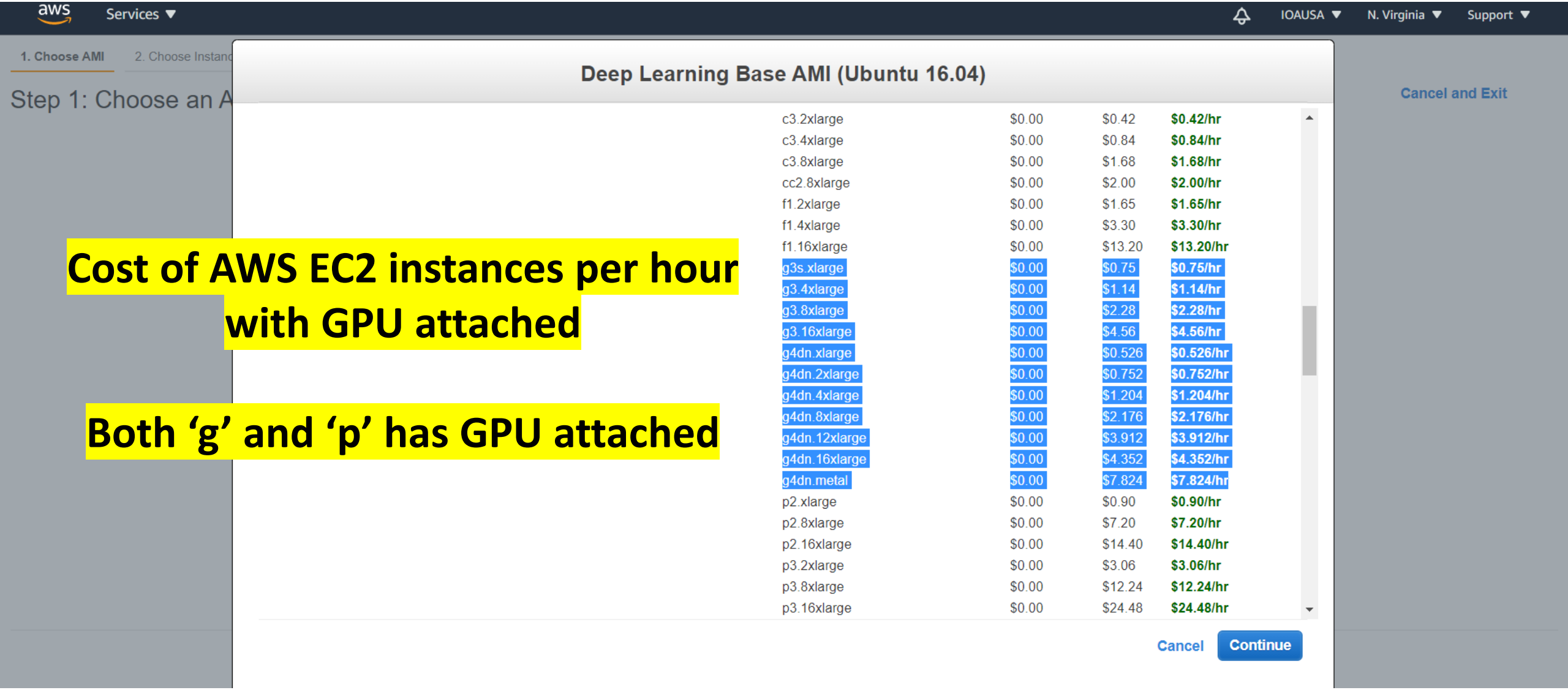


Launch a new instance



Choose AMI (Amazon Machine Image)





Cost of AWS EC2 instances per hour with GPU attached

Both ‘g’ and ‘p’ has GPU attached

Detailed information of instance type ‘g’

Product Details

	Instance Size	vCPUs	Memory (GB)	GPU	Storage (GB)	Network Bandwidth (Gbps)	EBS Bandwidth (GBps)	On-Demand Price/hr*	1-yr Reserved Instance Effective Hourly* (Linux)	3-yr Reserved Instance Effective Hourly* (Linux)
Single GPU VMs	g4dn.xlarge	4	16	1	125	Up to 25	Up to 3.5	$0.526	$0.316	$0.210
	g4dn.2xlarge	8	32	1	225	Up to 25	Up to 3.5	$0.752	$0.452	$0.300
	g4dn.4xlarge	16	64	1	225	Up to 25	4.75	$1.204	$0.722	$0.482
	g4dn.8xlarge	32	128	1	1x900	50	9.5	$2.176	$1.306	$0.870
	g4dn.16xlarge	64	256	1	1x900	50	9.5	$4.352	$2.612	$1.740
Multi GPU VMs	g4dn.12xlarge	48	192	4	1x900	50	9.5	$3.912	$2.348	$1.564
	g4dn.metal	96	384	8	2x900	100	19	$7.824	$4.694	$3.130

* Prices shown are for US East (Northern Virginia) AWS Region. Prices for 1-year and 3-year reserved instances are for "Partial Upfront" payment options or "No Upfront" for instances without the Partial Upfront option.

Choose 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

			CPU	RAM	STORAGE			
☐	f1	f1.16xlarge	64	976	4 x 940 (SSD)	Yes	25 Gigabit	Yes
☐	g3	g3.4xlarge	16	122	EBS only	Yes	Up to 10 Gigabit	Yes
☐	g3	g3.8xlarge	32	244	EBS only	Yes	10 Gigabit	Yes
☐	g3	g3.16xlarge	64	488	EBS only	Yes	25 Gigabit	Yes
☐	g3s	g3s.xlarge	4	30.5	EBS only	Yes	Up to 10 Gigabit	Yes
☑	g4dn	g4dn.xlarge	4	16	1 x 125 (SSD)	Yes	Up to 25 Gigabit	Yes
☐	g4dn	g4dn.2xlarge	8	32	1 x 225 (SSD)	Yes	Up to 25 Gigabit	Yes
☐	g4dn	g4dn.4xlarge	16	64	1 x 225 (SSD)	Yes	Up to 25 Gigabit	Yes
☐	g4dn	g4dn.8xlarge	32	128	1 x 900 (SSD)	Yes	50 Gigabit	Yes
☐	g4dn	g4dn.12xlarge	48	192	1 x 900 (SSD)	Yes	50 Gigabit	Yes
☐	g4dn	g4dn.16xlarge	64	256	1 x 900 (SSD)	Yes	50 Gigabit	Yes
☐	g4dn	g4dn.metal	96	384	2 x 900 (SSD)	Yes	100 Gigabit	Yes

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

Configure Security Group

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: Deep Learning Base AMI -Ubuntu 16-04--31-0-AutogenByAWSMP-1

Description: This security group was generated by AWS Marketplace and is based on recomn

Type	Protocol	Port Range	Source		Description	
SSH	TCP	22	Anywhere	0.0.0.0/0, ::/0	SSH for access from local computer	⊗
HTTPS	TCP	443	Anywhere	0.0.0.0/0, ::/0	HTTPS for Jupyter	⊗
Custom TCP F	TCP	8888	Anywhere	0.0.0.0/0, ::/0	Port to host Jupyter	⊗

Add Rule

Warning

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Cancel | Previous | **Review and Launch**

Download and keep the .pem file safe

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

By launching this product, you will be subscribed to this software and agree that your use of this software is subject to the applicable terms and conditions and the seller may access and use certain information to facilitate your use.
End User License Agreement

Instance Type

Edit instance type

Instance Type	ECUs	vCPUs	Me	ork Performance
g4dn.xlarge	-	4	16	25 Gigabit

Security Groups

Edit security groups

Security group name Deep Learning Base AMI -U
Description This security group was gen
Web Services

ersion 31.0 provided by Amazon

Type	Protocol	ption
SSH	TCP	r access fro...
SSH	TCP	r access fro...
HTTPS	TCP	for Jupyter
HTTPS	TCP	for Jupyter

Cancel | Previous | 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

deep-learning-instance

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**

Instance is now running

Launch Status

Click to see instance summary

Your instances are now launching

The following instance launches have been initiated: i-0a933ec30999527bc View launch log

Get notified of estimated charges

Create billing alerts to get an email notification when estimated charges on your AWS bill exceed an amount you define (for example, if you exceed the free usage tier).

How to connect to your instances

Your instances are launching, and it may take a few minutes until they are in the **running** state, when they will be ready for you to use. Usage hours on your new instances will start immediately and continue to accrue until you stop or terminate your instances.

Click **View Instances** to monitor your instances' status. Once your instances are in the **running** state, you can **connect** to them from the Instances screen. Find out how to connect to your instances.

Getting started with your software

To get started with Deep Learning Base AMI (Ubuntu 16.04)

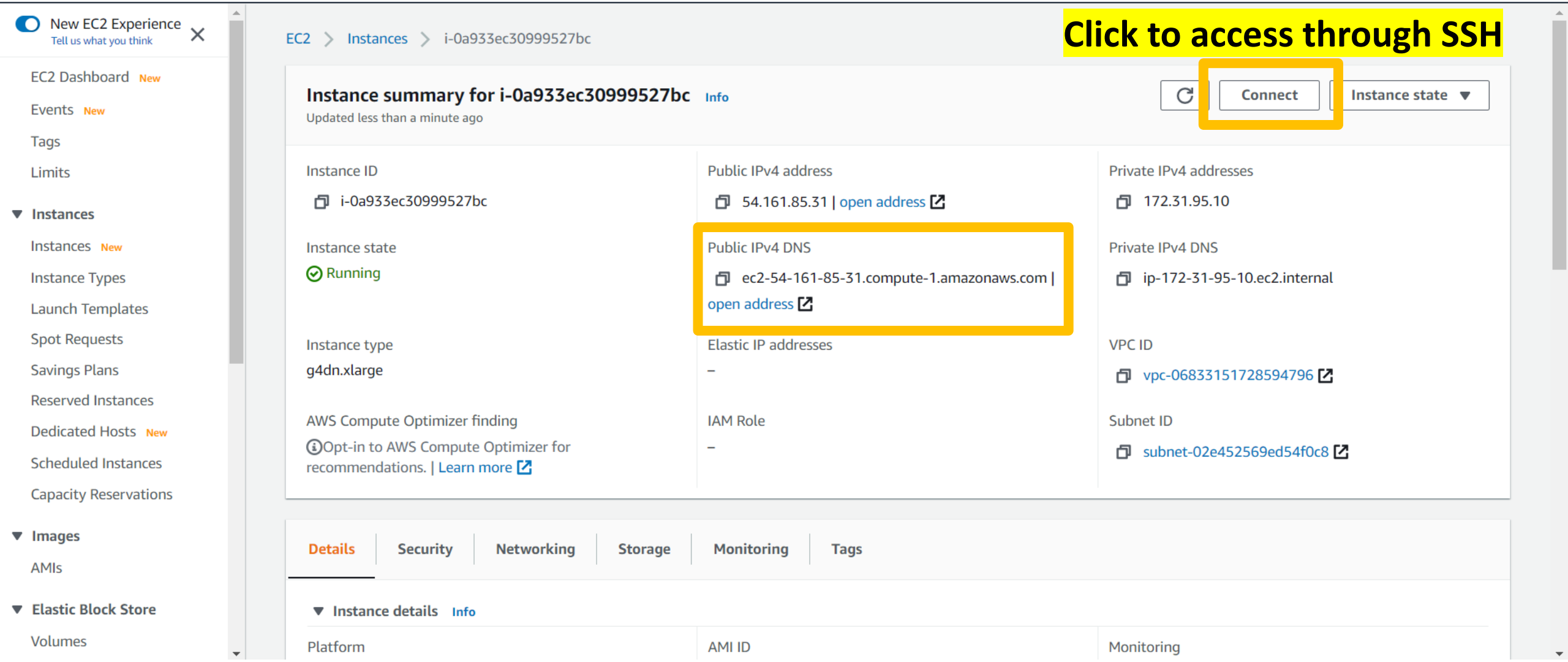
View Usage Instructions

To manage your software subscription

Open Your Software on AWS Marketplace

Here are some helpful resources to get you started

Instance summary



EC2 > Instances > i-0a933ec30999527bc > Connect to instance

Connect to instance Info

Connect to your instance i-0a933ec30999527bc using any of these options

EC2 Instance Connect | Session Manager | SSH client

Instance ID

i-0a933ec30999527bc

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is deep-learning-instance.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.

   chmod 400 deep-learning-instance.pem

4. Connect to your instance using its Public DNS:

   ec2-54-161-85-31.compute-1.amazonaws.com

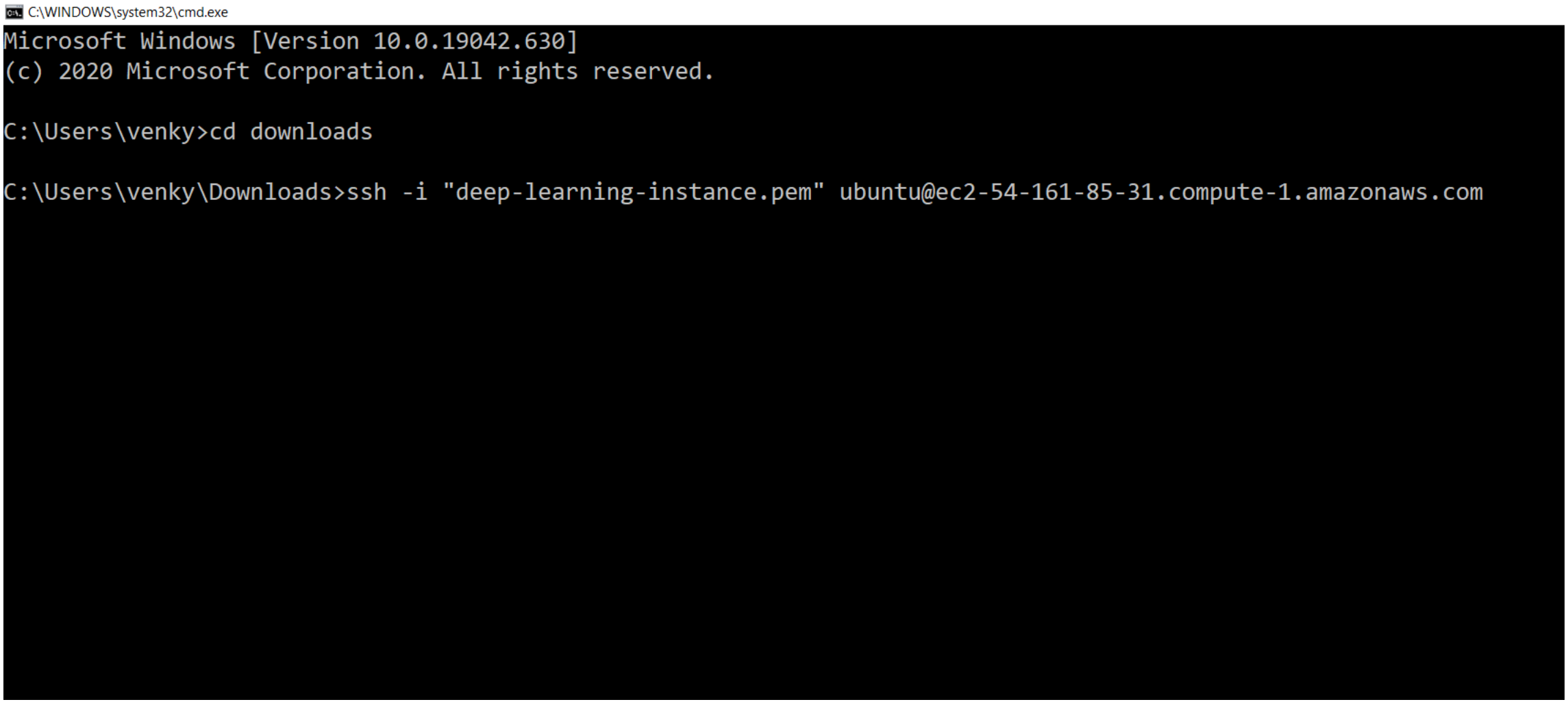
Copy

Example:

ssh -i "deep-learning-instance.pem" ubuntu@ec2-54-161-85-31.compute-1.amazonaws.com

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

Open command prompt in the directory where you downloaded the .pem file and Paste



ubuntu@ip-172-31-95-10: ~

```
Warning: Permanently added 'ec2-54-161-85-31.compute-1.amazonaws.com,54.161.85.31' (ECDSA) to the list of known hosts.
=============================================================================
      __|  __|_  )
      _|  (     /   Deep Learning Base AMI (Ubuntu 16.04) Version 31.0
     ___|\___|___|
=============================================================================

Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 4.4.0-1117-aws x86_64v)

Nvidia driver version: 450.80.02
CUDA versions available: cuda-10.0 cuda-10.1 cuda-10.2 cuda-11.0
Default CUDA version is 10.0
Libraries: cuDNN, NCCL, Intel MKL-DNN

AWS Deep Learning AMI Homepage: https://aws.amazon.com/machine-learning/amis/
Developer Guide and Release Notes: https://docs.aws.amazon.com/dlami/latest/devguide/what-is-dlami.html
Support: https://forums.aws.amazon.com/forum.jspa?forumID=263
For a fully managed experience, check out Amazon SageMaker at https://aws.amazon.com/sagemaker
When using INF1 type instances, please update regularly using the instructions at: https://github.com/aws/aws-neuron-sdk/tree/master/release-notes
=============================================================================

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

 * Introducing self-healing high availability clusters in MicroK8s.
   Simple, hardened, Kubernetes for production, from RaspberryPi to DC.

     https://microk8s.io/high-availability

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



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

ubuntu@ip-172-31-95-10:~$
```

Now we are connected to the AWS EC2 instance through SSH

Download Anaconda 3

```
ubuntu@ip-172-31-95-10:~$ wget https://repo.anaconda.com/archive/Anaconda3-2019.03-Linux-x86_64.sh
```

Install

```
ubuntu@ip-172-31-95-10:~$ bash Anaconda3-2019.03-Linux-x86_64.sh
```

(Type 'Yes' for all prompts. Type 'No' if it asks to install conda init feature)

Configure .bashrc to use python/jupyter from Anaconda

```
ubuntu@ip-172-31-95-10:~$ sudo nano .bashrc
```

Add the below line at the end of .bashrc file

```
export PATH=/home/ubuntu/anaconda3/bin:$PATH
```

Run the below command to make these changes to take effect

```
ubuntu@ip-172-31-95-10:~$ source .bashrc
```

Configuring Jupyter Notebook settings

First, you need to create our Jupyter configuration file. In order to create that file, you need to run:

```
jupyter notebook --generate-config
```

After creating your configuration file, you will need to generate a password for your Jupyter Notebook using ipython:

Enter the IPython command line:

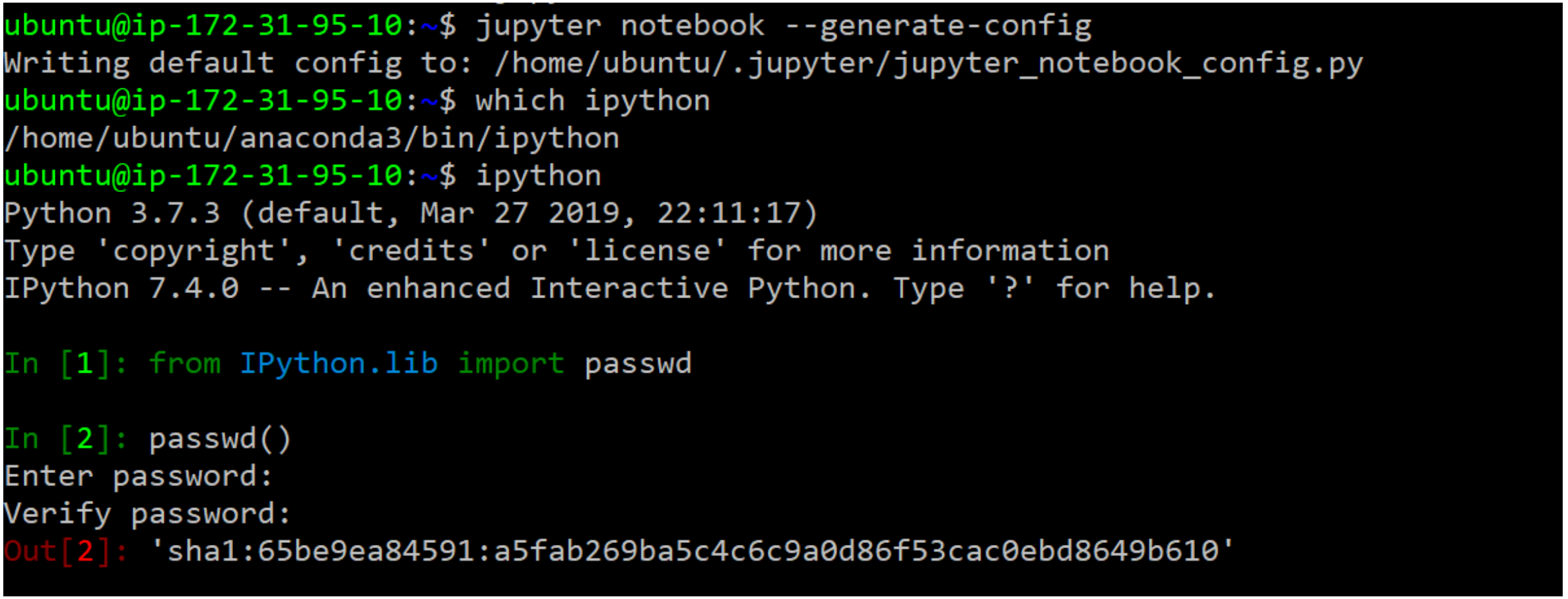
```
ipython
```

Now follow these steps to generate your password:

```
from IPython.lib import passwd

passwd()
```

Preview


ubuntu@ip-172-31-95-10:~$ jupyter notebook --generate-config
Writing default config to: /home/ubuntu/.jupyter/jupyter_notebook_config.py
ubuntu@ip-172-31-95-10:~$ which ipython
/home/ubuntu/anaconda3/bin/ipython
ubuntu@ip-172-31-95-10:~$ ipython
Python 3.7.3 (default, Mar 27 2019, 22:11:17)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.4.0 -- An enhanced Interactive Python. Type '?' for help.
In [1]: from IPython.lib import passwd
In [2]: passwd()
Enter password:
Verify password:
Out[2]: 'sha1:65be9ea84591:a5fab269ba5c4c6c9a0d86f53cac0ebd8649b610'

Connecting to your EC2 Jupyter Server

You should be ready to run your notebook and access your EC2 server. To run your Notebook simply run the command:

jupyter notebook

From there you should be able to access your server by going to:

https://(your AWS public dns):8888/

For example it should look like:

http://ec2-54-161-85-31.compute-1.amazonaws.com:8888/

You will be prompted to enter and re-enter your password. IPython will then generate a hash output, COPY THIS AND SAVE IT FOR LATER. We will need this for our configuration file.

Next go into your jupyter config file:

```
cd .jupyter
```

```
sudo nano jupyter_notebook_config.py
```

And add the following code:

```
conf = get_config()

conf.NotebookApp.ip = '0.0.0.0'
conf.NotebookApp.password = u'YOUR PASSWORD HASH'
conf.NotebookApp.port = 8888
```

```
 GNU nano 2.5.3                File: jupyter_notebook_config.py                Modif

conf = get_config()

conf.NotebookApp.ip = '0.0.0.0'
conf.NotebookApp.password = u'sha1:65be9ea84591:a5fab269ba5c4c6c9a0d86f53cac0ebd8649b610'
conf.NotebookApp.port = 8888

# Configuration file for jupyter-notebook.

#------------------------------------------------------------------------------
# Application(SingletonConfigurable) configuration
#------------------------------------------------------------------------------

## This is an application.

## The date format used by logging formatters for %(asctime)s
#c.Application.log_datefmt = '%Y-%m-%d %H:%M:%S'

## The Logging format template
#c.Application.log_format = '[%(name)s]%(highlevel)s %(message)s'

## Set the log level by value or name.
#c.Application.log_level = 30

#------------------------------------------------------------------------------
# JupyterApp(Application) configuration
#------------------------------------------------------------------------------

## Base class for Jupyter applications


^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     ^Y Prev Page   M-\ First Line
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Linter   ^_ Go To Line  ^V Next Page   M-/ Last Line
```

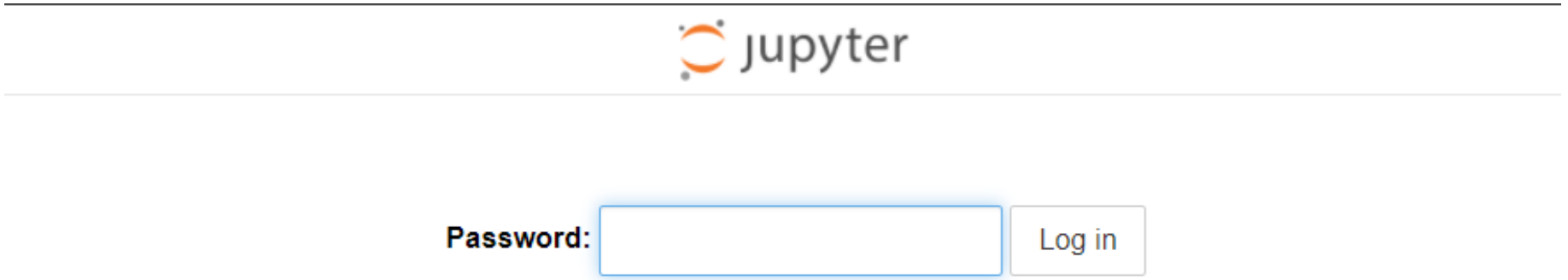
Create a directory for your notebooks

This step is easy. In order to make a folder to store all of your Jupyter Notebooks simply run:

mkdir MyNotebooks

You can call this folder anything, for this example we call it “MyNotebooks”

You should be brought to a page like this:



After successful login

Verify GPU

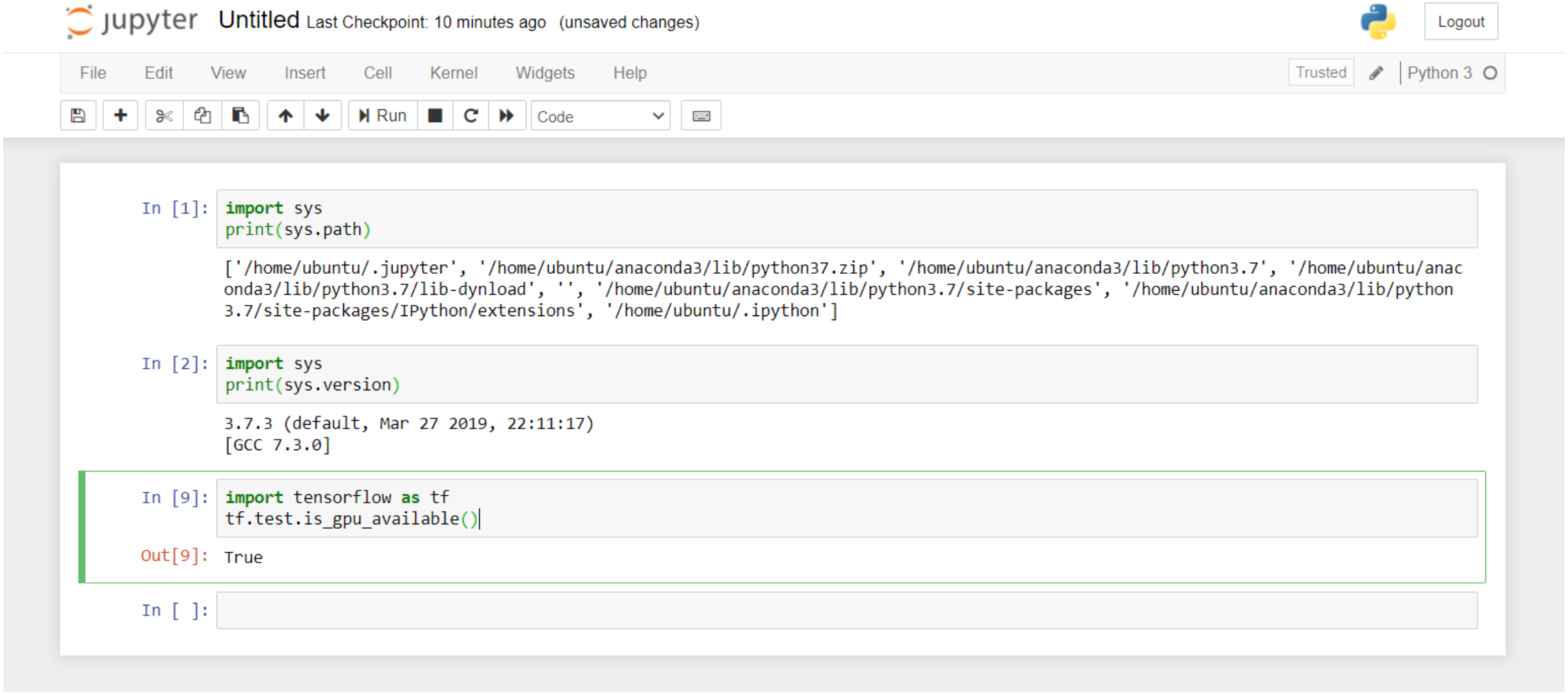
You can confirm that the GPU is working by opening a notebook and typing:

```
from tensorflow.python.client import device_lib

def get_available_devices():
    local_device_protos = device_lib.list_local_devices()
    return [x.name for x in local_device_protos]

print(get_available_devices())
```

Preview


jupyter Untitled Last Checkpoint: 10 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
In [1]:
import sys
print(sys.path)
['/home/ubuntu/.jupyter', '/home/ubuntu/anaconda3/lib/python37.zip', '/home/ubuntu/anaconda3/lib/python3.7', '/home/ubuntu/anaconda3/lib/python3.7/lib-dynload', '', '/home/ubuntu/anaconda3/lib/python3.7/site-packages', '/home/ubuntu/anaconda3/lib/python3.7/site-packages/IPython/extensions', '/home/ubuntu/.ipython']
In [2]:
import sys
print(sys.version)
3.7.3 (default, Mar 27 2019, 22:11:17)
[GCC 7.3.0]
In [9]:
import tensorflow as tf
tf.test.is_gpu_available()
Out[9]:
True
In [ ]:

5 QUIZ QUESTIONS

- Why can't we just use Google Colab?
- What is the minimum best configuration for a GPU compute engine (instance)? Describe the properties of this system – RAM, Disk Space, Broadband exchange, Cost
- Why do we need SSH?
- What are the three different ways, the security is upheld?
- Why not setup a GPU based compute machine at home with your

TERMINOLOGIES

- GPU – Graphical Processing Unit
- SSH or Secure Shell is a cryptographic network protocol for operating network services securely over an unsecured network.
- PEM stands for Privacy Enhanced Mail. It is a security encoded message sent to you that also has the PRIVATE key for your connection to AWS (in addition to security of the message itself
- EBS - Amazon Elastic Block Store (Storage) (**EBS**) is an easy to use, high performance block storage service designed for use with Amazon Elastic Compute Cloud (**EC2**) for both throughput and transaction intensive workloads at any scale.