

High Performance Computing

Introduction

DS730

In this activity, you will be getting familiar with many of the tools we will use in this course. We will be using a preconfigured Linux machine in the cloud that has all of the software we need on it. In order to do some true high performance computing, we will be using a cloud service.

Important:

You are welcome to use newer versions of the software but *you do so at your own risk*. At the time of this writing, the software shown is the most appropriate software available and everything has been tested with this setup. It's possible a newer version comes out between the modification of this document and when you are taking this course. It's likely that everything written here will apply to future updates. However, if something goes wrong with newer software, you will be on your own for troubleshooting it.

If you already have access to a cluster/machine with Hadoop/Pig/Hive/Spark installed on it, you can skip Task 1. The goal of Task 1 is to spin up a virtual machine in the cloud with the Hortonworks sandbox installed on it. Task 2 is optional and will show you how to create a virtual machine locally if you wish to install Hortonworks on your local machine¹. But be warned, using such a sandbox locally requires a lot of disk space and a lot of memory. I recommend having about 120GB of free space available. I also recommend having 10GB of memory available (not total memory, but available memory).

Books & Other Software

Because of the fast paced nature of this area, there is no required book for this course. However, many of the activities are quite long and detailed so one could consider the set of activities the book for this course. The advantage of this kind of "book" is that it can be updated quickly if there is an update that needs to be made. If you come across

¹ Please note that installing Hortonworks locally is done on your own. The instructors do not have the ability to troubleshoot a broken/failed local install. The instructions for the remainder of the course are written for the Hortonworks version on Azure.

a portion of text that no longer makes sense or see an outdated image, let your instructor know right away so we can update the document.

One of the most common questions asked in this course is why we don't learn tool XYZ where XYZ is the *best* thing we can learn in this course. As the world of high performance computing changes, the goal of this course is to keep up with the newest technologies and present you the newest versions of each software we use. At the same time, we want to teach the core concepts of these technologies. We do not want you to be pigeonholed into using a specific software tool to solve a problem. If you learn the core concepts of several paradigms, then you should be able to pick up whatever the software du jour is when you need it.

Hortonworks Sandbox

It is important to understand what is going on inside the sandbox. The sandbox is essentially simulating a cluster of machines that run Hadoop, Pig, Hive and other pieces of software. You do not have to install each piece individually. Rather, you can use the software straight out of the box without having to deal with the administrative part of it. If you want to know more about how to install some of the software, see the optional activities for how to setup a Linux machine, install Hadoop, Pig, Hive, etc.

In this course, we will focus on two parts of the Hadoop environment but there are many other parts and pieces of software you can learn about.

The first part is the Hadoop Distributed File System (HDFS). HDFS allows for scalable and reliable data storage. It is designed to run on many “cheap” computers. HDFS stores data in large chunks and each chunk is stored on multiple machines (in general) to provide reliable access to the data. For example, consider chunk X is stored on machines A, B and C. Assume you start a job that needs to use chunk X. If A and B are busy doing other work, you can run your job on chunk X on machine C. This runs into the classic “space vs time” problem. You can replicate your data as much as you want across numerous machines. If you replicate a lot, your total storage capacity will go down but your processing time will also go down. If you replicate a little, your total storage capacity will go up but your processing time will also go up. We won't spend much time looking into the innards of HDFS. A general knowledge that HDFS is distributed storage is sufficient for this course.

The main part we will concern ourselves with is how to access and manipulate the data. In this course, we will use MapReduce, Pig, Hive and Spark. There are many other

software titles available inside Hortonworks and if there is enough interest, optional activities can be created to learn about Storm, Kafka, Tez or other applications.

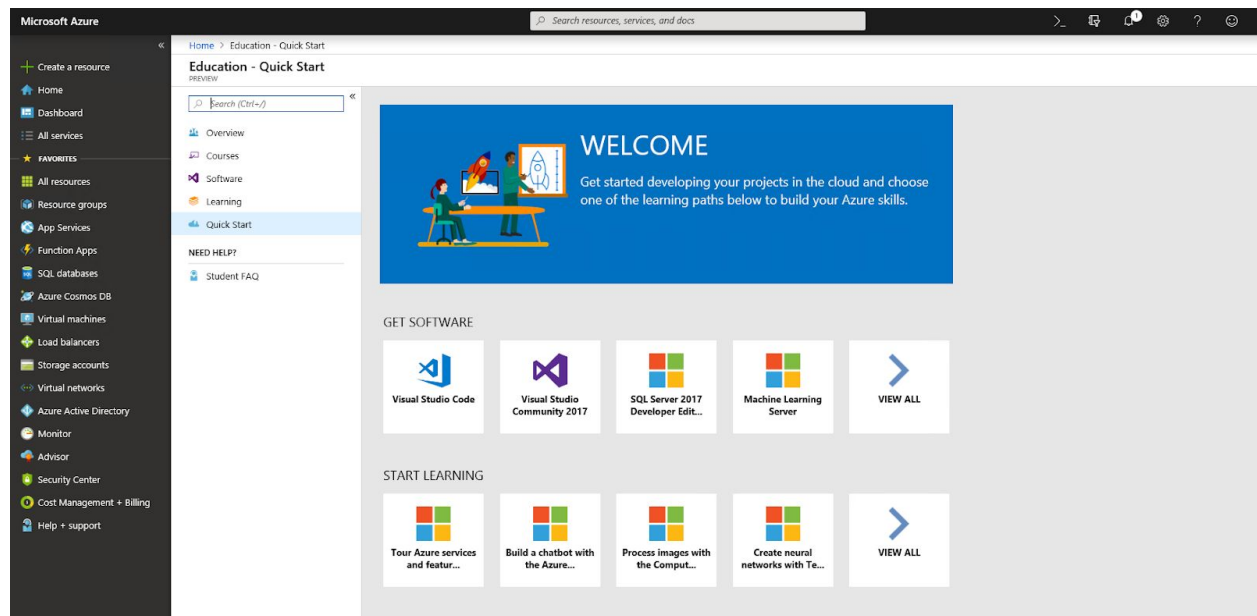
Activity Tasks

Task 1: Create a Hortonworks Sandbox in the Cloud

This task creates a sandbox in the cloud with most of the required software installed on it. The main advantage of this is that you do not have to use 1 single machine for this course and can connect to your sandbox from anywhere. If there is a problem, your instructor can connect to your machine and troubleshoot any issues.

1. Go to <https://azure.microsoft.com/en-us/free/students/> and click on the **Activate now** button. If you already have an account with your school e-mail address, then sign in. Otherwise, click on **Create one!** and sign up for an account. I do not know how Microsoft's verification works so your prompts and pages may be different from mine. Essentially, do what you need to do to create an account. This was how I created mine.

I was not able to create a new account with my .edu address so I simply used an old hotmail account. I was eventually able to get signed in. When signed in, it asked me to verify my student status with a school email address. I typed in my .edu email account and clicked on **Verify academic status**. After a few seconds, an email showed up in my inbox and I clicked on the verification link. It asked me to verify again by phone. I was not able to verify a phone number from google voice so my guess is other VOIP numbers will also fail. I was able to get a text to my regular cell number so that worked fine. A call to your office number would probably work as well. In either case, my identity was verified and I accepted the agreement and hit the **Sign up** button. You may see a popup about starting a tour. Take the tour if you want or simply hit **Maybe later**. Once I was all signed up, I ended up on this page:



2. On the left hand side, click on **Create a resource**.
3. Do a search for **Hortonworks** and click on the option for **Hortonworks Sandbox with HDP 2.6.4**. You should see something that looks like this:

Home > New > Hortonworks Sandbox with HDP 2.6.4

Hortonworks Sandbox with HDP 2.6.4

Hortonworks

Bring Your Own License enabled.

About To Deploy?

For a step-by-step guide on how to deploy the Hortonworks Sandbox on Azure, visit: [Deploying Hortonworks Sandbox on Microsoft Azure](#).

Already Set Up and Looking to Learn?

There are a series of tutorials to get you going with HDP fast. To learn more about the HDP Sandbox check out: [Learning the Ropes of the Hortonworks HDP Sandbox](#). To get started using Hadoop to store, process and query data try this HDP 2.6 tutorial series: [Hello HDP an introduction to Hadoop](#)


Have Questions?

For all your Hadoop and Big Data questions, and to get answers directly from the pros fast, visit: [Hortonworks Community Connection](#)

[Learn More](#)

- [Browse: Big Data Tutorials](#)
- Tutorial: [Deploying Hortonworks Sandbox on Microsoft Azure](#)
- Tutorial: [Learning the Ropes of the Hortonworks Sandbox](#)

Sandbox is an enterprise Hadoop environment with many interactive tutorials to jumpstart your learning. Sandbox includes many of the most exciting developments from the latest HDP distribution, which you can get up and running in under 15 minutes. It comes with tutorials to help you learn Hadoop, Spark, Pig, Hive, Kafka, Storm, HBase, Ranger, Falcon, Ambari and YARN.

 [Save for later](#)

4. Scroll to the bottom and click on the **Create** button. From here we just need to customize our sandbox.
5. On the **Basics** tab, click on **Create new** for the resource group. You can call your resource anything you want. I called my DS730. For the **Virtual machine name** I called it Hortonworks. Under Instance Details, look at the **Size** option. We are only allowed 4 cores with our student account. Therefore, click on the **Change size** link and choose the **Standard B4ms** option. I changed the **Administrator Account Authentication type** to be a password. It is less secure than an SSH public key but it is sufficient for this course. Create any username/password you want. You will never need to use this username/password in this course. Click on the **Review + create** button. Feel

free to click on the Next buttons if you would like to configure your virtual machine more. However, the default options are sufficient for this course.

6. Review the options you chose and click on the **Create** button at the bottom. Deployment will take a while so simply wait for it to finish. You should see something like this (I clicked on the bell icon to get the notifications on the right hand side):

7. After about 5 minutes, my deployment succeeded. Yours will likely take longer as you will be doing it at the beginning of the semester when everyone else is trying to do it. Click on the **Go to resource** button in the middle of the screen. You will see something like this:

8. Under the **Settings** option, click on **Networking**. By default, essentially all ports are closed for security. However, we need to connect to them so we need to open up some of the ports. Click on the **Add inbound port rule** button. Add the following port using these rule:
 - a. **Destination port ranges** - Set it to be 8080. It might default at that port already. If that is the case, then leave it.
 - b. **Name** - Not critical but change it to **Ambari** so you know what the port is.
 - c. The rest of the options can be left at their defaults. Feel free to change the **Source** if you are super concerned about security.

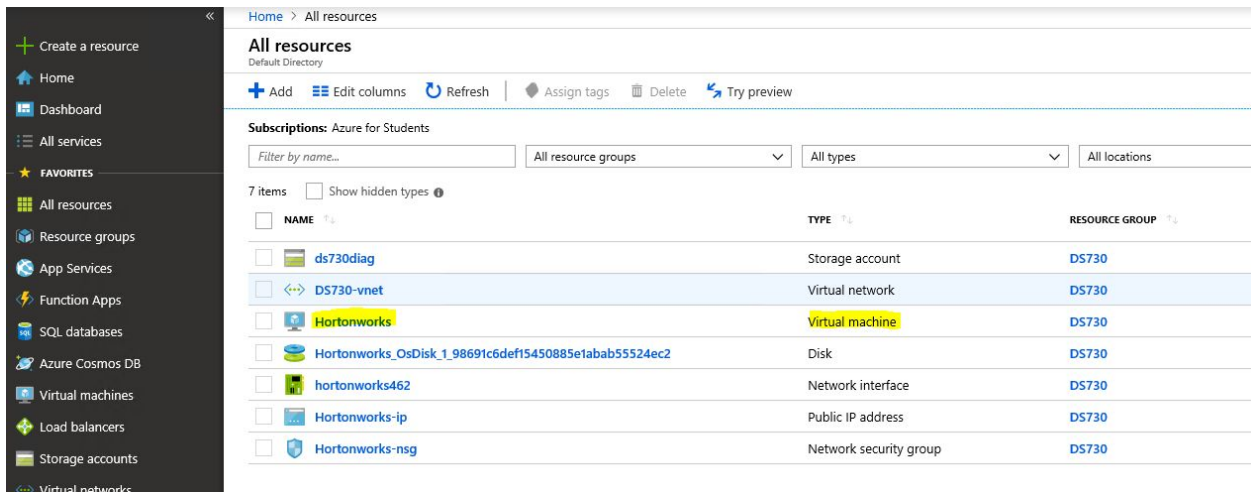
It will take a minute to create the rule. Add the following Port/Name inbound port rules as well: (4200/ssh, 9995/zeppelin). You will have to change the priority to be a different number for each rule. Change it to whatever you want.

9. Your Hortonworks Sandbox is setup and is now ready to use. A great guide to learning about your Sandbox is here:

<https://hortonworks.com/tutorial/learning-the-ropes-of-the-hortonworks-sandbox>

However, everything you need to know for this course will be shown in this and future activities.

10. Click on the **All resources** option on the left. Click on the **Hortonworks** link where the type is **Virtual machine**:



NAME	TYPE	RESOURCE GROUP
ds730diag	Storage account	DS730
DS730-vnet	Virtual network	DS730
Hortonworks	Virtual machine	DS730
Hortonworks_OsDisk_1_98691c6def15450885e1abab55524ec2	Disk	DS730
hortonworks462	Network interface	DS730
Hortonworks-ip	Public IP address	DS730
Hortonworks-nsg	Network security group	DS730

11. You will find your Public IP address on the right hand side. Mine looks like this:



Property	Value
Resource group	DS730
Status	Running
Location	East US
Subscription	Azure for Students
Computer name	Hortonworks
Operating system	Linux
Size	Standard B4ms (4 vcpus, 16 GB memory)
Public IP address	40.117.228.13

12. Note that my IP address is 40.117.228.13 and I will use that IP in the steps below. Replace my IP with yours when you do the following steps.

13. We should reset our root password as soon as possible. Go to <http://40.117.228.13:4200> and use **root** as the username and **hadoop** as the password. You will be forced to change this password immediately. Change it to whatever you want. You will not need to use the root username/password nor will you need to use this ambari-admin password in this course. However, it is important to change them from the defaults. Once you are logged in, type in **ambari-admin-password-reset** as seen below:

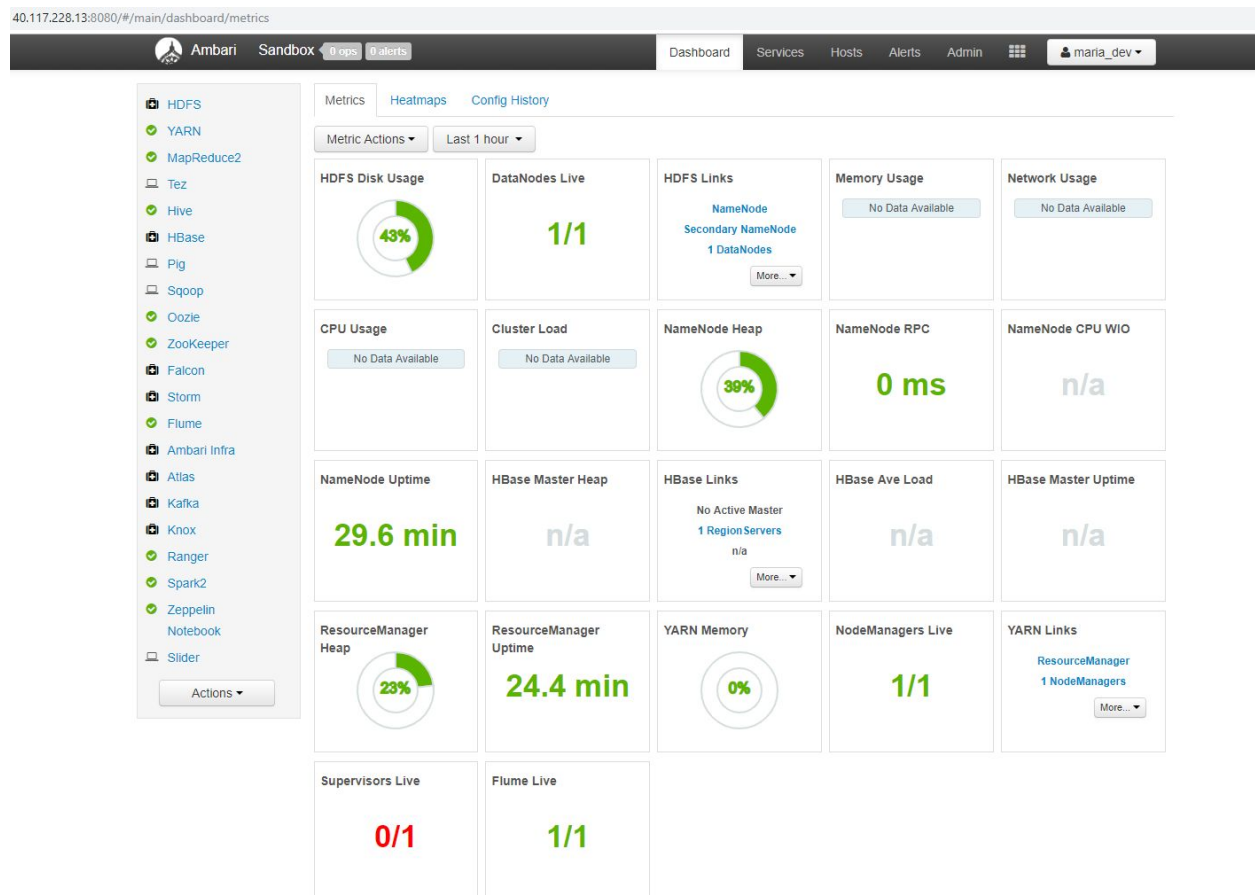

```

sandbox login: root
root@sandbox.hortonworks.com's password:
You are required to change your password immediately (root enforced)
Last login: Mon Dec 31 17:31:32 2018 from 127.0.0.1
Changing password for root.
(current) UNIX password:
New password:
Retype new password:
[root@sandbox-hdp ~]# ambari-admin-password-reset

```

14. Once your password is reset, type in **exit** and you can close out of this browser window.

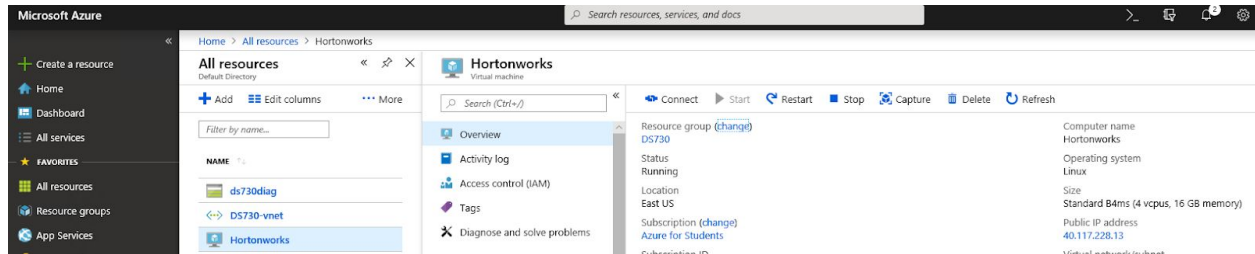
15. Go to <http://40.117.228.13:8080> and enter in **maria_dev** as the username and password. You should see something like this:



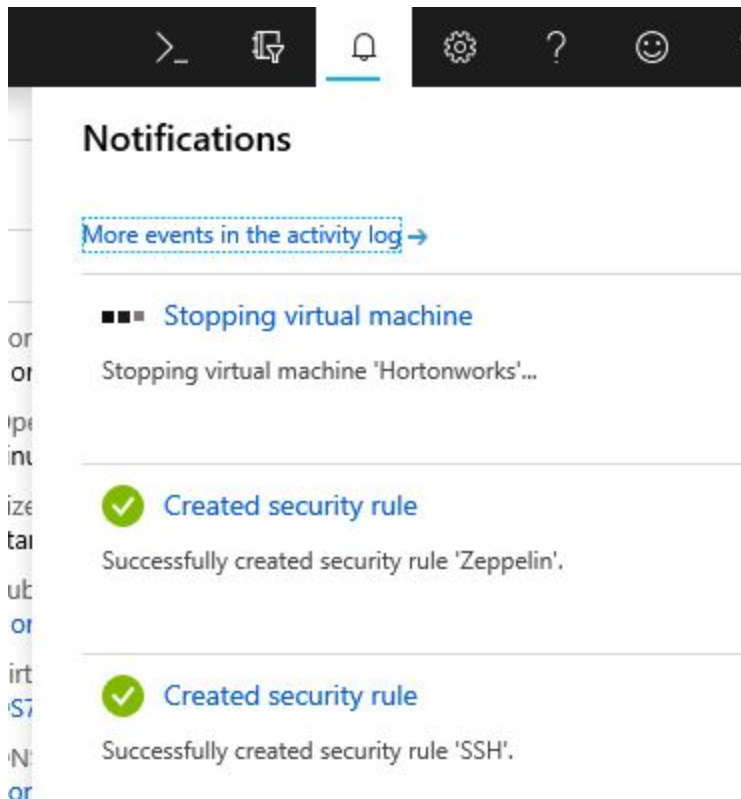
16. As long as there are 0 alerts at the top, you are all set. If you see some alerts, simply wait for the alerts to clear before starting. The virtual machine does take a few minutes to load up all of the software. Feel free to explore the dashboard to

see all of the tools available for you to use. Once you are done exploring, you can close out of this window.

17. Go back to your Azure browser window and click on **All resources**. Click on the **Hortonworks Virtual Machine** and you should see something like this:



18. Click on the **Stop** button in the top-middle of the page. Your virtual machine will shut down and you will no longer be charged for the machine. You are only using your credits when the virtual machine is running. Wait a few minutes and ensure that the virtual machine shuts down. You can click on the bell icon in the upper right hand corner to see what is happening. You should see something like this showing that the virtual machine is being stopped:



19. Eventually, the machine will stop and you are finished. If you want to setup a Hortonworks sandbox on your local machine, continue with step 2. Otherwise, continue with step 3.

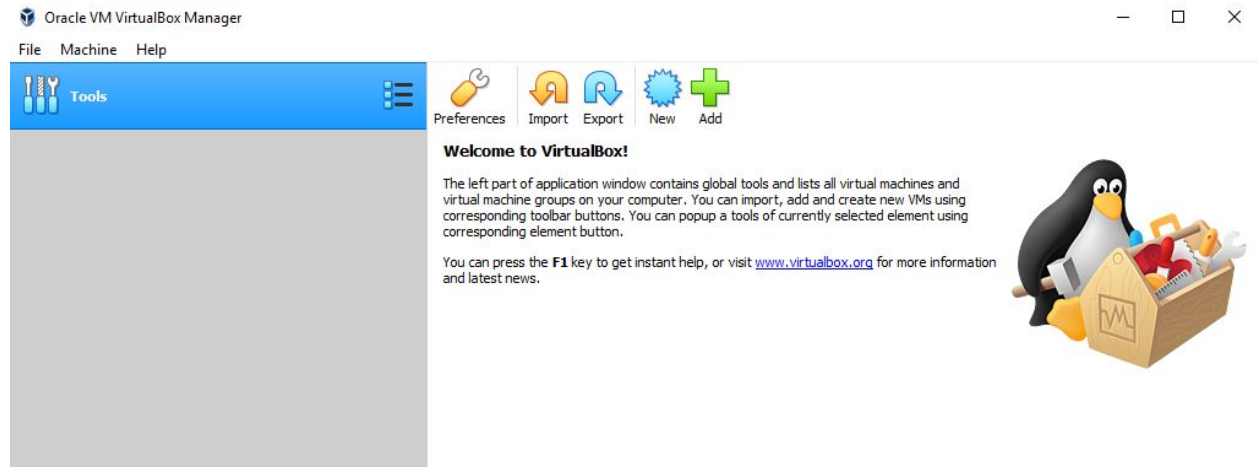
Task 2: Create a Hortonworks Sandbox (Optional)

The newest version on Hortonworks is different from the one on Azure and there are considerable differences in the UI. There are also considerable differences in how you run each of the programs. All instructions for the remainder of the course will assume that you are running Hortonworks on Azure. The directions for running software on your local Hortonworks virtual machine are different. The actual code you create *should* run the same on either setup. However, if there is a difference between how the code runs locally vs the cloud, the version on Azure will be used for testing/grading purposes. Therefore, you should consider your local sandbox a test-only sandbox.

This task creates a sandbox for you and installs all of the software we are using in this course. You will be creating a virtual machine. This task takes a couple of hours depending on your Internet connection speed. However, the majority of it is waiting for everything to download. You will need about 120GB of hard drive space available and a machine that has at least 10GB of memory, preferably more². These instructions have worked on a machine running a 64-bit version of Windows 10 and a 64-bit version of Windows 7.

1. **Download VirtualBox.** The current version is 6.0.0. You can find the download here: www.uwosh.edu/faculty_staff/krohne/ds730/virtualbox.html
2. Install VirtualBox on your machine. It is a simple install that requires you to click **Next** and **Install** a few times.
 - If it asks about installing network devices, this is no problem; you can simply click **Install** on those prompts.
3. When the install is complete, click **Finish**.
4. Open VirtualBox up once it is installed. You should see something like this:

² I had success running the sandbox while allocating only 4GB of RAM to the system and it has worked fairly well. It was a little slow but it was acceptable. I tried running this with 2GB of RAM allocated and it didn't work.



5. **Download Hortonworks Sandbox.** The current version of Hortonworks Data Platform (HDP) 3.0.1. You can download it from here:

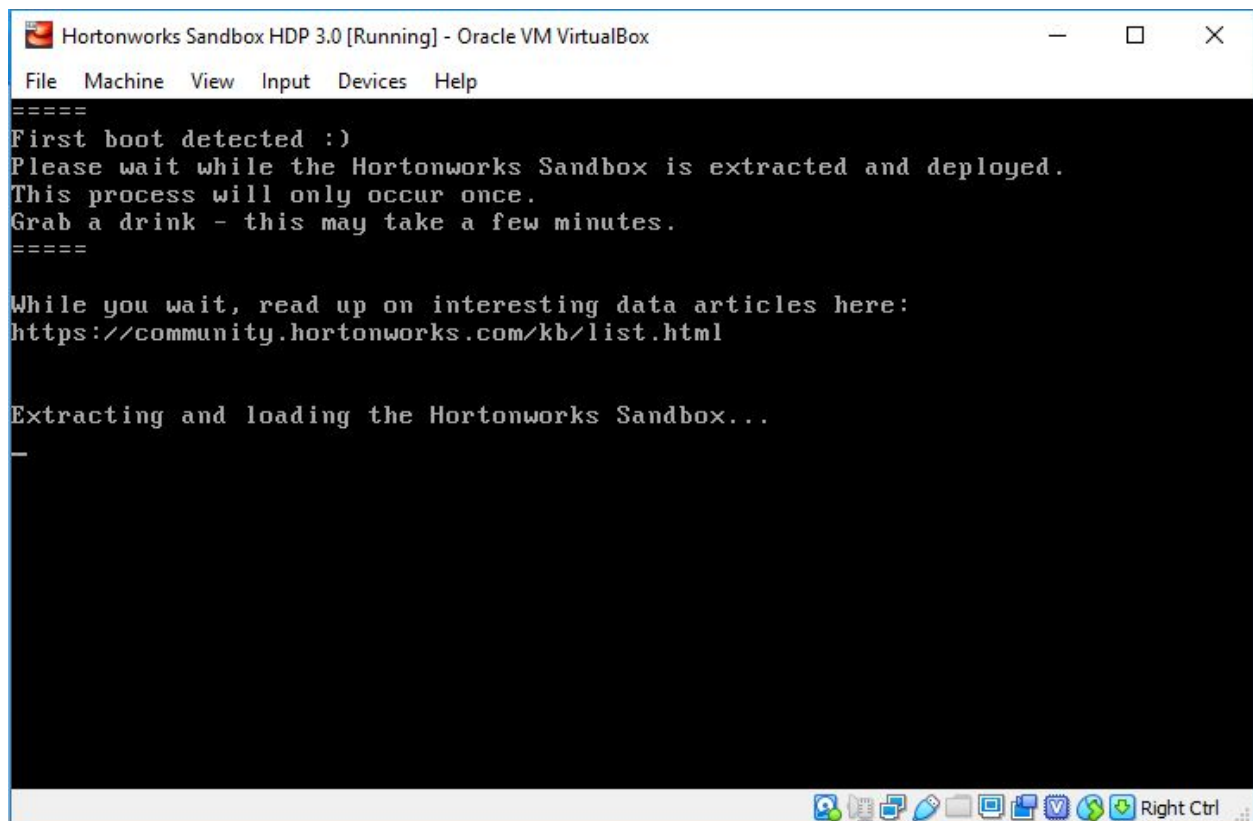
http://www.uwosh.edu/faculty_staff/krohne/ds730/hortonworks.html

- Be sure to download the one for VirtualBox:

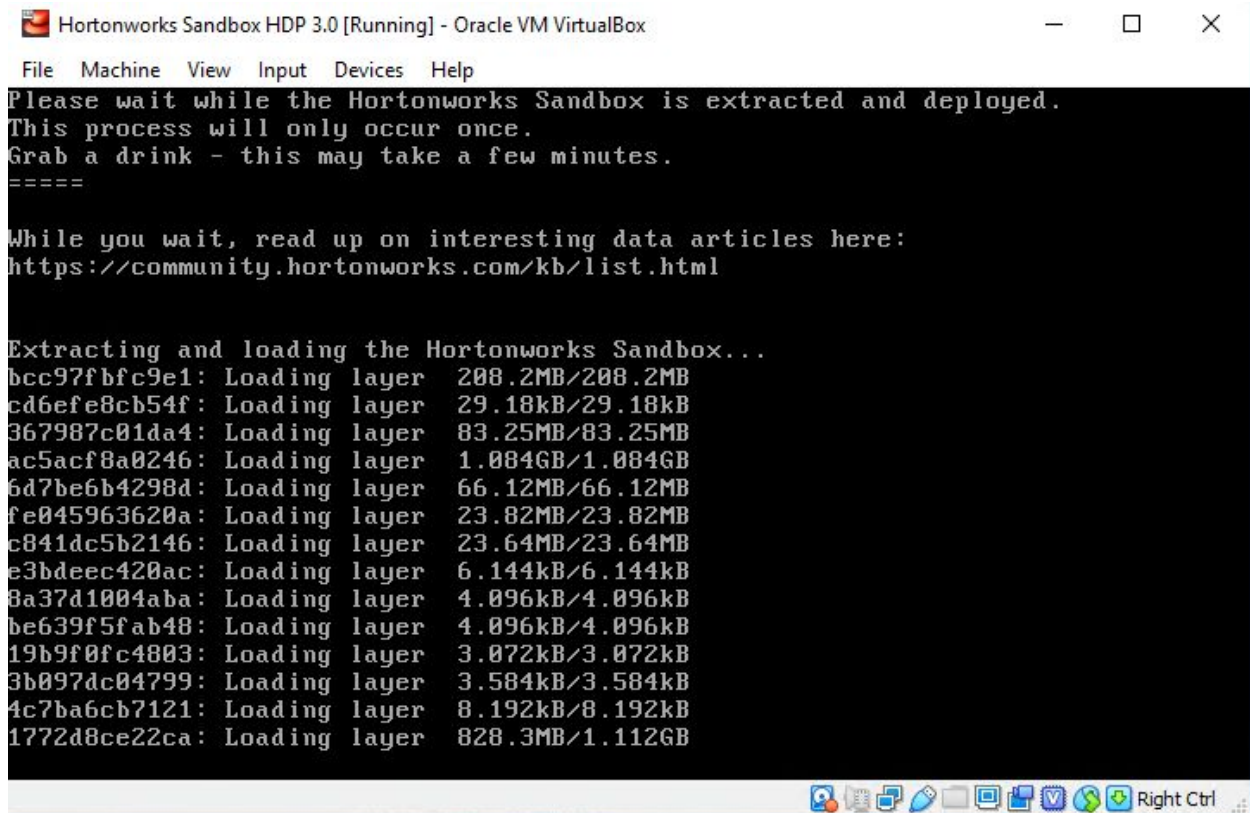
Hortonworks Sandbox	Hortonworks Sandbox on a VM
Sandbox on a VM	Hortonworks Data Platform (HDP®) 3.0.1 on Hortonworks Sandbox Tutorials Release Notes Install Guide on VirtualBox MD5 : cc17e47c3ada7137edb550d26fe0bd49 VIRTUALBOX
Sandbox in the Cloud	
Sandbox Archive	
Sandbox Archive	
Hortonworks DataFlow	Hortonworks Data Platform (HDP®) 3.0.1 on VMware Tutorials Release Notes Install Guide on VMware MD5 : 5114f5731fae0654bcf8f4cd2ca5b827 VMWARE
HDF 3.3.1	
HDF Archive	
HDF Archive	
Hortonworks Data Platform	Hortonworks Data Platform (HDP®) 3.0.1 on Docker Tutorials Release Notes Install Guide on Docker MD5 : fd1a56c3260291818b6eacbc598dfe72
HDP 3.1.0: Ready for the enterprise	
HDP Documentation	
HDP Add-Ons	
HDP Archive	

- It will likely ask you to fill out your name, e-mail address, etc before downloading. Once you submit your information, the download will begin. Note that this is a 20GB download so it may take a while if you have a slower home connection.
6. Once the file has download, go to VirtualBox and click on **File** → **Import Appliance**.
 7. Click on the folder icon and find the ***.ova** file you just downloaded and click **Open**. For reference, mine was called HDP_3.0.1_virtualbox_181205.ova. Once you have selected it, click **Next**.

8. All of the settings can remain unchanged. If you wish to store your sandbox on an external drive, scroll down and click on the **Virtual Disk Image** location setting at the bottom and change the location. You may also change the **RAM** setting if you would like to use more/less memory. Do not use less than 4GB of memory or your sandbox may not run well. The recommended amount of 10GB is preferred. Click on **Import**.
9. As a reference, my initial *Importing virtual disk image* import took roughly 10 minutes. Once everything is imported, click on the name of the virtual box (probably called Hortonworks Sandbox HDP 3.0) and click on **Start**. You should see something like this:



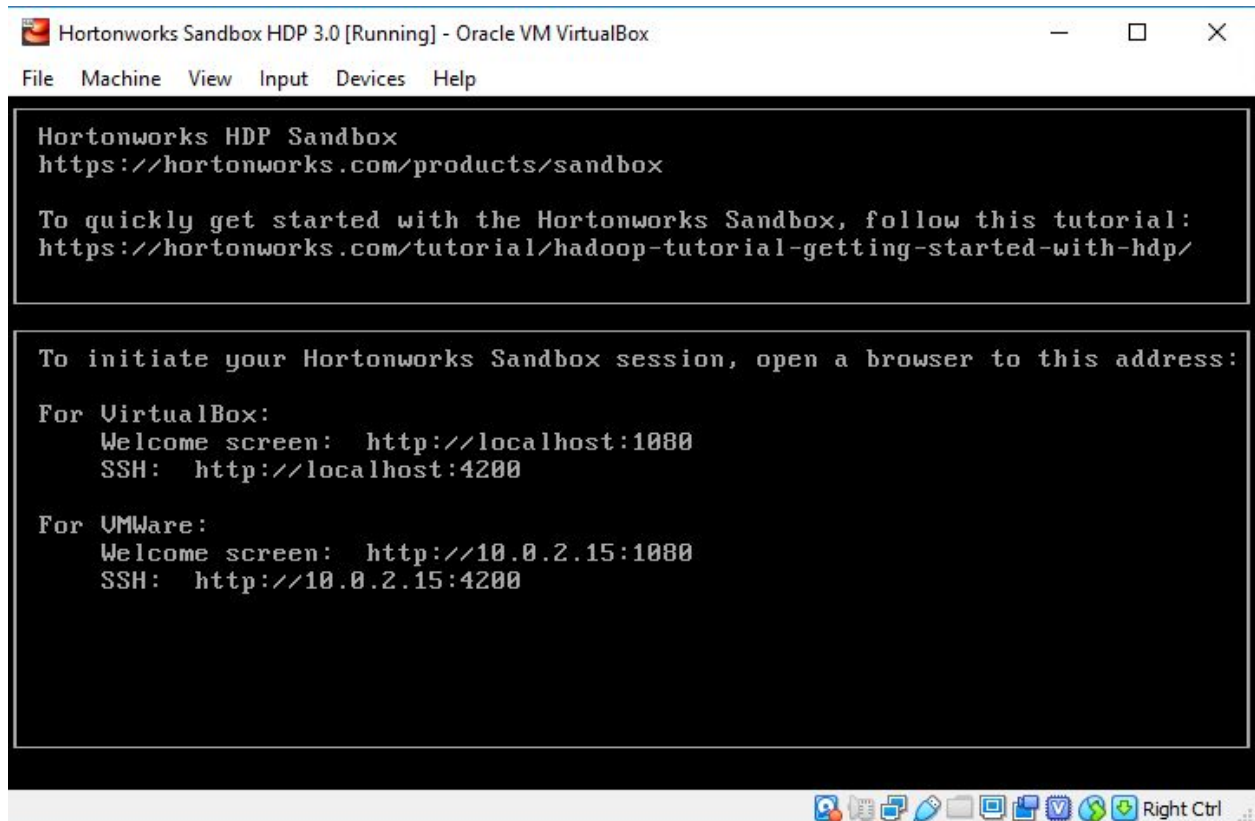
10. This process will take a fair amount of time because you are downloading a lot of software for your sandbox. There is no need to interact with it so just let it run. As a reference, I installed this on my work computer on my very fast 90Mbps school connection. The process took 20 minutes to finish with the *Extracting and loading the Hortonworks Sandbox...* portion before it started downloading a bunch of software:



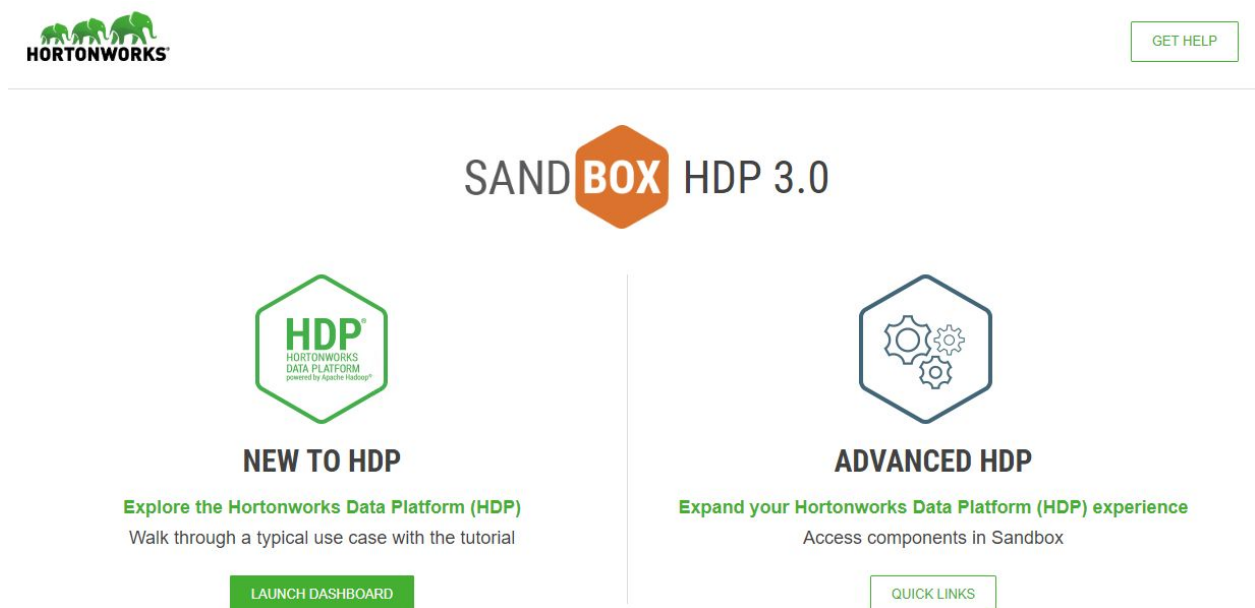
```
Hortonworks Sandbox HDP 3.0 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Please wait while the Hortonworks Sandbox is extracted and deployed.
This process will only occur once.
Grab a drink - this may take a few minutes.
=====
While you wait, read up on interesting data articles here:
https://community.hortonworks.com/kb/list.html

Extracting and loading the Hortonworks Sandbox...
bcc97fbfc9e1: Loading layer 208.2MB/208.2MB
cd6efe8cb54f: Loading layer 29.18kB/29.18kB
367987c01da4: Loading layer 83.25MB/83.25MB
ac5acf8a0246: Loading layer 1.084GB/1.084GB
6d7be6b4298d: Loading layer 66.12MB/66.12MB
fe045963620a: Loading layer 23.82MB/23.82MB
c841dc5b2146: Loading layer 23.64MB/23.64MB
e3bdeec420ac: Loading layer 6.144kB/6.144kB
8a37d1004aba: Loading layer 4.096kB/4.096kB
be639f5fab48: Loading layer 4.096kB/4.096kB
19b9f0fc4803: Loading layer 3.072kB/3.072kB
3b097dc04799: Loading layer 3.584kB/3.584kB
4c7ba6cb7121: Loading layer 8.192kB/8.192kB
1772d8ce22ca: Loading layer 828.3MB/1.112GB
```

11. The virtual machine then spent 25 minutes downloading and installing software needed for the sandbox. Starting from Task 2, step 1 until this point, it took about 75 minutes to reach the following screen.



12. Once you see the previous screen, open up your favorite Internet browser and navigate to <http://localhost:1080>. You should see something like this:



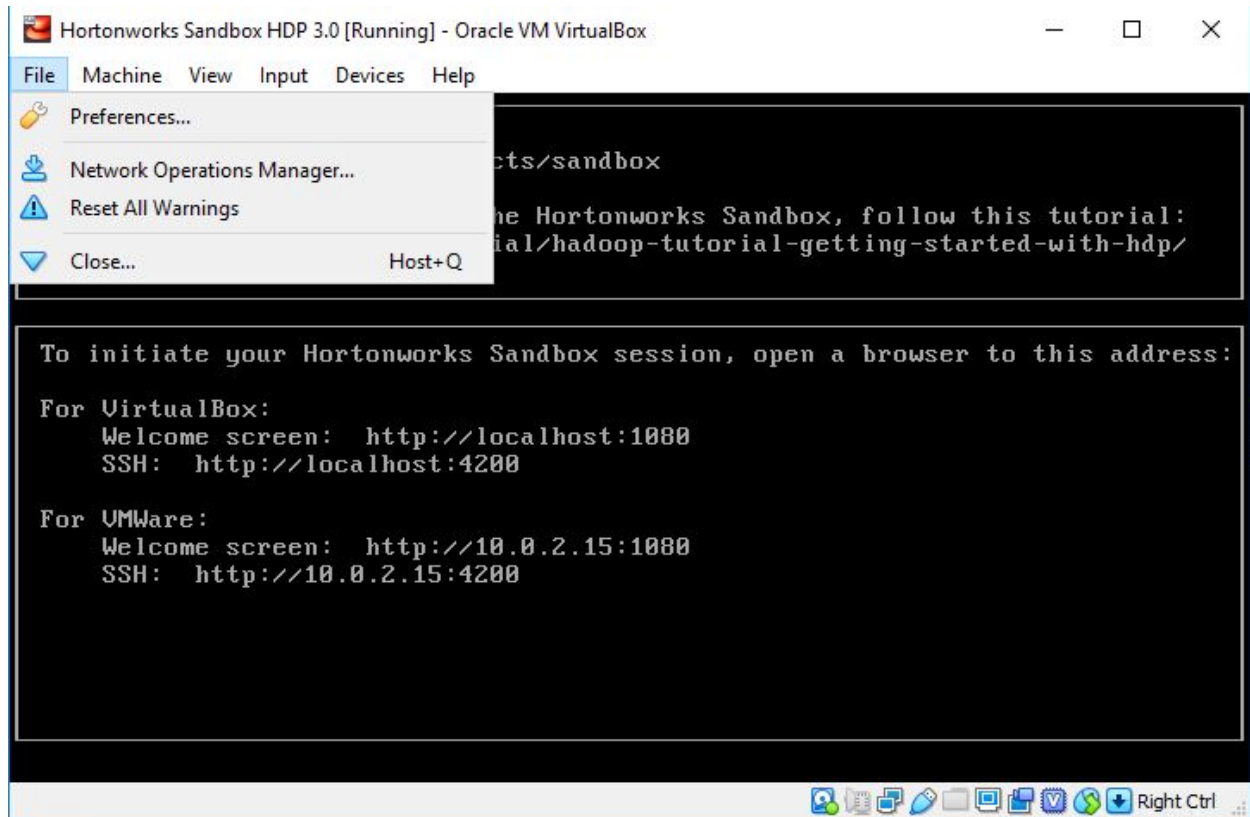
Shell Web Client

The virtual machine that you setup is like any other machine that you have created. There is an operating system, a filesystem, etc. For those of you familiar with the command line, you may want to use the command line to do certain things. Hortonworks provides a nice web client that allows you to connect to your machine.

13. Open up an Internet browser and go to <http://localhost:4200>. You should see something like this:

```
sandbox-hdp login: root
root@sandbox-hdp.hortonworks.com's password:
You are required to change your password immediately (root enforced)
Last login: Thu Jul  5 19:17:23 2018 from 172.18.0.2
Changing password for root.
(current) UNIX password:
New password:
Retype new password:
[root@sandbox-hdp ~]#
```

14. Type in **root** for your username and **hadoop** for the password. It will prompt you to change your password (see above). Enter in a new password that is at least 8 characters in length. You can login using root and make any changes that you want to your system.
15. Type in the following command to create an admin account for the Ambari server:
ambari-admin-password-reset
16. Type in a password that is at least 8 characters long.
17. When you are finished, type in **exit** to close your connection. It will say *Session closed.* and you will see a **Connect** button in the middle of the screen. You can close this window in your browser.
18. In order to shut down your machine, go back to the VirtualBox window that contains your Hortonworks sandbox and click on **File** → **Close...** as seen below:



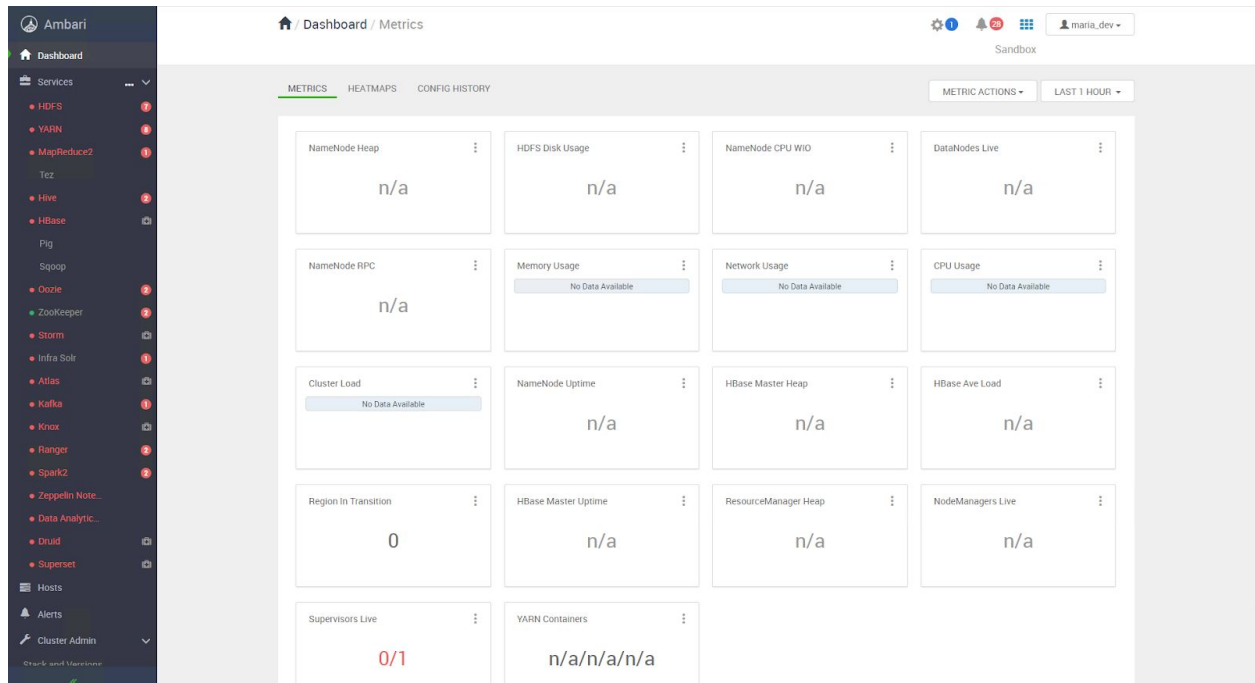
19. Select the **Send the shutdown signal** option and click OK. Your virtual machine will shutdown in a couple of minutes. It may look like nothing is happening but do not force it to close as this might put your virtual machine in a bad state.

Explore Ambari

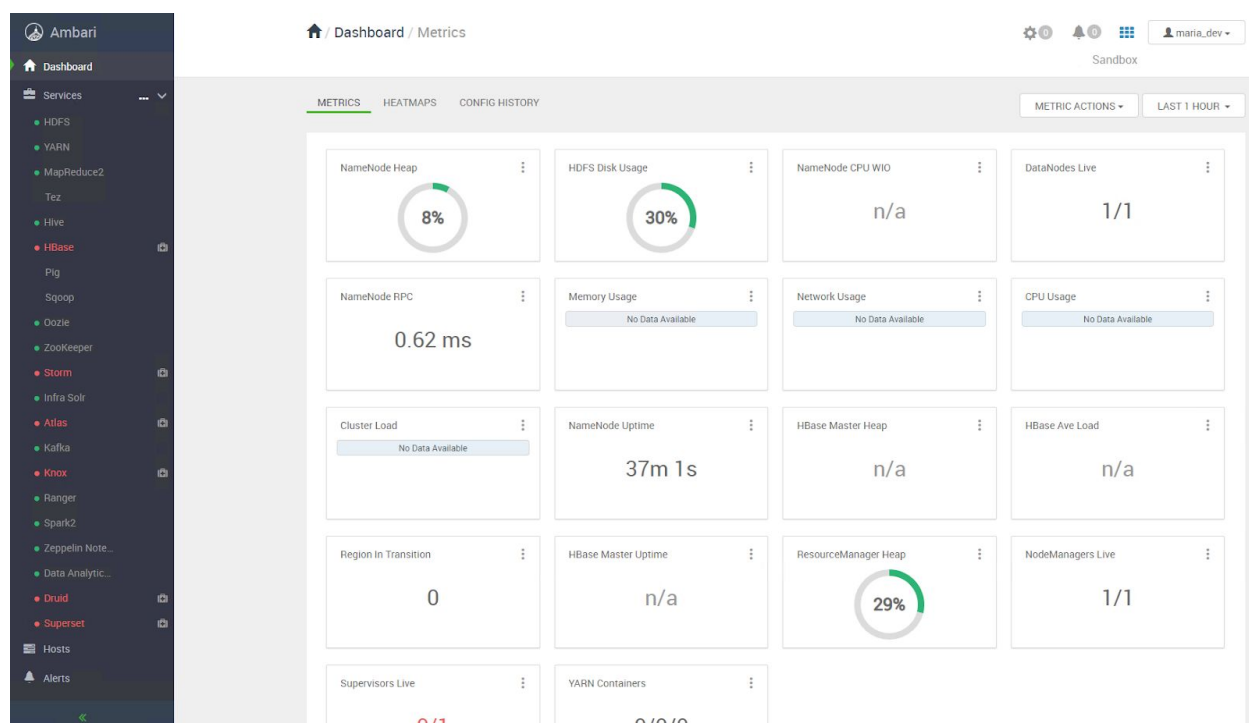
The Ambari dashboard is a nice administrative tool that comes with the sandbox. You will find many applications running but will also find some that are stopped. This is fine as we won't be able to explore everything in the sandbox. Our goal is to view/use some of the main ones.

20. Once your virtual machine has shutdown, we will open it up again. In order to do this, click on the **Hortonworks Sandbox HDP 3.0** and select Start. Your machine will load up in a few minutes. If you see the "welcome screen" in VirtualBox, that means your system is started but it doesn't mean everything is running. If you quickly go to <http://localhost:8080>, you may get a 502 Bad Gateway error or some other error message. This is normal and it simply means your Hortonworks system is not started completely yet.
21. Open up an Internet browser and go to <http://localhost:8080> and type in **maria_dev** for the username and the password. This username will be our main

username throughout the semester. If you log in too quickly, you might see something like this:



This is normal and there is no concern. Just let the system load itself up. It may take several minutes but eventually the red alerts will disappear and everything will be running. If you still have red alerts to the right of the software titles after 15 minutes, then something is likely wrong. Once everything is loaded up, feel free to poke around the dashboard to see what options you have. There isn't much to do right now but we will be using many of these software tools in this course. When you are finished exploring, feel free to shutdown the virtual machine. Your final running page will look something like this:

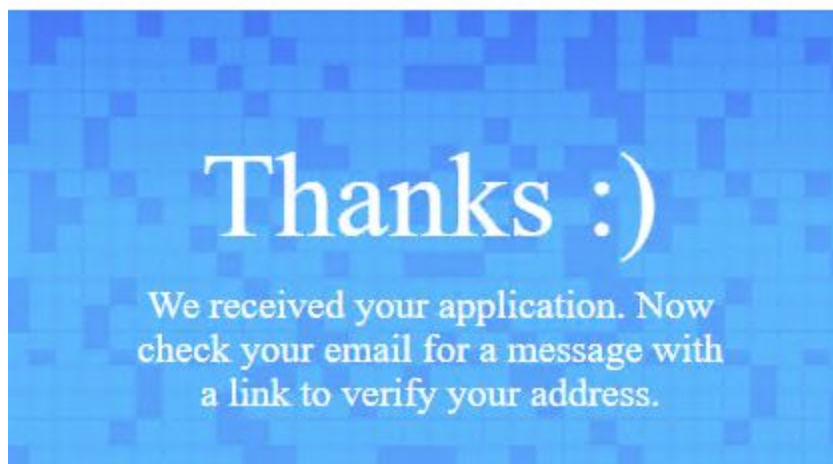


Task 3: Sign Up for Amazon Web Services

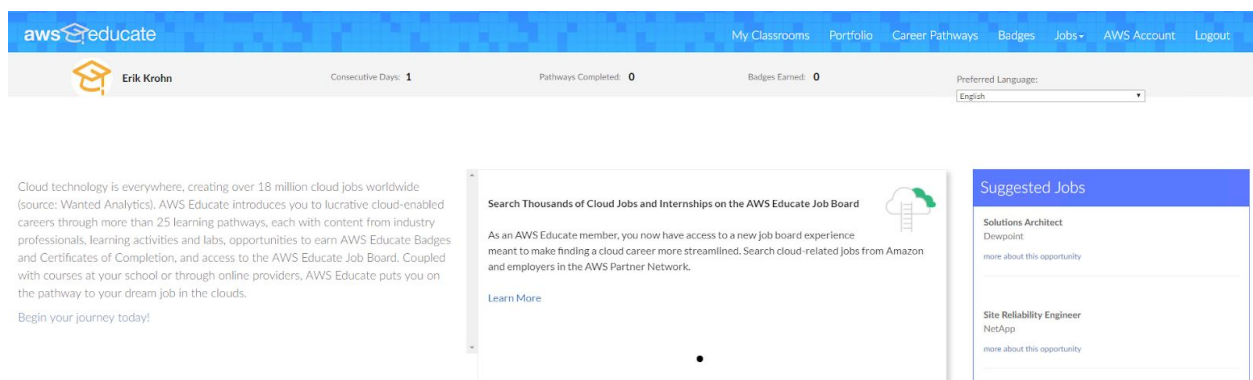
Using the Hortonworks sandbox on Azure is a nice way to simulate running your code on many machines. The reality is, your “cluster” on Azure is simply 1 machine that is running 4 CPUs and has 16GB of memory. Using Azure is a nice way to test your code. If you want to do true high performance computing to tackle terabytes or higher of data, you would want to use a larger cluster. Amazon Web Services (AWS) allows you to spin up as large of a cluster as you need and shut it down whenever you are finished with it.

1. You will receive an e-mail from Amazon. The e-mail I received was from AWS Educate Support at support@awseducate.com. Inside the e-mail, there is a sentence like this: *Click here to complete the AWS Educate application process...* Simply click on the “here” link.
2. The link took me to step 2 of 3 of the AWS Educate sign up. I do not know what step 1 is. If anyone sees a step 1, please tell us so we can update this document. Step 2 simply has you filling in your details. Put your home campus for the university. If there is a *Promo Code* field, you can leave it blank. Once everything is filled in, click **Next**.
3. Choose the **Click here to select an AWS Educate Starter Account**. Click **Next**.
4. You will receive a verification e-mail shortly. Click on the link in that e-mail.

5. Read through the terms and conditions and assuming you accept them, click on the **I Agree** checkbox. Click **Submit**. You will likely be directed to a page that looks like this:

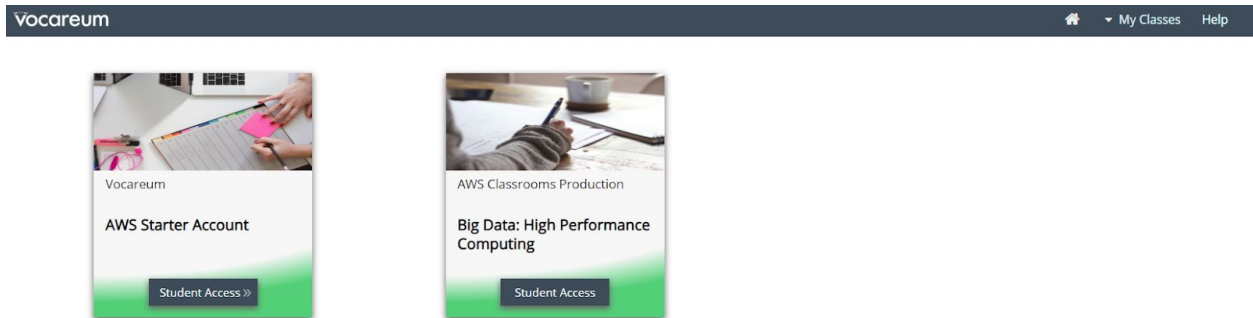


6. Within seconds, I received an e-mail that my application was approved. It is possible that your application will remain under review. If this happens, your application will likely be approved in a few hours. Inside that e-mail there is a link to setup your password and login. Click on that link and create a password.
7. Once you have created a password, you should see something like this:

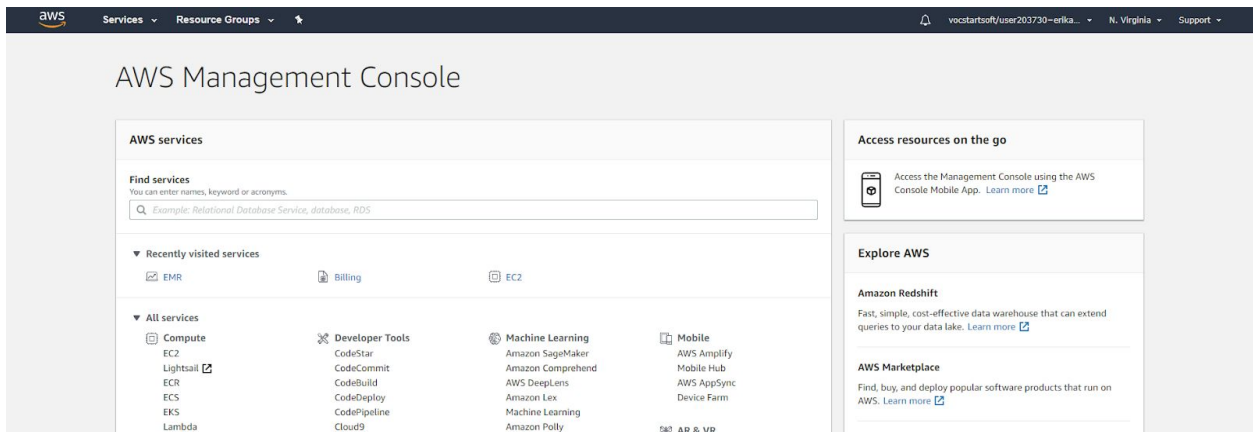


8. Click on the **My Classrooms** link at the top. On the next page, click on the **Go to classroom** button. There is a pop up about going to another page, click

Continue. You will end up on a page with a bunch of terms and conditions. Read those and assuming you accept, click on **I Agree** at the bottom. This redirected me to a page that looks like this:



9. If you do not see the options in the middle, click on the Home icon at the top. Click on the **Student Access** button for the **AWS Starter Account** tile. There are a bunch of FAQ on the left hand side that you can mostly ignore. Click on the **AWS Console** button on the right hand side. The page was a pop-up that my pop-up blocker blocked. Once I allowed the page to load up, this is what I saw:



10. Once you have reached this point, you are done and everything is setup. This is the AWS Console. In order to get back to the AWS Console, you'll have to follow these steps:
- Go to <https://aws.amazon.com/education/awseducate/> and click on the **Login to AWS Educate** link.
 - Sign in with your e-mail address and the password you just created.
 - Repeat steps 8 and 9 again to get back to your AWS Console.

Task 4: Connect to a Linux Machine on AWS

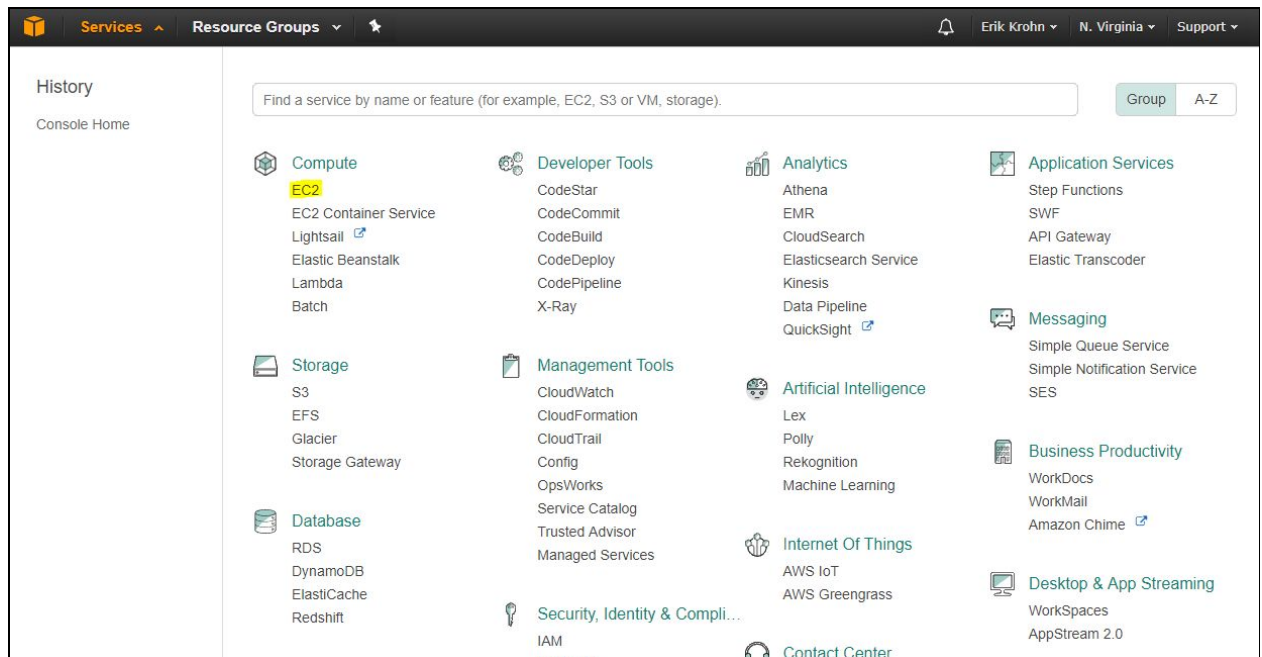
Connecting to a machine in the cloud is a good start for the cloud based computing we will be using in future activities and projects. We will not connect many times to a remote machine in the cloud but it is something you should know how to do. Whenever you spin up a cluster on AWS or some other cloud service, you should be aware of how to connect to the cluster to make any tweaks you want.

1. Go here: www.uwosh.edu/faculty_staff/krohne/ds730/putty.html
2. Download these two files:
 - **putty.exe**
 - **puttygen.exe**

Note:

You can download the Windows installer to install more than you need. However, downloading what is specified above is sufficient.

3. Sign into your AWS console.
4. Click **Services**.
5. In the **Compute** section, click **EC2**:



6. Click the **Launch Instance** button.

Note:

As a comment for the future, if you are looking to save some money, you should look into Spot Requests (Instances). Everything explained in this course is with *On Demand* instances. Basically, when you start up an *On Demand* server, it will run continuously without being interrupted.

7. Select the **Ubuntu Server...** option.
 - As of now, the best option is the 18.04 version.
8. Choose **General Purpose t2.micro** to stay in the free tier. We will eventually be switching our server to a larger instance when we get to the Java portion of this course. But for now, all computing can be done with the t2.micro instance. To ensure you don't waste your credits, make sure to stop your instance when you are done using it and stay within the recommended bounds as described in future documents. Another thing to note is that there are only 1 CPU available and 1GB of memory on the t2.micro machine. If you need more processing power or more memory, adjust accordingly.

Note:

The call is yours to make on how big your instance is initially. Once we get to Java, you will need at least t2.medium. If you need to adjust your instance type, look at step 84 and that will show you how to change the type. You can find all of the pricing details here: <https://aws.amazon.com/ec2/pricing/>.

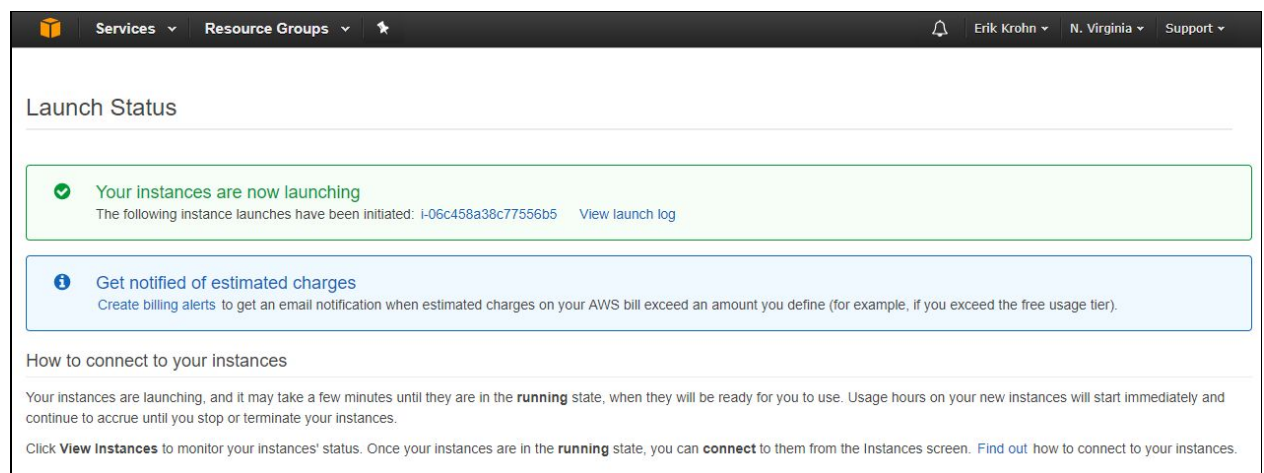
9. Once you've made your decision, click **Next: Configure Instance Details**.
10. All of the defaults are good here. click **Next: Add Storage**.
11. Change the GB size from 8GB to 30GB.
12. Click **Next: Add Tags**.
13. The default values on this page can be kept as they are. Click **Next: Configure Security Group**.
14. Make sure **Create a new security** group is checked.
15. Change the **Source** from **Custom** to **Anywhere**.
 - This assumes you will be connecting to the instance from any machine. You should change it at **My IP** if you will only be connecting from 1 machine and are hyper concerned about security.
16. The type should be **SSH**.
17. The Protocol should be **TCP**.
18. The Port Range should be **22**.

19. Click **Review and Launch**.
20. Click the **Launch** button.
21. Click **Create a new key pair**.
22. Enter a **Key pair name** of whatever you want.
 - I called mine **UbuntuAWS**
23. Click **Download Key Pair**.
24. Save that file somewhere to your hard drive and **do not lose it!**

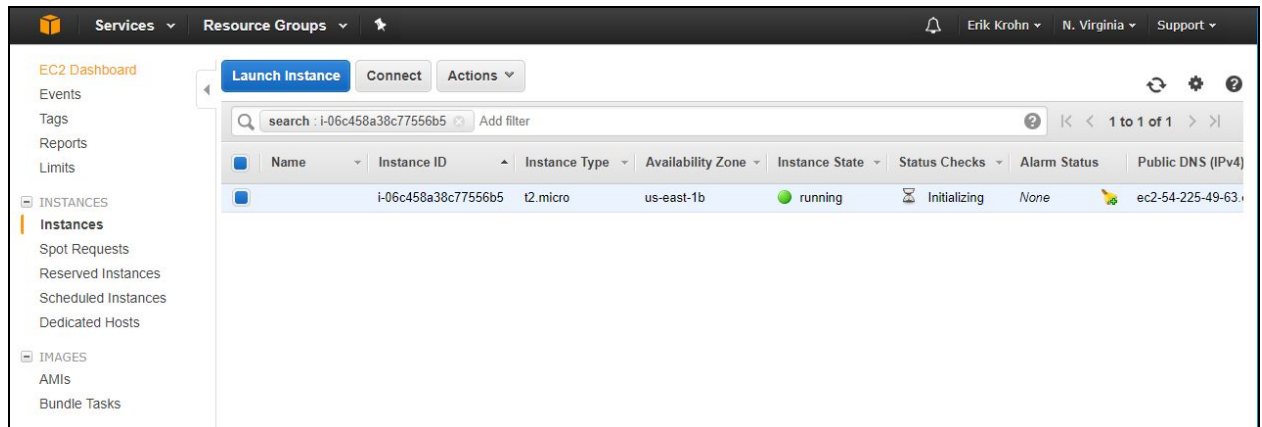
Important:

If you lose your key, you will **not** be able to access the server.

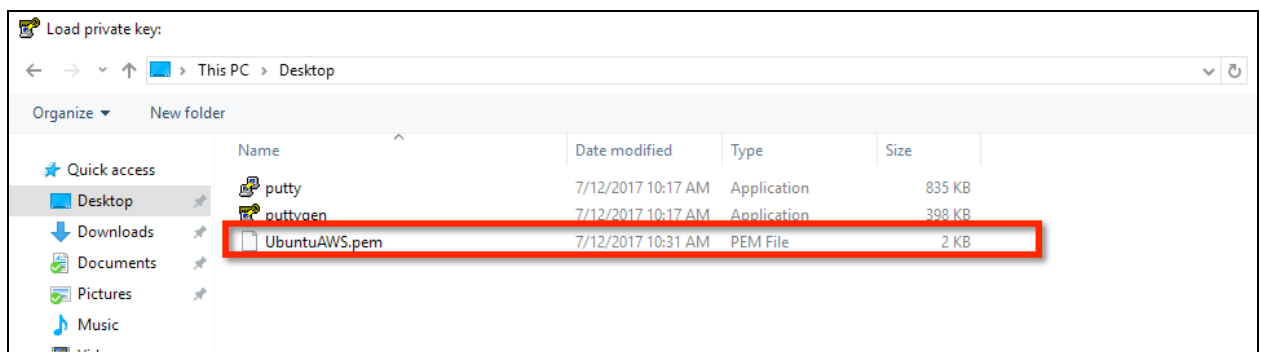
25. Click **Launch Instances**.
26. Confirm you see something like this:



27. Click the link that appears after **The following instance launches have been initiated**.
 - In the previous screen shot, the link appears as **i-06c458a38c77556b5**. Yours will have a different name.
28. Once you click that link, confirm you see something similar to this:



29. In the **Instance State** column, notice it says **running**. If it is still initializing, pending or waiting, then wait for the instance state to be in running mode before continuing.
30. Scroll to the right and find your **Public DNS (IPv4)**. Write this address down.
 - For reference, mine was called: **ec2-54-225-49-63.compute-1.amazonaws.com**
31. The .pem file downloaded in Step 24 is not compatible with PuTTY. Because of this, we need to create a key file that PuTTY can read. Open up the **puttygen.exe** file that you downloaded in Step 2.
32. Click the **Load** button.
33. By default, PuTTYgen only looks for files with extensions of ppk so you must change it to look for all files. Find your .pem file that you downloaded in Step 24, which will look something like this:



34. Click **open**.
35. PuTTYgen will give you some message about successfully importing the foreign key. Simply click **OK**.
36. In order to save the key that PuTTY can use, click the **Save private key** button.
37. The system will ask you if you want to save it without a passphrase. Click **yes**.
38. Save this **ppk** file somewhere safe.

Important:

If you lose this file, you will **not** be able to access your server.

39. Close out of PuTTYgen.

40. We are now ready to connect to our server. Open **putty.exe**.

41. In the **Host Name** section, enter **ubuntu@** followed by the IP address you wrote down in Step 31.

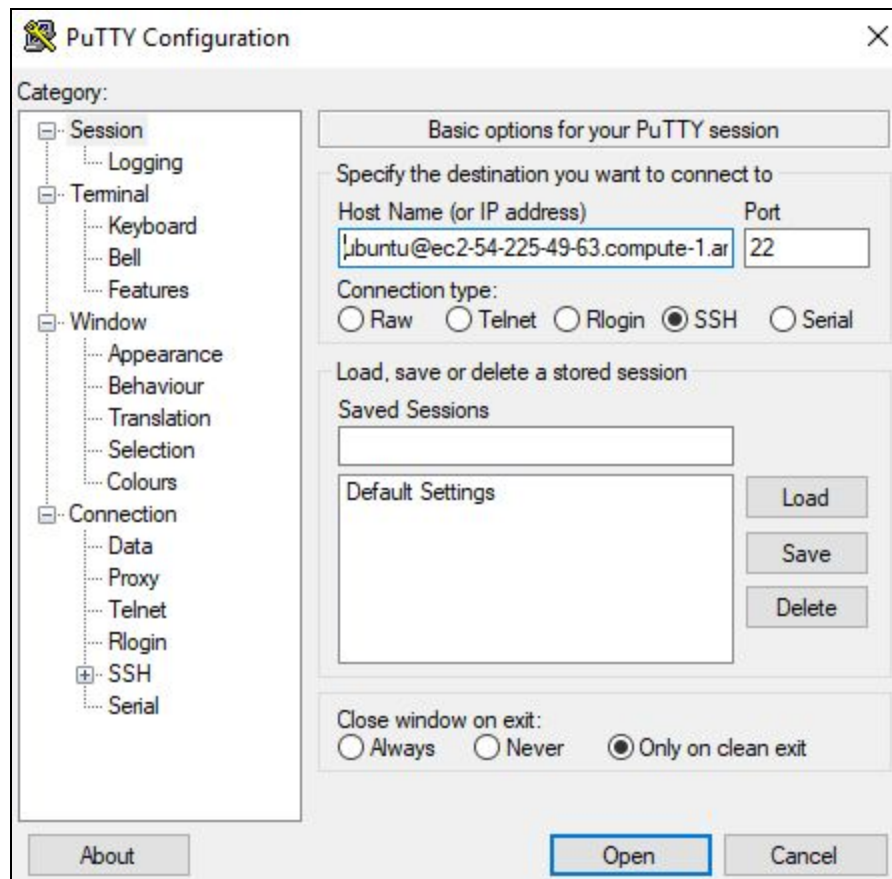
- My **Host Name** was:

ubuntu@ec2-54-225-49-63.compute-1.amazonaws.com

Important:

Be aware that every time you restart your server, you will likely get a new IP address.

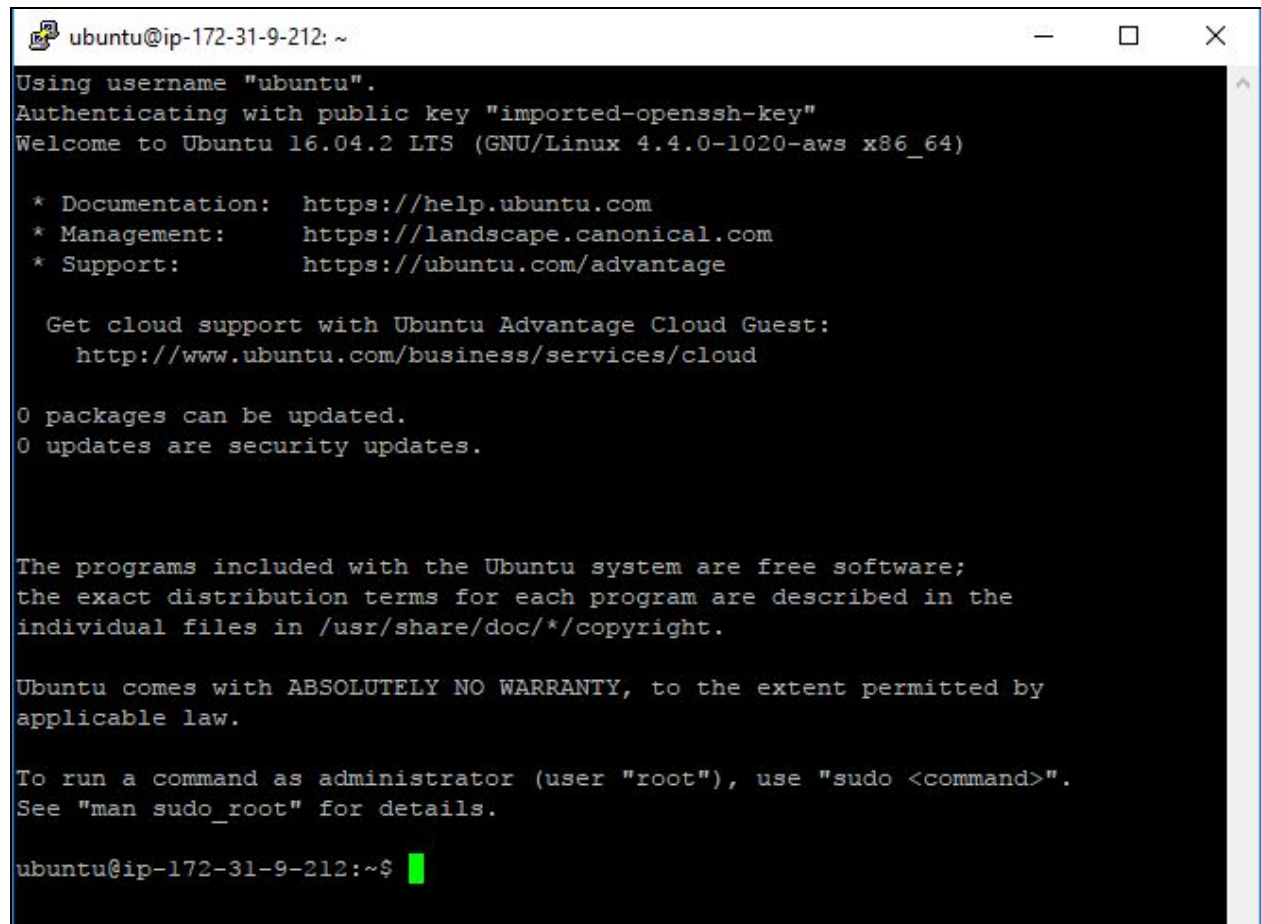
42. Enter in a **Port** of 22. You should have something similar to this:



43. On the left side, click the plus symbol next to **SSH**.

44. Click the **Auth** option.

45. Click the **Browse** button for the **Private key file for authentication**
46. Find your ppk file that you saved in Step 39.
47. Click the **Open** button.
48. When the system asks you about accepting an RSA fingerprint, indicate **Yes**.
49. Assuming everything went correctly, confirm you see something like this:

A terminal window titled 'ubuntu@ip-172-31-9-212: ~' with standard window controls. The terminal output shows the login process for user 'ubuntu' using an imported OpenSSH key. It displays the Ubuntu 16.04.2 LTS welcome message, system information (GNU/Linux 4.4.0-1020-aws x86_64), and links for documentation, management, and support. It also shows that 0 packages can be updated and 0 security updates are available. A disclaimer about warranty and instructions on using 'sudo' are also present. The prompt 'ubuntu@ip-172-31-9-212:~\$' is shown with a green cursor.

```
ubuntu@ip-172-31-9-212: ~
Using username "ubuntu".
Authenticating with public key "imported-openssh-key"
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-1020-aws x86_64)

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

Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud

0 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.

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

ubuntu@ip-172-31-9-212:~$
```

50. We will install Java as we will need it later in the course. Before we can install it, we need to update our package list. In the PuTTY terminal window, type in:
sudo apt-get update
51. In order to install Java, enter this command:
sudo apt-get -y install openjdk-11-jdk-headless
52. Enter **javac -version**
 - The system will say it is 1.8.0_XXX.
53. Make sure your software is up to date by entering:
sudo apt-get upgrade

54. If necessary, enter 'Y' to the upgrade.
55. If you get messages about updating some kind of grub file, choose the default option of **keep the local version currently installed**.

Using VIM

We will have to edit files on our Linux filesystem from time to time. If you have a favorite editor, feel free to use that. A simple one explained here is called Vim. There is a presentation that covers Linux and how to install a GUI if you want. Please view that presentation to learn how to get a graphical interface instead of the command line.

56. To edit a file, enter

vim nameOfFile

This will open up a file called nameOfFile in a program called Vim, which you can think of as Notepad.

57. Press the letter i to enter Insert mode.

Notice the word -- **INSERT** -- on the bottom. Once you are in insert mode, you can type like you normally would.

58. Add the following text to the file:

Hello, this is a test.

59. After you have added that text to your file, Press the ESC key.

You should notice the -- INSERT -- disappear from the bottom.

60. Enter the following exactly as it's written:

:wq

That's a colon, then w, then q, then the <enter> or <return> key. This will save your file and take you back to the command prompt.

61. To ensure the file was created successfully, use the following command to display it on the screen: **cat nameOfFile**.

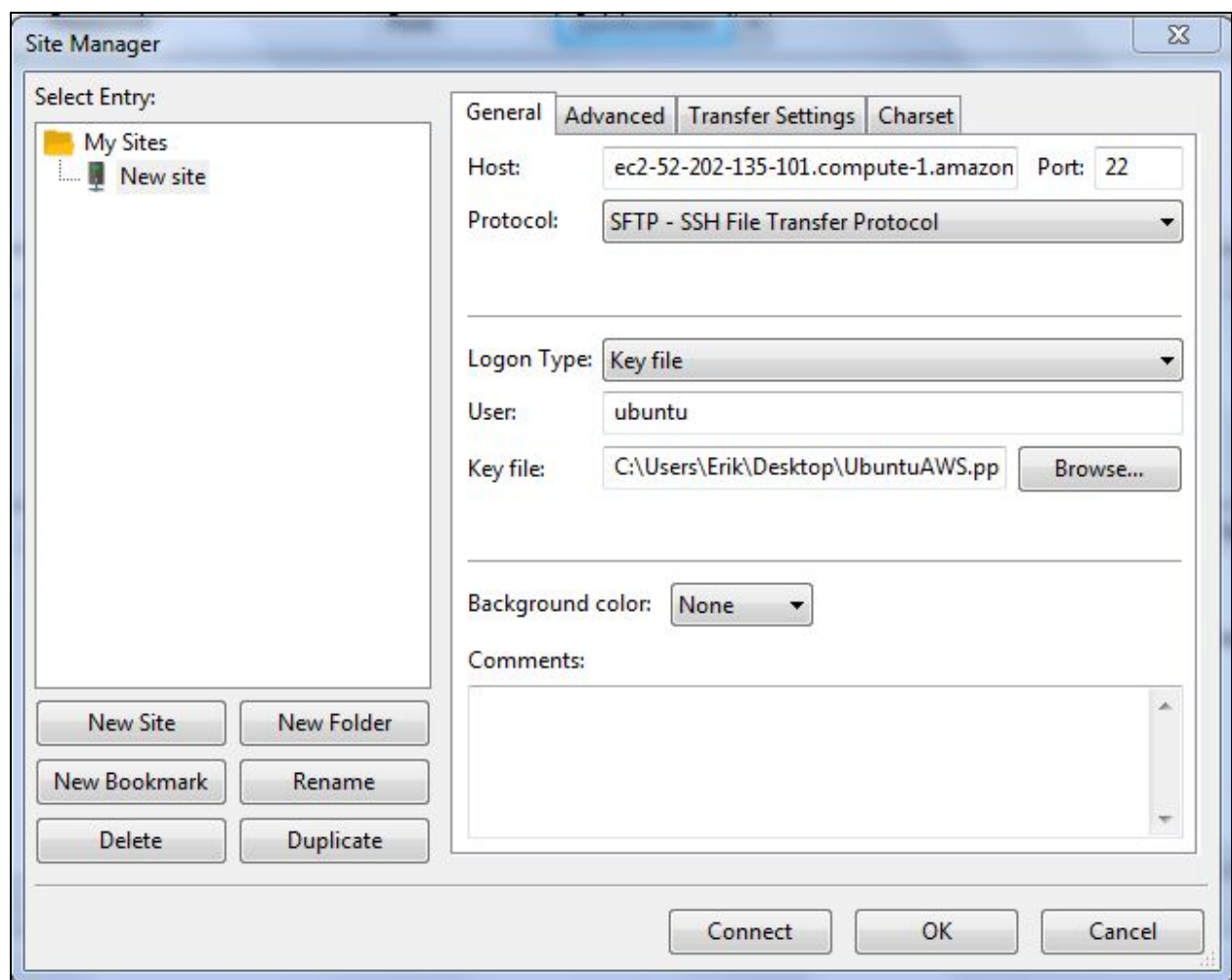
Note:

If you are struggling with Vim, search online for Vim tutorials such as <http://www.openvim.com/> and read up on the commands. Once you've learned a few of the commands, it becomes very easy to use. Another common editor is called **nano** and instructions for how to use it are in the Linux presentation.

Connect to EC2 with FileZilla

In order to connect to your EC2 instance using FileZilla to transfer files, follow these steps:

62. Download FileZilla from http://www.uwosh.edu/faculty_staff/krohne/ds730/filezilla.html and install it. The install is straightforward.
63. Open FileZilla.
64. Go up to **File > Site Manager**.
65. Click the **New Site** button.
66. Enter in your IP address in the **Host**.
67. For the **Port**, enter **22**
68. Change the **Protocol** to SFTP.
69. Change the **Logon Type** to Key File.
70. For the **User**, enter **ubuntu**
71. Click the **Browse** button.
72. Find the .ppk file that you created using PuTTYgen previously.
73. Confirm that your screen looks something like this:



74. Click **OK**.

75. Directly below File, find this icon: . Click the dropdown box and select **New Site**.

76. If the system asks you to trust the key, simply indicate **OK**. You should be connected.

Note:

Whenever you need to reconnect to your EC2 instance, go to **File > Site Manager** and update the IP address (i.e. **Host**) as needed.

Test Connection and Stop Instance

77. To test what you just did, type **python3** and the python interpreter should load and you'll see something like this:

```
ubuntu@ip-172-31-86-162:~$ python3
Python 3.5.2 (default, Nov 23 2017, 16:37:01)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

78. To quit out of the Python interpreter, type **exit**.

79. When you are finished, enter **exit** to quit your session.

Important:

You now need to go back to the AWS console page and stop your instance. This is especially important if you didn't chose the t2.micro instance. You don't want to pay for idle time if you are not connected to your machine.

80. In order to stop your instance, first click your instance.

81. Go up to the **Actions** button and go down to **Instance State**.

82. Click the **Stop** option.

83. You might see a box that says something about losing ephemeral storage. This is fine; click **Yes**. Wait on this page until the instance is stopped.

84. This is not a step that you need to complete right now but is being provided for the future. If you want to change your instance type, this is the webpage where you do it (when the instance is stopped). I recommend using t2.2xlarge or above to see the power of multithreading near the end of the semester. It's a CPU issue; smaller

instances do not provide multiple CPUs. The t2.2xlarge instance provides 8 CPUs and 32GB of memory. The t2.micro instance, when you are not in the free tier, costs about 1 cent an hour. The t2.2xlarge instance, by contract, is about 38 cents an hour. In order to change your instance type, ensure that your instance is stopped. Right click on the instance you want to change. Choose **Instance Settings** and choose **Change Instance Type**. Choose the instance type you want and hit **Apply**.

85. **At the end of the semester**, make sure you terminate your instance. An instance that is not running will not cost anything in processing time. However, you will still be using 30GB of storage with the instance. This storage is not free after the 12 month free tier is over. Be sure to terminate your instance(s) at the end of the semester.

Important:

During the semester, I recommended *stopping* as you will be able to restart your instance as it were without having to reconfigure anything. If you *terminate* it each time throughout the semester, you will lose everything and have to redo everything.

Be aware that when you stop your instance, you lose your IP address. In order to restart your instance:

1. Come back to this page.
2. Click your instance.
3. Click **Actions**, **Instance state** and then **Start**.

Every time you start your instance, you'll likely have a new IP address.

Task 5: Testing Java

1. Connect to your EC2 instance in the cloud using PuTTY.
2. Create a folder called **JavaExamples**. To do this, type in **mkdir JavaExamples**. To enter that directory, type in **cd JavaExamples**.
3. Create a file called **Test.java** in that folder (see VIM from earlier).
4. Enter the following code into that file³:

```
public class Test{
    public static void main(String args[]){
        System.out.println("It works!");
    }
}
```

³ In general, copy and paste is safe with the notes unless there are quotations.

5. When you are back at the terminal window, compile your program by entering
`javac *.java`
6. In order to run the file, enter `java Test`
 - You should see **It works!** printed to the screen.

Task 6: Test Python

It is assumed that you have some generic programming skills before starting this course. You should be familiar with the following topics before starting this course:

- Selection Statements (e.g. if)
- Repetition Statements (e.g. while, for)
- Variables
- Lists/Arrays
- Calling Functions/Methods
- Creating Functions/Methods

If you do not know what one of those topics is or do not have a good grasp on how to use each one, you might want to reconsider whether you are prepared for this course. We will write a few short Python programs to test out our Python installation and also to assess your programming ability. You must test your code on your EC2 instance using the command line. If you find yourself taking many hours to solve these tasks, then it might be best to take a refresher programming course before continuing with this course.

Important:

- You are **not** allowed to use any external libraries (e.g. numpy, sympy, itertools etc.) for these problems as they make the problems trivial. If you are importing something other than sys, then you are not doing this correctly.
- Make sure to follow the directions **exactly** as my tester code will not work if you don't. For example, in the first problem, if you do not call your file **first.py** or create a function called **fact** instead of **factorial**, my automated tests will fail. Also, be sure to output exactly what the problem asks for and nothing else. For example, for the factorial problem, only output the actual factorial number. If the answer is 720, only output **720**. Do not output something along the lines of: "The factorial is 720." The final Python problem addresses the importance of creating appropriate output.

Run Python program from terminal

Many of you may be unfamiliar with the terminal (command line) and we will be using it for a portion of this course. The following short instructions describe how to create a Python file using only the terminal and how to run your Python code from the terminal.

1. Connect to your EC2 instance in the cloud using PuTTY. All
2. Create your Python programs using vim as described before. You may also create your **first.py** file locally and transfer it over to your EC2 Linux machine using Filezilla.
3. Once you have created your **first.py** program, go back to the terminal so that you can test your code.
4. In order to run your program, enter:
python3 first.py

Create a Factorial Calculation Program

You will be creating a program that calculates the factorial of a number.

1. Create a file called **first.py**.
2. Inside that python file, create a function called **factorial** that accepts a number as an argument and returns the factorial of that number. Do not print out anything in the factorial function. Your factorial function should work as expected no matter what you write in future steps. If the factorial function is called with a negative number, a -1 is returned from the function.
3. Make sure your function definition looks like this:
def factorial(val) :
4. Inside your python file (but outside of the factorial function), prompt the user to enter in an integer.
 - You can assume that the value being entered is an integer but you should not assume that it is greater than or equal to 0. If the number entered is less than 0, prompt the user again until the number entered is greater than or equal to 0.
5. Once you have a number that is greater than or equal to 0, call the factorial function and print out the answer returned by the factorial function.

Output Average of Integers

Create a program that reads in integers and outputs the average of all of the numbers. Your program will be called using the following command:

python3 second.py < someInputFile

Some things to note:

- Only integers will appear in the input file.
- All of the integers in the input file will be separated by a space.
- Create a separate file called second.py to solve this problem.
- Do not read in from a specific file. If you are using the open function, you are doing this problem wrong.

Output Prime Numbers

The goal of this task is to create a Python program that prompts the user to enter 2 numbers. Your Python code must be stored in a file called **third.py**. Your program then prints out all of the prime numbers **strictly** between those 2 numbers in increasing order. If there are no prime numbers strictly between those 2 numbers, then a **No Primes** message is printed out. The order that the numbers were entered does not matter.

Output: If there are no primes between the two numbers, your code outputs exactly:

No Primes

If there is one prime number, only that 1 number prints out.

If there are multiple prime numbers, the following pattern is used:

firstNum:secondNum!thirdNum&fourthNum:fifthNum!sixthNum&seventhNum

In other words, you separate the first number from the second number with a colon. You separate the second number from the third number with an exclamation point. You separate the third number from the fourth number with an ampersand. This delimiter pattern is repeated in subsequent numbers (colon, exclamation point, ampersand) until there are no more numbers left to be printed. There is no delimiter before the first element nor is there a delimiter after the last element.

Example:

For example, if **5** and **24** were entered in that order, then **7:11!13&17:19!23** would print out.

If **24** and **5** were entered in that order, then **7:11!13&17:19!23** would print out.

Interpret Dirty Data

Throughout this course, your output will be tested using automated testers. Being able to test your code quickly allows us more time to dig into your code and provide better feedback. If we have to spend a lot of time trying to figure out what your output is or how to run your program, then this cuts into the time we have to look at your code.

Therefore, it is important that your code is contained in the files we specify and produces the exact output we expect. Having incorrect delimiters, incorrect spacing, etc will cause our automated tests to fail. Be sure you read the output specifications closely so that you are producing the correct output. The output of the previous problem is quite arbitrary (although it assesses good problem solving skills) but this is done to ensure you are prepared for future output requirements. Future output requirements will be much more natural and obvious.

To show you why it is helpful to have all output be the same, consider the following problem. You are a manager for a global weather company and you have asked your regional offices to send you information in the following comma separated format:

Year,Month,Day,TimeCST,TemperatureF,WindMPH

All offices put a header row in each file to explain the data. A key thing is that the temperature should have been in Fahrenheit and located in the second to last column. Another key thing is that the wind should have been located in the last column and should have been in miles per hour. None of the offices sent the data exactly how you asked for it. Your job is to figure out what each of them did and then answer the following 2 questions:

1. Take the average of all of the wind speeds for March, 2006. Which city had the closest wind speed to 8.30 mph?
2. Take the average of all of the temperatures for 2006. Which city had the closest temperature, in fahrenheit, to 49.65.

You can download the 4 files from

http://www.uwosh.edu/faculty_staff/krohne/ds730/a1Weather.zip.

The 4 cities are ABC, KLM, PQR and XYZ (see filenames). The way you clean the data and obtain the answers is entirely up to you. You only need to create a text file called **answers.txt** that contains the answer to each of the above questions. You do not need to upload any code or explain how you obtained your answers.

Task 7: Submitting your Work

We want to transfer the files to your local machine and zip them up for submitting. Once you have finished writing your Python code, you want to transfer them to your local machine. This can be done with Filezilla. In order to transfer files:

1. Open FileZilla.
2. Click **File > Site Manager**.
3. Click **New Site**.
4. Enter in your host name. As a reminder, mine was **ec2-54-225-49-63.compute-1.amazonaws.com**. Yours will be different.
5. Set the **Port** to **22**.
6. Change the protocol to **SFTP**.
7. Change the **Logon Type** to be: **Key file**.
8. For the User, enter **ubuntu**
9. Click **Browse**.
10. Find your .ppk file.
11. Click the **Connect** button.
12. There may be an option to save the password. You can choose whatever you want here.

You will see your local machine's files on the left side and your EC2 instance files on the right side. You can transfer files now. You should have a total of 3 Python files: **first.py**, **second.py**, and **third.py**. You should also have an **answers.txt** file. Zip up these 4 files into 1 zip file called **a1.zip** and submit **a1.zip** to the dropbox on d2l.

Task 8: Introduce Yourself

Lastly, go to the online discussion board on Piazza (use the Introductions thread that is pinned at the top left of the page) and enter in some information about yourself. Since this is not a face-to-face course, I will likely never meet most of you. However, I would still like to know something about you. You can put whatever you want in that post. Tell me who you are, why you are interested in our data science degree, any interests or hobbies, what you hope to learn from this course, whether or not you signed up for AWS, etc. You can write as much or as little as you want. If you have questions for me, feel free to add those and I'll answer them directly. I know you've probably had to do something similar in other courses so feel free to copy and paste that "intro document" so I can get to know you a little bit.