Pecos

Why use it?

With a study as important as this one, we want to ensure that our data is of the utmost quality. Pecos, a product of Sandia National Laboratories, is well renowned as a staple in the quality assurance sector of photovoltaic systems. In our study, large quantities of data are being collected every minute which means that it is highly impractical to manually assess every collection. That is where Pecos comes in. According to its documentation, “Pecos is an open-source Python package designed to conduct automated quality control and performance monitoring which can allow system operators to quickly detect performance issues.” This is exactly what we were looking for.

How to use it?

Reference “Pecos Guide” Below

How it works?

Pecos is a Python Package. That means that it is a large collection of various scripts and files that work together in order to accomplish a task. While Sandia Labs designed this code in order to conduct data quality control and panel efficiency, our primary concern is purely on data quality control. Therefore, large portions of the original script have been removed in order to only conduct necessary evaluations. Furthermore, its primary function is to serve as a quality assurance measure to flag erroneous data and generate easy to read reports.

Table of Contents

Step 1: Download and Install Oracle VM VirtualBox....2
Step 2. Download Ubuntu Desktop....3
Step 3. Creating a New Virtual Machine....4
Adding a VM....15
Launching a VM....21
Changing a VM's Settings....22
Step 4: Running Virtual Machine....25
Powering Off/Restarting VM....29
Adding Administrator Privileges....30
Step 5: Install Anaconda on Virtual Machine....36
Step 6: Install Needed Packages....45
Step 7: Download Modified Pecos zip file on Virtual Machine....59
Step 8: Launch Anaconda Navigator....60
Step 9: Change View and Resolution of VM....61
Step 10: Launch Spyder....64
Step 11: Opening Quality Control Script....65
Step 12: Specify Data File of Interest....69
Step 13: Add and/or Remove Grouped Sensors....70
Step 14: Change Hours of Interest....71
Step 15: Specify Frequency of Timestamp....72
Step 16: (Filter 1) Change Expected Ranges for Sensors....73
Step 17: (Filter 2) Check for Stagnant Data....74
Step 18: (Filter 3) Change Abrupt Data Spike Parameters....75
Step 19: Quality Control Index....76
Step 20: Running Script....77
Step 21: Evaluate Report....78
Step 22: Address any Issues (Maintenance)....81
Areas for Improvement (Literature Suggestions)....82
Other Lines of Interest in Code....82
References....84

Pecos Guide

Step 1: Download and Install Oracle VM VirtualBox

Visit: **https://www.virtualbox.org/**

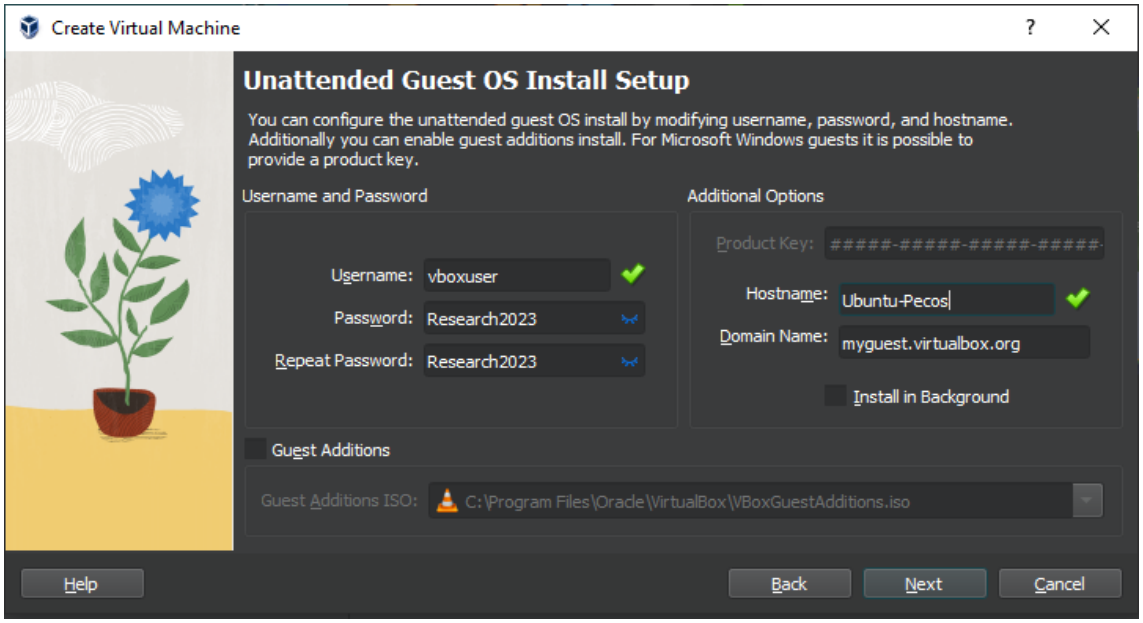
You should now be seeing a window like this:

Click the Download button to begin downloading VirtualBox:

Once the download has finished, allow the program to install.

(* If provided with VM Disk Image, Go to Launching VM section of Step 3 (Follow steps on how to launch the VM, skip to Step 5, and use login information below. *)

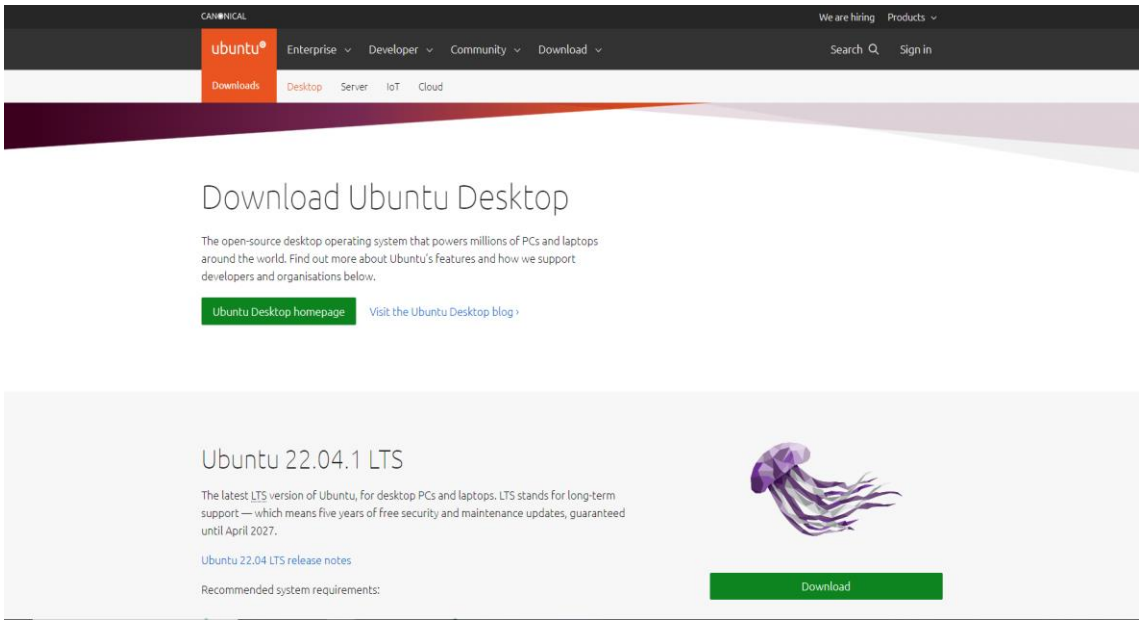
Login Information:



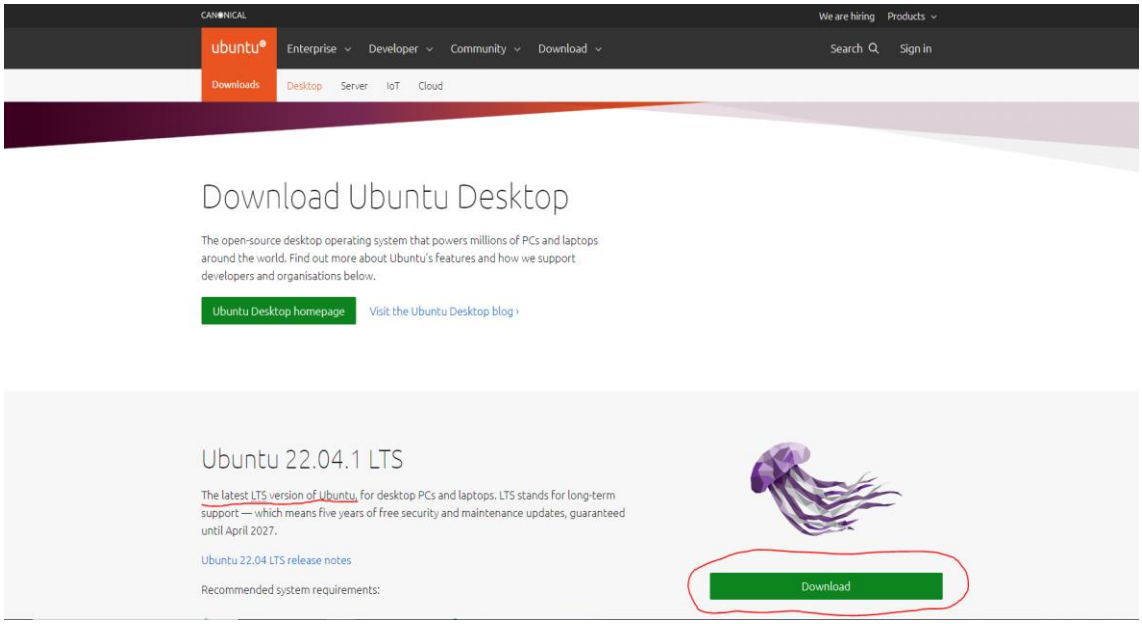
Step 2. Download Ubuntu Desktop

Visit: **https://ubuntu.com/download/desktop**

You should now be seeing a window like this:



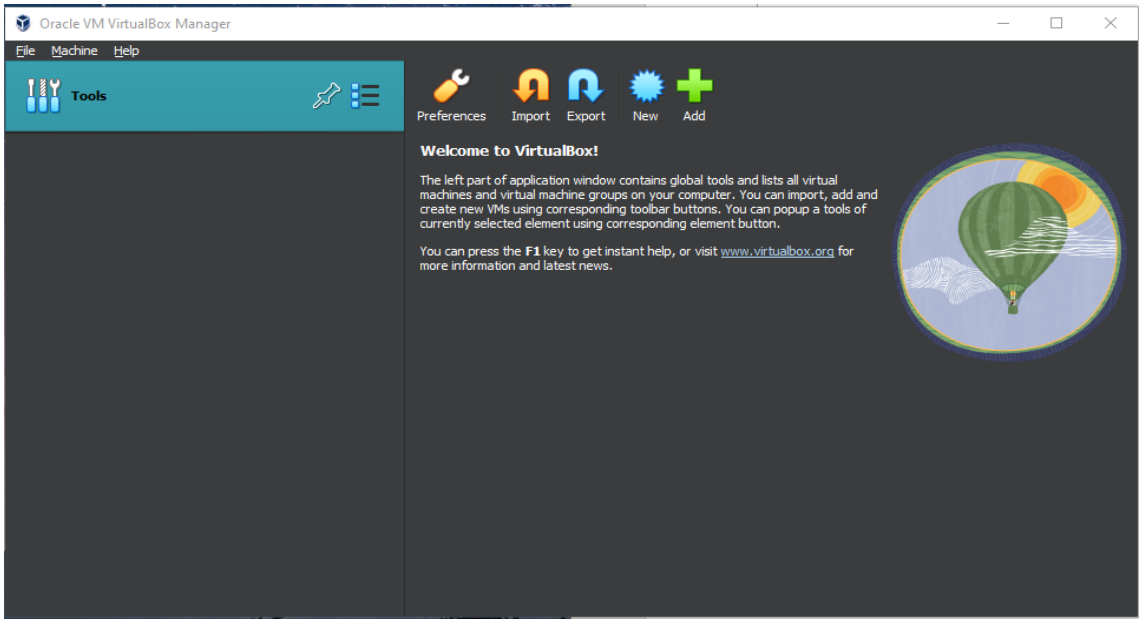
Click the Download button on the latest version to begin downloading Ubuntu Desktop:



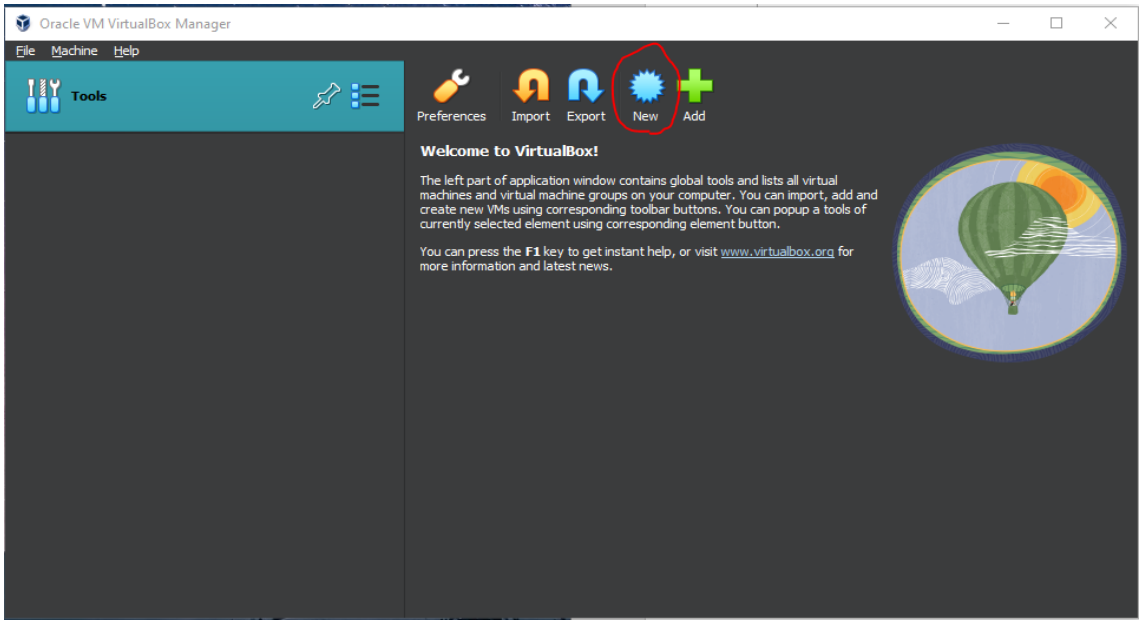
Once the download has finished, be sure to unzip the file. (Right click on zip folder and select *Extract All*.)

Step 3. Creating a New Virtual Machine

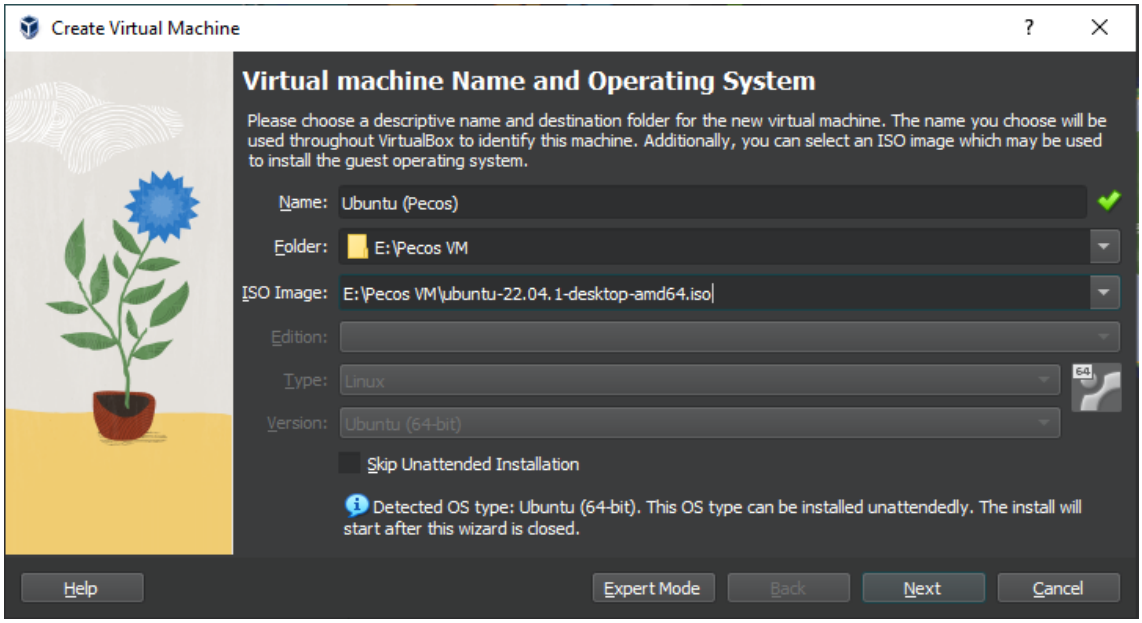
Launch Oracle VM VirtualBox. Once launched, you should see a window like the one below:



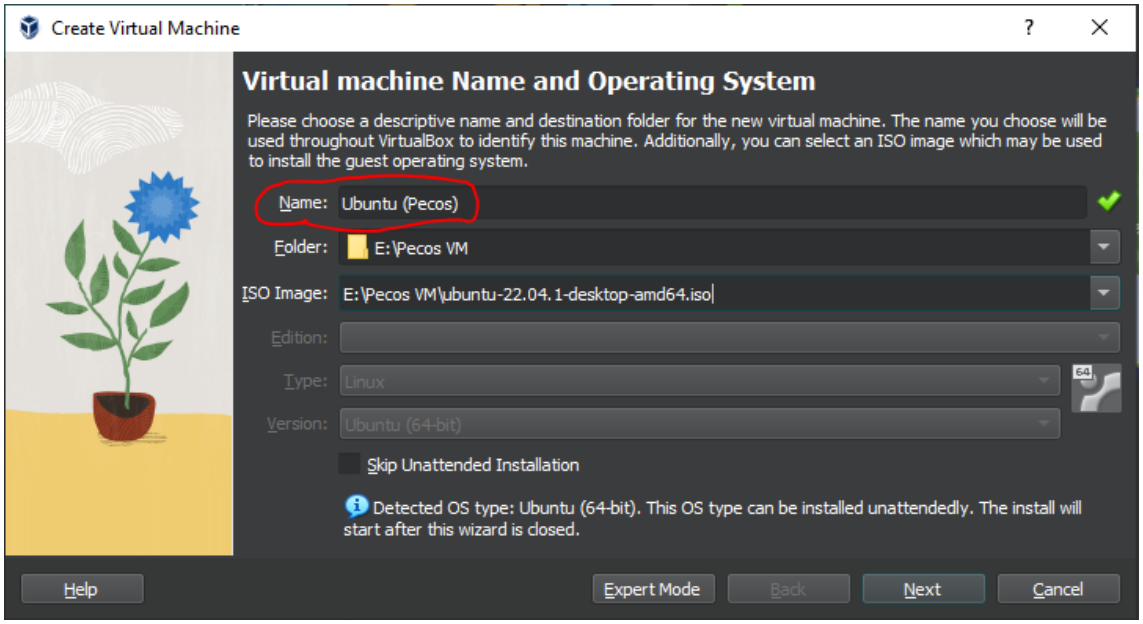
To create a new VM select NEW:



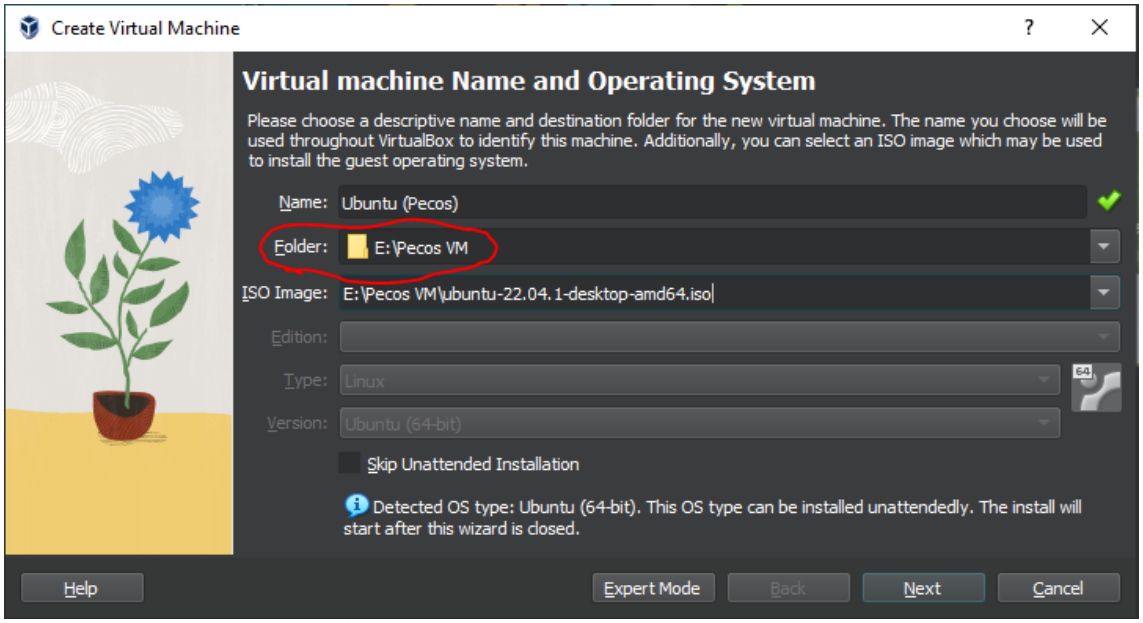
Now you should be seeing a window like the one below:



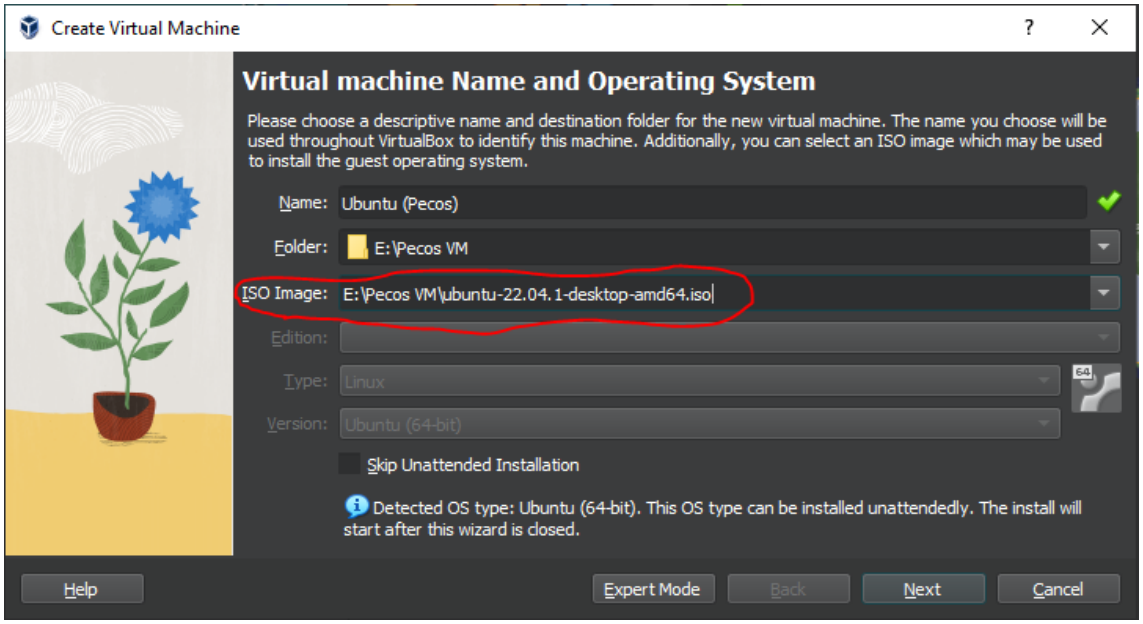
Input the name that you want to assign to the new VM:



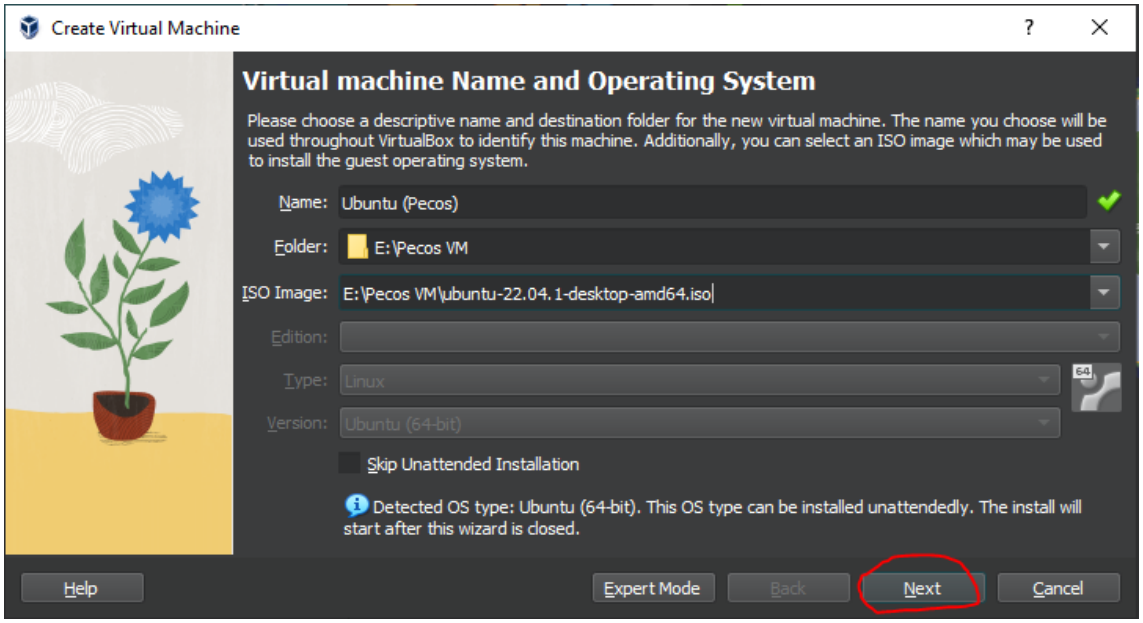
Designate the folder that you want the VM to reside within:



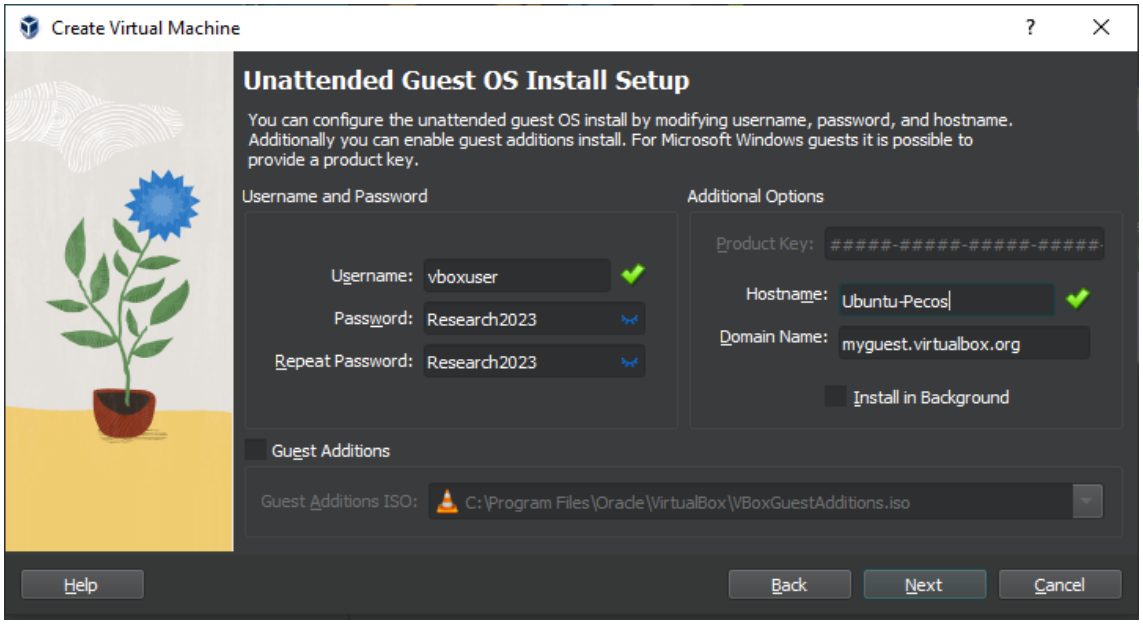
Select the ISO Image (*This is the Ubuntu Desktop that we downloaded in Step 2*):



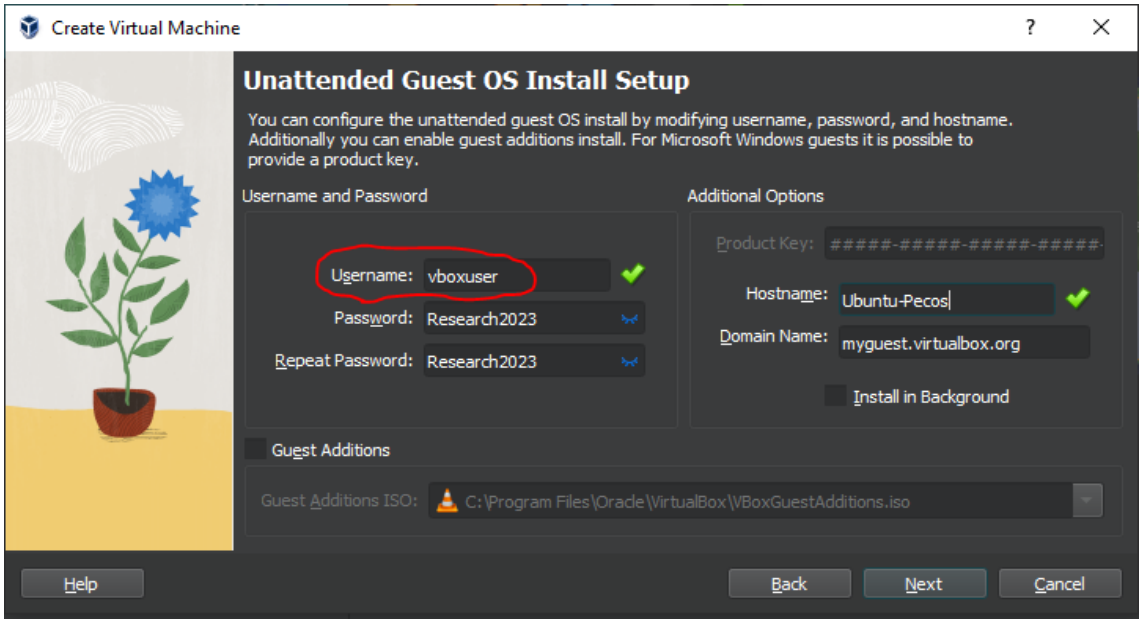
Now hit Next to go on to the next step:



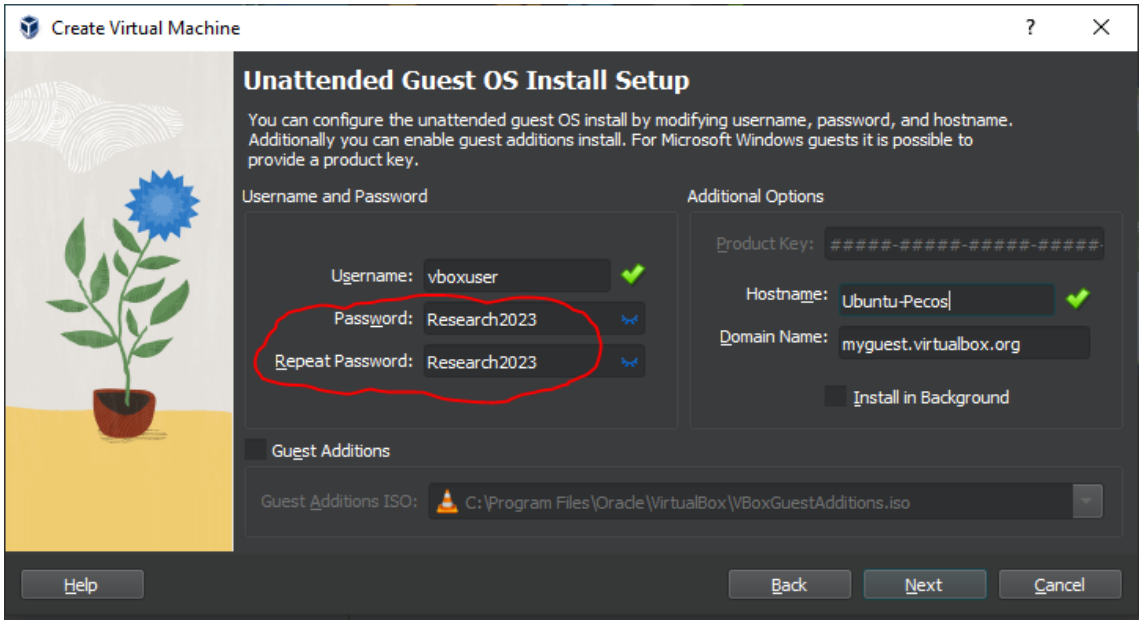
You should now be seeing a window like below:



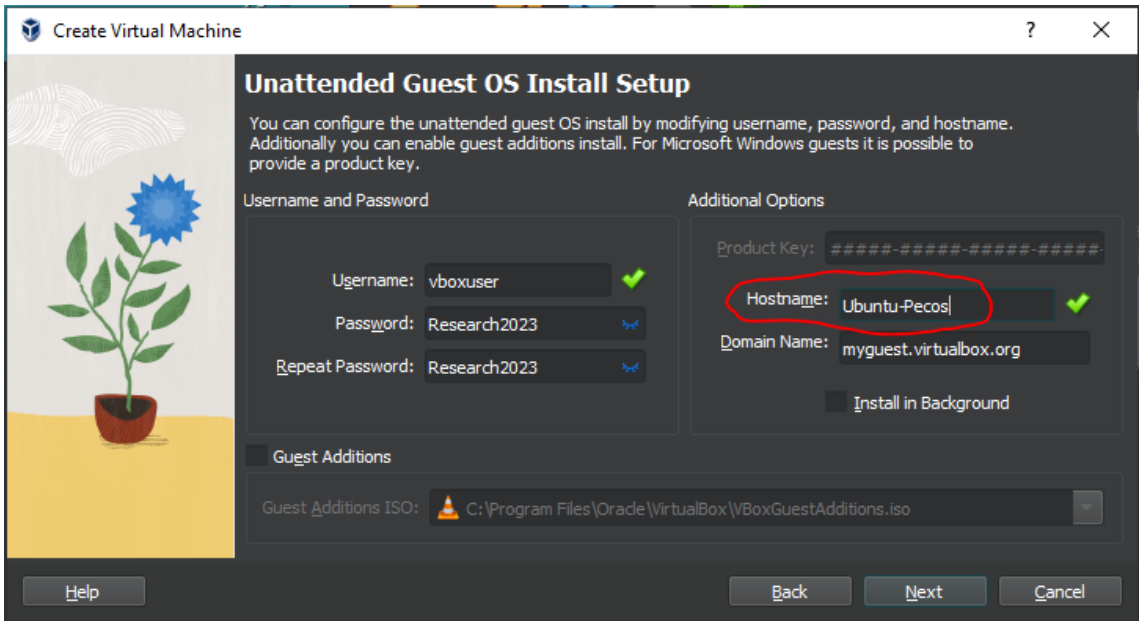
Here we can designate the Username that will be associated with the system:



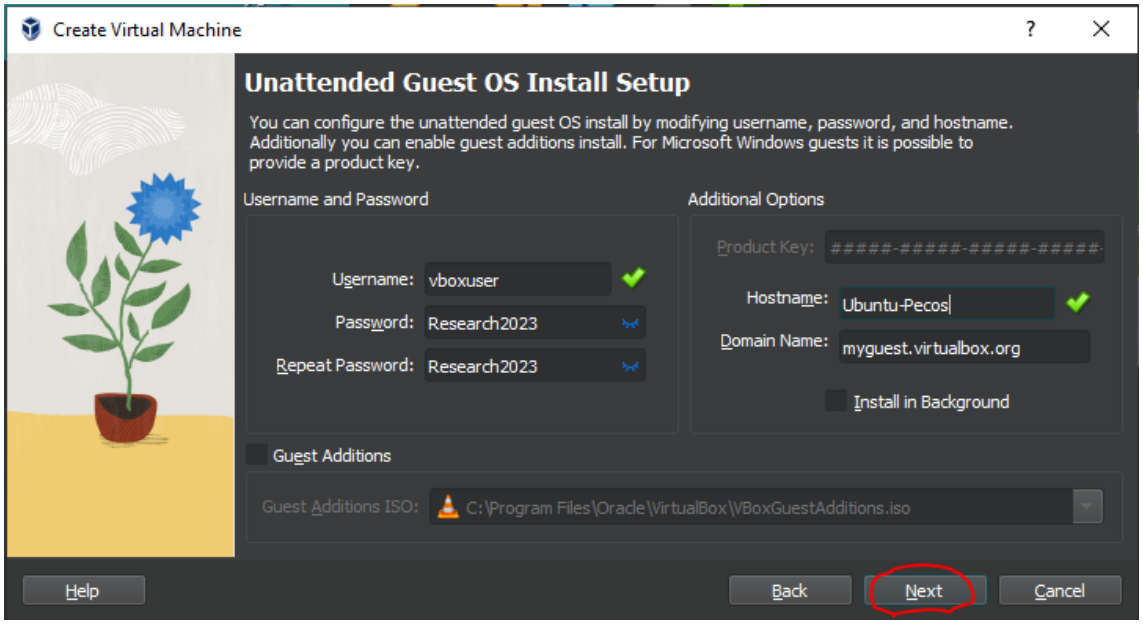
Setup the Password:



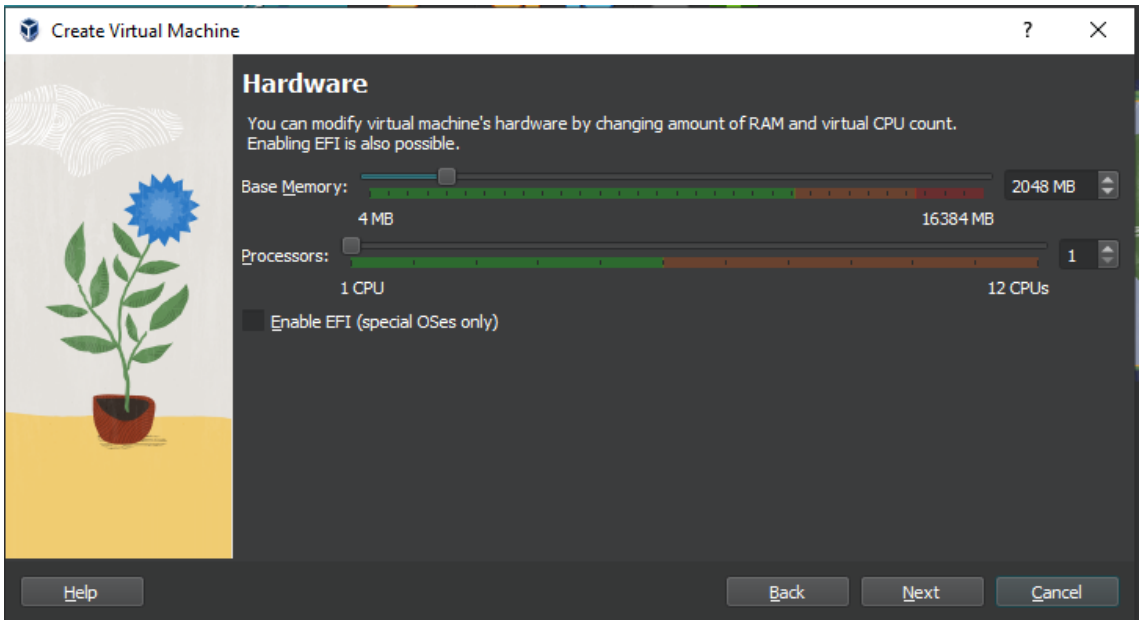
Setup the Host Name (*This will default to the name of the VM which may need to be changed due to the presence of invalid characters*):



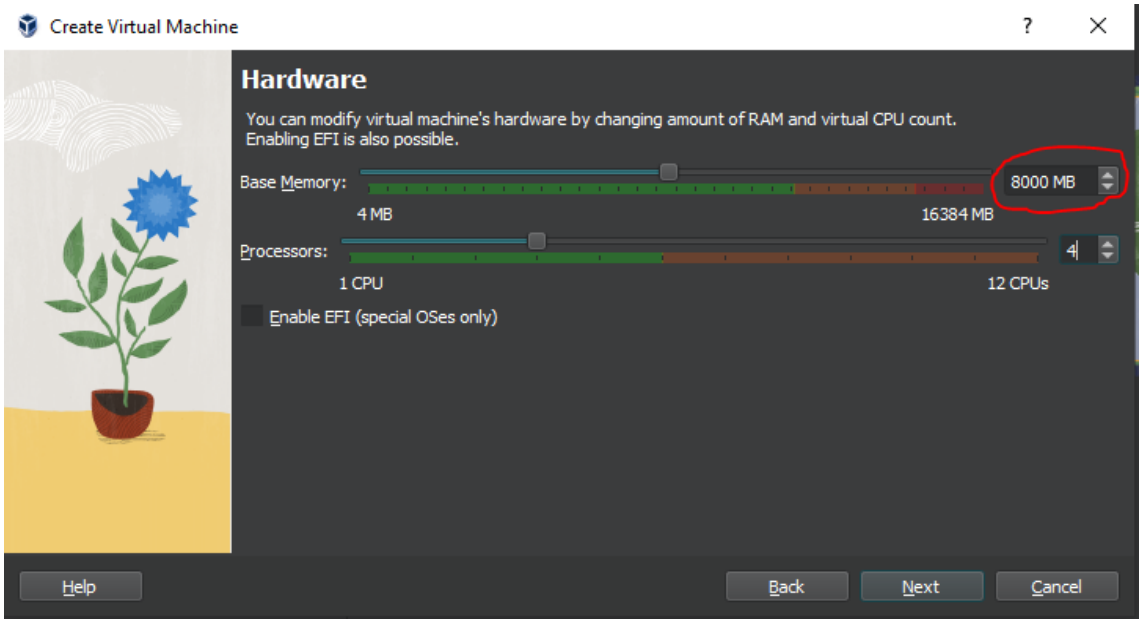
Now hit Next to go on to the next step:



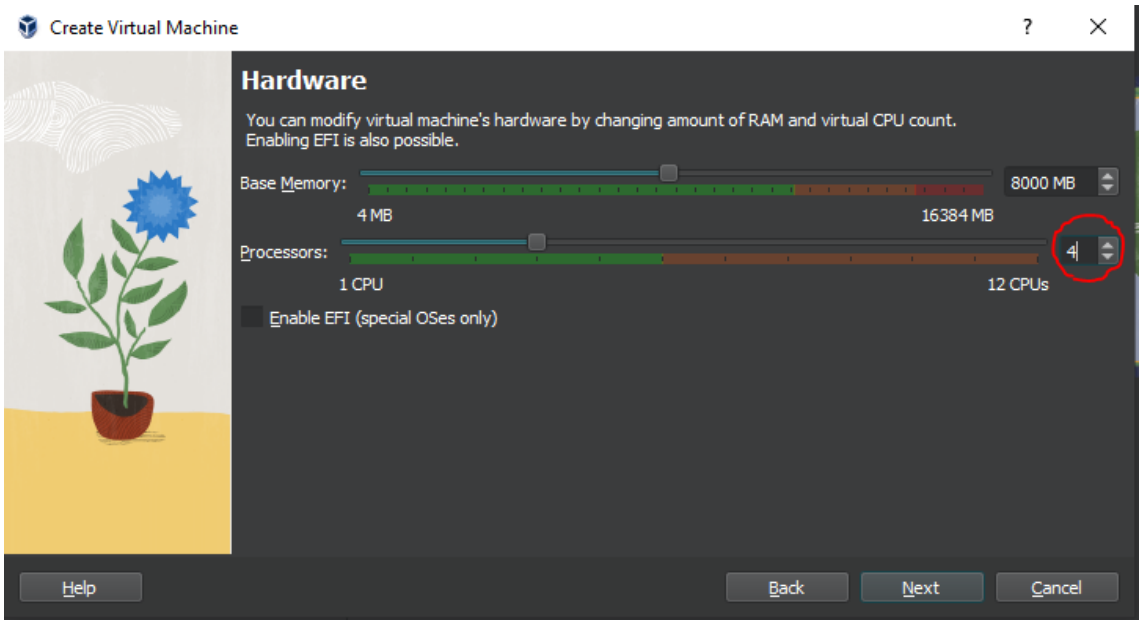
You should now have a window like the one shown below:



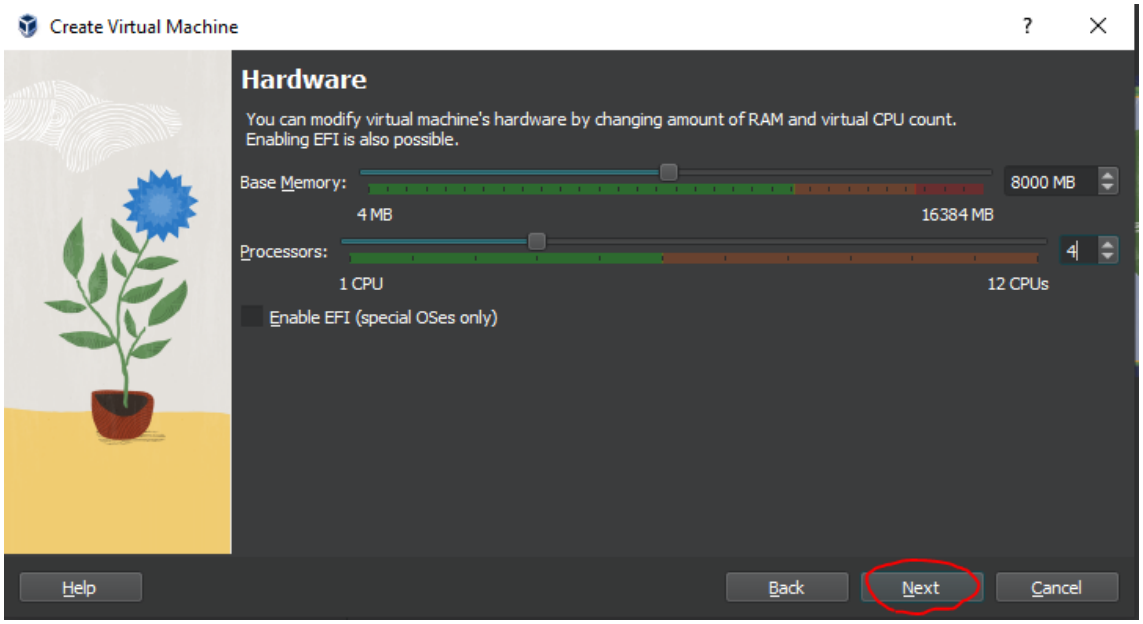
Here we can change the amount of RAM allocated to the system. I recommend using at least 8 GB (8000 MB) of RAM:



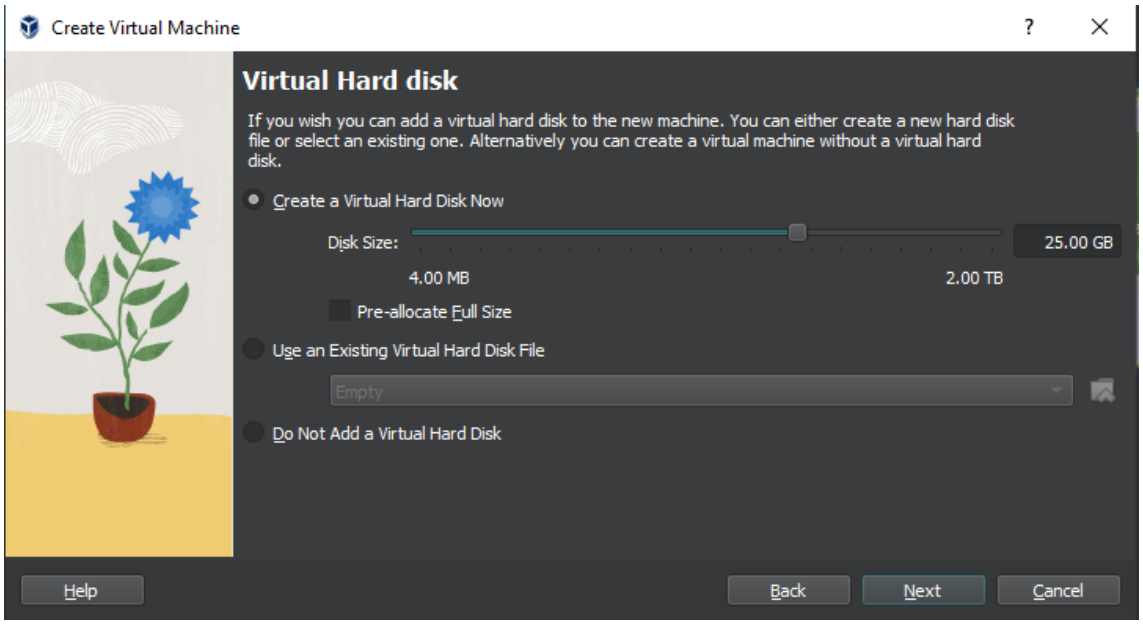
We can also designate the number of CPUs that we want to allocate to the VM. I recommend using 4 CPUs:



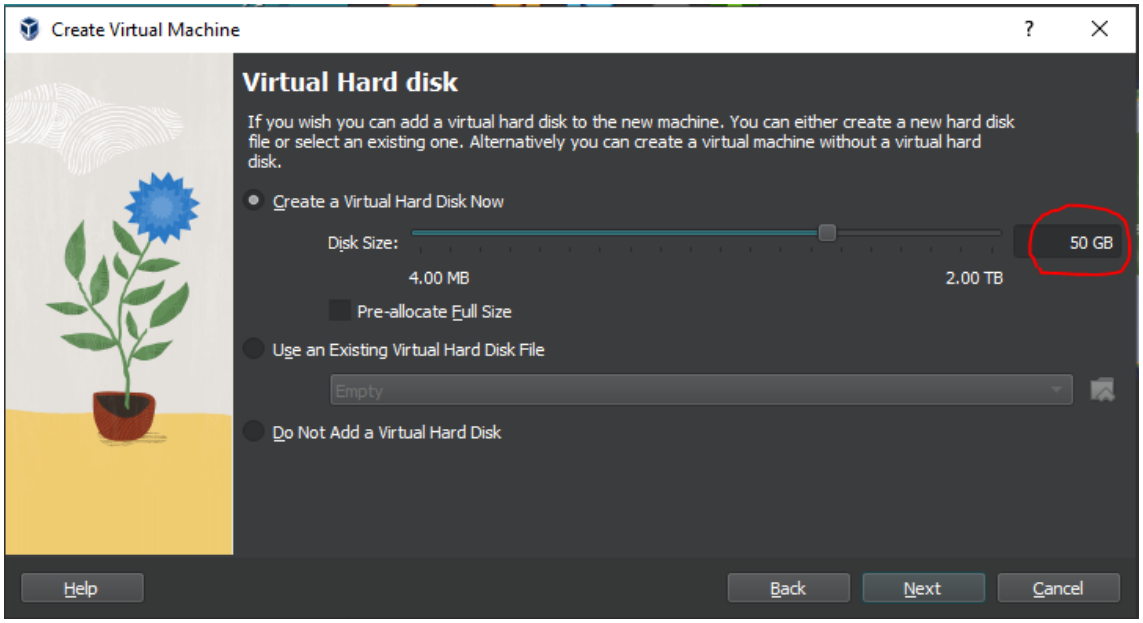
Now hit Next to go on to the next step:



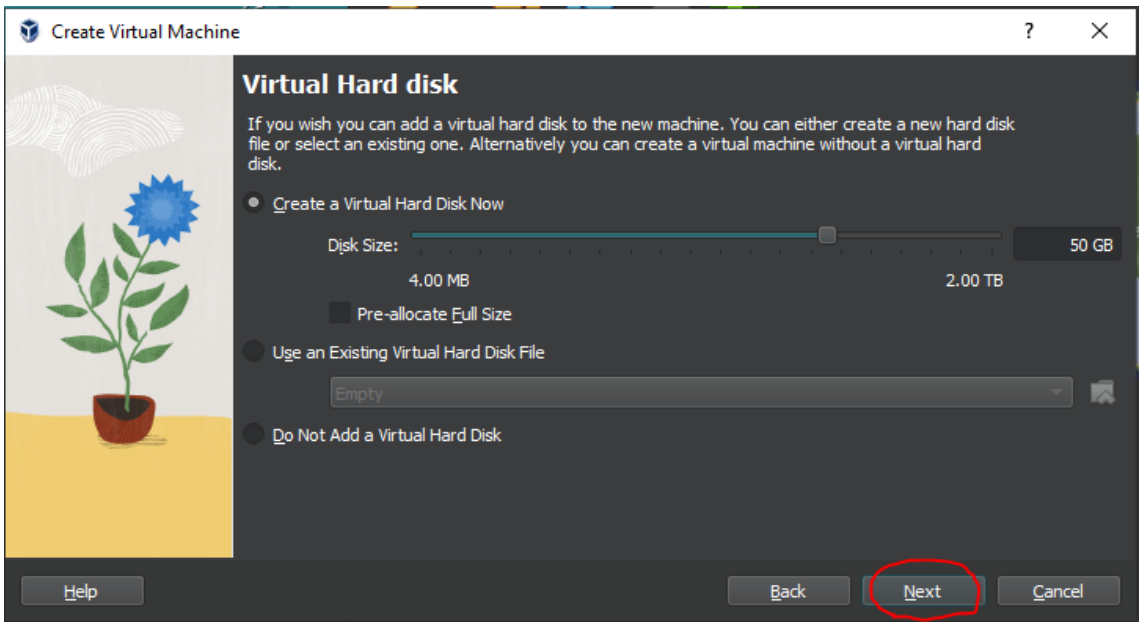
You should now have a window like the one shown below:



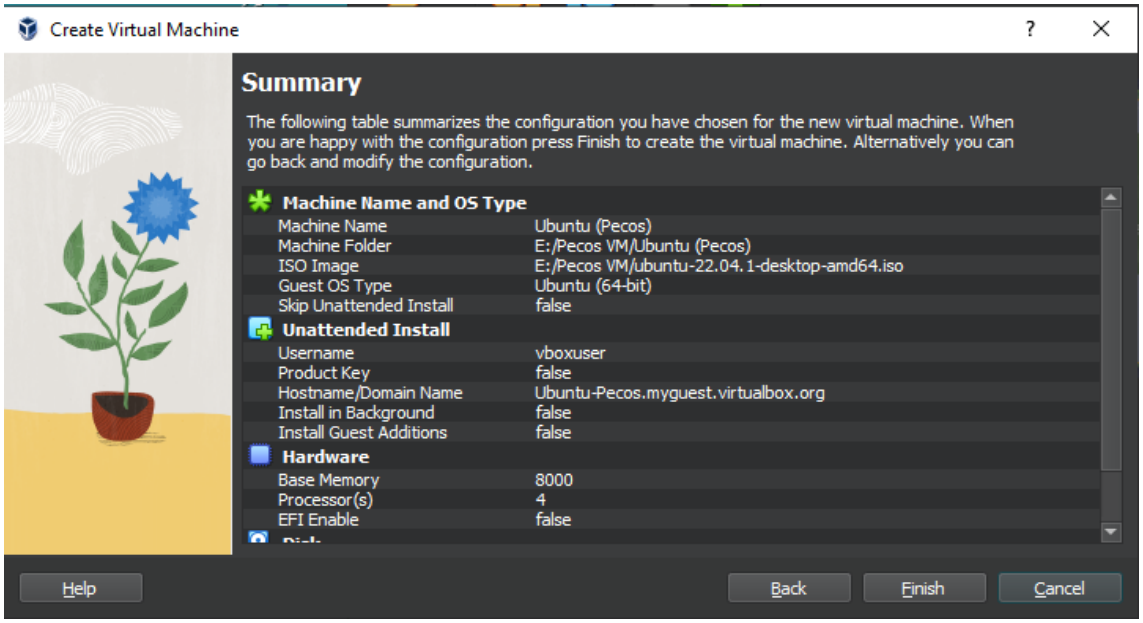
Here we can allocate the amount memory that we want available for the machine. The previously allocated 25 GB should be sufficient for our task, but I recommend allocating 50 GB or more if you are planning on doing anything else with the system:



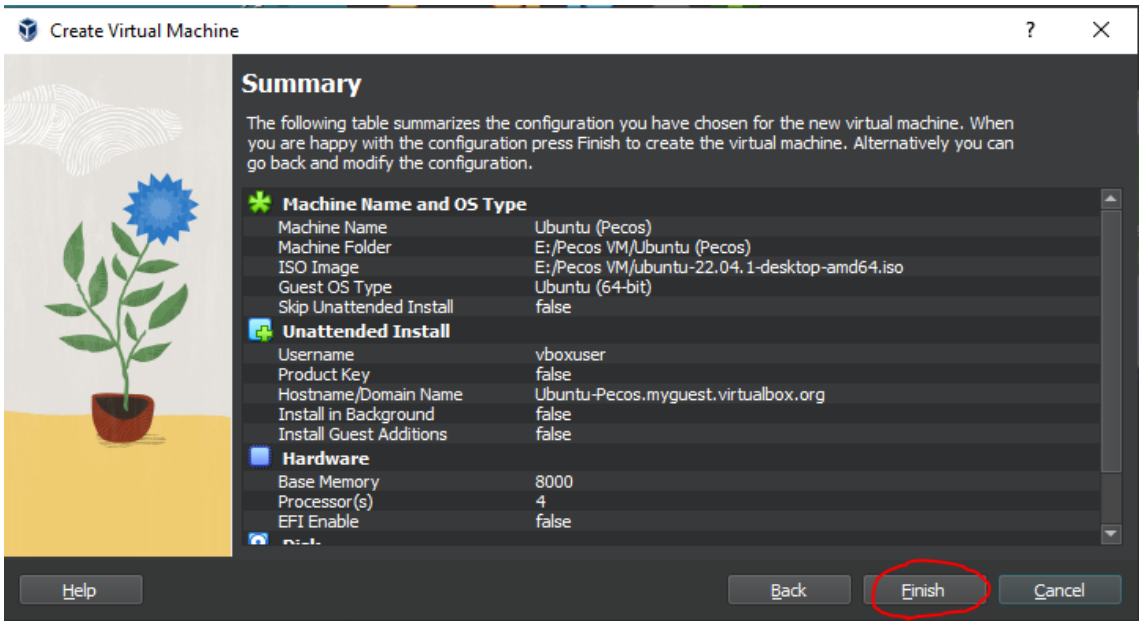
Now hit Next to go on to the next step:



You should now see a window like the one shown below. This allows you to scroll back through the configuration that you have selected. If any changes need to be made, do it now:



If no changes are needed, then hit finish to complete the creation of the VM:

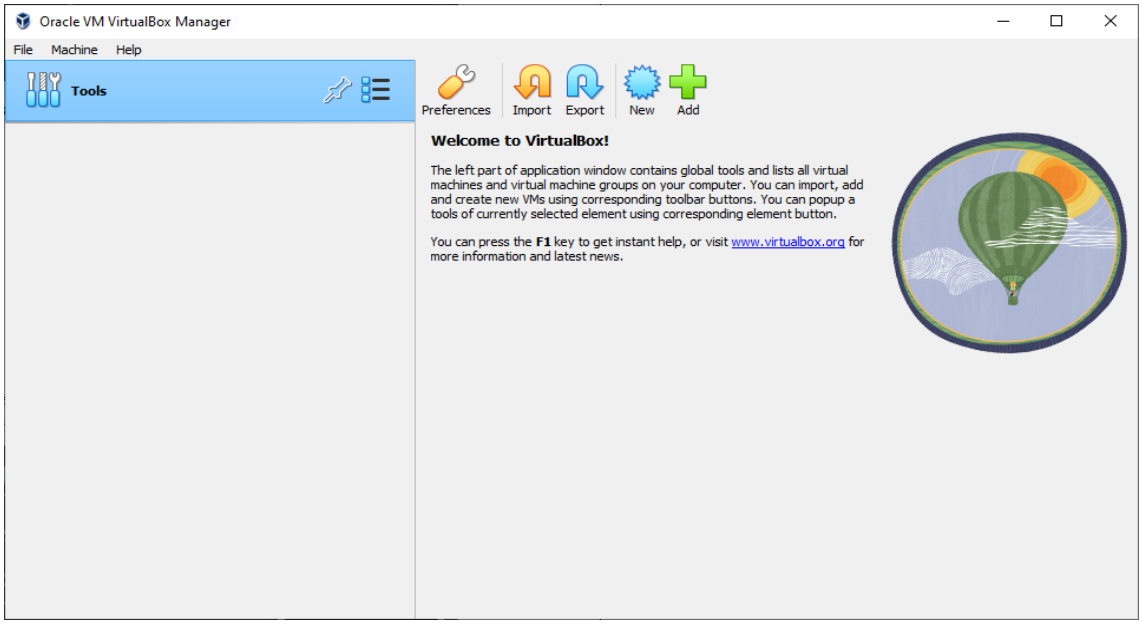


After clicking finish, the VM should automatically launch.

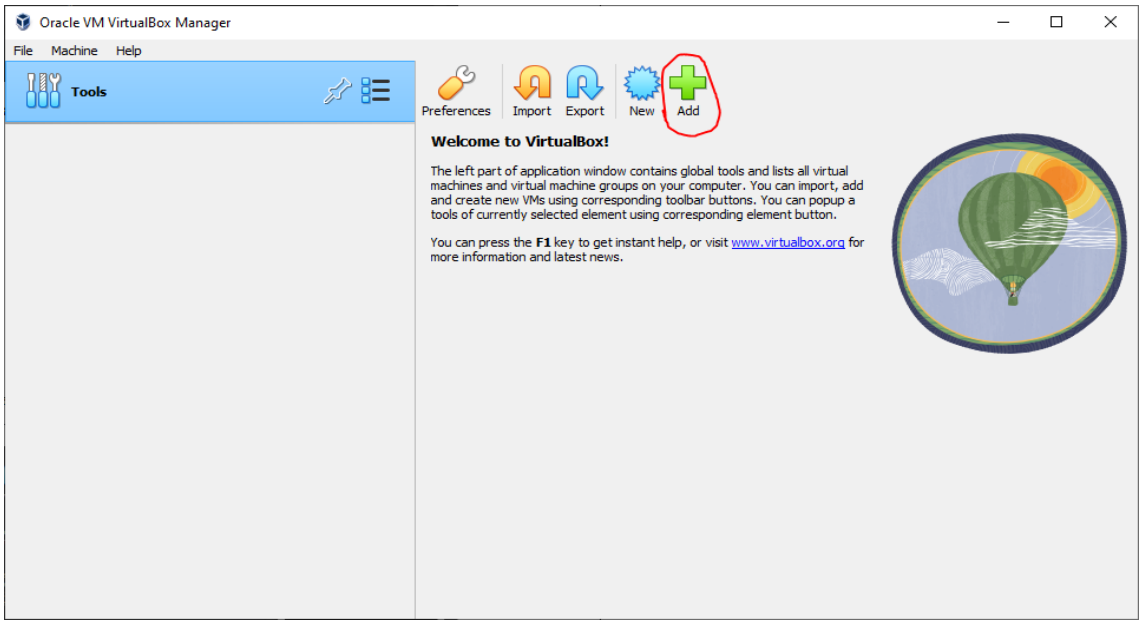
Adding a VM

If you already have a VM Disk Image, you can follow these steps to add it to your VM list in Oracle VirtualBox.

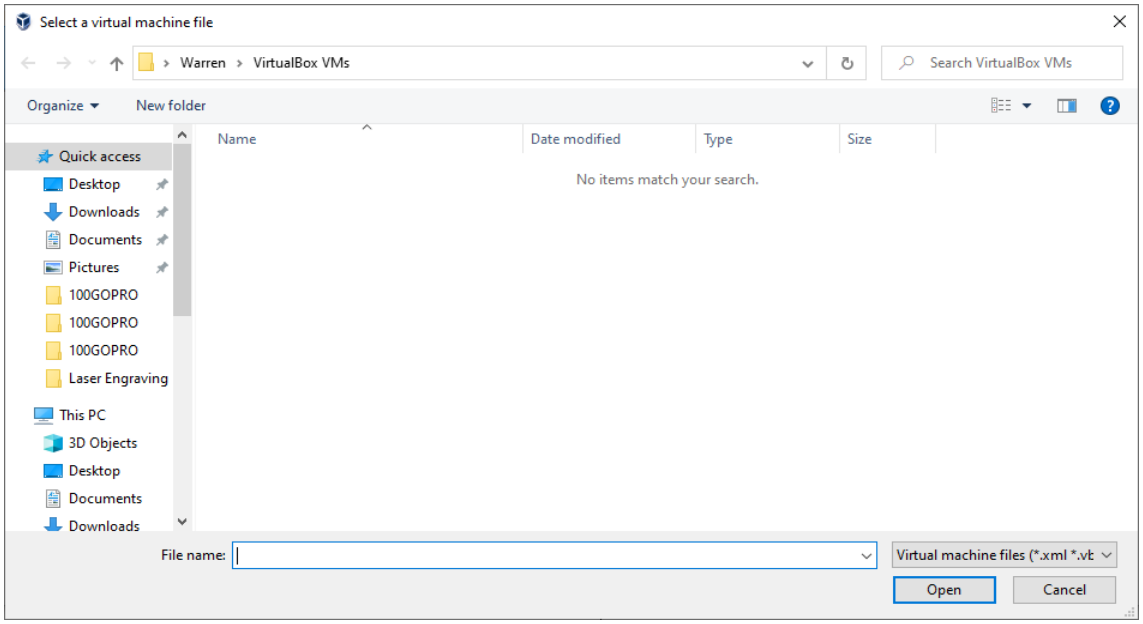
Open Oracle VirtualBox:



Click the green + sign to add a VM:

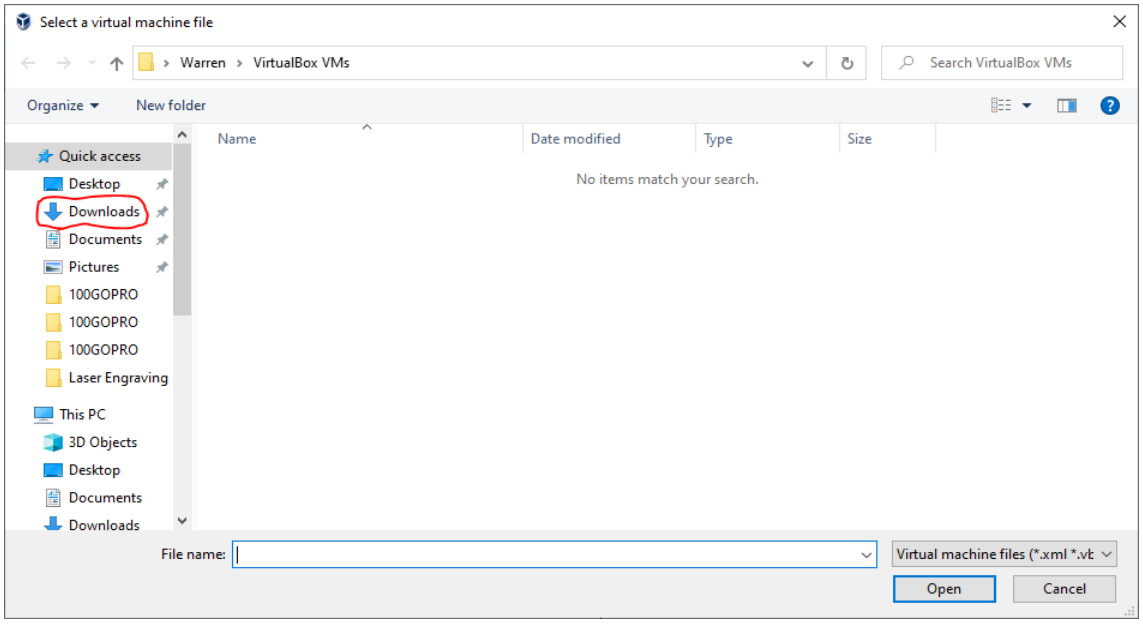


You will now be in the file explorer:

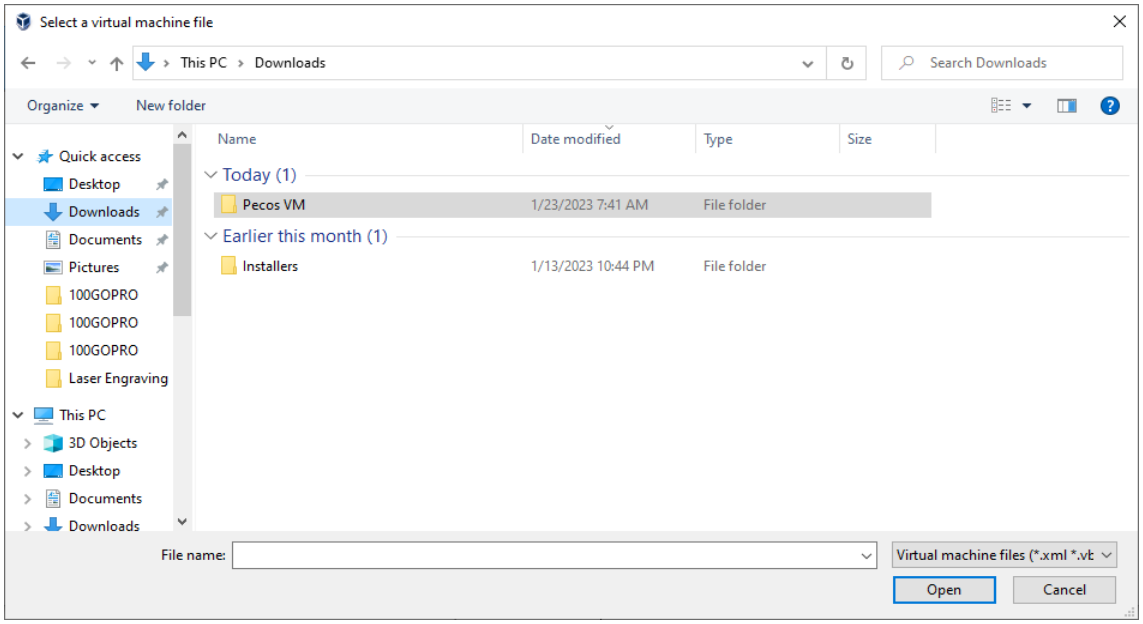


Locate the folder where your VM is located.

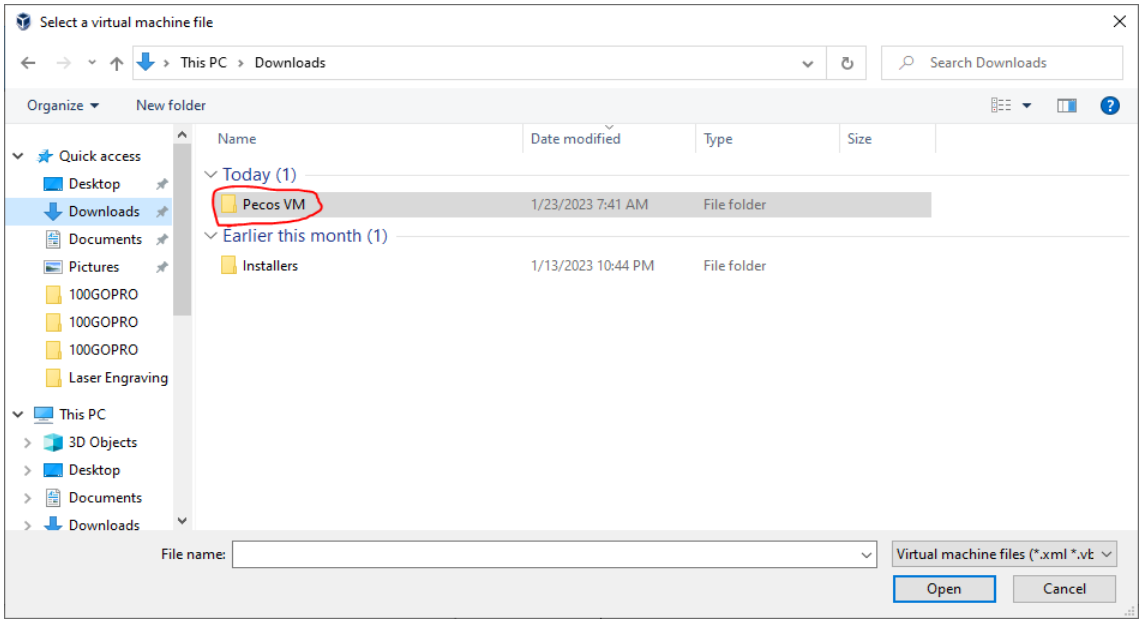
(In this instance, it is the Downloads folder):



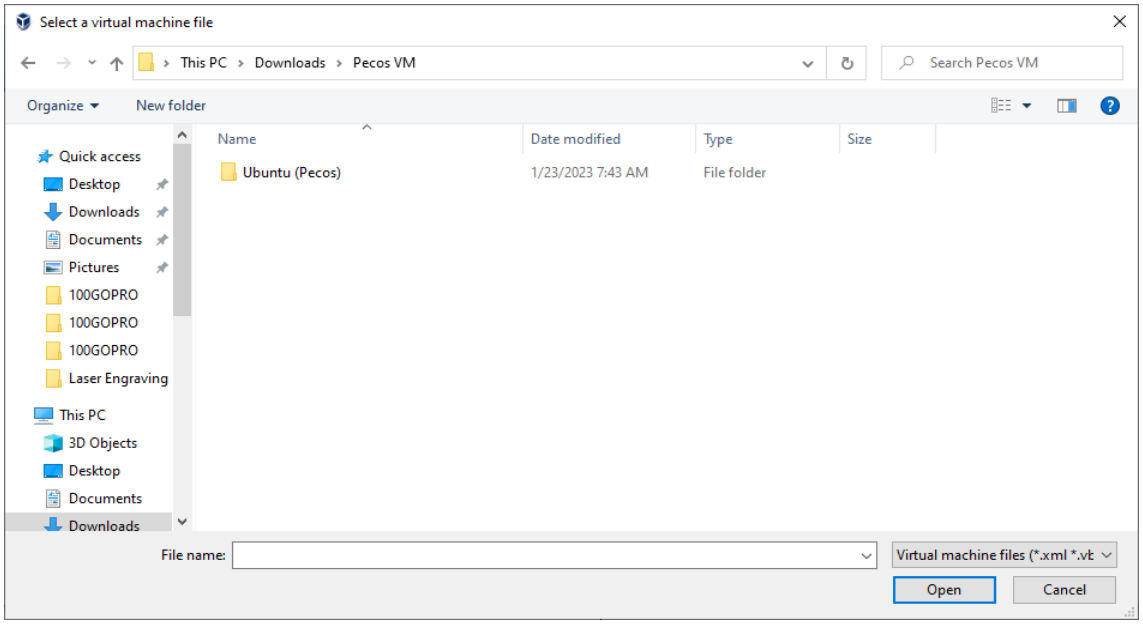
After clicking on the folder, you should now see the outer folder for the VM:



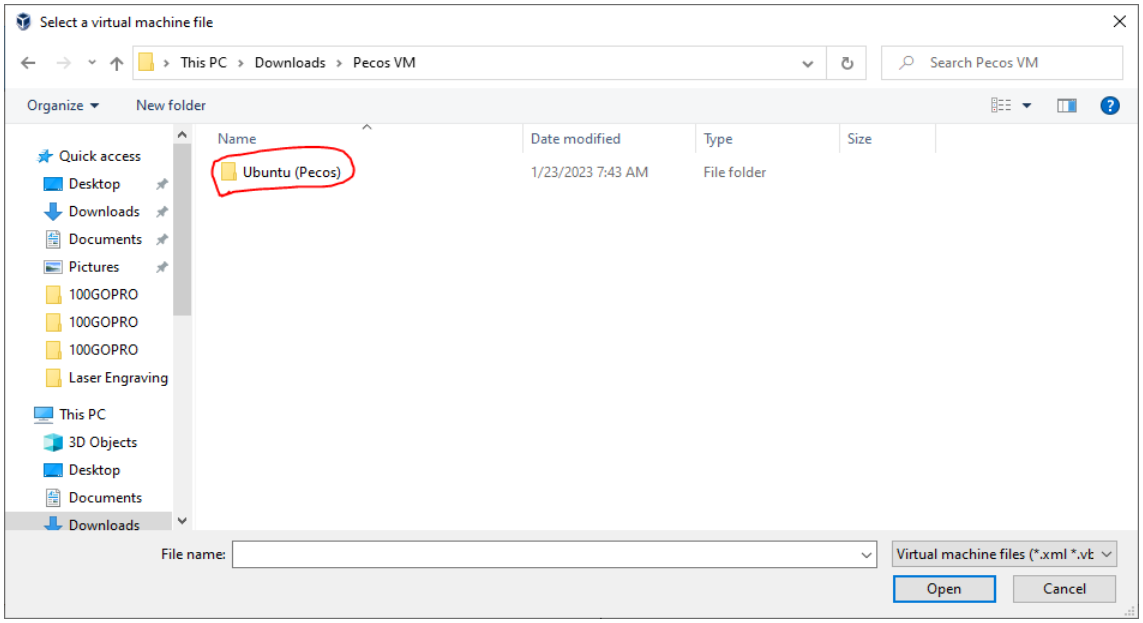
Double click this folder:



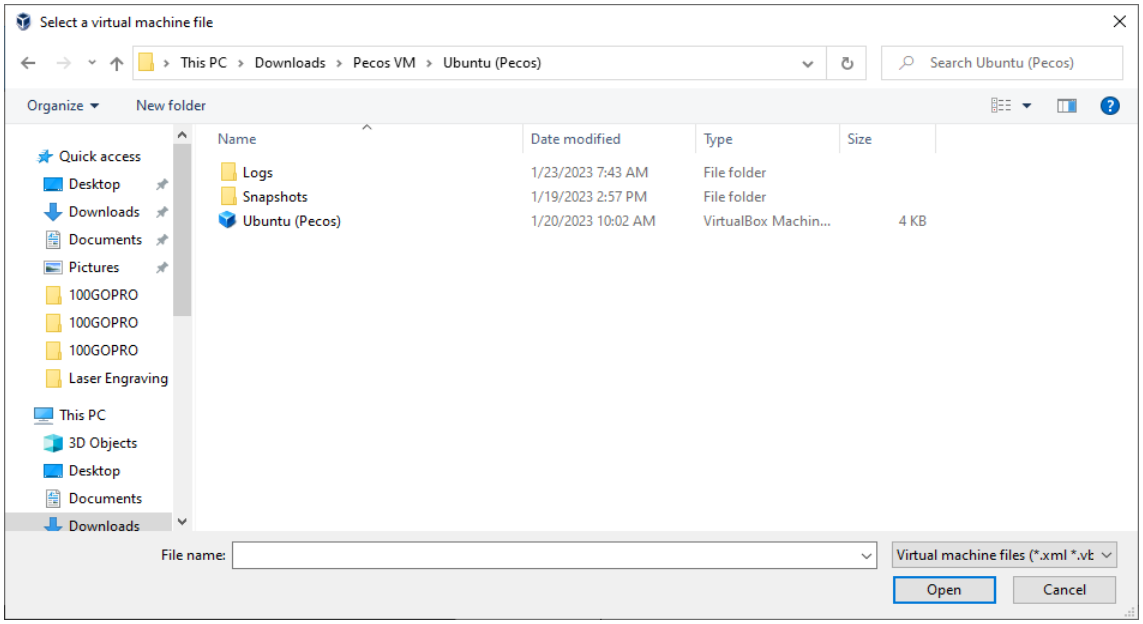
You will now see the inner folder with the name of the VM:



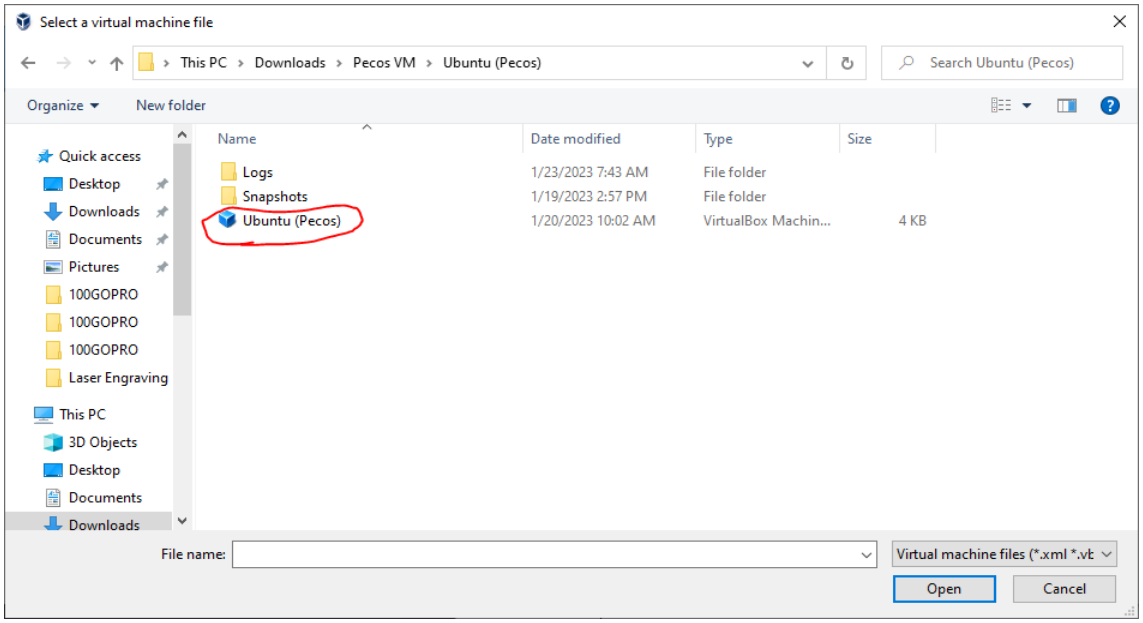
Double click this folder:



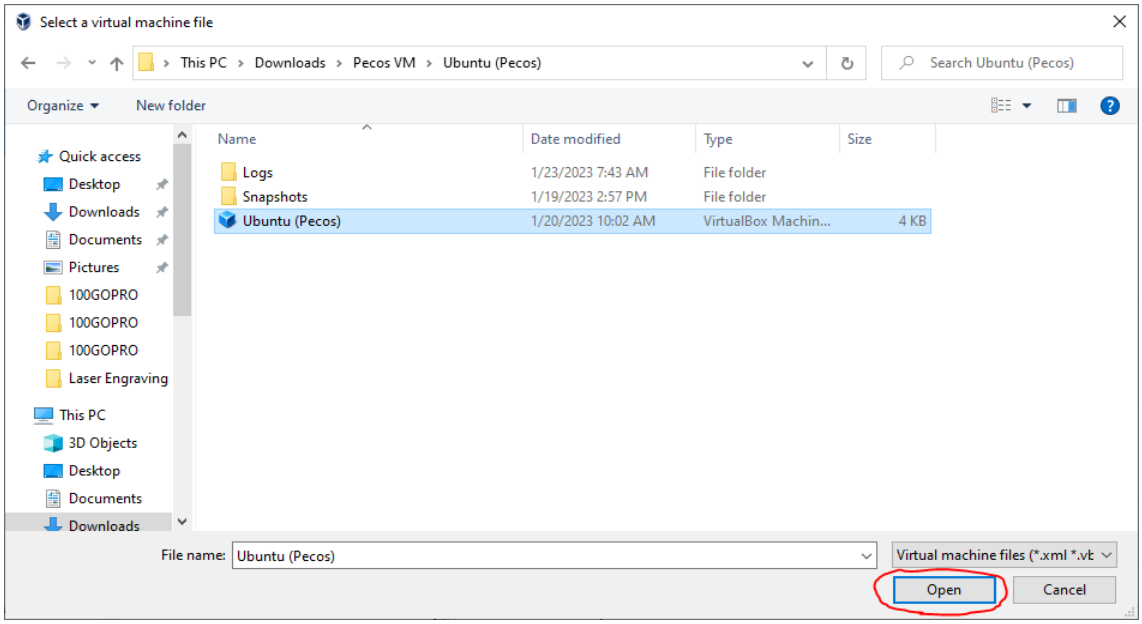
Now you will see a window like below:



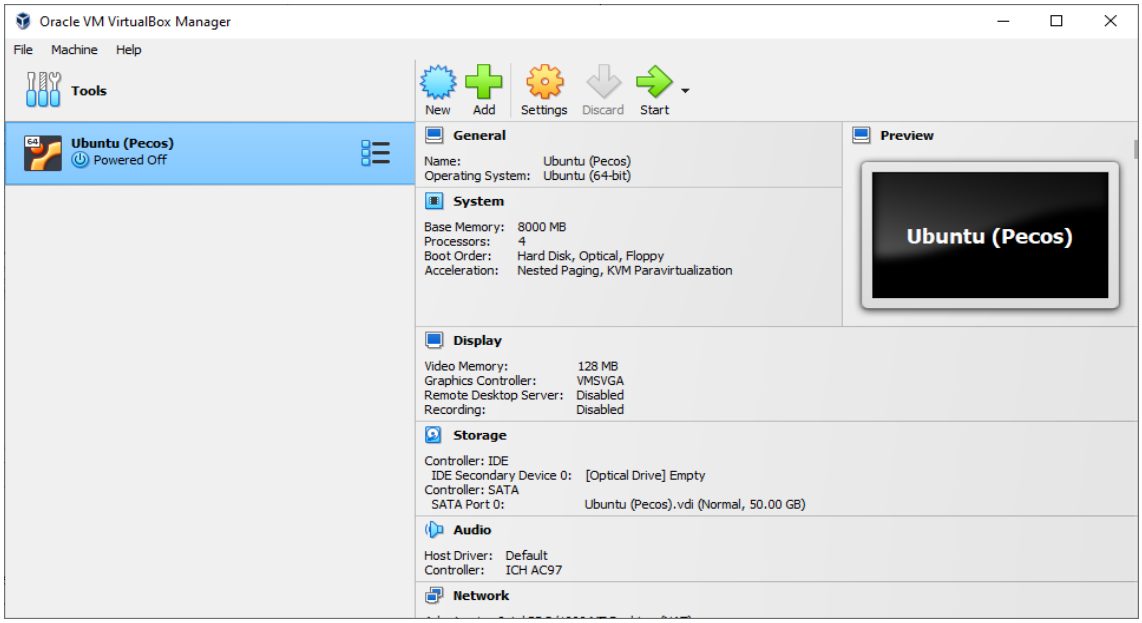
Select the VMD file:



Now hit open:

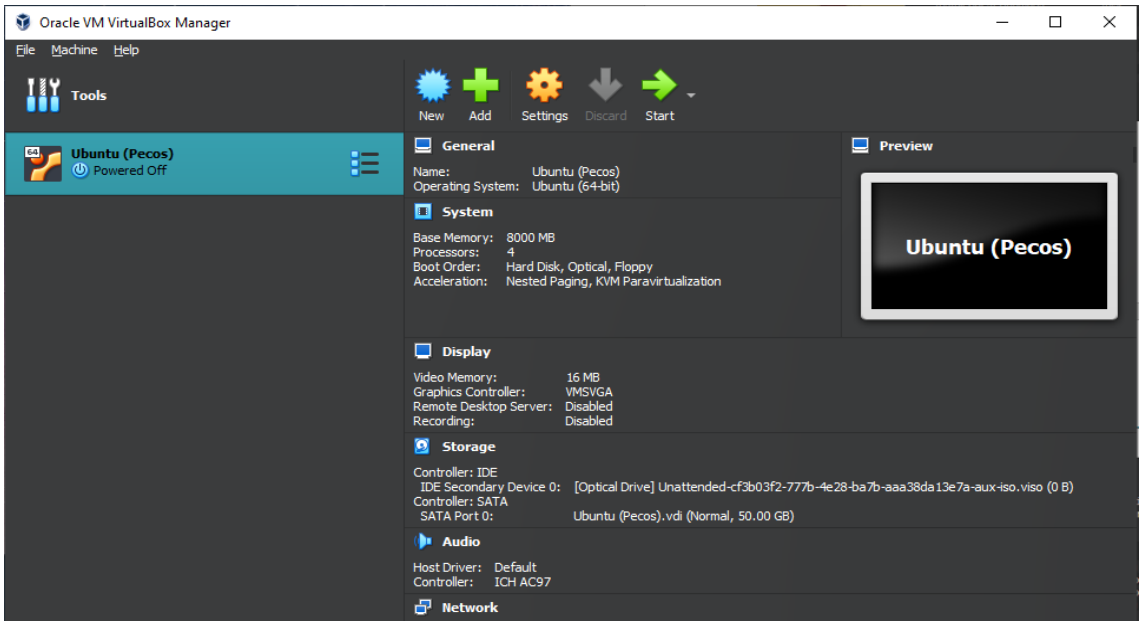


The VM should now show up on your VM list in Oracle VirtualBox:

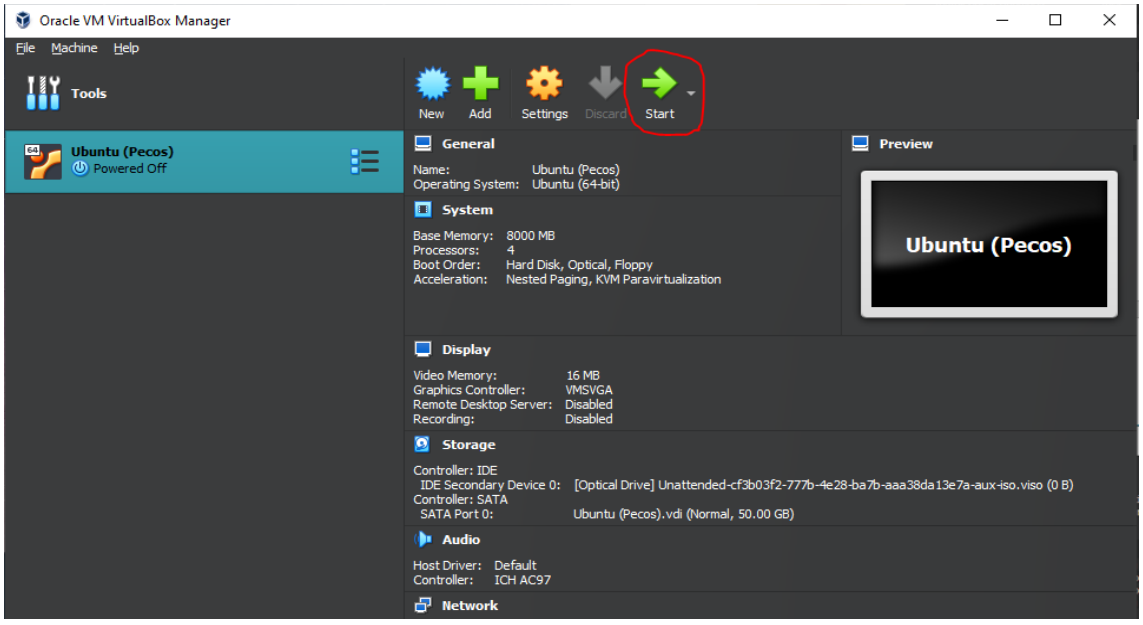


Launching a VM

If the VM does not launch automatically, select the newly created VM in the VM list:

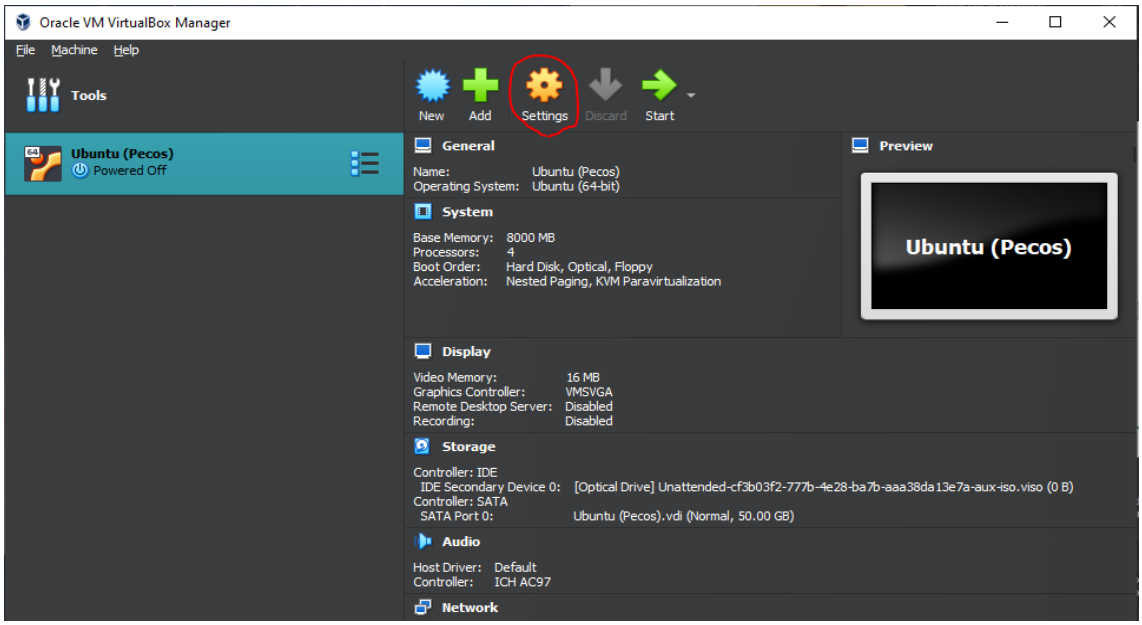


Then, click start:



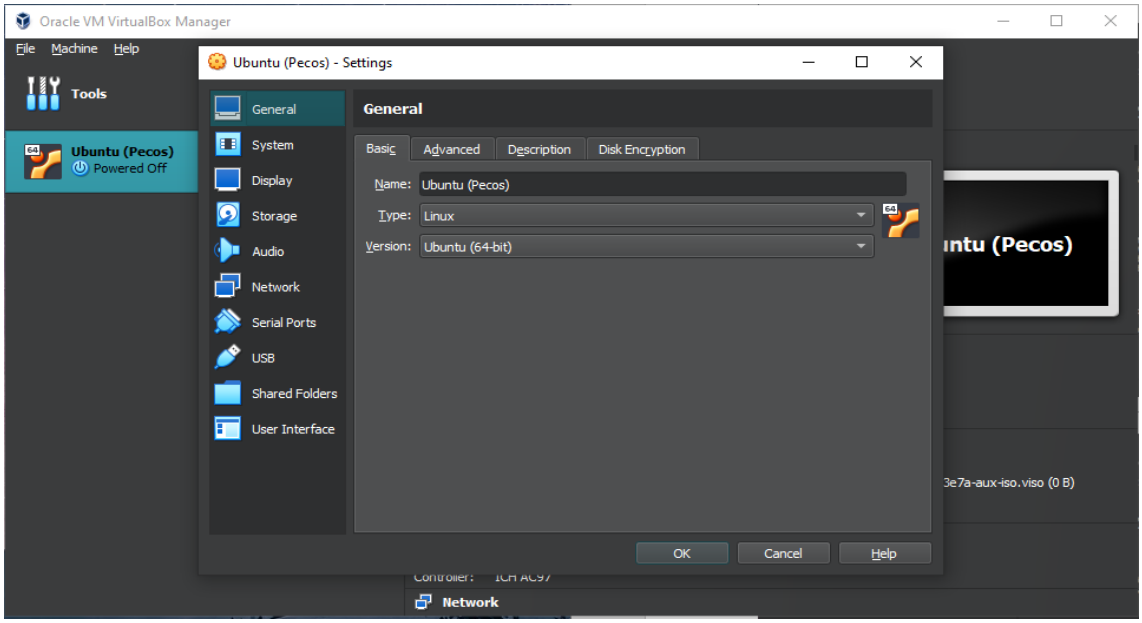
Changing a VM's Settings

If there are any changes that you need to make to the machine, go to settings:

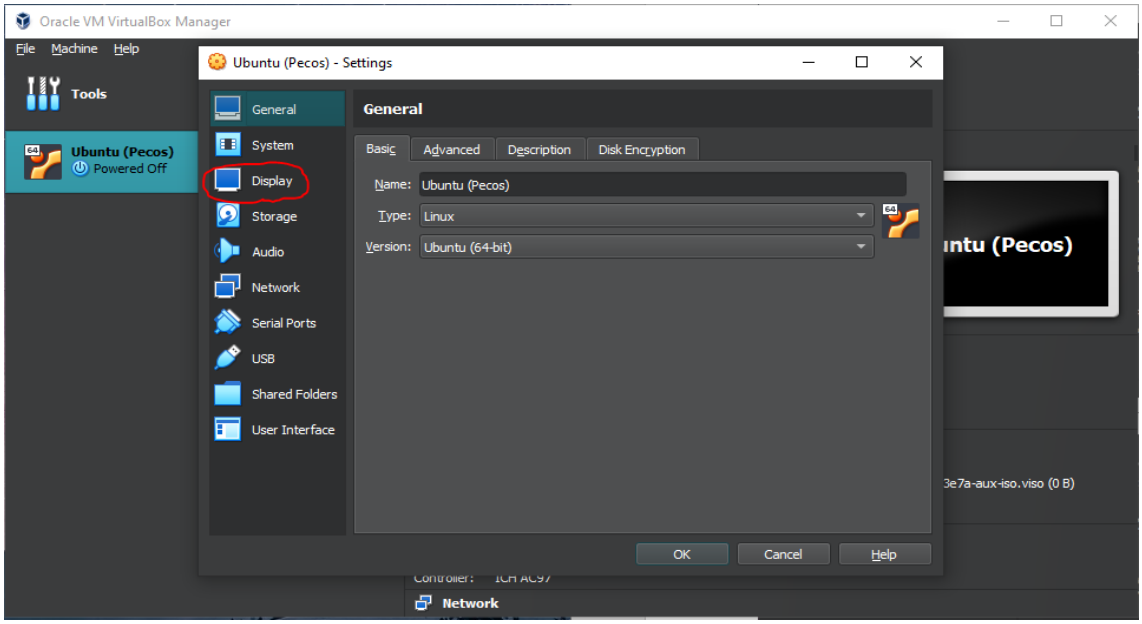


While we are here, we need to change one thing.

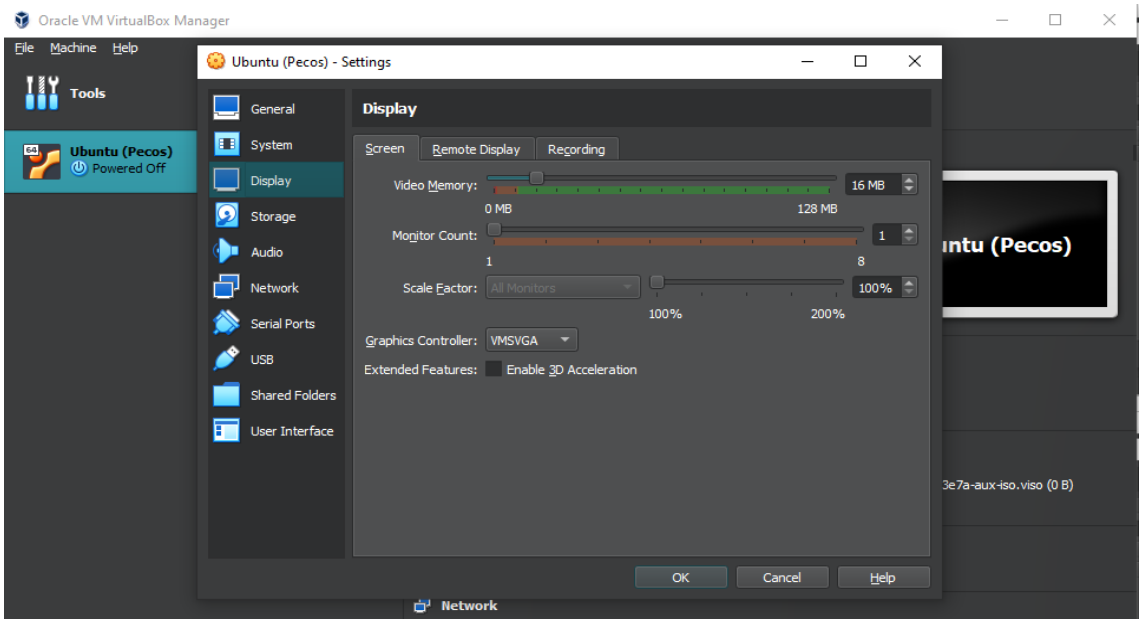
Click settings:



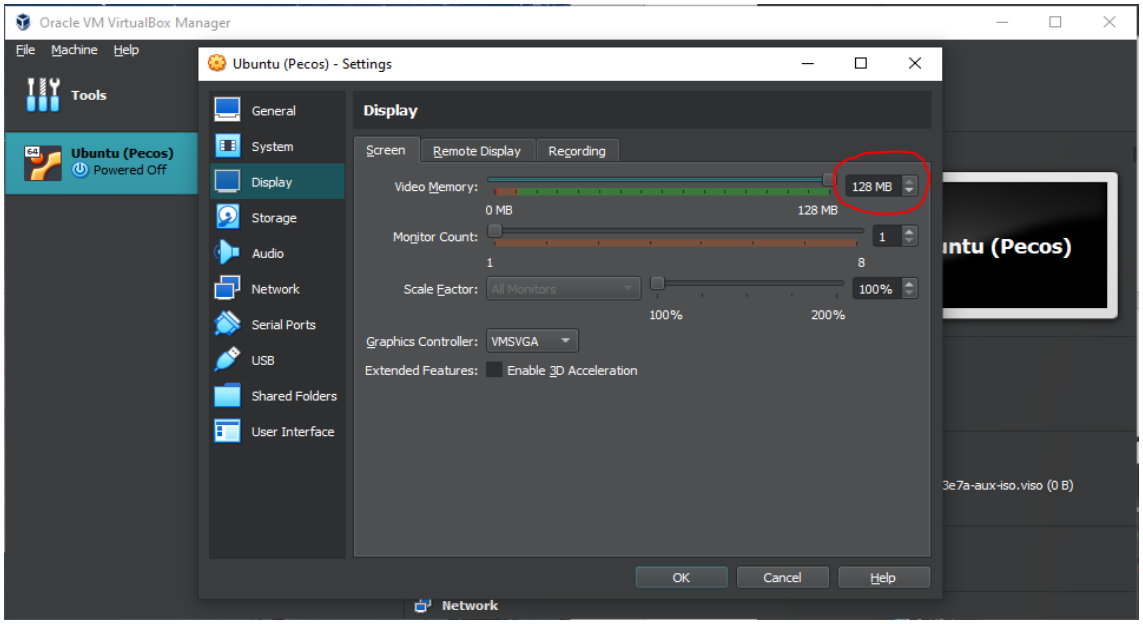
Select Display:



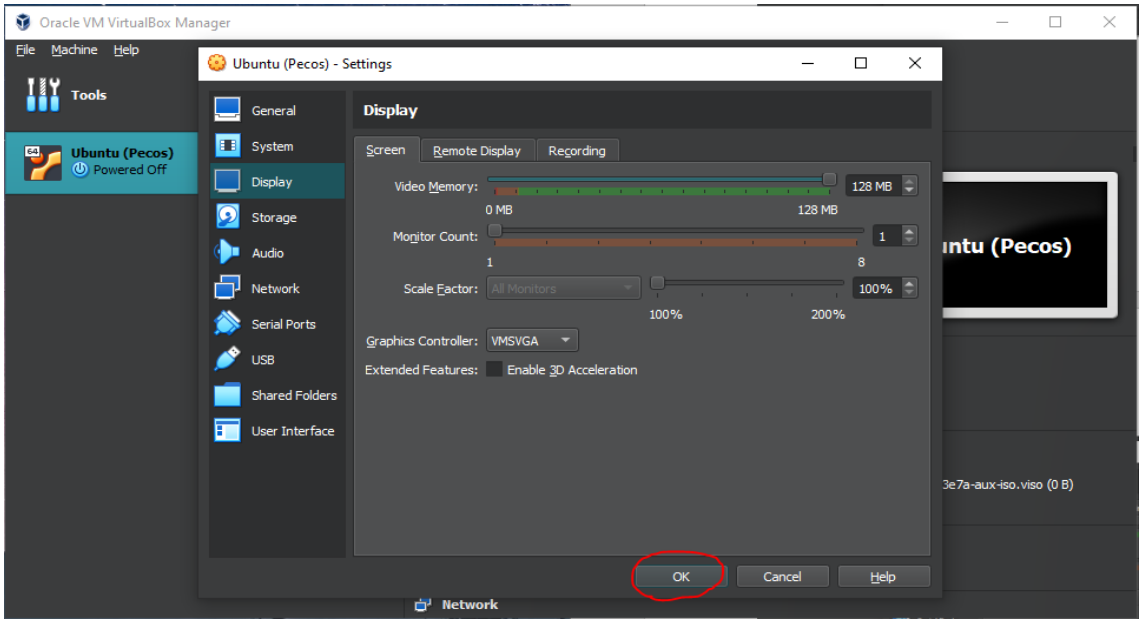
You should now be seeing a window like the one shown below:



Set Video Memory to max:

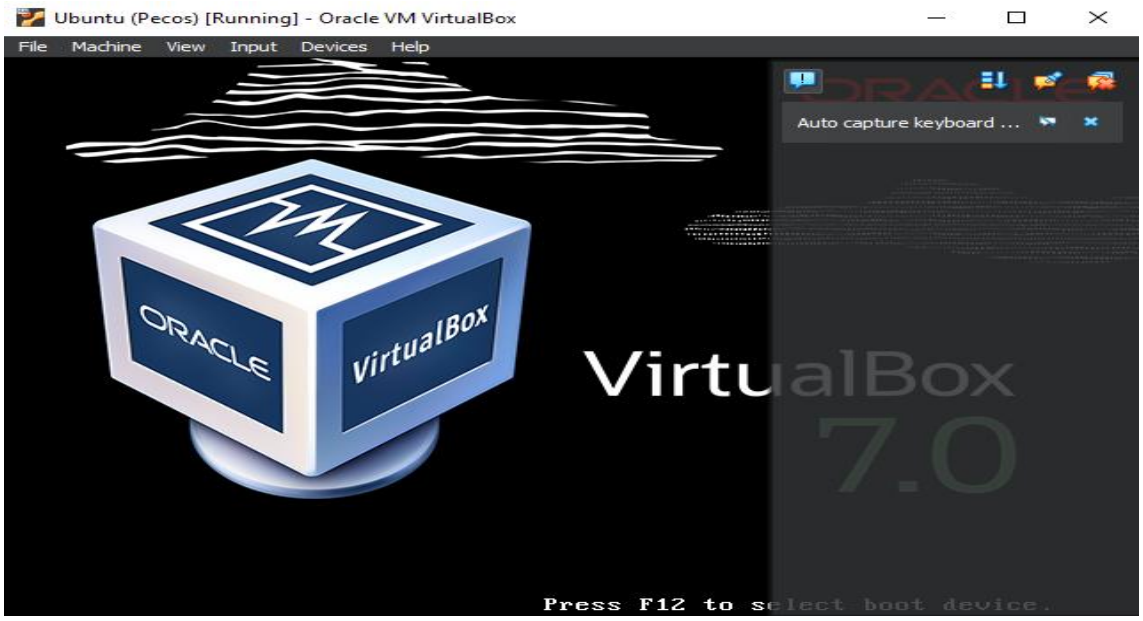


Now hit OK:

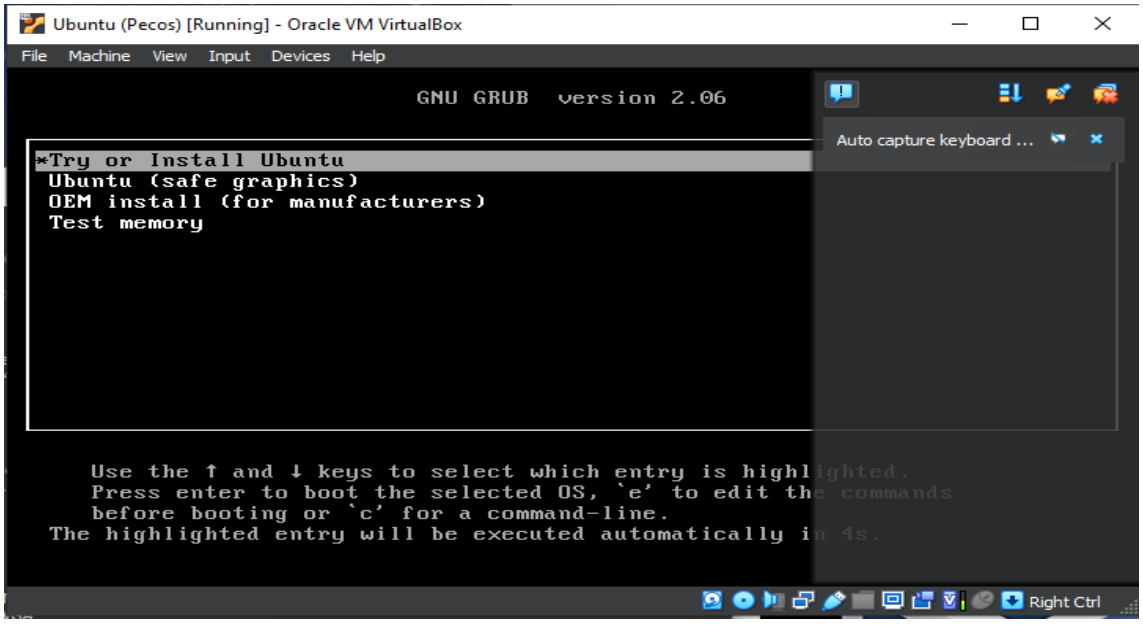


Step 4: Running Virtual Machine

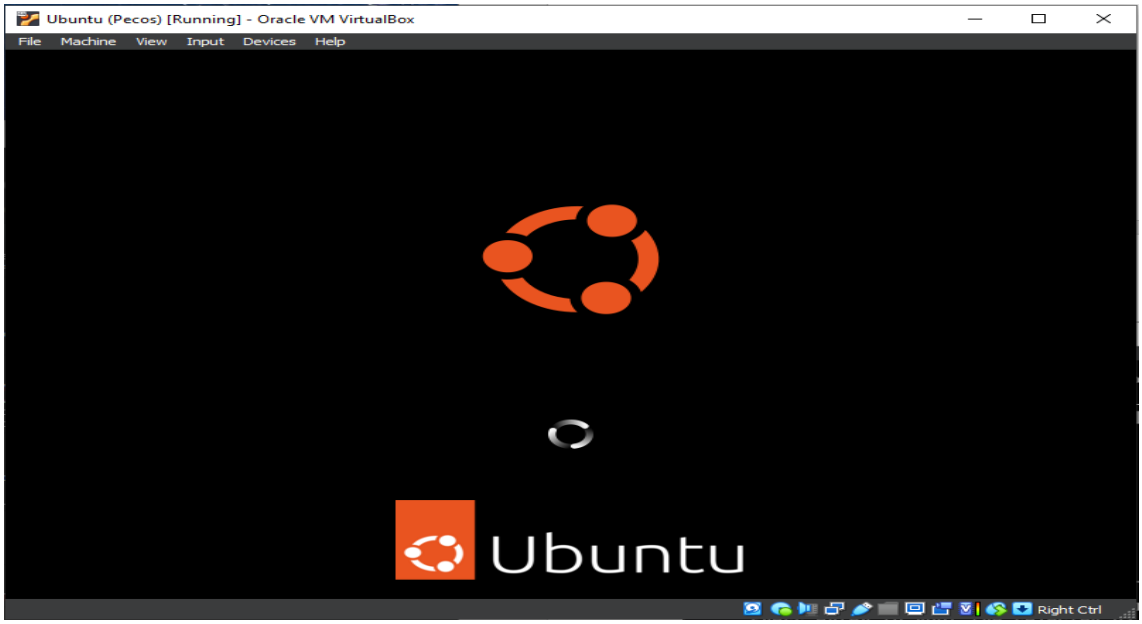
Once you launch the newly created VM, you will be greeted with the following window:



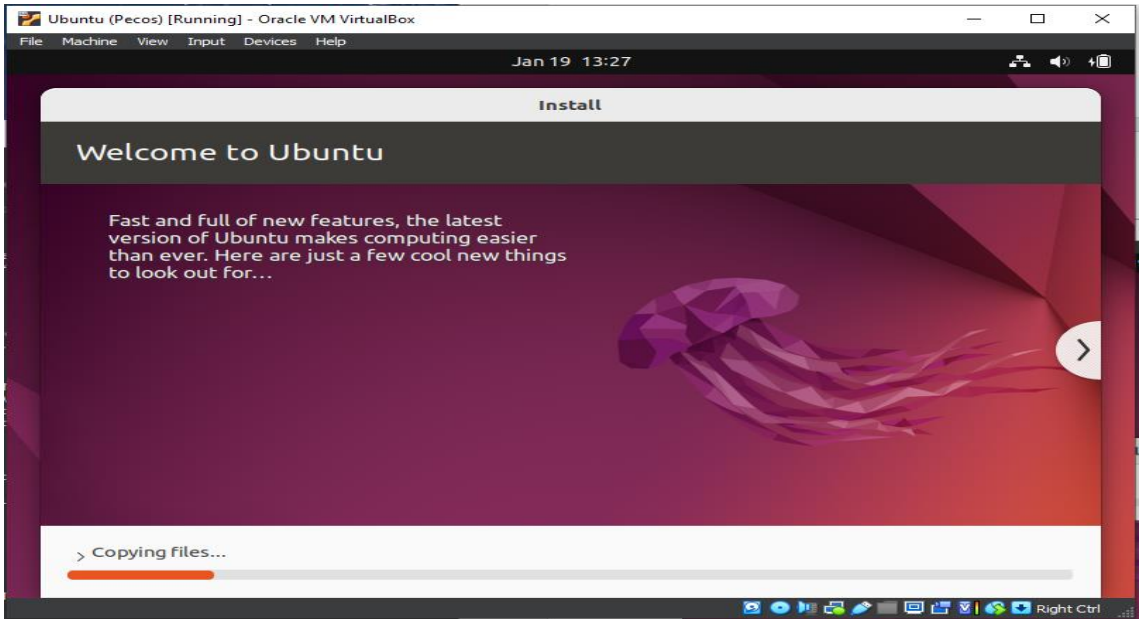
Hit Enter to begin Ubuntu Install. (If there is a delay, the VM will automatically do this.):



You should now see a window like the one below:

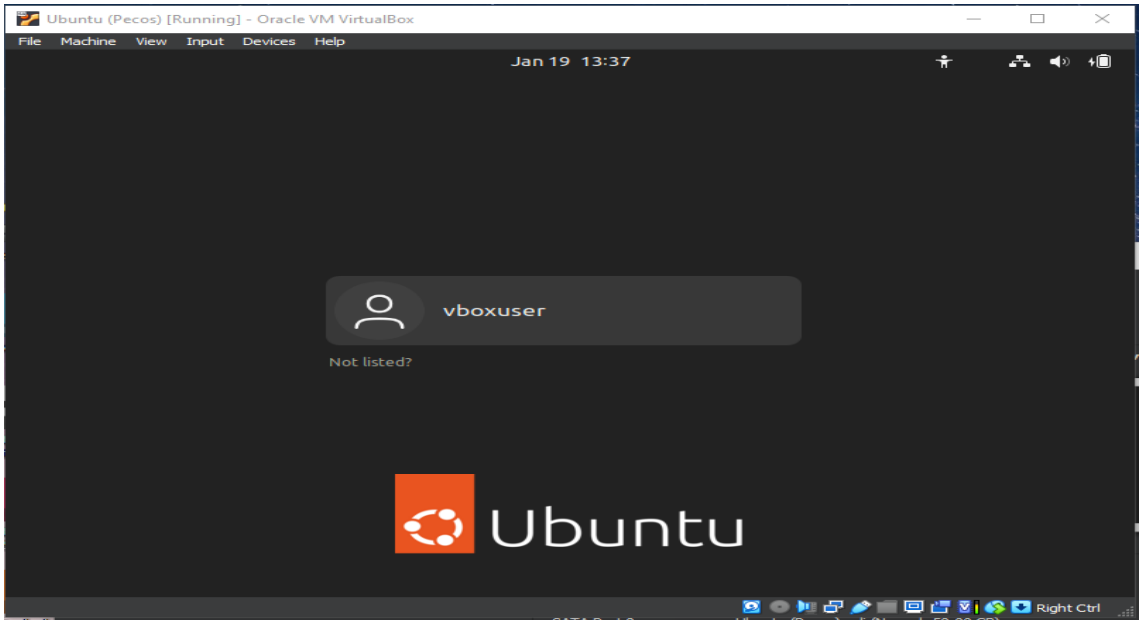


Then one like this:

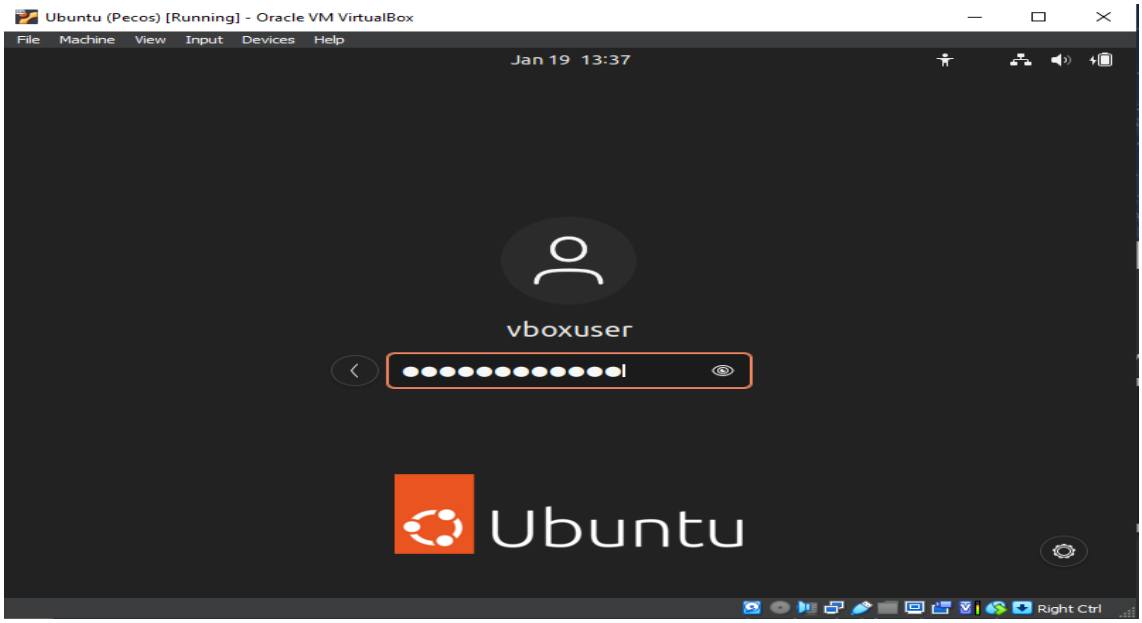


This process will take about 20 minutes, and once it is completed the machine will restart.

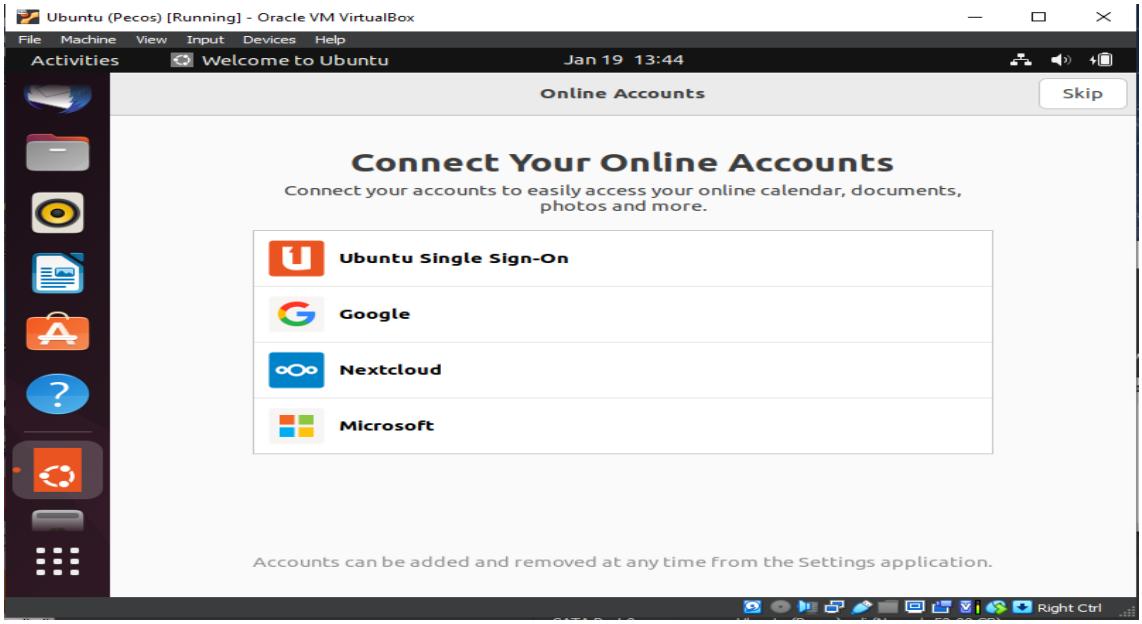
After restart, you will see a window like the one shown below:



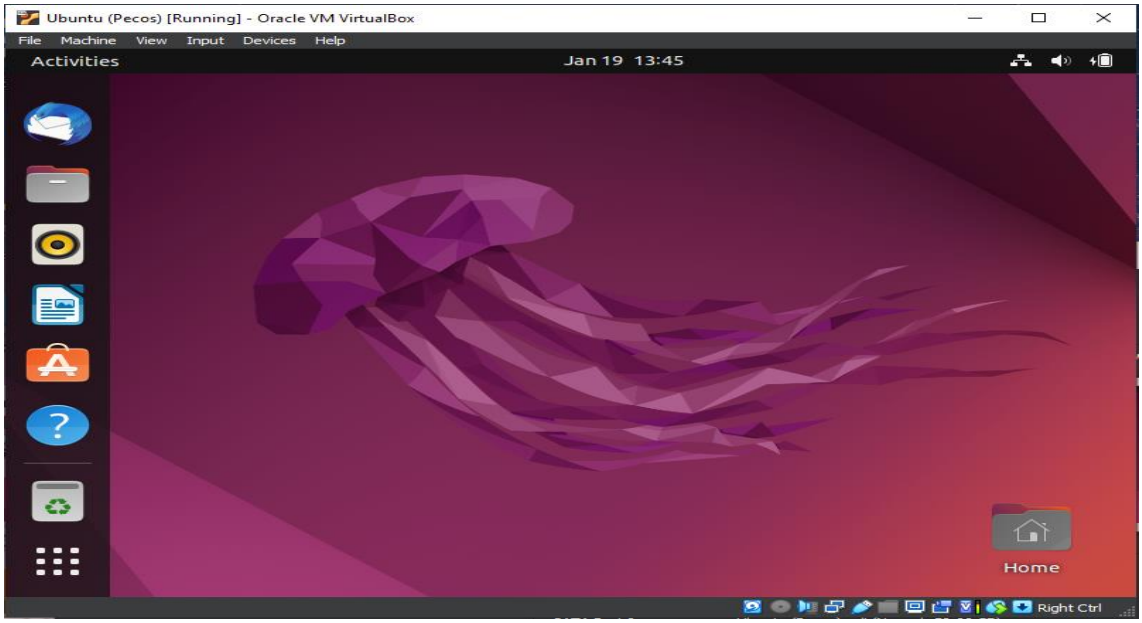
You'll need to login again:



I recommend skipping through this entire process. (Accounts can be added later, if needed):

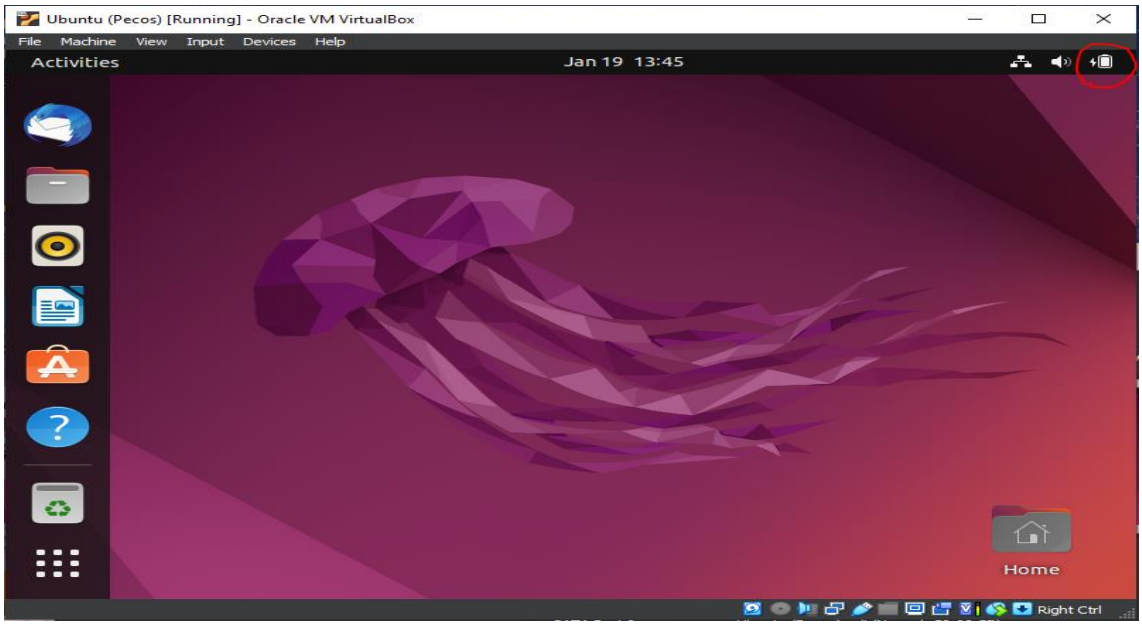


You should now be on your VM's desktop:

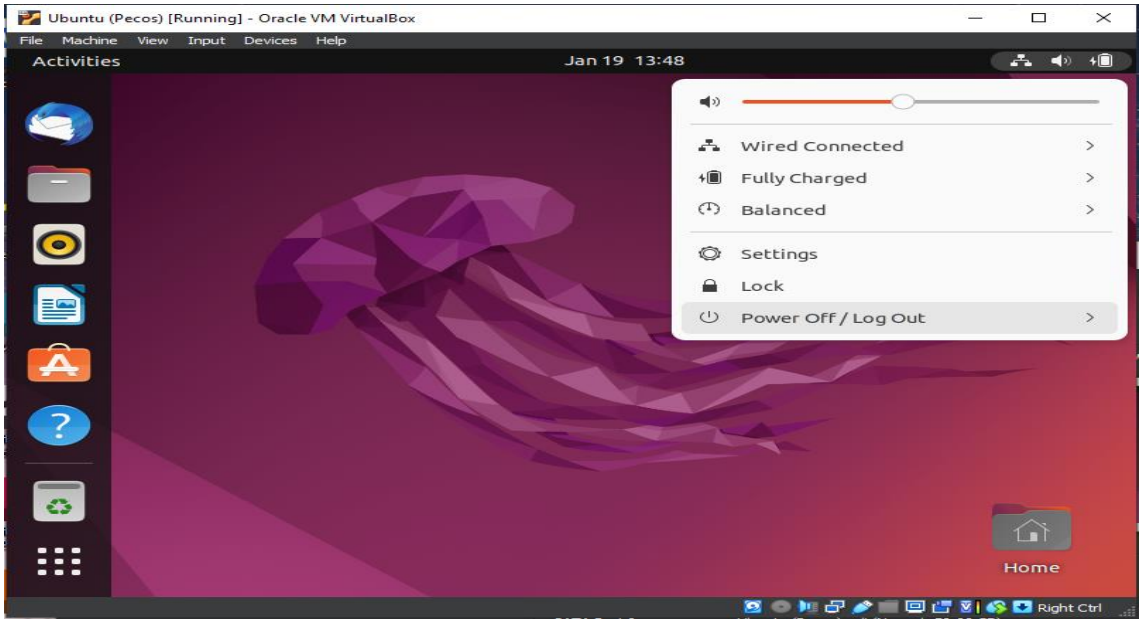


Powering Off/Restarting VM

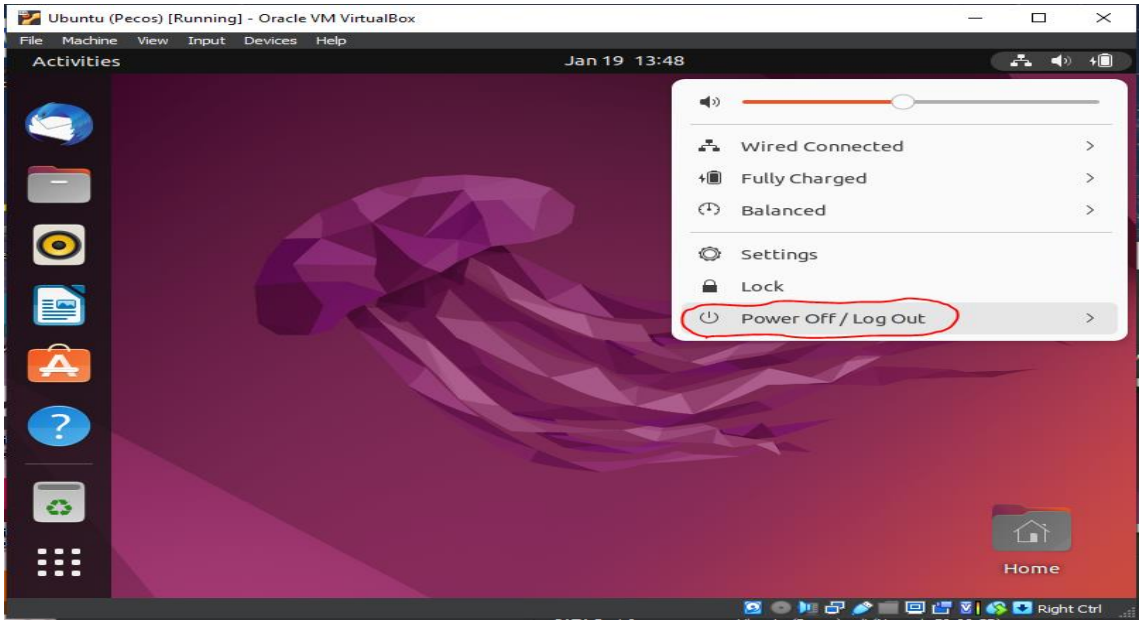
Select the power icon located to the top right



You will now see a window like the one below:



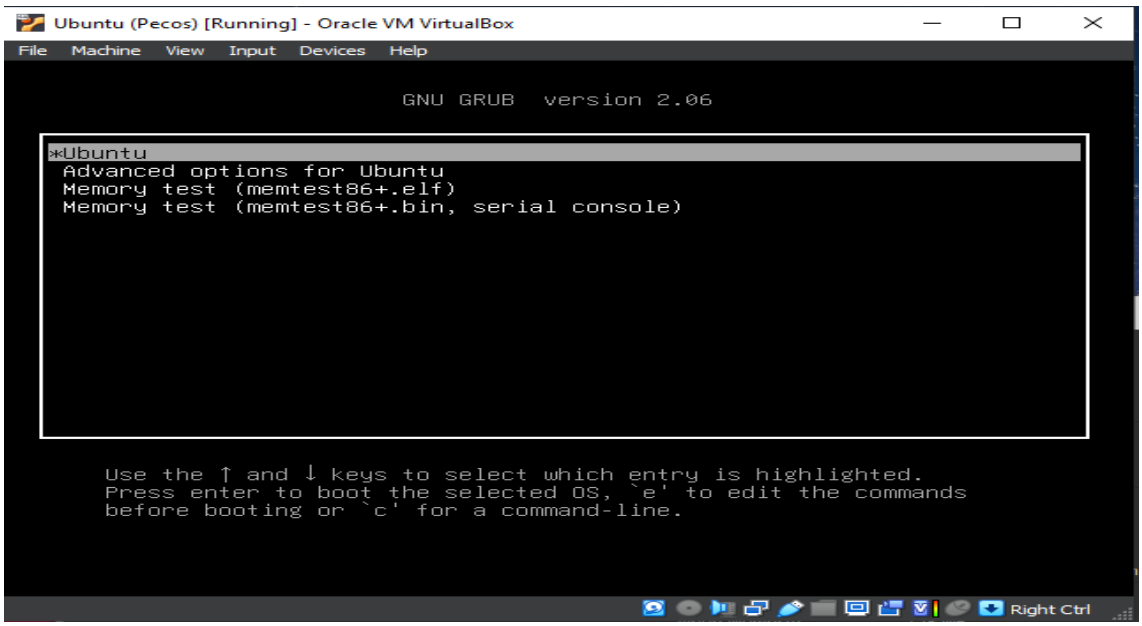
Select Power Off/Log Out:



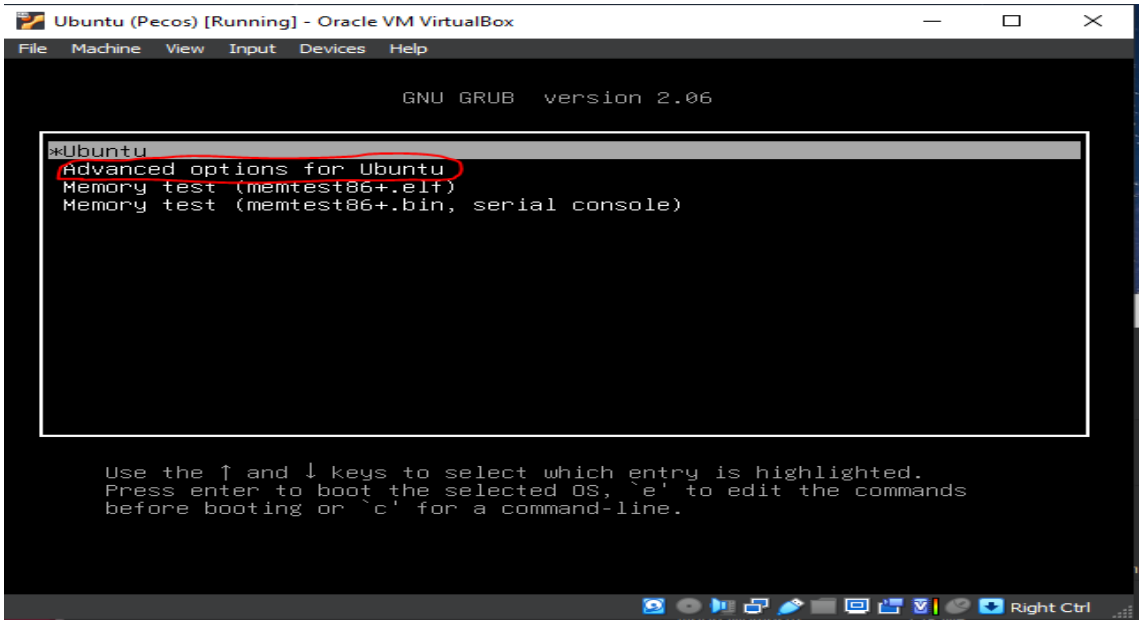
Adding Administrator Privileges

You must go through this process before installing Anaconda or adding packages

Restart VM using the steps described previously and hold down shift when the system begins to restart. You should now see a window like the one below:

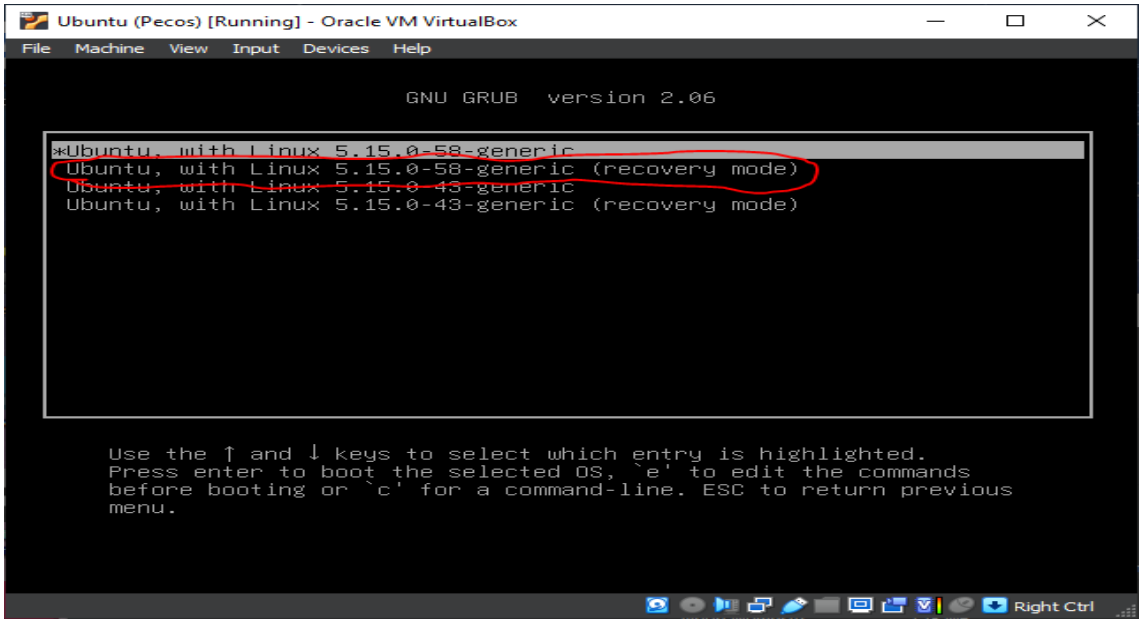


Select Advanced options for Ubuntu:



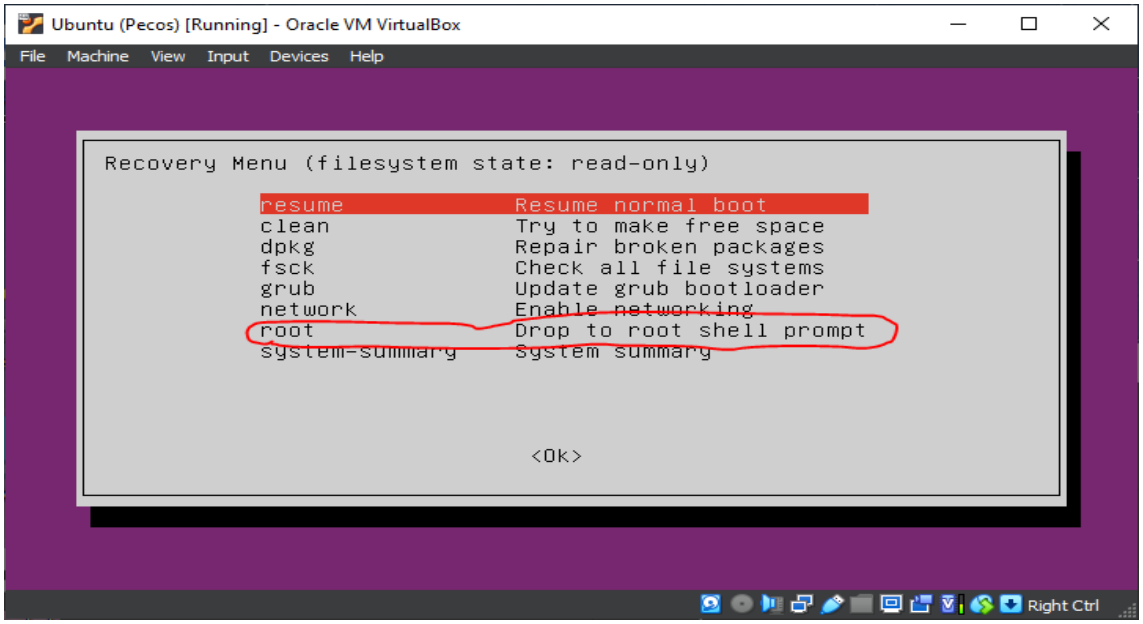
Select recovery mode for the most recent version of Linux.

(In this case it's Linux 5.15.0-58-generic):

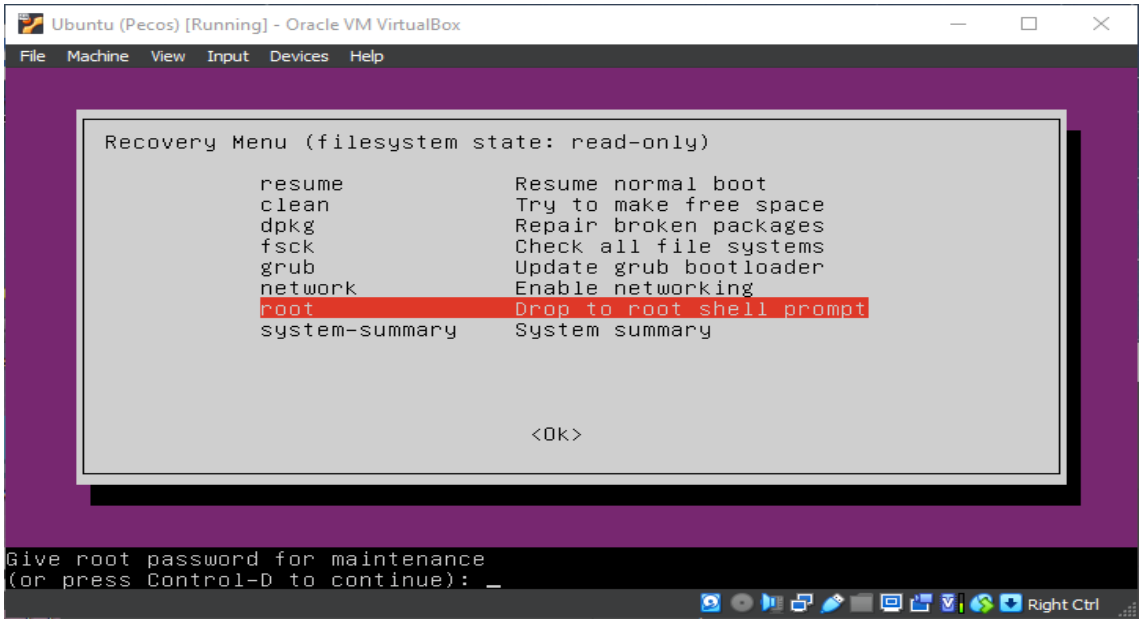


Now you should be looking at the window below.

We are interested in the root filesystem:



Arrow down to root and hit Enter:

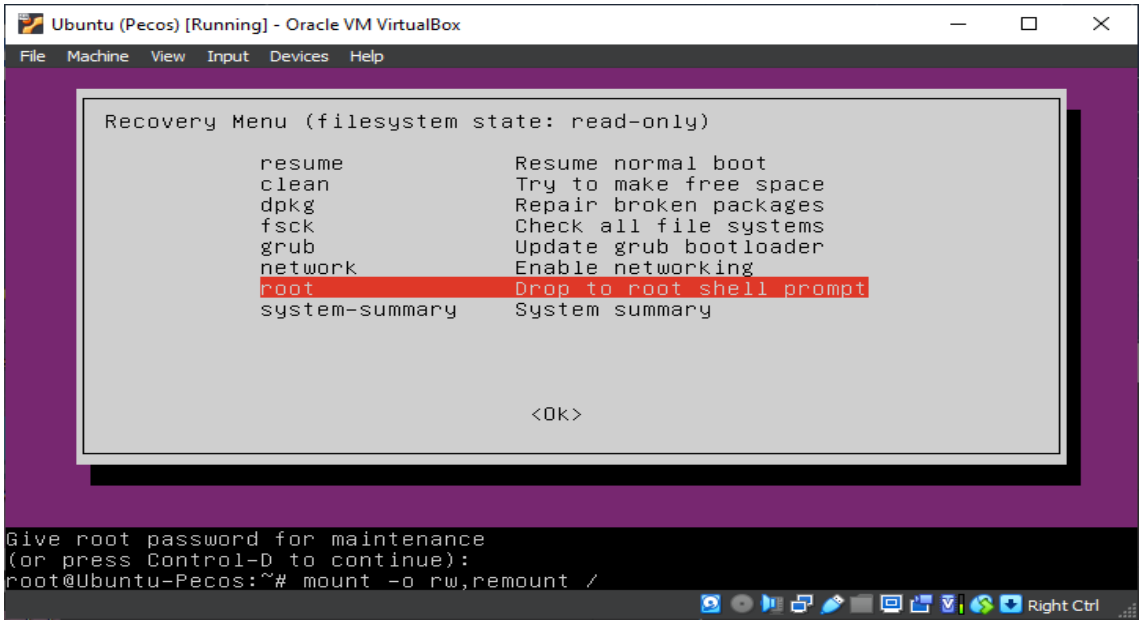


Input password and hit Enter once again.

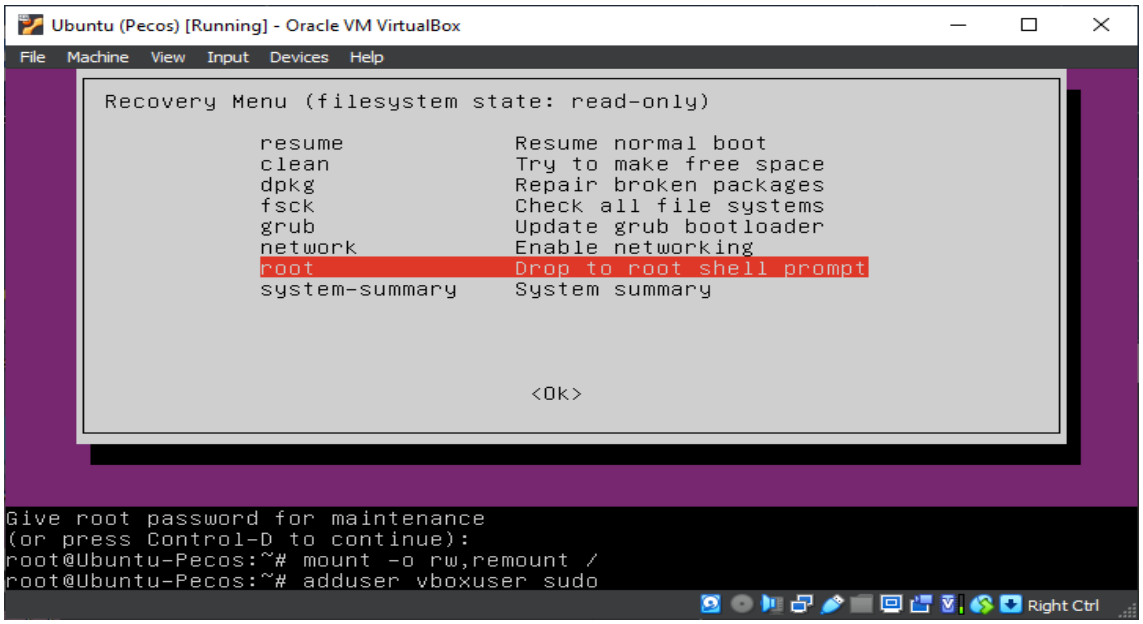
In terminals, passwords are invisible

Now you should be looking at the window below.

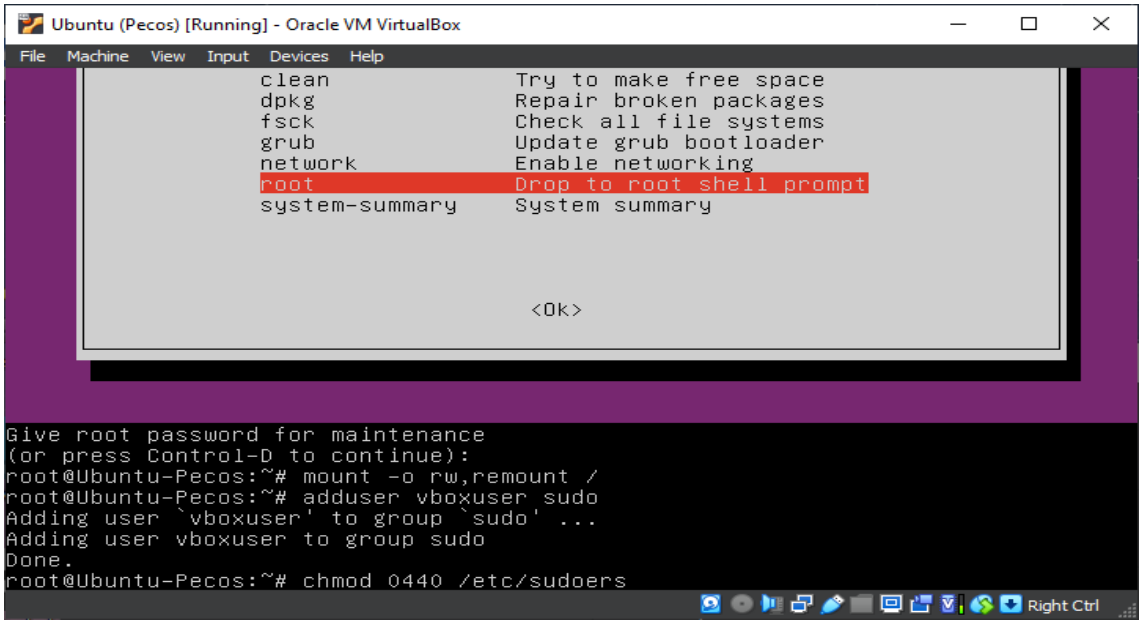
Type command *mount –o rw,remount /* and hit Enter:



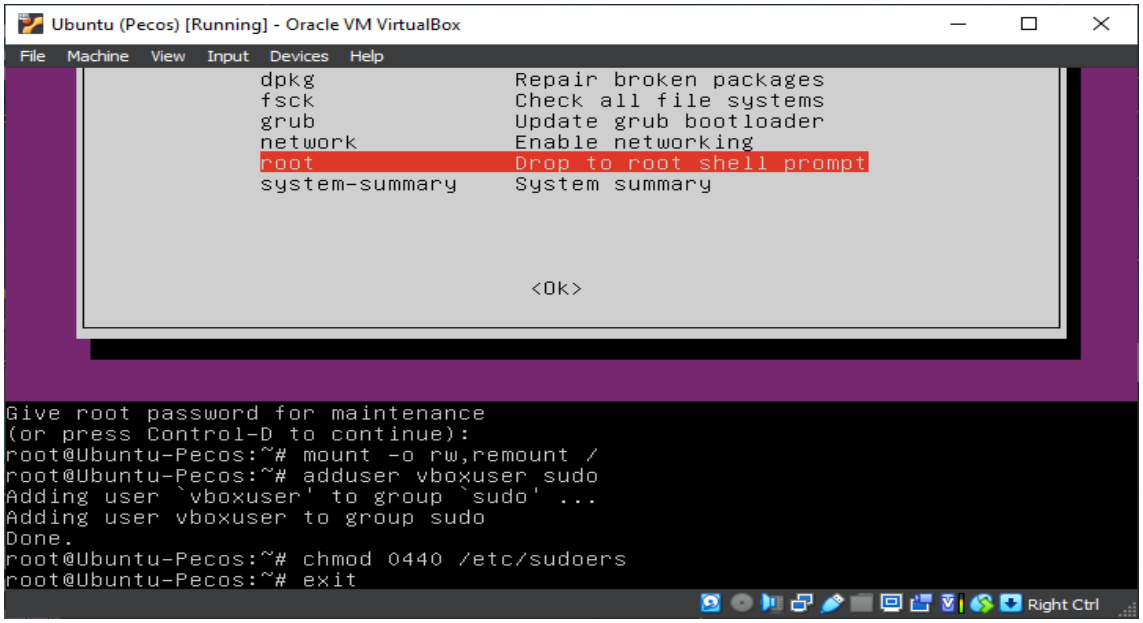
After that, type command *adduser vboxuser sudo* and hit Enter:



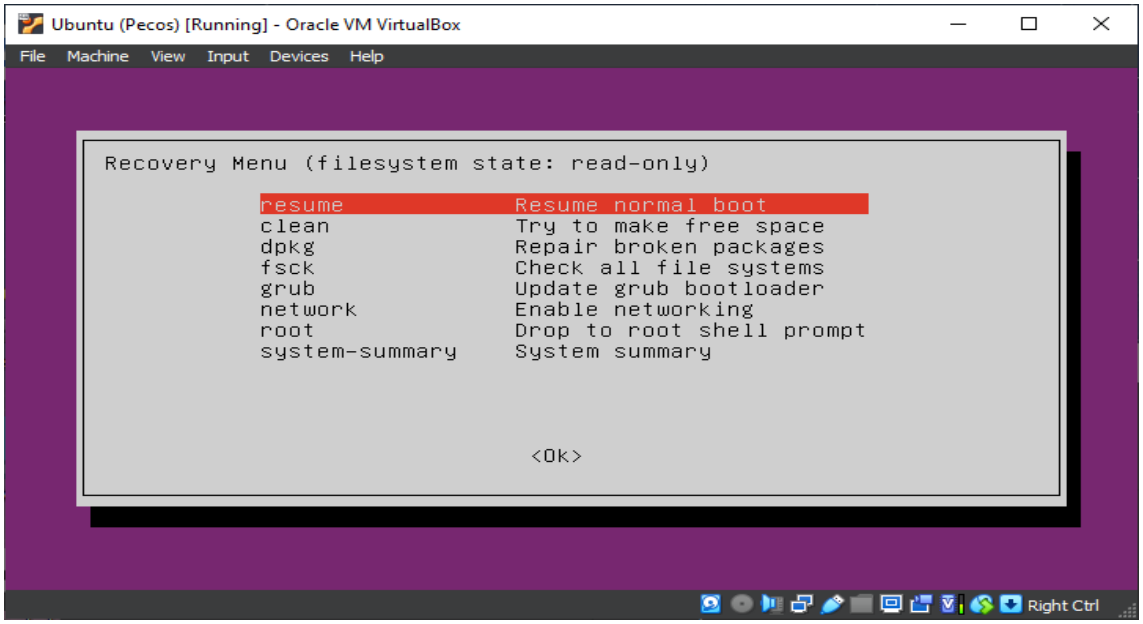
Then, type command *chmod 0440 /etc/sudoers* and hit Enter:



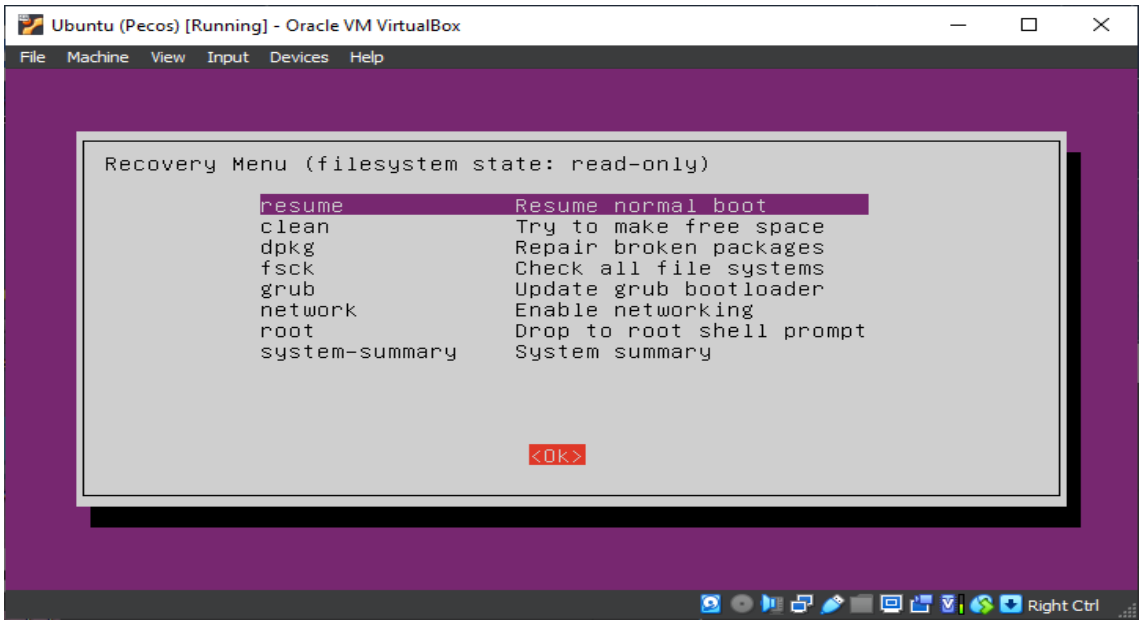
Finally, type command *exit* and hit Enter:



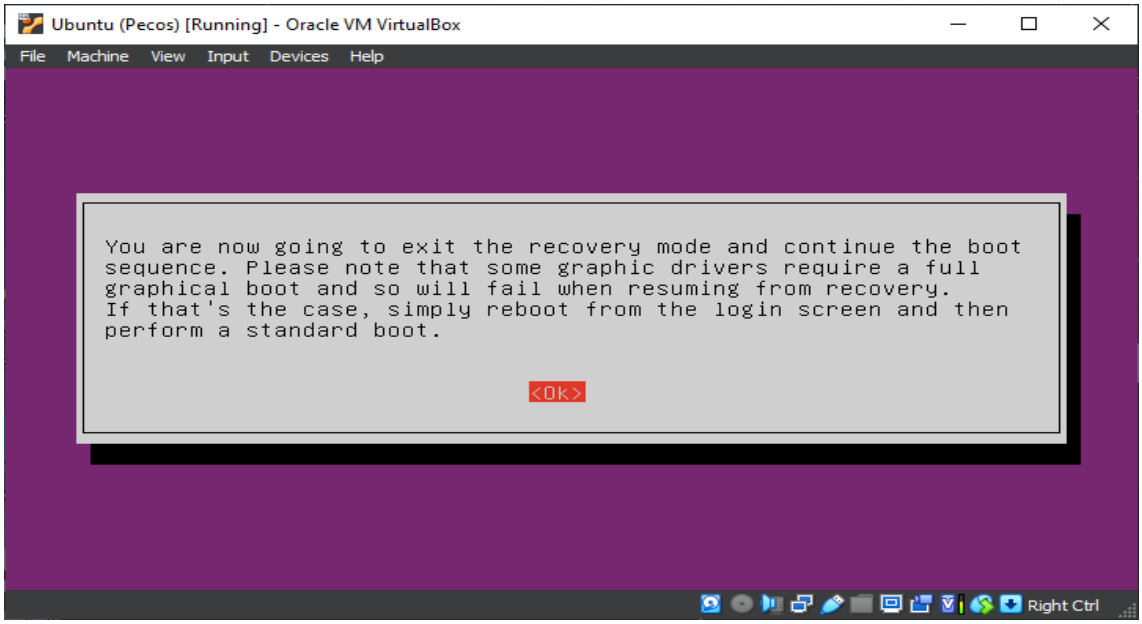
You should now be back in the normal Recovery Menu:



Hit the right arrow key:



Hit Enter:

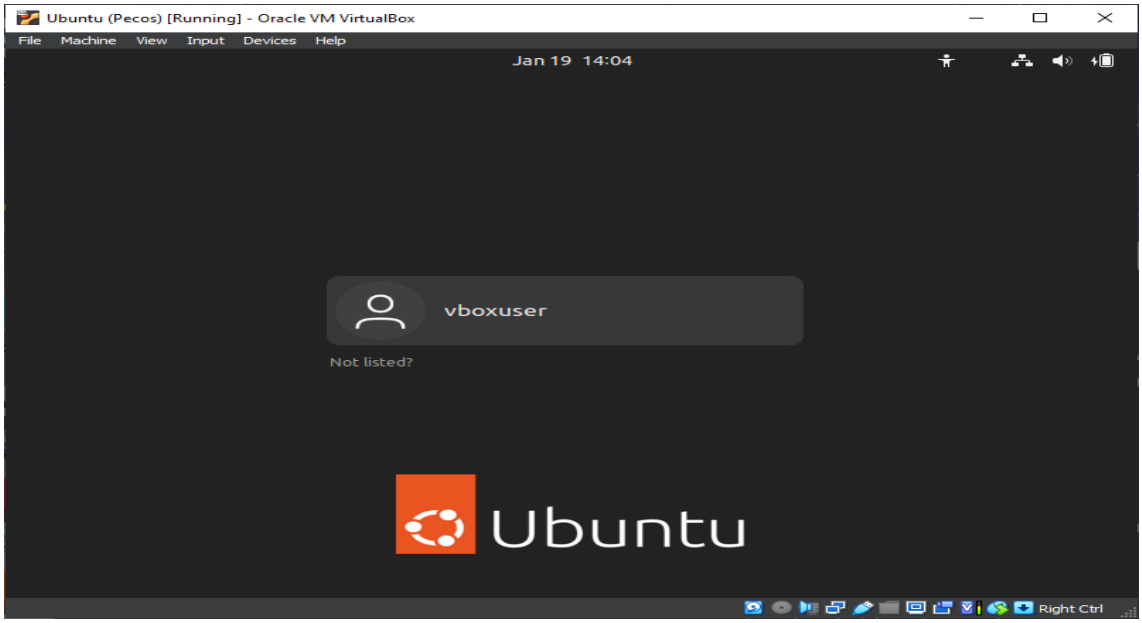


Now the system will restart to apply the changes.

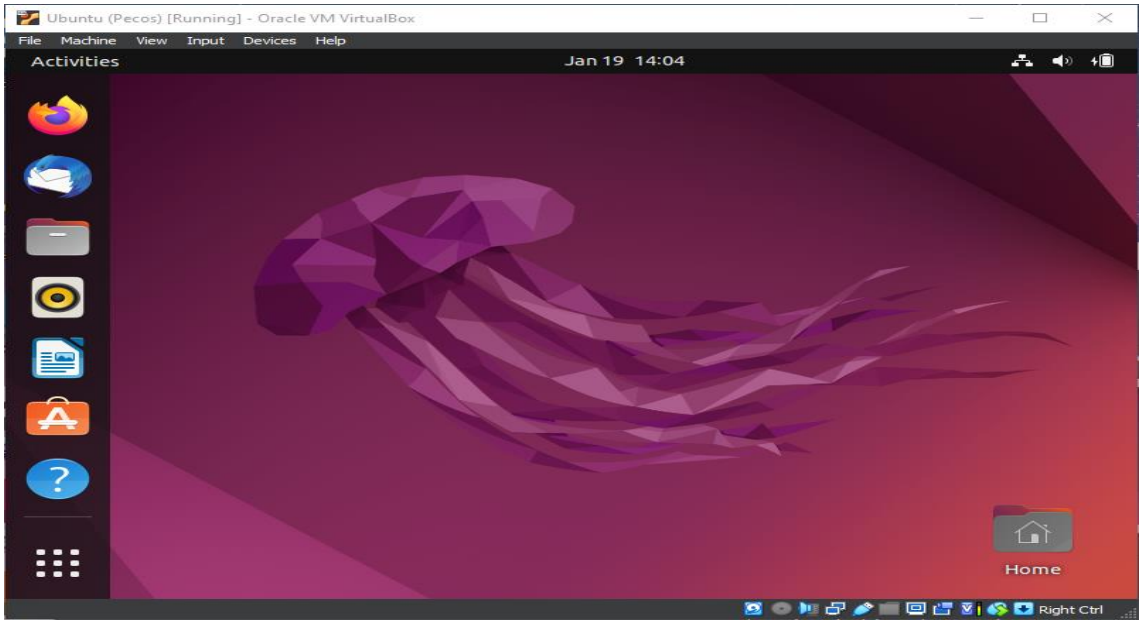
Step 5: Install Anaconda on Virtual Machine

Launch Oracle VM VirtualBox. Select virtual machine and hit start.

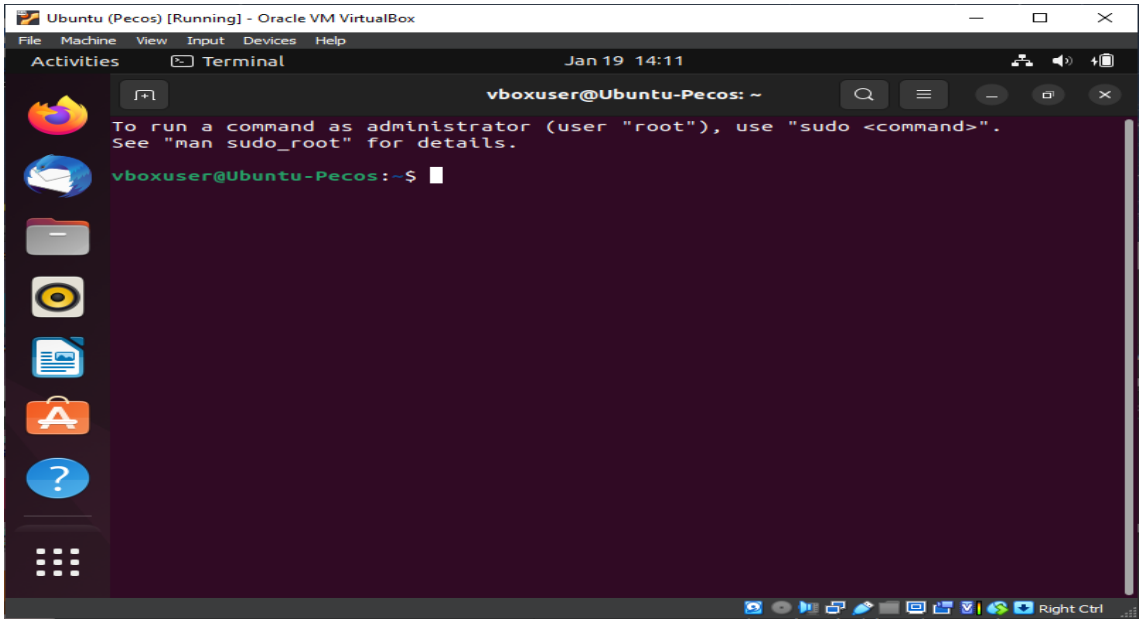
Select user and enter password:



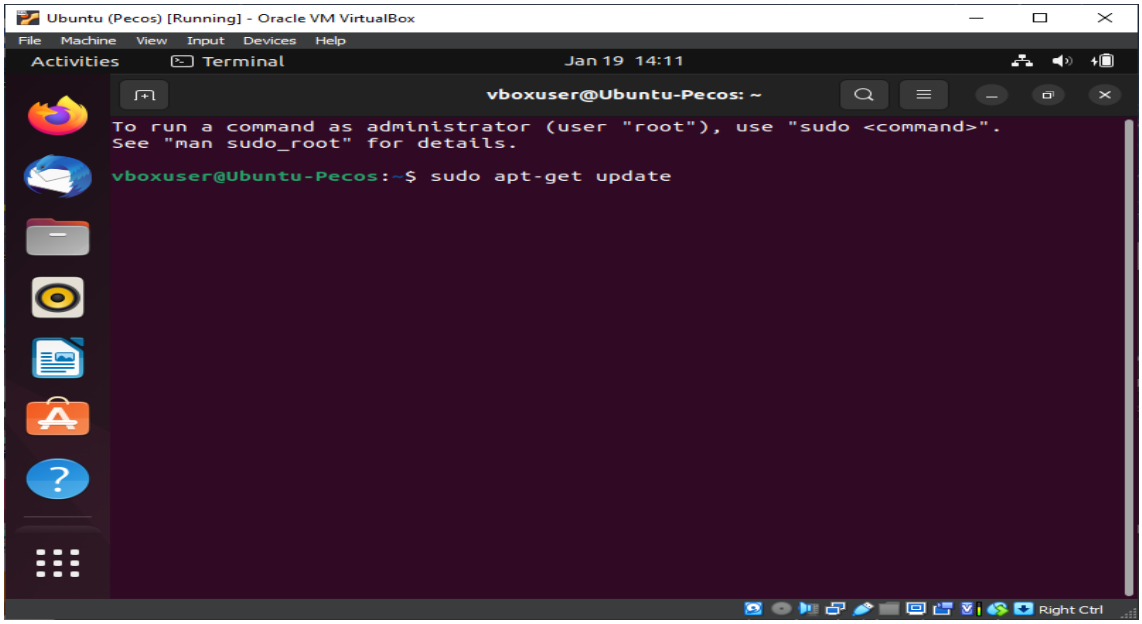
You should now be on the desktop of your VM:



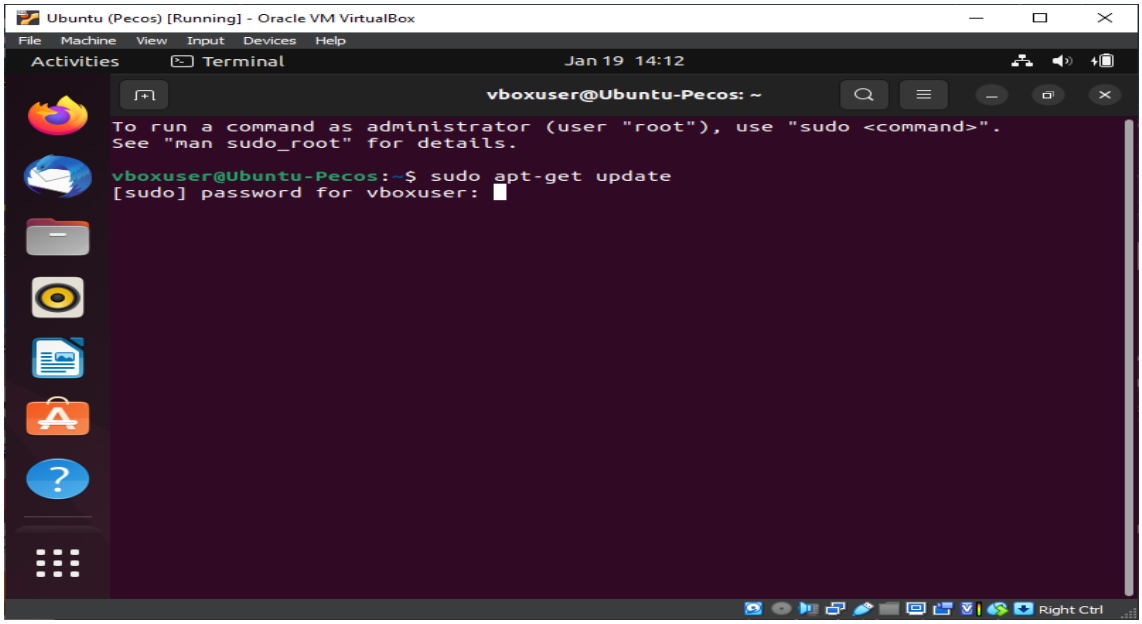
Hit ctrl+alt+t to open the Terminal:



Type the command *sudo apt-get update* and hit Enter:

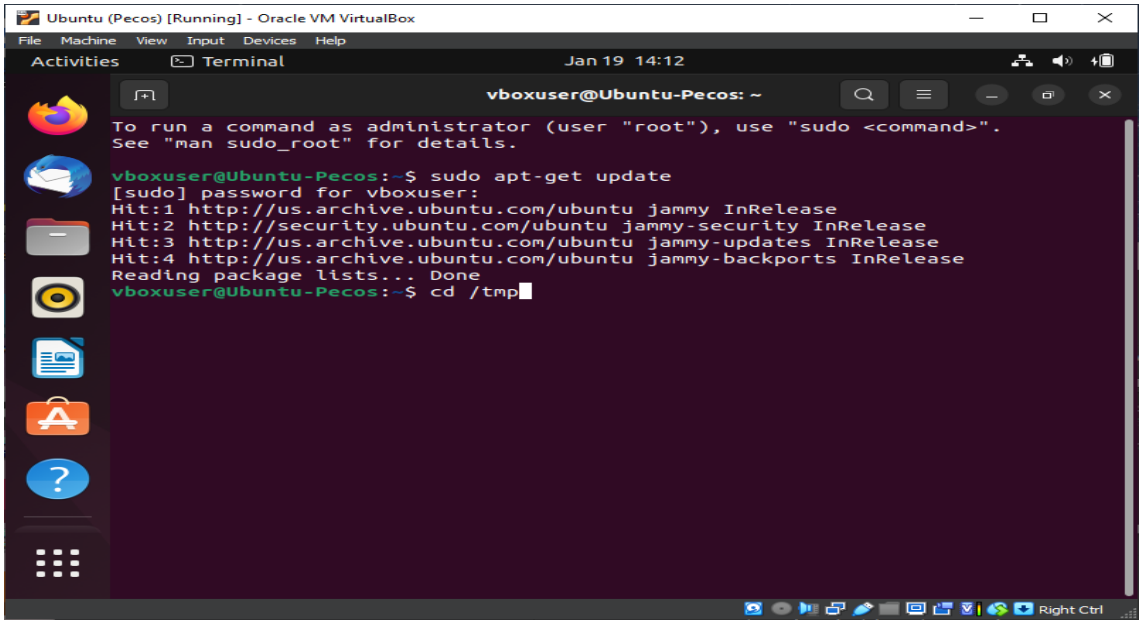


You will now need to enter your password and hit Enter:

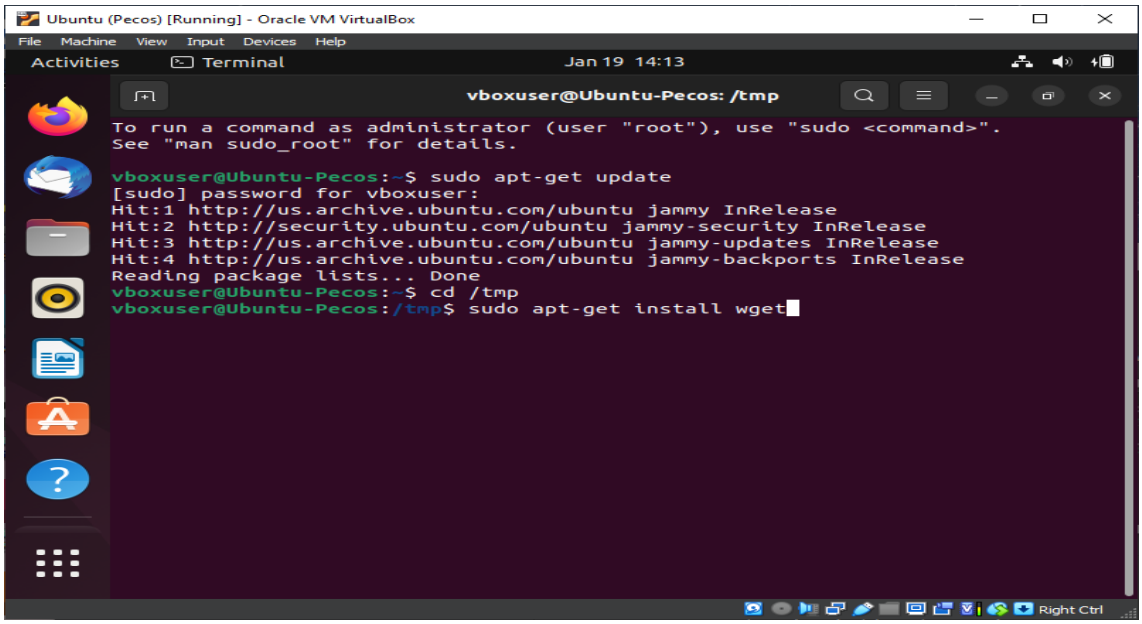


Remember passwords are invisible in the terminal

Now type *cd /tmp* and hit Enter:

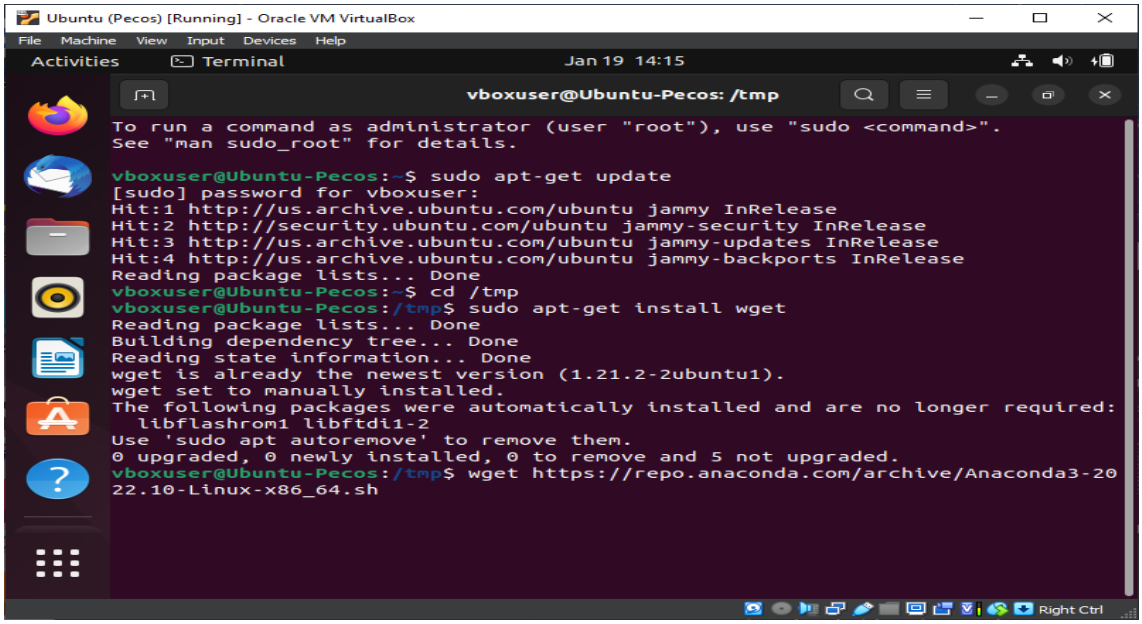


Next, type command *sudo apt-get install wget* and hit Enter:



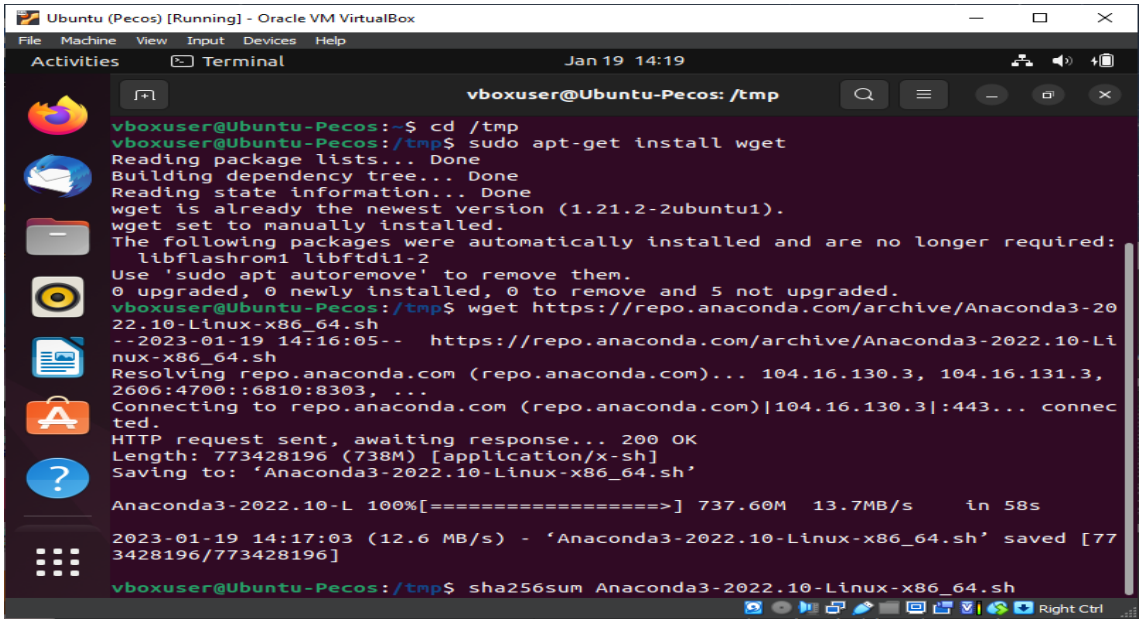
Next, input *wget https://repo.anaconda.com/archive/Anaconda3-2022.10-Linux-x86_64.sh*

and hit Enter:



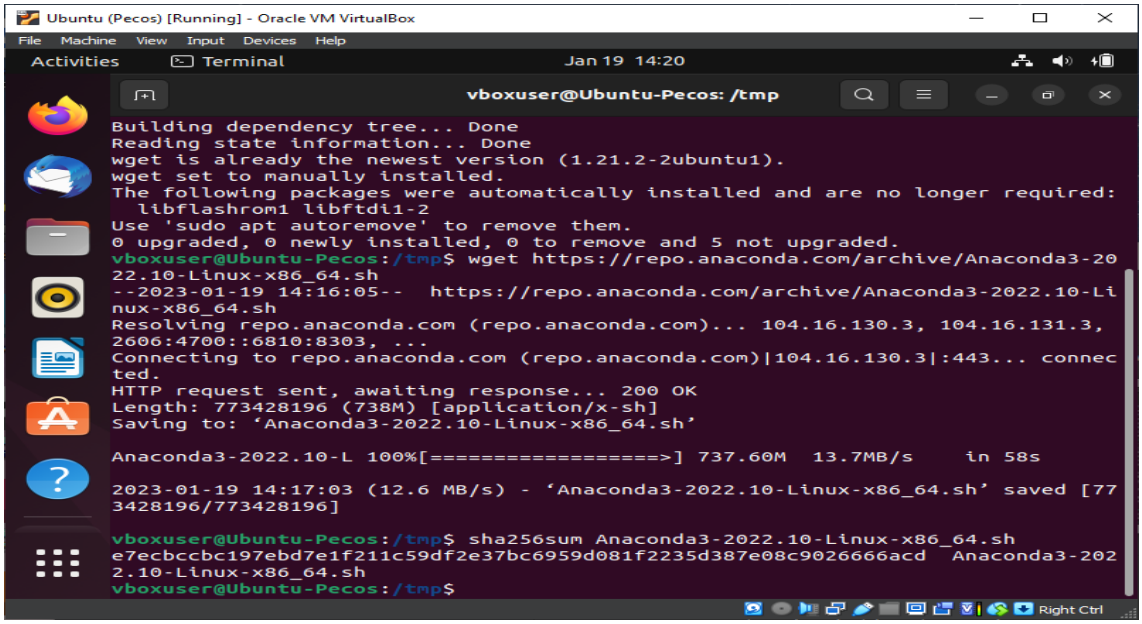
This line will differ depending on version number of Anaconda

Next, input *sha256sum Anaconda3-2022.10-Linux-x86_64.sh* and hit Enter:

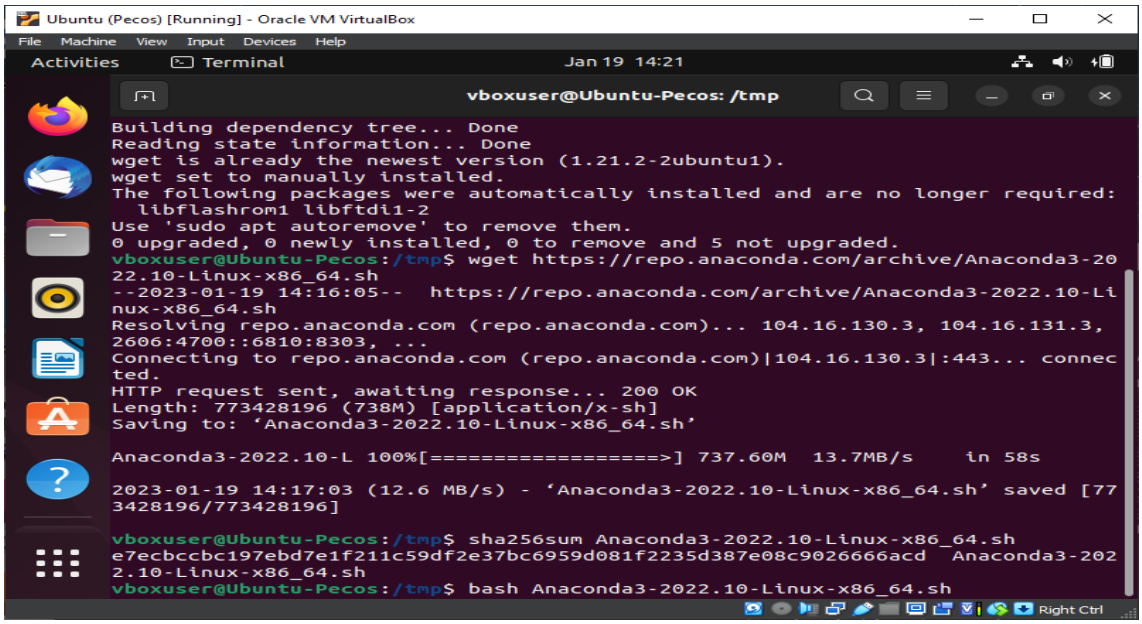


This line will differ depending on version number of Anaconda

If there are no errors, then continue:

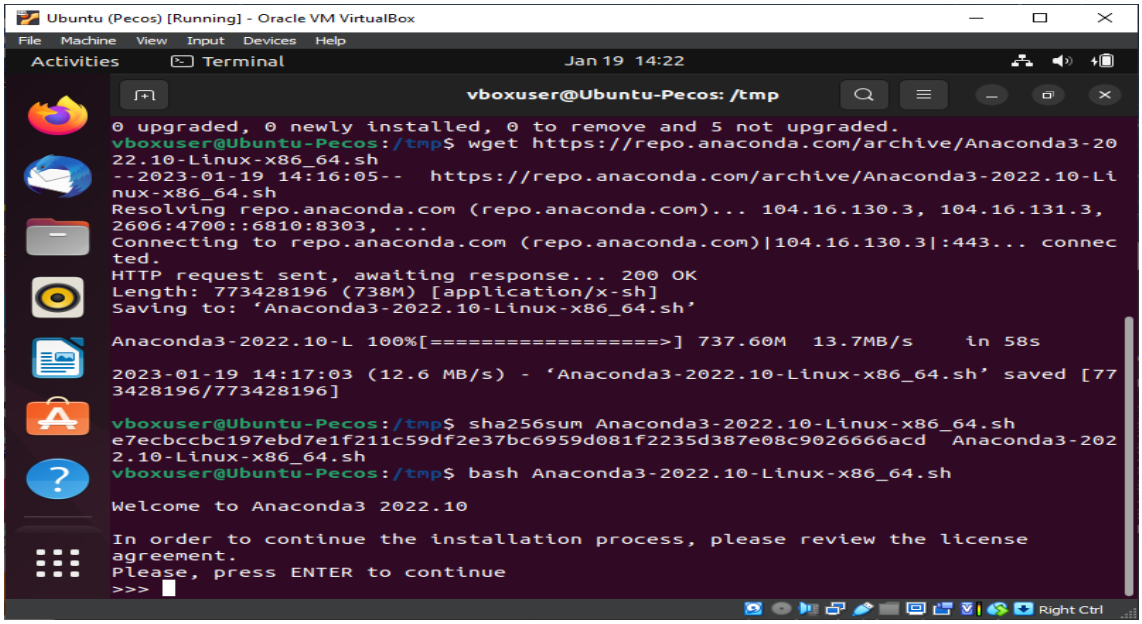


Type *bash Anaconda3-2022.10-Linux-x86_64.sh* and hit Enter:

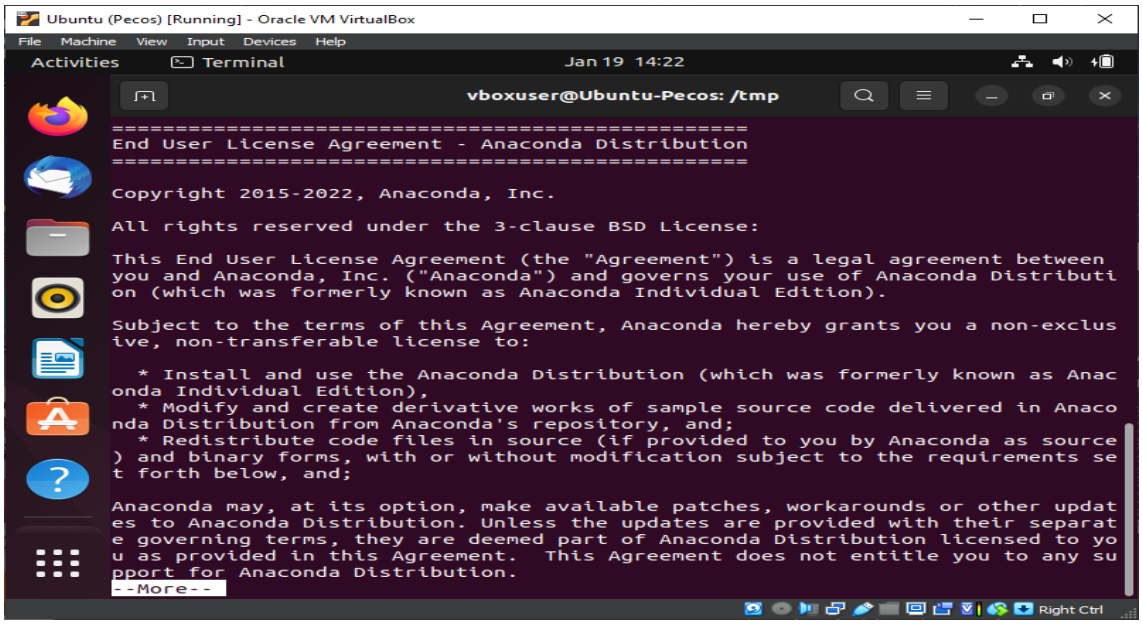


This line will differ depending on version number of Anaconda

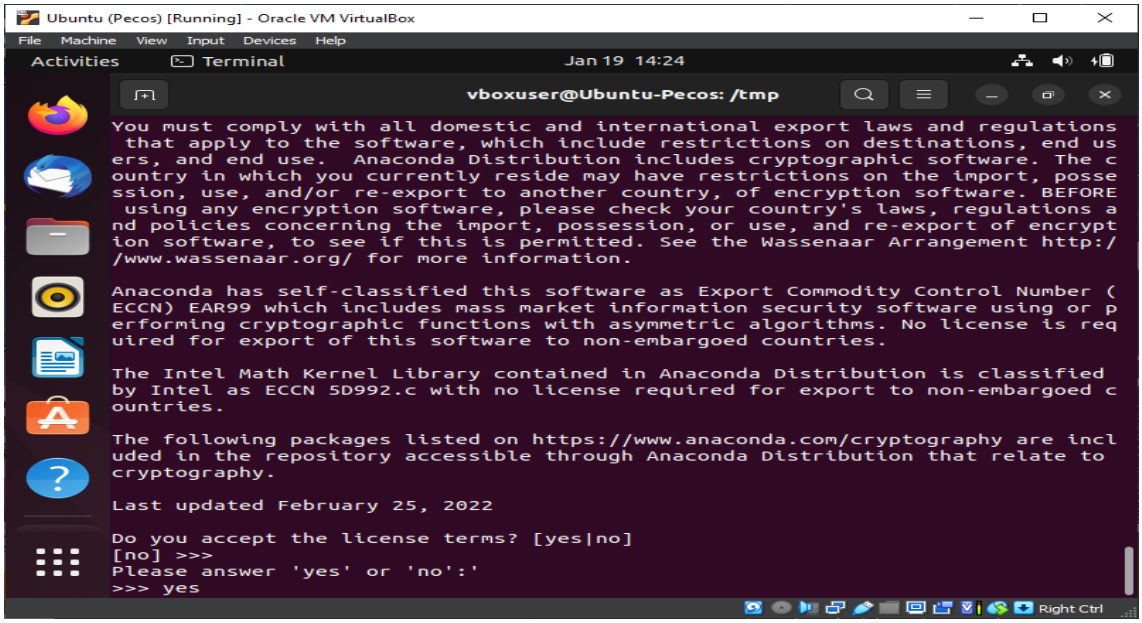
You should now see Anaconda's License Agreement:



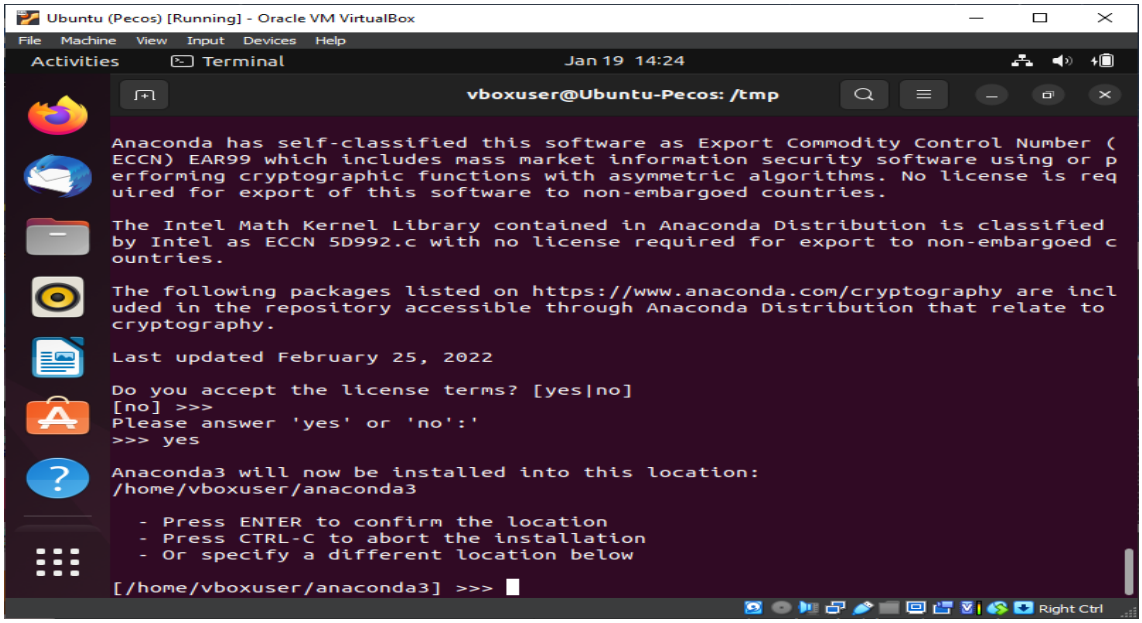
Hit enter until you make it through the agreement:



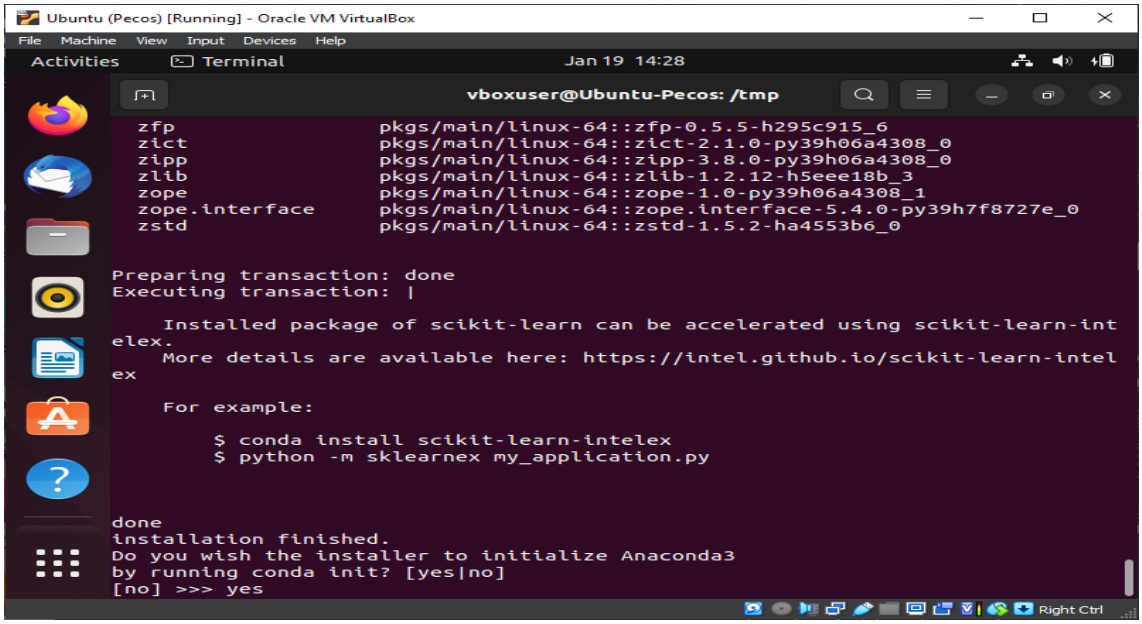
When you get to the end of the Agreement you will need to type *yes* to agree to the terms and conditions and then hit Enter:



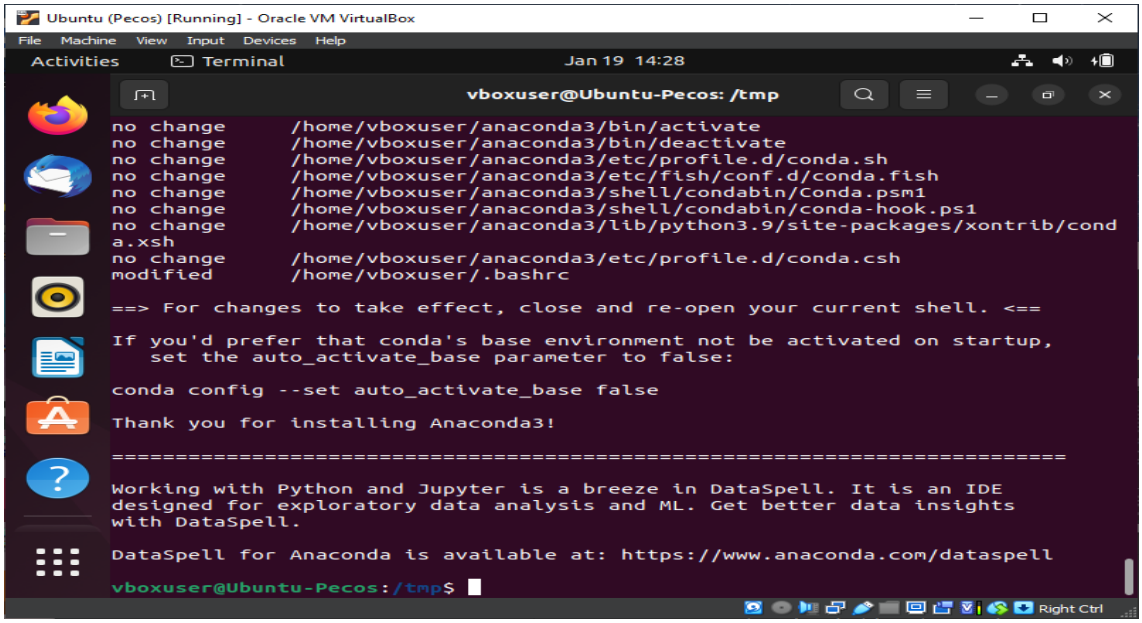
Next, you can choose where Anaconda will be installed. In our case, the specified install location is perfectly fine, so hit Enter:



The installation is now complete. Type *yes* and hit Enter to initialize Anaconda:



Finished, you should now be seeing a screen like the one below:

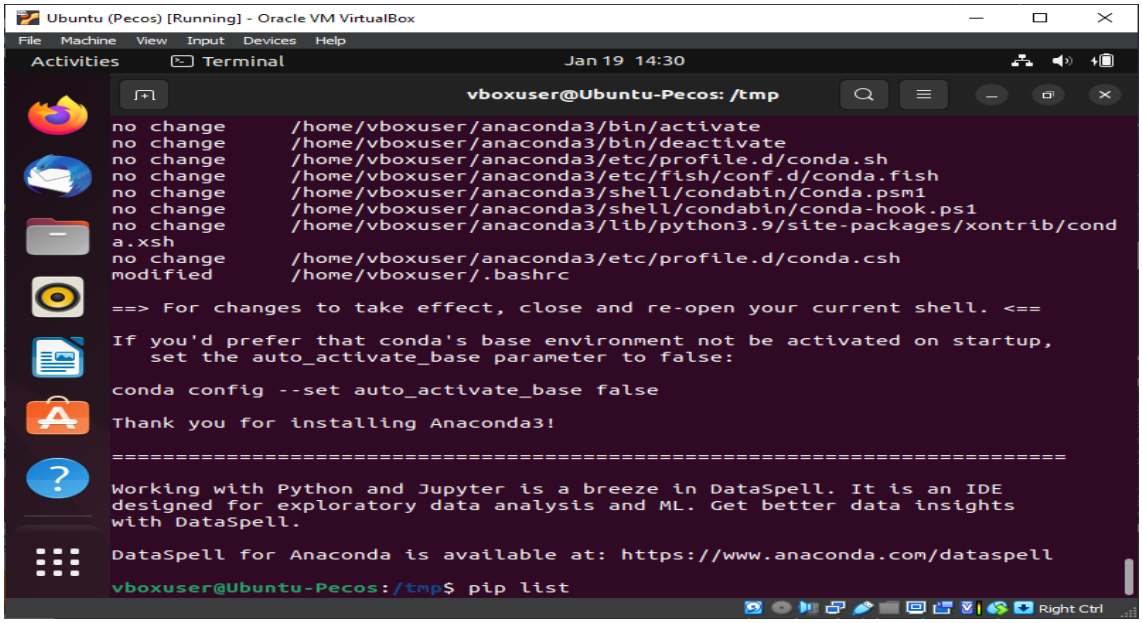


Step 6: Install Needed Packages

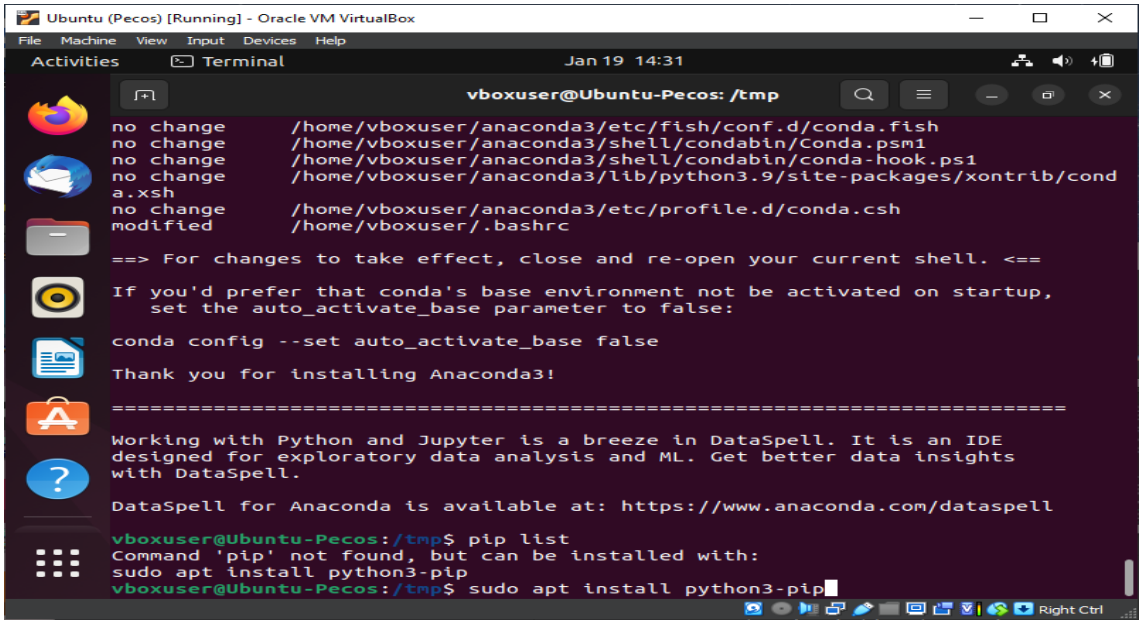
- pandas
- matplotlib
- numpy
- jinja2
- nose
- pvlib
- plotly
- ephem
- sqlalchemy
- pecos

You can check what packages are installed by typing the command *pip list*. Some of these packages may already be installed with Anaconda, however most of these will need to be manually installed. If any packages are missing, use command *pip install* along with the name of the package needed.

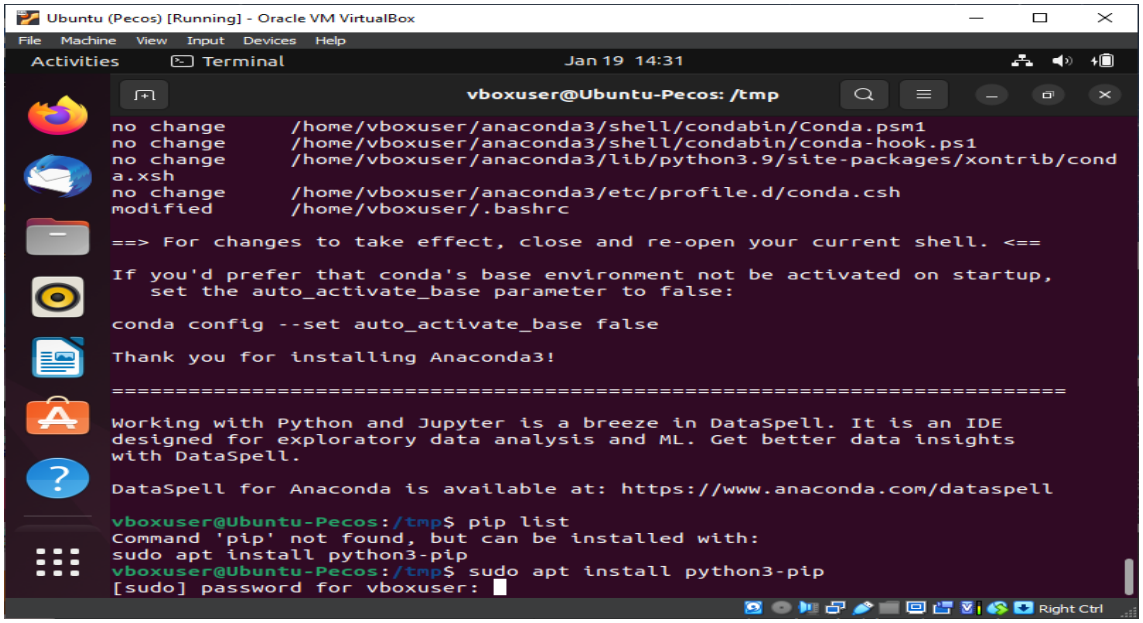
Check for needed packages. Use command *pip list* and hit Enter:



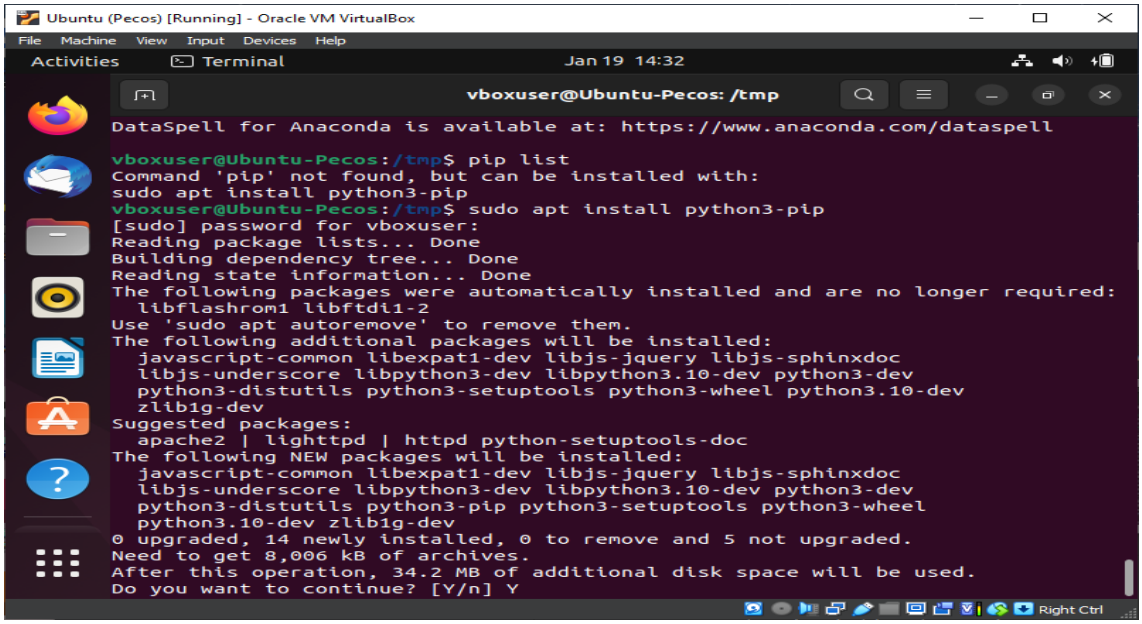
If pip command is not installed, use command *sudo apt install python3-pip* and hit Enter:



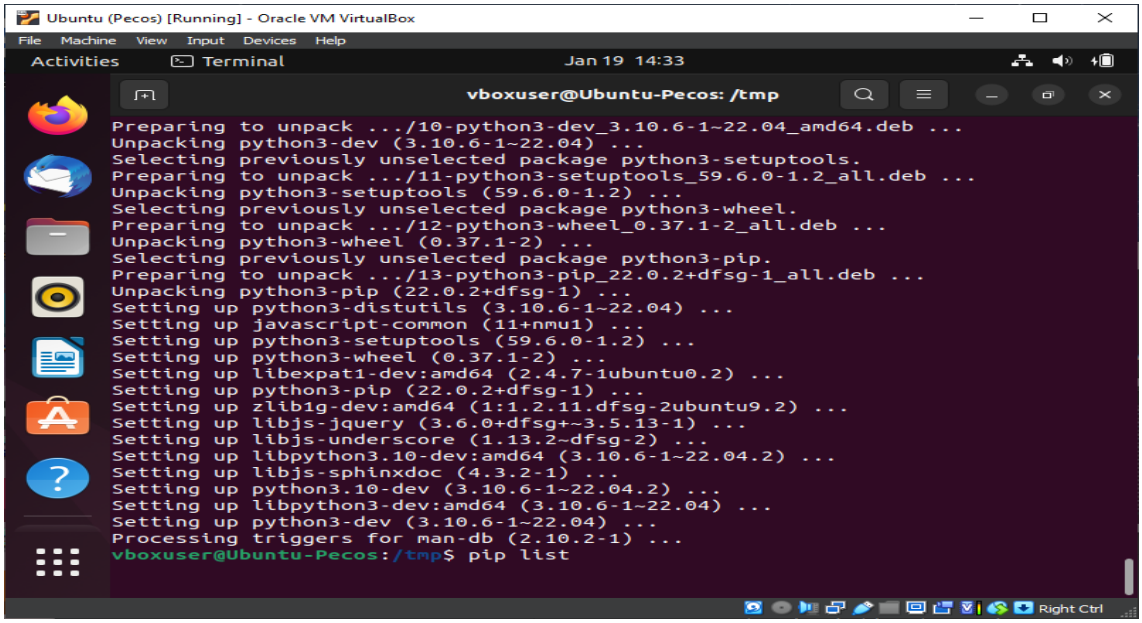
Now, enter password and hit Enter:



Type Y to approve and hit Enter:

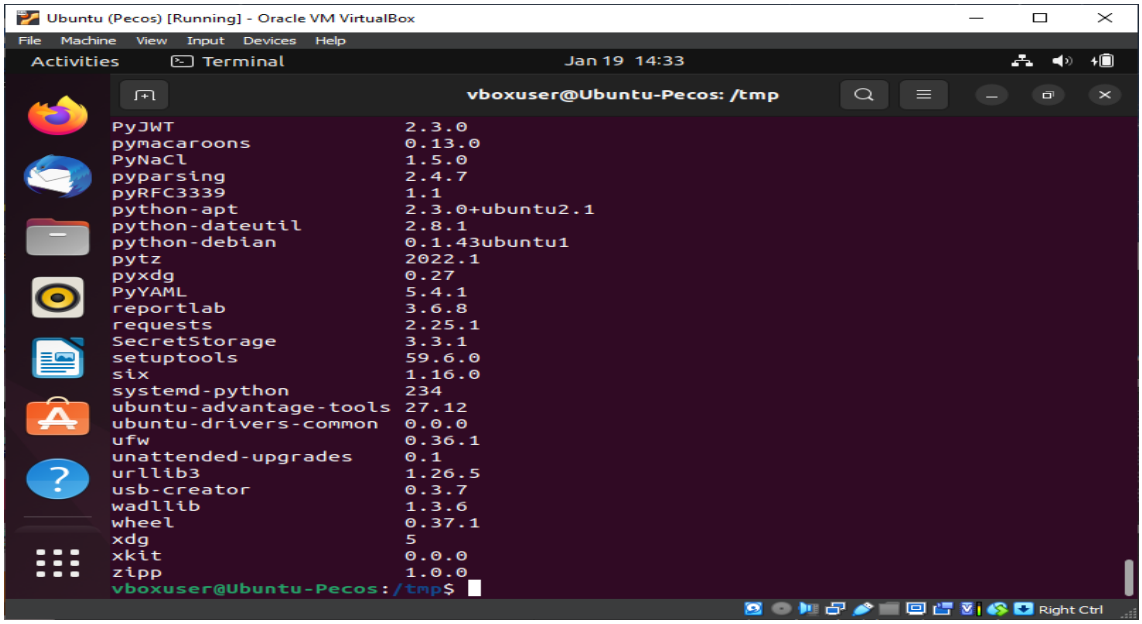


Now type *pip list* and hit Enter:

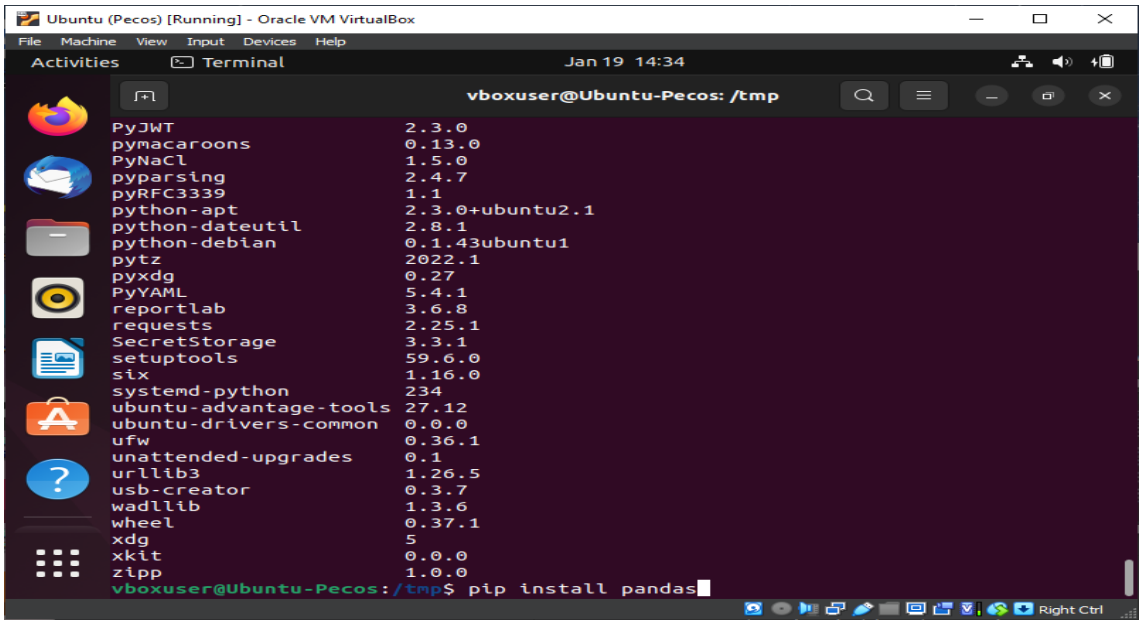


A list of the installed packages should now be visible.

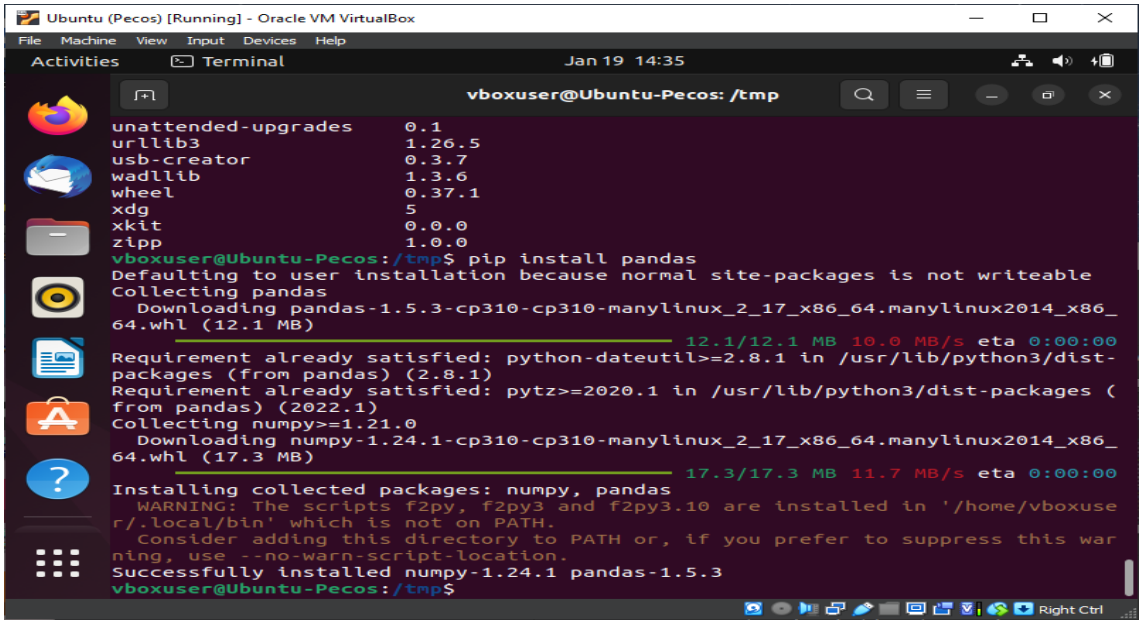
Scroll up and down to check installed packages:



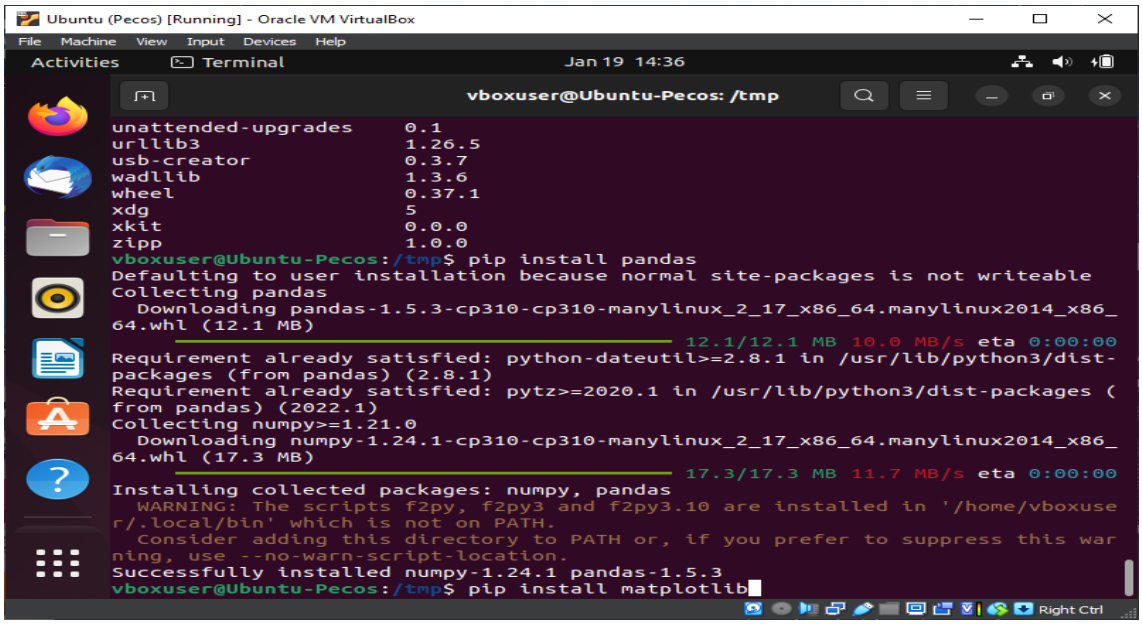
Install *pandas* by typing the command *pip install pandas* and hit Enter:



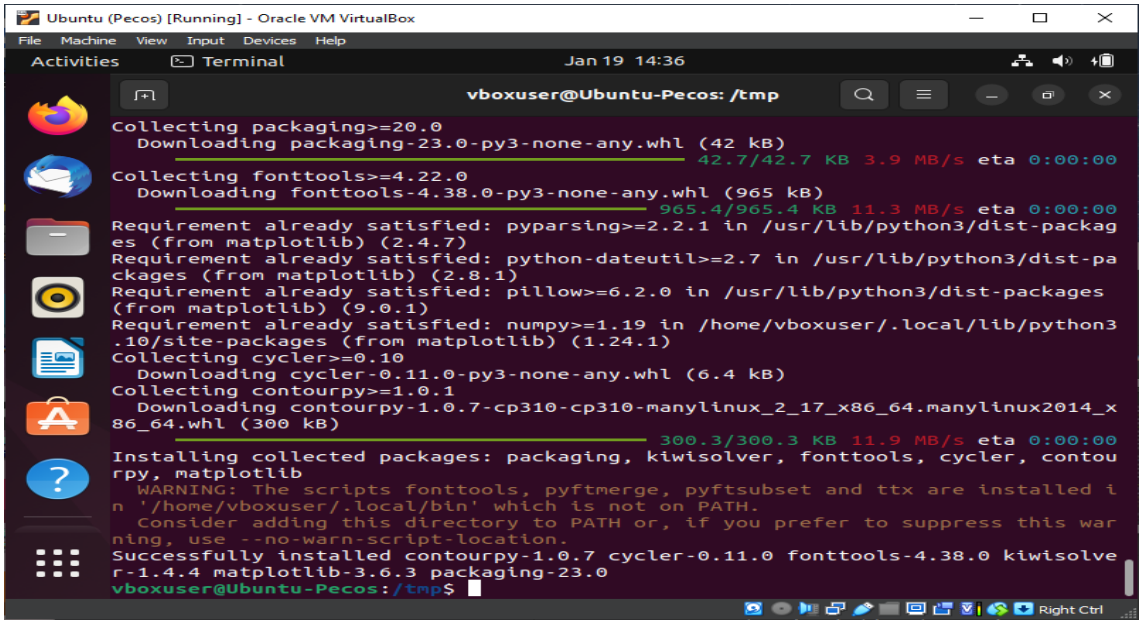
Successful Install:



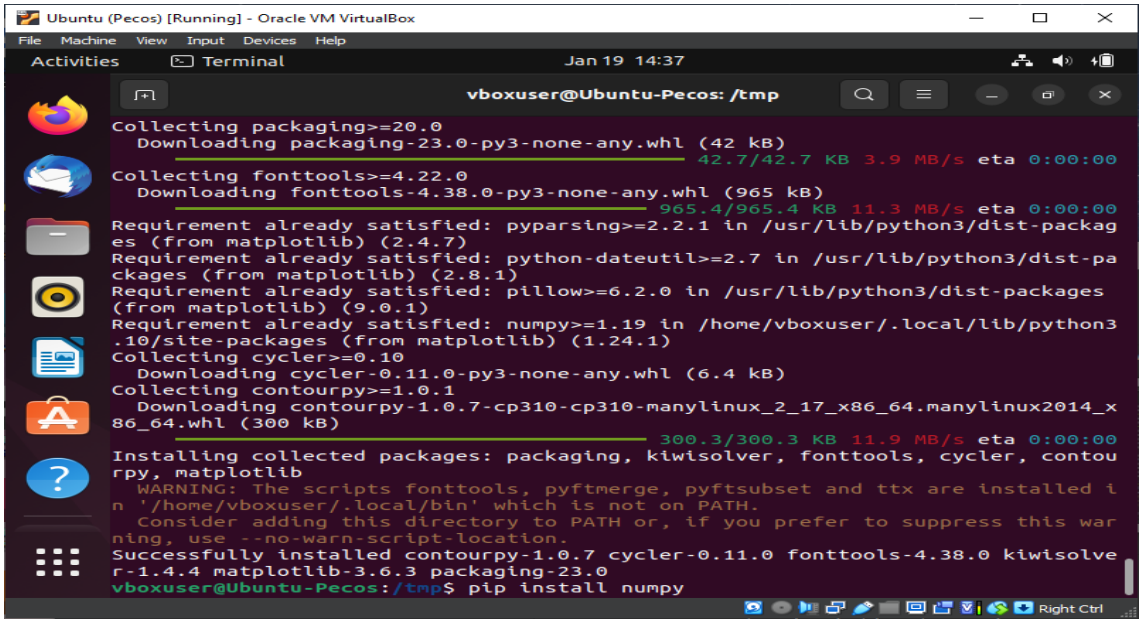
Install *matplotlib* by typing the command *pip install matplotlib* and hit Enter:



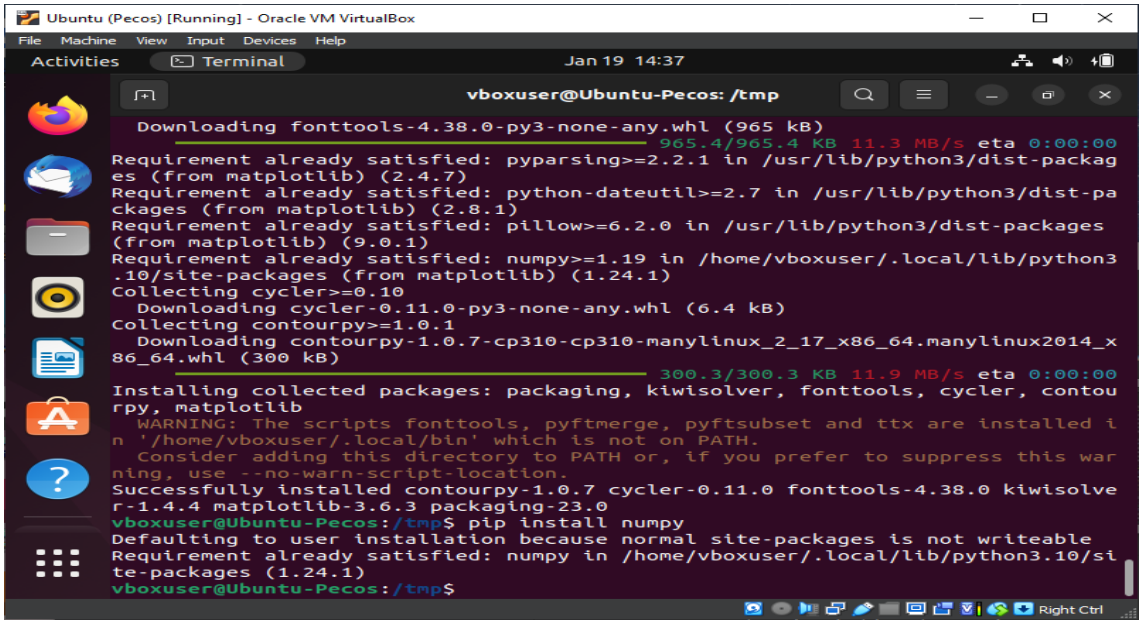
Successful Install:



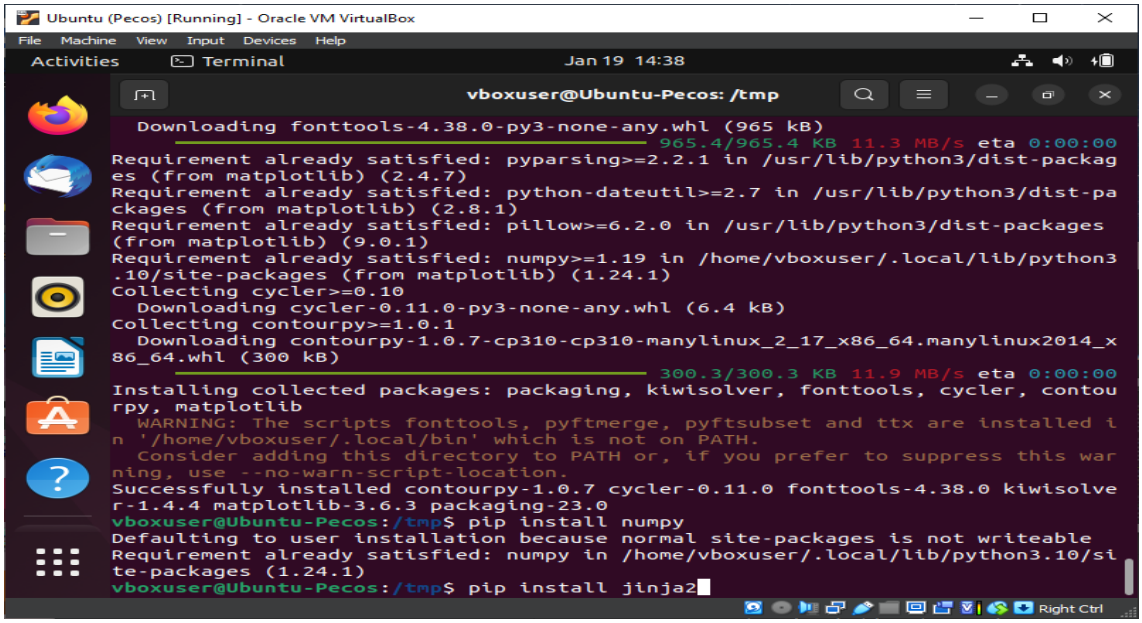
Install *numpy* by typing the command *pip install numpy* and hit Enter:



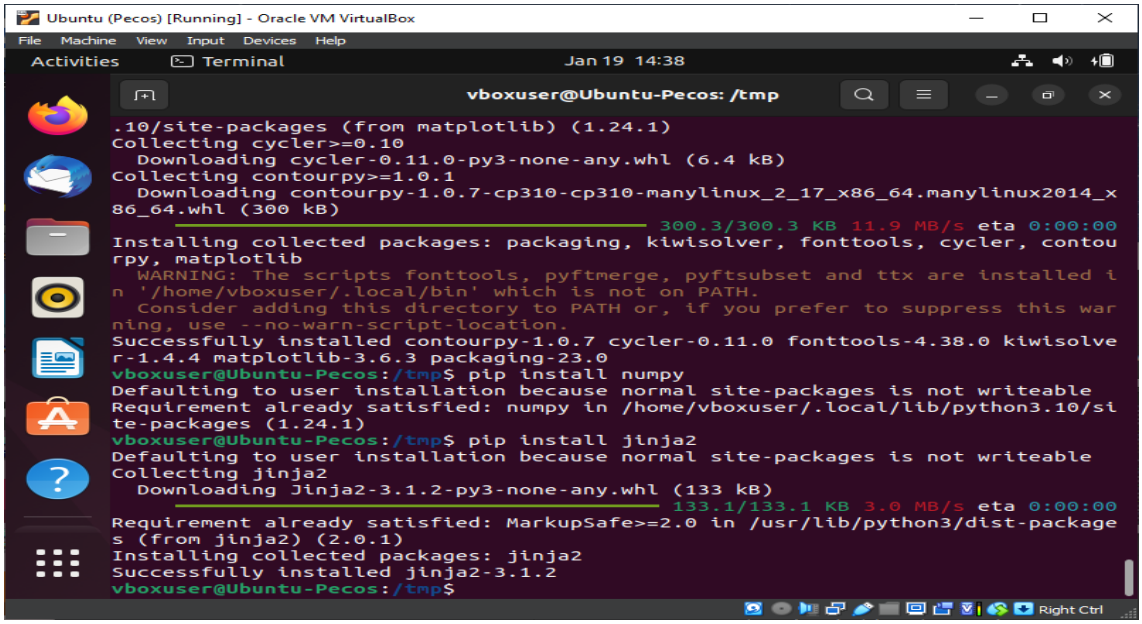
Successful Install:



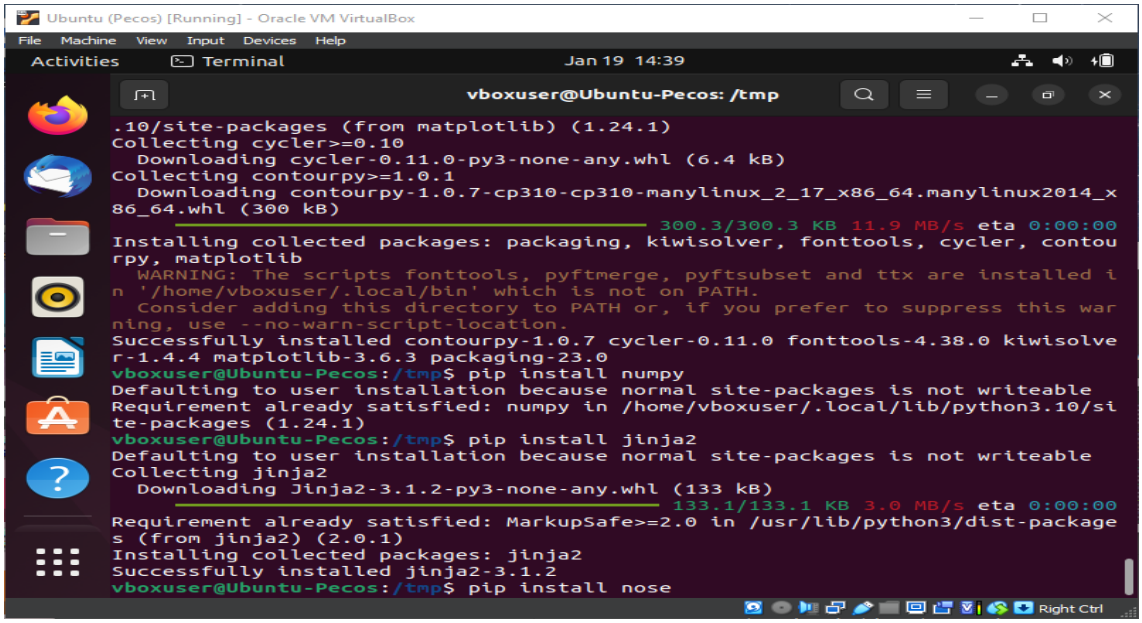
Install *jinja2* by typing the command *pip install jinja2* and hit Enter:



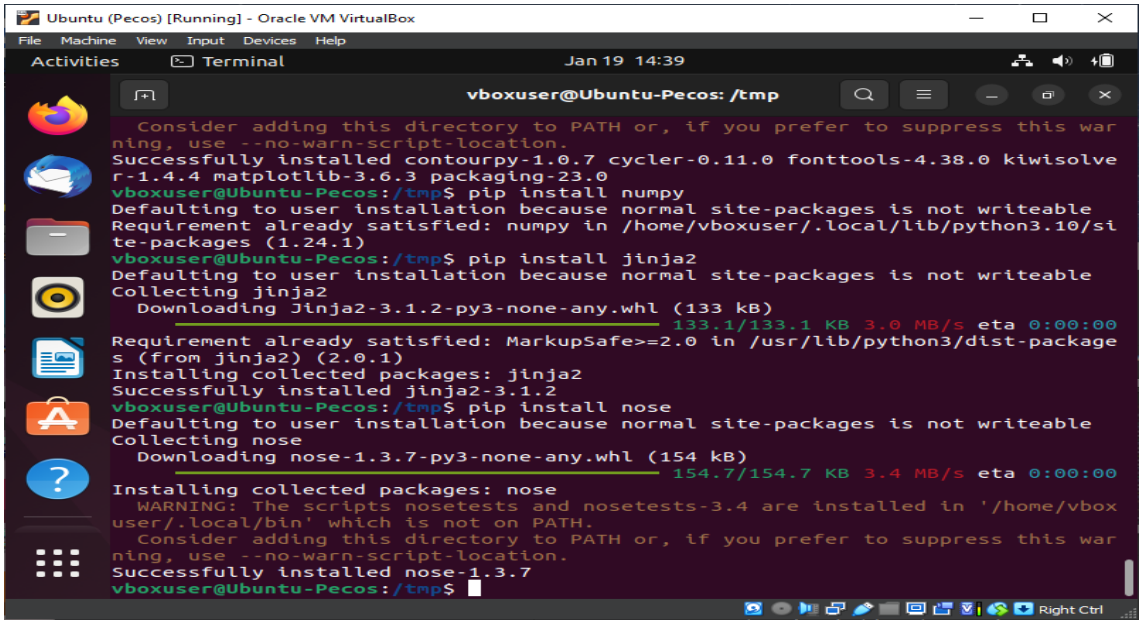
Successful Install:



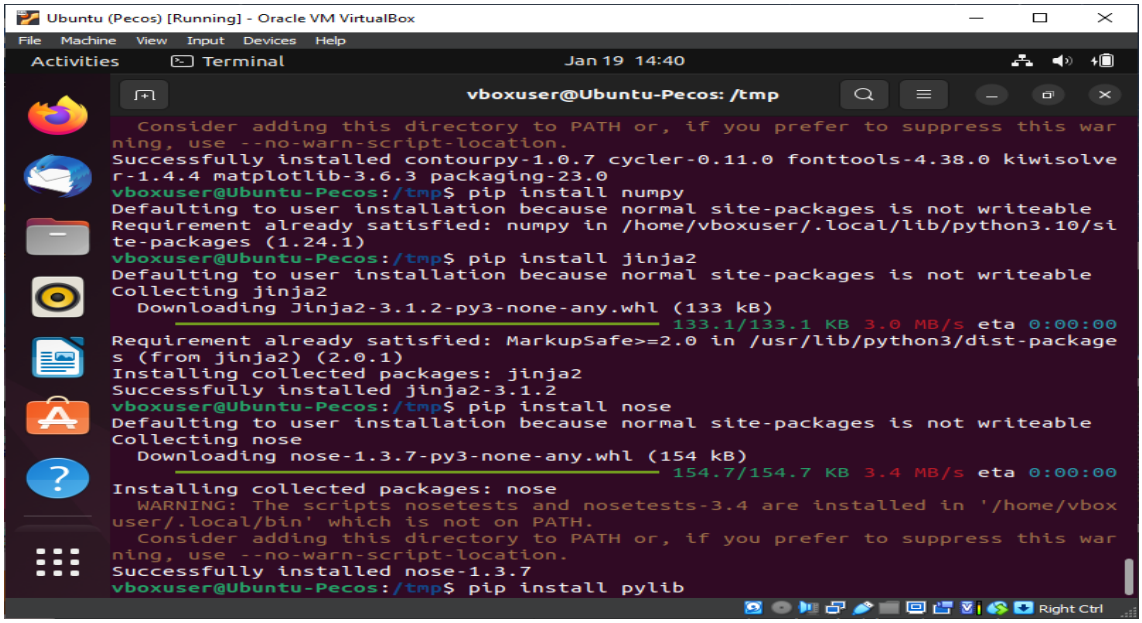
Install *nose* by typing the command *pip install nose* and hit Enter:



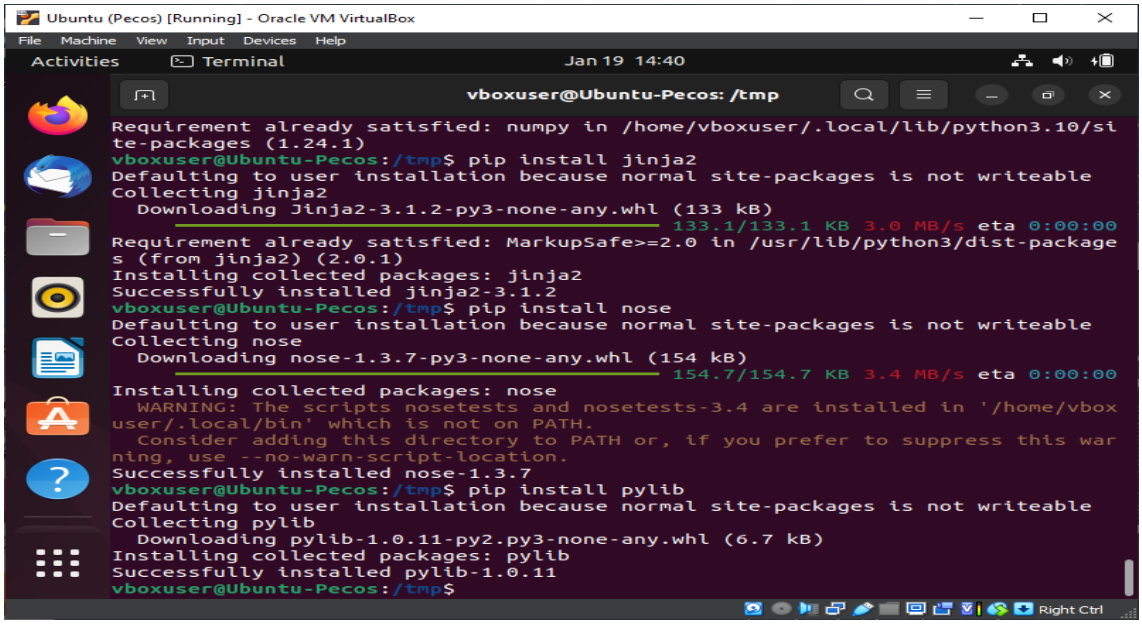
Successful Install:



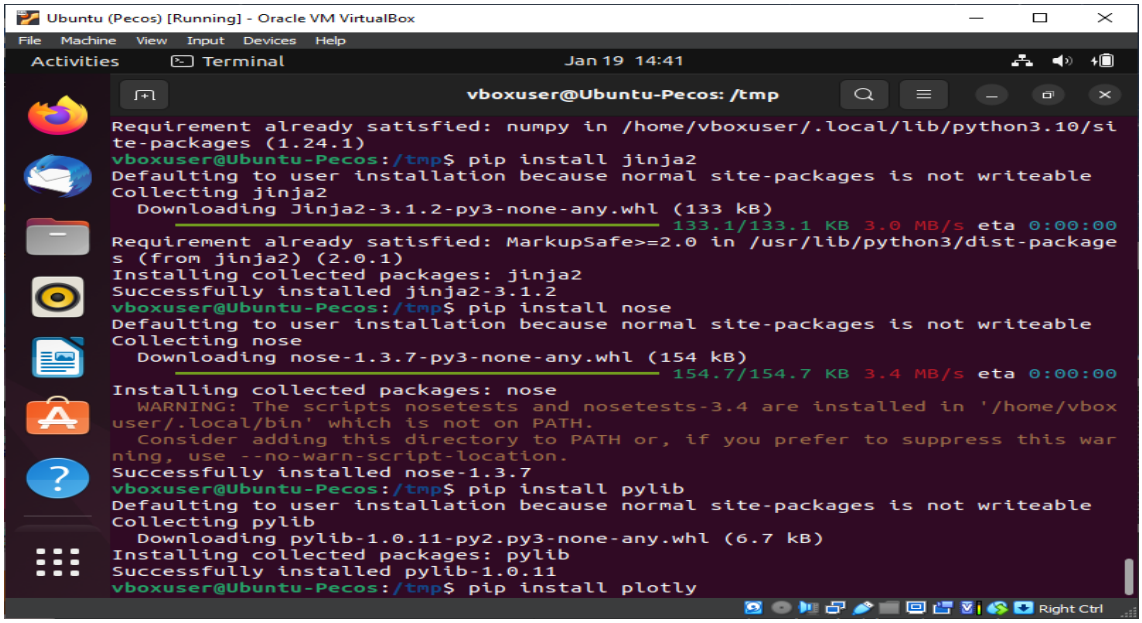
Install *pylib* by typing the command *pip install pylib* and hit Enter:



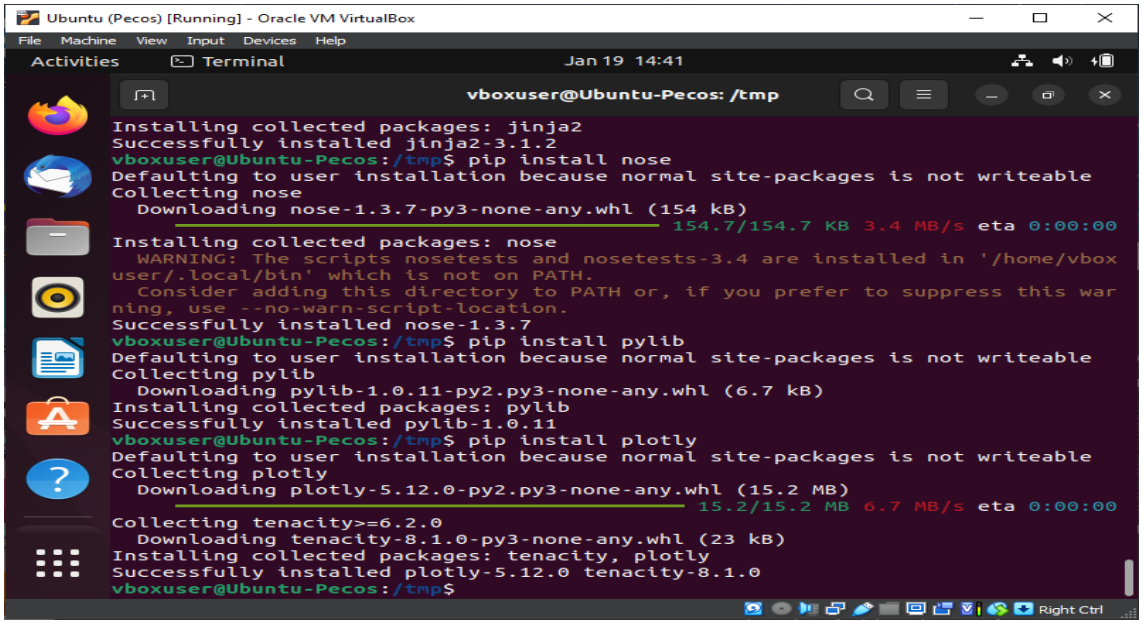
Successful Install:



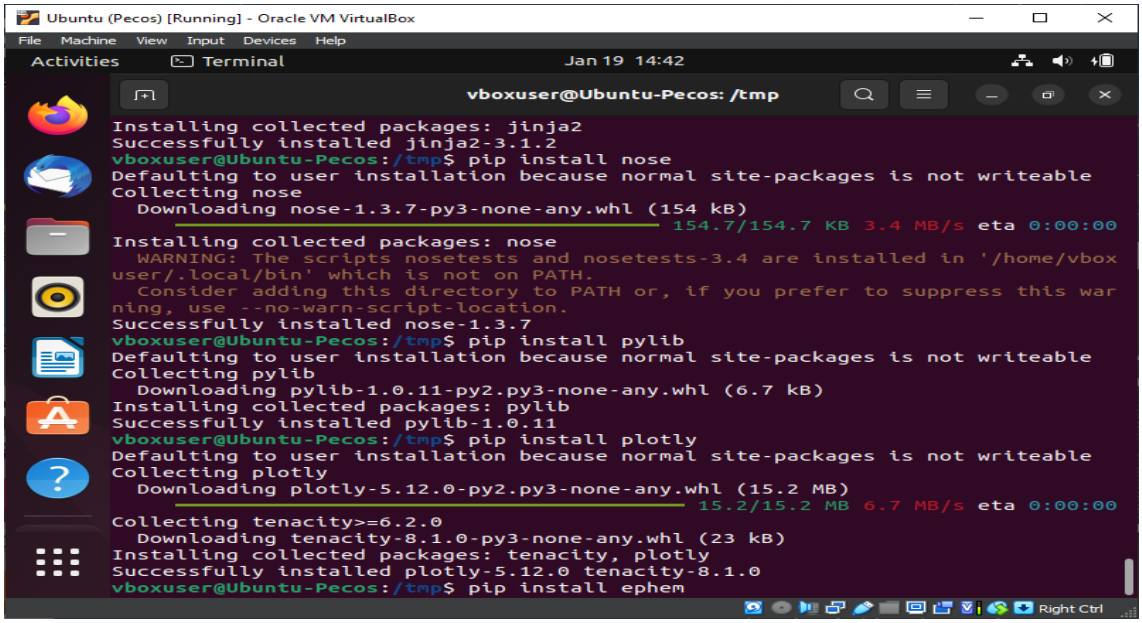
Install *plotly* by typing the command *pip install plotly* and hit Enter:



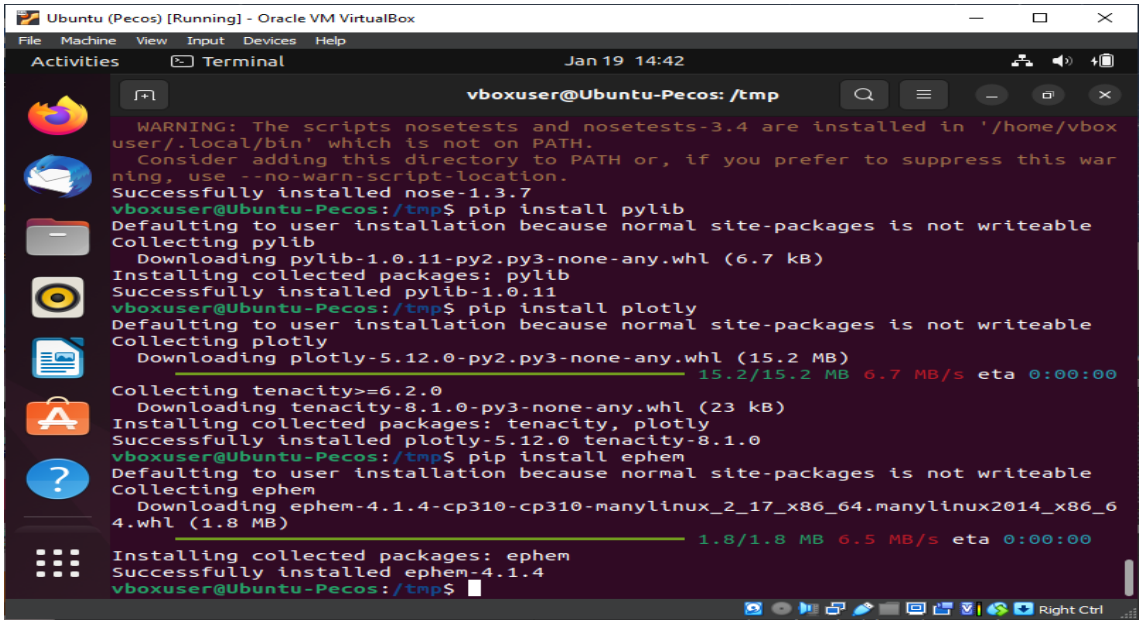
Successful Install:



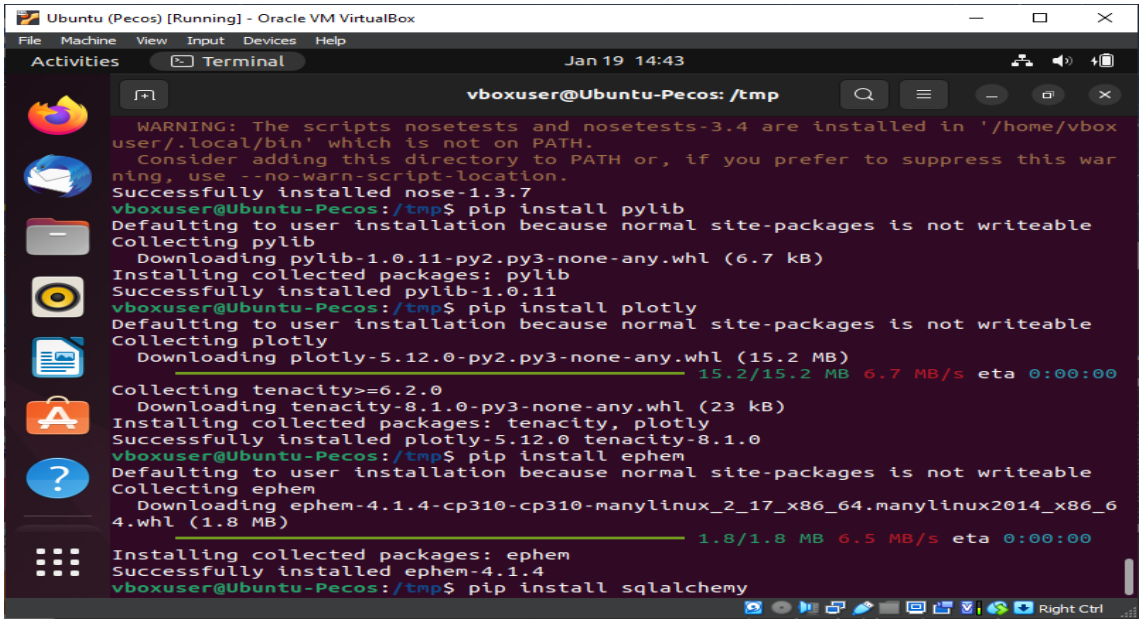
Install *ephem* by typing the command *pip install ephem* and hit Enter:



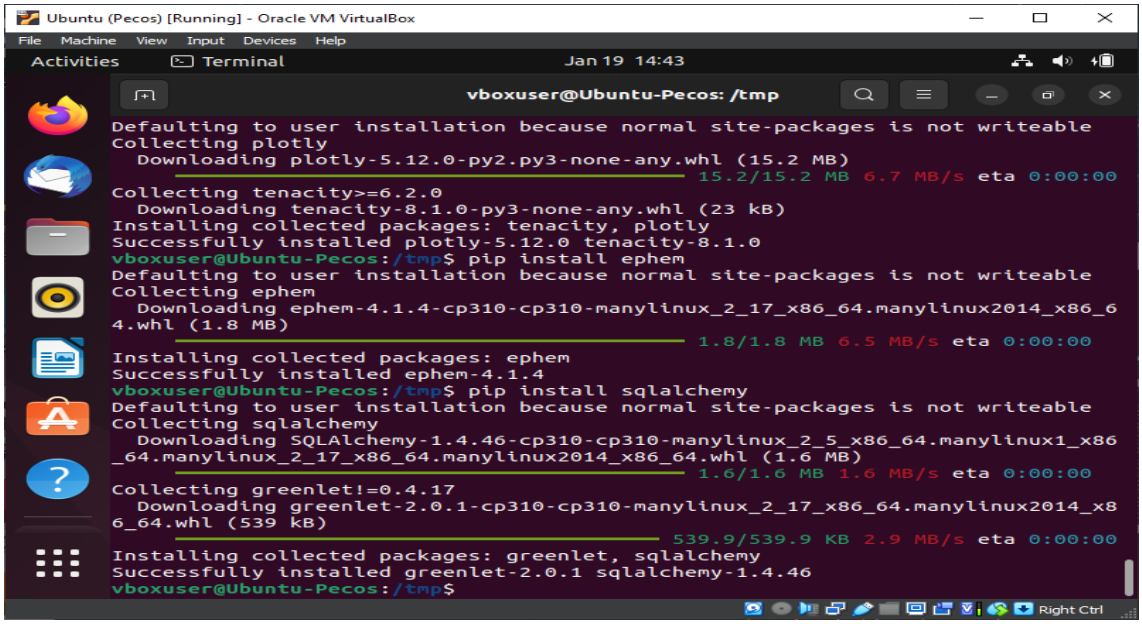
Successful Install:



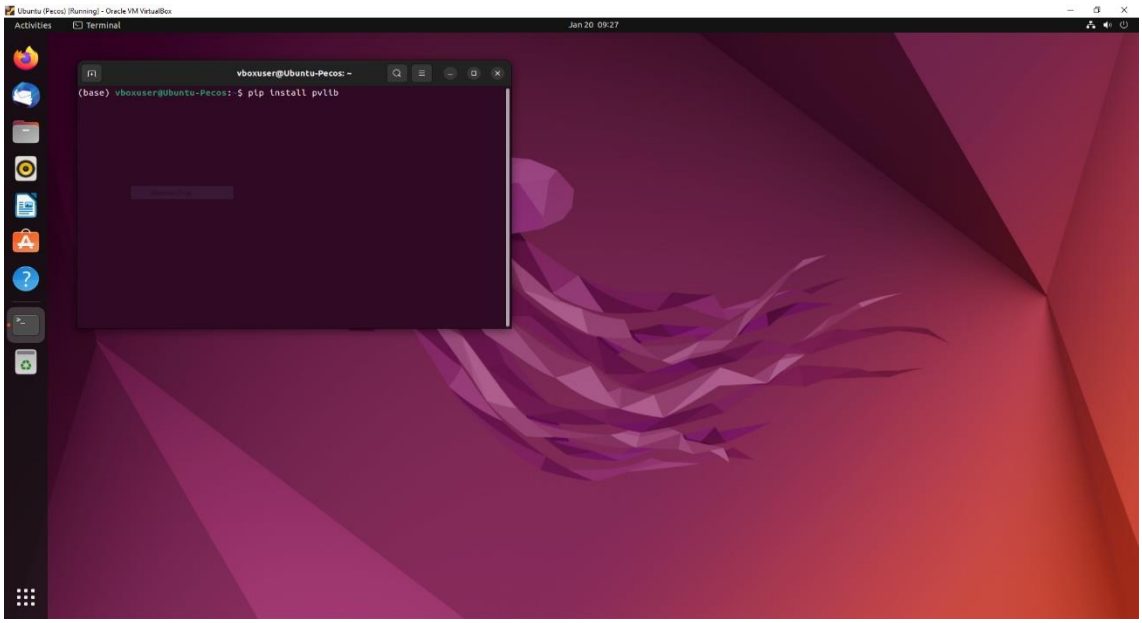
Install *sqlalchemy* by typing the command *pip install sqlalchemy* and hit Enter:



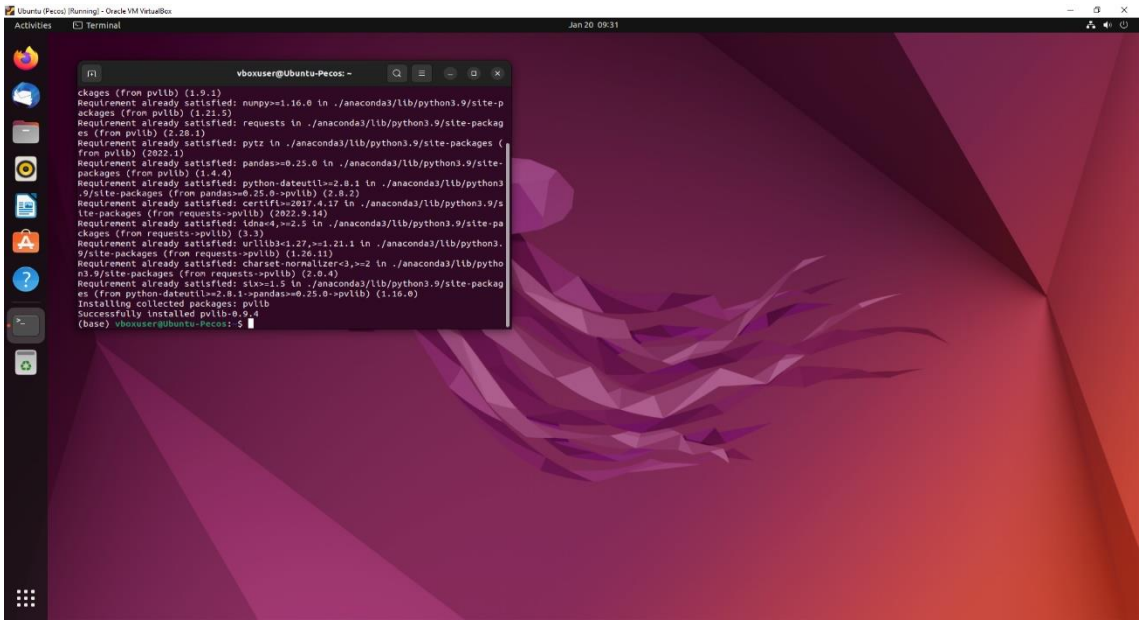
Successful Install:



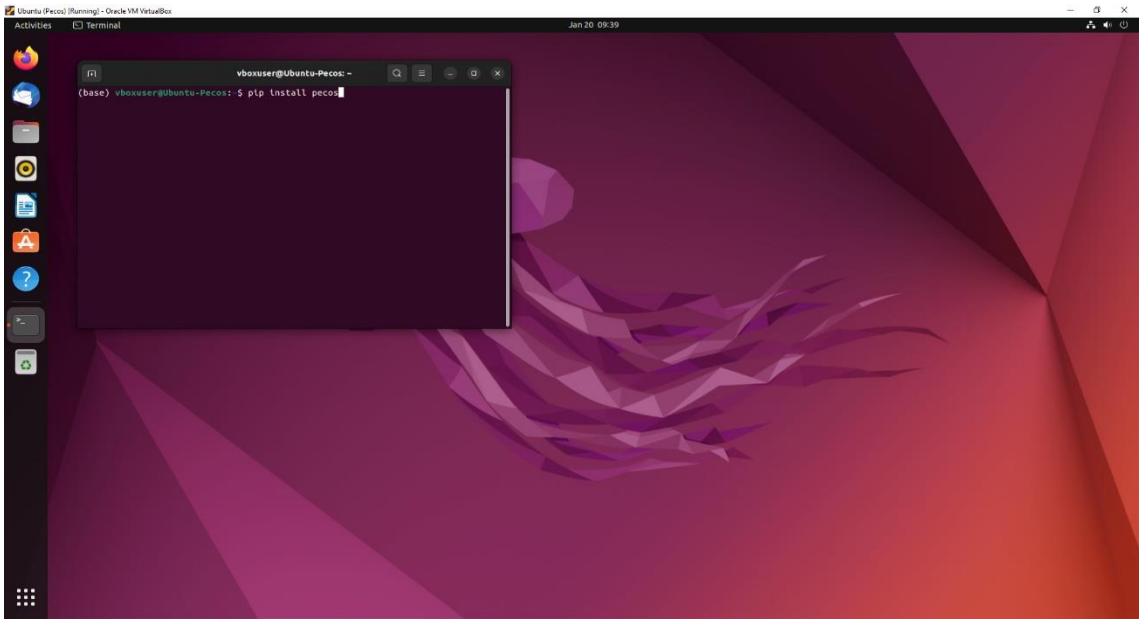
Install *pvlib* by typing the command ***pip install pvlib*** and hit Enter:



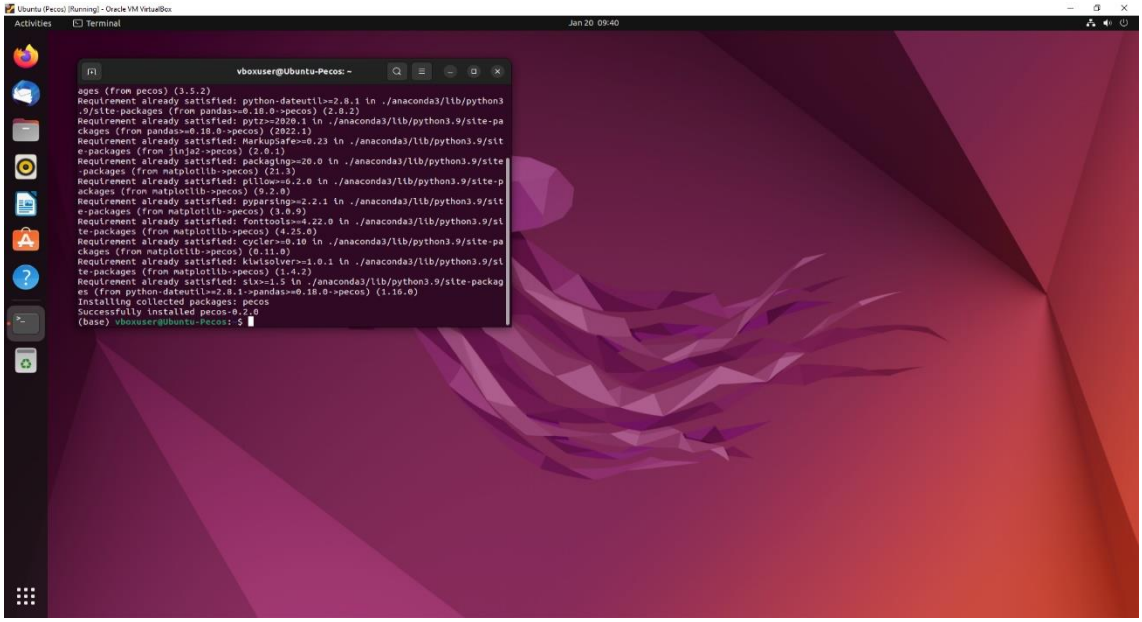
Successful Install:



Install *pecos* by typing the command *pip install pecos* and hit Enter:



Successful Install:



Step 7: Download Modified Pecos zip file on Virtual Machine

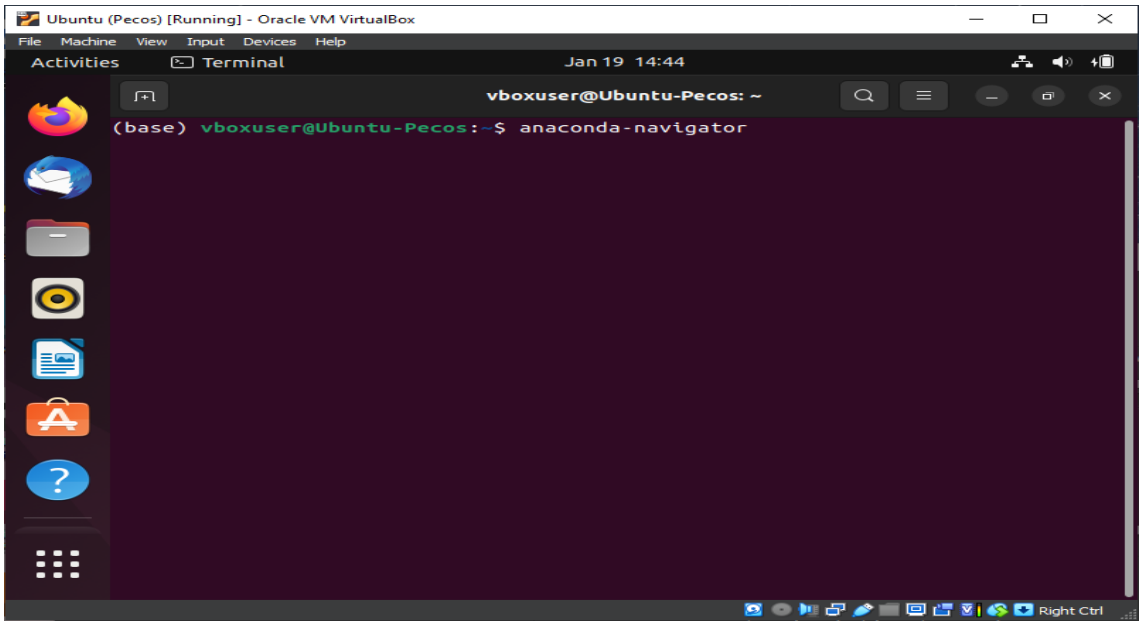
Visit: https://github.com/wcsims42/Quality-Assurance.git

Original Source Files: https://github.com/sandialabs/pecos

Source files will not work without being altered. Use first source

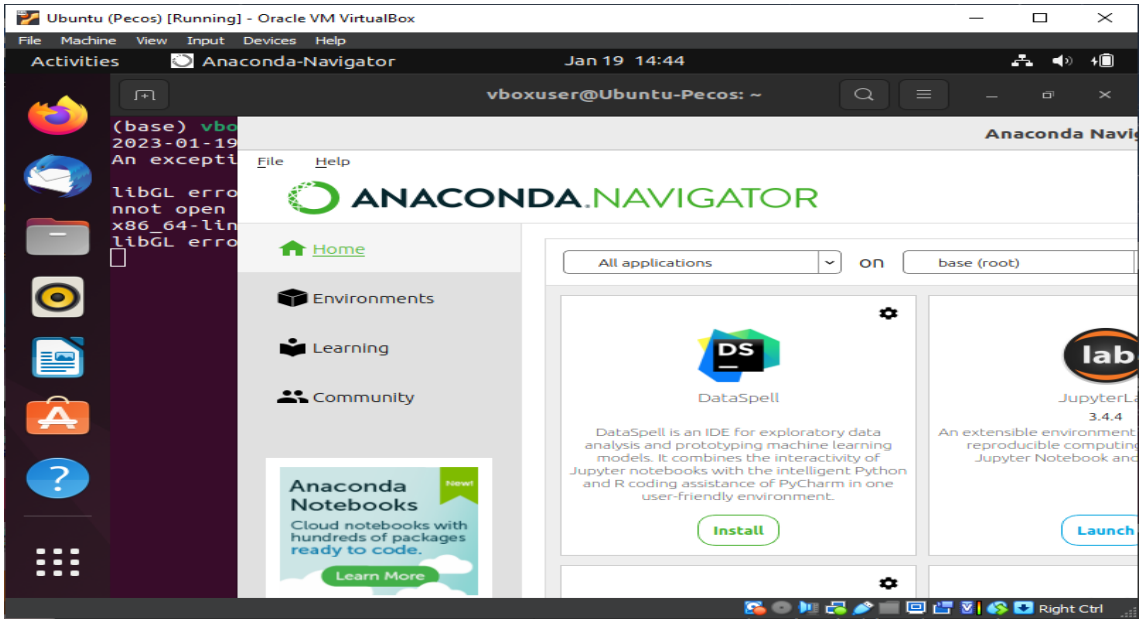
Step 8: Launch Anaconda Navigator

Open Terminal, type *anaconda-navigator* and hit Enter:



Anaconda Navigator should then begin loading.

An array of applications should now be available to you like shown below:

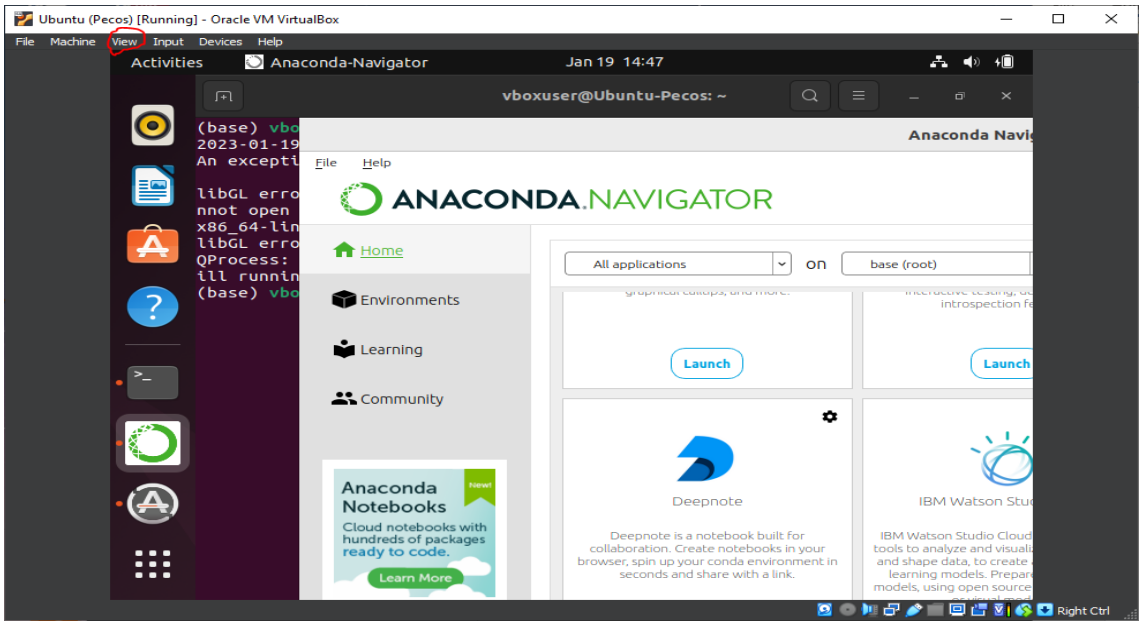


As you can probably tell, the scale and resolution of the VM is all wrong.

This will be fixed in the following step.

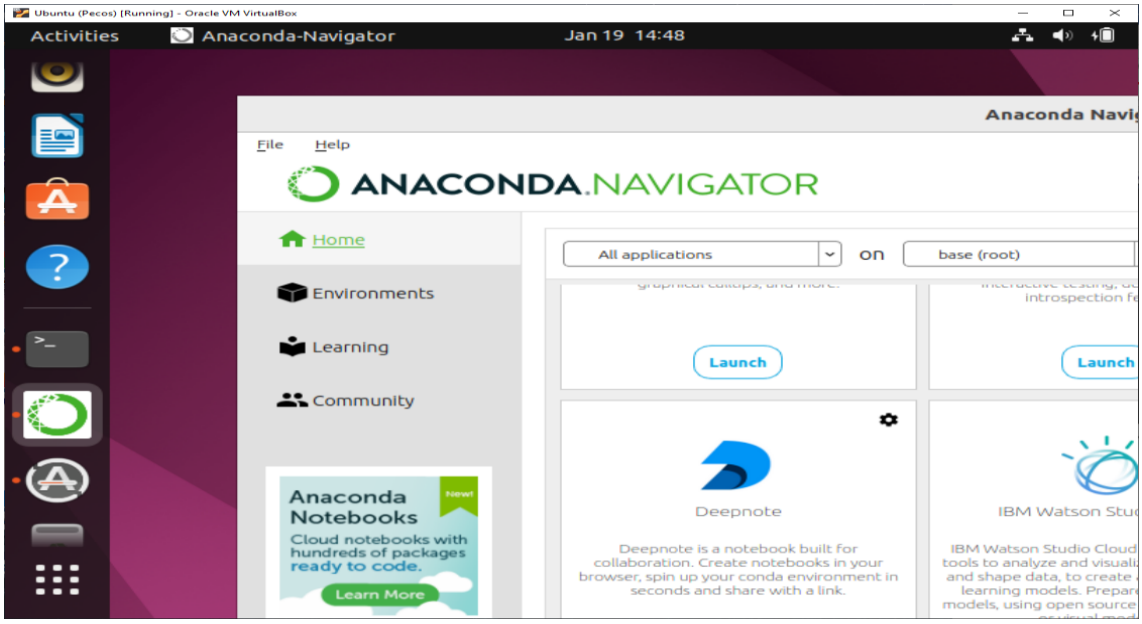
Step 9: Change View and Resolution of VM

Click on View in the top left corner of the VM window and select Scaled Mode:

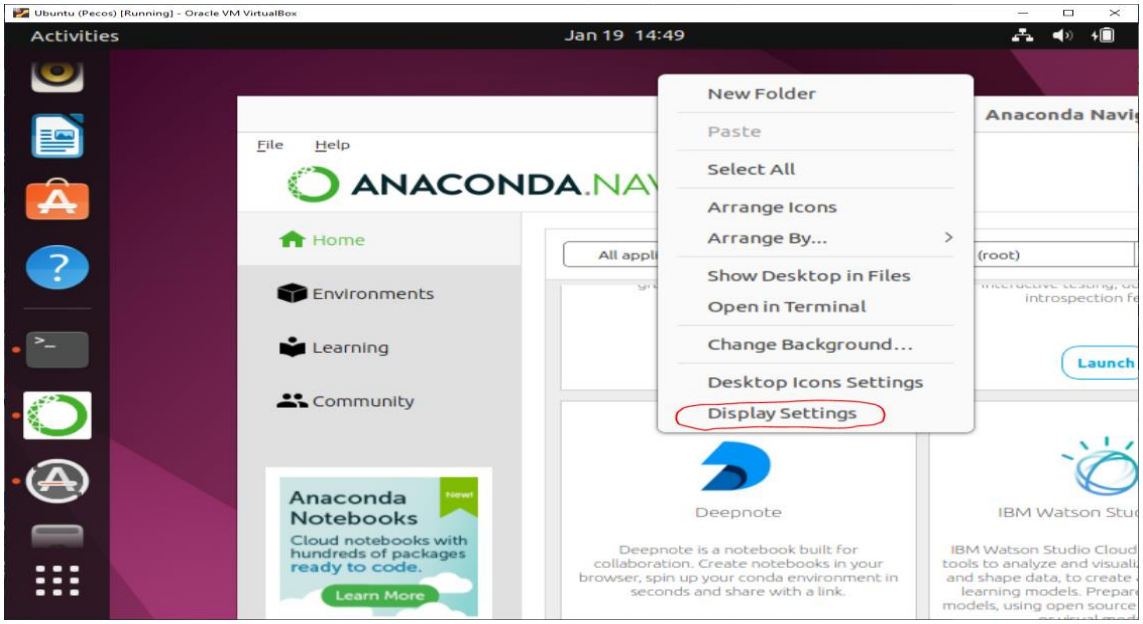


The VM should now be full screen

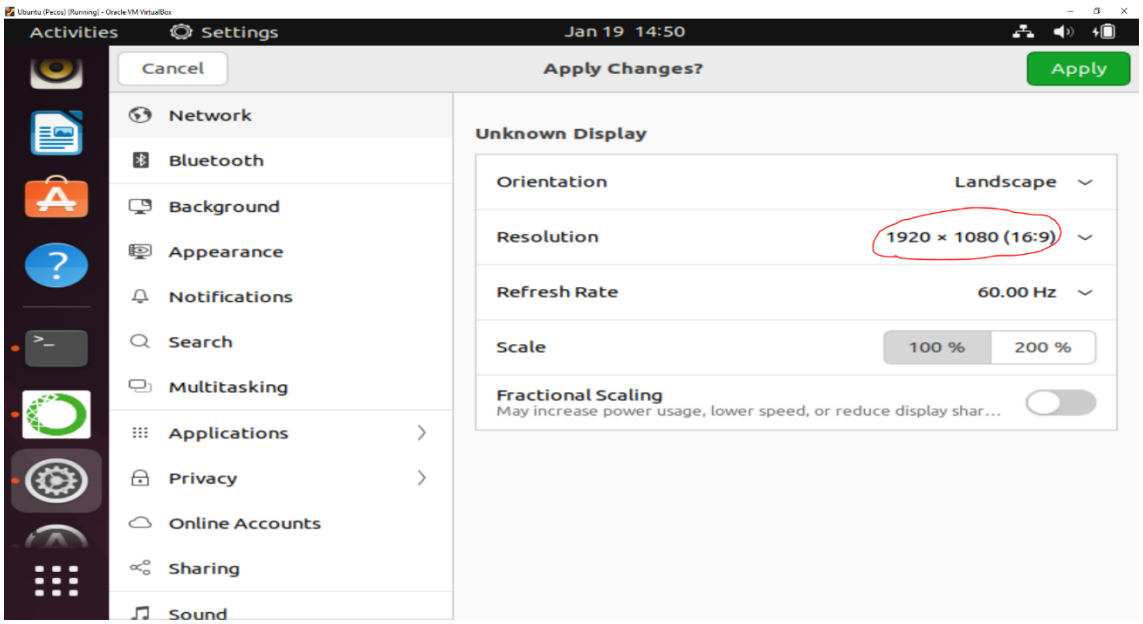
Right click on desktop:



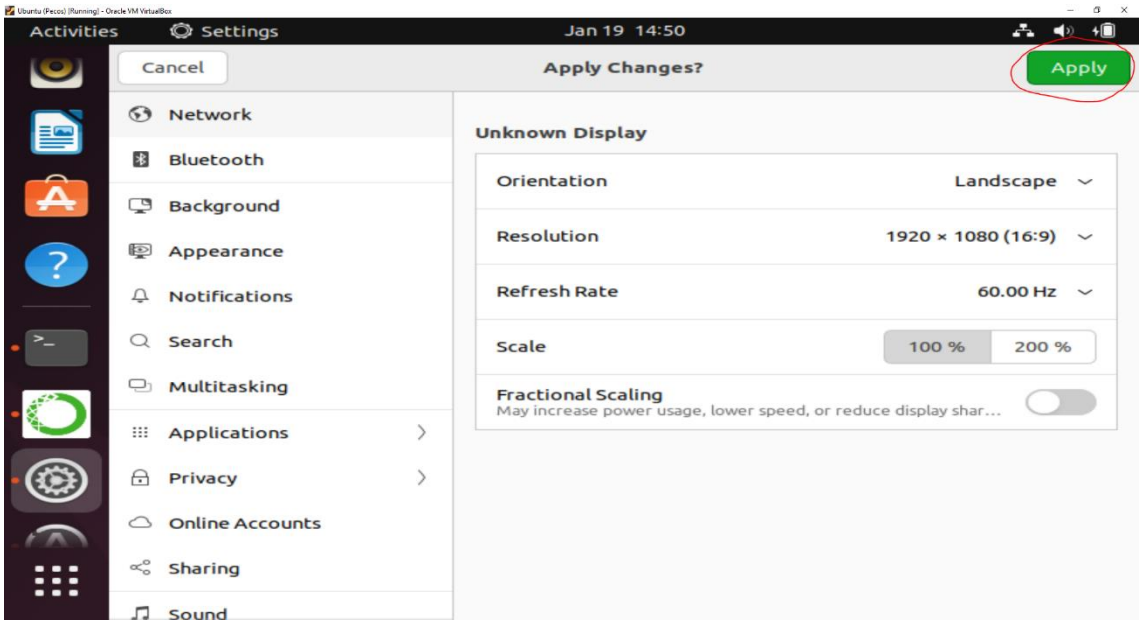
Select Display Settings:



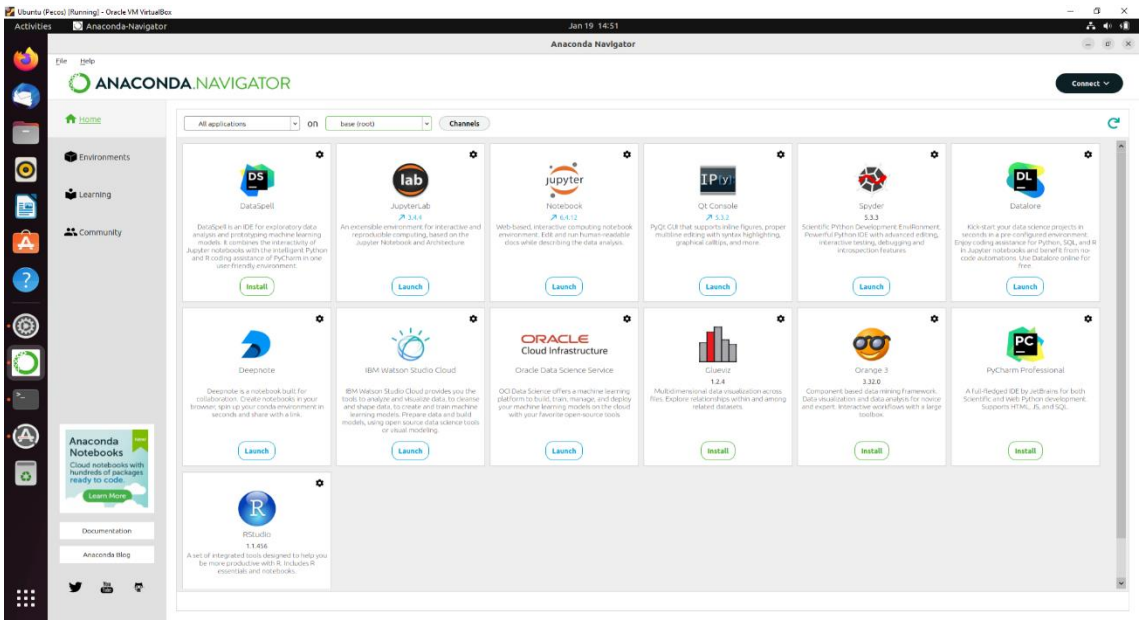
Select the Resolution of your monitor:



Hit Apply:



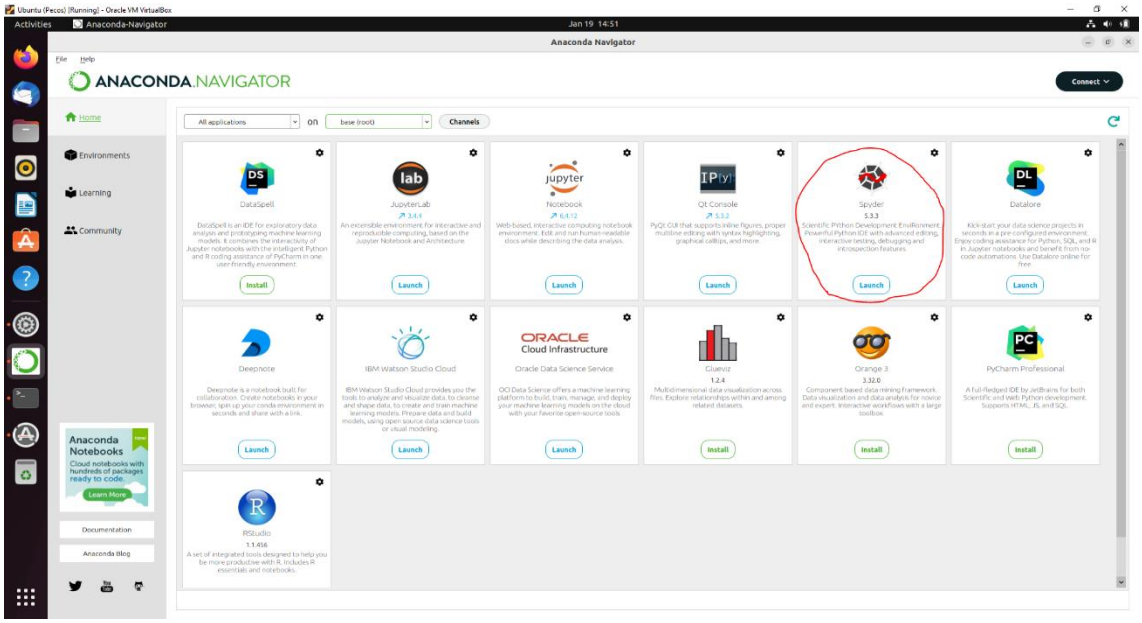
The resolution of the VM should now be significantly better and user-friendly:



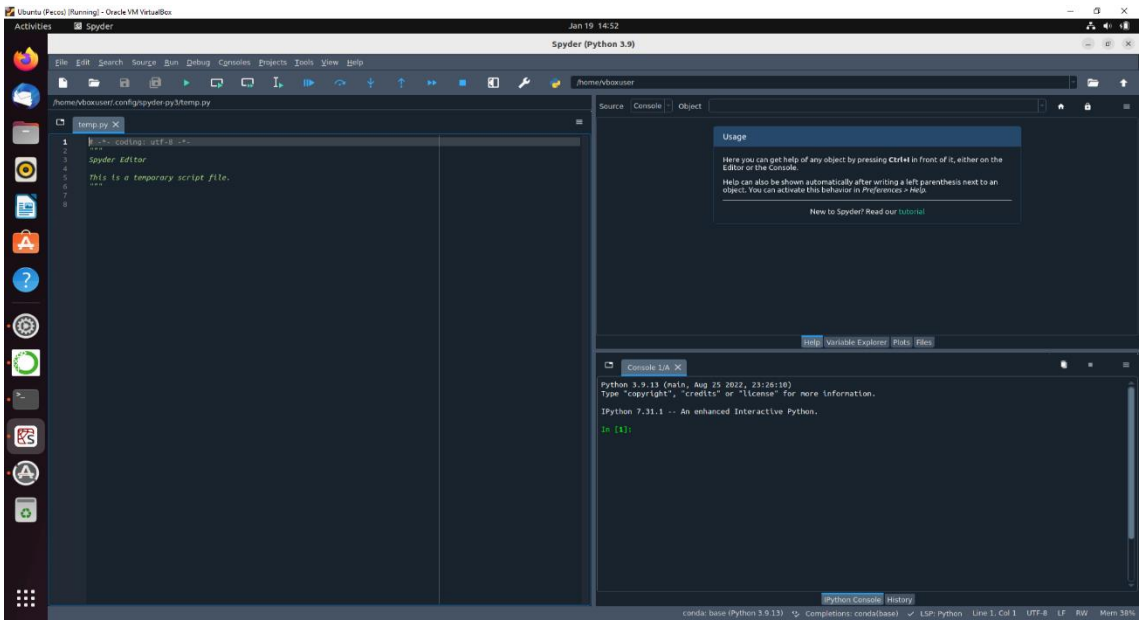
Step 10: Launch Spyder

Locate the Spyder application on the Anaconda Navigator Home page.

Click Launch:



You should now be seeing a window like the one below:

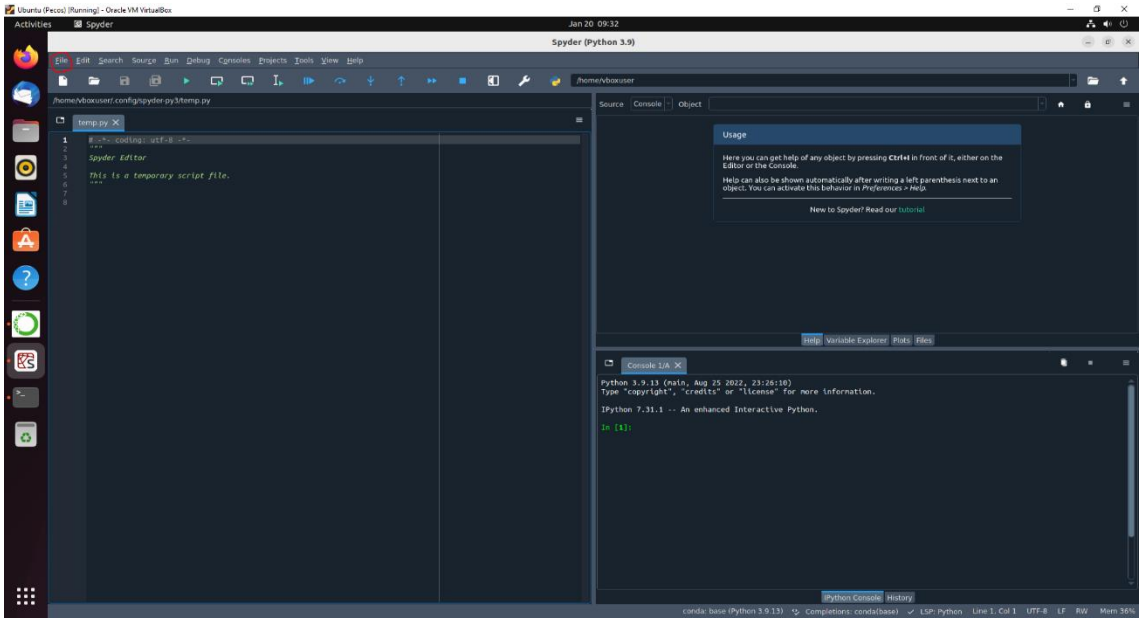


This tool provides an excellent platform for editing, testing, and running python scripts.

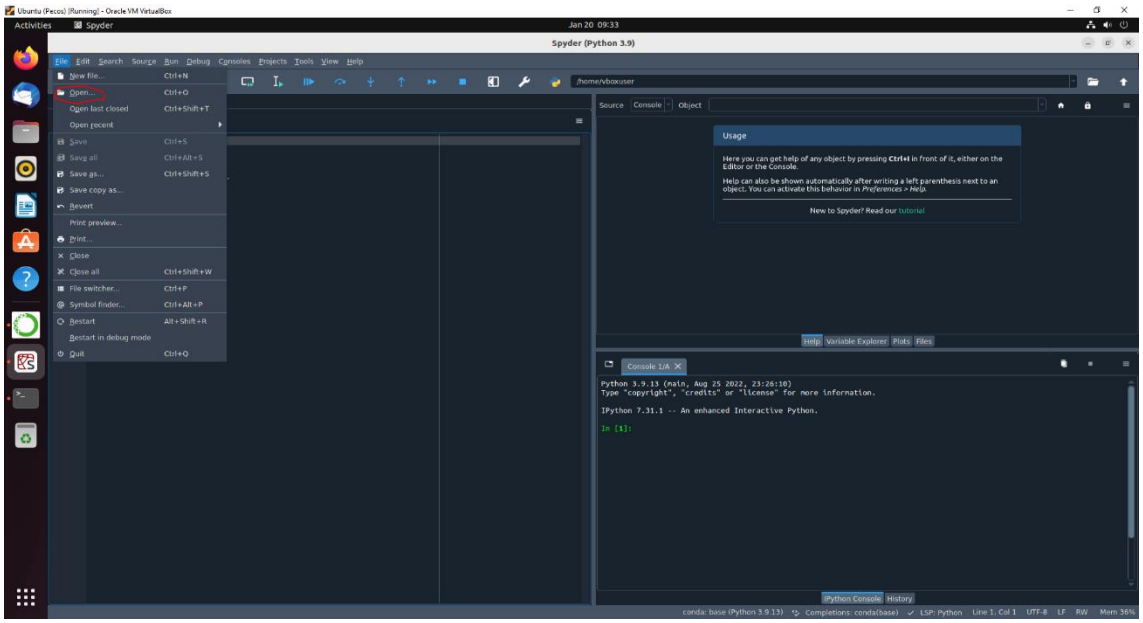
(Very similar to MATLAB)

Step 11: Opening Quality Control Script

In the Spyder window, locate the File table located to the top left and click on it:

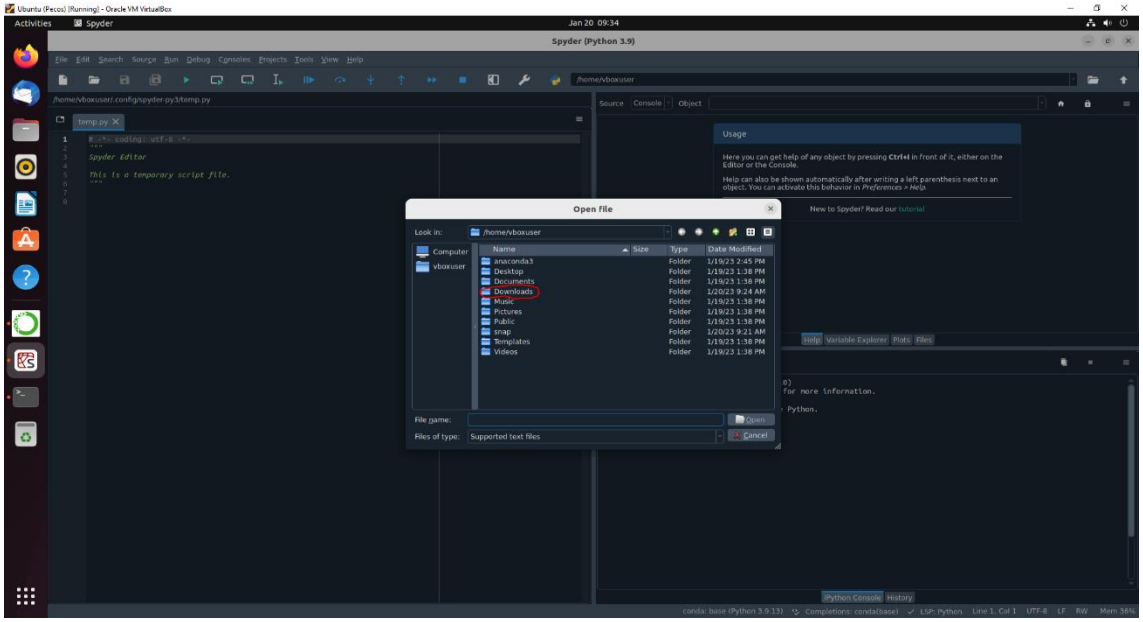


Now, select Open from the drop-down menu:



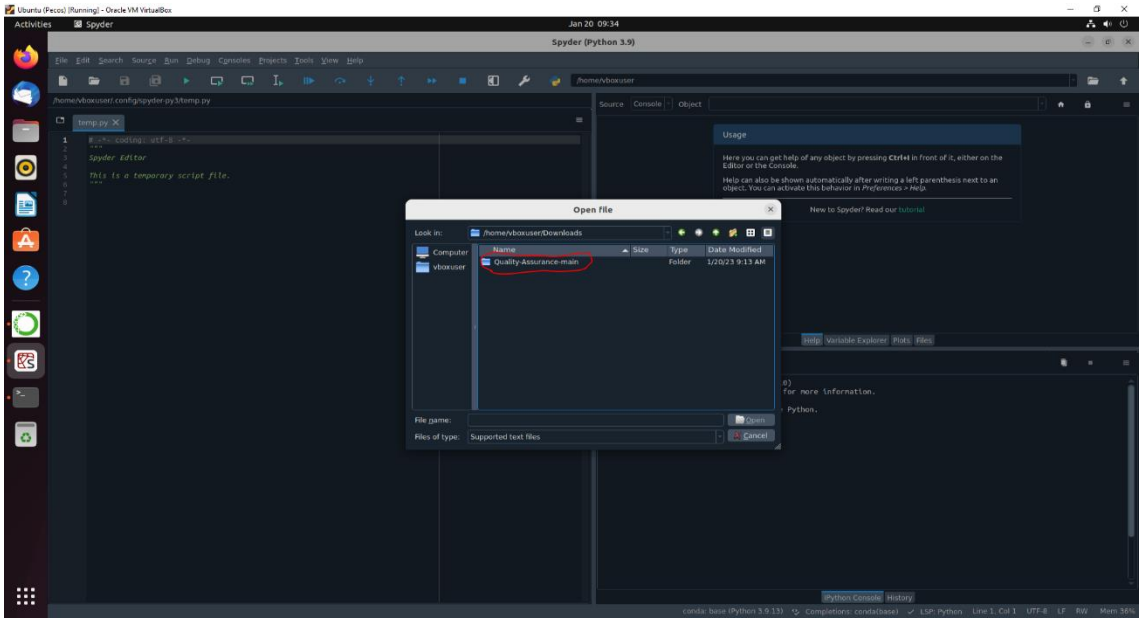
Now, locate the folder where your Pecos files are stored. (In our case, it is the Downloads folder.)

Double click the folder:

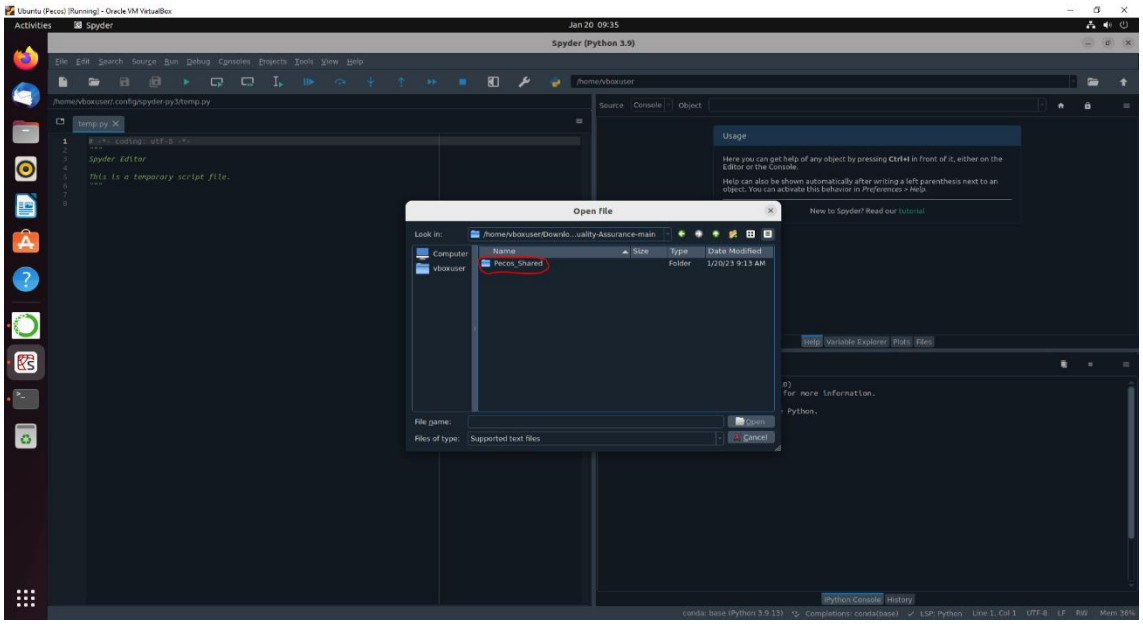


You should now see the downloaded folder from Step 7.

Double click it:

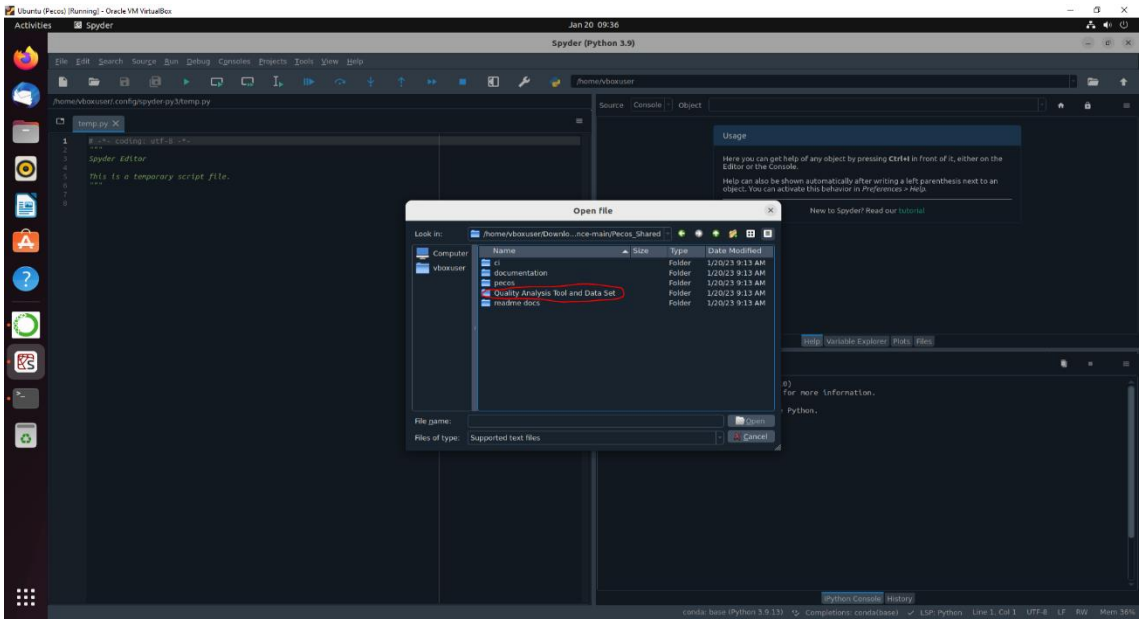


There is another nested folder. Go ahead and double click it as well:



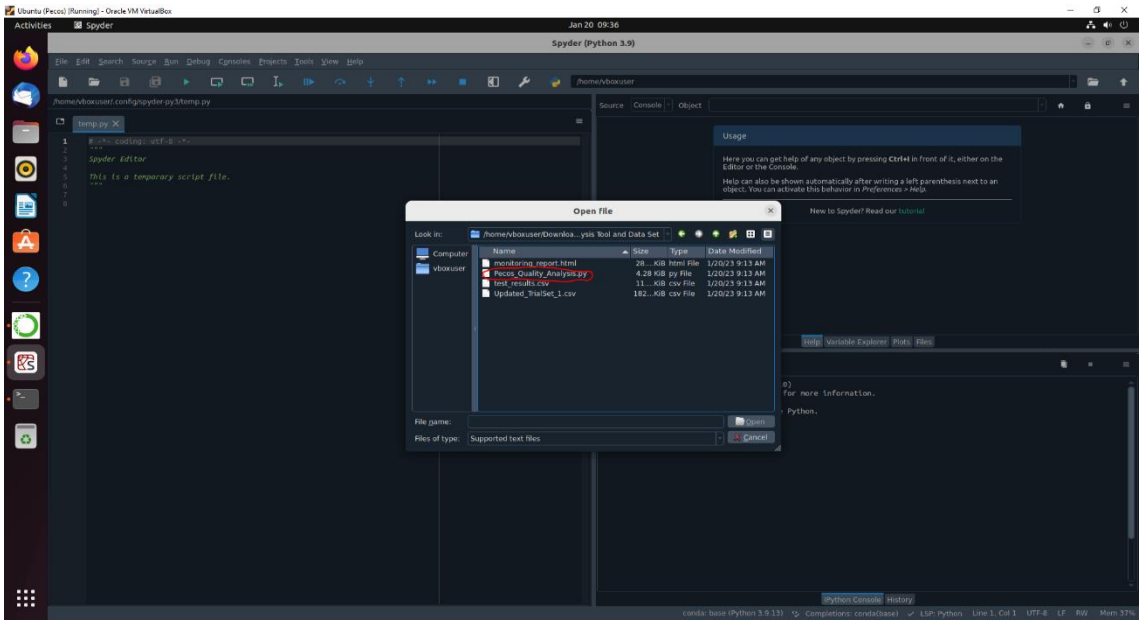
Now you should see multiple folders.

Locate the *Quality Analysis Tool and Data Set Folder and double click it:*



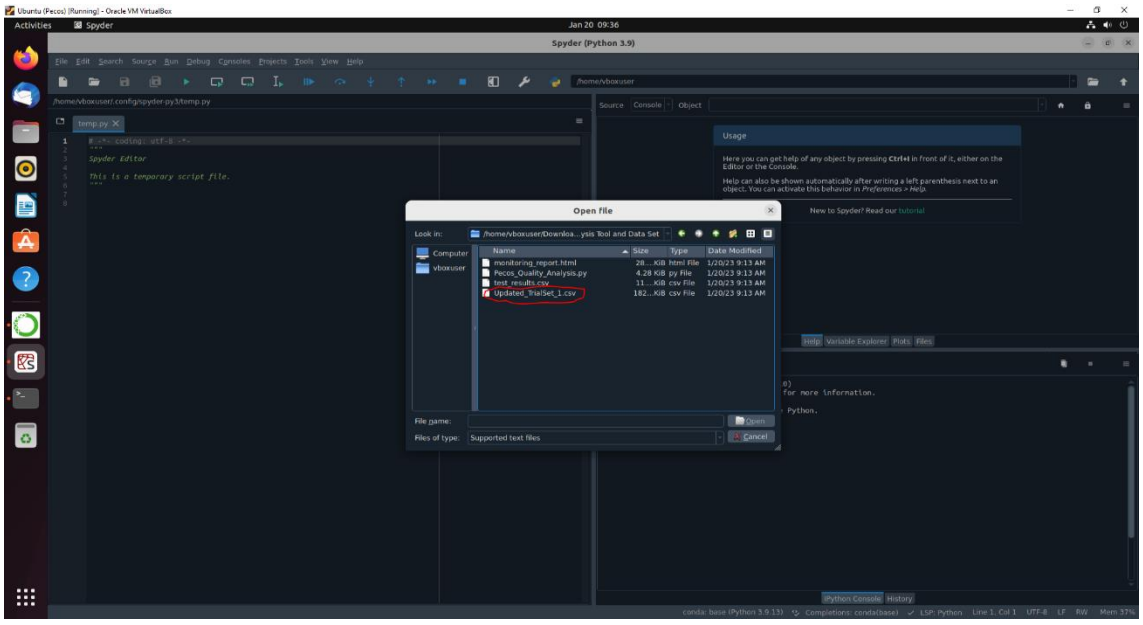
Now, Locate the *Pecos_Quality_Analysis.py* file.

This is the script that you need to run, so double click it:



This script can run a quality check on measured data when provided with a csv file containing the data of interest as well as user established parameters. (This is a simplified Pecos analysis that only focuses on quality control.)

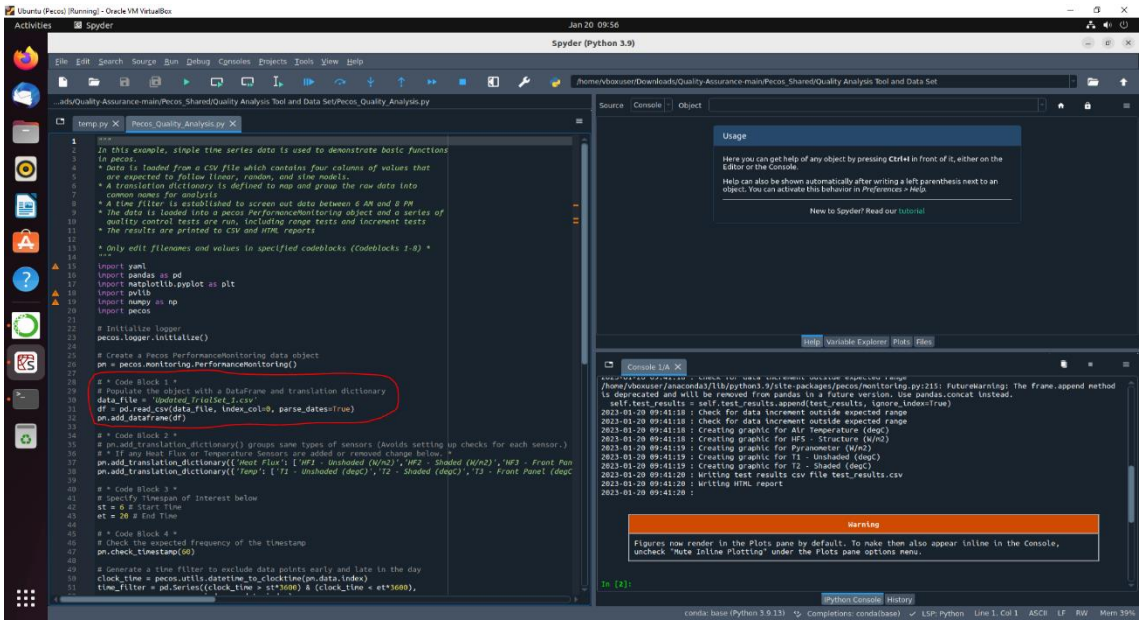
Also, in this folder, is the data file that will be analyzed:



Make sure before running the script that the data file that you are wanting to evaluate is in this folder

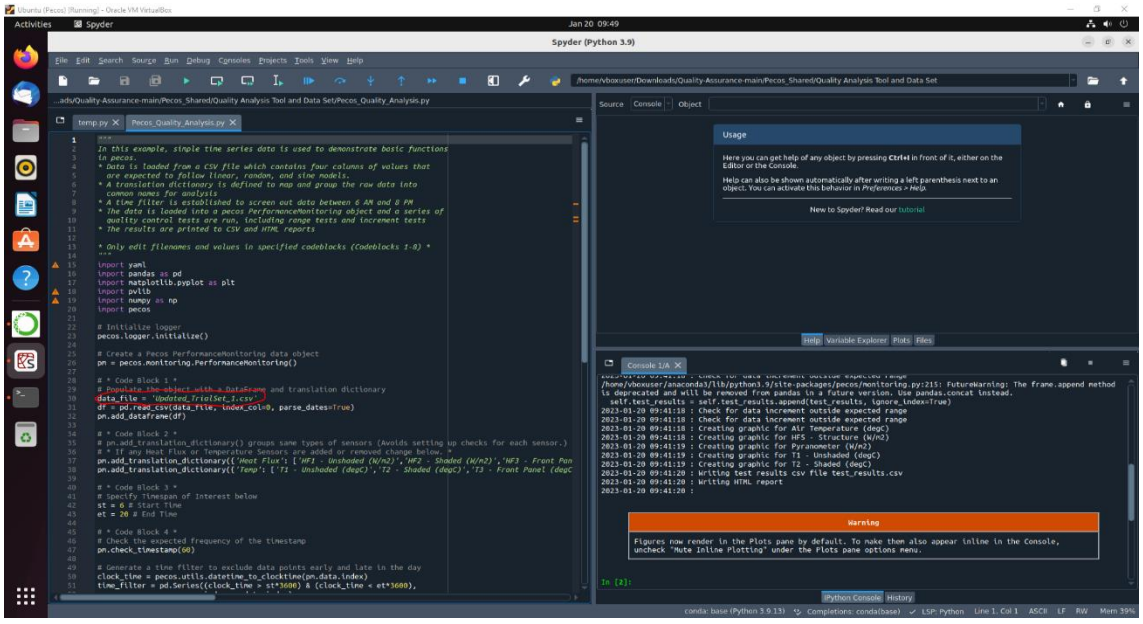
Step 12: Specify Data File of Interest

Locate Code Block 1:



Type file name inside of single quotes:

data_file = 'Filename.csv'



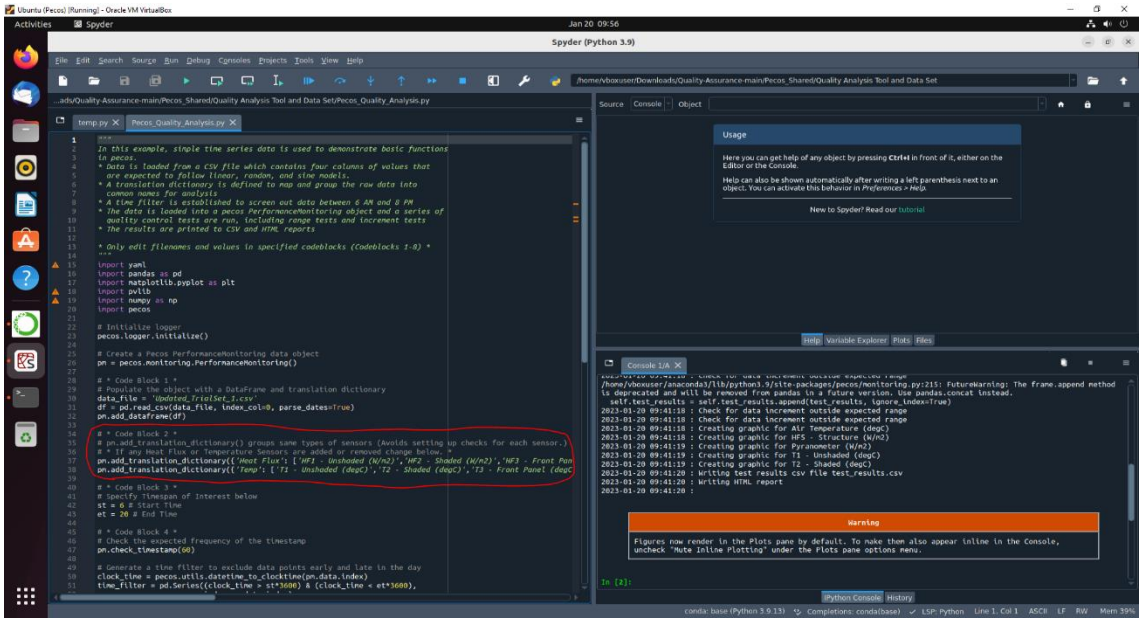
This command designates the file that will be evaluated. This file must be formatted as a csv file. Also, ensure that the file is saved in the same folder as the python script being run.

* Timestamp must be in form *month/day/year 24-hour:minutes* *

Example: *01/12/2023 14:28*

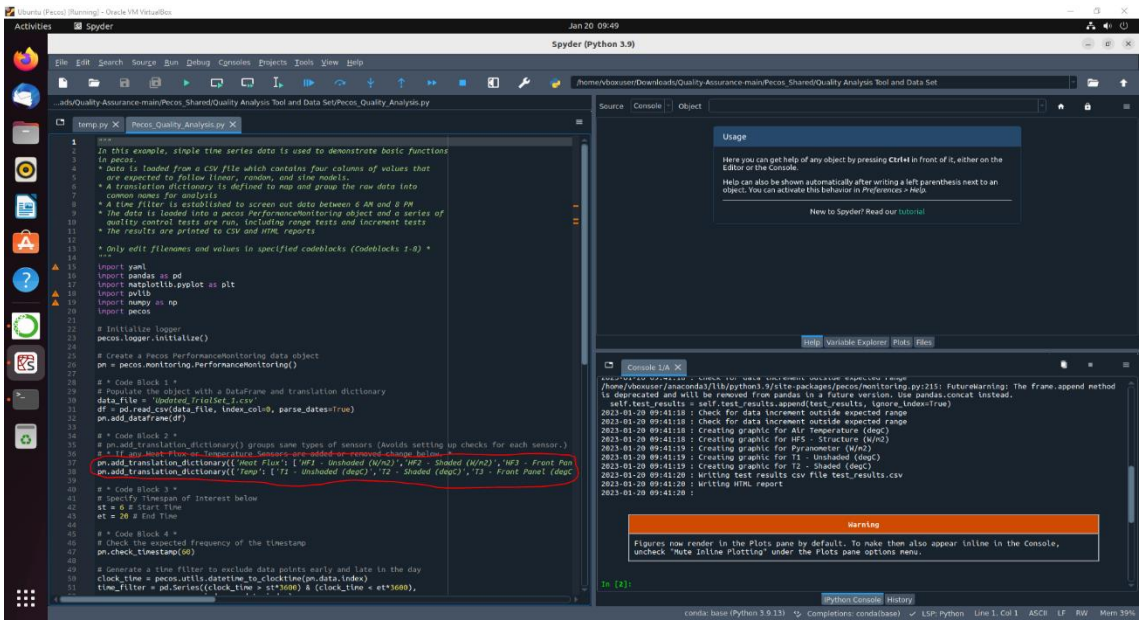
Step 13: Add and/or Remove Grouped Sensors

Locate Code Block 2:



Add and/or remove grouped sensors:

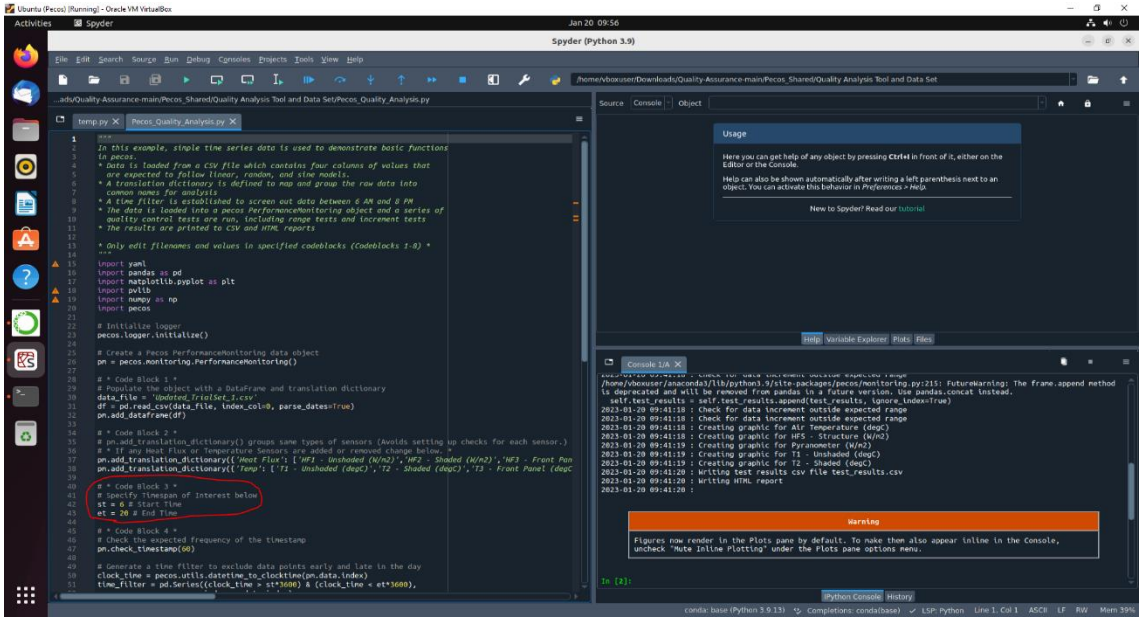
pm.add_translation_dictionary({'Groupname': ['Sensor1', 'Sensor2', 'etc']})



These lines of code group similar sensors together that can be evaluated using the same metrics. In our case, it groups all the heat flux sensors together under the name 'Heat Flux' and groups all the temperature sensors under the name 'Temp'.

Step 14: Change Hours of Interest

Locate Code Block 3:

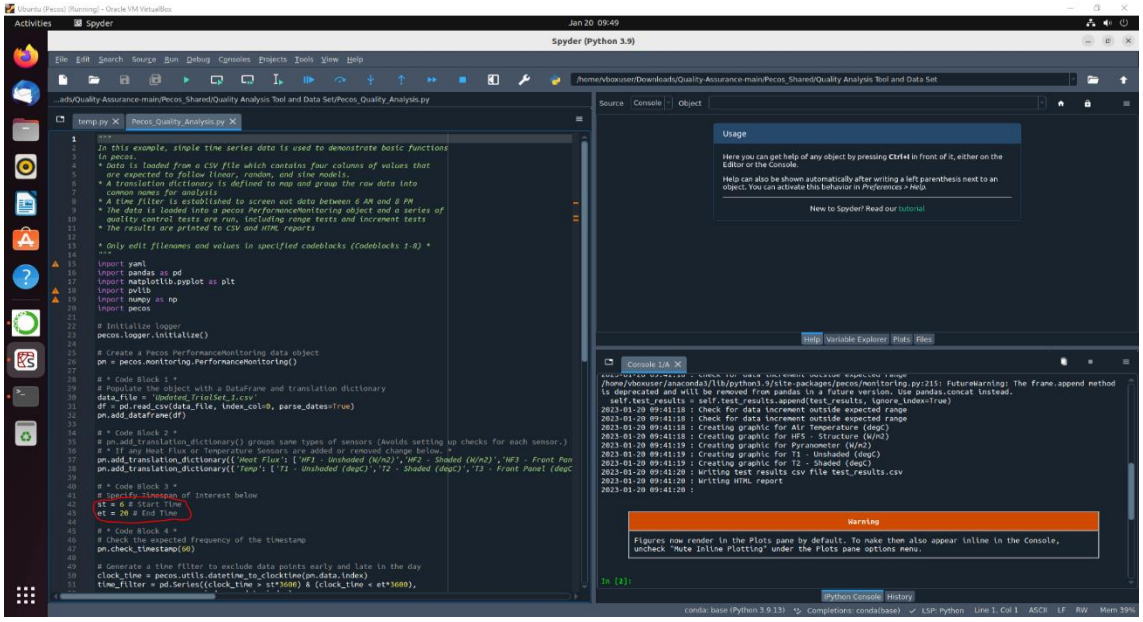


Designate hours of interest:

* Use 24-hour format*

st = a (Start Time)

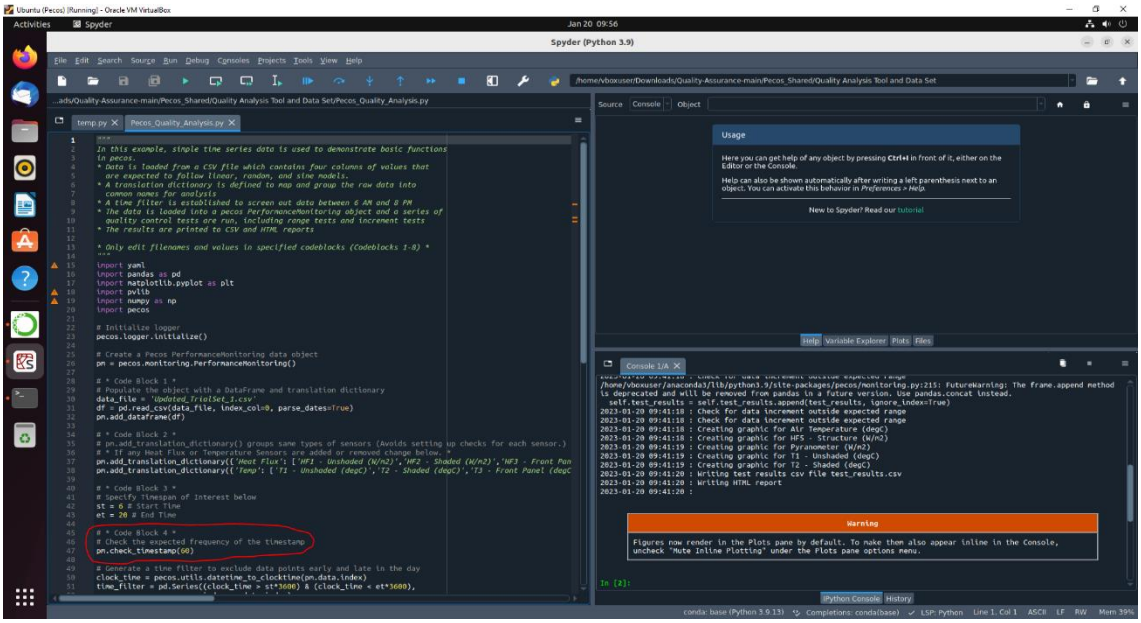
et = b (End Time)



The above establishes a timespan of interest. Other data will still be evaluated, but the report will only report flagged data within the timespan of interest.

Step 15: Specify Frequency of Timestamp

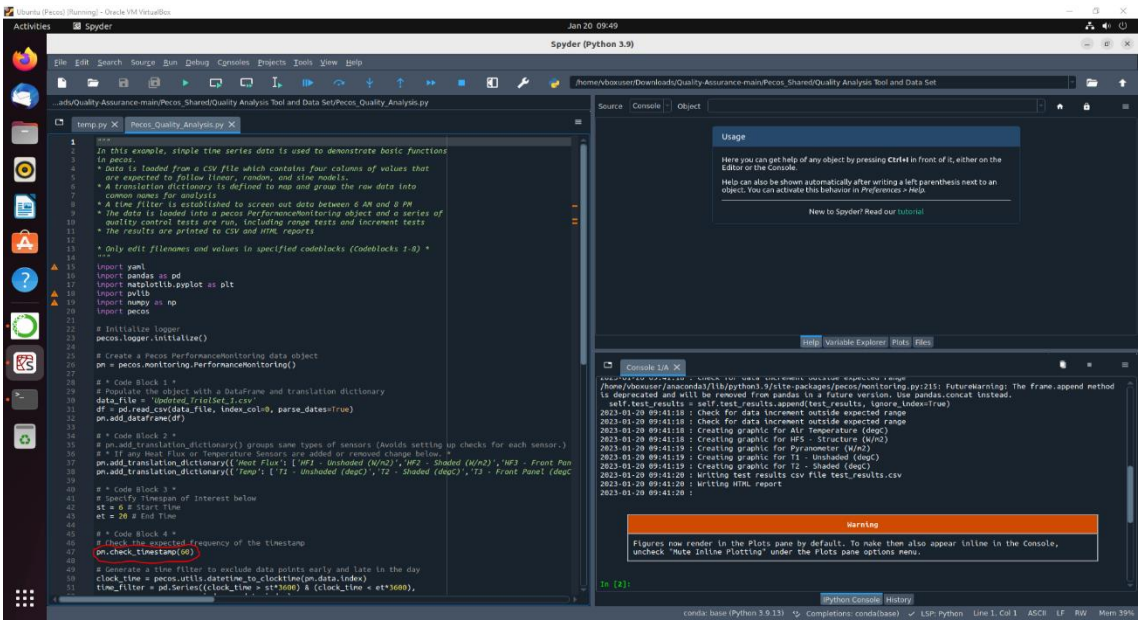
Locate Code Block 4:



Specify Timestamp Frequency:

pm.check_timestamp(t)

t: Time value (in seconds) of how often we are sampling data. * Report in seconds*



The above establishes the frequency of your measurements. For example, in our current setup, we are sampling once every 60 seconds. It is crucial that the value specified here and the frequency of the measurements in the data file are the same.

Step 16: (Filter 1) Change Expected Ranges for Sensors

Locate Code Block 5.

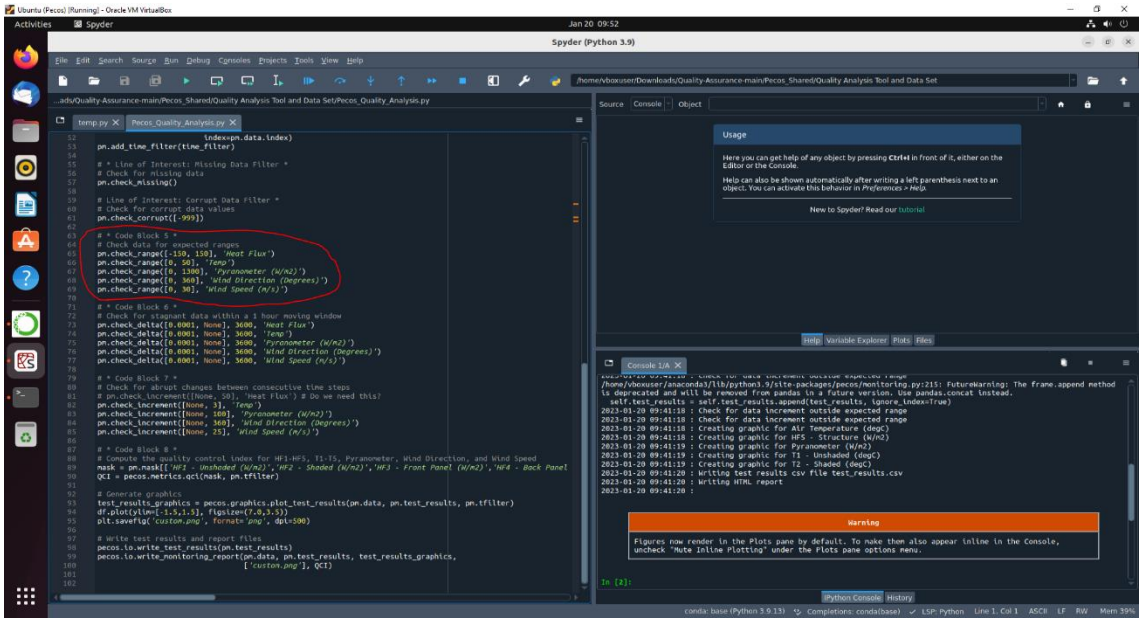
Change expected ranges for each sensor:

pm.check_range([a, b], 'Sensor1')

a: Lower bound of the expected range of the sensor.

b: Upper bound of the expected range of the sensor.

Ensure values are reported in the same units as in data file



The above establishes a reasonable range for the sensor indicated. If a value exceeds the specified range (Higher than the upper bound or lower than the lower bound), then it will be flagged and indicated in the report.

Step 17: (Filter 2) Check for Stagnant Data

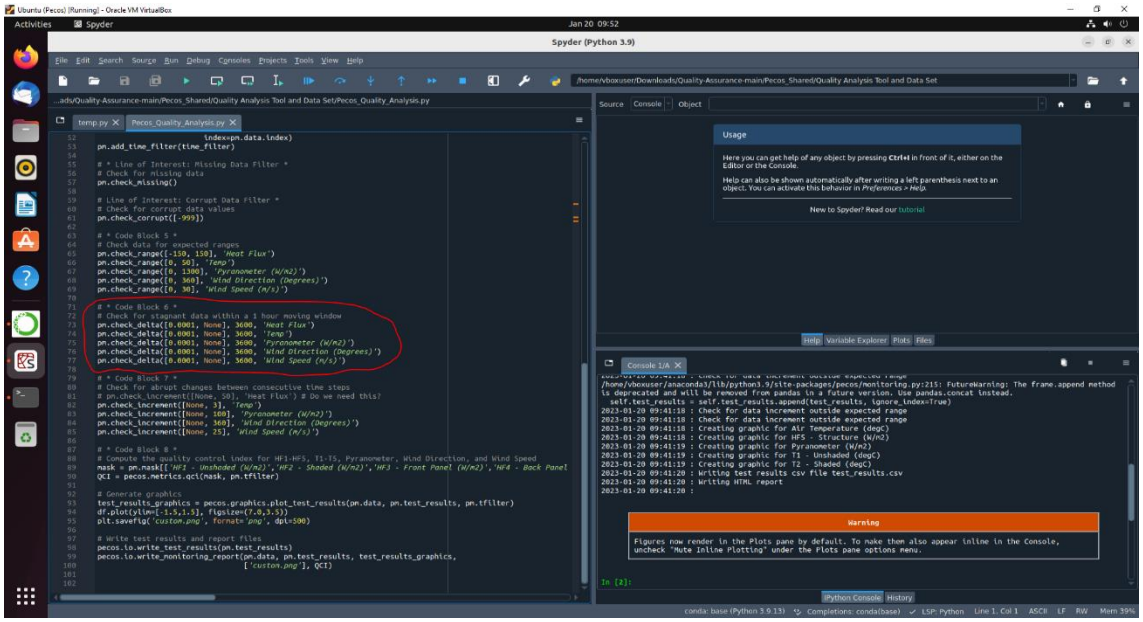
Locate Code Block 6.

Change stagnation check for each sensor:

pm.check_delta([a, None], b, 'Sensor1')

a: Small increment of change

b: Time interval (in seconds) that is being evaluated for stagnant data



The above runs a check for stagnant data. If the sensor does not output a value differing by 0.0001 for 1 hour (3600 s), then it will be flagged. This is useful, because it also indicates that we may be dealing with a faulty sensor.

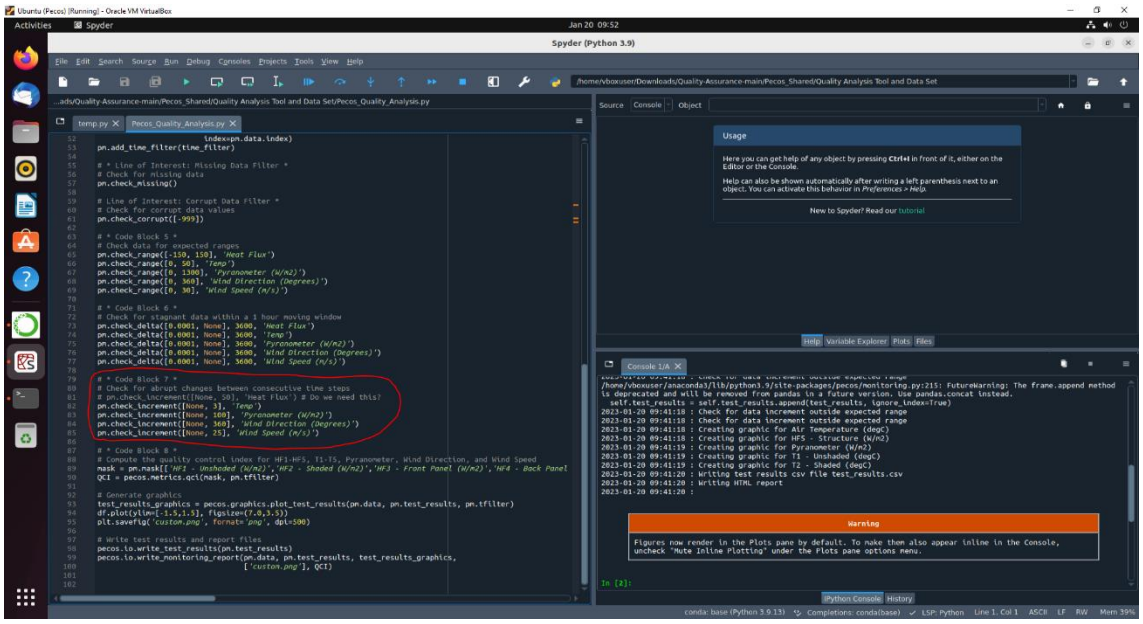
Step 18: (Filter 3) Change Abrupt Data Spike Parameters

Locate Code Block 7.

Change abrupt increment for each sensor:

pm.check_increment([None, a], ' Sensor1')

a: Largest reasonable change in value expected to occur from one timestamp to the next



The above runs a check to ensure that any immediate spikes in the data are flagged. This could be indicative of a spontaneous environmental event or a slight hiccup with either the sensor or the data logger.

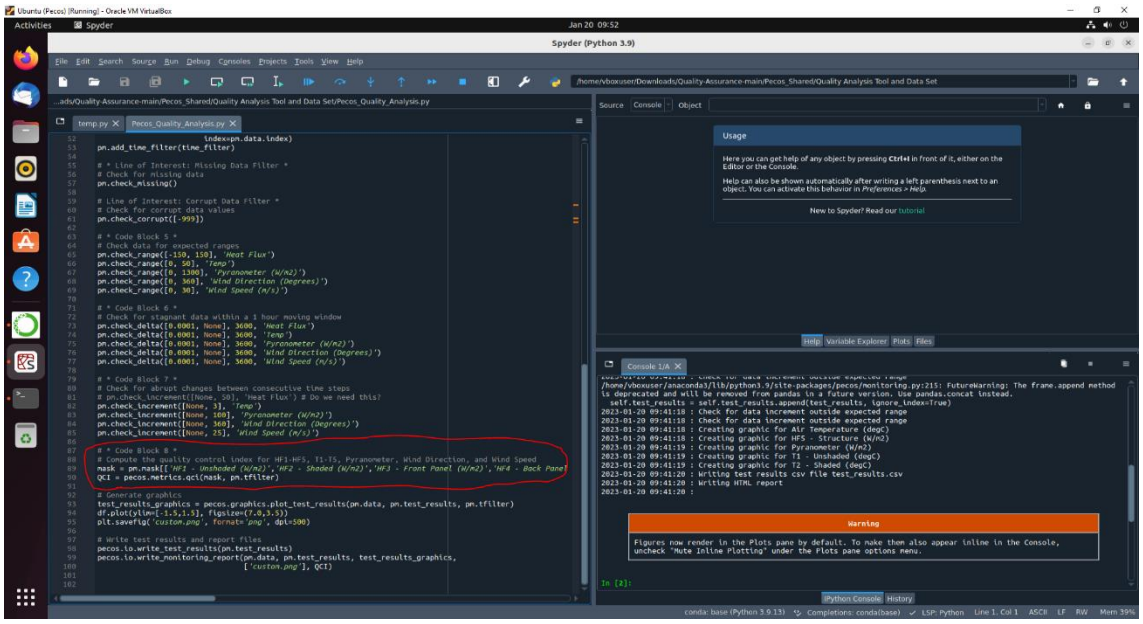
Step 19: Quality Control Index

Locate Code Block 8.

Ensure every sensor being evaluated is listed:

mask = pm.mask[[‘Sensor1’,’Sensor2’,’Sensor3’,’etc’]]

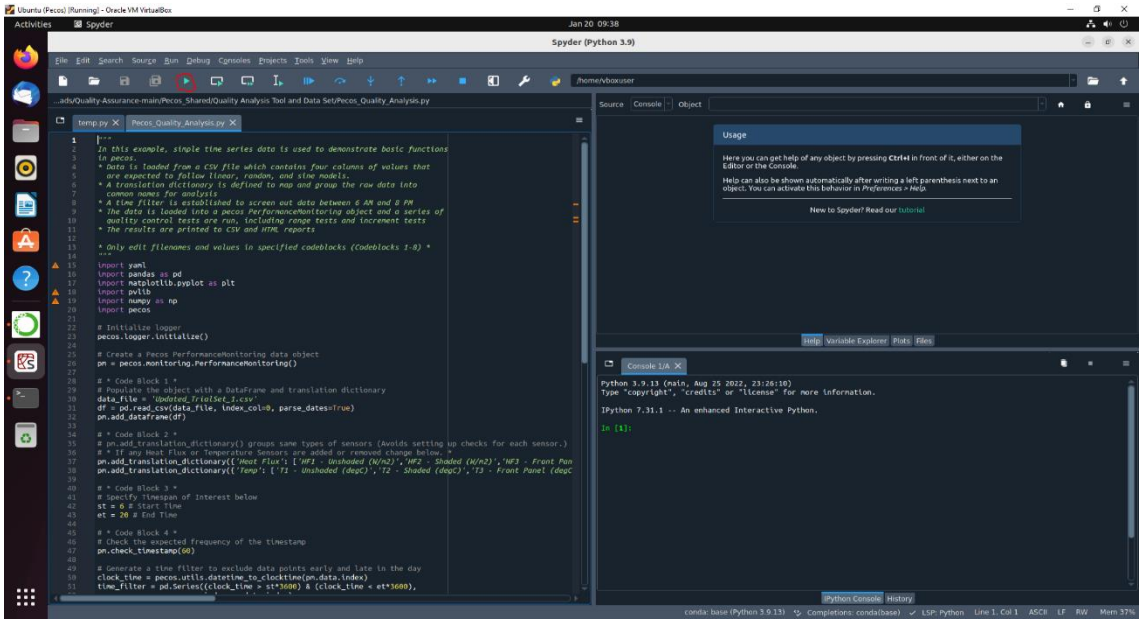
* All sensors must be listed individually; group names cannot be used. *



The above lines compute a quality control index for the sensors being evaluated. This serves as a good indicator of which sensors are performing well and which sensors are exhibiting persistent issues.

Step 20: Running Script

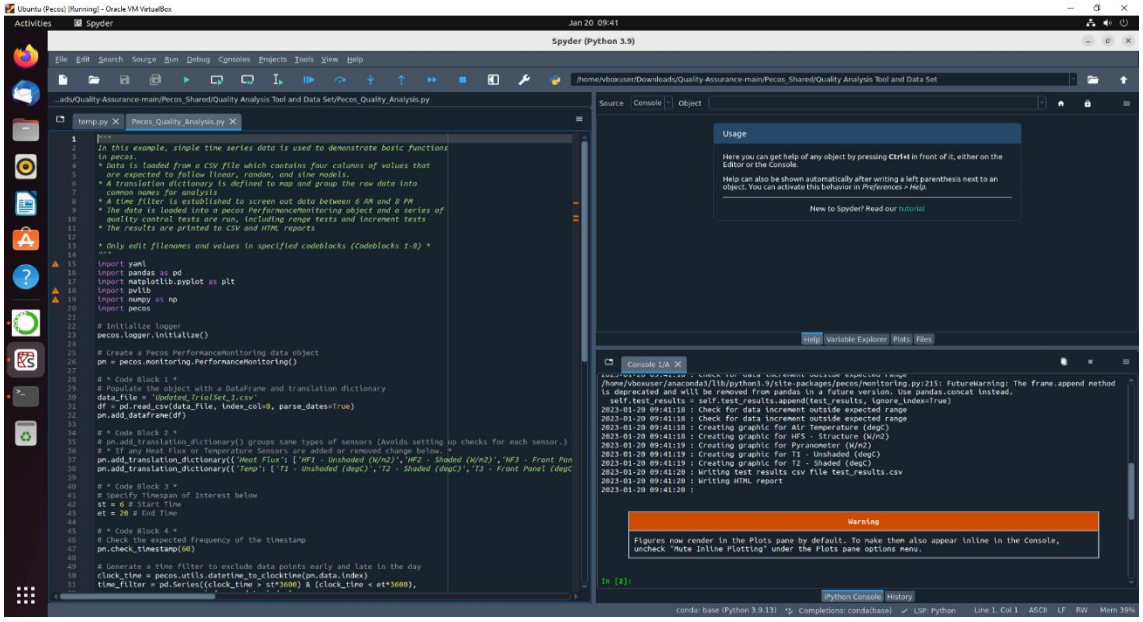
To run, click the green play button located above the script window.



Run with default configuration.

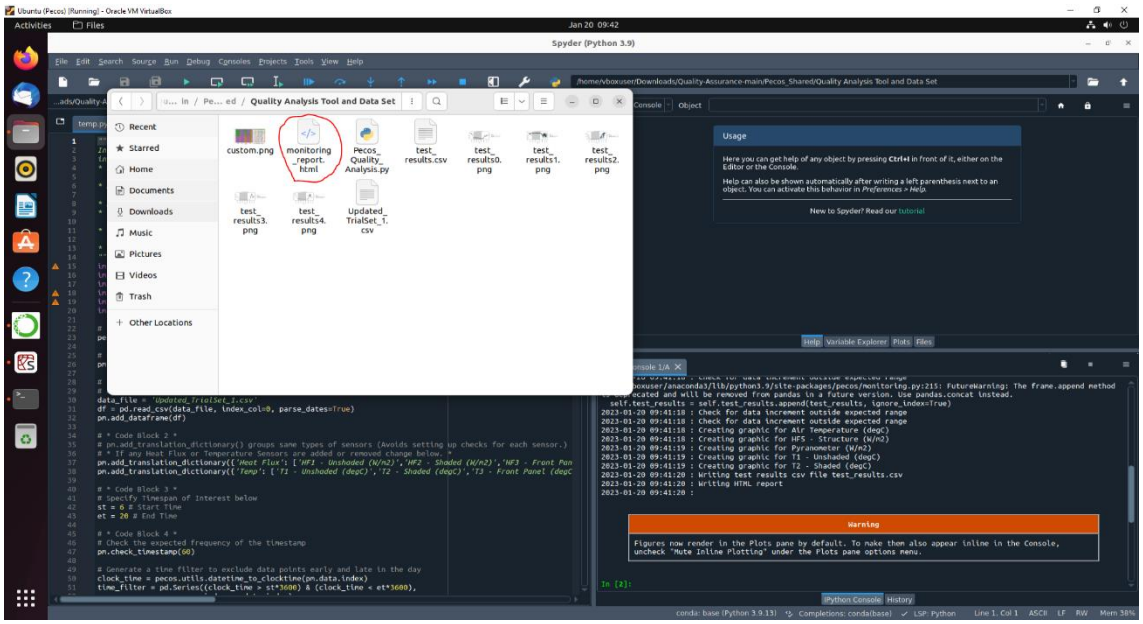
Looking at the bottom right window, we can see that the script has successfully run.

There are no errors, and the Report has been successfully generated:

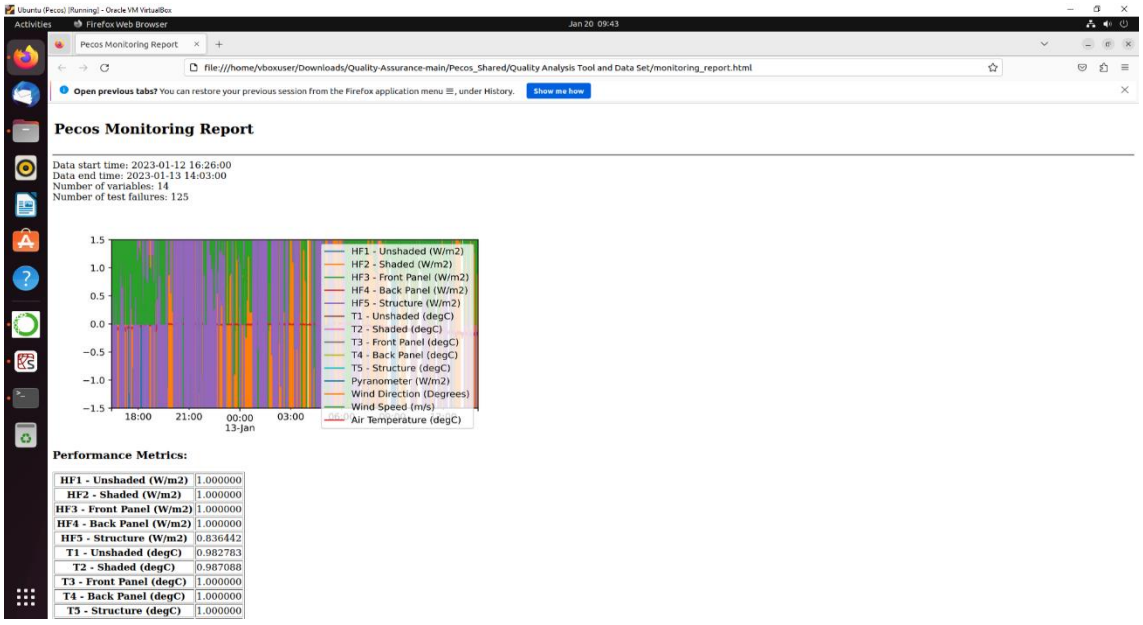


Step 21: Evaluate Report

Locate the document titled “monitoring_report.html”. (*This document should be in the same folder as the one that you are running the script from*):

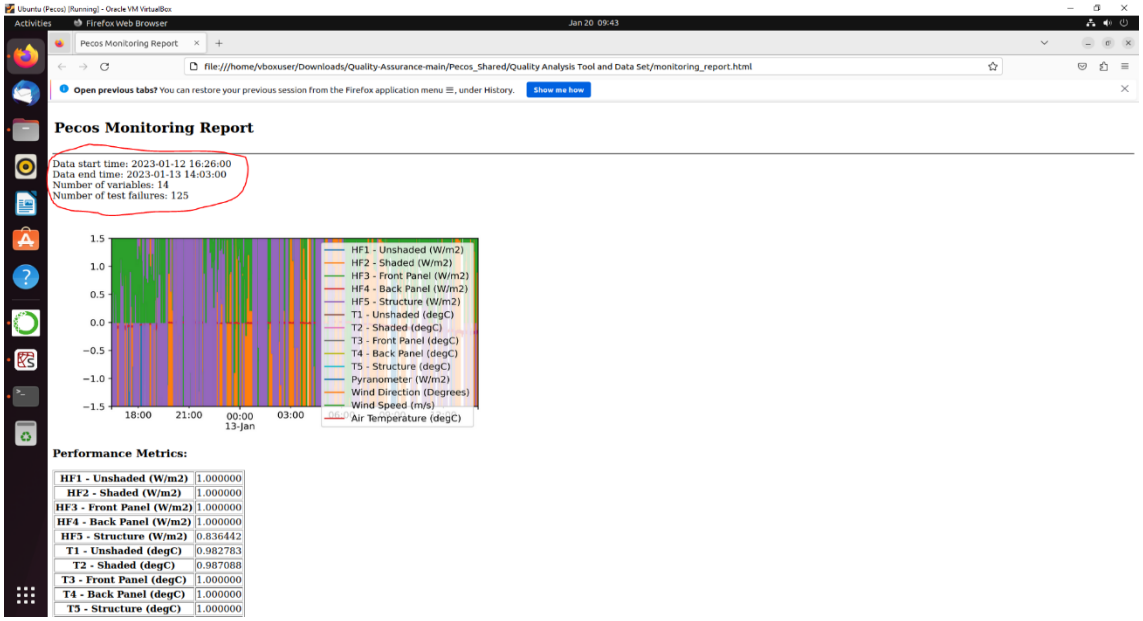


Once opened, you should see a report like the one below:

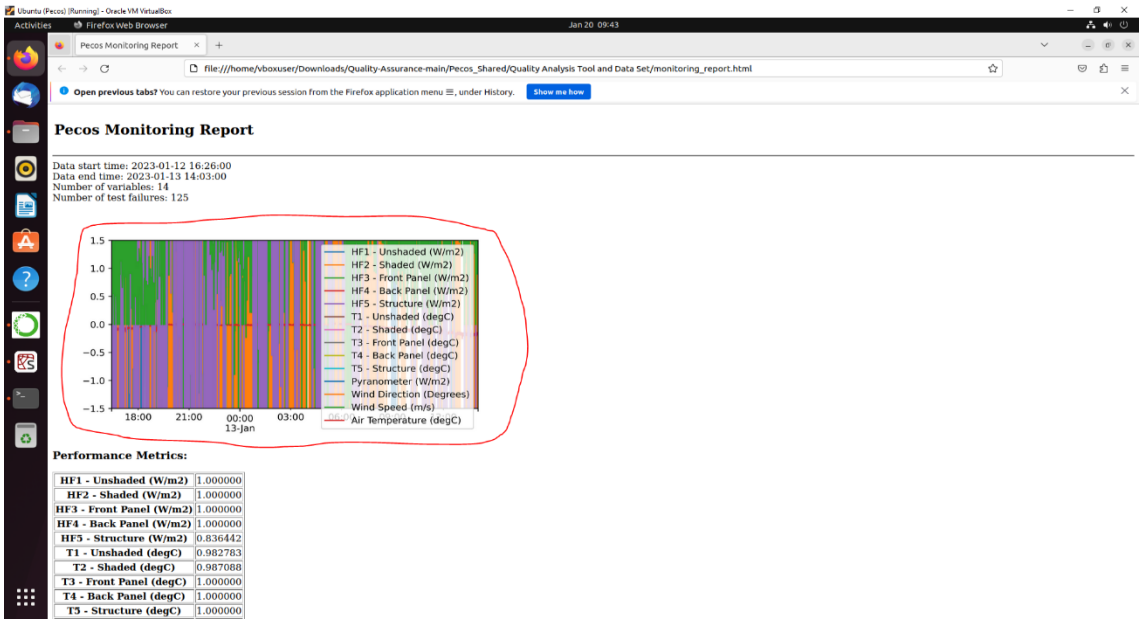


HF1 - Unshaded (W/m2)	1.000000
HF2 - Shaded (W/m2)	1.000000
HF3 - Front Panel (W/m2)	1.000000
HF4 - Back Panel (W/m2)	1.000000
HF5 - Structure (W/m2)	0.836442
T1 - Unshaded (degC)	0.982783
T2 - Shaded (degC)	0.987088
T3 - Front Panel (degC)	1.000000
T4 - Back Panel (degC)	1.000000
T5 - Structure (degC)	1.000000

At the top of the document, below the title, is a summary of the data analyzed. It lists the window of time that the data was recorded for, as well as the number of sensors being evaluated and the number of test failures that were experienced:

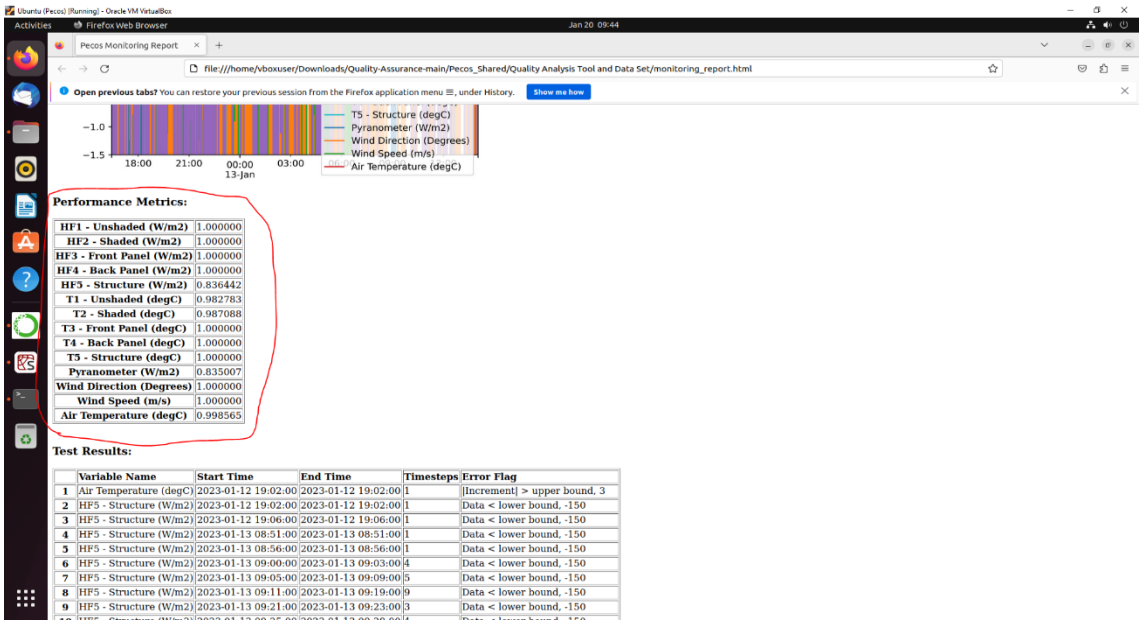


This graphic at the top depicts a comparison of data quality for each sensor:



The quality index of each sensor is a good indicator of if a sensor is beginning to fail.

A table of these performance metrics can be viewed immediately below the figure:



Performance Metrics:

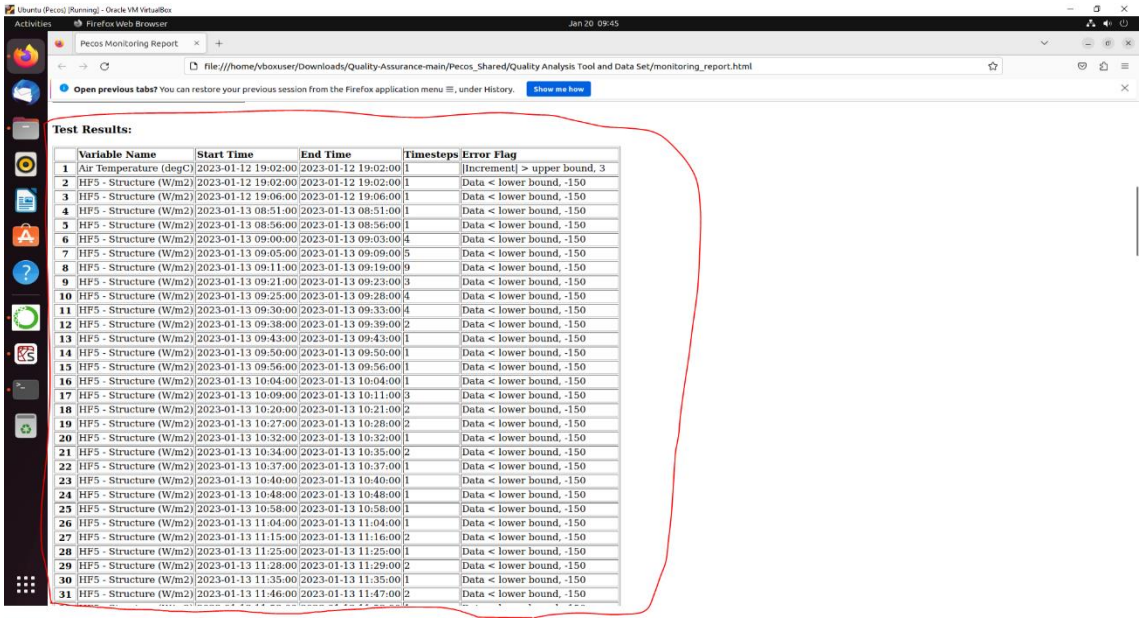
HF1 - Unshaded (W/m2)	1.000000
HF2 - Shaded (W/m2)	1.000000
HF3 - Front Panel (W/m2)	1.000000
HF4 - Back Panel (W/m2)	1.000000
HF5 - Structure (W/m2)	0.836442
T1 - Unshaded (degC)	0.982783
T2 - Shaded (degC)	0.987088
T3 - Front Panel (degC)	1.000000
T4 - Back Panel (degC)	1.000000
T5 - Structure (degC)	1.000000
Pyranometer (W/m2)	0.835007
Wind Direction (Degrees)	1.000000
Wind Speed (m/s)	1.000000
Air Temperature (degC)	0.998565

Test Results:

	Variable Name	Start Time	End Time	Timesteps	Error Flag
1	Air Temperature (degC)	2023-01-12 19:02:00	2023-01-12 19:02:00	1	[Increment] > upper bound, 3
2	HF5 - Structure (W/m2)	2023-01-12 19:02:00	2023-01-12 19:02:00	1	Data < lower bound, -150
3	HF5 - Structure (W/m2)	2023-01-12 19:06:00	2023-01-12 19:06:00	1	Data < lower bound, -150
4	HF5 - Structure (W/m2)	2023-01-13 08:51:00	2023-01-13 08:51:00	1	Data < lower bound, -150
5	HF5 - Structure (W/m2)	2023-01-13 08:56:00	2023-01-13 08:56:00	1	Data < lower bound, -150
6	HF5 - Structure (W/m2)	2023-01-13 09:00:00	2023-01-13 09:03:00	4	Data < lower bound, -150
7	HF5 - Structure (W/m2)	2023-01-13 09:05:00	2023-01-13 09:09:00	5	Data < lower bound, -150
8	HF5 - Structure (W/m2)	2023-01-13 09:11:00	2023-01-13 09:19:00	9	Data < lower bound, -150
9	HF5 - Structure (W/m2)	2023-01-13 09:21:00	2023-01-13 09:23:00	3	Data < lower bound, -150

The closer to 1 that a sensor scores, the less issues it experienced. Therefore, if a sensor scores high, then it is in good operating condition. However, if it scores low, then it will need to be checked immediately.

Under “Test Results” every flagged data point is listed along with which test(s) it failed:

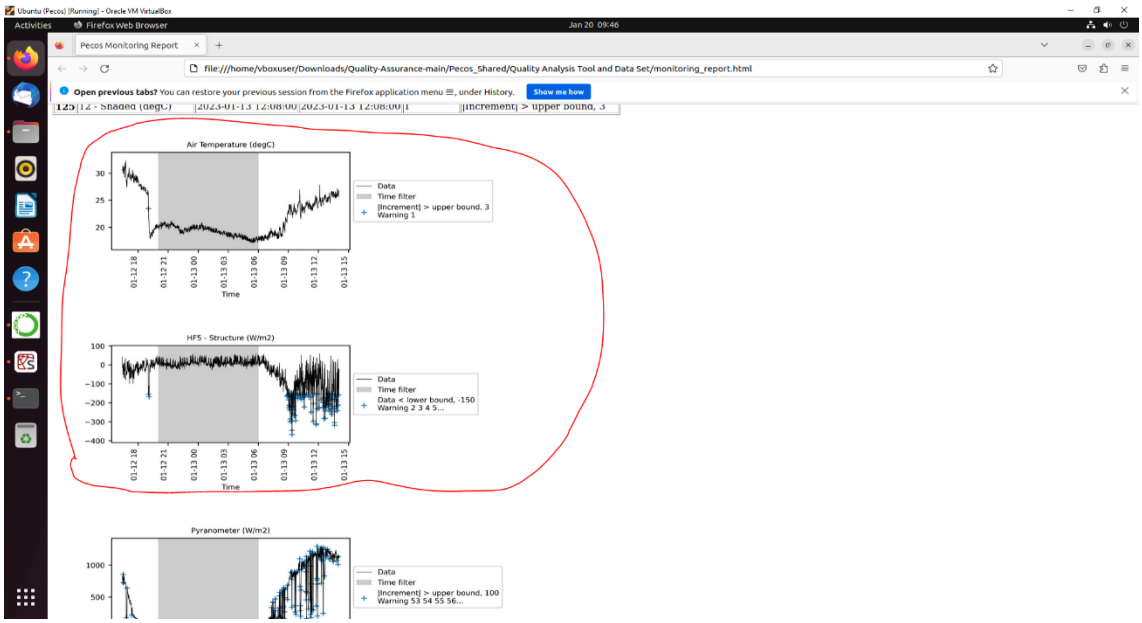


Test Results:

	Variable Name	Start Time	End Time	Timesteps	Error Flag
1	Air Temperature (degC)	2023-01-12 19:02:00	2023-01-12 19:02:00	1	[Increment] > upper bound, 3
2	HF5 - Structure (W/m2)	2023-01-12 19:02:00	2023-01-12 19:02:00	1	Data < lower bound, -150
3	HF5 - Structure (W/m2)	2023-01-12 19:06:00	2023-01-12 19:06:00	1	Data < lower bound, -150
4	HF5 - Structure (W/m2)	2023-01-13 08:51:00	2023-01-13 08:51:00	1	Data < lower bound, -150
5	HF5 - Structure (W/m2)	2023-01-13 08:56:00	2023-01-13 08:56:00	1	Data < lower bound, -150
6	HF5 - Structure (W/m2)	2023-01-13 09:00:00	2023-01-13 09:03:00	4	Data < lower bound, -150
7	HF5 - Structure (W/m2)	2023-01-13 09:05:00	2023-01-13 09:09:00	5	Data < lower bound, -150
8	HF5 - Structure (W/m2)	2023-01-13 09:11:00	2023-01-13 09:19:00	9	Data < lower bound, -150
9	HF5 - Structure (W/m2)	2023-01-13 09:21:00	2023-01-13 09:23:00	3	Data < lower bound, -150
10	HF5 - Structure (W/m2)	2023-01-13 09:25:00	2023-01-13 09:28:00	4	Data < lower bound, -150
11	HF5 - Structure (W/m2)	2023-01-13 09:30:00	2023-01-13 09:33:00	4	Data < lower bound, -150
12	HF5 - Structure (W/m2)	2023-01-13 09:38:00	2023-01-13 09:39:00	2	Data < lower bound, -150
13	HF5 - Structure (W/m2)	2023-01-13 09:43:00	2023-01-13 09:43:00	1	Data < lower bound, -150
14	HF5 - Structure (W/m2)	2023-01-13 09:50:00	2023-01-13 09:50:00	1	Data < lower bound, -150
15	HF5 - Structure (W/m2)	2023-01-13 09:56:00	2023-01-13 09:56:00	1	Data < lower bound, -150
16	HF5 - Structure (W/m2)	2023-01-13 10:04:00	2023-01-13 10:04:00	1	Data < lower bound, -150
17	HF5 - Structure (W/m2)	2023-01-13 10:09:00	2023-01-13 10:11:00	3	Data < lower bound, -150
18	HF5 - Structure (W/m2)	2023-01-13 10:20:00	2023-01-13 10:21:00	2	Data < lower bound, -150
19	HF5 - Structure (W/m2)	2023-01-13 10:27:00	2023-01-13 10:28:00	2	Data < lower bound, -150
20	HF5 - Structure (W/m2)	2023-01-13 10:32:00	2023-01-13 10:32:00	1	Data < lower bound, -150
21	HF5 - Structure (W/m2)	2023-01-13 10:34:00	2023-01-13 10:35:00	2	Data < lower bound, -150
22	HF5 - Structure (W/m2)	2023-01-13 10:37:00	2023-01-13 10:37:00	1	Data < lower bound, -150
23	HF5 - Structure (W/m2)	2023-01-13 10:40:00	2023-01-13 10:40:00	1	Data < lower bound, -150
24	HF5 - Structure (W/m2)	2023-01-13 10:48:00	2023-01-13 10:48:00	1	Data < lower bound, -150
25	HF5 - Structure (W/m2)	2023-01-13 10:58:00	2023-01-13 10:58:00	1	Data < lower bound, -150
26	HF5 - Structure (W/m2)	2023-01-13 11:04:00	2023-01-13 11:04:00	1	Data < lower bound, -150
27	HF5 - Structure (W/m2)	2023-01-13 11:15:00	2023-01-13 11:16:00	2	Data < lower bound, -150
28	HF5 - Structure (W/m2)	2023-01-13 11:25:00	2023-01-13 11:25:00	1	Data < lower bound, -150
29	HF5 - Structure (W/m2)	2023-01-13 11:28:00	2023-01-13 11:29:00	2	Data < lower bound, -150
30	HF5 - Structure (W/m2)	2023-01-13 11:35:00	2023-01-13 11:35:00	1	Data < lower bound, -150
31	HF5 - Structure (W/m2)	2023-01-13 11:46:00	2023-01-13 11:47:00	2	Data < lower bound, -150

This allows the user to immediately locate at which timestamp the error occurred.

Immediately below the table of test results is a set of graphs for each sensor that was flagged during the analysis:



This provides for an easy visual analysis of the data, where the user can readily identify any gaps and/or spontaneous spikes in the data.

Step 22: Address any Issues (Maintenance)

- Why did we get these values?
  - Faulty Sensor
    - Replace Sensor
  - Faulty Datalogger
  - Bad Connection
    - Loose Wire
    - Compromised Connection
  - Irregular Conditions
    - Weather
    - Temporary Shading
    - Animal Tampering
    - Other Environmental Factors

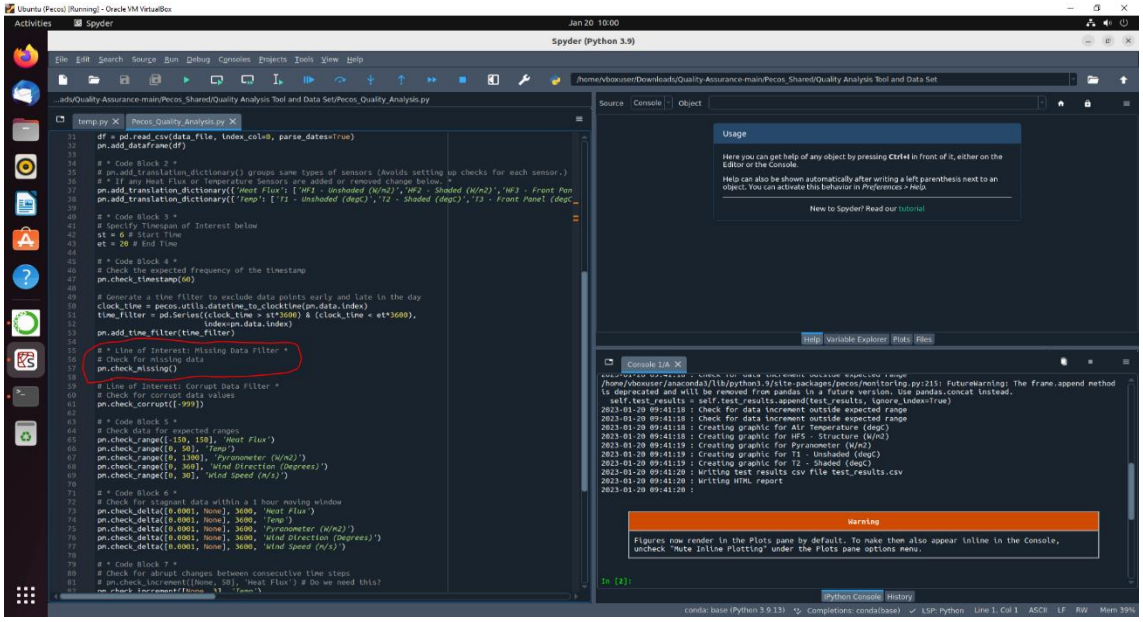
Regularly scheduled maintenance is crucial to the success of a system such as this. Pecos serves as a wonderful indicator of issues within the system, but the user is responsible for maintaining the normal operational conditions of the system.

Areas for Improvement (Literature Suggestions)

- Automatic data retrieval
- Multiple file analysis and report generation

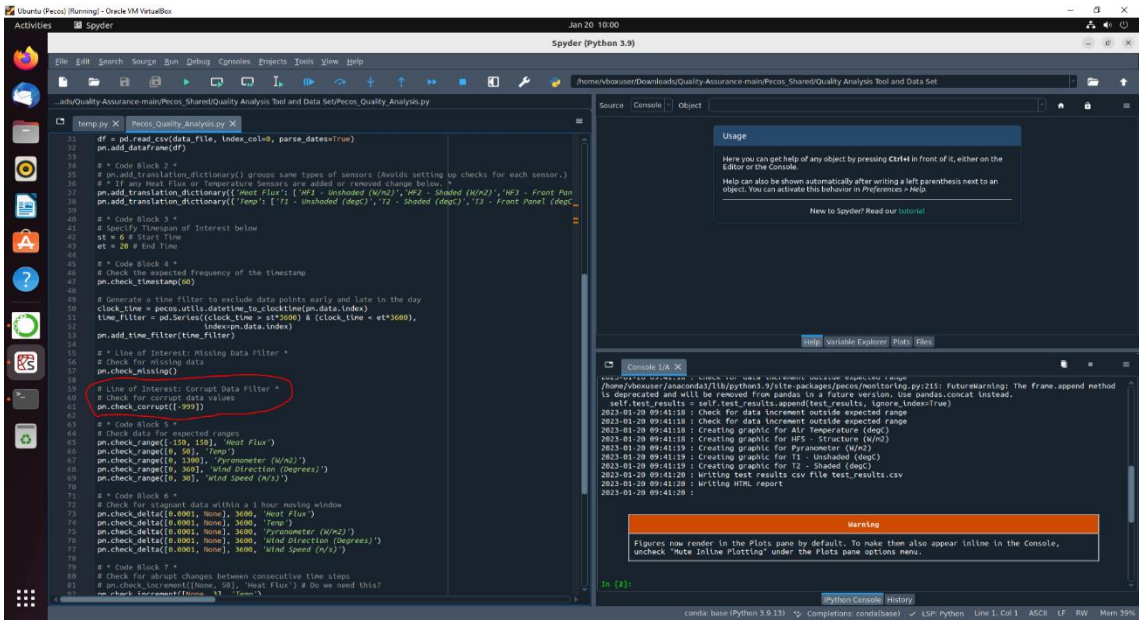
Other Lines of Interest in Code

pm.check_missing()



The line above was not included in the guide above due to it never needing to be changed. It acts as an additional filter by flagging any occurrences of missing data. This is extremely helpful because it can be indicative of a failing sensor or a fault in the system that needs to be addressed.

pm.check_corrupt([-999])



The above line of code runs as a check for any corrupt data values. This is also a great indicator of a failing sensor.

References

K.A. Klise and J.S. Stein (2016), Performance Monitoring using Pecos, Technical Report SAND2016-3583, Sandia National Laboratories, Albuquerque, NM.

K.A. Klise and J.S. Stein (2016), Automated Performance Monitoring for PV Systems using Pecos, 43rd IEEE Photovoltaic Specialists Conference (PVSC), Portland, OR, June 5-10. pdf

Latest Link: https://pecos.readthedocs.io/en/latest/

Sturtevant, C., Metzger, S., Nehr, S., Foken, T. (2021). Quality Assurance and Control. In: Foken, T. (eds) Springer Handbook of Atmospheric Measurements. Springer Handbooks. Springer, Cham. https://doi.org/10.1007/978-3-030-52171-4_3