

tesr_ros_six_servo_robot_pkg

&

tesr_ros_six_servo_arm_moveit_config

[Install Ubuntu using VMWare player](#)

[Installation of ROS Noetic on Ubuntu](#)

[Installation of moveit](#)

[How to launch tesr_ros_six_servo_robot_pkg](#)

[How to launch tesr_ros_six_servo_arm_moveit_config](#)



Ubuntu and ROS Noetic Installation

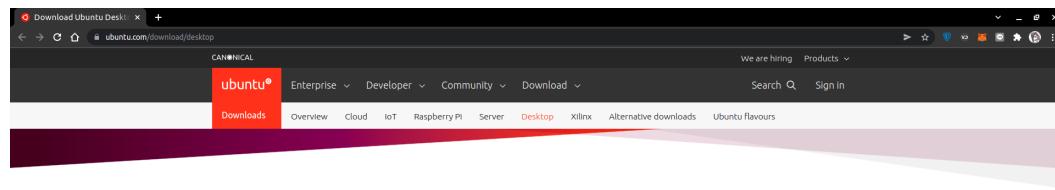
*You can skip this Installation if you already have Ubuntu 20.04 LTS and ROS Noetic on your own PC/Laptop. Go to the Main Topic by [“Click Here”](#)

Install Ubuntu using VMware player

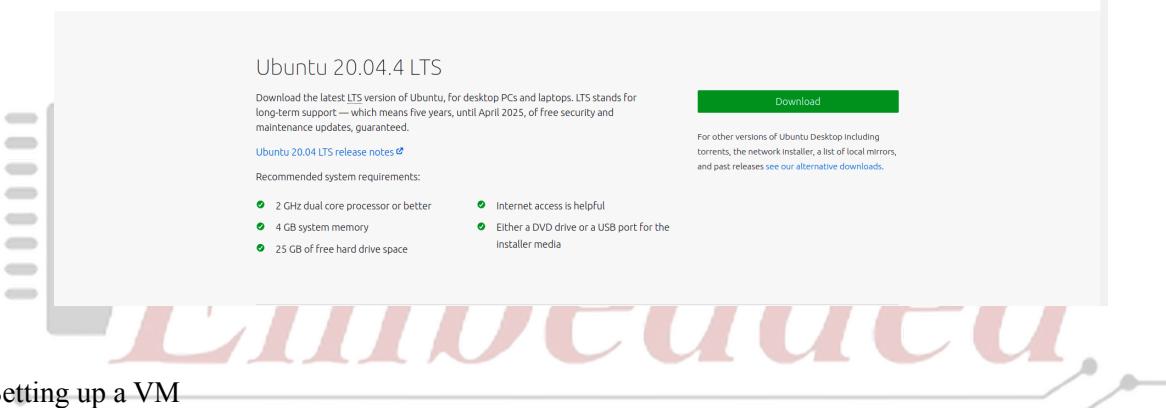
1. Download Ubuntu Image

You can download an Ubuntu image at link: <https://ubuntu.com/download/desktop>

Make sure to save it to a memorable location on your PC! For this tutorial, we will use the Ubuntu 20.04 LTS release.

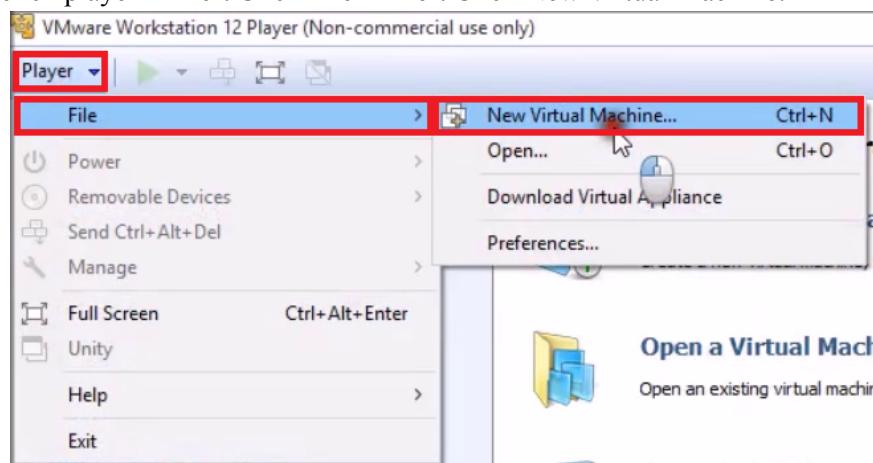


Download Ubuntu Desktop

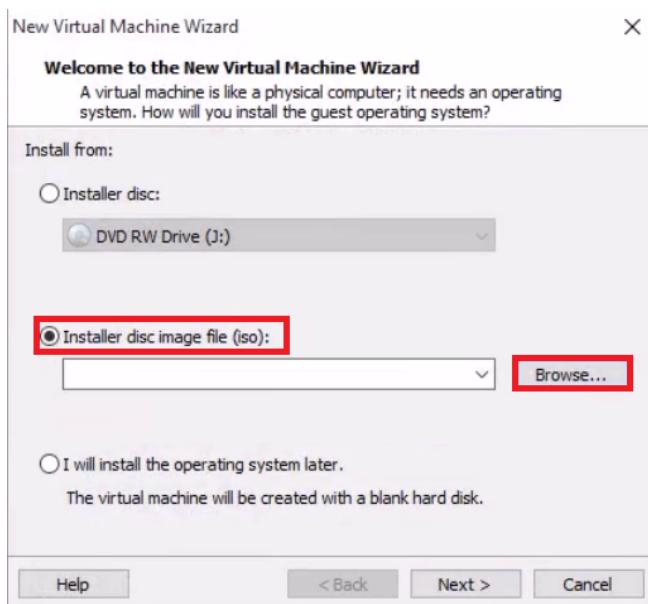


2. Setting up a VM

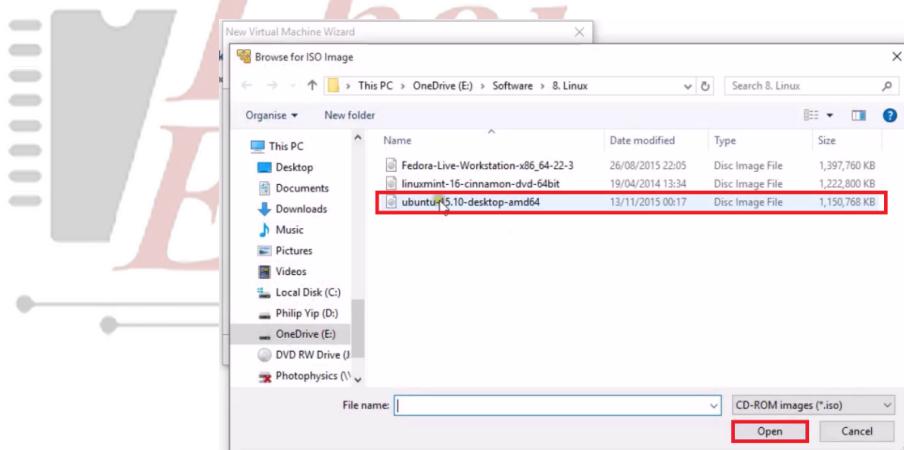
Left click player → Left Click File → Left Click New Virtual Machine:



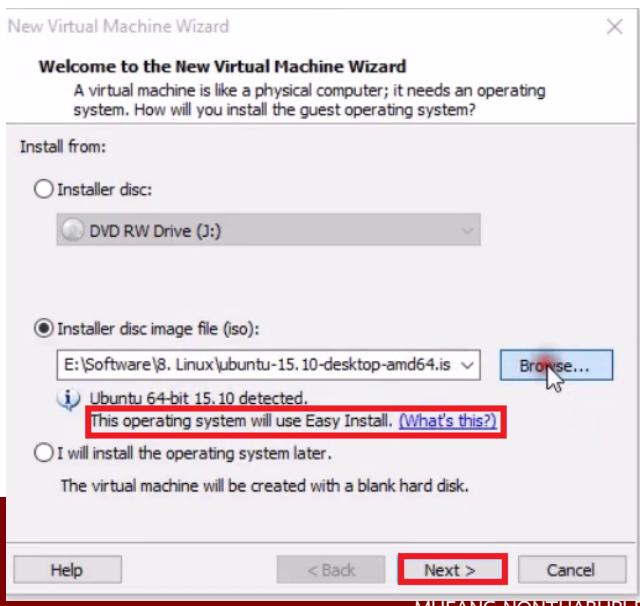
Select Installer Disk Image File (.iso):



Select browse.... and left click the Ubuntu .iso you downloaded:\



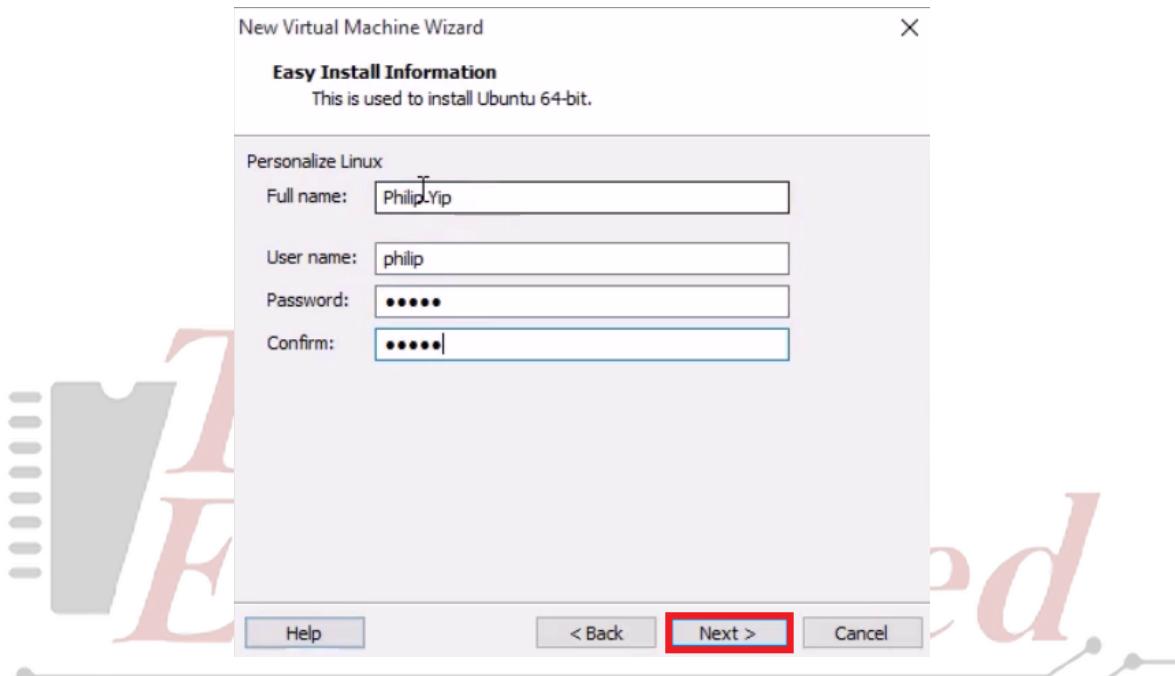
Select Open



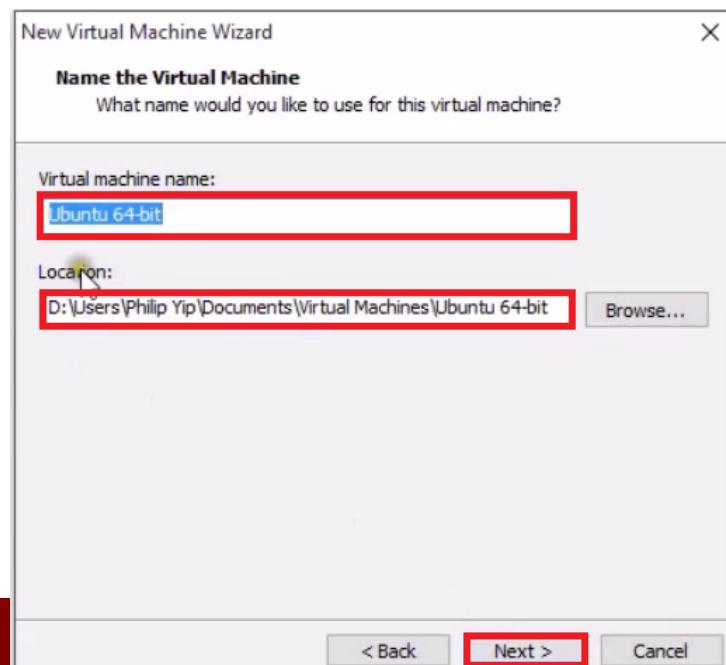
Ensure "This Operating System will use Easy Install" is shown.

With Easy Install VMWare Player will automatically install Ubuntu and VMWare tools without the need for user input. If easy_install is not selected one may have difficulty installing VMWare tools manually. The only disadvantage of easy install is there is no option to select language options using VMWare player and it defaults to English US language and keyboard :(. The language settings for both can easily be changed after installation and I will demonstrate the change of English US to proper English UK.

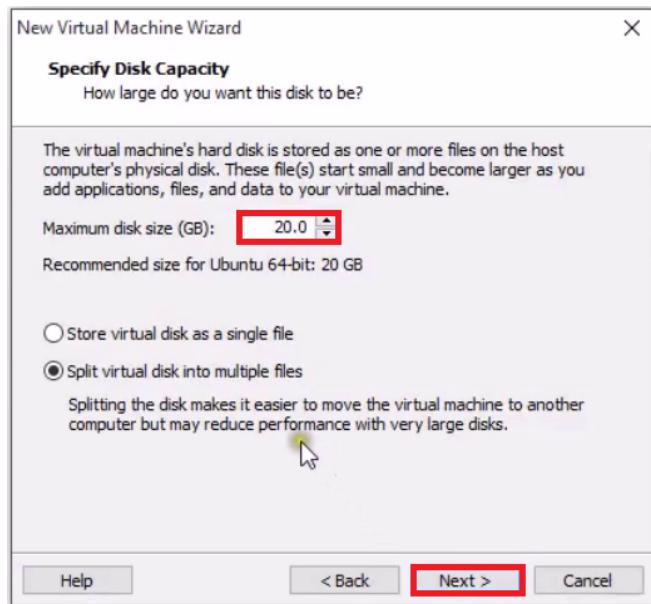
Type in your full name and username. The username has to all be lower case with no spaces. Then type in your password, Ubuntu won't let you install without a password:



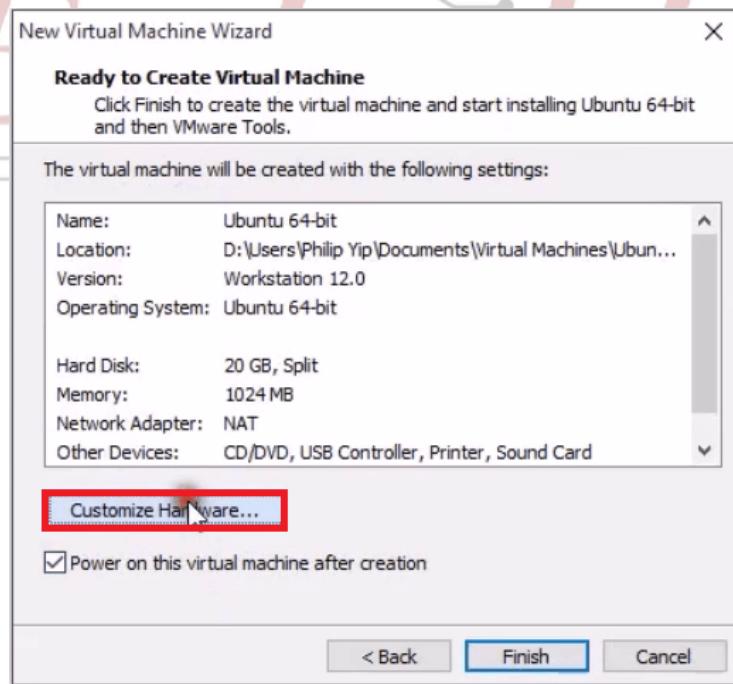
Here you can name your VM and you can also opt to change the location of the VM e.g. to a different HDD/SSD depending what suits.



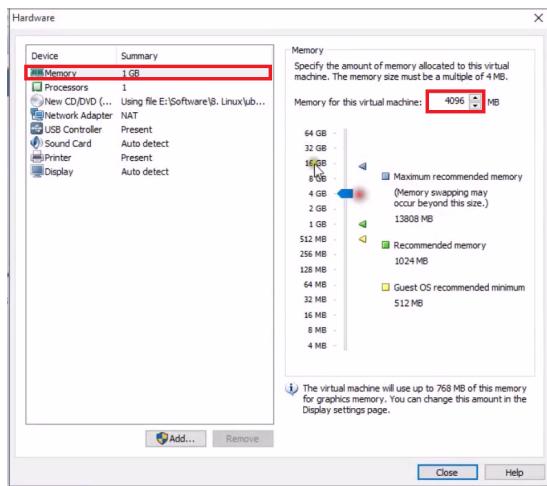
Select next. The default storage is 20 GB but you may increase it. I recommend leaving Split virtual disk into multiple files.



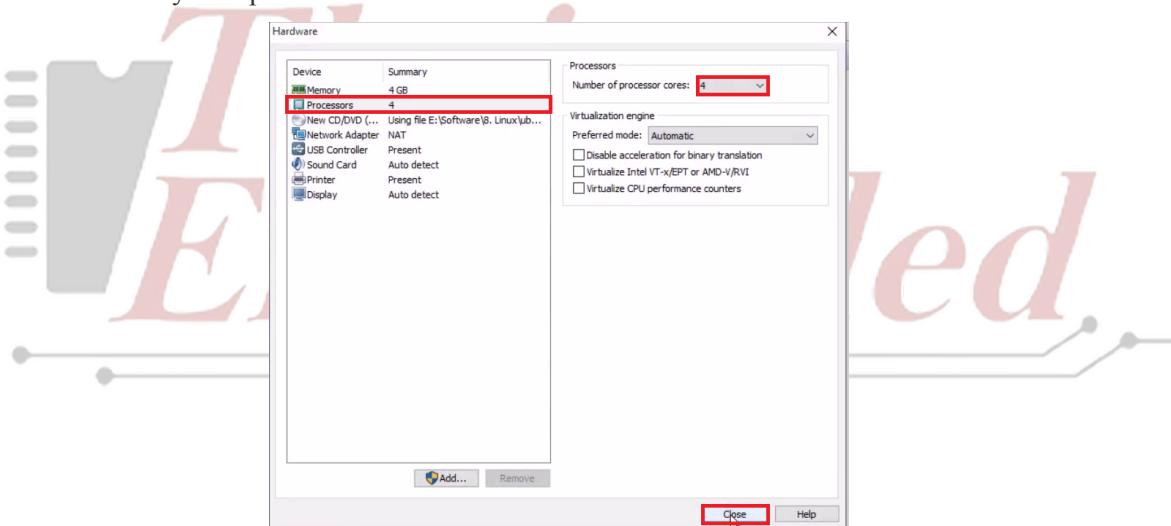
Before installing Ubuntu select Customize Hardware because the default hardware settings are pretty weak:



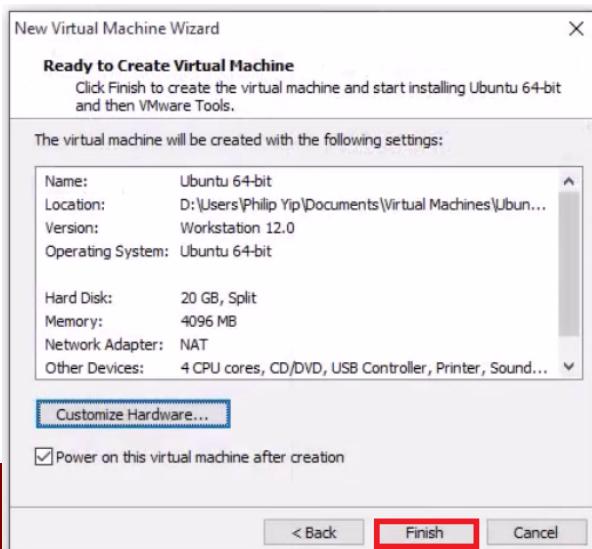
If your computer is capable (has at least 8 GB of RAM) up the RAM to 4 GB:



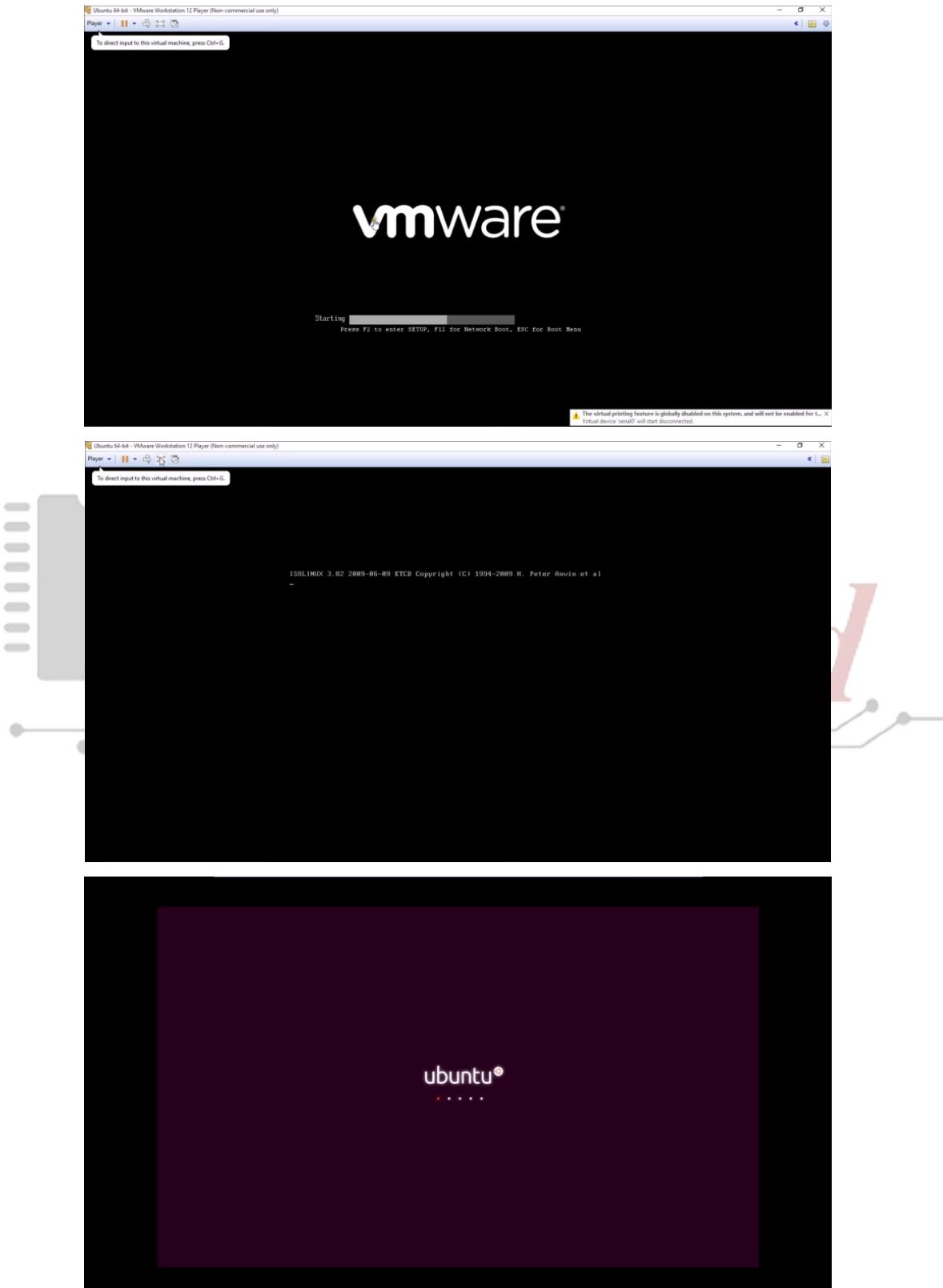
If your computer is capable of at least 8 threads, change the Number of processor cores to 4. If your processor has 4 threads change the Number of processor cores to 2. This will substantially increase the Ubuntu VMs system performance.

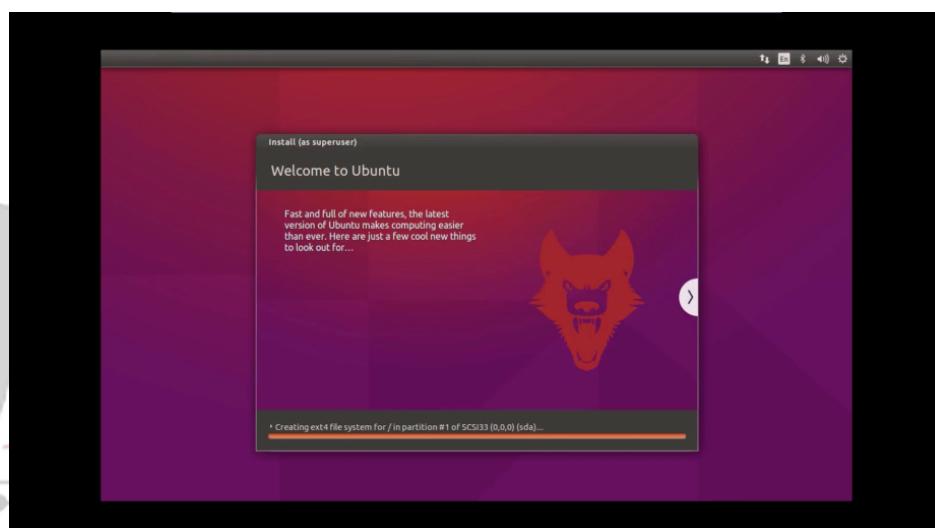
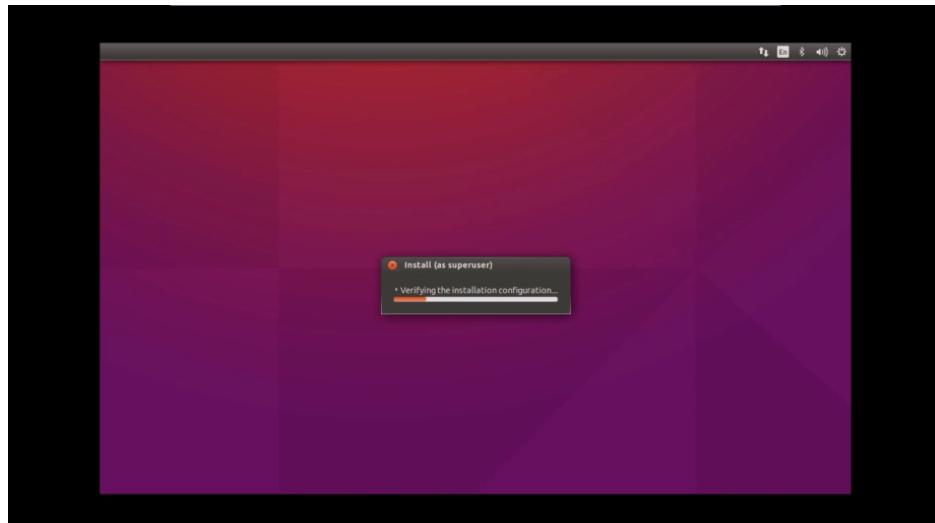


Select close and then Finish



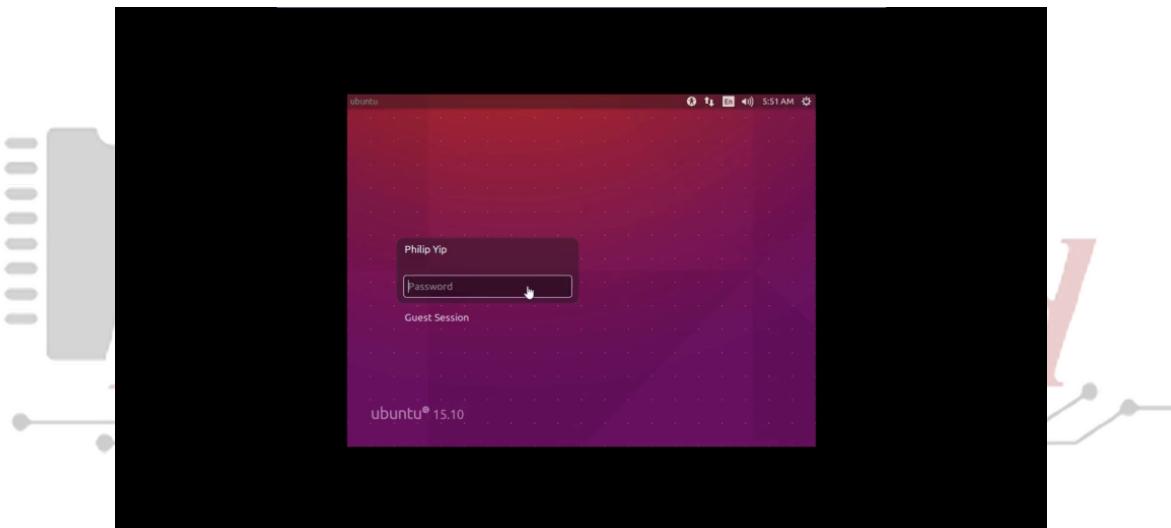
As mentioned the rest of the install will be automated. Make yourself a cup of tea and come back in 5-10 minutes:





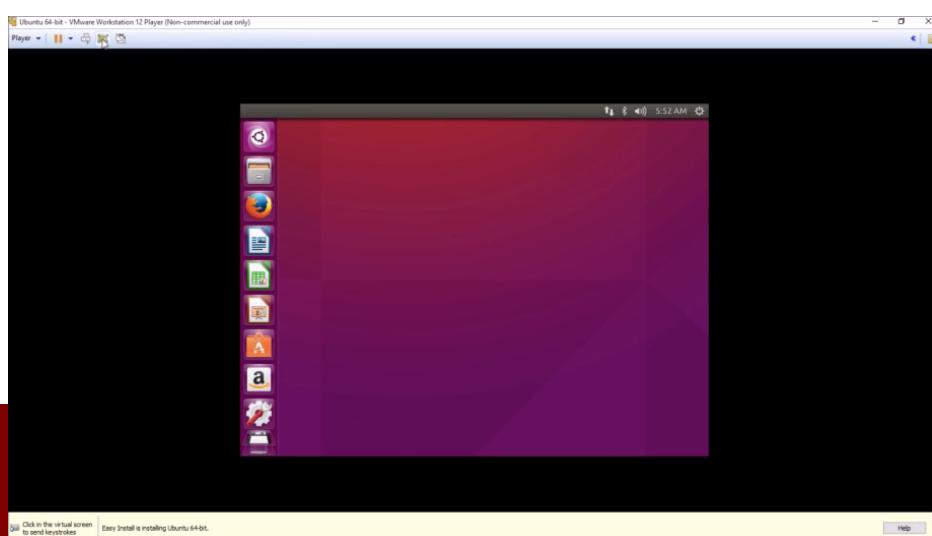


When the install is done you will be prompted to login:

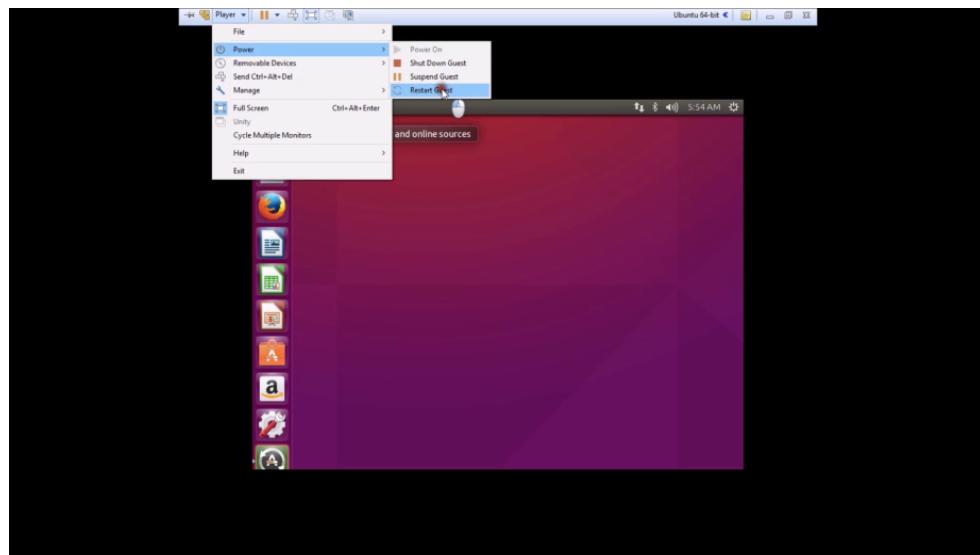


Making Sure VMWare Tools are Installed

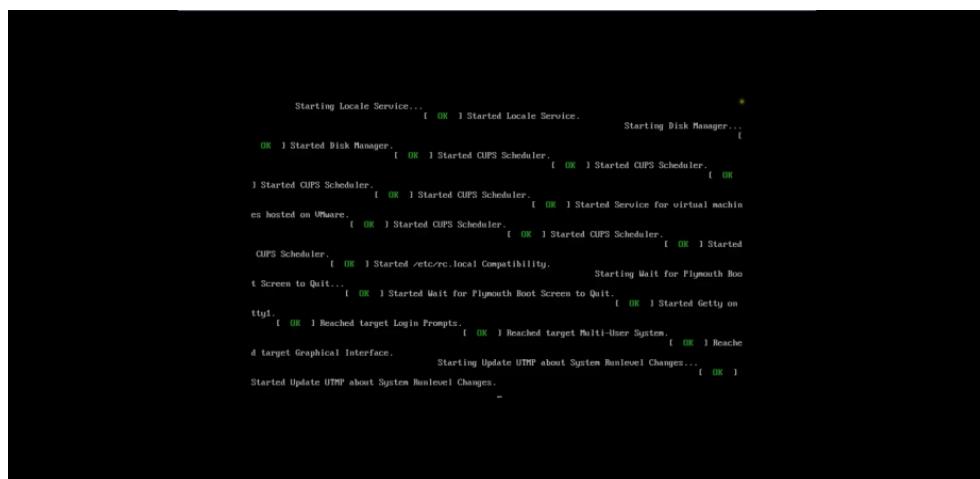
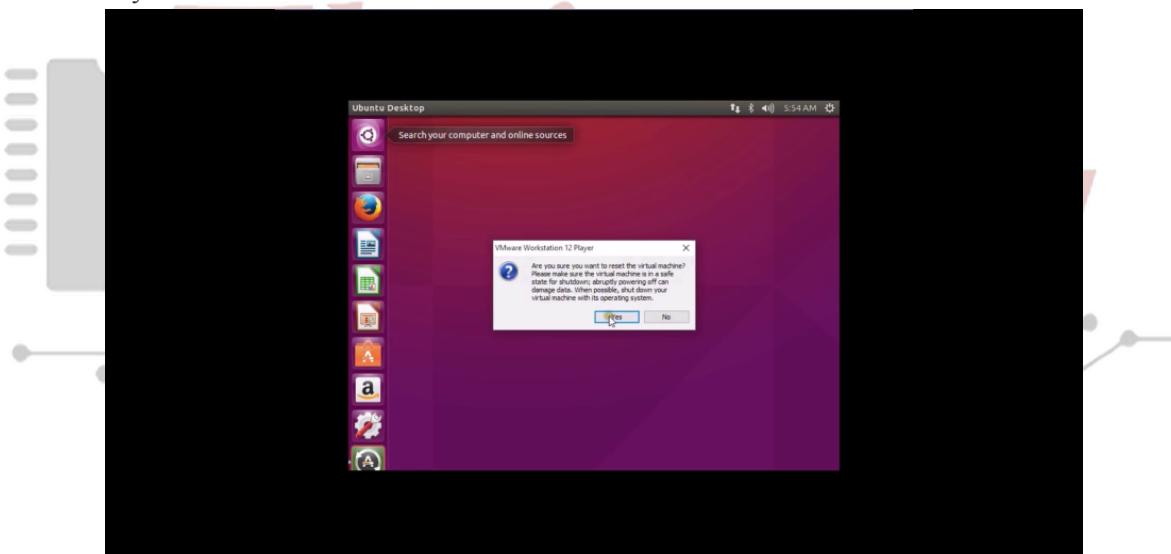
During the first launch the user experience may be terrible because VMWare tools are not installed therefore I recommend signing in waiting 5-10 minutes and then restarting:



Select the player button at the top left and then power and then restart guest:



Select yes:

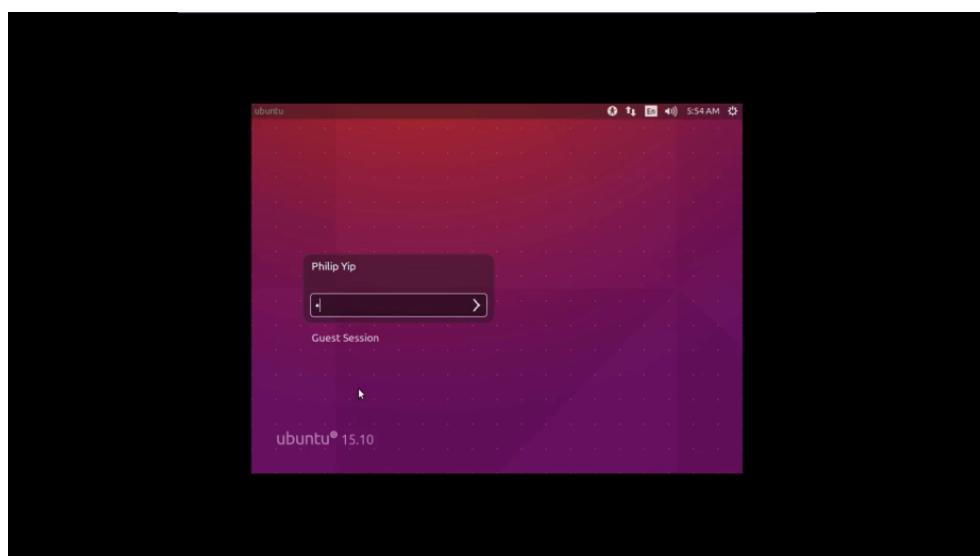


MA Sub-district,
MUEANG Nonthaburi District, Nonthaburi Province 11000

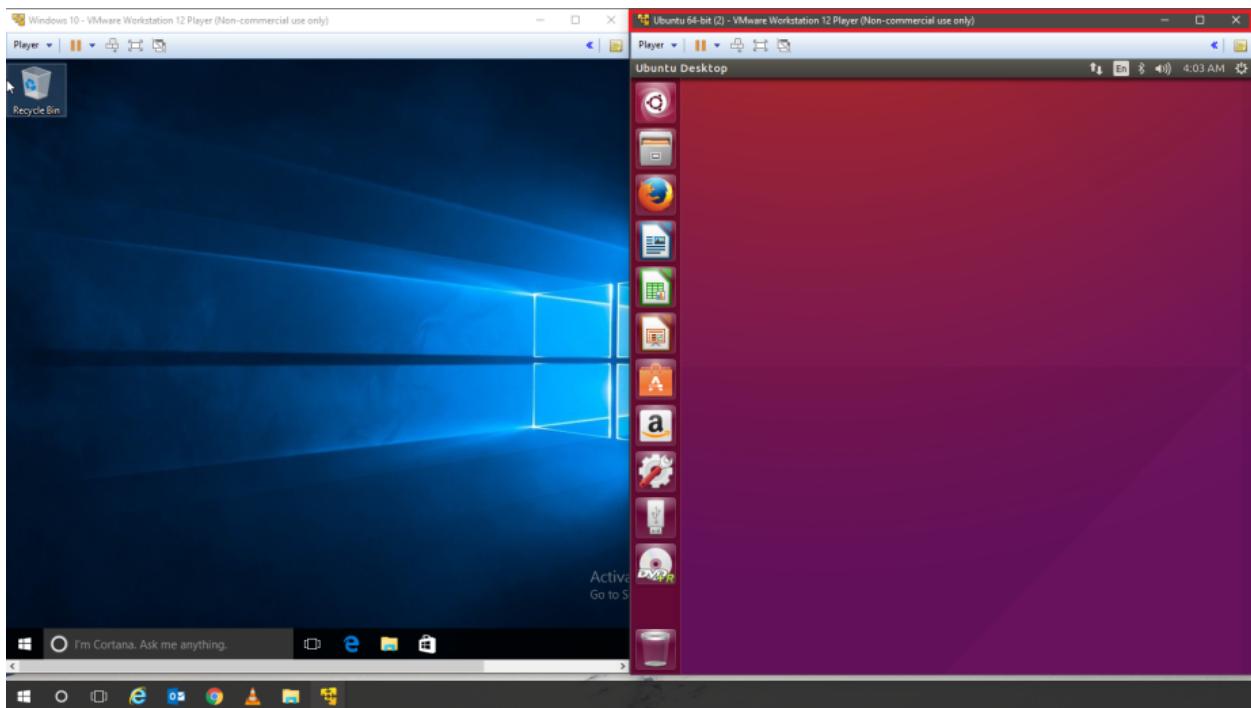




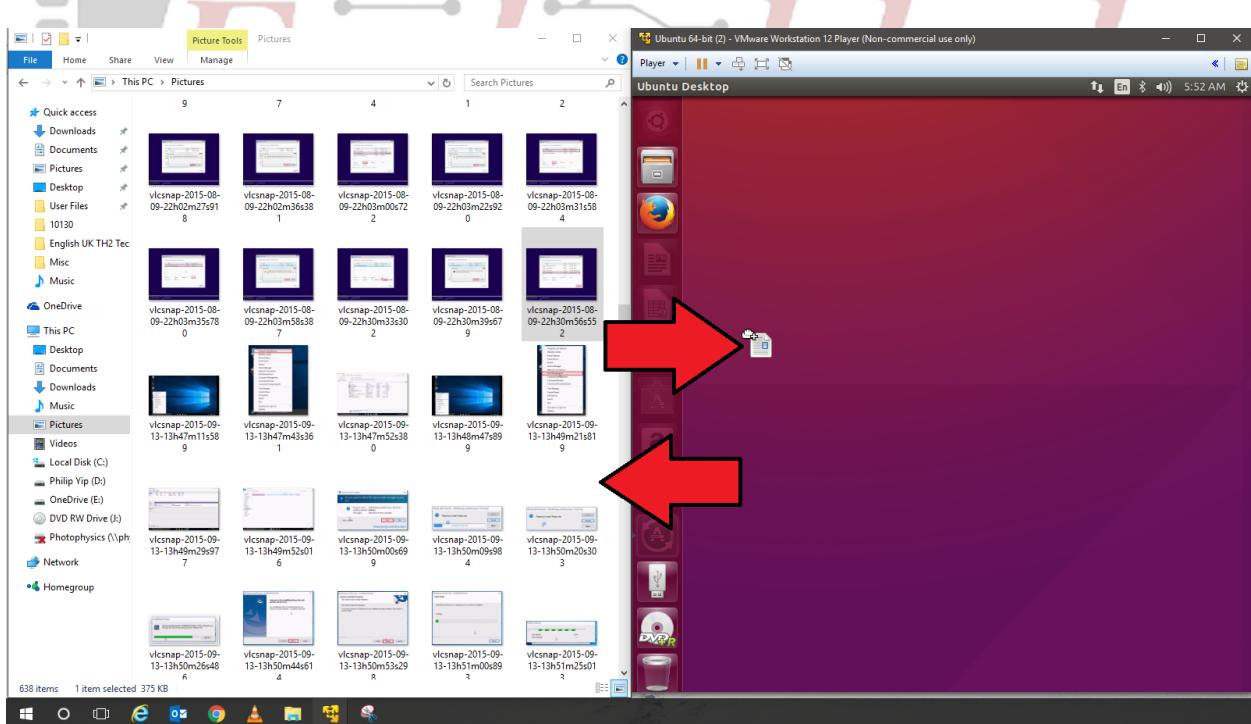
You will be prompted to login again and now it should be full screen:



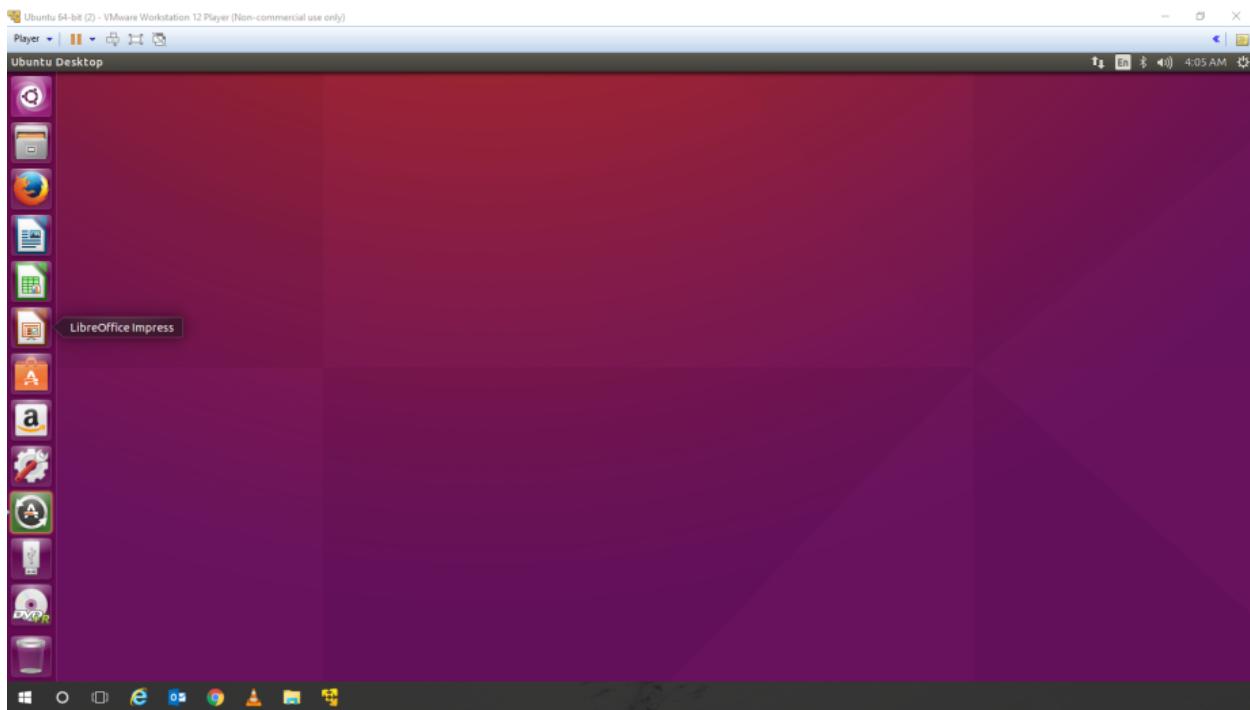
The VM is now full screen if it does not drag and drop the VMWare player Window to the left or right to aero snap half the screen. The VM should automatically resize to fill the Window:



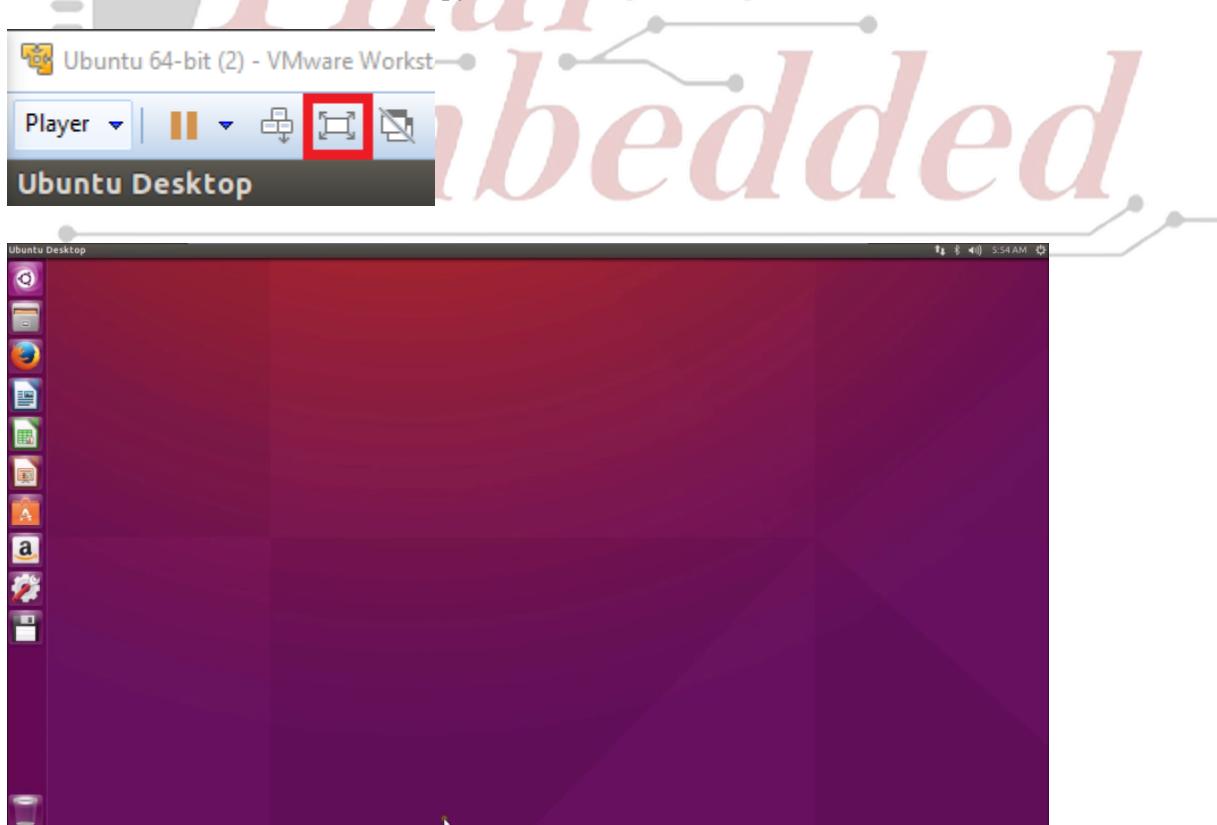
This also allows 2 way drag and drop from the host to the VM:



You can also aero snap it to the top. The VMWare player Window will then be maximized:

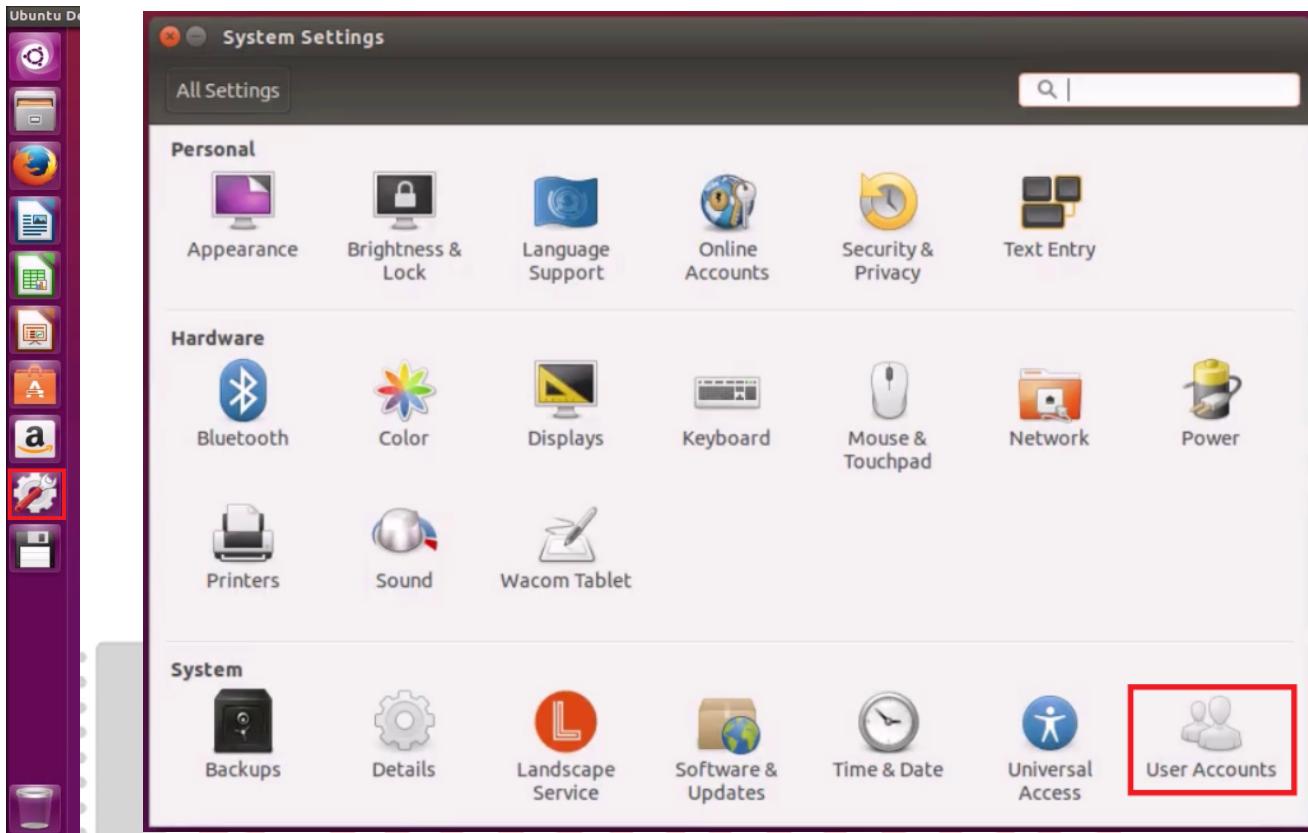


You can also select maximize to occupy the full monitor:

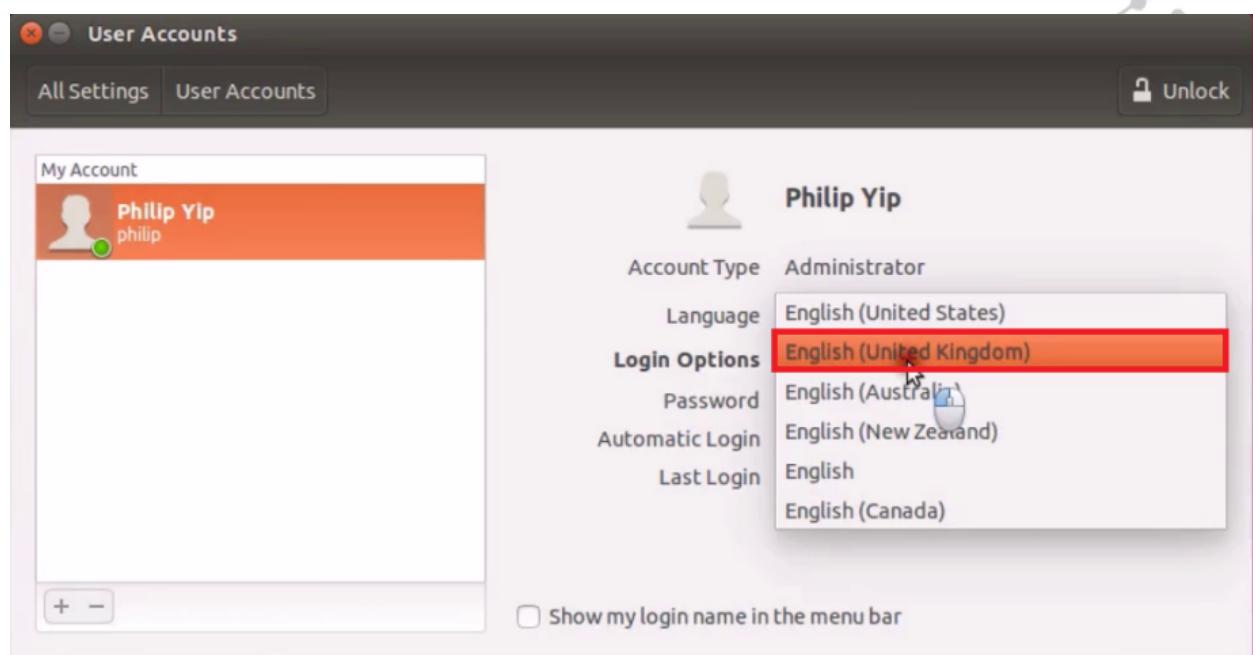


Changing Language and Keyboard

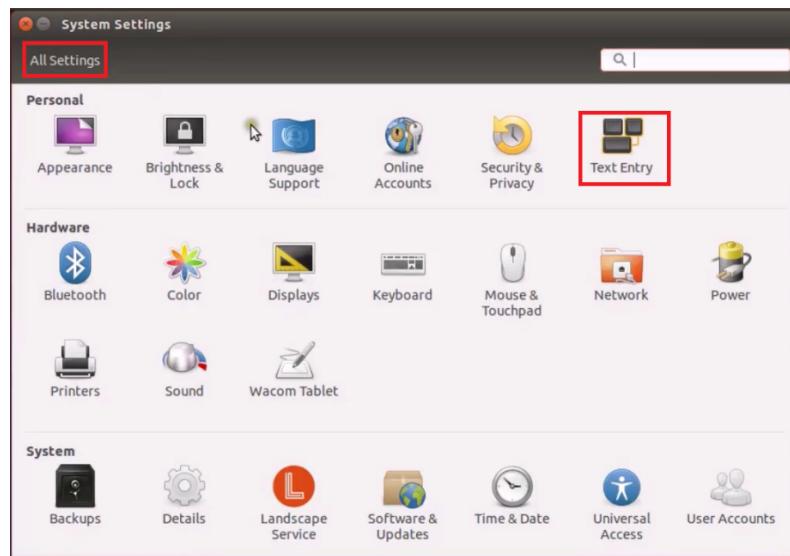
Select the settings crank to the left sidebar and Select User account:



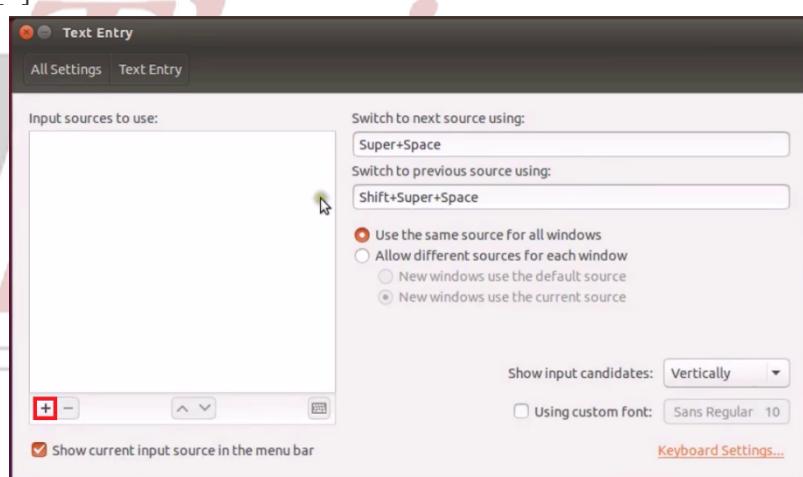
Change language from English (United States) to your desired Language:



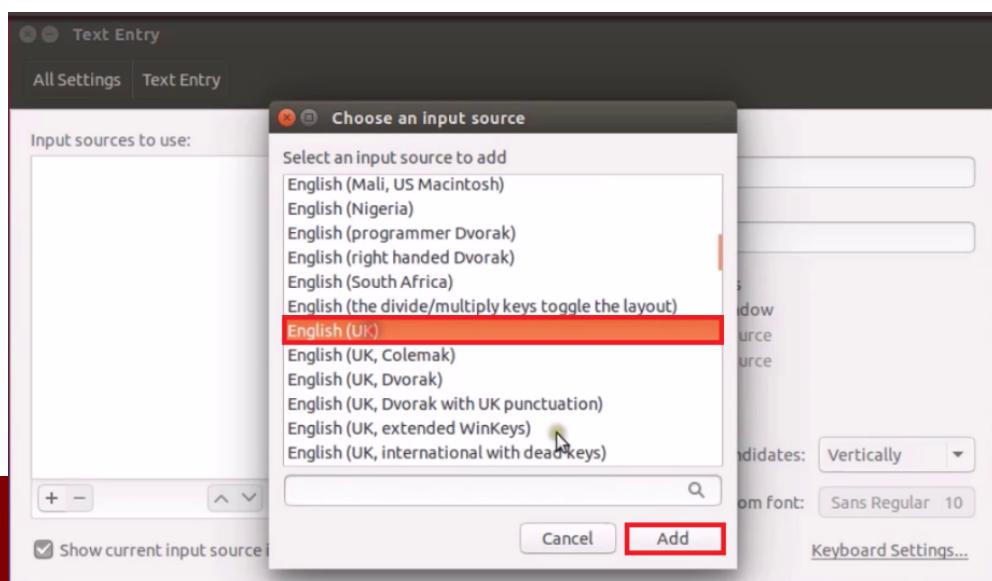
Select All settings and then go to Text Entry:



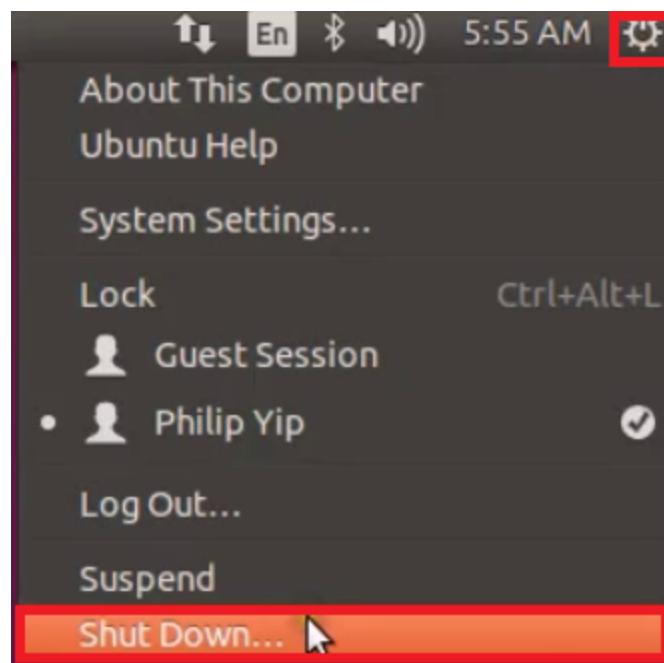
Select [+]



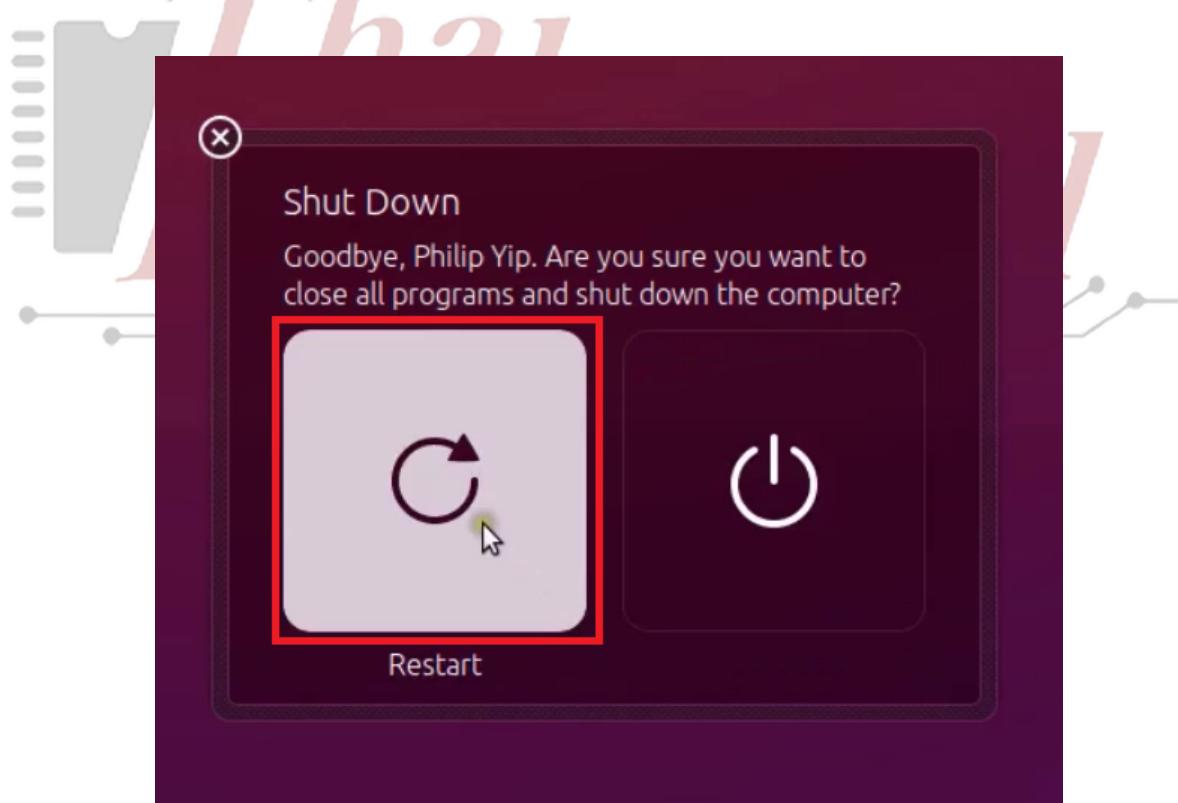
Select English (UK) and then select Add:



Select the power button and then shutdown:



Then restart:



You may now use your Ubuntu VM with your proper keyboard and language settings



Installation of ROS Noetic on Ubuntu

We have 2 methods for install ROS-Noetic are by terminal and synaptic package manager

#Before Install, Let's install useful applications such as Visual Studio and File transfer. It will helpful for programming in Ubuntu

Install Visual Studio for amd64(Work Station)

```
sudo snap install --classic code
```

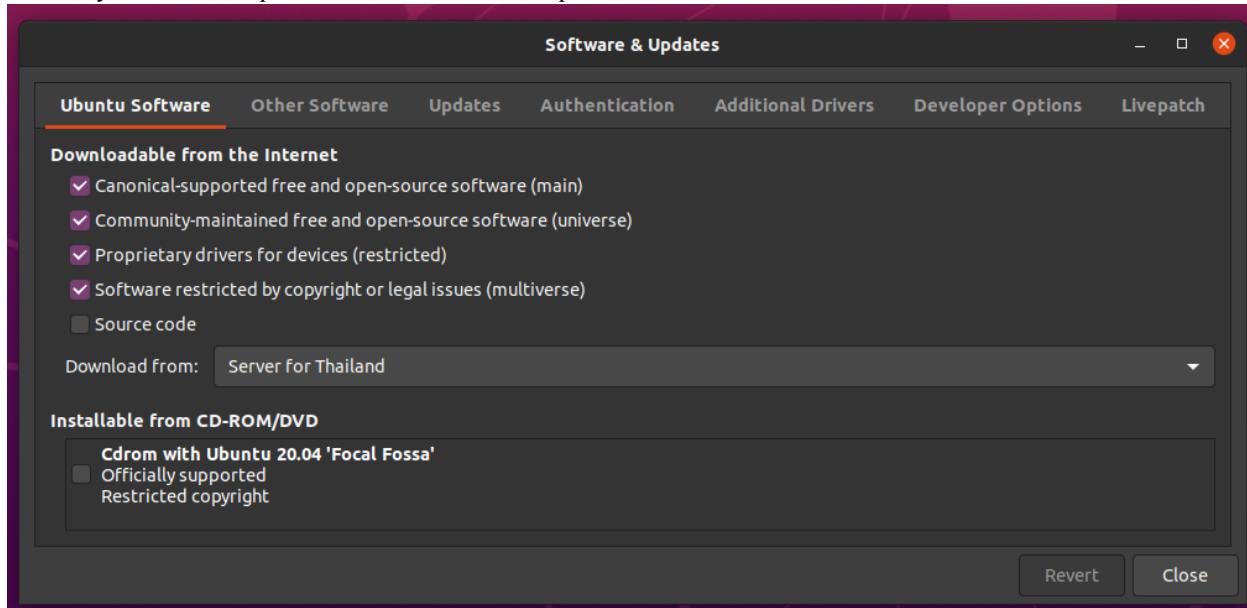
Install File transfers

```
sudo apt-get install openssh-server openssh-client
```

So, Let's get started!

ROS Noetic Terminal method

1. Make your Ubuntu repositories in “Software & Updates”



2. Setup your sources.list

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

3. Setup your keys

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80'  
--recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```



4. Installation

Check software is up to date and upgrade
 sudo apt update; sudo apt upgrade

```
rengy@tesr-9939:~$ sudo apt update; sudo apt upgrade
[sudo] password for rengy:
Hit:1 http://dl.google.com/linux/chrome/deb stable InRelease
Hit:2 http://th.archive.ubuntu.com/ubuntu focal InRelease
Hit:3 http://th.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://archive.canonical.com/ubuntu focal InRelease
Get:5 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Hit:6 https://librealsense.intel.com/Debian/apt-repo focal InRelease
Hit:7 http://packages.ros.org/ros-testing/ubuntu focal InRelease
Fetched 114 kB in 2s (74.4 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
All packages are up to date.
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.

Install ROS desktop-full
sudo apt install ros-noetic-desktop-full
```

```
rengy@tesr-9939:~$ sudo apt install ros-noetic-desktop-full
```

5. Environment setup

Open .bashrc to setup environment by gedit or nano
 gedit ~/.bashrc

And then, fill this data to last line of .bashrc
 source /opt/ros/noetic/setup.bash

```
111
112
113
114
115
116
117
118
119
120
121
122
```

Source .bashrc
 source ~/.bashrc

6. Check ROS version

rosversion -d

```
rengy@tesr-9939:~$ rosversion -d
noetic
```

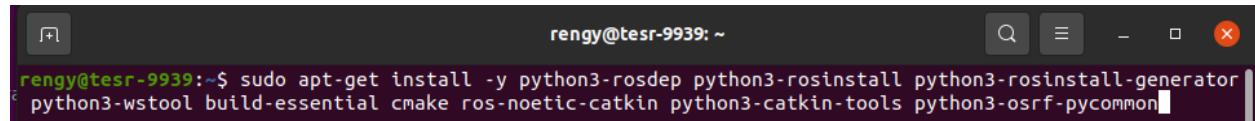
If your installation is complete and by right version “noetic” will show up



[Optional]

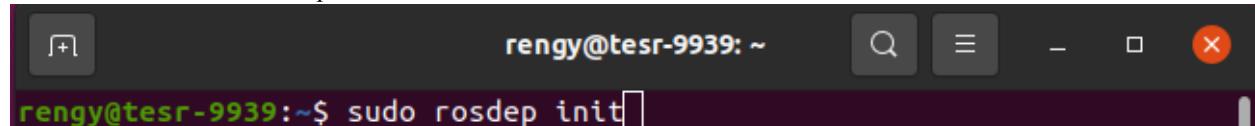
Install dependencies packages for building packages

```
sudo apt install python3-rosdep python3-rosinstall python3-rosinstall-generator python3-wstool  
build-essential cmake python3-catkin-tools python3-osrf-pycommon
```



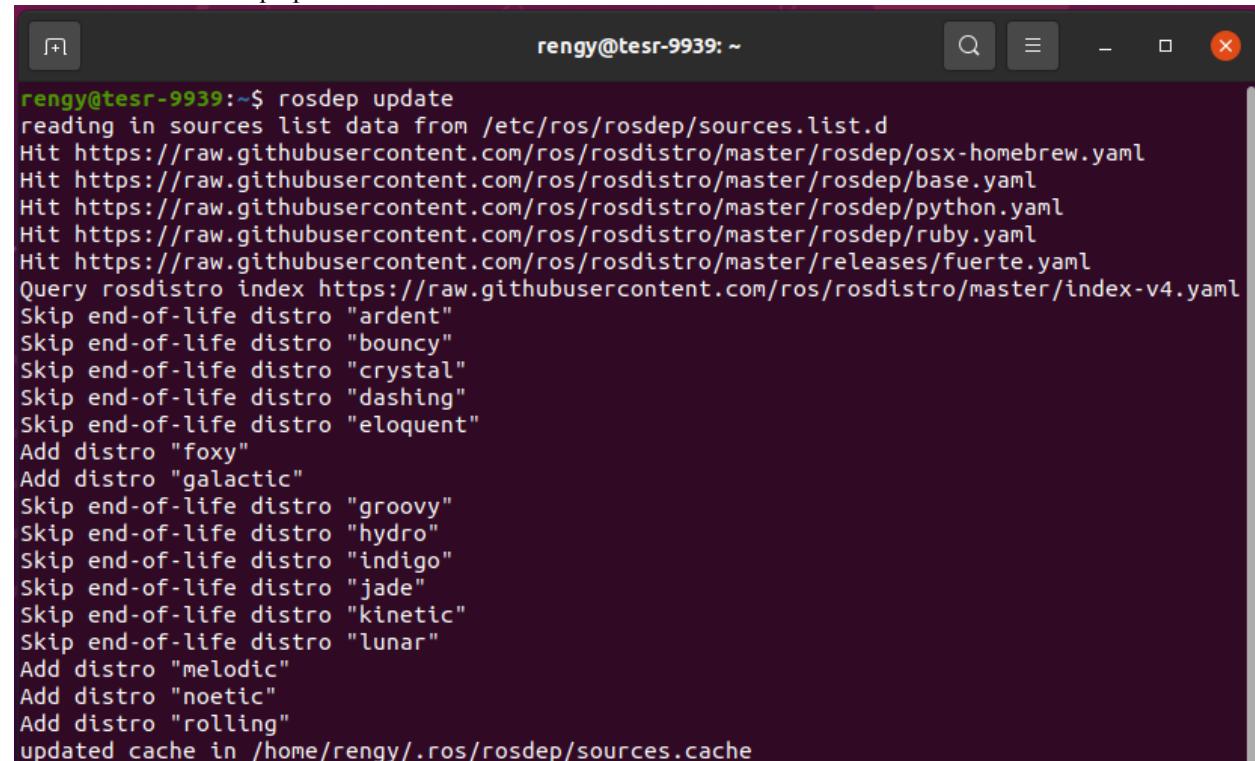
```
rengy@tesr-9939:~$ sudo apt-get install -y python3-rosdep python3-rosinstall python3-rosinstall-generator  
python3-wstool build-essential cmake ros-noetic-catkin python3-catkin-tools python3-osrf-pycommon
```

Initialize and update rosdep
sudo rosdep init



```
rengy@tesr-9939:~$ sudo rosdep init
```

rosdep update

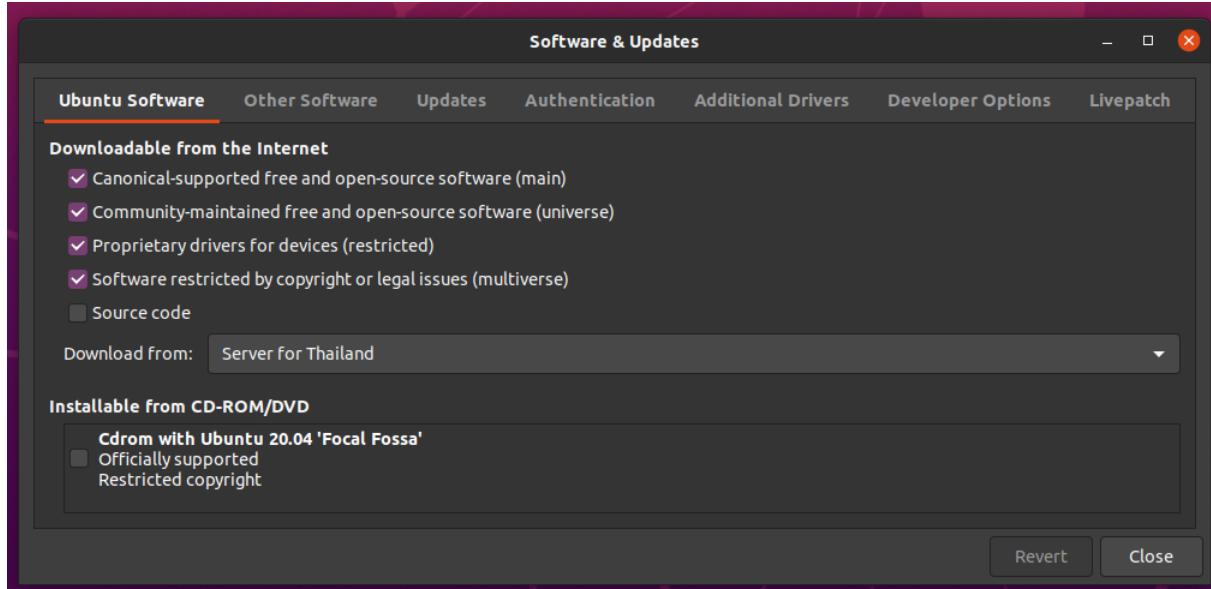


```
rengy@tesr-9939:~$ rosdep update  
reading in sources list data from /etc/ros/rosdep/sources.list.d  
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/osx-homebrew.yaml  
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/base.yaml  
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/python.yaml  
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/ruby.yaml  
Hit https://raw.githubusercontent.com/ros/rosdistro/master/releases/fuerte.yaml  
Query rosdistro index https://raw.githubusercontent.com/ros/rosdistro/master/index-v4.yaml  
Skip end-of-life distro "ardent"  
Skip end-of-life distro "bouncy"  
Skip end-of-life distro "crystal"  
Skip end-of-life distro "dashing"  
Skip end-of-life distro "eloquent"  
Add distro "foxy"  
Add distro "galactic"  
Skip end-of-life distro "groovy"  
Skip end-of-life distro "hydro"  
Skip end-of-life distro "indigo"  
Skip end-of-life distro "jade"  
Skip end-of-life distro "kinetic"  
Skip end-of-life distro "lunar"  
Add distro "melodic"  
Add distro "noetic"  
Add distro "rolling"  
updated cache in /home/rengy/.ros/rosdep/sources.cache
```



ROS Noetic Synaptic package manager method

- ## 1. Make your Ubuntu repositories in “Software & Updates”



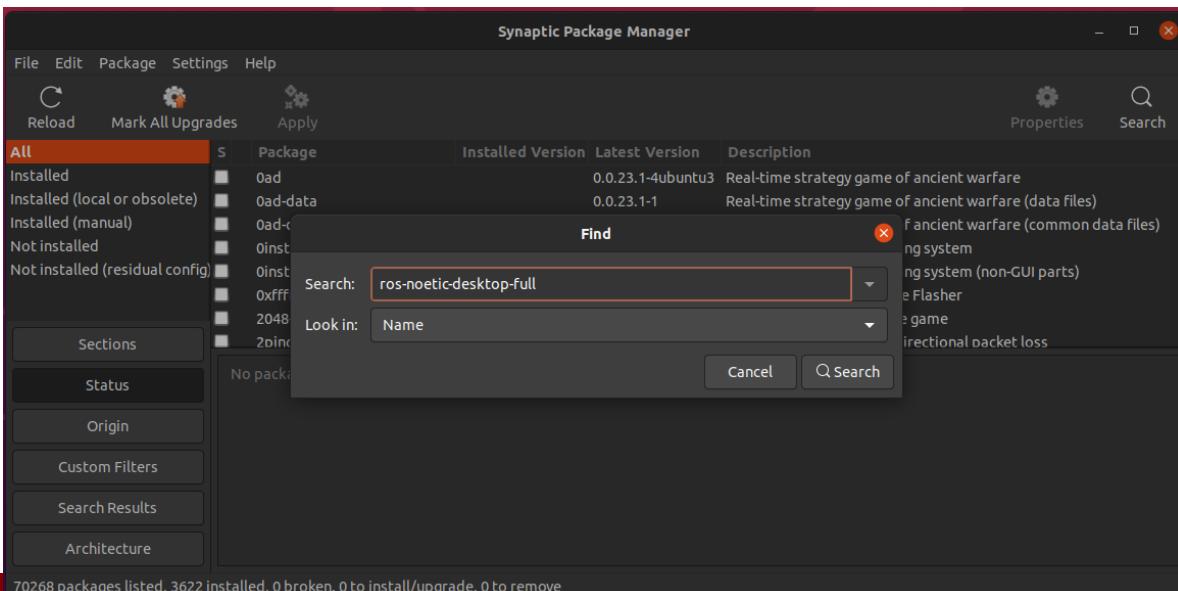
- ## 2. Setup Synaptic package manager

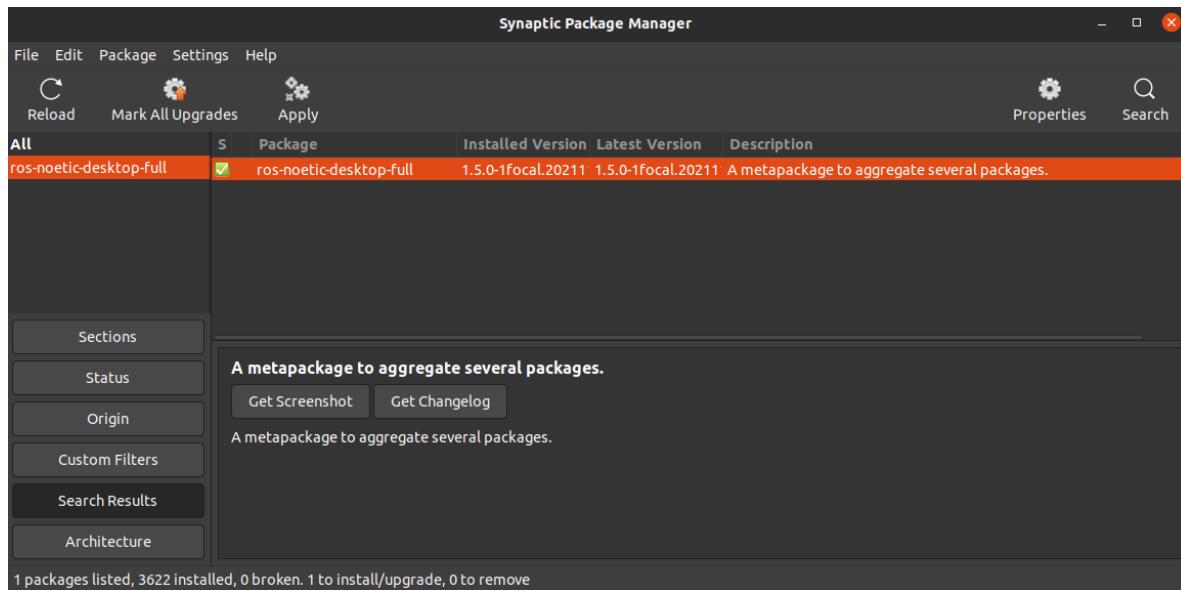
Install “Synaptic package manager”
sudo apt synaptic

```
rengy@tesr-9939:~$ sudo apt install synaptic
Reading package lists... Done
Building dependency tree
Reading state information... Done
synaptic is already the newest version (0.84.6ubuntu5).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

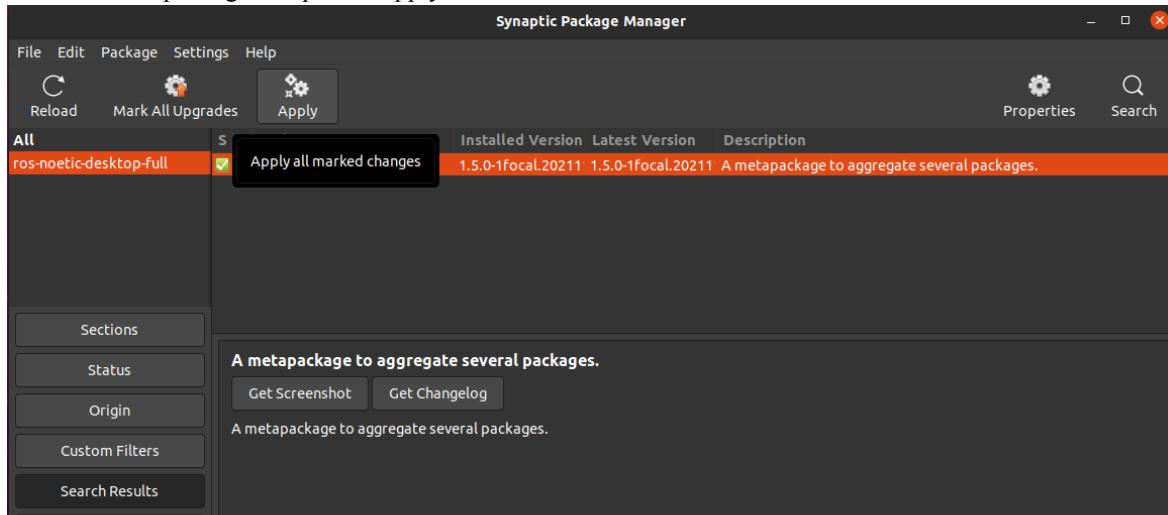
- ### 3. Installation

Open Synaptic packages manager and type “ros-noetic-desktop-full” to search box and mark for installation.





Mark package and press “Apply”



4. Environment setup

Open .bashrc to setup environment by gedit or nano

gedit ~/.bashrc

And then, fill this data to last line of .bashrc

source /opt/ros/noetic/setup.bash

The terminal window shows the contents of the .bashrc file. The user has added the line 'source /opt/ros/noetic/setup.bash' to the end of the file. Below the terminal, a second terminal window shows the command 'gedit -./.bashrc' being run, indicating the file is currently being edited.

```

90 # some more ls aliases
91 alias ll='ls -alF'
92 alias la='ls -A'
93 alias l='ls -CF'
94
95 # Add an "alert" alias for long running commands. Use like so:
96 # sleep 10; alert
97 alias alert='notify-send --urgency=low -l "[$(($? == 0) && echo terminal) || echo error)" "$(
98
99 # Alias definitions.
100 # You may want to put all your additions into a separate file like
101 # ~/.bash_aliases, instead of adding them here directly.
102 # See /usr/share/doc/bash-doc/examples in the bash-doc package.
103
104 if [ -f ./_.bash_aliases ]; then
105 . ./_.bash_aliases
106 fi
107
108 # enable programmable completion features (you don't need to enable
109 # this, if it's already enabled in /etc/bash.bashrc and /etc/profile
110 # sources /etc/bash.bashrc)
111 if ! shopt -oq posix; then
112   if [ -f /usr/share/bash-completion/bash_completion ]; then
113     . /usr/share/bash-completion/bash_completion
114   elif [ -f /etc/bash_completion ]; then
115     /etc/bash_completion
116   fi
117 fi
118
119 source /opt/ros/noetic/setup.bash
120

```



Source .bashrc

source ~/.bashrc

```
rengy@tesr-9939:~$ source ~/._.bashrc
rengy@tesr-9939:~$
```

5. Check ROS version

If your installation is complete and by right version “noetic” will show up
rosversion -d

```
rengy@tesr-9939:~$ rosversion -d
noetic
```

[Optional]

Install dependencies packages for building packages. Type this list in search box and mark it to installation

- python3-rosdep
- python3-rosinstall
- python3-rosinstall-generator
- Python3-wstool
- Build-essential
- Cmake
- ros-noetic-catkin
- python3-catkin-tools
- python3-osrf-pycommon

* You can see marked list at “Custom Filter > Marked Changes”

Synaptic Package Manager

All	S	Package	Installed Version	Latest Version	Description
Broken	<input checked="" type="checkbox"/>	build-essential	12.8ubuntu1.1	12.8ubuntu1.1	Informational list of build-essential packages
Community Maintained (insta	<input checked="" type="checkbox"/>	cmake	3.16.3-1ubuntu1	3.16.3-1ubuntu1	cross-platform, open-source make system
Marked Changes	<input checked="" type="checkbox"/>	python3-catkin-tools	0.7.2-1	0.7.2-1	Command line tools for working with catkin.
Missing Recommends	<input checked="" type="checkbox"/>	python3-osrf-pycommon	1.0.0-1	1.0.0-1	Commonly needed Python modules, used by Python software deve
Package with Debconf	<input checked="" type="checkbox"/>	python3-rosdep	0.21.0-1	0.21.0-1	rosdep package manager abstraction tool for ROS
Search Filter	<input checked="" type="checkbox"/>	python3-rosinstall		0.7.8-4	Installer for Robot OS (Python 3)
Upgradable (upstream)	<input checked="" type="checkbox"/>	python3-rosinstall-generator		0.1.22-1	A tool for generating rosinstall files
	<input checked="" type="checkbox"/>	python3-wstool	0.1.18-2	0.1.18-2	Commands to manage multi-VCS repositories (For Robot OS) Python 3
	<input checked="" type="checkbox"/>	ros-noetic-catkin	0.8.10-1focal.2021	0.8.10-1focal.2021	Low-level build system macros and infrastructure for ROS.

No package is selected.

9 packages listed, 3622 installed, 0 broken. 9 to install/upgrade, 0 to remove; 214 kB will be used



Initialize and update rosdep

Open terminal and initialize rosdep
sudo rosdep init

```
rengy@tesr-9939:~$ sudo rosdep init
```

rosdep update

```
rengy@tesr-9939:~$ rosdep update
reading in sources list data from /etc/ros/rosdep/sources.list.d
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/osx-homebrew.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/base.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/python.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/ruby.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/releases/fuerte.yaml
Query rosdistro index https://raw.githubusercontent.com/ros/rosdistro/master/index-v4.yaml
Skip end-of-life distro "ardent"
Skip end-of-life distro "bouncy"
Skip end-of-life distro "crystal"
Skip end-of-life distro "dashing"
Skip end-of-life distro "eloquent"
Add distro "foxy"
Add distro "galactic"
Skip end-of-life distro "groovy"
Skip end-of-life distro "hydro"
Skip end-of-life distro "indigo"
Skip end-of-life distro "jade"
Skip end-of-life distro "kinetic"
Skip end-of-life distro "lunar"
Add distro "melodic"
Add distro "noetic"
Add distro "rolling"
updated cache in /home/rengy/.ros/rosdep/sources.cache
```

Install Dependent ROS Packages

Run command line following below:

```
sudo apt-get install ros-noetic-joy ros-noetic-teleop-twist-joy \
ros-noetic-teleop-twist-keyboard ros-noetic-laser-proc \
ros-noetic-rgbd-launch ros-noetic-rosserial-arduino \
ros-noetic-rosserial-python ros-noetic-rosserial-client \
ros-noetic-rosserial-msgs ros-noetic-amcl ros-noetic-map-server \
ros-noetic-move-base ros-noetic-urdf ros-noetic-xacro \
ros-noetic-compressed-image-transport ros-noetic-rqt* ros-noetic-rviz \
ros-noetic-gmapping ros-noetic-navigation ros-noetic-interactive-markers
```

```
rengy@tesr-9939:~$ sudo apt-get install ros-noetic-joy ros-noetic-teleop-twist-joy \
> ros-noetic-teleop-twist-keyboard ros-noetic-laser-proc \
> ros-noetic-rgbd-launch ros-noetic-rosserial-arduino \
> ros-noetic-rosserial-python ros-noetic-rosserial-client \
> ros-noetic-rosserial-msgs ros-noetic-amcl ros-noetic-map-server \
> ros-noetic-move-base ros-noetic-urdf ros-noetic-xacro \
> ros-noetic-compressed-image-transport ros-noetic-rqt* ros-noetic-rviz \
> ros-noetic-gmapping ros-noetic-navigation ros-noetic-interactive-markers
[sudo] password for rengy: [REDACTED]
```



Installation of moveit

Step1 Prepare dependencies package

First thing first, make sure you're keep most of packages are up-to-date

- Open the terminal by press “Ctrl + Alt + T”
- Type “rosdep update”
“sudo apt update”
“sudo apt dist-upgrade”

Install catkin the ROS build system

- Type “sudo apt install ros-noetic-catkin python3-catkin-tools python3-osrf-pycommon”

Install wstool

- Type “sudo apt install python3-wstool”

Step2 Create a Catkin Workspace and Download Moveit Source

*Moveit is require to have a catkin workspace setup because of the version of tutorials use master branch which is being actively developed, you will most likely need to build all of Moveit from source

Open the terminal by press “Ctrl + Alt + T” and type command following as:

```
“mkdir -p ~/ws_moveit/src”
“cd ~/ws_moveit/src”
“wstool init .”
“wstool merge -t . https://raw.githubusercontent.com/ros-planning/moveit/master/moveit.rosinstall”
“wstool remove moveit_tutorials”
“wstool update -t .”
```

Download Example Code

```
“cd ~/ws_moveit/src”
“git clone https://github.com/ros-planning/moveit_tutorials.git -b master”
“git clone https://github.com/ros-planning/panda_moveit_config.git -b noetic-devel”
```

Step3 Build your Catkin Workspace

This step below will install from debian any package dependencies not in the workspace yet.

Open the terminal by press “Ctrl + Alt + T” and type:

```
“cd ~/ws_moveit/src”
“rosdep install -y --from-paths . --ignore-src --rosdistro noetic”
```

*If you've found any error while this step. Try to type following this line:

```
“sudo sh -c ‘echo “deb http://packages.ros.org/ros-testing/ubuntu ${lsb_release -sc} main”
> /etc/apt/sources.list.d/ros-latest.list’”
“sudo apt update”
```

Configure your catkin workspace:

```
“cd ~/ws_moveit”
“catkin config –extend /opt/ros/${ROS_DISTRO} --cmake-args -DCMAKE_BUILD_TYPE=Release”
“catkin build”
```

Source the catkin workspace:

```
“source ~/ws_moveit/devel/setup.bash”
```

[Recommended] Add the previous command to your .bashrc:

```
“echo ‘source ~/ws_moveit/devel/setup.bash’ >> ~/.bashrc”
```



Download and install the dependencies source packages

First thing first, type “sudo apt install ros-noetic-franka-description” to install franka-description. And then, Download files below.

- tesr_ros_six_servo_robot_pkg.zip
- six_servo.zip
- tesr_ros_six_servo_arm_moveit_config.zip

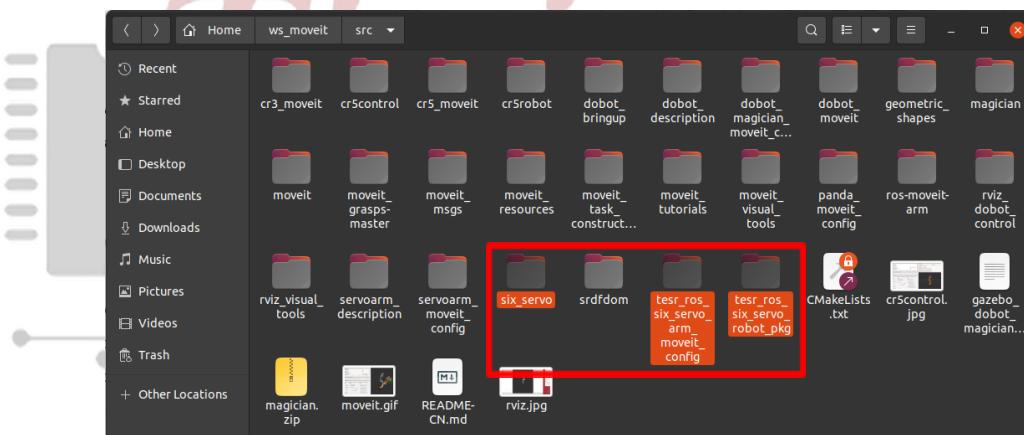
Source package explanation

- tesr_ros_six_servo_robot_pkg is contains the launch file for joystick control and shows the monitoring. And the launch file shows the simulation of a six-servo robot’s motion on Rviz using the python script.
- six_servo.zip is contain the URDF file that is required to use in moveit
- tesr_ros_six_servo_arm_moveit_config is contain the configuration files, launch files and python using control six_servo_arm in moveit

*Keep that in mind, the name of extracted source pkg should as same as original name before extract. If not and error about can’t find the dependencies pkg. So, the package.xml file “must” edit to fix a problem that occur.

Setup package in workspace

- After download the zip files, move and extract it separately to ws_moveit/src



- Type “cd ws_moveit”, “catkin_make” wait until finish [100%]
 - And then“source devel/setup.bash”

```
rengy@tesr-9939:~/ws_moveit$ cd ws_moveit
rengy@tesr-9939:~/ws_moveit$ catkin_make
```

```
[ 94%] Automatic MOC for target motion_planning_tasks_rviz_plugin
[ 94%] Built target motion_planning_tasks_rviz_plugin_autogen
[ 95%] Built target moveit_setup_assistant_tools
[ 96%] Built target moveit_motion_planning_rviz_plugin
[ 96%] Automatic MOC for target moveit_setup_assistant_updater
[ 96%] Automatic MOC for target moveit_setup_assistant_widgets
[ 96%] Built target moveit_setup_assistant_updater_autogen
[ 96%] Built target moveit_setup_assistant_widgets_autogen
[ 97%] Built target motion_planning_tasks_rviz_plugin
[ 97%] Built target moveit_setup_assistant_updater
[ 98%] Built target moveit_setup_assistant_widgets
[ 98%] Automatic MOC for target moveit_setup_assistant
[ 98%] Built target moveit_setup_assistant_autogen
[ 98%] Built target moveit_setup_assistant
[100%] Linking CXX shared library /home/rengy/ws_moveit/devel/lib/python3/dist-packages/moveit_ros_planning_interface/_moveit_move_group_interface.so
[100%] Built target moveit_move_group_interface_python
rengy@tesr-9939:~/ws_moveit$ source devel/setup.bash
rengy@tesr-9939:~/ws_moveit$
```



How to launch tesr_ros_six_servo_robot_pkg

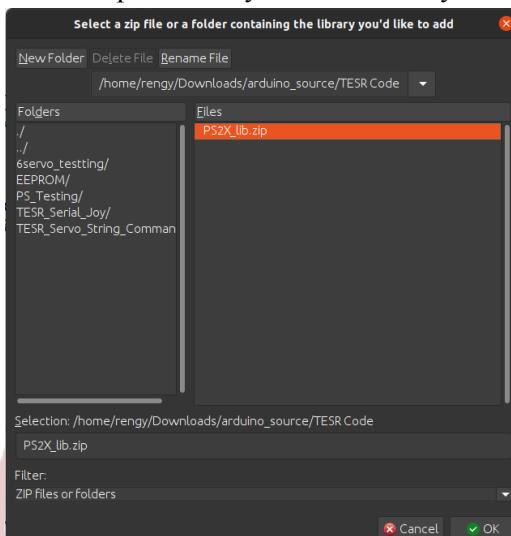
Setup of Arduino board and Setup Robot home pose

*This setup is for robot operation core. So, please set up the following step by steps to make the robot arm operate properly and prevent unpredicted issues.

Step 1

Download TESR Code.zip and Execute.

1.1 Open Arduino and go to Sketch > Import Library... > Add Library... and select "PS2X_lib.zip"



Step 2

Verify and Upload code in Folder as following:

- 2.1 /TESR Code/EEPROM/EEPROM_Write/EEPROM_Write.ino
- 2.2 /TESR Code/TESR_Serial_Joy/TESR_Serial_Joy.ino



* 2.1 for Set default for home position into microcontroller memory and 2.2 is to make the robot available to connect with the joystick.

** If Robot Arm doesn't go to home pose while Turn-On. So, Turn-off robot with an Unplug power adapter and USB cable. And then, turn the robot on and repeat step 2 again.



Step 3

After step 2, the robot should be able to be controlled by a Joystick. So, turn-on the joystick and observe the green light status labeled as "RX" on the microcontroller board. if it is bright without blinking. it means the joystick and robot are connected and it ready to work



Operate Joystick control and monitoring on rviz by ROS launch file



Step 1

Connect the USB cable to your PC or Laptop and type "ls /dev/tty*" to find which port do we can connect such as /dev/ttyUSB0

give the permission to connect for /dev/ttyUSB0 by type "sudo chmod +x /dev/ttyUSB0"

```
rengy@tesr-9939:~$ ls /dev/tty*
/dev/tty  /dev/tty2  /dev/tty31  /dev/tty43  /dev/tty55  /dev/ttys0  /dev/ttys31
/dev/tty0  /dev/tty20  /dev/tty32  /dev/tty44  /dev/tty56  /dev/ttys0  /dev/ttys4
/dev/tty1  /dev/tty21  /dev/tty33  /dev/tty45  /dev/tty57  /dev/ttys1  /dev/ttys21
/dev/tty10 /dev/tty22  /dev/tty34  /dev/tty46  /dev/tty58  /dev/ttys10 /dev/ttys22
/dev/tty11 /dev/tty23  /dev/tty35  /dev/tty47  /dev/tty59  /dev/ttys11 /dev/ttys23
/dev/tty12 /dev/tty24  /dev/tty36  /dev/tty48  /dev/tty6  /dev/ttys12 /dev/ttys24
/dev/tty13 /dev/tty25  /dev/tty37  /dev/tty49  /dev/tty60  /dev/ttys13 /dev/ttys25
/dev/tty14 /dev/tty26  /dev/tty38  /dev/tty5  /dev/tty61  /dev/ttys14 /dev/ttys26
/dev/tty15 /dev/tty27  /dev/tty39  /dev/tty50  /dev/tty62  /dev/ttys15 /dev/ttys27
/dev/tty16 /dev/tty28  /dev/tty4  /dev/tty51  /dev/tty63  /dev/ttys16 /dev/ttys28
/dev/tty17 /dev/tty29  /dev/tty40  /dev/tty52  /dev/tty7  /dev/ttys17 /dev/ttys29
/dev/tty18 /dev/tty3  /dev/tty41  /dev/tty53  /dev/tty8  /dev/ttys18 /dev/ttys3
/dev/tty19 /dev/tty30  /dev/tty42  /dev/tty54  /dev/tty9  /dev/ttys19 /dev/ttys30
rengy@tesr-9939:~$ sudo chmod +x /dev/ttyUSB0
```

* ttyUSB or USB port definition can be another number. Make sure you're connected with the right USB port.

Step 2

Change directory to the scripts by type "cd catkin_ws/src/tesr_ros_six_servo_robot_pkg/scripts/" and type "ls" to check permission of file in scripts. After that, we will see a list of code with the white font.

So, we must to verify permission to those codes by type
"sudo chmod +x tesr_six_dof_joint_control.py tesr_six_dof_joint_control_simulation.py TESR_dobot.py UART.py"

```
rengy@tesr-9939:~$ cd catkin_ws/src/tesr_ros_six_servo_robot_pkg/scripts/
rengy@tesr-9939:~/catkin_ws/src/tesr_ros_six_servo_robot_pkg/scripts$ sudo chmod +x tesr_six_dof_joint_
control.py tesr_six_dof_joint_control_simulation.py TESR_dobot.py UART.py
```

After this, type "ls" you will see a list of code with the green font.

```
rengy@tesr-9939:~$ cd catkin_ws/src/tesr_ros_six_servo_robot_pkg/scripts/
rengy@tesr-9939:~/catkin_ws/src/tesr_ros_six_servo_robot_pkg/scripts$ ls
__pycache__  tesr_six_dof_joint_control.py  tesr_six_dof_joint_control_simulation.py  UART.py
rengy@tesr-9939:~/catkin_ws/src/tesr_ros_six_servo_robot_pkg/scripts$ 
```

* If [Errno 13] Permission denied: '/dev/ttyUSB0' type "sudo chmod 777 /dev/ttyUSB0" if chmod +x doesn't work.

Code Explain

tesr_six_dof_joint_control.py: receive string data from microcontroller using joystick to command the motion of robot and publish value to robot model is included.

tesr_six_dof_joint_control_simulation.py: simulate and publish the joint data to robot model.

UART.py: Coding for check the string data from microcontroller.



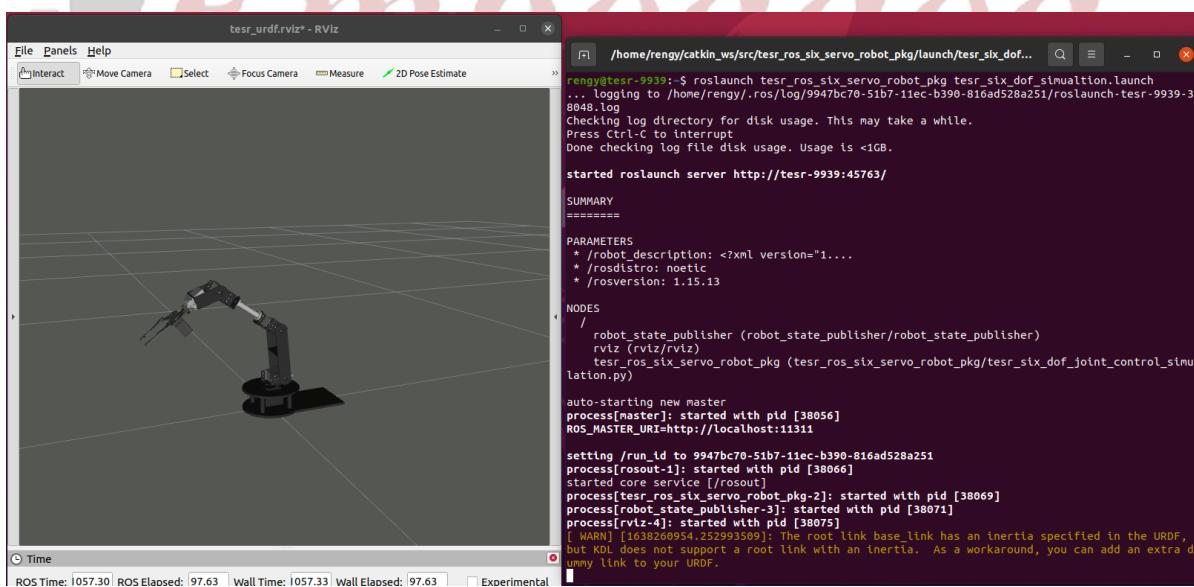
Step 3

Type "roslaunch tesr_ros_six_servo_robot_pkg tesr_six_dof.launch" for control the Robot Arm by Joystick



Or

Type "roslaunch tesr_ros_six_servo_robot_pkg tesr_six_dof_simulation.launch" to watch a motion simulation using python scripts.



=====
Enjoy



How to launch tesr_ros_six_servo_arm_moveit_config



Step 1

First thing first, make sure your PC/Laptop already connect with USB check it as same as Step1 to “Operate Joystick control and monitoring on rviz by ROS launch file”

Type “ls /dev/tty*” #/dev/ttyUSB0 should be show up or can be ttyUSB1, ttyUSB2

```
rengy@tesr-9939:~$ ls /dev/tty*
/dev/tty  /dev/tty2  /dev/tty31  /dev/tty43  /dev/tty55  /dev/ttypunkt  /dev/tty52  /dev/ttys31
/dev/tty0  /dev/tty20  /dev/tty32  /dev/tty44  /dev/tty56  /dev/tty50  /dev/ttys20  /dev/ttys4
/dev/tty1  /dev/tty21  /dev/tty33  /dev/tty45  /dev/tty57  /dev/tty51  /dev/ttys21  /dev/ttys5
/dev/tty10 /dev/tty22  /dev/tty34  /dev/tty46  /dev/tty58  /dev/tty50  /dev/ttys22  /dev/ttys6
/dev/tty11 /dev/tty23  /dev/tty35  /dev/tty47  /dev/tty59  /dev/tty511  /dev/ttys23  /dev/ttys7
/dev/tty12 /dev/tty24  /dev/tty36  /dev/tty48  /dev/tty6  /dev/ttys12  /dev/ttys24  /dev/ttys8
/dev/tty13 /dev/tty25  /dev/tty37  /dev/tty49  /dev/tty60  /dev/ttys13  /dev/ttys25  /dev/ttys9
/dev/tty14 /dev/tty26  /dev/tty38  /dev/tty5  /dev/tty61  /dev/ttys14  /dev/ttys26  /dev/ttys0
/dev/tty15 /dev/tty27  /dev/tty39  /dev/tty50  /dev/tty62  /dev/ttys15  /dev/ttys27  /dev/ttys1
/dev/tty16 /dev/tty28  /dev/tty4  /dev/tty51  /dev/tty63  /dev/ttys16  /dev/ttys28
/dev/tty17 /dev/tty29  /dev/tty40  /dev/tty52  /dev/tty7  /dev/ttys17  /dev/ttys29
/dev/tty18 /dev/tty3  /dev/tty41  /dev/tty53  /dev/tty8  /dev/ttys18  /dev/ttys3
/dev/tty19 /dev/tty30  /dev/tty42  /dev/tty54  /dev/tty9  /dev/ttys19  /dev/ttys30
rengy@tesr-9939:~$ sudo chmod +x /dev/ttyUSB0
```

* If [Errno 13] Permission denied: '/dev/ttyUSB0' type "sudo chmod +x /dev/ttyUSB0" or type "sudo chmod 777 /dev/ttyUSB0" if chmod +x doesn't work.

Step 2

We must verify the permission of python scripts in tesr_ros_six_servo_arm_moveit_config by type the command following as:

“cd ws_moveit/src/tesr_ros_six_servo_arm_moveit_config/scripts” #go to file’s path directory.
“ls” #check the file permission if list of file are white or green font.

```
rengy@tesr-9939:~$ cd ws_moveit/src/tesr_ros_six_servo_arm_moveit_config/scripts
rengy@tesr-9939:~/ws_moveit/src/tesr_ros_six_servo_arm_moveit_config/scripts$ ls
tesr_ros_moveit_to_six_servoarm.py
```

The list with white font is means the file is not verify yet. So, we must verify it by chmod.

“sudo chmod +x *” #After chmod compile list should be turn to green font.

```
rengy@tesr-9939:~/ws_moveit/src/tesr_ros_six_servo_arm_moveit_config/scripts$ sudo chmod +x *
[sudo] password for rengy:
rengy@tesr-9939:~/ws_moveit/src/tesr_ros_six_servo_arm_moveit_config/scripts$ ls
tesr_ros_moveit_to_six_servoarm.py
```

Code Explain

tesr_ros_moveit_to_six_servoarm.py: receive the joint data from moveit and use it to control robot arm to move follow moveit’s path planning.



Step 3

Next, Let's launch the file from package by type:

```
"roslaunch tesr_ros_six_servo_arm_moveit_config tesr_six_servo_moveit_sync_pose.launch"
```

```
rengy@tesr-9939:~$ rosrun roslaunch tesr_ros_six_servo_arm_moveit_config tesr_six_servo_moveit_sync_pose.launch
... logging to /home/rengy/.ros/log/bddee3fc0-cf0e-11ec-be8f-310e46885196/roslaunch-tesr-9939-75494.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.1.20:40611/

SUMMARY
=====

PARAMETERS
* /joint_state_publisher/source_list: ['move_group/fake...
* /move_group/allow_trajectory_execution: True
* /move_group/capabilities:

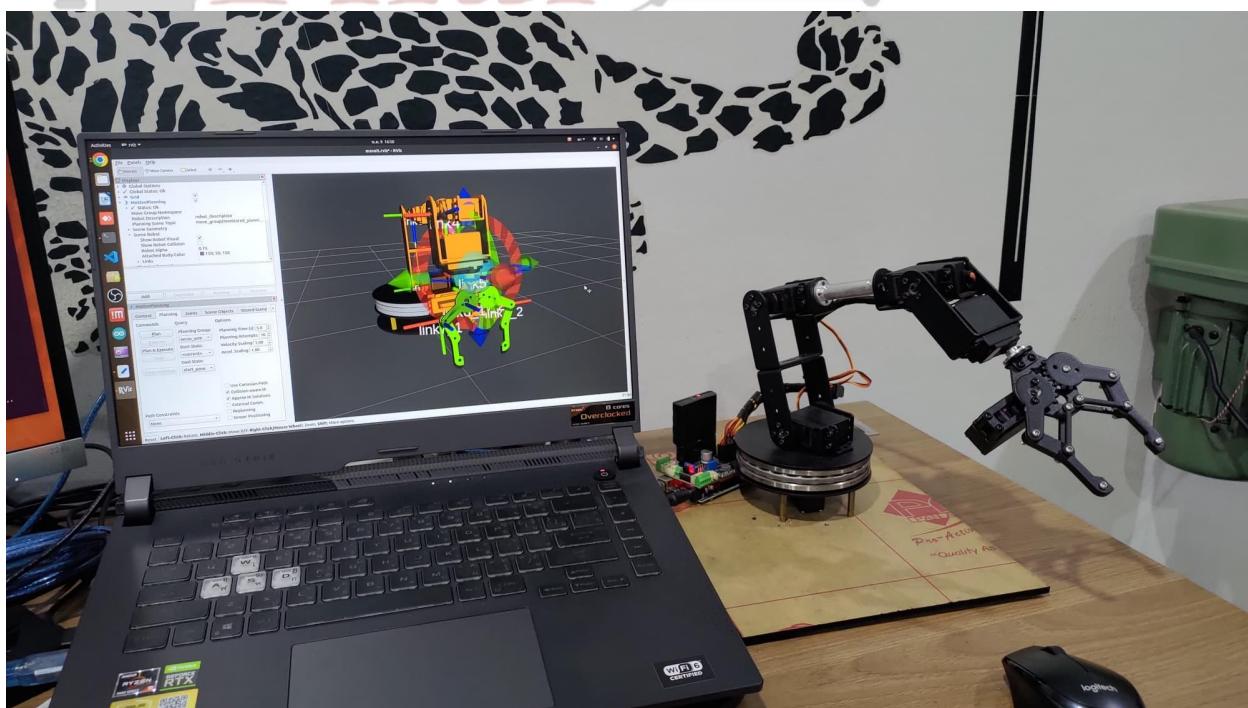
[ INFO] [1652089002.811261713]: MoveGroup context using planning plugin ompl_interface/OMPLPlanner
[ INFO] [1652089002.811277503]: MoveGroup context initialization complete

You can start planning now!

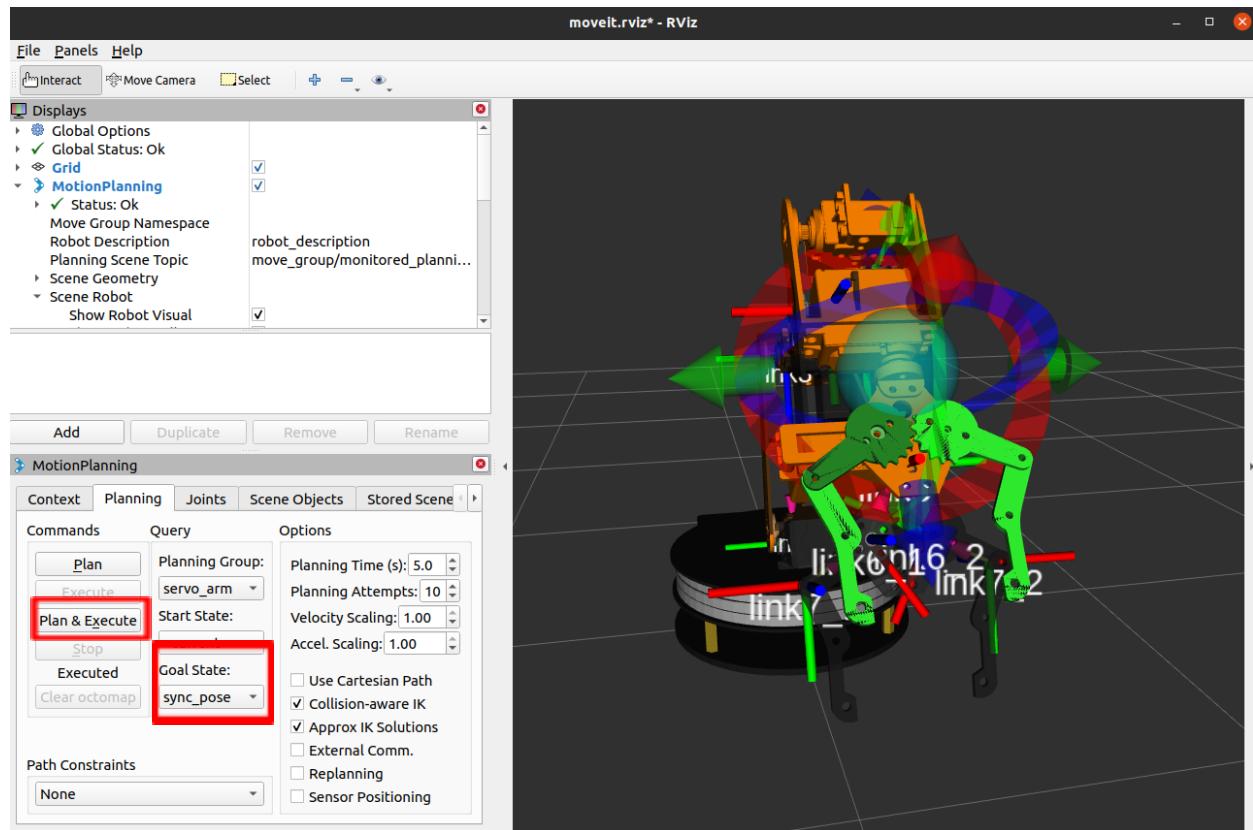
[ INFO] [1652089005.304289819]: Loading robot model 'six_servo'...
[ INFO] [1652089005.750240847]: Starting planning scene monitor
[ INFO] [1652089005.751492885]: Listening to '/move_group/monitored_planning_scene'
[ INFO] [1652089006.441400804]: Constructing new MoveGroup connection for group 'servo_arm' in namespace ''
[ INFO] [1652089007.430465720]: Ready to take commands for planning group servo_arm.
```

After the log info show “Ready to take command for planning group servo_arm” that's mean you're ready to use moveit for path planning of six servo arm.

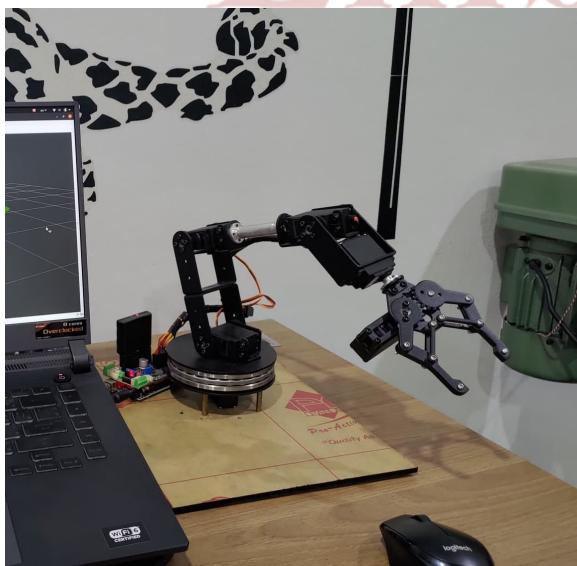
But when you start the launch file robot arm's pose and moveit will not sync.



So, Let's sync first pose of robot arm and moveit by select "sync_pose" in Goal State and press "Plan & Execute"



After Plan & Execute robot arm's pose and moveit will sync. So, You can use it properly from now on.



BEFORE

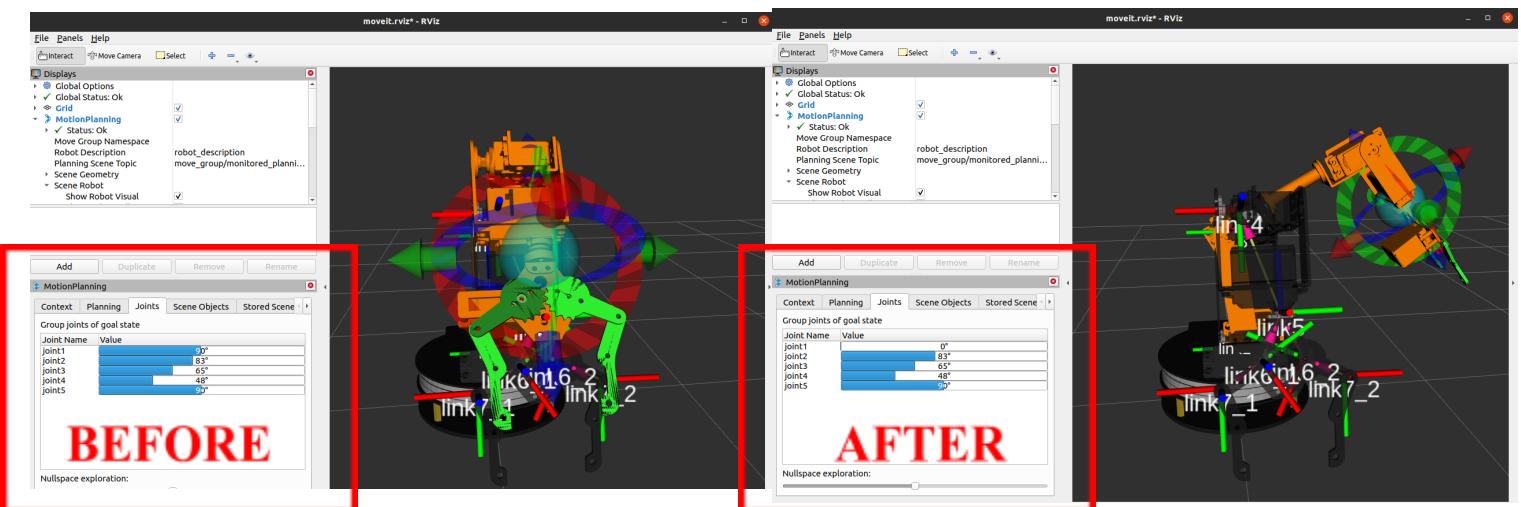
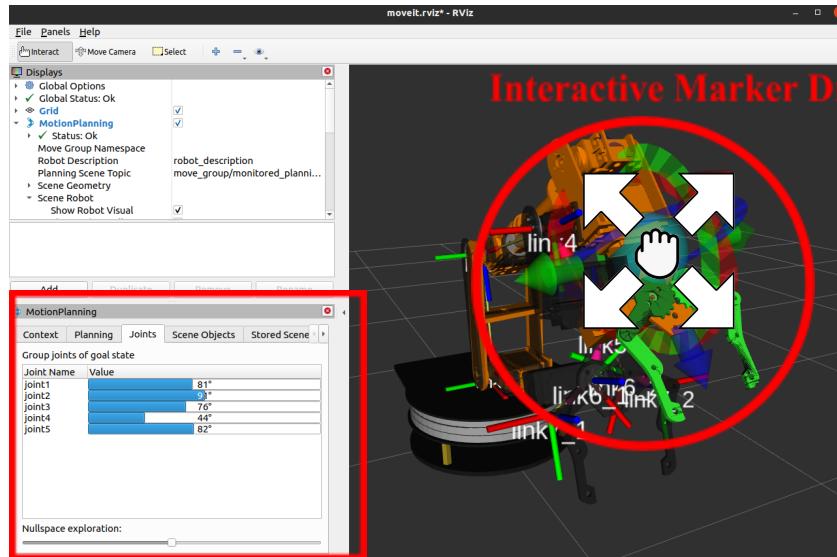


AFTER

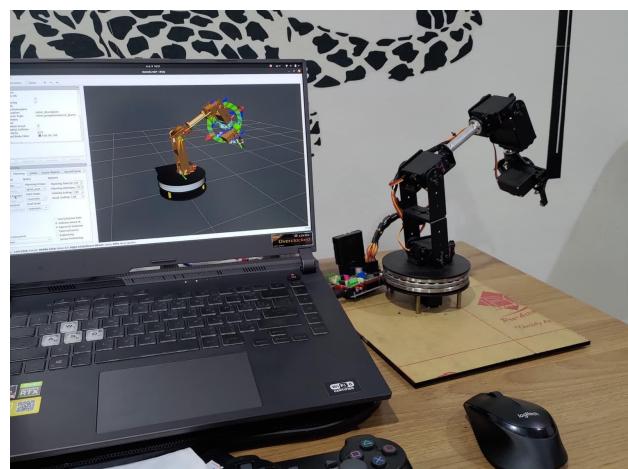


Now, you can use Interactive marker or Joint slider bars to move the Goal state to use as robot's goal

**Joint
Slide Bar**



BEFORE



AFTER

