PRE-REQUISITES

- Download UBUNTU 20.04 and Install on your PC
- Install ROS Noetic on Remote PC:
  $ sudo apt update
  $ sudo apt upgrade
  $ wget https://raw.githubusercontent.com/ROBOTIS-GIT/robotis_tools/master/install_ros_noetic.sh
  $ chmod 755 ./install_ros_noetic.sh
  $ bash ./install_ros_noetic.sh
- Install Dependent ROS Packages:
  $ 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
- Install Turtlebot3 Packages:
  $ sudo apt install ros-noetic-dynamixel-sdk
  $ sudo apt install ros-noetic-turtlebot3-msgs
  $ sudo apt install ros-noetic-turtlebot3
- $ sudo apt-get update
- $ sudo apt-get upgrade
- Install Simulation Package:
  $ cd ~/catkin_ws/src/
  $ git clone -b noetic-devel https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git
  $ cd ~/catkin_ws && catkin_make
- Install GIT in UBUNTU:
  $ sudo apt-get update
  $ sudo apt-get install git-all
- Download the repository from an organization:
  $ git clone git@github.com:THD-autonomous-system/ros-2023-thd.git
- Install Docker:
  $ cd ros-2023-thd/docker
  $ ./install_docker.bash *// add -n in the end if you have NVidia graphics card.*
  //to Build docker
  $ sudo ./build_docker.sh *// add -n in the end if you have NVidia graphics card.*
  //to run docker:
  $ sudo ./run_docker.sh *// add -n in the end if you have NVidia graphics card.*

Network Configuration:

Make sure both PC & Turtlebot should be connected to same WIFI network.
// go to the netplan directory to .yaml file
$ cd /media/$USER/writable/etc/netplan
$ sudo nano 50-cloud-init.yaml

```
network:
  version: 2
  renderer: networkd
  ethernets:
    eth0:
      dhcp4: yes
      dhcp6: yes
      optional: true
  wifis:
    wlan0:
      dhcp4: yes
      dhcp6: yes
      access-points:
        WIFI_SSID:
          password: WIFI_PASSWORD
```

//When the editor is opened, replace the WIFI_SSID and WIFI_PASSWORD with your wifi SSID and password.
//Save the file with Ctrl+S and exit with Ctrl+X.
$ sudo reboot *//after rebooting turtlebot will connect to the desired WIFI as of PC.*

For TURTLEBOT

- $ ifconfig
  OR
- $ hostname – I *//it will give IP address of turtlebot copy it.*
- $ nano ~/.bashrc *// a window will appear where you have to go to the end of the line and add two more lines which are as follows:*

  export ROS_MASTER_URI=https://{*IP address of PC*}:11311
  export ROS_HOSTNAME={*IP address of turtlebot}*

- $ source ~/.bashrc

For PC

- $ ifconfig
  OR
- $ hostname – I *//it will give IP address of PC copy it.*
- $ sudo ./into_docker.sh *//opens terminal inside docker*
- $ roscore (New Terminal in docker) //
- $ nano ~/.bashrc (New Terminal in docker)*// go to the end of the line and add two more lines:*

  export ROS_MASTER_URI=https://{*IP address of PC*}:11311
  export ROS_HOSTNAME={*IP address of PC}*

  $ source ~/.bashrc

Try pinging from PC to Turtlebot and vice versa:

- $ ping {*IP address}*

In your PC:

- $ ssh ubuntu@{*IP address of turtlebot*}
  // It will ask for turtlebot ubuntu password, enter and it will open the terminal of turtlebot ubuntu.
  //with this you can now disconnect your turtlebot with monitor and keyboard and start simulation with the master PC.
- Install the LDS-02 driver and update TurtleBot3 package in the same terminal.
  $ sudo apt update
  $ sudo apt install libudev-dev
  $ cd ~/catkin_ws/src
  $ git clone -b develop https://github.com/ROBOTIS-GIT/ld08_driver.git
  $ cd ~/catkin_ws/src/turtlebot3 && git pull
  $ rm -r turtlebot3_description/ turtlebot3_teleop/ turtlebot3_navigation/ turtlebot3_slam/ turtlebot3_example/
  $ cd ~/catkin_ws && catkin_make
- Export the LDS_MODEL to the bashrc file. Depending on your LDS model, use LDS-01 or LDS-02.
  $ echo 'export LDS_MODEL=LDS-02' >> ~/.bashrc
  $ source ~/.bashrc
  //proceed with Bringup