

Name : Al Amin Hossain Nayem

## Labwork 5

First, I installed Gazebo Harmonic in my Ubuntu system. Then I created a workspace and built a custom robot model using SDF.

A terminal window with a dark purple background. The prompt is 'al\_amin@al-amin:~\$'. The commands entered are 'mkdir -p ~/ros2\_ws/src', 'cd ~/ros2\_ws', 'colcon build', and 'source install/setup.bash'. The output shows 'Summary: 0 packages finished [0.34s]'. On the left side of the terminal, there is a green bar with '9 AM' and a blue checkmark, and a grey bar with a microphone icon.

```
al_amin@al-amin:~$ mkdir -p ~/ros2_ws/src
al_amin@al-amin:~$ cd ~/ros2_ws
al_amin@al-amin:~$ colcon build
al_amin@al-amin:~$ source install/setup.bash

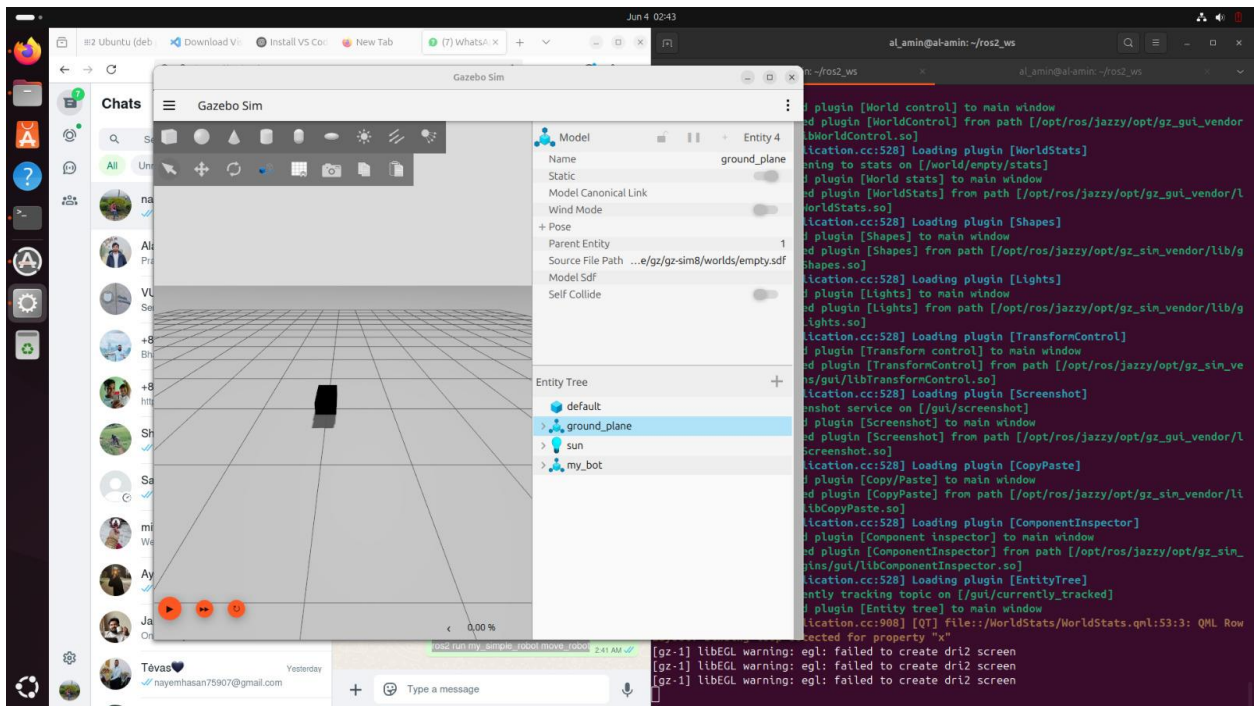
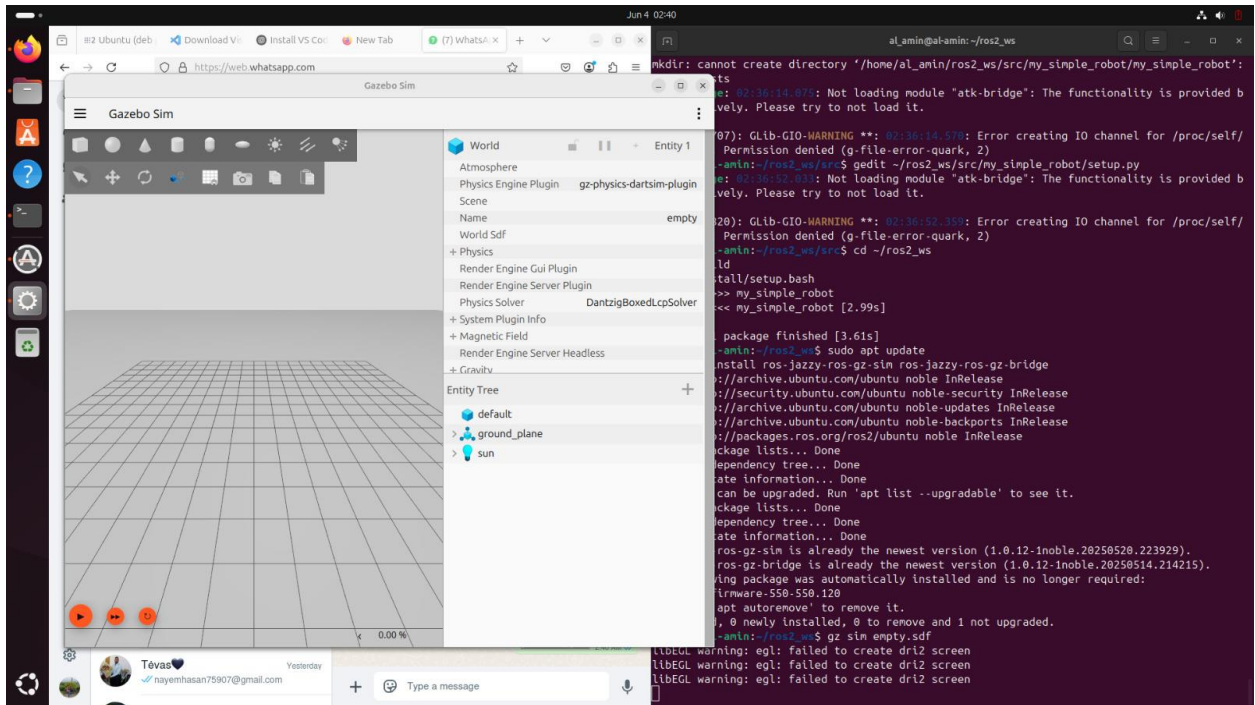
Summary: 0 packages finished [0.34s]
```

Name : Al Amin Hossain Nayem  
Labwork 5

```
al_amin@al-amin:~  
Setting up libpoconet80t64:amd64 (1.11.0-4.1build2) ...  
Setting up libkrb5-dev:amd64 (1.20.1-6ubuntu2.6) ...  
Setting up libpocodataodbc80t64:amd64 (1.11.0-4.1build2) ...  
Setting up qml-module-qtquick-extras:amd64 (5.15.13-1) ...  
Setting up libpocodatasqlite80t64:amd64 (1.11.0-4.1build2) ...  
Setting up libzmq3-dev:amd64 (4.3.5-1build2) ...  
Setting up libpocojwt80t64:amd64 (1.11.0-4.1build2) ...  
Setting up cppzmq-dev:amd64 (4.10.0-1build1) ...  
Setting up libpocoutil80t64:amd64 (1.11.0-4.1build2) ...  
Setting up libpocomongodb80t64:amd64 (1.11.0-4.1build2) ...  
Setting up libpoconetssl80t64:amd64 (1.11.0-4.1build2) ...  
Setting up libavfilter-dev:amd64 (7:6.1.1-3ubuntu5) ...  
Setting up libavdevice60:amd64 (7:6.1.1-3ubuntu5) ...  
Setting up libpocoredis80t64:amd64 (1.11.0-4.1build2) ...  
Setting up libavdevice-dev:amd64 (7:6.1.1-3ubuntu5) ...  
Setting up libpoco-dev:amd64 (1.11.0-4.1build2) ...  
Setting up ros-jazzy-gz-ogre-next-vendor (0.0.5-1noble.20250424.111422) ...  
Setting up ros-jazzy-gz-common-vendor (0.0.8-1noble.20250424.110920) ...  
Setting up rake (13.0.6-3) ...  
Setting up libruby:amd64 (1:3.2-ubuntu1) ...  
Setting up ruby-sdbm:amd64 (1.0.0-5build4) ...  
Setting up libruby3.2:amd64 (3.2.3-1ubuntu0.24.04.5) ...  
Setting up ruby3.2 (3.2.3-1ubuntu0.24.04.5) ...  
Setting up ruby (1:3.2-ubuntu1) ...  
Setting up ruby-rubygems (3.4.20-1) ...  
Setting up ros-jazzy-gz-tools-vendor (0.0.6-1noble.20250424.121722) ...  
Setting up ros-jazzy-sdformat-vendor (0.0.9-1noble.20250520.201232) ...  
Setting up ros-jazzy-gz-plugin-vendor (0.0.5-1noble.20250424.121915) ...  
Setting up ros-jazzy-gz-msgs-vendor (0.0.6-1noble.20250424.121919) ...  
Setting up ros-jazzy-gz-physics-vendor (0.0.6-1noble.20250520.202118) ...  
Setting up ros-jazzy-gz-rendering-vendor (0.0.6-1noble.20250424.132237) ...  
Setting up ros-jazzy-gz-transport-vendor (0.0.6-1noble.20250424.132310) ...  
Setting up ros-jazzy-gz-fuel-tools-vendor (0.0.6-1noble.20250424.132253) ...  
Setting up ros-jazzy-gz-sensors-vendor (0.0.6-1noble.20250520.202411) ...  
Setting up ros-jazzy-gz-gui-vendor (0.0.5-1noble.20250424.134842) ...  
Setting up ros-jazzy-ros-gz-bridge (1.0.12-1noble.20250514.214215) ...  
Setting up ros-jazzy-gz-sim-vendor (0.0.8-1noble.20250520.213730) ...  
Setting up ros-jazzy-ros-gz-sim (1.0.12-1noble.20250520.223929) ...  
Processing triggers for man-db (2.12.0-4build2) ...  
Processing triggers for install-info (7.1-3build2) ...  
Processing triggers for fontconfig (2.15.0-1.1ubuntu2) ...  
Processing triggers for libc-bin (2.39-0ubuntu8.4) ...  
al_amin@al-amin:~$
```

# Name : Al Amin Hossain Nayem

## Labwork 5

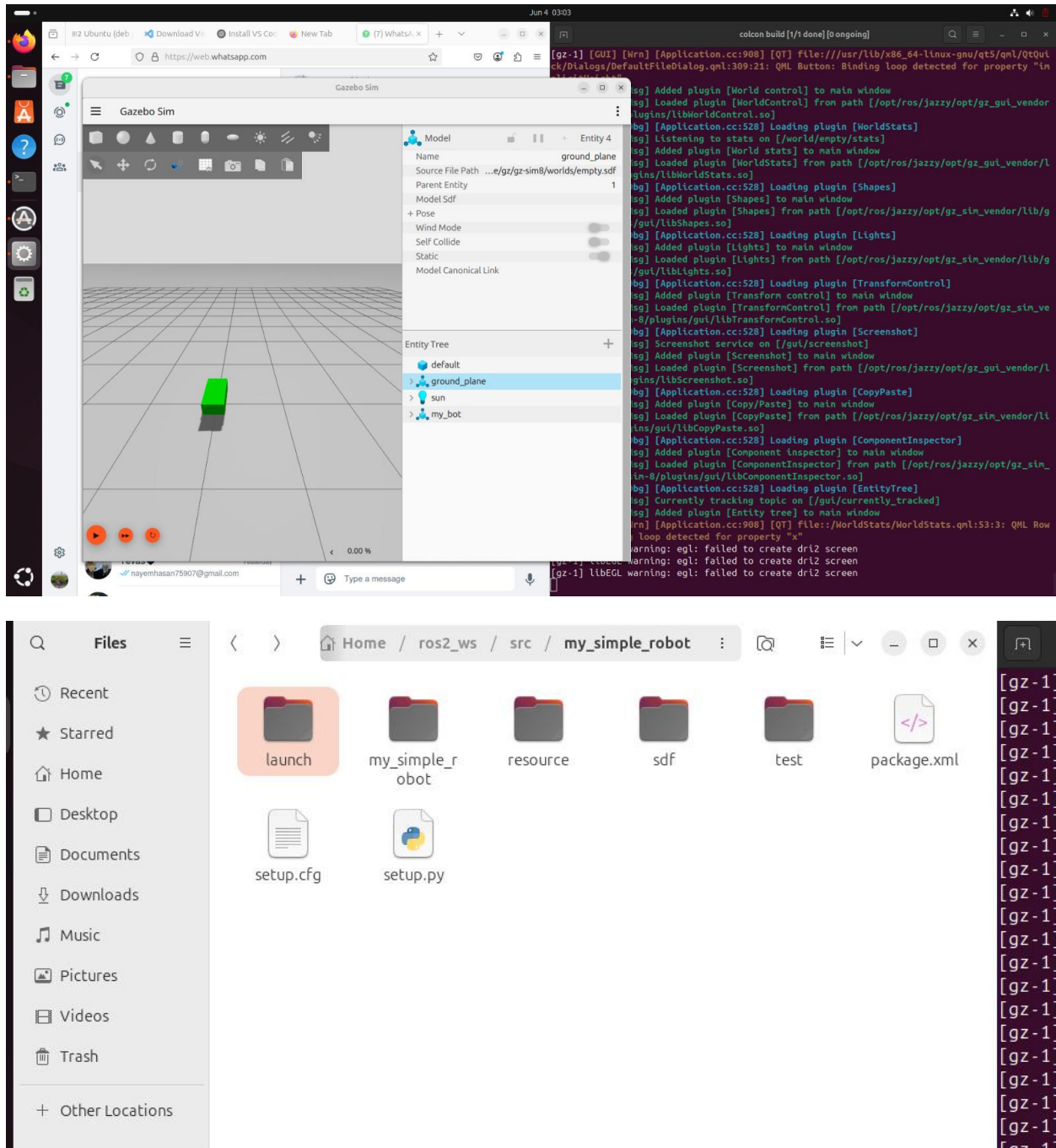


The model is a simple box with dimensions and mass defined, and I set its color to green using the ambient and diffuse material properties.



## Name : Al Amin Hossain Nayem

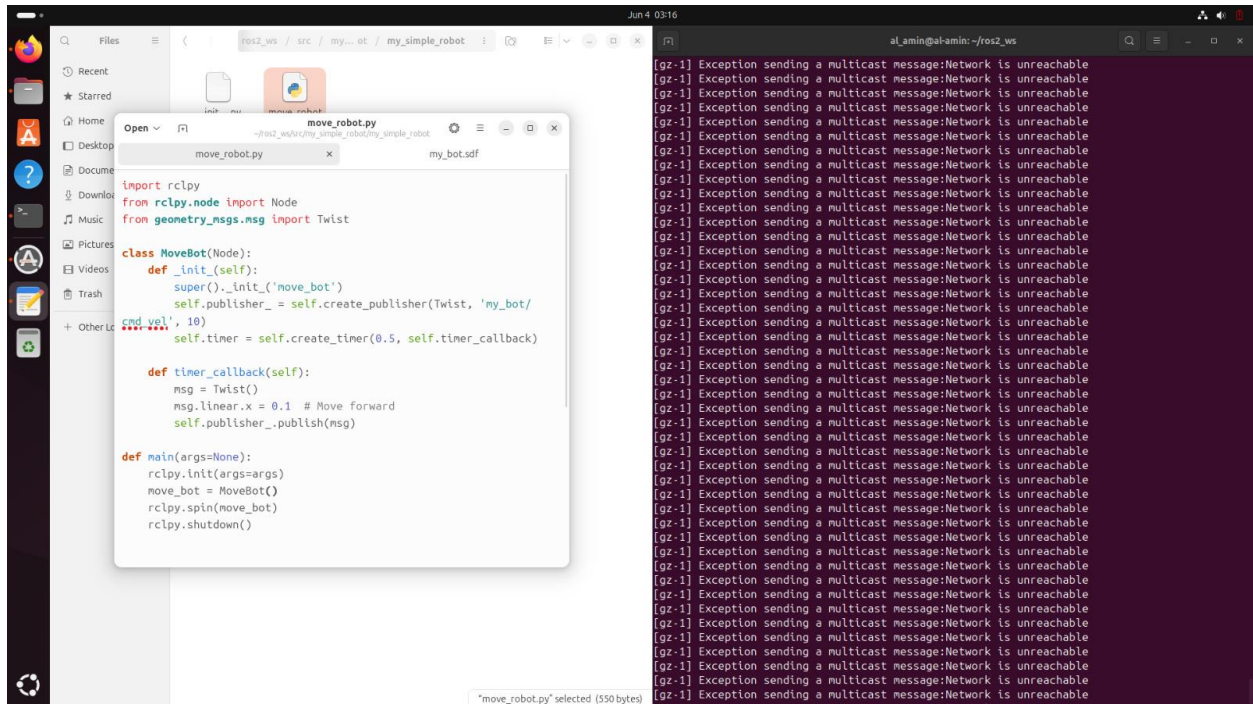
### Labwork 5



After that, I created a launch file to spawn the robot in an empty Gazebo world. For movement, I wrote a Python ROS2 node that publishes velocity commands to the topic /my\_bot/cmd\_vel. I used a timer callback to send forward velocity (0.1 m/s) every 0.5 seconds. The robot moves forward in simulation, and the entire system works through

Name : Al Amin Hossain Nayem  
Labwork 5

## ROS2 publisher-subscriber communication.



The screenshot displays a ROS2 development environment. On the left, a file explorer shows the directory structure of a workspace named 'my\_simple\_robot'. The 'src' directory contains two files: 'move\_robot.py' and 'my\_bot.sdf'. The 'move\_robot.py' file is open in a code editor, showing the following Python code:

```
import rclpy
from rclpy.node import Node
from geometry_msgs.msg import Twist

class MoveBot(Node):
    def __init__(self):
        super().__init__('move_bot')
        self.publisher_ = self.create_publisher(Twist, 'my_bot/cmd_vel', 10)
        self.timer = self.create_timer(0.5, self.timer_callback)

    def timer_callback(self):
        msg = Twist()
        msg.linear.x = 0.1 # Move forward
        self.publisher_.publish(msg)

def main(args=None):
    rclpy.init(args=args)
    move_bot = MoveBot()
    rclpy.spin(move_bot)
    rclpy.shutdown()
```

On the right, a terminal window shows the output of the node. It displays a series of exceptions: 'Exception sending a multicast message: Network is unreachable'. This indicates that the node is attempting to publish messages but is unable to reach the network. The terminal output is repeated for many iterations, suggesting a continuous loop of failed publication attempts.

## Conclusion

In this task, I learned how ROS2 interacts with Gazebo using the `ros_gz_bridge` and simulation launch files. I now understand the process of writing a robot SDF file, spawning the model, and sending motion commands using a Python node. I faced some small errors with node initialization and SDF material, but I fixed those and everything worked fine in the end. This project helped me understand the complete flow of robot simulation in ROS2.