

**Control and Simulation Tasks Report**

**By**

**Abdullah Abdelgalel Abouda**

Table of Contents

[**Introduction** 3](#_Toc158751828)

[**Task I - Physical Modeling** 3](#_Toc158751829)

[**Solution Approach** 4](#_Toc158751830)

[**Mechanical Translation Equations:** 4](#_Toc158751831)

[**Simulink Model:** 5](#_Toc158751832)

[**Simulink Model Output:** 7](#_Toc158751833)

[**Task II- System Identification** 8](#_Toc158751834)

[**Solution Approach** 8](#_Toc158751835)

[**Task III- Linux** 11](#_Toc158751836)

[**Installation and Testing** 11](#_Toc158751837)

List of Figures and Tables

[Table 1: System Parameters 3](#_Toc158751862)

[Figure 1: Suspension System Model 3](#_Toc158751863)

[Figure 2: Integrator Blocks 5](#_Toc158751864)

[Figure 3: Subtract Blocks, Gain Blocks, and Scope 5](#_Toc158751865)

[Figure 4: Subtract, Gain, and Step Input Blocks 5](#_Toc158751866)

[Figure 5: Final Simulink Model 6](#_Toc158751867)

[Figure 6: Simulation Output 7](#_Toc158751868)

[Figure 7: Sorting Input/Output data from the file to single variables 8](#_Toc158751869)

[Figure 8: PID Tunner APP Interface 9](#_Toc158751870)

[Figure 9: I/O data Window 9](#_Toc158751871)

[Figure 10: PID Solver Window 9](#_Toc158751872)

[Figure 11: Plant Identification Progress Window. 10](#_Toc158751873)

[Figure 12: Plant after fitting Process showing the Values of K and T1 in the bottom right Corner. 10](#_Toc158751874)

[Figure 14: Plant 1 Process Model 10](#_Toc158751875)

[Figure 13: Plant 1 Variable in Workspace 10](#_Toc158751876)

[Figure 15: Username Creation Window 11](#_Toc158751877)

[Figure 16: Some Linux Terminal Commands 11](#_Toc158751878)

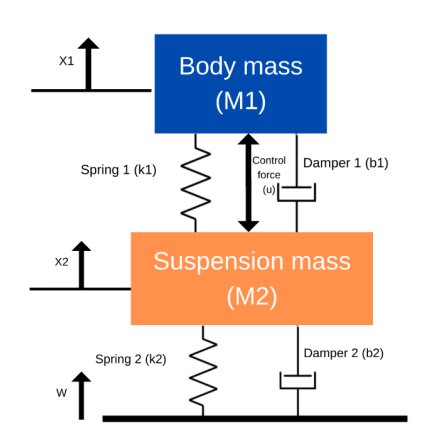
[Figure 17: Ubuntu OS with some Terminal Commands 12](#_Toc158751879)

# **Introduction**

In the realm of engineering and technology, ensuring the reliability and safety of systems is crucial. Control and simulation tools are incredibly valuable for analyzing, designing, and improving complex systems across various fields, including aerospace, automotive, robotics, and manufacturing. This report delves into the specifics of each task, outlining the methods employed and our approach to them.

# **Task I - Physical Modeling**

***Required:*** *Model The Suspension System in Figure 1 Using Simulink.*

***Expected Output****: The Difference Between X1 and X2 over Time with specified Inputs.*

***Assumptions:***

* ***W*** *is the ground profile (i.e., road disturbance). If W = 0, this means that the road is flat and there are no disturbances. It could be modeled as* ***step input.***
* *The control force,* ***u****, could be modeled as a* ***step input***
* *parameters to run the simulation:*

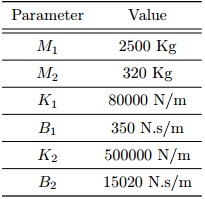
******

Table 1: System Parameters

Figure 1: Suspension System Model

## **Solution Approach**

We begin by creating the mechanical translation equations that describe our suspension system. Next, we'll use these equations to build blocks in a Simulink model. Lastly, we'll fine-tune the parameters and run simulations to generate and analyze the results.

### **Mechanical Translation Equations:**

We'll formulate equations for all systems that influence a particular mass. These equations will be in the form of differential equations in the **time domain.**

**Note: ,**

***Equation 1*** *(Influencing Body Mass or M1)*

***Equation 2*** *(Influencing Suspension Mass or M2)*

**Rearrange Equations 1 & 2 to make both in the L.H.S**

***Equation 1*** *(Influencing Body Mass or M1)*

***Equation 2*** *(Influencing Suspension Mass or M2)*

### **Simulink Model:**

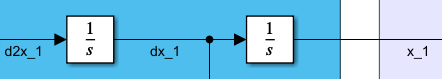
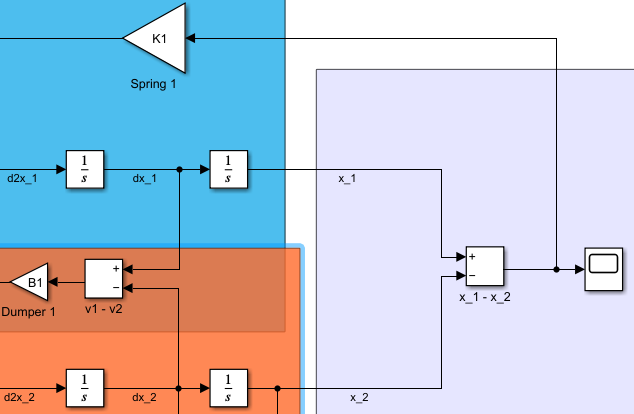
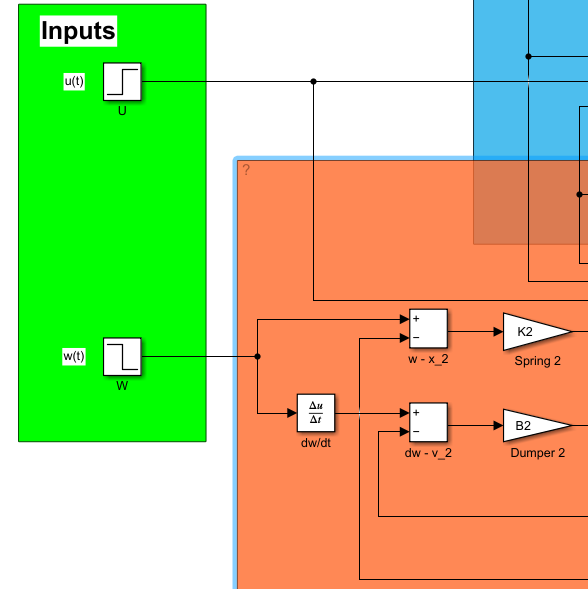
 Once the equations are written, we'll begin constructing our model blocks. Since we're aiming for and outputs with u and W as inputs, we'll start from and work our way to by adding integrator blocks, leveraging the equations for to connect the blocks and create the final model.

Figure 2: Integrator Blocks



Following that, we'll compute the difference between the signals of and attaching a scope to Visualize the Output Signal, as well as between and .Then, we'll extract nodes from these signals and apply the gains of their respective systems, such as Spring 1and Damper 1 through multiplication.

Figure 3: Subtract Blocks, Gain Blocks, and Scope



Following that, we'll compute the difference between the signals of and , as well as between and . To derivewe will add a derivative block. Input blocks will be added with the given parameters outlined in the Recruitment Report; all *U* parameters will be set to zero, and for *W* parameters, the step time will be 5 seconds with a final value of -0.1. Then, we'll extract nodes from these signals and apply the gains of their respective systems, such as Spring 2 and Damper 2 through multiplication.

Figure 4: Subtract, Gain, and Step Input Blocks

In conclusion, we'll link these blocks with sum blocks and apply the gains through multiplication to compute and thereby finalizing the model by Using Equations 1 and 2.

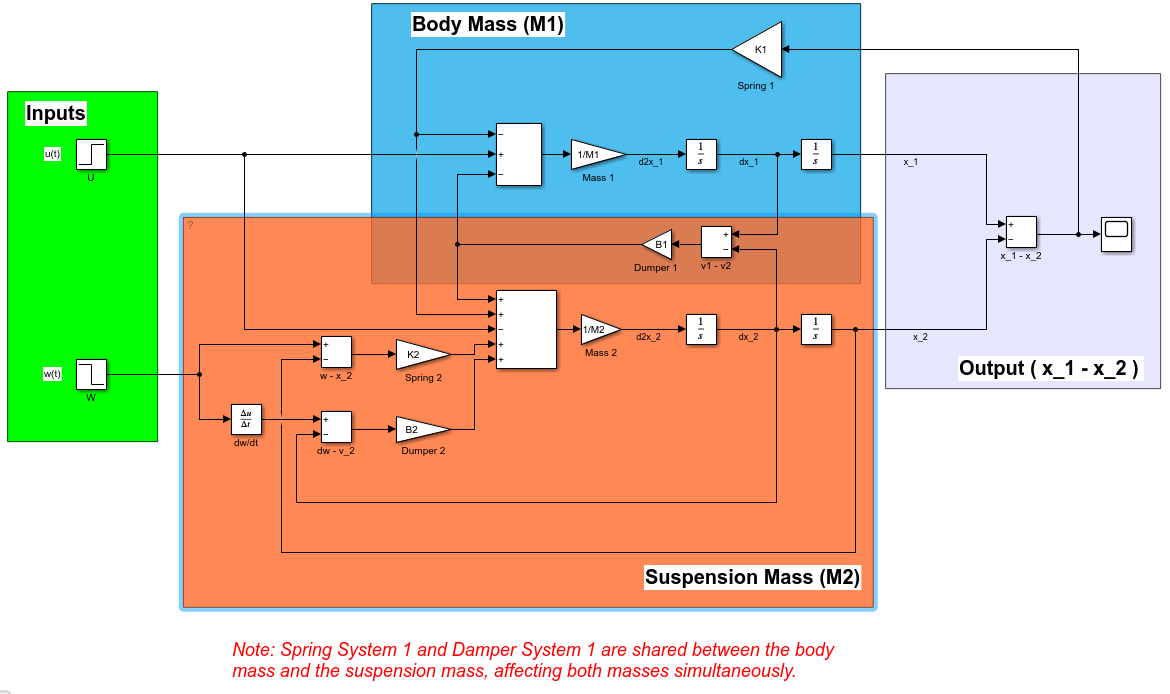


Figure 5: Final Simulink Model

Each gain is assigned a variable name corresponding to the parameters of the physical model, such as *M1, K1, B1*, and so on. All these parameters listed in Table 1 are stored in a MATLAB file named "Physical\_Modeling\_Parameters," which is linked with the Simulink model file named "Landing\_Gear\_Suspension\_System." If you want to test the same system with different parameters, you can modify the parameters in the MATLAB file named "Physical\_Modeling\_Parameters."

### **Simulink Model Output:**

Figure 6: Simulation Output

After the landing gear suspension system impacts the 10 cm pothole at seconds, we notice that the resulting disturbance causes the distance between the body mass and suspension mass , or , to oscillate between values of and . These oscillations decay over time and return to zero after approximately seconds.

# **Task II- System Identification**

***Required:*** *For the Given Input/Output data in the file “RampTest\_60hz.csv” it is required to obtain the values of and which describes the system.*

***Assumptions:***

* *The System has only a gain and one pole with no zeros*
* *The System can be modeled as the following:*

**Solution Approach**

First, we import the data file into MATLAB. Then, we slice the data into inputs and outputs, saving them as variables. Next, we import these variables into the PID Tuner tool.

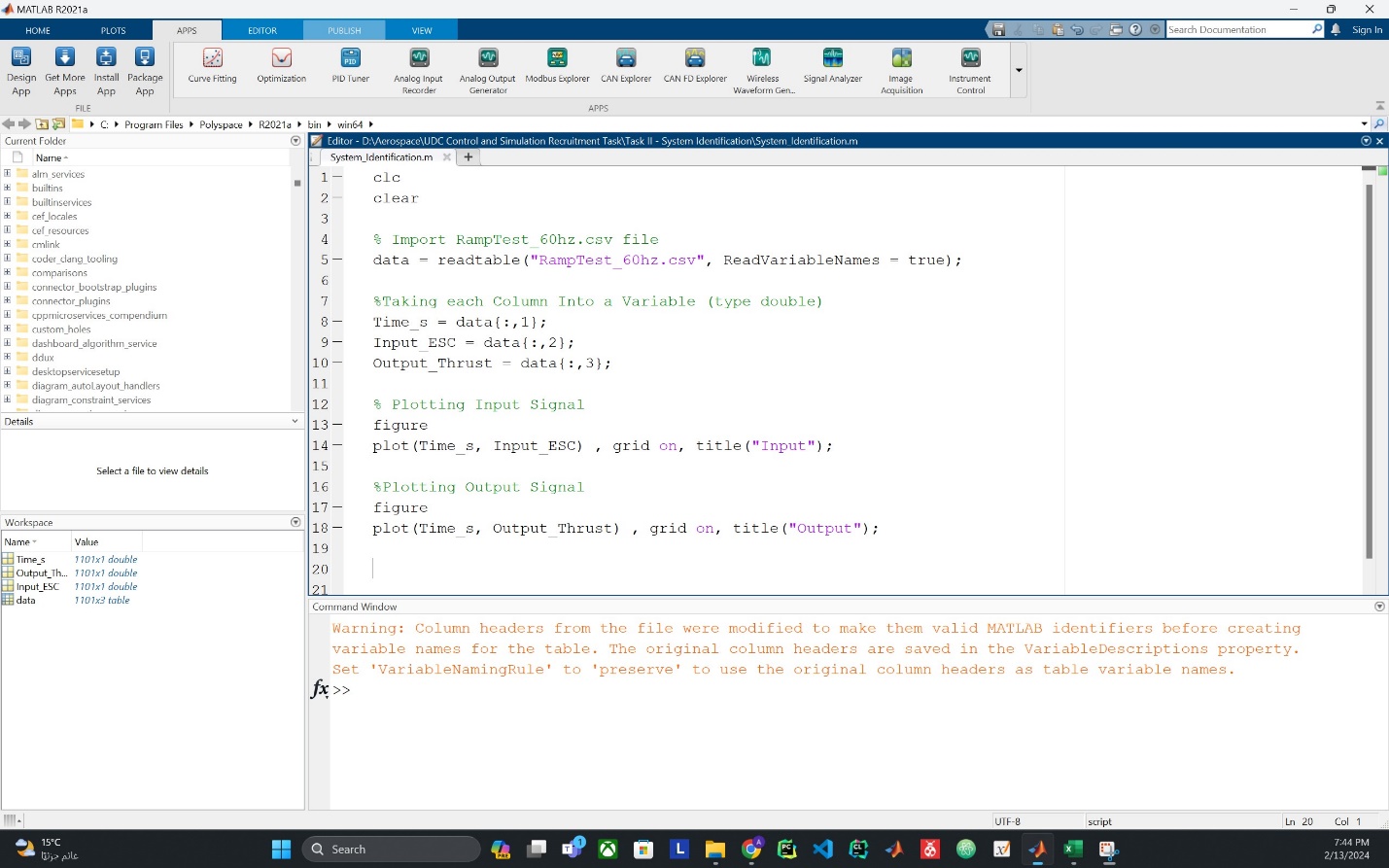


Figure 7: Sorting Input/Output data from the file to single variables

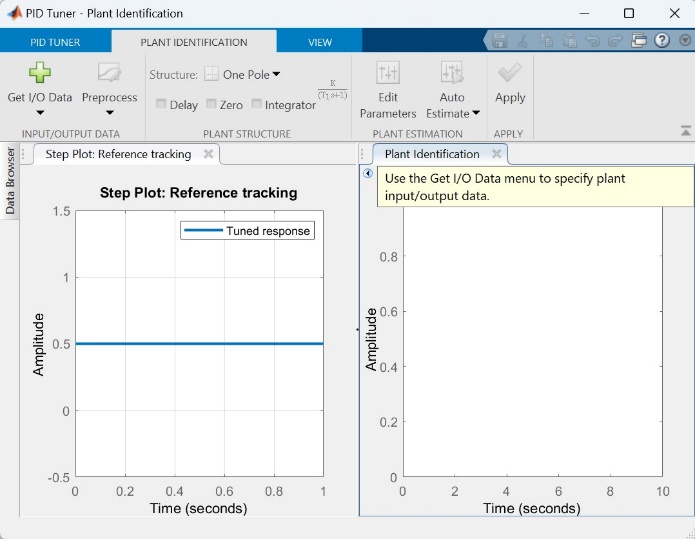
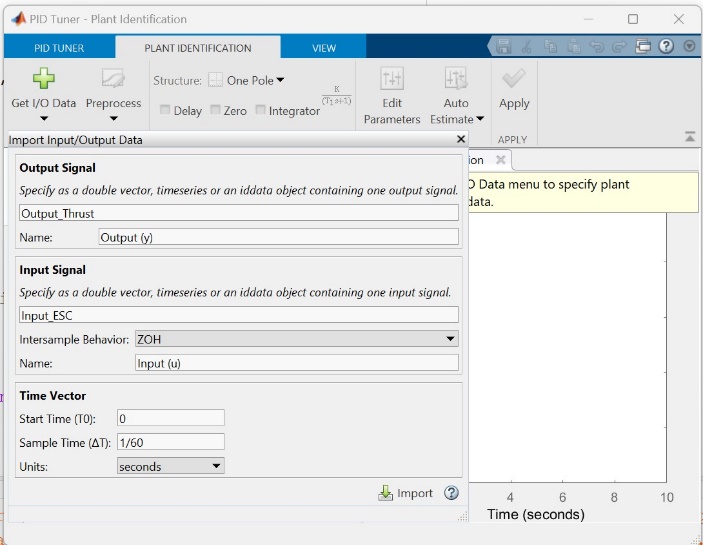
T To import the input/output variables, we access the PID Tuner Tool and designate a new Plant (System). Next, we select "Get I/O Data" followed by "Arbitrary I/O Data" to bring in our data variables, aiding in the identification of the Plant (System).

Figure 8: PID Tunner APP Interface

A window will appear prompting you to enter the names of the output and input variables that are already present in the Workspace.

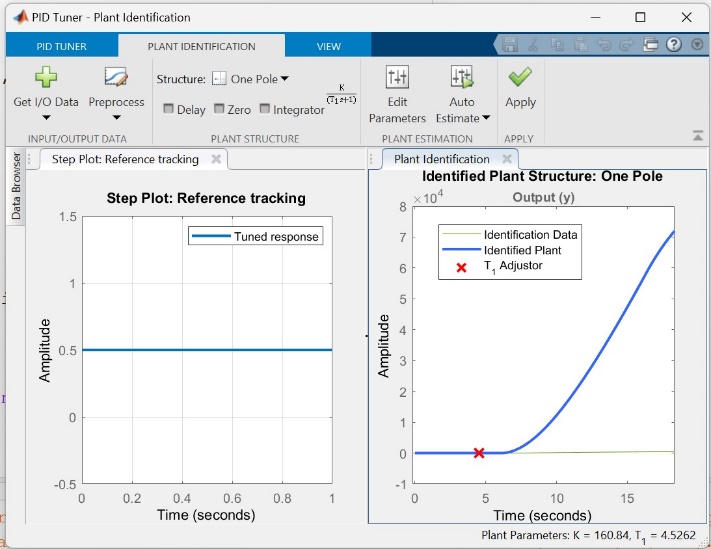


Figure 9: I/O data Window

Once you return to the PID Tuner tool window, the plant output data will be displayed as a green line, while the solver, identifying the plant, will be represented by a blue line. We can edit The Transfer Function Structure from the Plant Structure Section. To achieve the best fitting, we'll utilize the auto-estimate button, employing an optimization algorithm to obtain the optimal fit for the output data.

Figure 10: PID Solver Window

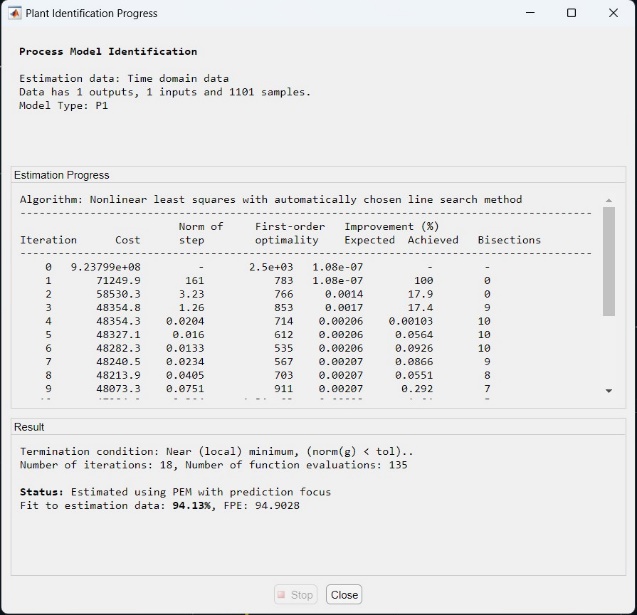
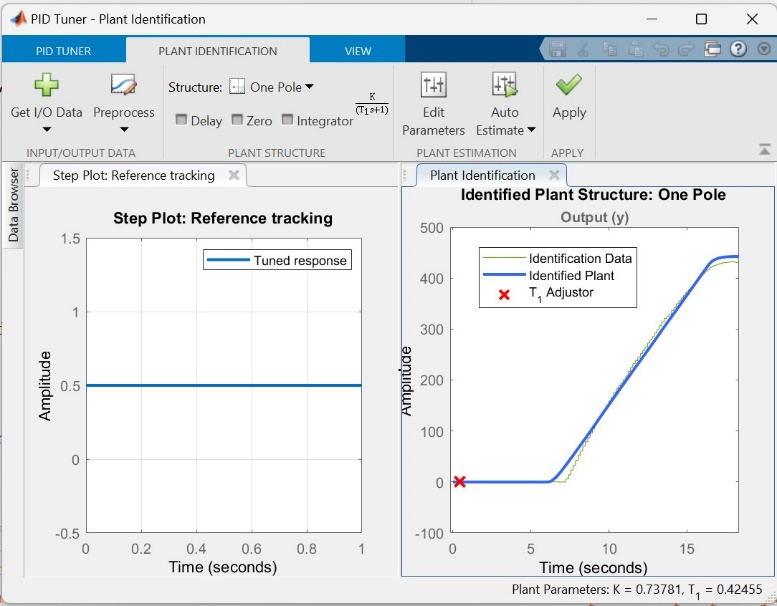
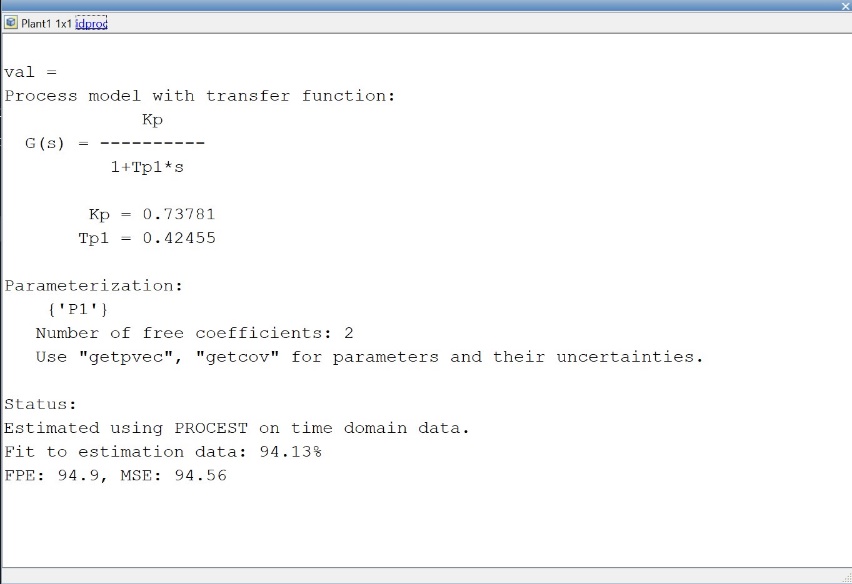


Figure 11: Plant Identification Progress Window.

Figure 12: Plant after fitting Process showing the Values of K and T1 in the bottom right Corner.

 The results of the System Identification process using the PID Tuner Tool yielded and . These analyzed plant values can be saved in the workspace (see Figure 13). Additionally, the plant parameters can be viewed via the "Plant -> Inspect" option (see Figure 14). Note since MATLAB files will have new workspace at every run a workspace variable file “System\_Identification\_Workspace\_Variables.mat” is saved to have access to all results generated in this task.

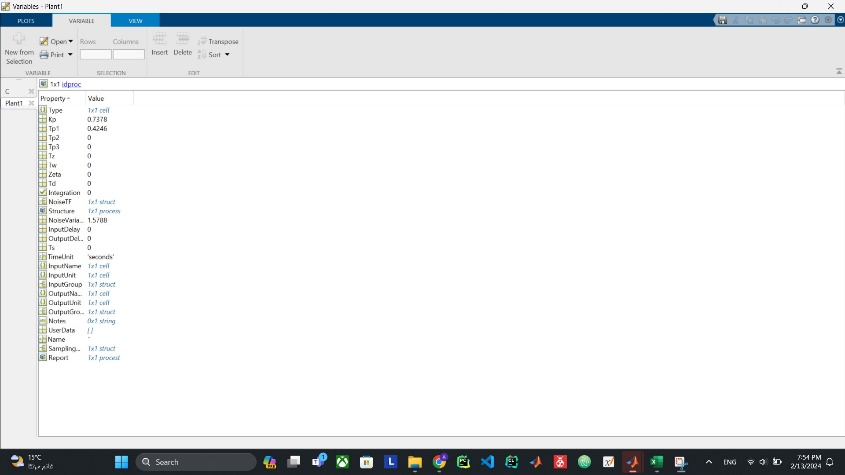


Figure 14: Plant 1 Process Model

Figure 13: Plant 1 Variable in Workspace

# **Task III- Linux**

***Required:*** *Install Ubuntu Linux version 20.04.6 on WSL2(Windows subsystem for Linux),.*

**Installation and Testing**

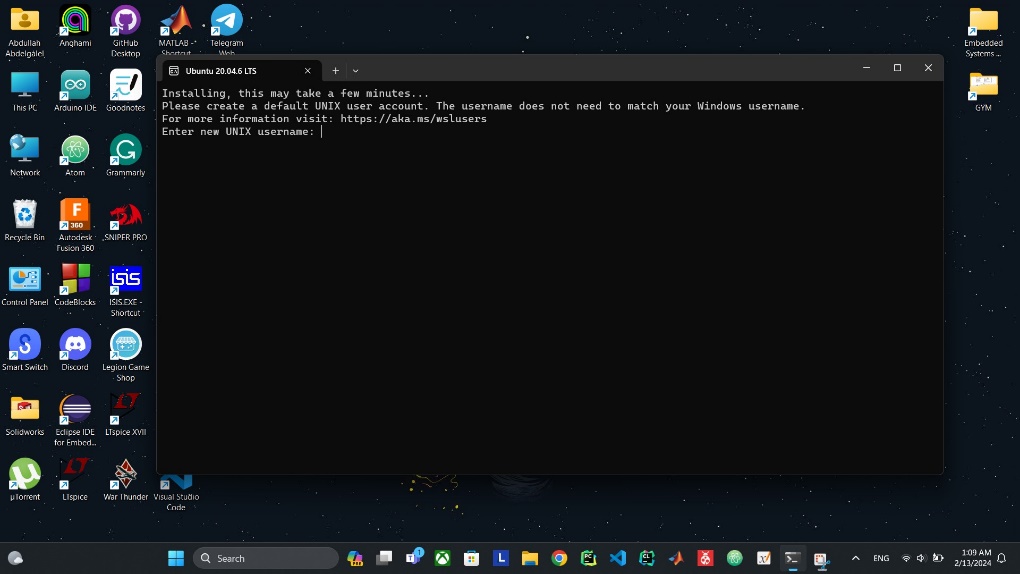
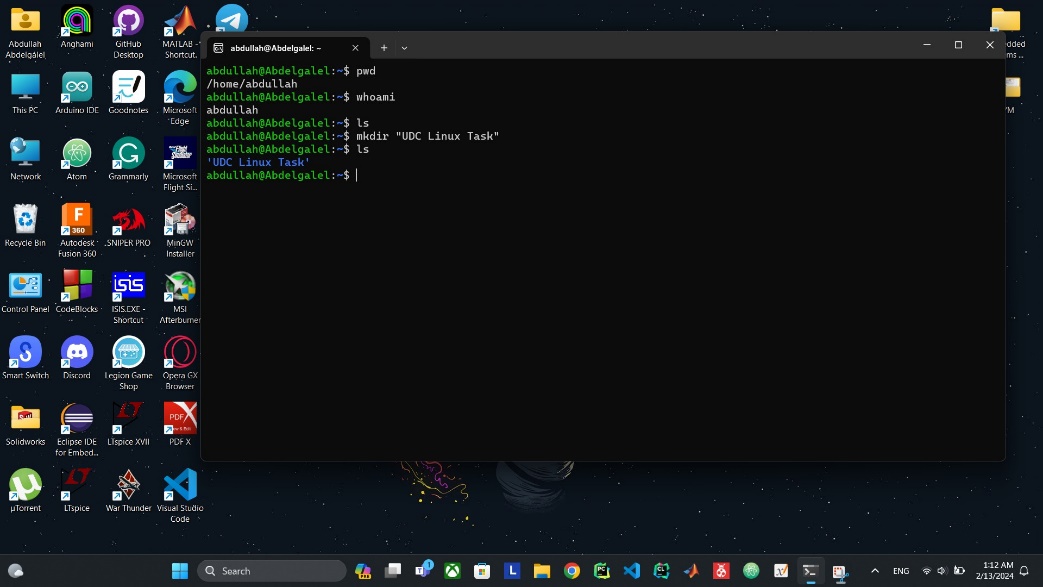
Initially, navigate to the Windows Store and search for "Ubuntu 20.04.6". Proceed to download the application. Ensure that Virtualization is enabled from the BIOS of the computer if it's not already enabled. Upon opening the app, it will initiate the installation process and prompt the user to input their username and password.

Figure 15: Username Creation Window

The images below depict a virtual machine environment created earlier on VM Workstation, running "Ubuntu 20.04.03".

Figure 16: Some Linux Terminal Commands

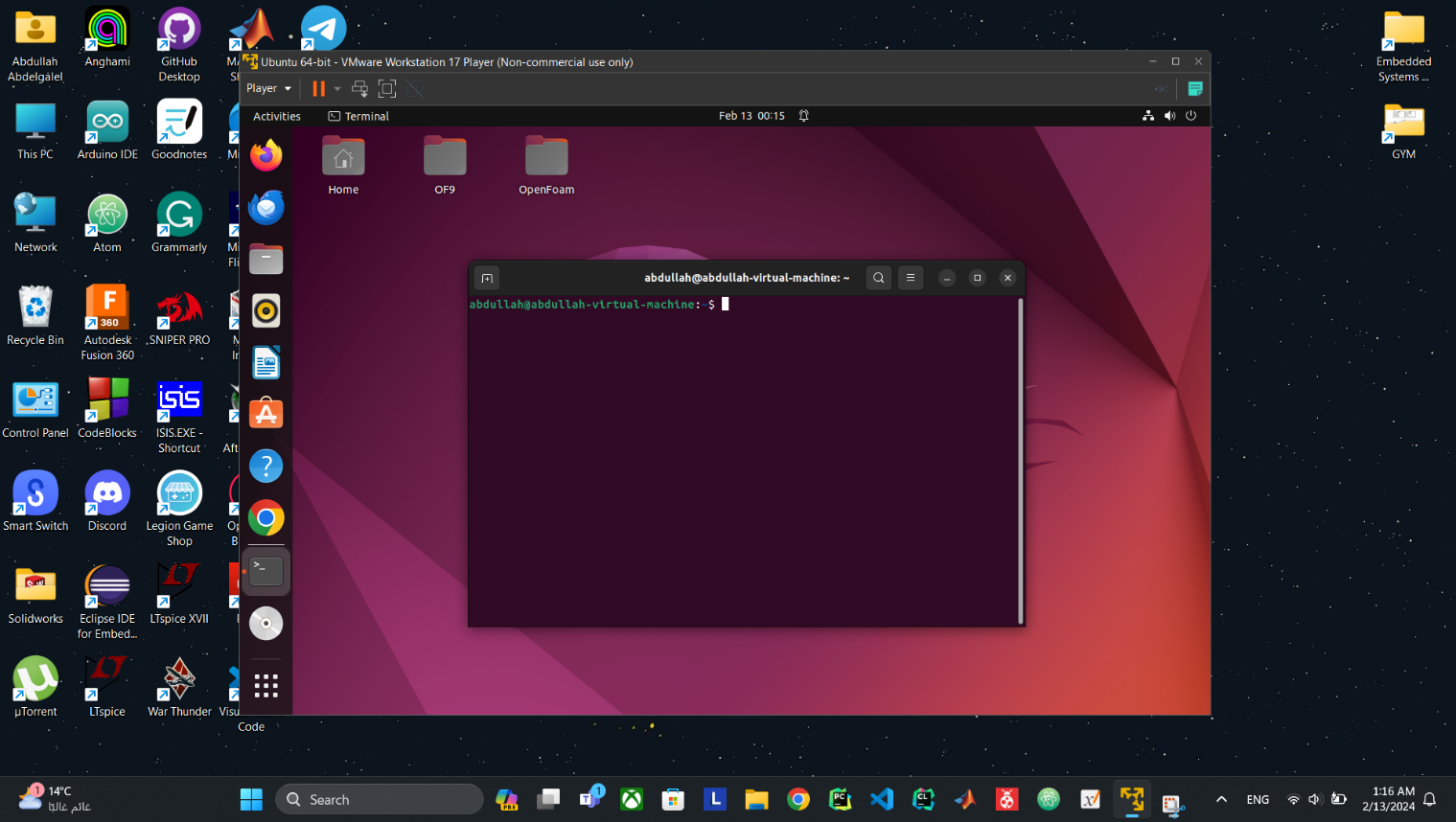
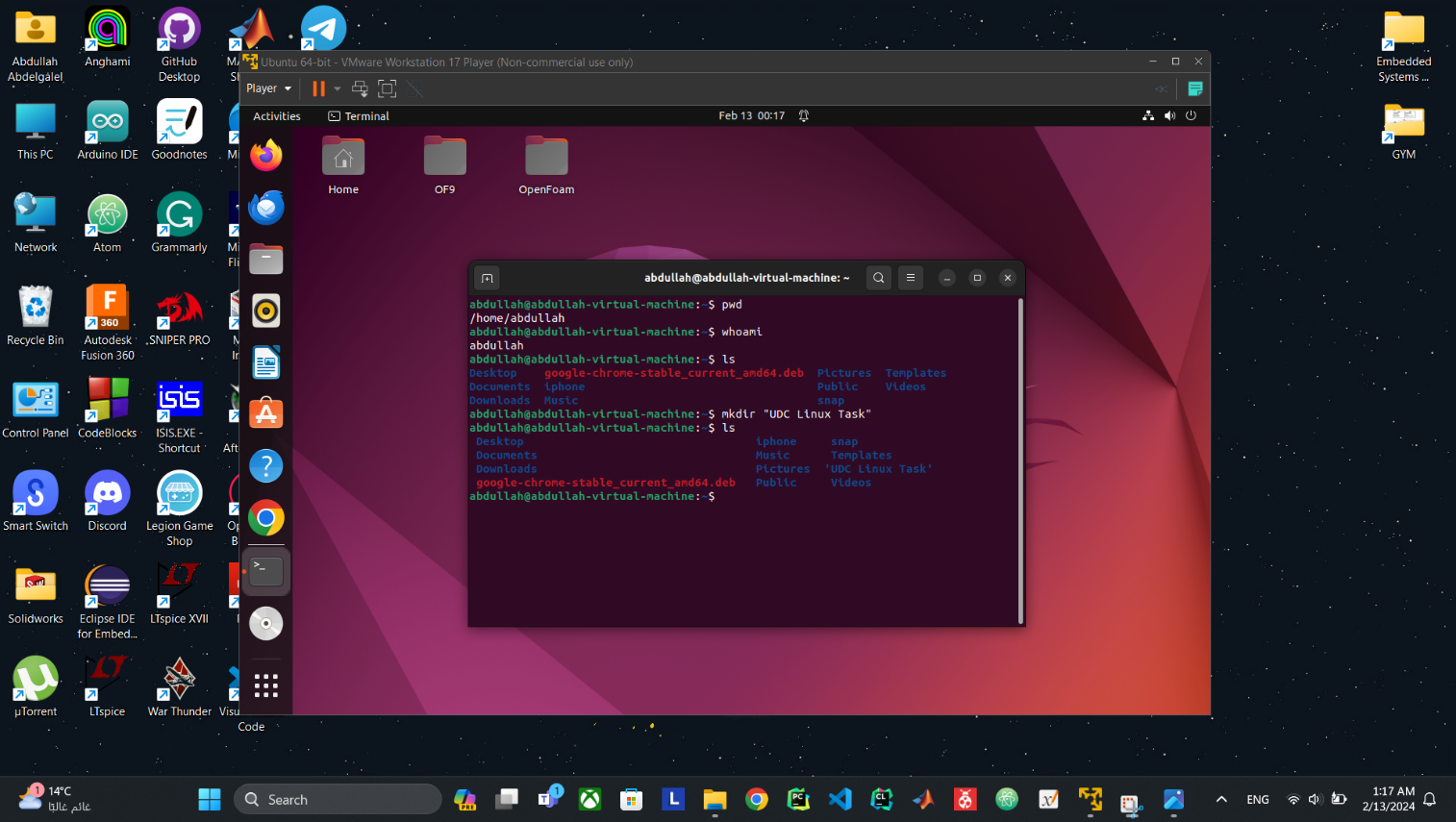


Figure 17: Ubuntu OS with some Terminal Commands