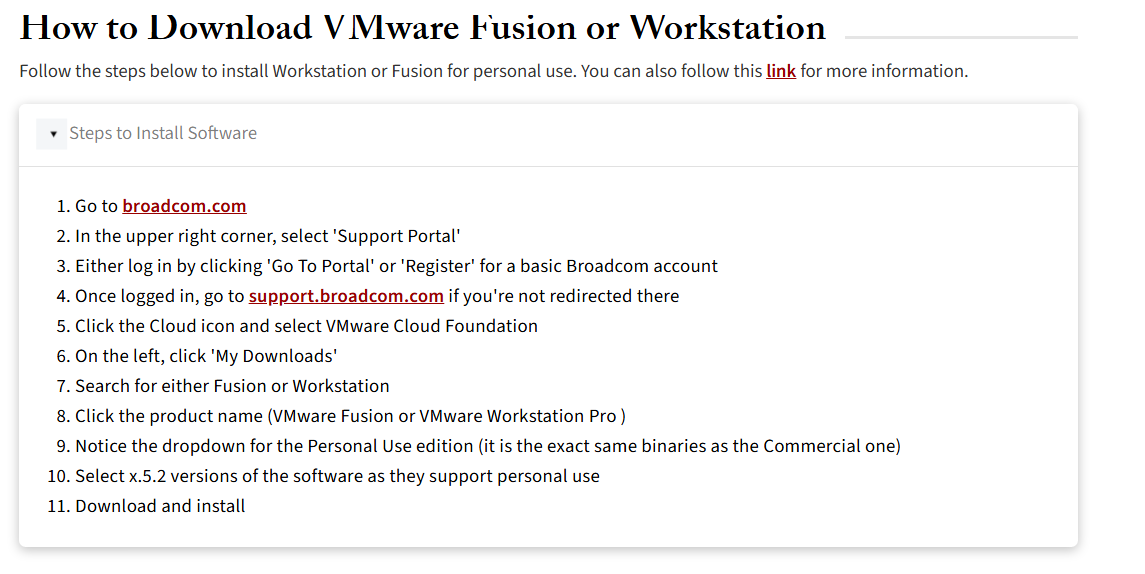
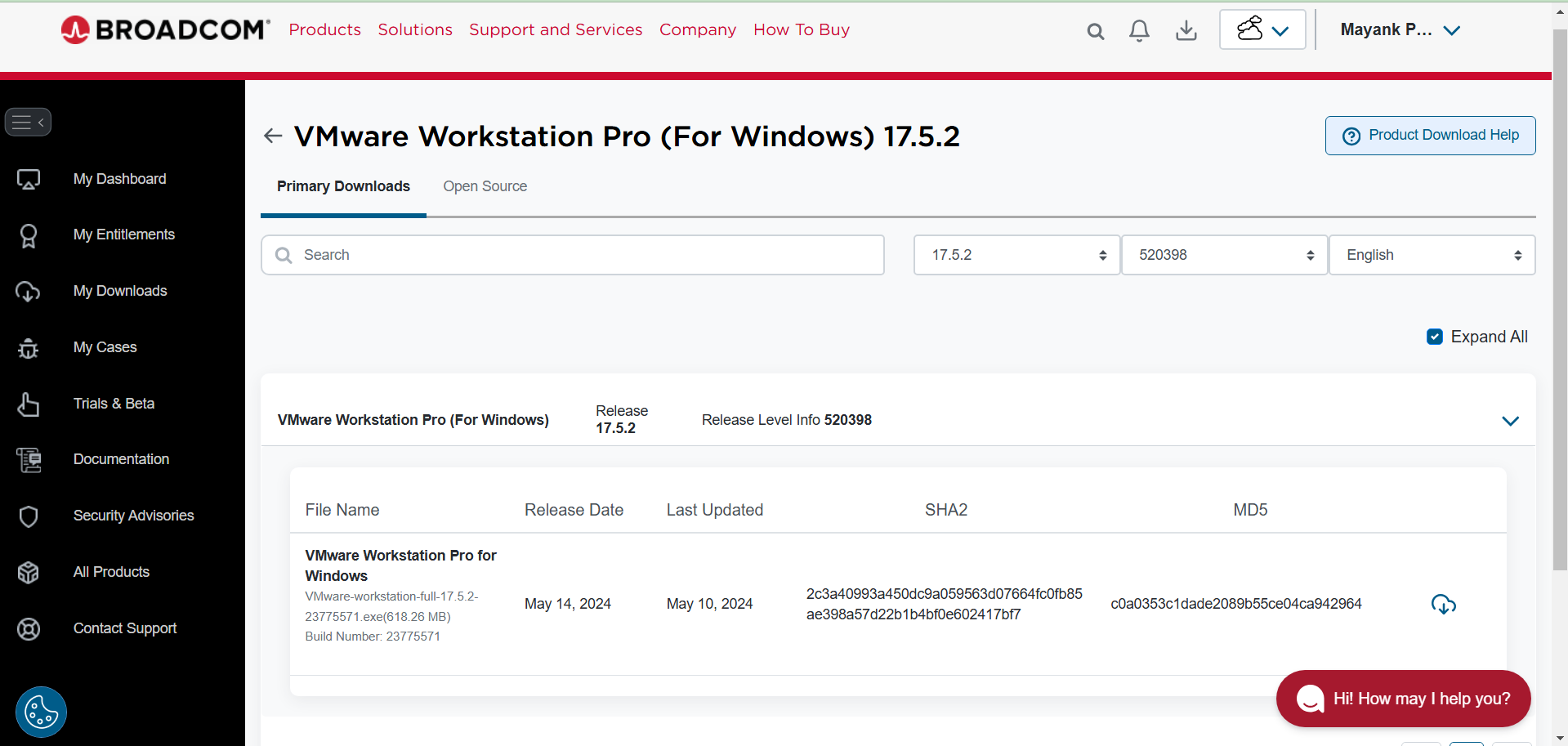
**Lab 1**

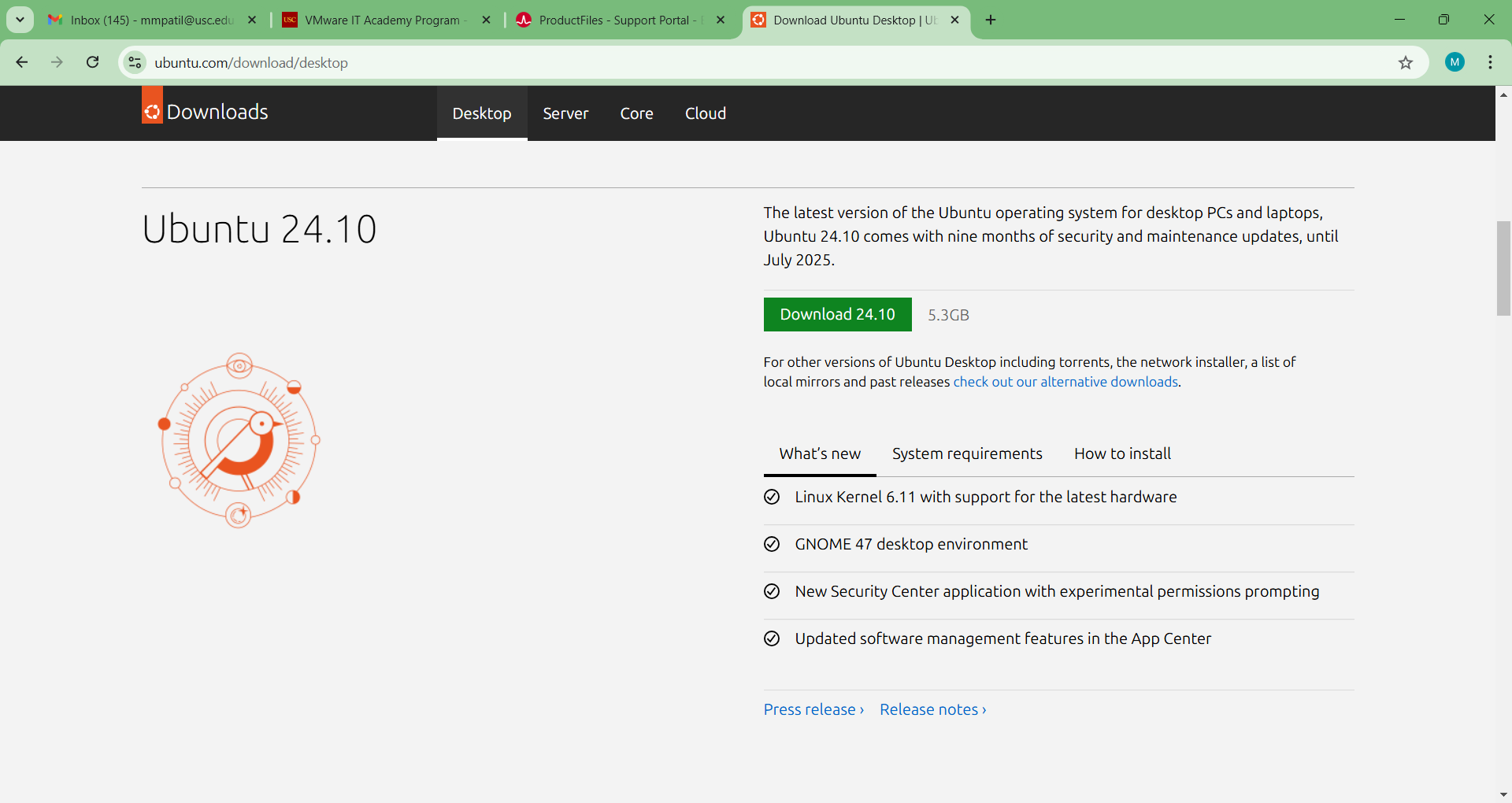
1. **Installation and Setup**

**1.1. Install VMware**

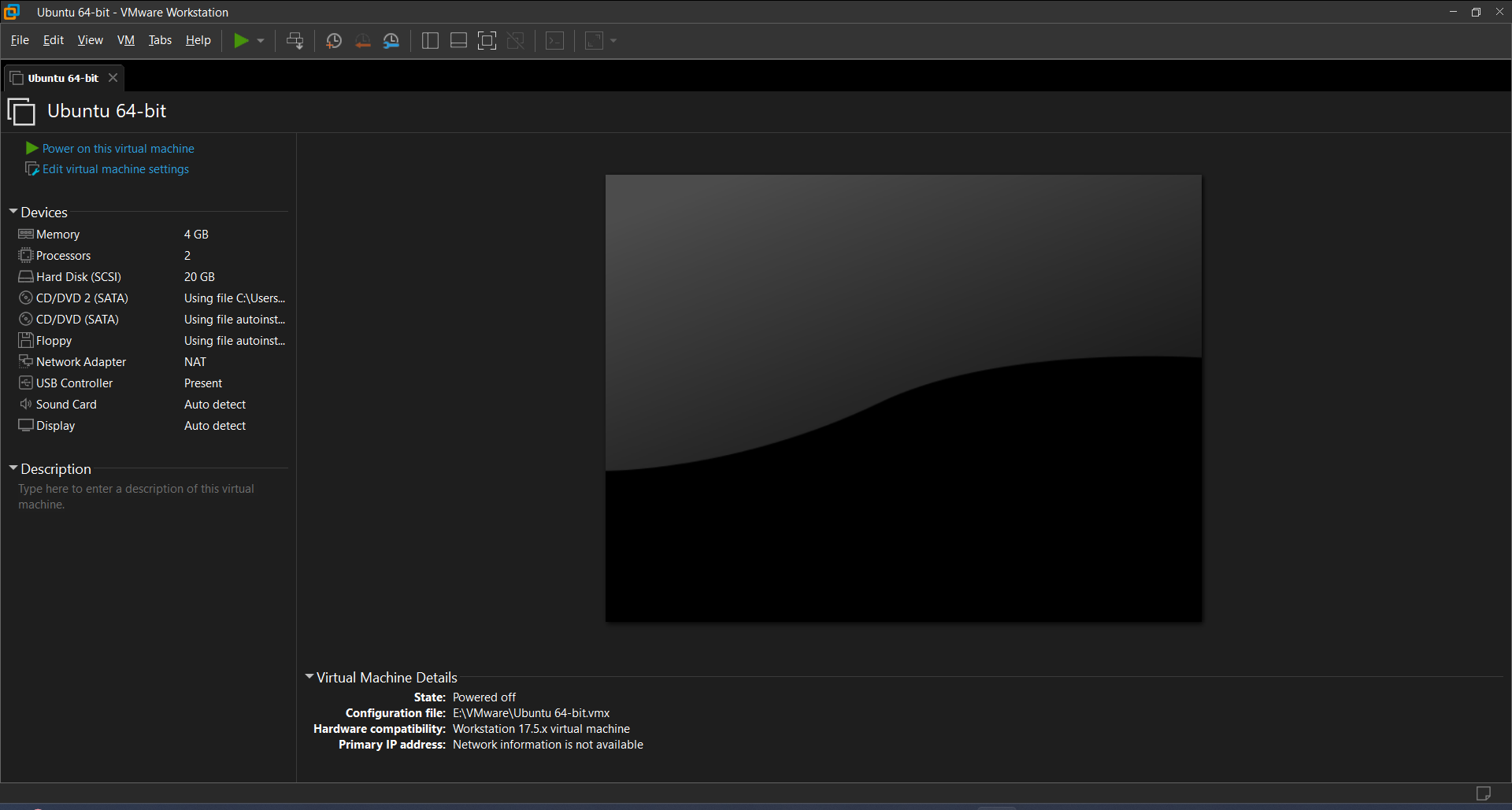




Downloaded & Installed VMware Workstation Pro 17.5.2

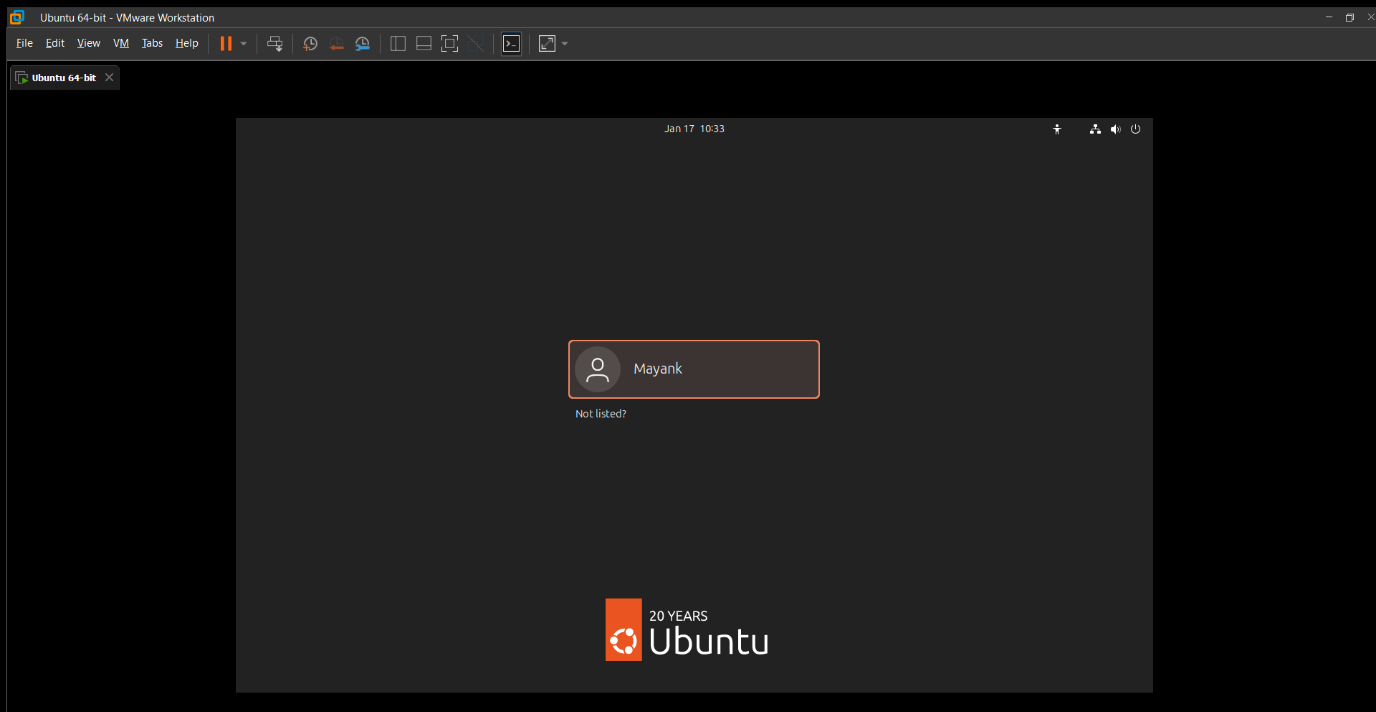
**1.2. Download Ubuntu ISO Image**

Downloaded Ubuntu ISO Image from the official Ubuntu website (Version: Ubuntu 24.10)



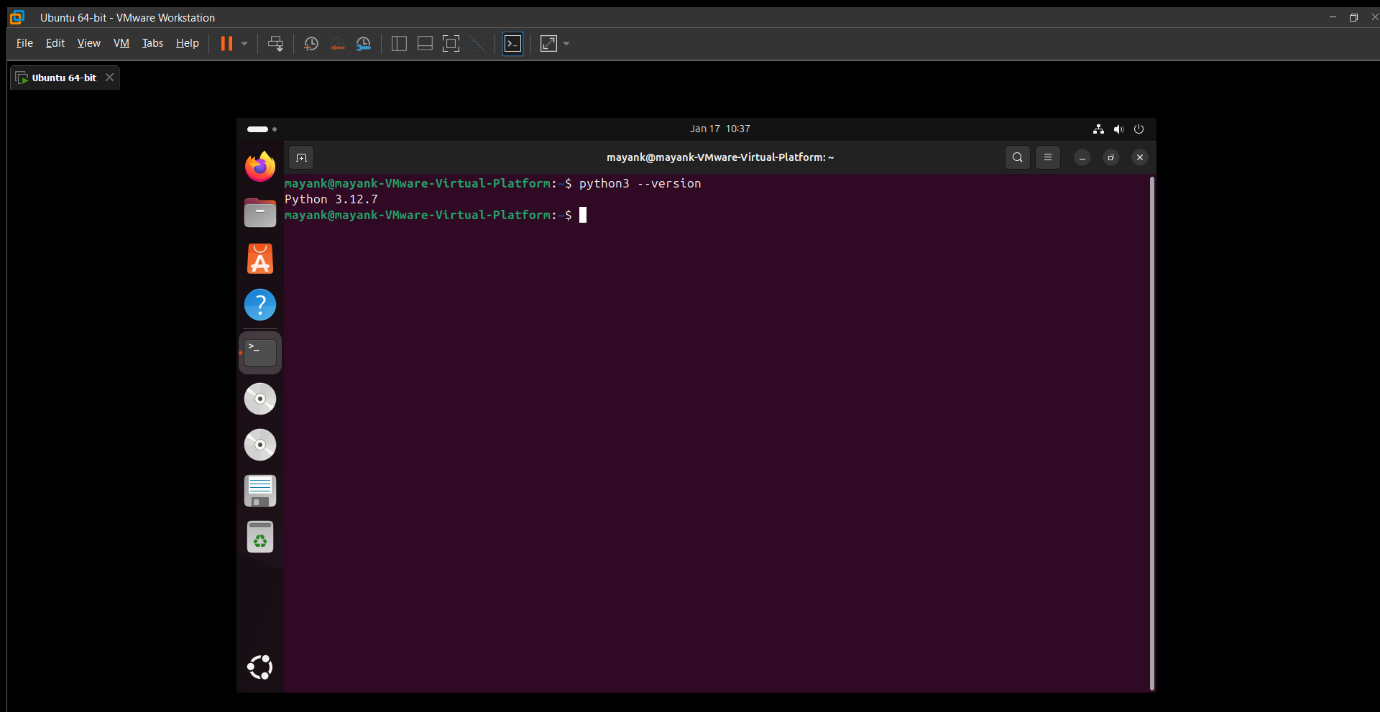
Created a new virtual machine with the ISO Image with the recommended specifications

RAM: 4 GB  
Disk Space: 20GB

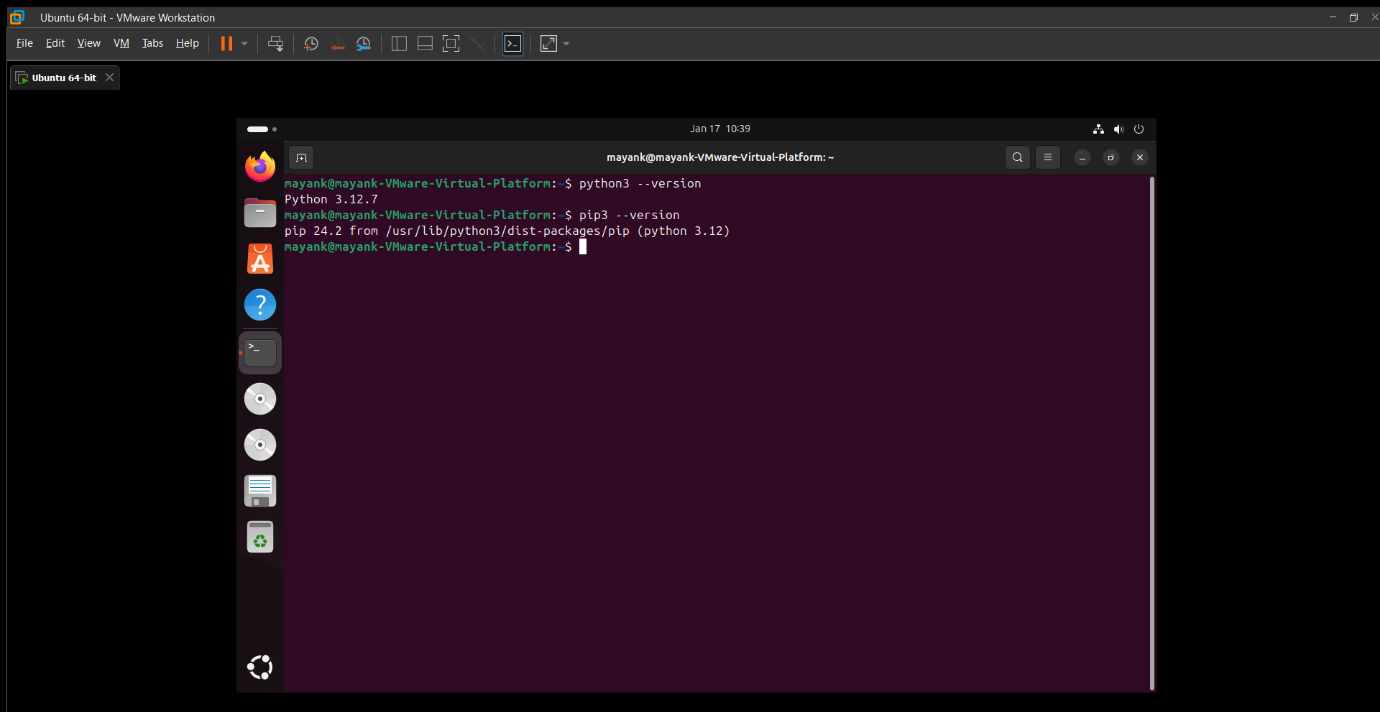


VM ready to use.

**1.3. Install Python on Linux**



Updated the packages and Installed python 3



Installed pip 3

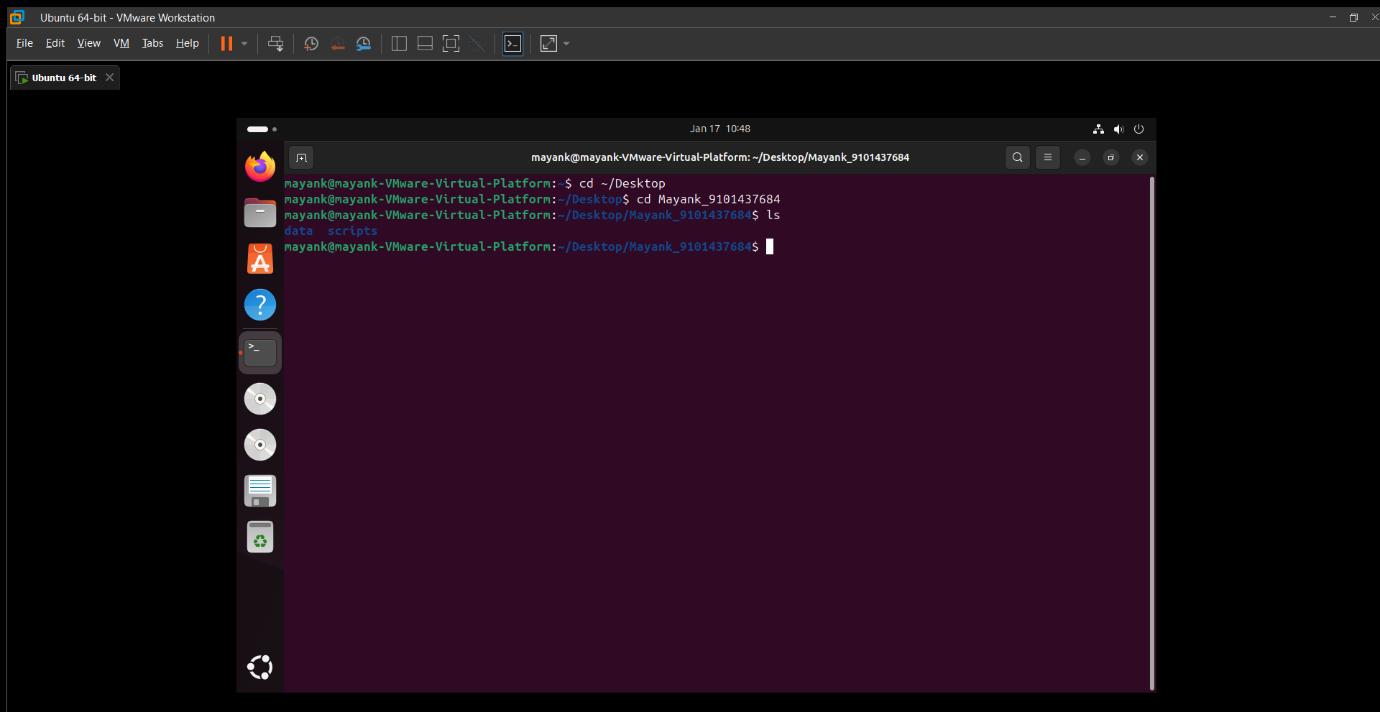
**1.4. Tutorials**

Went through the required tutorials

1. Introduction to Linux
2. Linux Command Cheat Sheet
3. Official Python website's tutorial
4. Python for Beginners

**2. Get Familiar with Linux and Python**

**2.1. Playing around with Linux Terminal**



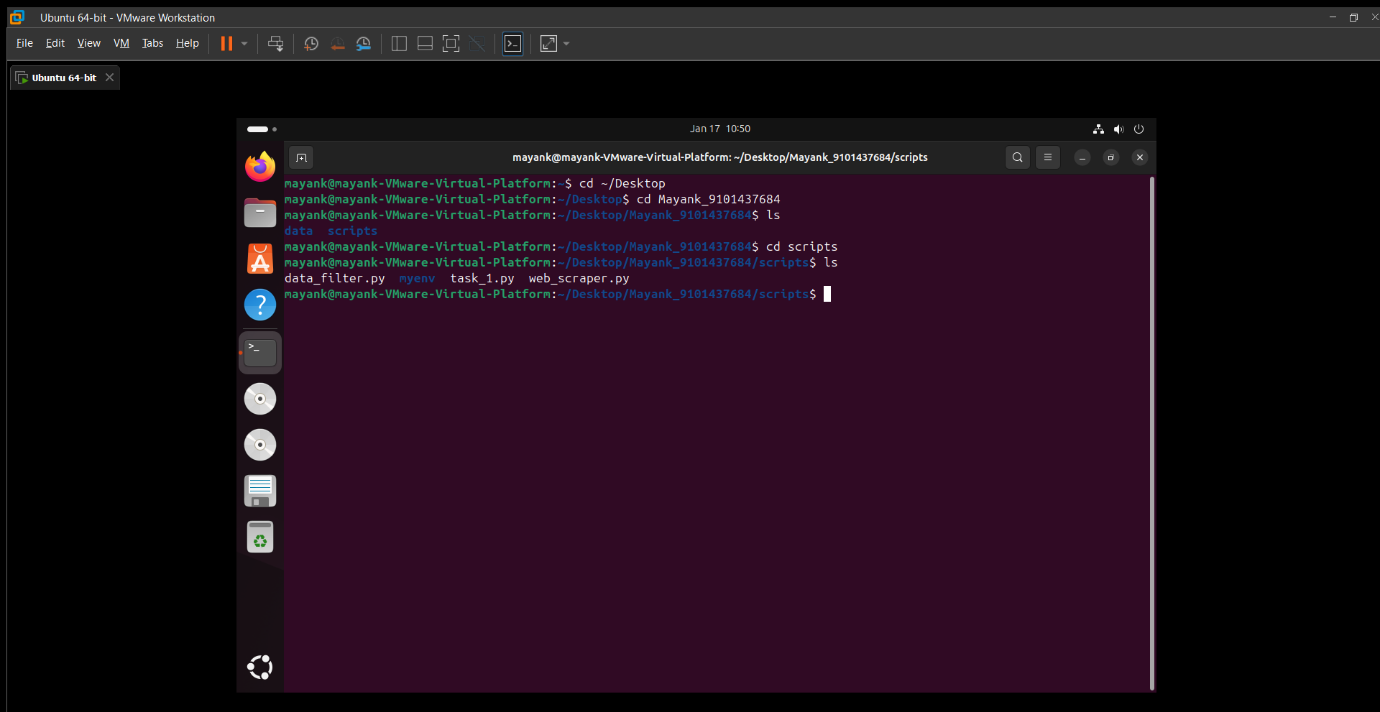
Created a new Directory on Desktop name Mayank\_9101437684 and created the required folders mentioned in the Lab using the terminal  
Commands used to do so:  
cd ~/Desktop  
mkdir Mayank\_9101437684

cd Mayank\_9101437684

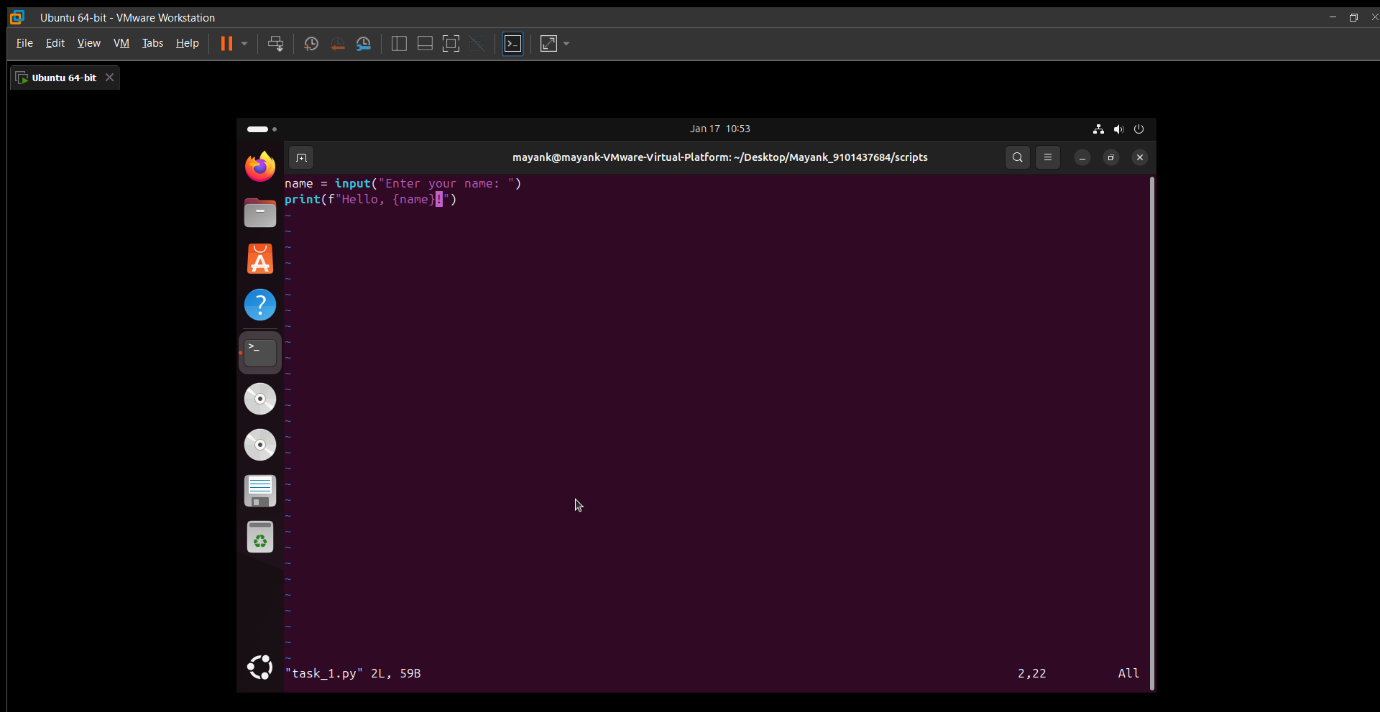
mkdir data scripts

cd scripts

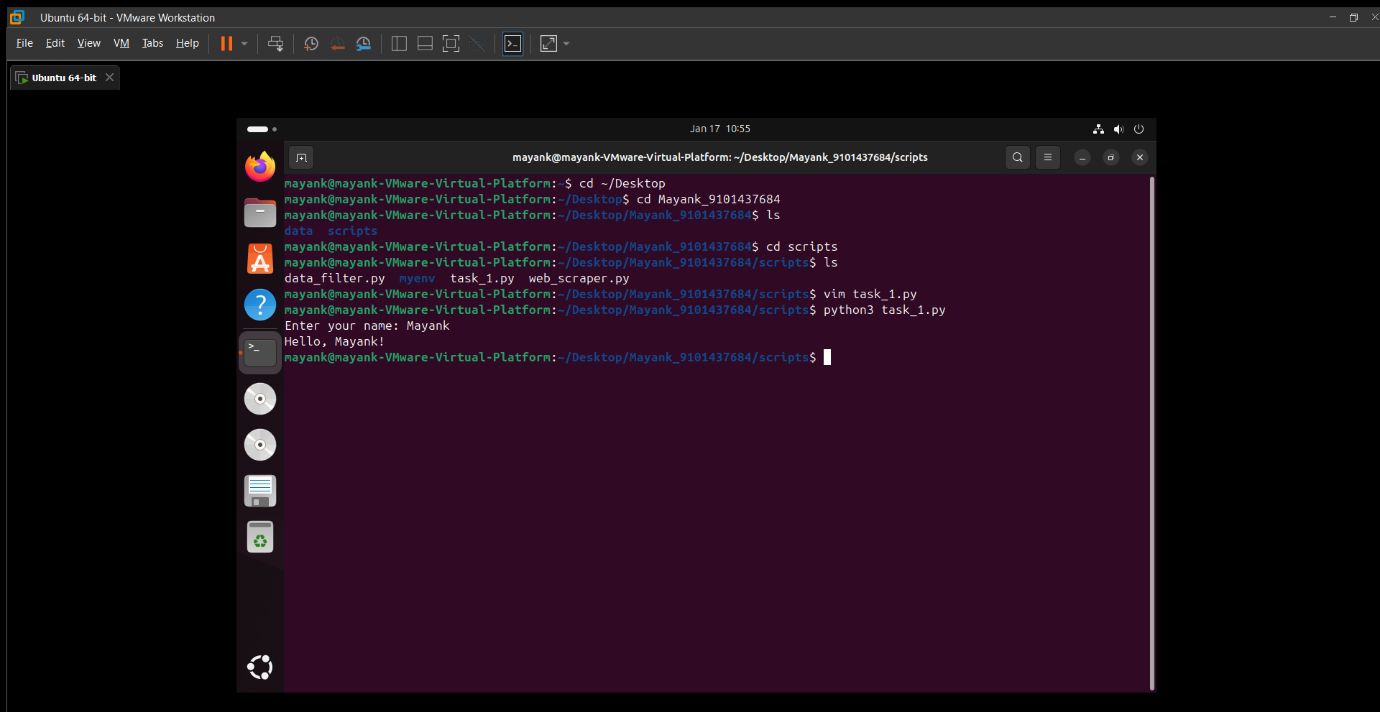
touch task\_1.py



**2.2. A basic Python Script**

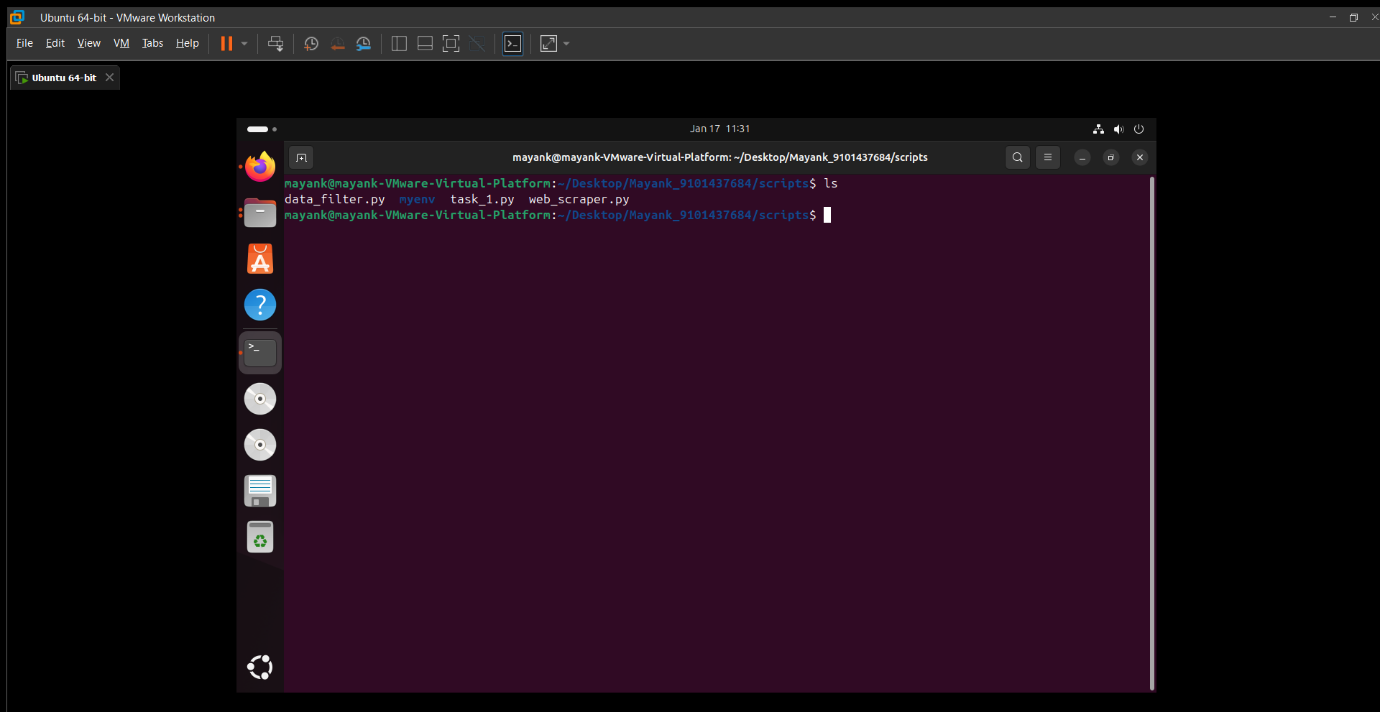


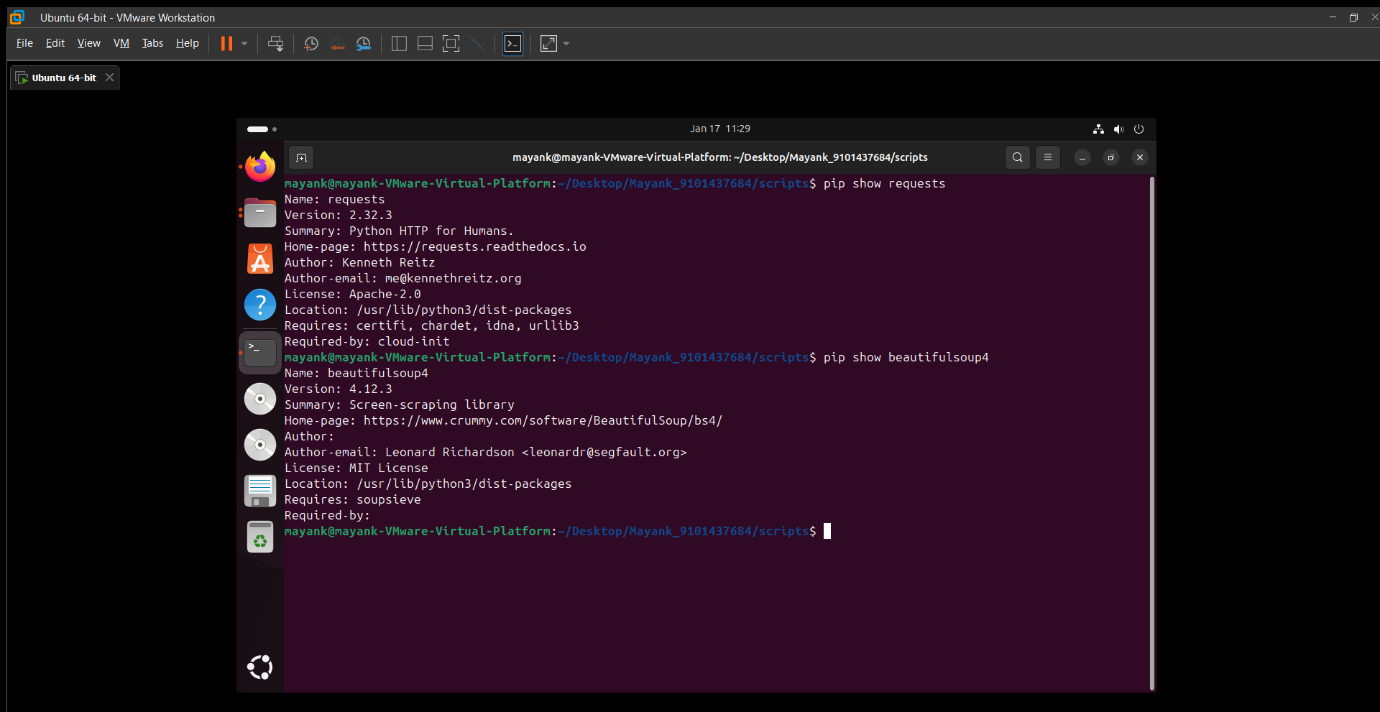
Script that reads a user's name as input and greets the user with "Hello, [name]!".



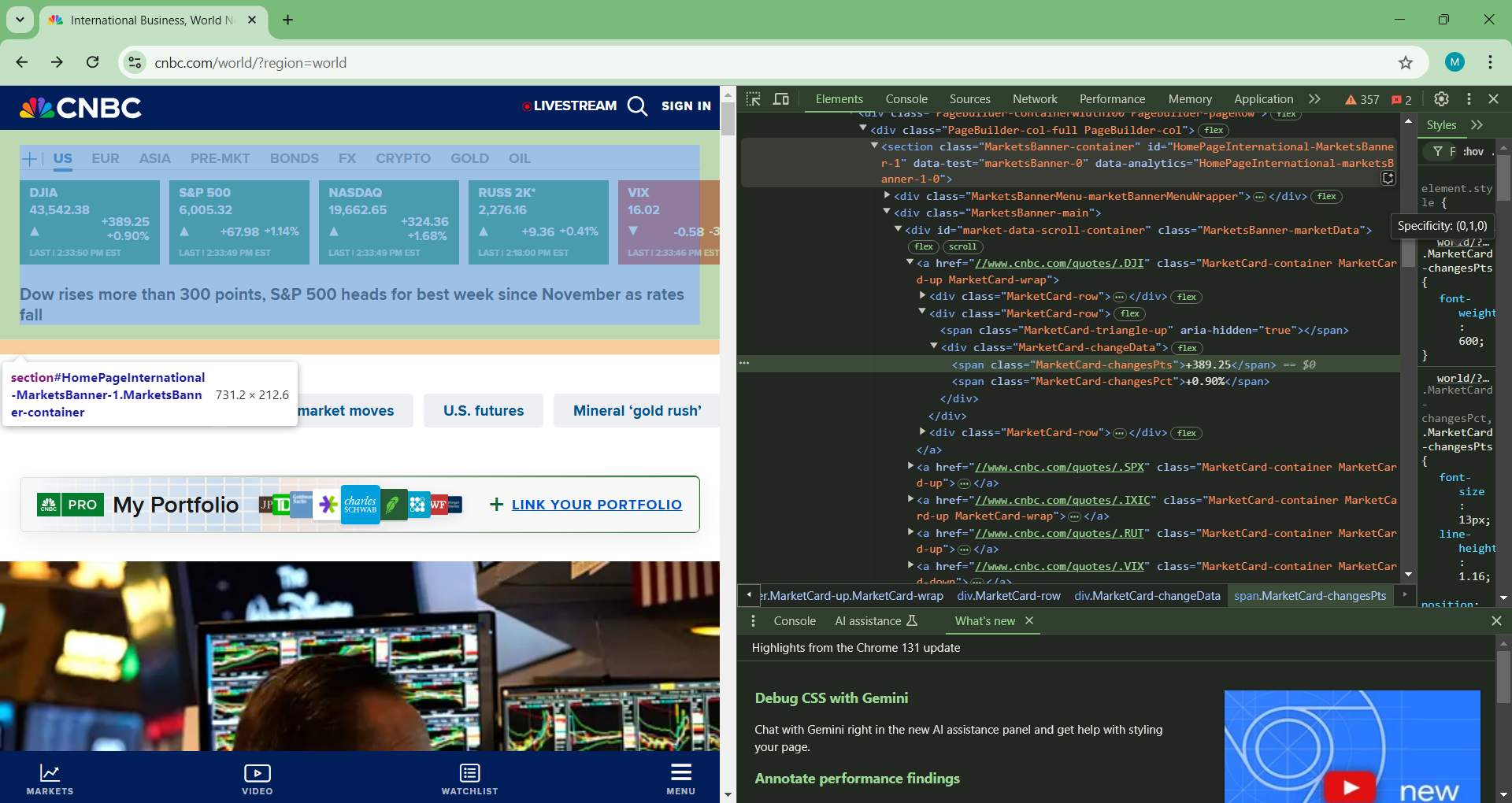
Executing the task\_1.py file   
Output: Hello, Mayank!

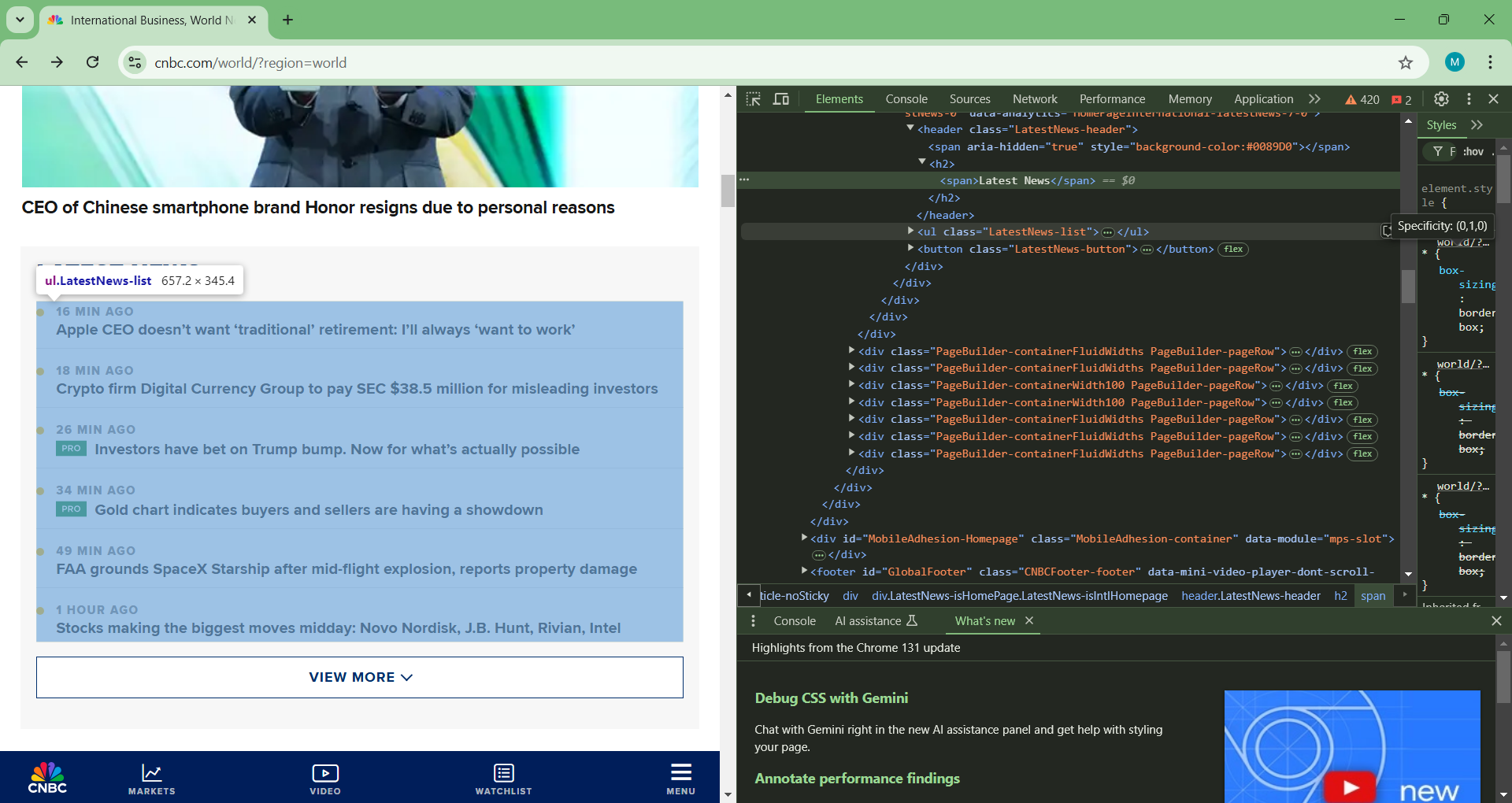
**2.3. Python Web-scraping Task**





Installed the required libraries (Requests and BeautifulSoup4)

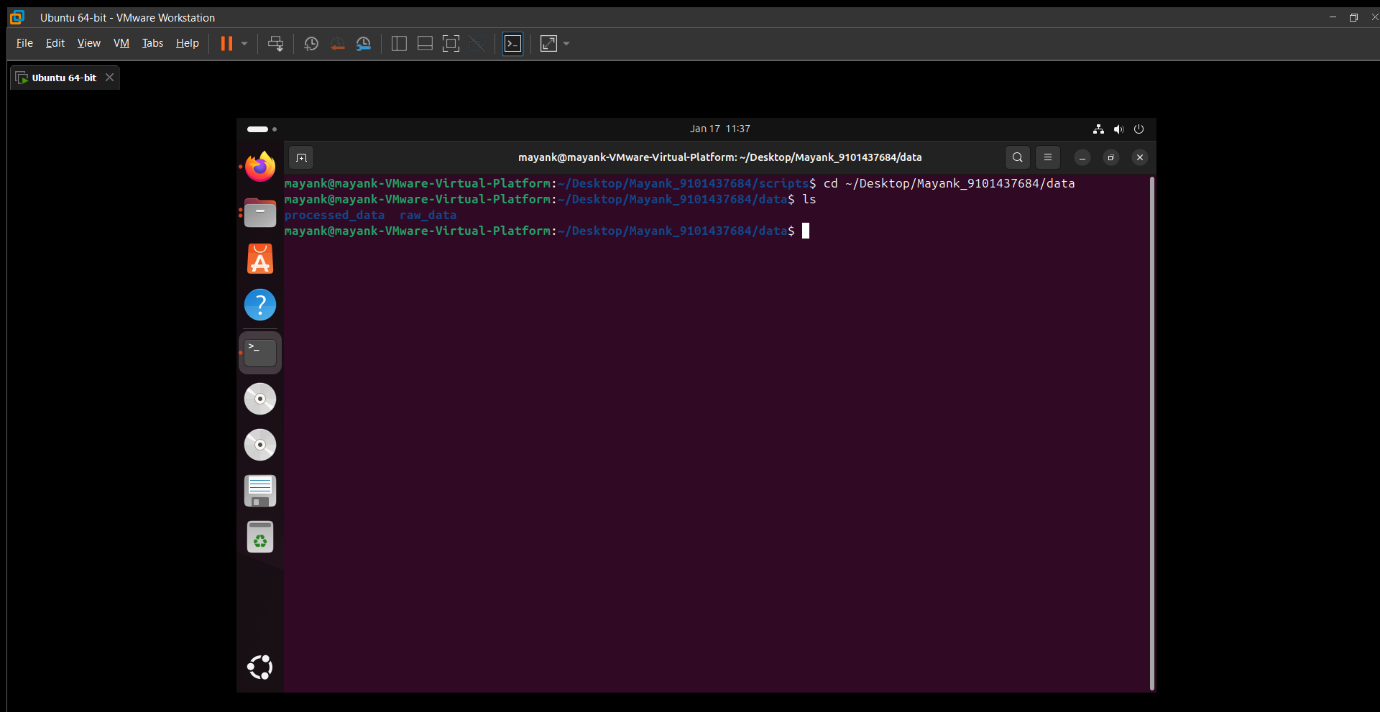
****

****

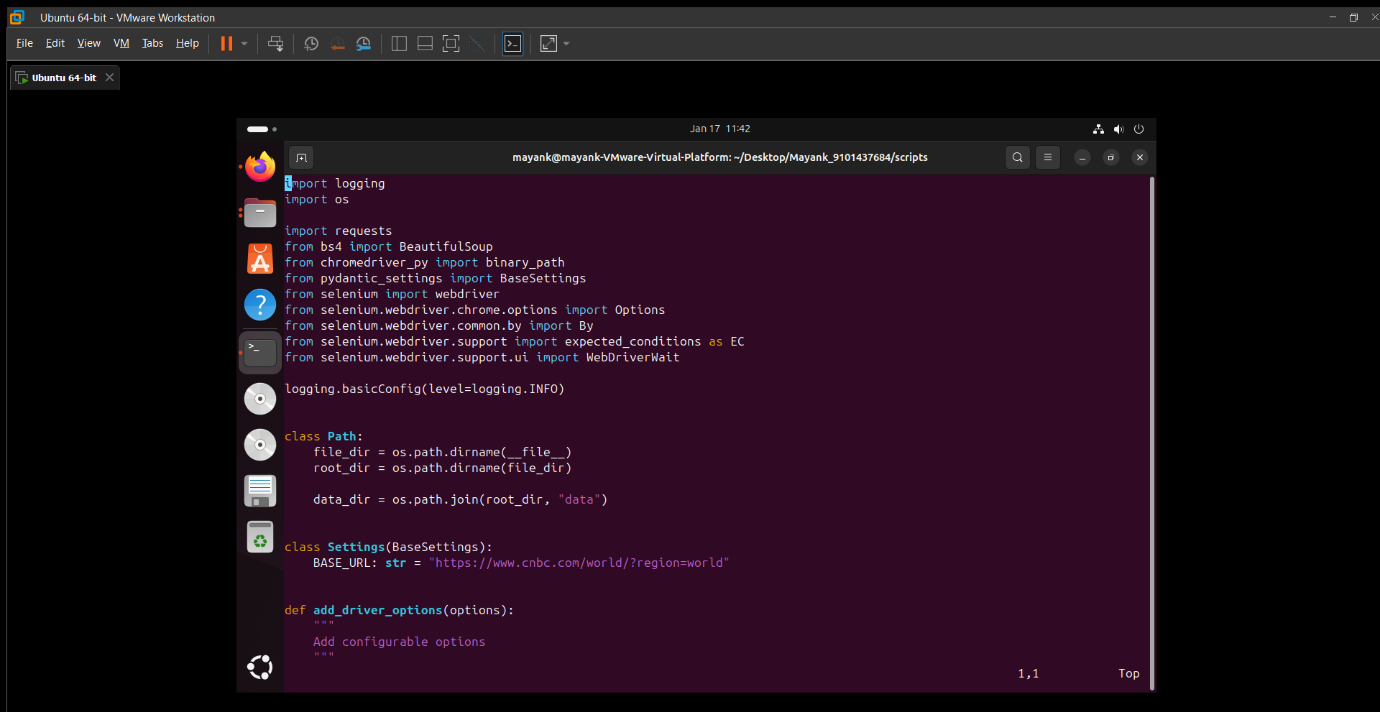
Analyzed the HTML structure by inspecting the elements on the Page.

Found and inspected the corresponding tags for the Market banner on the top and the Section

titled Latest News on the page.



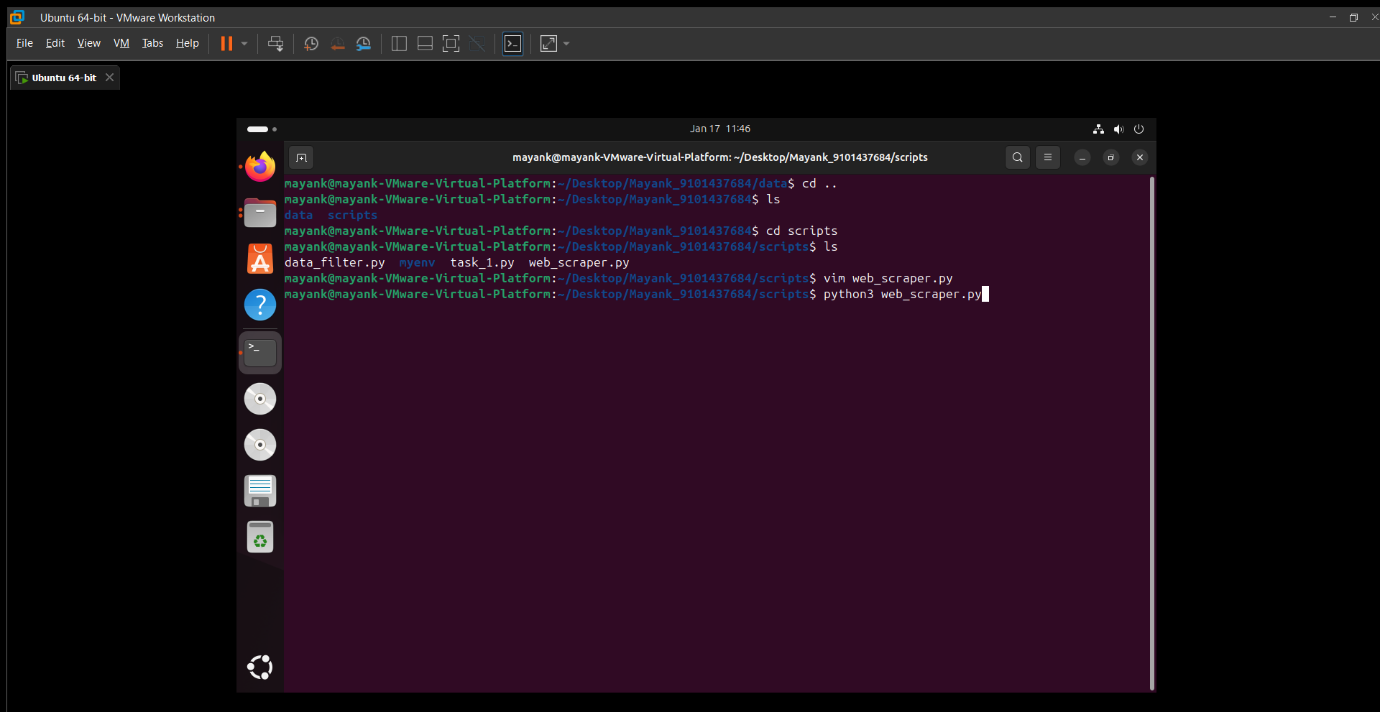
Data -> raw\_data , processed\_data



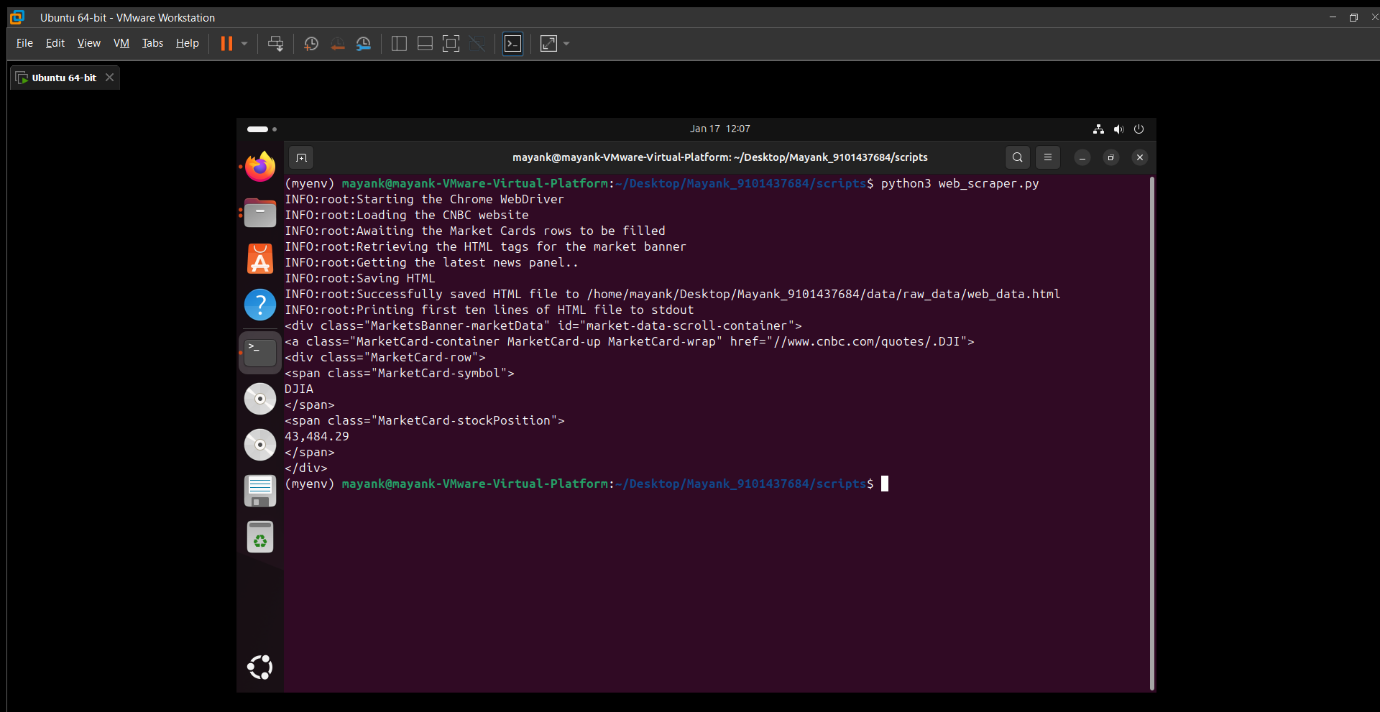
Python script that uses Requests and BeautifulSoup to collect data from the provided link.

This Python script is intended to collect data from the CNBC website pertaining to global markets. It starts by setting up logging to capture informative messages and specifying file and directory paths. The script uses the `pydantic\_settings` module to handle configuration settings, using a default base URL for CNBC. Next, it initializes a headless Chrome WebDriver through `chromedriver\_py` with adjustable options. The core functionality revolves around loading the CNBC webpage, waiting for the market card rows to become visible, and extracting the HTML content for both the market banner and the latest news panel with BeautifulSoup.

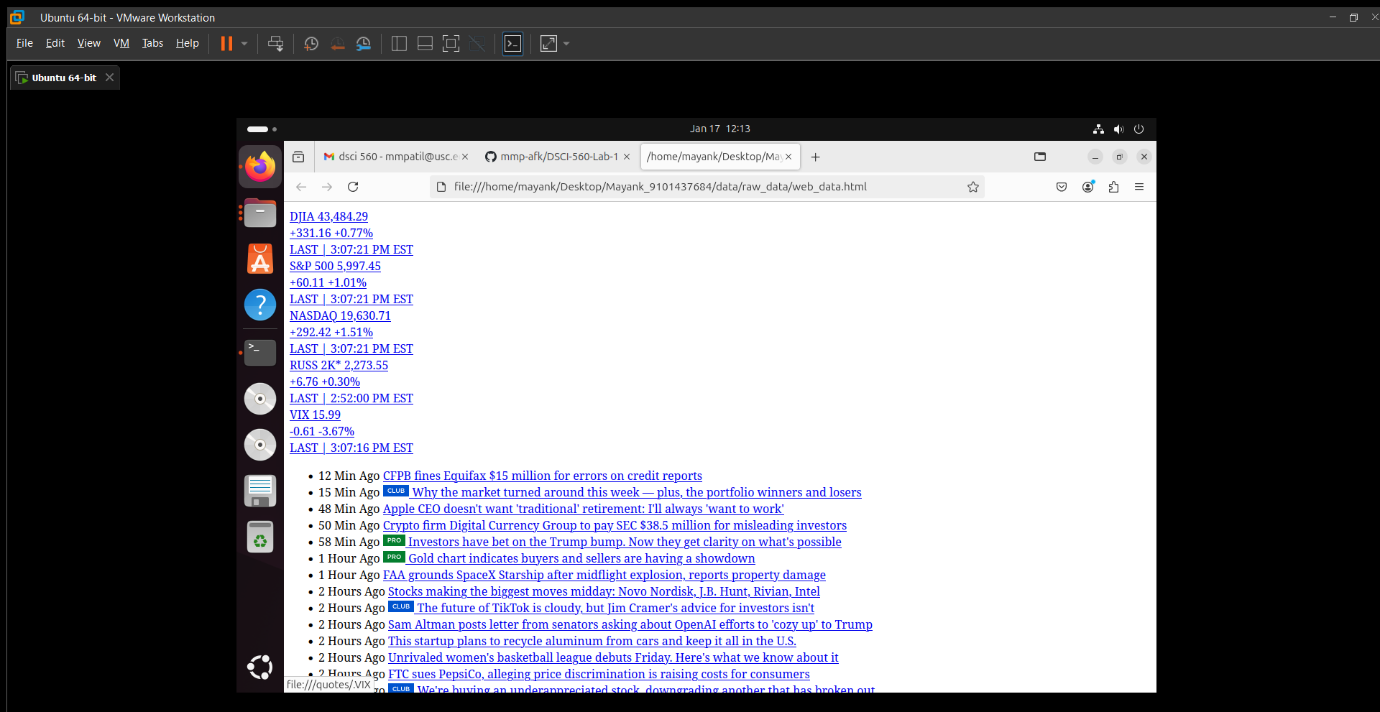
The HTML that was extracted is stored in a file located in the "raw\_data" folder. As part of its error management, the script records any exceptions that may arise during the data collection process. Ultimately, it displays the first ten lines of the saved HTML document on the standard output for a quick check. In summary, this script showcases the use of web scraping tools such as Selenium and BeautifulSoup, along with logging that offers insights into various stages of the data acquisition process.



Running the web\_scraper.py file.

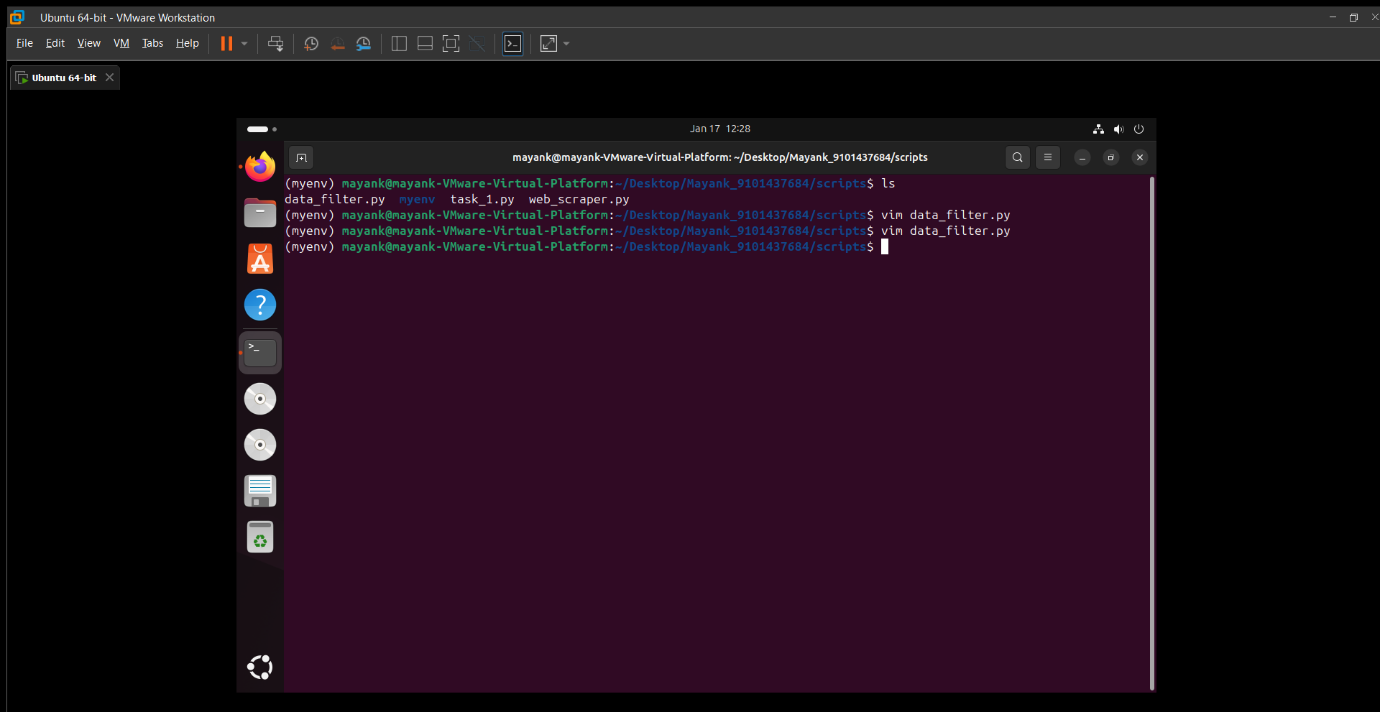


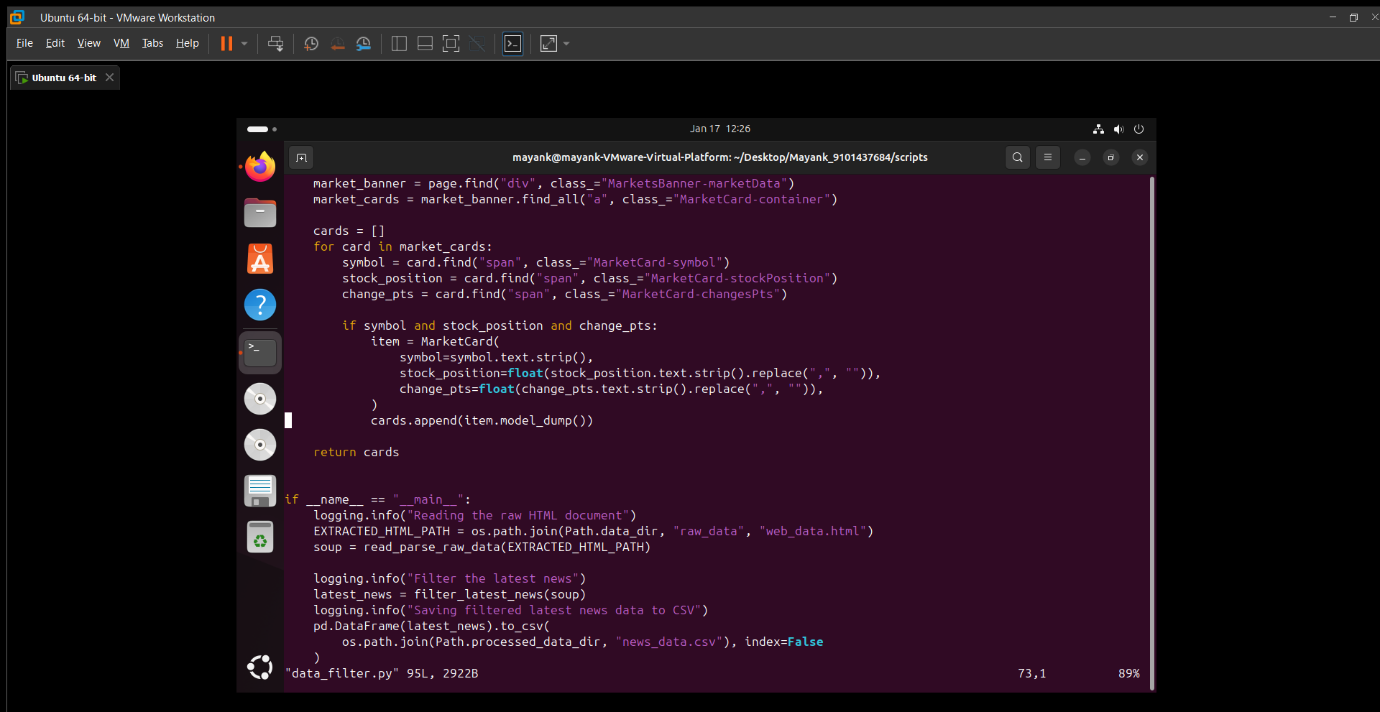
Successfully printed the first 10 lines of the created html file on the terminal.



Saved the collected data in the raw\_data folder to a file named web\_data.html.

**2.4. Data Filtering Task**





Python script to read the web\_data.html file into a Python list, extracting specific elements of interest from the data.

The script filters the data to identify the relevant fields. This Python code accomplishes the following objectives:

1. Logging Configuration: Sets up logging to display messages at the INFO level.

2. File and Directory Path Definitions: Establishes the paths for files and directories using the `Path` class.

3. Parsing HTML and Extracting Data:

a. "read\_parse\_raw\_data": Reads and interprets an HTML file with BeautifulSoup.

b. Defines Pydantic models "NewsItem" and "MarketCard" for organized data representation.

4. Functions for Data Filtering:

a. "filter\_latest\_news": Gathers the most recent news feed from the HTML, transforms it into instances of the "NewsItem" Pydantic model, and adds them to a list.

b. "filter\_market\_banner": Extracts market banner information from the HTML, converts it to instances of the "MarketCard" Pydantic model, and appends them to a list.

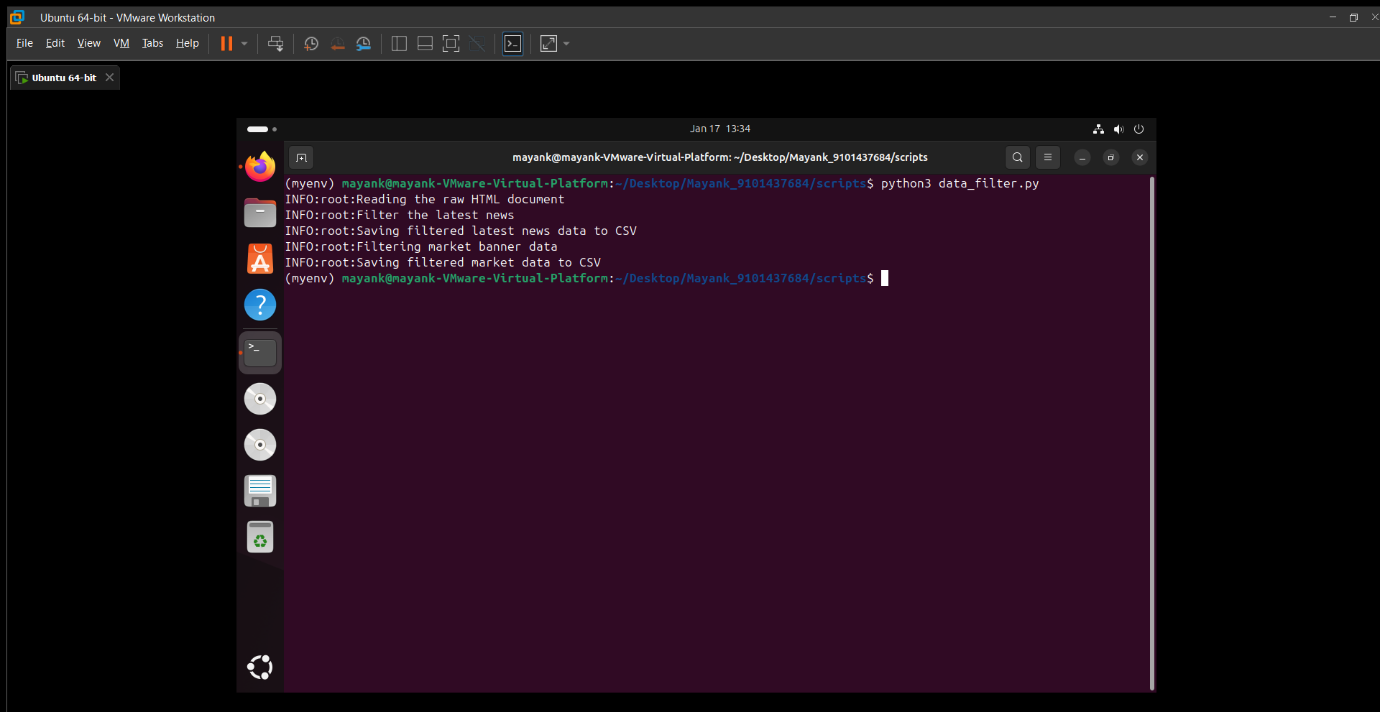
5. Main Workflow:

a. Reads and processes an HTML file ("web\_data.html").

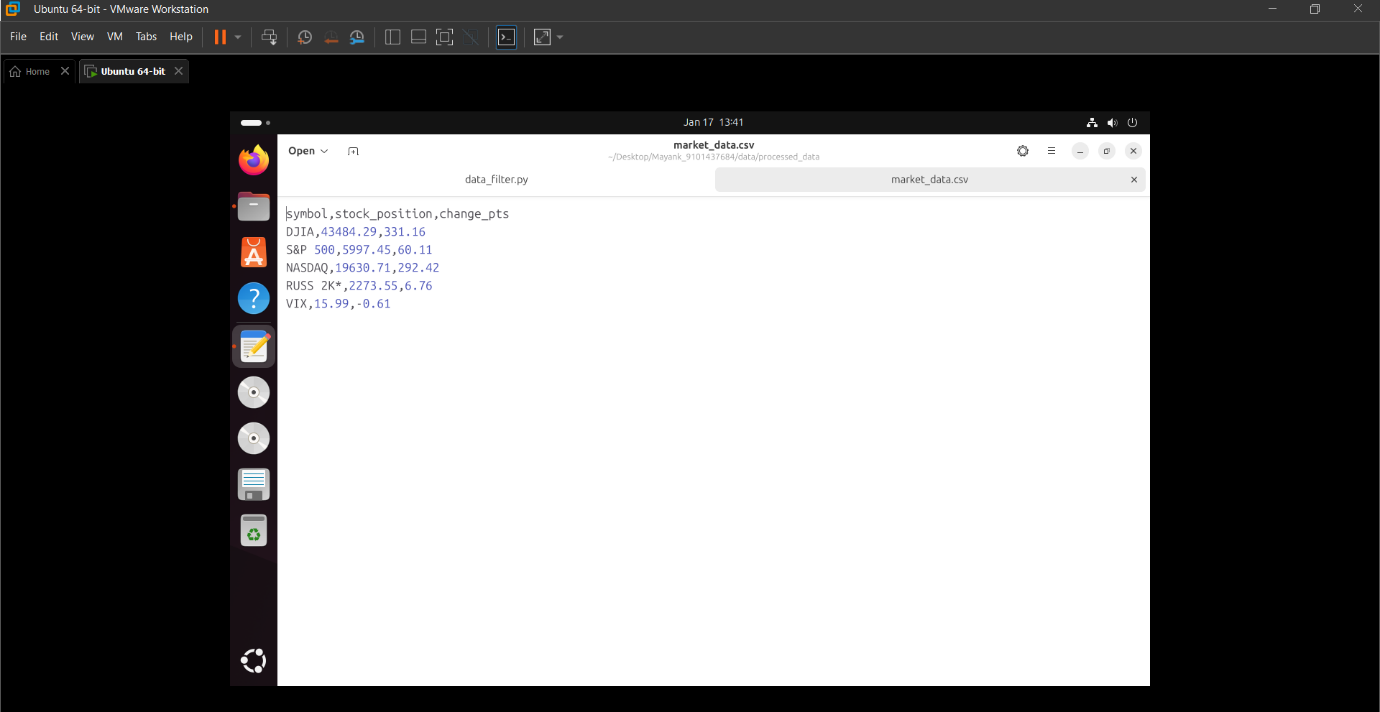
b. Utilizes the established functions to filter the latest news and market banner information.

c. Logs the activities at various stages (reading, filtering, and saving).

d. Exports the filtered information into CSV files ("news\_data.csv" and "market\_data.csv") in the "processed\_data" directory.



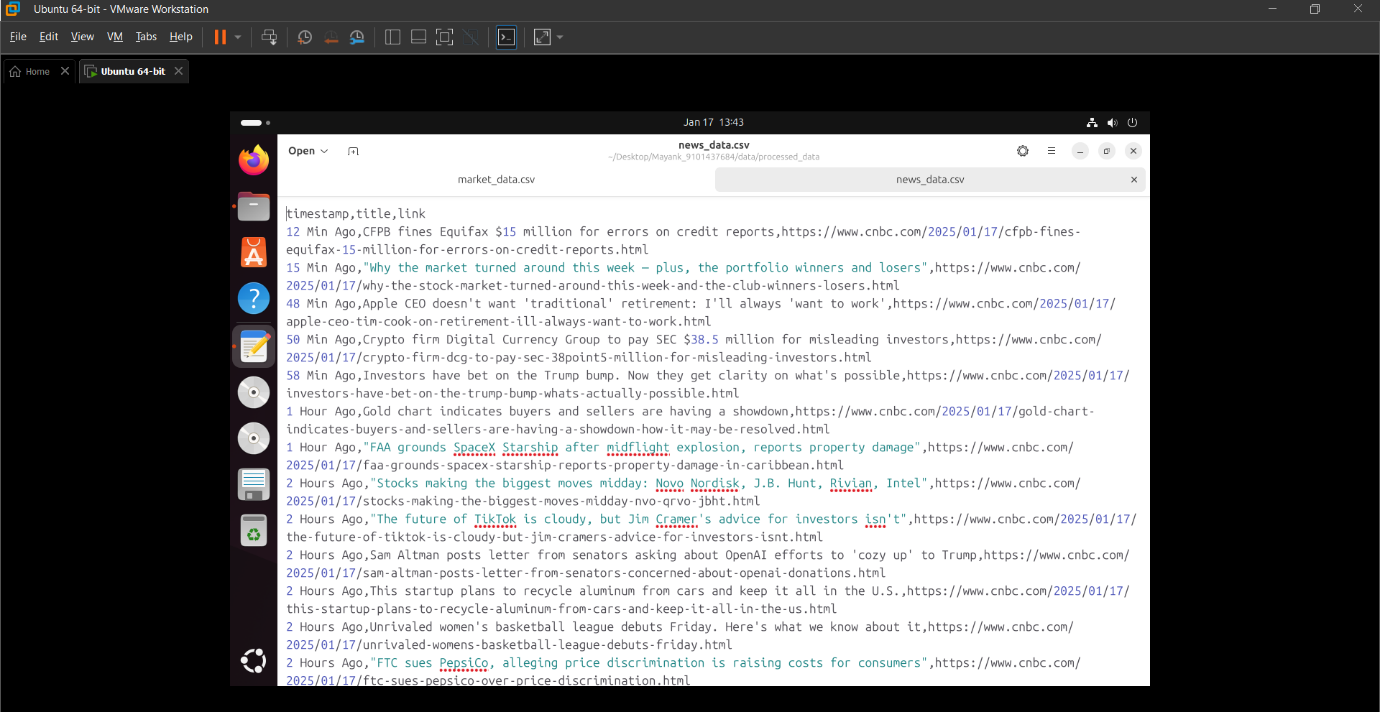
Running the script



The generated market\_data.csv file in the processed\_data folder.

Market data - Has the following fields:

* + - * symbol – abbreviations for a stock ticker.
      * stock\_position – current/latest price of the stock.
      * change\_pts – most recent change in price observed.

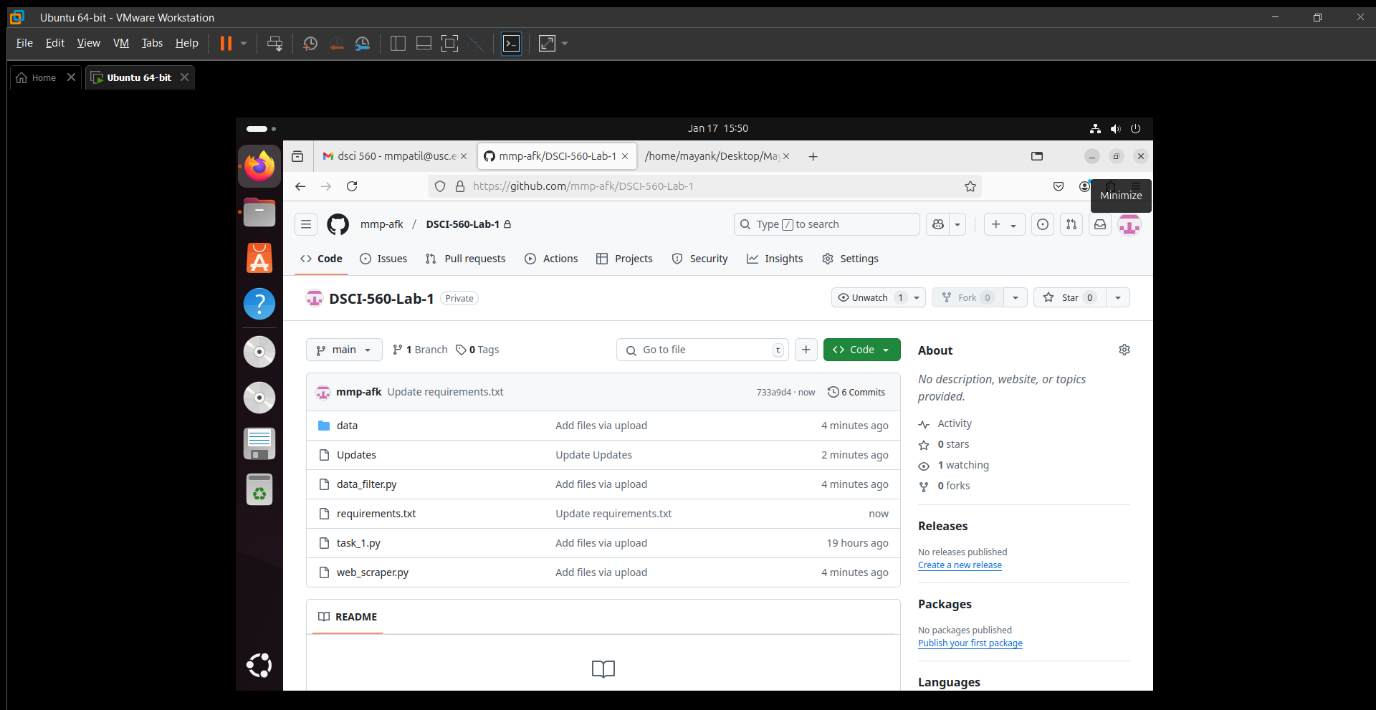


The generated news\_data.csv file in the processed\_data folder.

Latest news - Has the following fields:

* + - * timestamp – approximate time from the current time when the news was posted.
      * title – the headline of the news column
      * link – link to the official source of news on the website

**GIT HUB LINK -** [**https://github.com/mmp-afk/DSCI-560-Lab-1**](https://github.com/mmp-afk/DSCI-560-Lab-1)

****

Commit Updates