**DSCI 560 Lab 1: Report**

# Installation and Setup

## VirtualBox and Ubuntu Installation

* VirtualBox 7.1 x86\_64 was downloaded from official website and installed on local Windows PC.
* Ubuntu 24.04.1 STL x86\_64-bit image file was download from official website and installed on VirtualBox. The resources of the virtual machine were allocated as shown in the snapshot below:

A screenshot of a computer

AI-generated content may be incorrect.

## Python Environment Setup

* The Python environment was configured using Miniforge3 script, which was obtained from the official GitHub repository.
  + Subsequently, a new Python environment named “ds” was created with Python version 3.11, and the required libraries were installed.

A screenshot of a computer

AI-generated content may be incorrect.

# Get Familiar with Linux and Python

## Playing around with Linux Terminal

* The folder bearing my name and student ID on the Desktop was created using the "sudo mkdir" command, alongside other directories, as illustrated in the snapshots below:

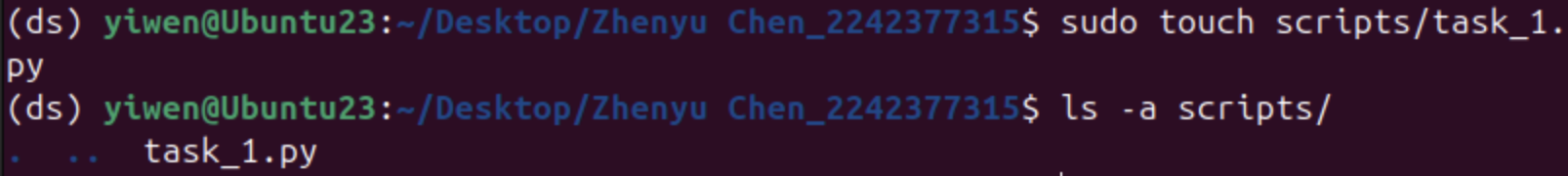
A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

* A Python script named “task\_1.py” was created using the “touch” command, as shown below:



## A basic Python Script

* For the first task, the script was opened and edited using the GNU nano editor via “nano” command.
  + The script defines a main function to read the input username variable and subsequently print a greeting with user’s name.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Python Web-scraping Task

In the “web\_scraper.py” script, the web scraping process is divided into two main parts: *Static Data*, which can be easily retrieved using the requests library, and *Dynamic Data*, which requires the selenium library as it is generated and updated through JavaScript.

**Implementation Logic:**

* **Static Data Retrieval**:
  + The static data from the target website was retrieved using the requests library. Most of the required information was captured, except for the market data. The static data was then processed into a “soup” structure powering by the BeautifulSoup4 library.
* **Dynamic Data Retrieval:**
  + The dynamic data (primarily the market data within the “***section***” tag) was extracted using the selenium library, along with the Chrome WebDriver. This data was also converted into a “soup” structure for further processing.
* **Data Integration:** 
  + After obtaining both static and dynamic data in “soup” structures, the complete dataset was created by replacing the “***section***” tag in the static data with the corresponding content from the dynamic data.
* **Data Storage:**
  + Finally, the combined web data was saved as an HTML file in the “raw\_data” folder. The snapshots of the resulting are shown below:

A screenshot of a computer

AI-generated content may be incorrect.

A computer screen shot of a computer code

AI-generated content may be incorrect.

## Data Filtering Task

In the “data\_filter.py” script, the program first reads the raw web data created and stored in the previous task. This is accomplished using the BeautifulSoup4 library, which parses the data into a soup structure. The data filtering process is divided into two parts: *filtering market data* and *filtering the latest news data*.

**Market Data Filtering:**

* **Identifying Parent Element:** The parent element containing market data was identified using the “div” tag with the class “MarketsBanner-marketData”.
* **Extracting Market Items:** Sub-elements containing market cards were located using the “a” tag with the class “MarketCard-container”.
* **Extracting Individual Market Item Details**:
  + For each market card, the following details were obtained:
    - **Symbol**: Retrieved from the “span” tag with the class “MarketCard-symbol”.
    - **Stock Position**: Retrieved from the “span” tag with the class “MarketCard-stockPosition”.
    - **Change Percentage**: Retrieved from the “span” tag with the class “MarketCard-changePct”.
* **Data Storage**: After extracting the above details for each market item, the data was saved as a CSV file in the “processed\_data” folder. The snapshot of the market result is provided below:

A screenshot of a graph

AI-generated content may be incorrect.

**Latest News Data Filtering:**

* **Identifying Parent Element:** The parent element containing the latest news data was identified using the ul tag with the class LatestNews-list.
* **Extracting News Items:** Sub-elements containing individual news items were located using the li tag with the class LatestNews-item.
* **Extracting Individual News Item Details**:
  + For each news item, the following details were obtained:
    - **Timestamp:** Retrieved from the “time” tag with the class “LatestNews-timestamp”.
    - **Title:** Retrieved from the headline element (“a” tag) with the class “LatestNews-headline”.
    - **Link:** Retrieved from the same headline element as the title.
* **Data Storage**: After extracting the above details for each market item, the data was saved as a CSV file in the “processed\_data” folder. The snapshot of the latest news result is provided below:

A screenshot of a computer

AI-generated content may be incorrect.