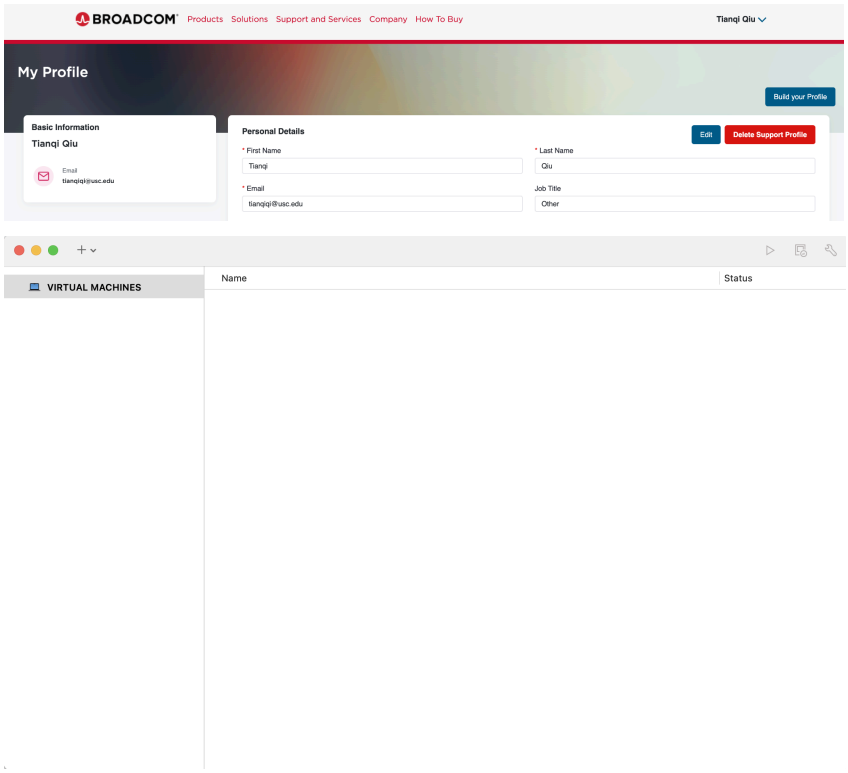
Github links: https://github.com/tianqi0301/DSCI560

1. Installation and Setup

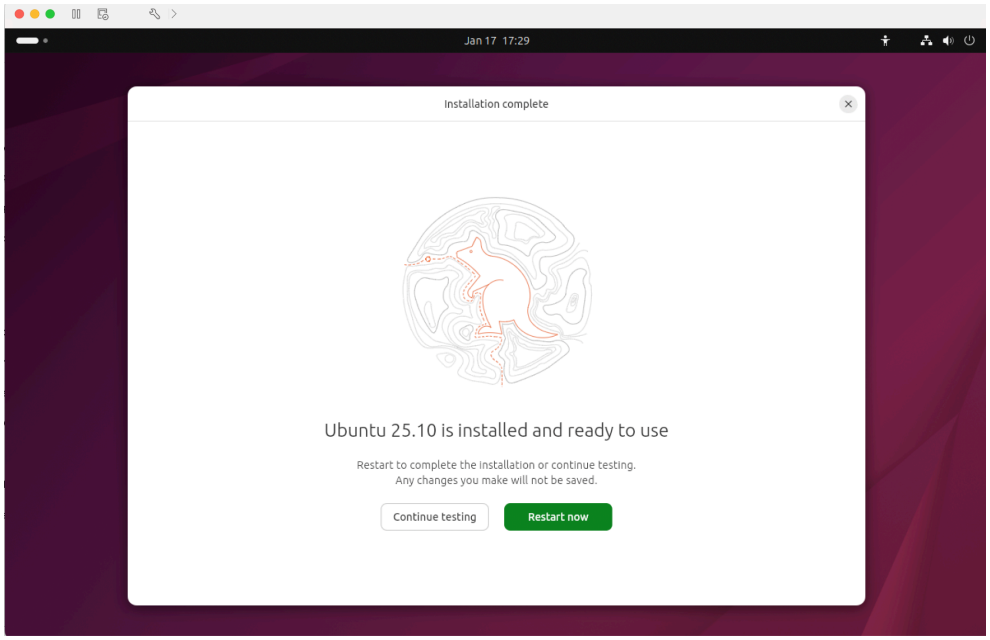
1.1. Register at Broadcom and then download and install VMware

I registered for a Broadcom account and downloaded VMware Fusion.



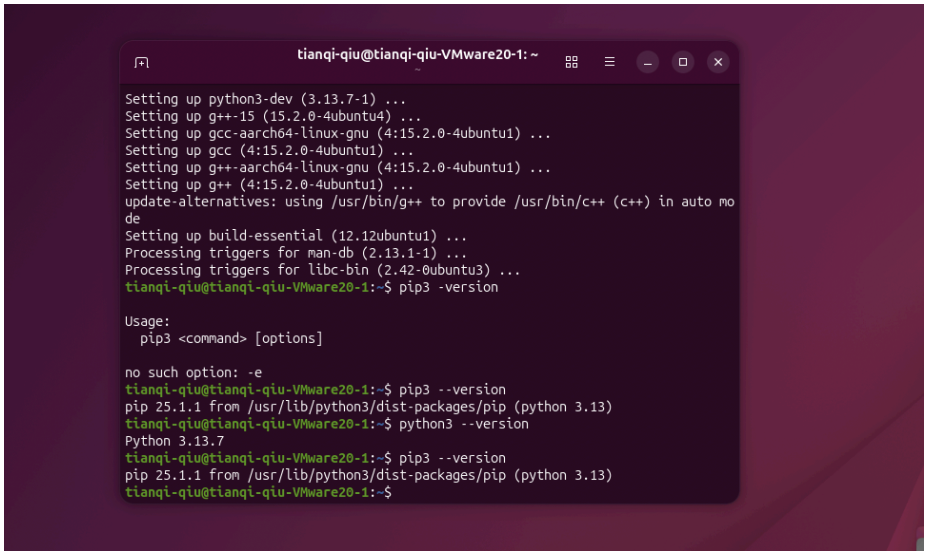
1.2. Download Ubuntu ISO Image

Ubuntu installed



1.3. Install Python on Linux

Python and pip3 installed



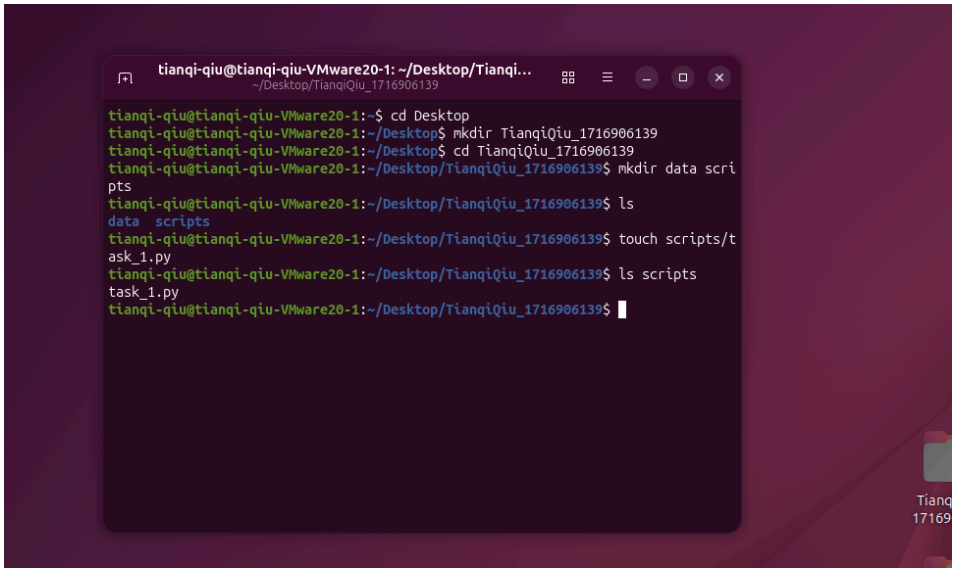
1.4. Tutorials

Go over python and linux tutorials

2. Get Familiar with Linux and Python

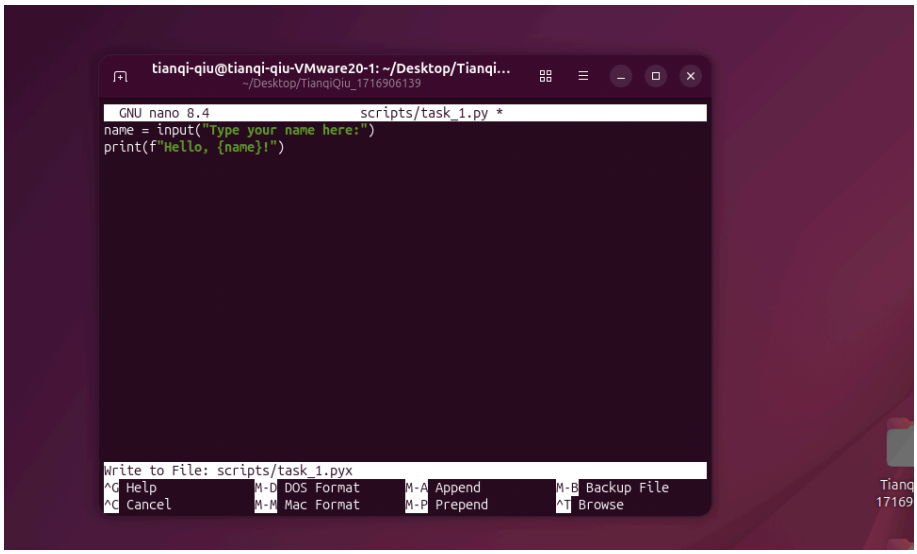
2.1. Playing around with Linux Terminal

Folders and file created

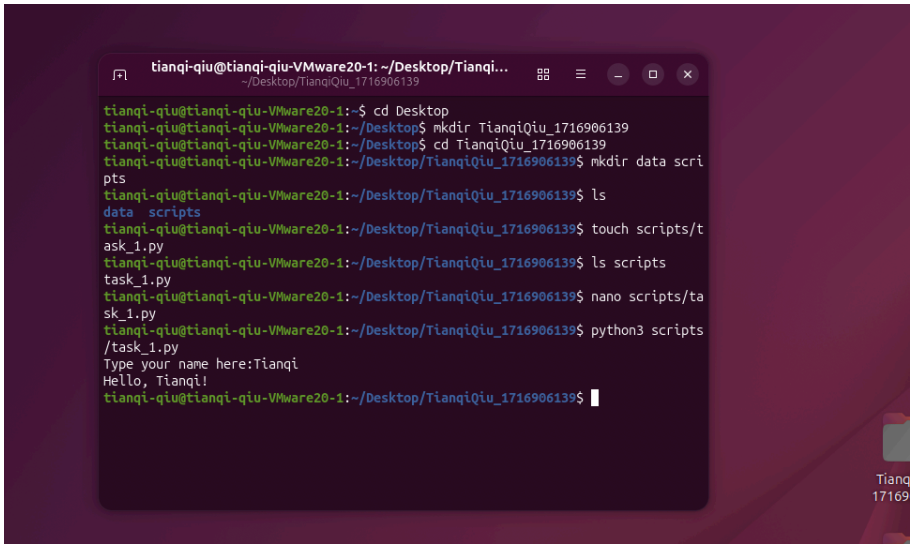


2.2. A basic Python Script

task1.py



Hello, name!



2.3. Python Web-scraping Task

scraper.py

```python
import logging
import os
import shutil

from bs4 import BeautifulSoup
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

logging.basicConfig(level=logging.INFO)

BASE_URL = "https://www.cnbc.com/world/?region=world"


class Path:
    file_dir = os.path.dirname(__file__)
    root_dir = os.path.dirname(file_dir)
    data_dir = os.path.join(root_dir, "data")


def find_chromium_binary():
    return shutil.which("chromium-browser") or shutil.which("chromium")


def initialize_driver():
    chromium_path = find_chromium_binary()
    if not chromium_path:
        raise RuntimeError("Chromium browser not found")

    options = Options()
    options.binary_location = chromium_path
    options.add_argument("--headless=new")
    options.add_argument("--no-sandbox")
    options.add_argument("--disable-dev-shm-usage")

    service = Service("/usr/bin/chromedriver")

    return webdriver.Chrome(service=service, options=options)
```

```python
def save_html_page(page: BeautifulSoup, save_to: str):
    market_banner = page.find("div", class_="MarketsBanner-marketData")
    latest_news = page.find("ul", class_="LatestNews-list")

    with open(save_to, "w", encoding="utf-8") as f:
        if market_banner:
            f.write(market_banner.prettify())
        f.write("\n\n")
        if latest_news:
            f.write(latest_news.prettify())


if __name__ == "__main__":
    driver = None
    try:
        logging.info("Initializing Chromium WebDriver...")
        driver = initialize_driver()

        logging.info("Opening CNBC webpage...")
        driver.get(BASE_URL)

        logging.info("Waiting for Market Cards to load...")
        WebDriverWait(driver, 60).until(
            EC.presence_of_element_located((By.CLASS_NAME, "MarketCard-row"))
        )

        soup = BeautifulSoup(driver.page_source, "html.parser")

        output_path = os.path.join(
            Path.data_dir, "raw_data", "web_data.html"
        )

        save_html_page(soup, output_path)

        logging.info("Saved HTML to %s", output_path)

        with open(output_path, "r", encoding="utf-8") as f:
            for _ in range(10):
                print(f.readline().strip())

    except Exception as e:
        logging.error("Unable to fetch data from page: %s", e)

    finally:
        if driver:
            driver.quit()
```

first 10 lines

```
(venv) tianqi-qiu@tianqi-qiu-VMware20-1:~/Desktop/TianqiQiu_1716906139/scripts$ head -n 10 ../data/raw_data/web_data.html
<div class="MarketsBanner-marketData" id="market-data-scroll-container">
 <a class="MarketCard-container MarketCard-down" href="//www.cnbc.com/quotes/.STOXX">
  <div class="MarketCard-row">
   <span class="MarketCard-symbol">
    STOXX600*
   </span>
   <span class="MarketCard-stockPosition">
    614.38
   </span>
  </div>
(venv) tianqi-qiu@tianqi-qiu-VMware20-1:~/Desktop/TianqiQiu_1716906139/scripts$
```

2.4. Data Filtering Task

data_filter.py

```
  GNU nano 8.4
import logging
import os

import pandas as pd
from bs4 import BeautifulSoup
from pydantic import BaseModel

logging.basicConfig(level=logging.INFO)


class Path:
    file_dir = os.path.dirname(__file__)
    root_dir = os.path.dirname(file_dir)

    data_dir = os.path.join(root_dir, "data")
    processed_data_dir = os.path.join(data_dir, "processed_data")


def read_parse_raw_data(path):
    logging.info("Reading and parsing the raw HTML file...")
    with open(path, "r", encoding="utf-8") as f:
        html_content = f.read()
    return BeautifulSoup(html_content, "html.parser")


class NewsItem(BaseModel):
    timestamp: str
    title: str
    link: str


class MarketCard(BaseModel):
    symbol: str
    stock_position: float
    change_pts: float


def filter_latest_news(page):
    logging.info("Filtering Latest News fields...")

    latest_news = page.find("ul", class_="LatestNews-list")
    latest_news = latest_news.find_all("li", class_="LatestNews-item")

    news_items = []
    for news_item in latest_news:
        headline = news_item.find("a", class_="LatestNews-headline")
        timestamp = news_item.find("time", class_="LatestNews-timestamp")

        if headline and timestamp:
            item = NewsItem(
                timestamp=timestamp.text.strip(),
                title=headline.get("title").strip(),
                link=headline.get("href").strip(),
            )
            news_items.append(item.model_dump())

    logging.info("Finished filtering Latest News.")
    return news_items
```

```python
def filter_market_banner(page):
    logging.info("Filtering Market Banner fields...")

    market_banner = page.find("div", class_="MarketsBanner-marketData")
    market_cards = market_banner.find_all("a", class_="MarketCard-container")

    cards = []
    for card in market_cards:
        symbol = card.find("span", class_="MarketCard-symbol")
        stock_position = card.find("span", class_="MarketCard-stockPosition")
        change_pct = card.find("span", class_="MarketCard-changesPct")

        if symbol and stock_position and change_pct:
            item = MarketCard(
                symbol=symbol.text.strip(),
                stock_position=float(stock_position.text.strip().replace(",", "")),
                change_pts=float(
                        change_pct.text.strip()
                        .replace("%", "")
                        .replace(",", "")
                ),
            )
            cards.append(item.model_dump())

    logging.info("Finished filtering Market Banner.")
    return cards

if __name__ == "__main__":
    EXTRACTED_HTML_PATH = os.path.join(Path.data_dir, "raw_data", "web_data.html")
    soup = read_parse_raw_data(EXTRACTED_HTML_PATH)

    logging.info("Processing Latest News data...")
    latest_news = filter_latest_news(soup)
    logging.info("Storing Latest News data to CSV...")
    pd.DataFrame(latest_news).to_csv(
        os.path.join(Path.processed_data_dir, "news_data.csv"), index=False
    )
    logging.info("Latest News CSV created.")

    logging.info("Processing Market data...")
    market_banners = filter_market_banner(soup)
    logging.info("Storing Market data to CSV...")
    pd.DataFrame(market_banners).to_csv(
        os.path.join(Path.processed_data_dir, "market_data.csv"), index=False
    )
    logging.info("Market data CSV created.")

    logging.info("Data filtering task completed successfully.")
```

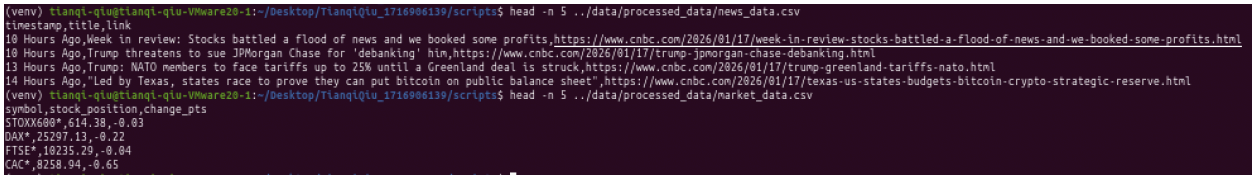
Terminal

Run successfully with appropriate messages to the console

```
(venv) tianqi-qiu@tianqi-qiu-VMware20-1:~/Desktop/TianqiQiu_1716906139/scripts$ python data_filter.py

INFO:root:Reading and parsing the raw HTML file...
INFO:root:Processing Latest News data...
INFO:root:Filtering Latest News fields...
INFO:root:Finished filtering Latest News.
INFO:root:Storing Latest News data to CSV...
INFO:root:Latest News CSV created.
INFO:root:Processing Market data...
INFO:root:Filtering Market Banner fields...
INFO:root:Finished filtering Market Banner.
INFO:root:Storing Market data to CSV...
INFO:root:Market data CSV created.
INFO:root:Data filtering task completed successfully.
```

First 5 lines of filtered news data and market data



3. Team Formation

I formed a team with Herun Kan and Shida Yan.