IMDb TOP 250 MOVIES SCRAPER

Project completed by:
Nithishwar T

INTRODUCTION:

In this project, we build a web scraper using Python and Selenium to extract IMDb's Top 250 movies. The script collects each movie's title, release year, and IMDb rating, and stores the data into a CSV file for future analysis. This project demonstrates skills in web scraping, data extraction, and data storage using Pandas.

.Prerequisites:

- **Software:**
  - Python 3.8+
  - Google Chrome browser
  - Microsoft Excel (or compatible spreadsheet software)
- **Python Libraries:**

  pip install selenium pandas

- **Basic Knowledge:**
  - Python programming
  - Basics of Excel file handling using Pandas

Step-by-Step Instructions:

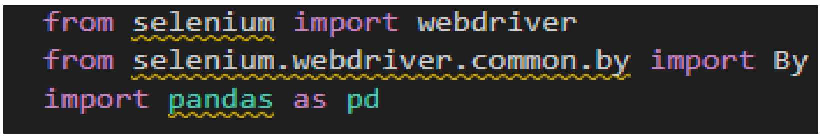
- **Step 1: Install required libraries:**

  pip install selenium pandas

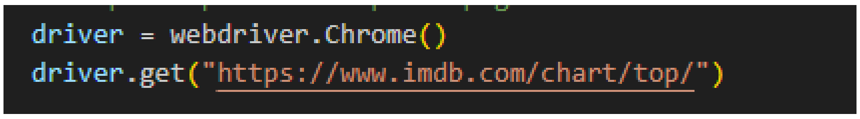
- **Step 2 — Create the script file:**

1. Inside your project folder, create a file named imdb_scraper.py.
2. The script will perform three main tasks:
   a) Open IMDb Top 250 movies page
   b) Scrape movie titles, years, and ratings
   c) Save results into a CSV file

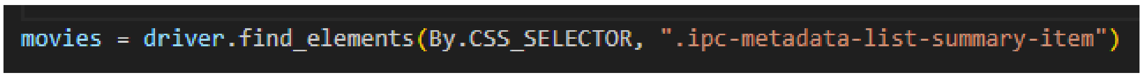
Step 4: Implementing the Script:


```python
from selenium import webdriver
from selenium.webdriver.common.by import By
import pandas as pd
```

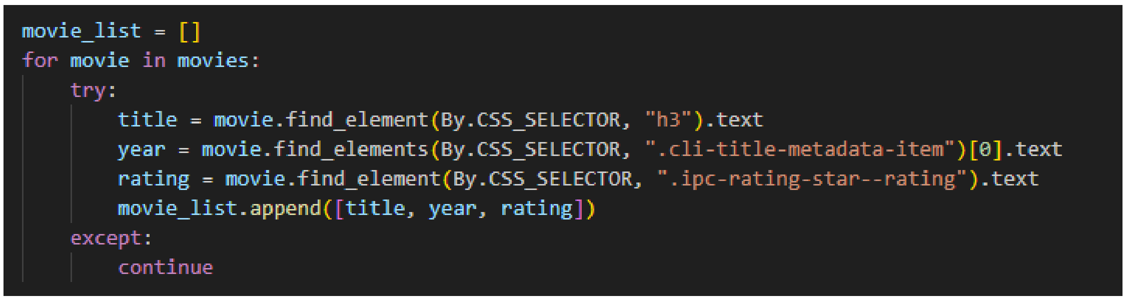
- **from selenium import webdriver** – lets Python control a web browser (like Chrome).
- **from selenium.webdriver.common.by import By** – gives ways to find elements on a webpage (by ID, class, CSS, etc.).
- **import pandas as pd** – brings in Pandas, which helps store and organize scraped data in tables/CSV format.


```python
driver = webdriver.Chrome()
driver.get("https://www.imdb.com/chart/top/")
```

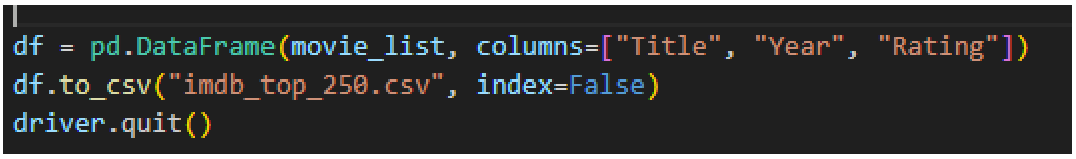
- **driver = webdriver.Chrome()** – launches a new Chrome browser window that Selenium can control.
- **driver.get("https://www.imdb.com/chart/top/**") – opens the IMDb Top 250 movies page in that browser.


```python
movies = driver.find_elements(By.CSS_SELECTOR, ".ipc-metadata-list-summary-item")
```

- **movies = driver.find_elements(By.CSS_SELECTOR, ".ipc-metadata-list-summary-item")** – finds **all movie cards** on the IMDb Top 250 page by using their CSS class (.ipc-metadata-list-summary-item) and stores them in a list.


```python
movie_list = []
for movie in movies:
    try:
        title = movie.find_element(By.CSS_SELECTOR, "h3").text
        year = movie.find_elements(By.CSS_SELECTOR, ".cli-title-metadata-item")[0].text
        rating = movie.find_element(By.CSS_SELECTOR, ".ipc-rating-star--rating").text
        movie_list.append([title, year, rating])
    except:
        continue
```

- **movie_list** = [] – creates an empty list to store movie details.
- **for movie in movies**: – loops through each movie card found on the IMDb page.
- **title** = movie.find_element(By.CSS_SELECTOR, "h3").text – extracts the movie title.
- **year** = movie.find_elements(By.CSS_SELECTOR, ".cli-title-metadata-item")[0].text – gets the release year from the metadata.
- **rating** = movie.find_element(By.CSS_SELECTOR, ".ipc-rating-star--rating").text – fetches the IMDb rating.
- **movie_list.append([title, year, rating])** – saves the collected info (title, year, rating) into the list.
- **except: continue** – if something goes wrong (like missing data), skip that movie and move to the next one.



```python
df = pd.DataFrame(movie_list, columns=["Title", "Year", "Rating"])
df.to_csv("imdb_top_250.csv", index=False)
driver.quit()
```

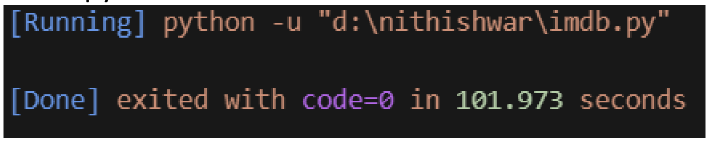
- **df = pd.DataFrame(movie_list, columns=["Title", "Year", "Rating"])** – converts the collected list of movies into a **Pandas DataFrame** (like a neat Excel table) with columns Title, Year, and Rating.
- **df.to_csv("imdb_top_250.csv", index=False**) – saves that table into a CSV file named **imdb_top_250.csv** (without row numbers).
- **driver.quit()** – closes the Chrome browser and ends the Selenium session.

FULL SCRIPT:

```python
from selenium import webdriver
from selenium.webdriver.common.by import By
import pandas as pd
driver = webdriver.Chrome()
driver.get("https://www.imdb.com/chart/top/")

movies = driver.find_elements(By.CSS_SELECTOR, ".ipc-metadata-list-summary-item")

movie_list = []
for movie in movies:
    try:
        title = movie.find_element(By.CSS_SELECTOR, "h3").text
        year = movie.find_elements(By.CSS_SELECTOR, ".cli-title-metadata-item")[0].text
        rating = movie.find_element(By.CSS_SELECTOR, ".ipc-rating-star--rating").text
        movie_list.append([title, year, rating])
    except:
        continue

df = pd.DataFrame(movie_list, columns=["Title", "Year", "Rating"])
df.to_csv("imdb_top_250.csv ", index=False)
driver.quit()
```

Running the Script

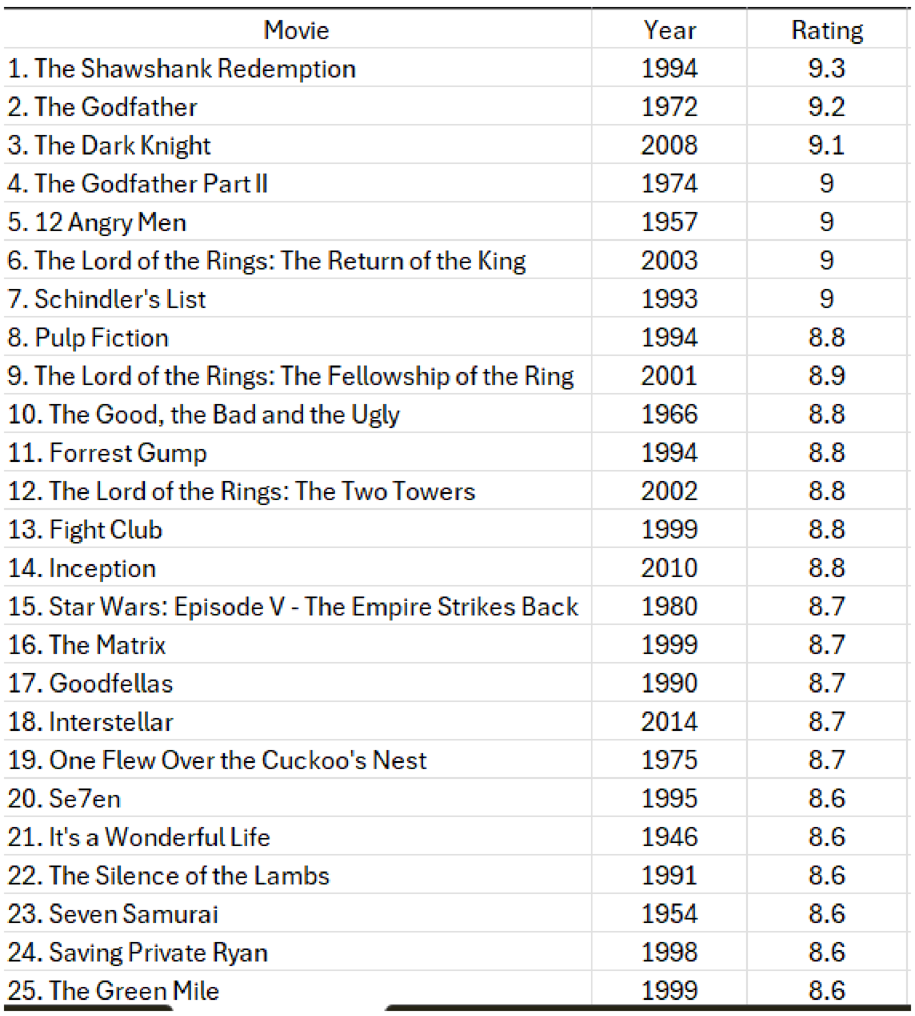
imdb.py



```
[Running] python -u "d:\nithishwar\imdb.py"

[Done] exited with code=0 in 101.973 seconds
```

OUTPUT:

- **imdb_top_250.csv:** Created/updated with new entries.



Movie	Year	Rating
1. The Shawshank Redemption	1994	9.3
2. The Godfather	1972	9.2
3. The Dark Knight	2008	9.1
4. The Godfather Part II	1974	9
5. 12 Angry Men	1957	9
6. The Lord of the Rings: The Return of the King	2003	9
7. Schindler's List	1993	9
8. Pulp Fiction	1994	8.8
9. The Lord of the Rings: The Fellowship of the Ring	2001	8.9
10. The Good, the Bad and the Ugly	1966	8.8
11. Forrest Gump	1994	8.8
12. The Lord of the Rings: The Two Towers	2002	8.8
13. Fight Club	1999	8.8
14. Inception	2010	8.8
15. Star Wars: Episode V - The Empire Strikes Back	1980	8.7
16. The Matrix	1999	8.7
17. Goodfellas	1990	8.7
18. Interstellar	2014	8.7
19. One Flew Over the Cuckoo's Nest	1975	8.7
20. Se7en	1995	8.6
21. It's a Wonderful Life	1946	8.6
22. The Silence of the Lambs	1991	8.6
23. Seven Samurai	1954	8.6
24. Saving Private Ryan	1998	8.6
25. The Green Mile	1999	8.6

Troubleshooting Guide:

Issue	Possible Cause	Solution
No data scraped	Table not loaded in time	Increase WebDriverWait time.
CSV not created	Wrong working directory	Use **os.getcwd()** to confirm path.
ChromeDriver error	Version mismatch	Run **pip install --upgrade webdriver-manager**

End of Document:

This concludes the **IMDb Top 250 Movies Scraper** mini-project.
By following the steps above, you can:

- Scrape live IMDb Top 250 movie data (Title, Year, Rating)
- Store it neatly in a CSV file for analysis
- Automate the process for regular updates

Possible Enhancements

- Scrape extra details like directors, genres, or runtime
- Sort or filter movies by rating or year
- Build a dashboard using Excel, Power BI, or Tableau