IMDb Movie Rating Scraper

Author: KAYATHRI

Type: Individual

Project Description

The IMDb Movie Rating Scraper is a Python mini-project that automatically collects the Top 25 movies from IMDb along with their ratings. Using the requests and BeautifulSoup libraries, the program fetches data from the IMDb Top Chart page, extracts the movie titles and ratings, and saves them into an Excel file. This project demonstrates basic web scraping, HTML parsing, and data handling using Python.

Features

- Automatically scrapes IMDb Top 25 movie titles and ratings.
- Extracts data directly from IMDb without manual searching.
- Saves the collected data into an Excel file for easy viewing.
- Fast, accurate, and works without opening a browser.
- Uses clean and modern IMDb selectors (2025 layout compatible).
- Prints all movie details in the console as well.

Technologies Used

- Python – Main programming language
- Requests – To fetch IMDb webpage
- BeautifulSoup (bs4) – To parse HTML and extract data
- Pandas – To store data and export it to Excel
- IMDb Website – Data source

Outcomes of the Project

- Successfully retrieves the Top 25 IMDb movie ratings.
- Learner understands how web scraping works.
- Gives hands-on practice with Python libraries like requests, bs4, and pandas.
- Produces a structured Excel file with Rank, Title, and Rating.
- Helps in data analysis and movie comparison.

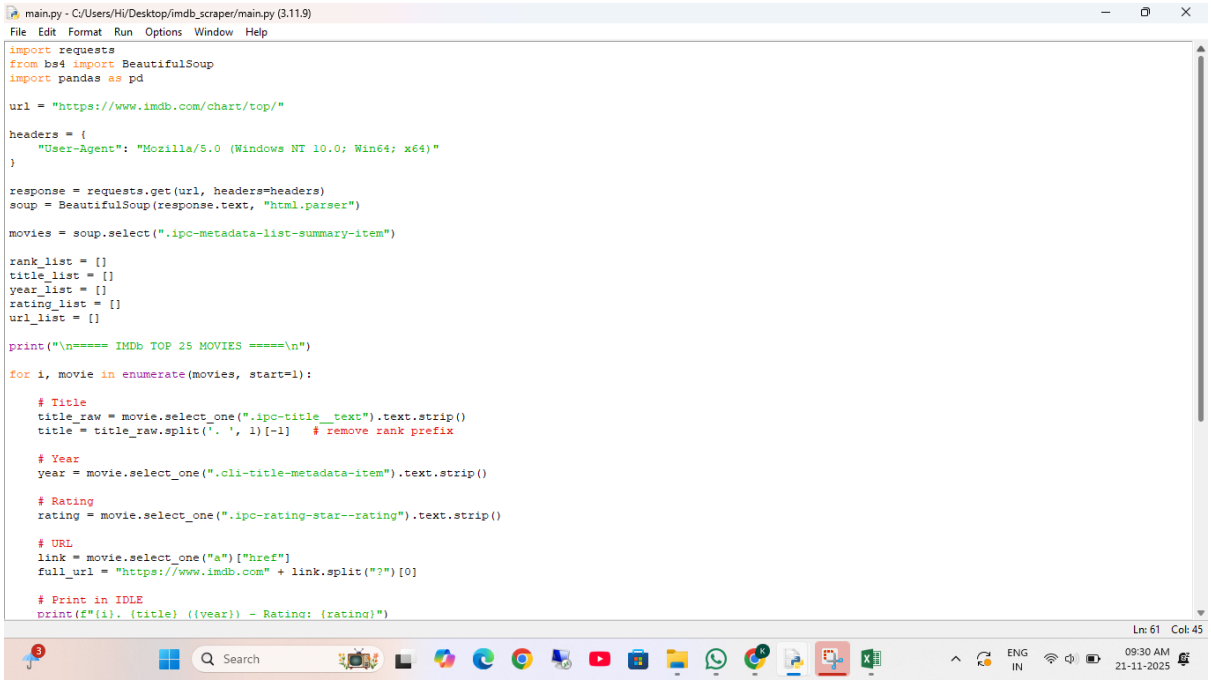
Advantages

- Saves time by automating manual movie rating searching.
- Can be extended to scrape more details (year, genre, summary, poster, etc.).
- Works without API or paid tools — free and simple.
- Helps beginners understand real-world data scraping.
- Useful for mini-projects, internships, and portfolio building.
- Data can be reused for analytics, visualization, or machine learning.

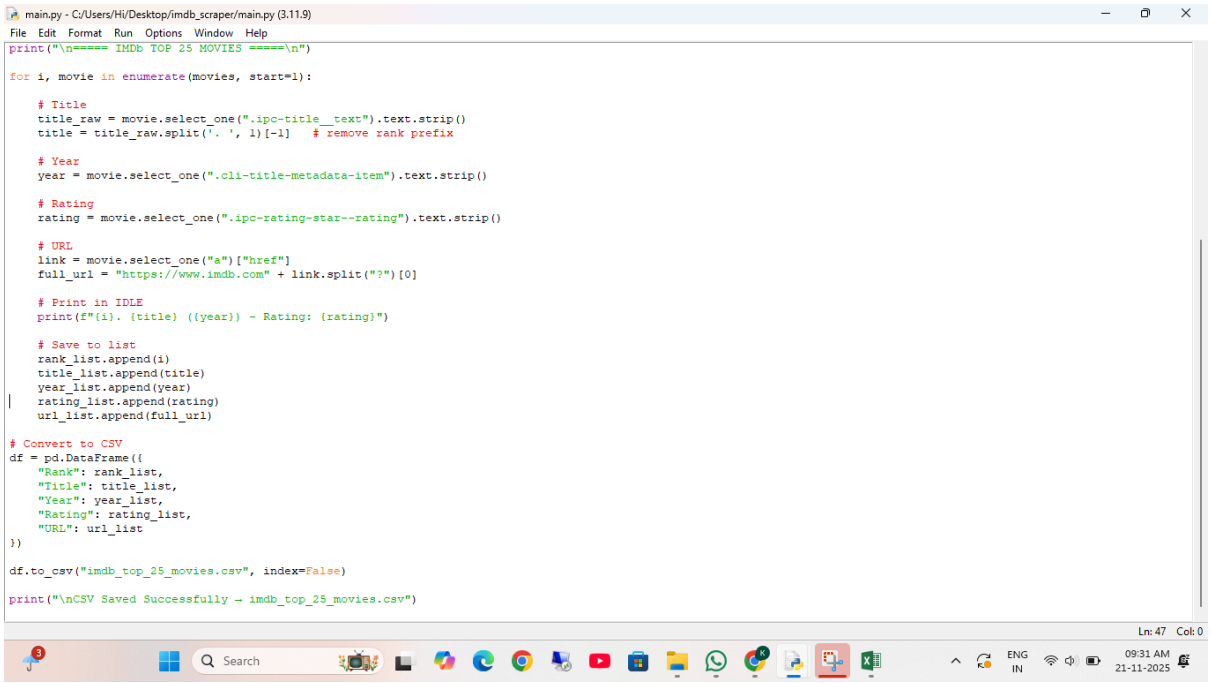
Future Enhancement

- More details like year, genre, actors, director can be added.
- Scraper can be upgraded to take user input for any movie name.
- Top 50 or 100 movies can also be scraped instead of 25.
- Data can be saved in PDF or CSV formats.
- A simple GUI app can be created for easy use.
- Tamil-only or English-only filter options can be added.

SOURCE CODE WITH IMPLEMENTATION SCREEN

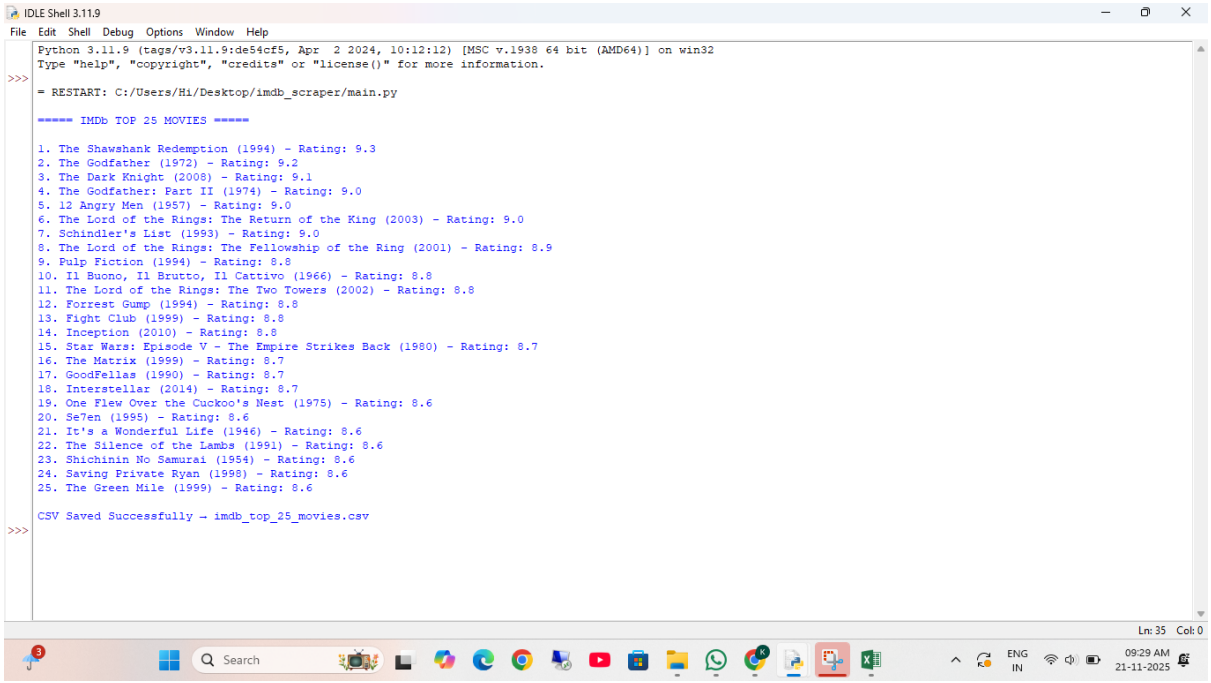


```
main.py - C:/Users/Hi/Desktop/imdb_scraper/main.py (3.11.9)
File  Edit  Format  Run  Options  Window  Help
import requests
from bs4 import BeautifulSoup
import pandas as pd

url = "https://www.imdb.com/chart/top/"

headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)"
}

response = requests.get(url, headers=headers)
soup = BeautifulSoup(response.text, "html.parser")

movies = soup.select(".ipc-metadata-list-summary-item")

rank_list = []
title_list = []
year_list = []
rating_list = []
url_list = []

print("\n===== IMDb TOP 25 MOVIES =====\n")

for i, movie in enumerate(movies, start=1):

    # Title
    title_raw = movie.select_one(".ipc-title__text").text.strip()
    title = title_raw.split('. ', 1)[-1]   # remove rank prefix

    # Year
    year = movie.select_one(".cli-title-metadata-item").text.strip()

    # Rating
    rating = movie.select_one(".ipc-rating-star--rating").text.strip()

    # URL
    link = movie.select_one("a")["href"]
    full_url = "https://www.imdb.com" + link.split("?")[0]

    # Print in IDLE
    print(f"{i}. {title} ({year}) - Rating: {rating}")
```

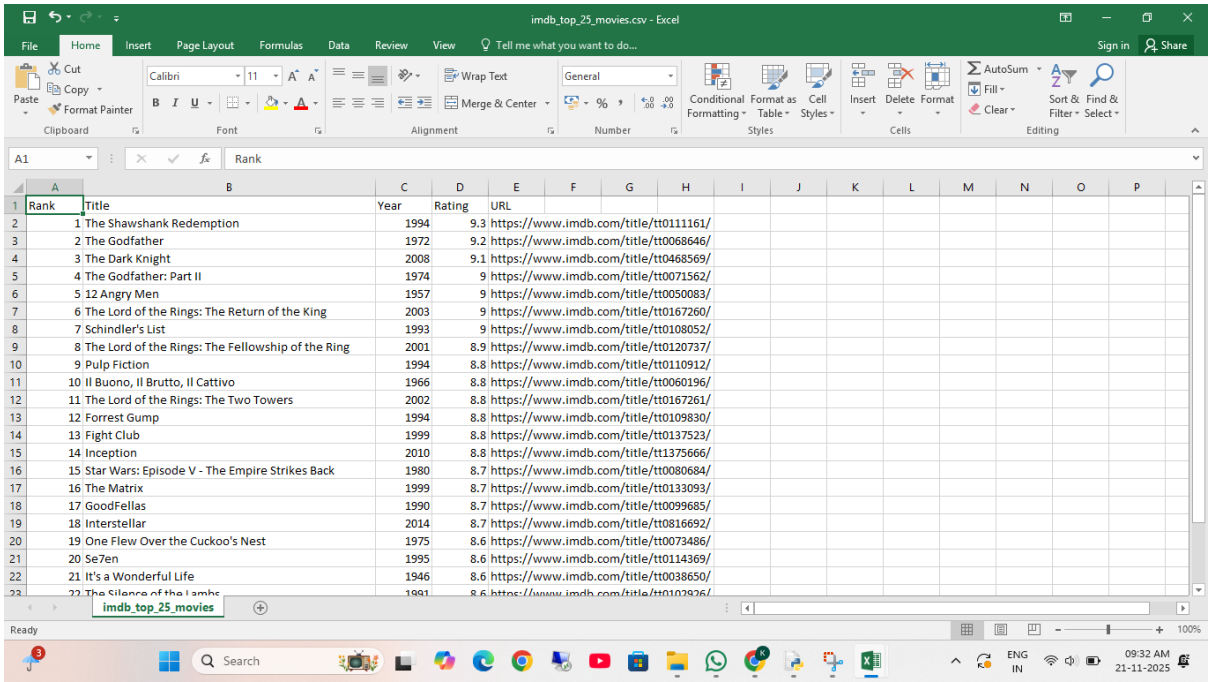



```
main.py - C:/Users/Hi/Desktop/imdb_scraper/main.py (3.11.9)
File  Edit  Format  Run  Options  Window  Help
print("\n===== IMDb TOP 25 MOVIES =====\n")

for i, movie in enumerate(movies, start=1):

    # Title
    title_raw = movie.select_one(".ipc-title__text").text.strip()
    title = title_raw.split('. ', 1)[-1]   # remove rank prefix

    # Year
    year = movie.select_one(".cli-title-metadata-item").text.strip()

    # Rating
    rating = movie.select_one(".ipc-rating-star--rating").text.strip()

    # URL
    link = movie.select_one("a")["href"]
    full_url = "https://www.imdb.com" + link.split("?")[0]

    # Print in IDLE
    print(f"{i}. {title} ({year}) - Rating: {rating}")

    # Save to list
    rank_list.append(i)
    title_list.append(title)
    year_list.append(year)
    rating_list.append(rating)
    url_list.append(full_url)

# Convert to CSV
df = pd.DataFrame({
    "Rank": rank_list,
    "Title": title_list,
    "Year": year_list,
    "Rating": rating_list,
    "URL": url_list
})

df.to_csv("imdb_top_25_movies.csv", index=False)

print("\nCSV Saved Successfully → imdb_top_25_movies.csv")
```

Result



CSV Format Screenshot



Rank	Title	Year	Rating	URL
1	The Shawshank Redemption	1994	9.3	https://www.imdb.com/title/tt0111161/
2	The Godfather	1972	9.2	https://www.imdb.com/title/tt0068646/
3	The Dark Knight	2008	9.1	https://www.imdb.com/title/tt0468569/
4	The Godfather: Part II	1974	9	https://www.imdb.com/title/tt0071562/
5	12 Angry Men	1957	9	https://www.imdb.com/title/tt0050083/
6	The Lord of the Rings: The Return of the King	2003	9	https://www.imdb.com/title/tt0167260/
7	Schindler's List	1993	9	https://www.imdb.com/title/tt0108052/
8	The Lord of the Rings: The Fellowship of the Ring	2001	8.9	https://www.imdb.com/title/tt0120737/
9	Pulp Fiction	1994	8.8	https://www.imdb.com/title/tt0110912/
10	Il Buono, Il Brutto, Il Cattivo	1966	8.8	https://www.imdb.com/title/tt0060196/
11	The Lord of the Rings: The Two Towers	2002	8.8	https://www.imdb.com/title/tt0167261/
12	Forrest Gump	1994	8.8	https://www.imdb.com/title/tt0109830/
13	Fight Club	1999	8.8	https://www.imdb.com/title/tt0137523/
14	Inception	2010	8.8	https://www.imdb.com/title/tt1375666/
15	Star Wars: Episode V - The Empire Strikes Back	1980	8.7	https://www.imdb.com/title/tt0080684/
16	The Matrix	1999	8.7	https://www.imdb.com/title/tt0133093/
17	GoodFellas	1990	8.7	https://www.imdb.com/title/tt0099685/
18	Interstellar	2014	8.7	https://www.imdb.com/title/tt0816692/
19	One Flew Over the Cuckoo's Nest	1975	8.6	https://www.imdb.com/title/tt0073486/
20	Se7en	1995	8.6	https://www.imdb.com/title/tt0114369/
21	It's a Wonderful Life	1946	8.6	https://www.imdb.com/title/tt0038650/
22	The Silence of the Lambs	1991	8.6	https://www.imdb.com/title/tt0102926/

Source Code

```
import requests
from bs4 import BeautifulSoup
import pandas as pd
url = "https://www.imdb.com/chart/top/"
headers = {
   "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)"
}
response = requests.get(url, headers=headers)
soup = BeautifulSoup(response.text, "html.parser")
movies = soup.select(".ipc-metadata-list-summary-item")
rank_list = []
title_list = []
year_list = []
rating_list = []
url_list = []
print("\n===== IMDb TOP 25 MOVIES =====\n")
for i, movie in enumerate(movies, start=1):

 # Title
   title_raw = movie.select_one(".ipc-title__text").text.strip()
   title = title_raw.split('. ', 1)[-1]   # remove rank prefix
  # Year
   year = movie.select_one(".cli-title-metadata-item").text.strip()
```

```
# Rating
    rating = movie.select_one(".ipc-rating-star--rating").text.strip()
  # URL
    link = movie.select_one("a")["href"]
    full_url = "https://www.imdb.com" + link.split("?")[0]
 # Print in IDLE
    print(f"{i}. {title} ({year}) - Rating: {rating}")
    # Save to list
    rank_list.append(i)
    title_list.append(title)
    year_list.append(year)
    rating_list.append(rating)
    url_list.append(full_url)
# Convert to CSV
df = pd.DataFrame({
    "Rank": rank_list,
    "Title": title_list,
    "Year": year_list,
    "Rating": rating_list,
    "URL": url_list
})

df.to_csv("imdb_top_250_movies.csv", index=False)

print("\nCSV Saved Successfully → imdb_top_250_movies.csv")
```

Conclusion

The IMDb Movie Rating Scraper project successfully extracts the Top 25 movie titles and ratings from IMDb using Python. It shows how web scraping can be used to collect real-time data automatically without manual effort. Through this project, we learn the basics of sending web requests, parsing HTML, extracting useful information, and saving it in an Excel file. Overall, it is a useful and beginner-friendly project that demonstrates how Python can automate data collection efficiently.