My Selenium Learning

When I first started with **Selenium**, I thought: *"Okay, this tool just clicks stuff for me."* But as I went deeper, I realized it's like a little robot that drives your browser, opens pages, clicks buttons, fills forms, and even grabs data behind the scenes.

Here's how I approached it step by step:

Basics I Learned About Selenium

1. **Setup:** Install Selenium and get ChromeDriver running.

   ```
   from selenium import webdriver

   driver = webdriver.Chrome()
   ```

2. **Navigation:** Use driver.get("url") to load a website.
3. **Finding Elements:**
   - find_element or find_elements let you grab stuff from the page.
   - You can locate elements using different strategies:
     - By.ID
     - By.CLASS_NAME
     - By.CSS_SELECTOR (my favorite, like using query selectors in JS)
     - By.XPATH
4. **Interacting:** You can .click(), .send_keys(), or even scroll down with JavaScript.

The MovieBox Adventure

I decided to scrape data from a site called *MovieBox*.
The idea was simple: grab titles, years, ratings, and images of **2000 movies**.

But here's the fun part: **it wasn't static at all!**

- At first, I thought I just needed to scroll down and load more movies.
- The page seemed to behave like *infinite scrolling*, so I kept trying to automate scrolling.
- BUT — here's the annoying (and hilarious) discovery: every time I scrolled, a "Load More" button appeared… but only for a **fraction of a second**.
- Before I could even think of clicking it, the page auto-refreshed itself and loaded the new content.
- No button click required at all! It was a mix of **infinite scroll + auto-refresh** that totally confused me for hours.

At the end of it, I learned that sites can load content in different ways, and you really have to test and observe carefully.

Content Loading Types I Encountered

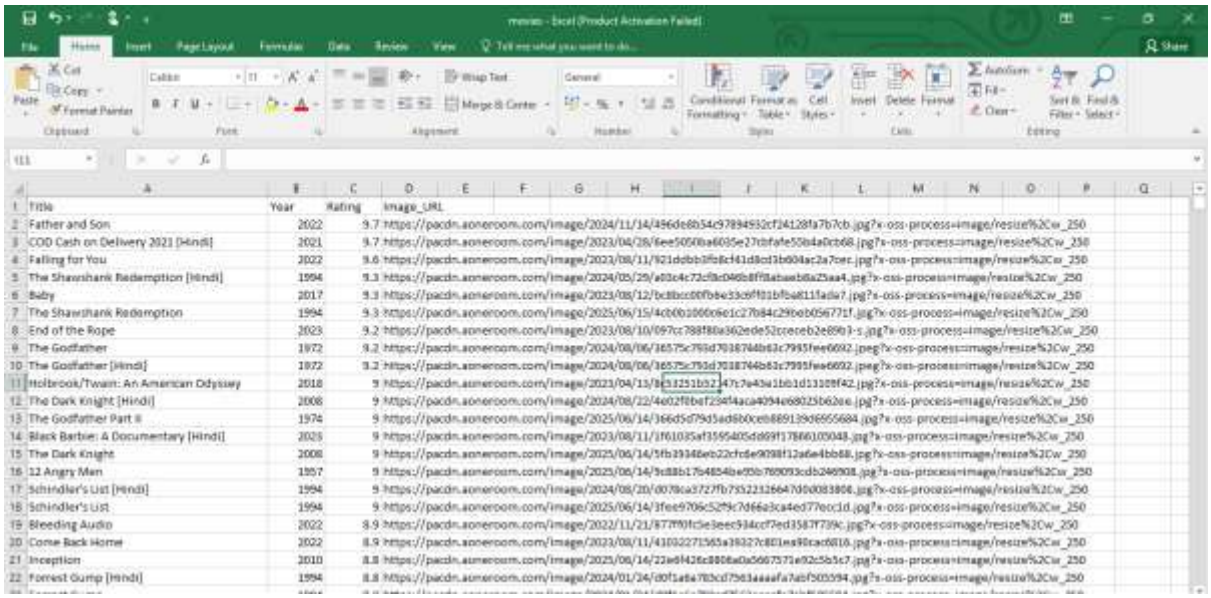
1. **Static content** → The page loads everything at once, easy to scrape.
2. **Load More button** → You scroll or click, then new data is fetched.
3. **Infinite scroll / auto-refresh** → Content is fetched as you scroll (sometimes with sneaky tricks like MovieBox did).

What I Took Away

- Selenium isn't just about "click this, type that" — it's about observing how a website behaves.
- Websites can trick you with how they load data. Sometimes it looks like scrolling, but actually it's automatic content fetching.
- Debugging this felt frustrating at first, but in the end, it was a fun detective story.

So yeah, that was my first proper Selenium project — a mix of curiosity, trial & error, and a weird movie site that kept me guessing.

Dataset



Title	Year	Rating	Image_URL
Father and Son	2022	9.7	https://pacdn.aoneroom.com/image/2024/11/14/496de8b54c97894932cf34128fa7b7cb.jpg?x-oss-process=image/resize%2Cw_250
COD Cash on Delivery 2021 [Hindi]	2021	9.7	https://pacdn.aoneroom.com/image/2023/04/28/6ee50508a6035e27cbfafe550b4a0cb68.jpg?x-oss-process=image/resize%2Cw_250
Falling for You	2022	9.6	https://pacdn.aoneroom.com/image/2023/08/11/921ddbb3fb8c461d8cd3b604ac2a7cec.jpg?x-oss-process=image/resize%2Cw_250
The Shawshank Redemption [Hindi]	1994	9.3	https://pacdn.aoneroom.com/image/2024/05/29/a03c4c72cf8c0460bff8abaeb8a25aa4.jpg?x-oss-process=image/resize%2Cw_250
Baby	2017	9.3	https://pacdn.aoneroom.com/image/2023/08/12/bc8bcc00fb6e33c6ff01bf0a8111a8a7.jpg?x-oss-process=image/resize%2Cw_250
The Shawshank Redemption	1994	9.3	https://pacdn.aoneroom.com/image/2025/06/15/4cb0b1066b6e1c27b84c29beb056771f.jpg?x-oss-process=image/resize%2Cw_250
End of the Rope	2023	9.2	https://pacdn.aoneroom.com/image/2023/08/10/097cc788f88a362ede52ceeceb2e89b3-s.jpg?x-oss-process=image/resize%2Cw_250
The Godfather	1972	9.2	https://pacdn.aoneroom.com/image/2024/08/06/36575c793d7018744b63c7995fee6692.jpeg?x-oss-process=image/resize%2Cw_250
The Godfather [Hindi]	1972	9.2	https://pacdn.aoneroom.com/image/2024/08/06/36575c793d7018744b63c7995fee6692.jpeg?x-oss-process=image/resize%2Cw_250
Holbrook/Twain: An American Odyssey	2018	9	https://pacdn.aoneroom.com/image/2023/04/13/8633251b32247c7e43d1b61d13108f42.jpg?x-oss-process=image/resize%2Cw_250
The Dark Knight [Hindi]	2008	9	https://pacdn.aoneroom.com/image/2024/08/22/4e02f8baf2a4aca4094e68025b62ae.jpg?x-oss-process=image/resize%2Cw_250
The Godfather Part II	1974	9	https://pacdn.aoneroom.com/image/2025/06/14/366d5d79d5ad6b0ceb88913986656684.jpg?x-oss-process=image/resize%2Cw_250
Black Barbie: A Documentary [Hindi]	2023	9	https://pacdn.aoneroom.com/image/2023/08/11/1f61035af3595405dd69f17866105048.jpg?x-oss-process=image/resize%2Cw_250
The Dark Knight	2008	9	https://pacdn.aoneroom.com/image/2025/06/14/5fb33446eb22cfc6e9098f12a6e4bb88.jpg?x-oss-process=image/resize%2Cw_250
12 Angry Men	1957	9	https://pacdn.aoneroom.com/image/2025/06/14/9c88b17b4854bef5b769093cdb246908.jpg?x-oss-process=image/resize%2Cw_250
Schindler's List [Hindi]	1994	9	https://pacdn.aoneroom.com/image/2024/08/20/d078ca372f7b7352232664765d083806.jpg?x-oss-process=image/resize%2Cw_250
Schindler's List	1994	9	https://pacdn.aoneroom.com/image/2025/06/14/3fee9706c52f9c7d66a3ca4ed77ecc1d.jpg?x-oss-process=image/resize%2Cw_250
Bleeding Audio	2022	8.9	https://pacdn.aoneroom.com/image/2022/11/21/877ff0fc5e3eec934ccf7ed3587f739c.jpg?x-oss-process=image/resize%2Cw_250
Come Back Home	2022	8.9	https://pacdn.aoneroom.com/image/2023/08/11/41032271565a39327c801ea98cac6816.jpg?x-oss-process=image/resize%2Cw_250
Inception	2010	8.8	https://pacdn.aoneroom.com/image/2025/06/14/22e6f426c8808a0a5667571e92c5b5c7.jpg?x-oss-process=image/resize%2Cw_250
Forrest Gump [Hindi]	1994	8.8	https://pacdn.aoneroom.com/image/2024/01/24/d0f1a6a783cd7563aaaafa7abf305594.jpg?x-oss-process=image/resize%2Cw_250

Code

```
import pandas as pd
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
```

```
driver = webdriver.Chrome()
driver.get("https://moviebox.ph/web/film?type=/home/movieFilter&tabId=2&classify=All&country=United+States&genre=All&sort=Rating&year=All")

movies_data = []
seen = set()

target_count = 2000

while len(movies_data) < target_count:

    WebDriverWait(driver, 10).until(
        EC.presence_of_all_elements_located((By.CSS_SELECTOR, "div.video-item.pc-card"))
    )

    cards = driver.find_elements(By.CSS_SELECTOR, "div.video-item.pc-card")

    for card in cards:
        try:
            title = card.find_element(By.CSS_SELECTOR, "div.card-title.pot").text.strip()
            year = card.find_element(By.CSS_SELECTOR, "div.pc-year").text.strip()
            rating = card.find_element(By.CSS_SELECTOR, "span.pc-rate").text.strip()

            img_elem = card.find_element(By.CSS_SELECTOR, "div.lazy-img img")
            image = (
                img_elem.get_attribute("src")
                or img_elem.get_attribute("data-src")
                or img_elem.get_attribute("data-original")
                or ""
            )

            key = (title, year)
            if key not in seen:
                seen.add(key)
                movies_data.append({
                    "Title": title,
                    "Year": year,
                    "Rating": rating,
                    "Image_URL": image
                })
        except:
            continue

    print(f"Collected: {len(movies_data)} movies")
```

```
    if len(movies_data) >= target_count:
        break


    try:
        see_more_btn = WebDriverWait(driver, 5).until(
            EC.element_to_be_clickable((By.CSS_SELECTOR, "div.pc-load-more"))
        )
        driver.execute_script("arguments[0].click();", see_more_btn)
        time.sleep(3)
    except:
        print("No more 'See More' button. Stopping.")
        break

df = pd.DataFrame(movies_data[:target_count])
df.to_csv("Selenium/movies.csv", index=False, encoding="utf-8")
print(f"✅ Saved {len(df)} movies to movies.csv")

driver.quit()
```