Table of Contents

Problem Statement:

To develop a robust web scraping application using Python to extract data from dynamic websites. The task involves scraping data from three different pages of the website "scrapethissite.com"

Scrapping Link:
1. https://www.scrapethissite.com/pages/ajax-javascript/#2015
2. https://www.scrapethissite.com/pages/forms/
3. https://www.scrapethissite.com/pages/advanced/

Setup and Installation:

Before running the scraping application, ensure that Python 3.x is installed on your system. Additionally, install the following dependencies:

requests
BeautifulSoup
pymongo (for MongoDB)

Installation commands:

- pip install requests
- pip install BeautifulSoup
- pip install pymongo

Create a virtual environment and install the required packages:

python -m venv venv
source venv/bin/activate # On Windows use `venv\Scripts\activate`

Technology and Tech Stack Used:

1. Python
2. VSCode IDE
3. MongoDB

Libraries Used:

requests: For making HTTP requests to fetch web pages and their content.

BeautifulSoup: For parsing HTML and XML documents, facilitating data extraction from web pages.

concurrent.futures.ThreadPoolExecutor: Utilized for concurrent execution of scraping scripts, improving performance by executing multiple tasks simultaneously.

subprocess: For running external processes, facilitating the execution of scraping scripts from the main script.

logging: Used for implementing logging functionality, allowing the application to log errors and messages to a file for debugging purposes.

pymongo: For interacting with MongoDB database, enabling data insertion and manipulation within the database.

requests.adapters.HTTPAdapter: For customizing HTTP requests, including setting up retries for handling network errors and timeouts.

requests.packages.urllib3.util.retry.Retry: Used with HTTPAdapter for defining retry strategies for HTTP requests.

MongoDB Atlas Setup:

1. Sign up/Login to MongoDB Atlas:

Go to the MongoDB Atlas website: MongoDB Atlas.
Sign up for an account or log in if you already have one.

2. Create a New Cluster:

Once logged in, click on the "Build a New Cluster" button.
Choose your preferred cloud provider, region, and cluster tier (e.g., M0 Sandbox, M2/M5/M10 shared, or dedicated cluster).
Configure additional settings such as cluster name, disk size, etc., according to your requirements.
Click on the "Create Cluster" button to create your cluster. This process may take a few minutes.

3. Create a Database User:

Go to the "Database Access" tab in the left sidebar.
Click on the "Add New Database User" button.
Enter the username and password for the new database user.
Assign appropriate roles to the user based on your project requirements

4. Connect to Your Cluster:

Once the cluster is created and configured, go to the "Clusters" tab in the left sidebar.
Click on the "Connect" button for your cluster.
Choose the "Connect Your Application" option.
Copy the connection string provided.

5. Update Connection String in Your Project:

Replace the placeholder <username>, <password>, and <clustername> in your MongoDB connection string with the credentials and cluster details obtained from MongoDB Atlas.

Update the connection string in your database.py file with the new MongoDB

Atlas connection string.

Created a database scraperdb

Created 3 collections namely :

1. advanced_data
2. ajax_data
3. forms_data

for 3 websites and storing the scraped data in these collectons.

Output Screenshots :

```
Inserted: Philadelphia Flyers - 2011
Inserted: Phoenix Coyotes - 2011
Inserted: Pittsburgh Penguins - 2011
Inserted: San Jose Sharks - 2011
Inserted: St. Louis Blues - 2011
Inserted: Tampa Bay Lightning - 2011
Inserted: Toronto Maple Leafs - 2011
Inserted: Vancouver Canucks - 2011
Inserted: Washington Capitals - 2011
Inserted: Winnipeg Jets - 2011
Scraping completed in 86.69874405860901 seconds.
(base) PS D:\1 VKDesk>
```

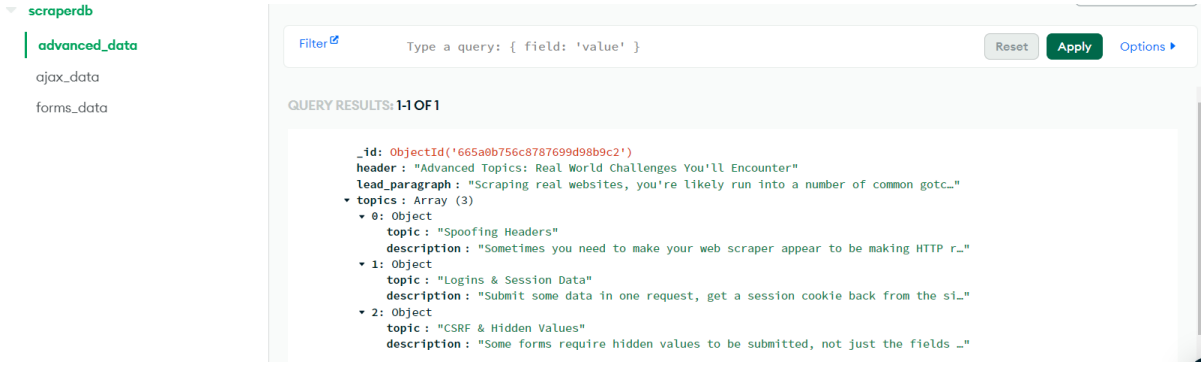
```
(base) PS D:\1 VKDesk> & "C:/Program Files/Python311/python.exe" "d:/1 VKDesk/main.py"
Advanced data already exists in the database.
Forms data already exists in the database.
Scraping completed in 2.85524845123291 seconds.
(base) PS D:\1 VKDesk> & "C:/Program Files/Python311/python.exe" "d:/1 VKDesk/main.py"
Advanced data already exists in the database.
Already exists: Boston Bruins - 1990
Already exists: Buffalo Sabres - 1990
Already exists: Calgary Flames - 1990
Already exists: Chicago Blackhawks - 1990
Already exists: Detroit Red Wings - 1990
Already exists: Edmonton Oilers - 1990
Already exists: Hartford Whalers - 1990
Already exists: Los Angeles Kings - 1990
Already exists: Minnesota North Stars - 1990
Already exists: Montreal Canadiens - 1990
Already exists: New Jersey Devils - 1990
Already exists: New York Islanders - 1990
```

```
Already exists: New York Islanders - 2011
Already exists: New York Rangers - 2011
Already exists: Ottawa Senators - 2011
Already exists: Philadelphia Flyers - 2011
Already exists: Phoenix Coyotes - 2011
Already exists: Pittsburgh Penguins - 2011
Already exists: San Jose Sharks - 2011
Already exists: St. Louis Blues - 2011
Already exists: Tampa Bay Lightning - 2011
Already exists: Toronto Maple Leafs - 2011
Already exists: Vancouver Canucks - 2011
Already exists: Washington Capitals - 2011
Already exists: Winnipeg Jets - 2011
Scraping completed in 26.82088804244995 seconds.
```

```
Advanced data already exists in the database.


Year: 2010

Title: The King's Speech, Nominations: 12, Awards: 4, Best Picture: Yes
Title: Inception, Nominations: 8, Awards: 4, Best Picture: No
Title: The Social Network, Nominations: 8, Awards: 3, Best Picture: No
Title: The Fighter, Nominations: 7, Awards: 2, Best Picture: No
Title: Toy Story 3, Nominations: 5, Awards: 2, Best Picture: No
Title: Alice in Wonderland, Nominations: 3, Awards: 2, Best Picture: No
Title: Black Swan, Nominations: 5, Awards: 1, Best Picture: No
Title: In a Better World, Nominations: 1, Awards: 1, Best Picture: No
Title: The Lost Thing, Nominations: 1, Awards: 1, Best Picture: No
Title: God of Love, Nominations: 1, Awards: 1, Best Picture: No
Title: The Wolfman, Nominations: 1, Awards: 1, Best Picture: No
Title: Strangers No More, Nominations: 1, Awards: 1, Best Picture: No
Title: Inside Job, Nominations: 1, Awards: 1, Best Picture: No
Already exists: The King's Speech - 2010
Already exists: Inception - 2010
Already exists: The Social Network - 2010
Already exists: The Fighter - 2010
Already exists: Toy Story 3 - 2010
Already exists: Alice in Wonderland - 2010
Already exists: Black Swan - 2010
Already exists: In a Better World - 2010
Already exists: The Lost Thing - 2010
Already exists: God of Love - 2010
Already exists: The Wolfman - 2010
Already exists: Strangers No More - 2010
Already exists: Inside Job - 2010

Year: 2011
```

```
Already exists: Florida Panthers - 2000
Already exists: Los Angeles Kings - 2000
Already exists: Minnesota Wild - 2000
Already exists: Montreal Canadiens - 2000
Already exists: Nashville Predators - 2000
Already exists: New Jersey Devils - 2000
Already exists: New York Islanders - 2000
Already exists: New York Rangers - 2000
Already exists: Ottawa Senators - 2000
Already exists: Philadelphia Flyers - 2000
Already exists: Phoenix Coyotes - 2000
Already exists: Pittsburgh Penguins - 2000
Already exists: San Jose Sharks - 2000
Already exists: St. Louis Blues - 2000
Already exists: Tampa Bay Lightning - 2000
Already exists: Toronto Maple Leafs - 2000
Already exists: Vancouver Canucks - 2000
Already exists: Washington Capitals - 2000
Already exists: Mighty Ducks of Anaheim - 2001
Already exists: Atlanta Thrashers - 2001
Already exists: Boston Bruins - 2001
Already exists: Buffalo Sabres - 2001
Already exists: Calgary Flames - 2001
Already exists: Carolina Hurricanes - 2001
Already exists: Columbus Blue Jackets - 2001
Already exists: Chicago Blackhawks - 2001
Already exists: Colorado Avalanche
```

MongoDB Atlas Outputs:



Conclusion :

The web scraping application developed for this project successfully achieved the objectives outlined in the assignment. Below are the key results and discussions regarding the implementation and performance of the scraping application:

1. Functionality:

The scraping application effectively extracts data from the provided URLs:
https://www.scrapethissite.com/pages/ajax-javascript/#2015
https://www.scrapethissite.com/pages/forms/
https://www.scrapethissite.com/pages/advanced/
Data from dynamic content rendered through JavaScript is successfully captured using the "requests" library, ensuring comprehensive data extraction.

2. Code Quality:

The codebase is well-structured, modular, and adheres to Python best practices. Clear separation of concerns is maintained with separate scripts for scraping individual pages (forms.py, advanced.py, etc.).
Logging is implemented to capture errors and provide insights into the scraping process for debugging purposes.

3. Error Handling:
Robust error handling mechanisms are in place to manage timeouts, network errors, and unexpected changes in website structure.
Errors encountered during scraping are appropriately logged to the scraper.log file, enabling easy identification and troubleshooting of issues.

4. Data Persistence:
Scraped data is efficiently stored in the MongoDB database, ensuring reliability and scalability.
Database schema design facilitates easy retrieval and manipulation of data for further analysis or application usage.
Duplicate data prevention is implemented to avoid redundant entries in the database.
5. Performance Optimization:

The application utilizes multi-threading to improve scraping performance by

concurrently executing scraping scripts.
Additional performance optimization techniques, such as parallel scraping or caching mechanisms, can be explored further to enhance efficiency and minimize server load.

6. Documentation:
Comprehensive documentation is provided, detailing setup, installation, and usage instructions for the scraping application.