Project Title: Multi-Threaded Web Crawler

By: Manish Kolla and Nishchay Patel

Description:
This project implements a multi-threaded web crawler written in Java, designed to efficiently explore and extract data from websites. The crawler uses **Jsoup** for parsing HTML content and **ExecutorService** for managing concurrent threads. It recursively crawls a website starting from a specified URL, extracting and following links up to a defined maximum depth. The crawler ensures that only valid and unvisited links are processed, preventing redundant requests and efficiently managing resources. Furthermore, it provides functionality for persisting the state of the crawl, allowing the process to resume from where it left off.

Key Features:

- **Multi-threading**: By using **ExecutorService**, the crawler processes multiple URLs concurrently, making it scalable for larger websites and reducing the time spent on crawling.
- **HTML Parsing**: The crawler employs **Jsoup** to extract and parse links from web pages, enabling easy navigation and extraction of relevant URLs.
- **Depth Control**: The crawler respects a configurable maximum crawl depth to prevent over-crawling and avoid wasting resources on unnecessary links.
- **State Persistence**: The list of visited URLs is saved to a file (`crawler_state.txt`), ensuring that the crawler can resume from the last visited URL if interrupted, improving fault tolerance.
- **Queue Management**: The crawler uses a **BlockingQueue** to manage URLs efficiently, ensuring thread-safe access to URLs by multiple concurrent threads.

Input:

1. **Starting URL**: The URL where the crawler begins its exploration (e.g., `https://www.linkedin.com/feed/`).
2. **Number of Threads**: The number of concurrent threads the crawler should use for processing (1-10).
3. **Maximum Crawl Depth**: The maximum depth the crawler will follow links recursively. This limits the scope of crawling to avoid excessive resource consumption.

Output:

1. **Crawler Output**: On the console, the crawler prints the title of each crawled page along with any extracted links.
2. **State File**: A file named `crawler_state.txt` stores all the visited URLs. This file allows the crawler to resume from where it left off in future runs.

Sample Outputs

Output may look something like this along with the titles and all the urls.

```
Enter the starting URL: https://www.linkedin.com/feed/
Enter the number of threads (1-10): 2
Enter the maximum crawl depth: 1
Starting crawler with 2 threads
pool-1-thread-1 - Crawling: https://www.linkedin.com/feed/
Title: LinkedIn Login, Sign in | LinkedIn
pool-1-thread-1 - Crawling:
https://www.linkedin.com/checkpoint/rp/request-password-reset?session_redir
ect=https%3A%2F%2Fwww%2Elinkedin%2Ecom%2Ffeed%2F
Queue size: 2844, Visited: 127
```

Sample output 2:

```
Enter the starting URL: https://www.gsu.edu/program-cards/
Enter the number of threads (1-10): 2
Enter the maximum crawl depth: 1
Loaded 127 URLs from previous session
Starting crawler with 2 threads
pool-1-thread-1 - Crawling: https://www.gsu.edu/program-cards/
Title: Degrees & Majors - Georgia State University
pool-1-thread-1 - Crawling: https://www.gsu.edu/a-z-index/
Title: A to Z Index - Georgia State University
pool-1-thread-1 - Crawling: https://www.gsu.edu/asset-works
Title: Login
pool-1-thread-1 - URL already visited: https://www.gsu.edu/a-z-index/
pool-1-thread-1 - Crawling: https://www.gsu.edu/collegetocareer
Title: Career Explorer - College To Career
pool-1-thread-1 - Crawling: https://www.gsu.edu
Title: Explore Georgia State University
pool-1-thread-1 - Crawling: https://www.gsu.edu/about/
Title: About - Georgia State University
Error crawling https://www.gsu.edu/?page_id=6506153: Read timed out
pool-1-thread-1 - Crawling: https://www.gsu.edu/contact-georgia-state/
Title: Contact Georgia State - Georgia State University
pool-1-thread-1 - URL already visited: https://www.gsu.edu/a-z-index/
pool-1-thread-2 - Crawling: https://www.gsu.edu/program-cards/#
Title: Degrees & Majors - Georgia State University
Error crawling https://www.gsu.edu/mission-statement/: Read timed out
```

```
Queue size: 0, Visited: 137
```

How to Run the Code:

Clone the repository from Git

Change the directory to the project directory

Compile the Java file:

```
javac -cp .;"jsoup-1.18.1.jar" "MultiWebCrawler.java"
```

Run the program with the necessary dependencies:

```
java -cp .;"jsoup-1.18.1.jar" "MultiWebCrawler.java"
```

Enter the starting URL, number of threads, and maximum depth when prompted.

The crawler will start crawling the website, printing the page titles and URLs to the console, and saving the state to `crawler_state.txt`.