Simplified Multithreaded Web Crawler with Word Counting

Project Overview

This project focuses on implementing a **basic multithreaded web crawler** that can fetch multiple web pages simultaneously, save their HTML content to files, and count occurrences of **important words** in the downloaded web pages.

Problem Statement

Traditional single-threaded web crawlers are slow and inefficient for fetching multiple web pages. The objective of this project is to use **multithreading** in **C** to improve efficiency and add basic **word frequency analysis** to count specific important words in the downloaded content.

Functionality Description

The web crawler should include the following functionalities:

- **Multithreading:** Fetch multiple web pages concurrently.
- **HTML Storage:** Save fetched HTML content to separate text files.
- **Word Counting:** Count occurrences of pre-defined important words.
- **Error Handling:** Handle errors such as invalid URLs and failed connections.
- **Input Handling:** Read URLs from an input file (urls.txt).
- **Output Files:** Store HTML pages and generate a word count report.

Features Documentation

1. Thread Management

- Use **pthreads** to handle multiple web page fetches in parallel.
- Assign each thread a separate URL to process.

2. HTML Fetching & Storage

- Fetch web pages using **libcurl**.
- Save the HTML content to files (page1.html, page2.html, etc.).

3. Word Counting

- Define a set of **important words** (e.g., data, science, algorithm).
- Count occurrences of these words in each downloaded HTML file.

4. Error Handling

- Handle **invalid URLs**, **network failures**, and **file access errors**.

5. Logging

- Print logs for fetched pages and word counts.

Implementation Steps

1. Input Parsing

- Read URLs from a file (urls.txt).

2. Threading

- Use **pthreads** to assign each URL to a thread for fetching.

3. HTTP Fetching

- Use **libcurl** to download HTML content.

4. Word Counting

- Read saved HTML files and count occurrences of important words.

Project Deliverables

1. Source Code

- A single **.c file** with threading, **libcurl**, and word counting.

2. Input File

- urls.txt: A text file with a list of URLs to crawl.

3. Output Files

- HTML files: page1.html, page2.html, etc.
- Word count results printed to the console.

Compilation Requirements

Ensure your project is compilable on **Linux** using the **gcc** compiler:

gcc -std=c11 -pedantic -pthread -lcurl crawler.c -o crawler

Note:

- -pthread is required for multithreading.
- -lcurl is required for **libcurl** networking.

Makefile Targets

Your **Makefile** must support:

- all: Compiles the web crawler binary.
- clean: Removes compiled files.
- run: Runs the web crawler with URLs from urls.txt.

Example Makefile:

```
all:
	gcc -std=c11 -pedantic -pthread -lcurl crawler.c -o crawler
clean:
	rm -f crawler page*.html
run:
	./crawler
```