**PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE**

**ACADEMIC YEAR: 2023-24**

## **DEPARTMENT of COMPUTER ENGINEERING DEPARTMENT**

**CLASS: B.E. SEMESTER: I**

**SUBJECT: LP-IV**

|  |  |
| --- | --- |
| **ASSIGNMENT NO.** | A5 |
| **TITLE** | **Web Crawling** |
| **PROBLEM STATEMENT /DEFINITION** | Write a simple program to implement a simple web crawler. |
| **OBJECTIVE** | * Learn and understand the concept of web crawling and web scraping. * Build a simple web crawler bot. |
| **OUTCOME** | Students will be able to:   * Differentiate between web crawling and web scraping. * Implement a simple web crawler / spiderbot. |
| **S/W PACKAGES AND**  **HARDWARE APPARATUS USED** | 64- bit Ubuntu OS / Windows 10 OS,  8 GB RAM,  Python 3.9.0,  pip or condo to install scrapy library provided by Python |
| **REFERENCES** | 1. C.J. Rijsbergen, &quot;Information Retrieval&quot;, (http://www.dcs.gla.ac.uk/Keith/Preface.html)  2. W.R. Hersh, ―Information Retrieval: A Health and Biomedical Perspective‖, Springer, 2002.  3. G. Kowalski, M.T. Maybury. &quot;Information storage and Retrieval System&quot; , Springer, 2005 |
| **STEPS** | Refer to theory, algorithm, test input, test output |
| **INSTRUCTIONS FOR**  **WRITING JOURNAL** | 1. Date  2. Assignment no.  3. Problem definition  4. Learning objective  5. Learning Outcome  6. Concepts related Theory  7. Algorithm  8. Test cases  10. Conclusion/Analysis |

**Prerequisites:**

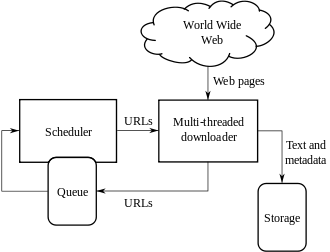
**Concepts related Theory:**

**Web Crawler**

A Web crawler, sometimes called a spider or spiderbot and often shortened to crawler, is an Internet bot that systematically browses the World Wide Web and that is typically operated by search engines for the purpose of Web indexing (web spidering).

Web search engines and some other websites use Web crawling or spidering software to update their web content or indices of other sites' web content. Web crawlers copy pages for processing by a search engine, which indexes the downloaded pages so that users can search more efficiently.

Crawlers consume resources on visited systems and often visit sites unprompted. Issues of schedule, load, and "politeness" come into play when large collections of pages are accessed. Mechanisms exist for public sites not wishing to be crawled to make this known to the crawling agent. For example, including a robots.txt file can request bots to index only parts of a website, or nothing at all.



*Architecture of a Web crawler*

**How does a Web Crawler works?**

A Web crawler starts with a list of URLs to visit. Those first URLs are called the seeds. As the crawler visits these URLs, by communicating with web servers that respond to those URLs, it identifies all the hyperlinks in the retrieved web pages and adds them to the list of URLs to visit, called the crawl frontier. URLs from the frontier are recursively visited according to a set of policies. If the crawler is performing archiving of websites (or web archiving), it copies and saves the information as it goes. The archives are usually stored in such a way they can be viewed, read and navigated as if they were on the live web, but are preserved as 'snapshots'.

The archive is known as the repository and is designed to store and manage the collection of web pages. The repository only stores HTML pages and these pages are stored as distinct files. A repository is similar to any other system that stores data, like a modern-day database. The only difference is that a repository does not need all the functionality offered by a database system. The repository stores the most recent version of the web page retrieved by the crawler.

**Difference between Web Scraping and Web Crawling**:

Web scraping is the process of using bots to extract content and data from a website.

Unlike screen scraping, which only copies pixels displayed on screen, web scraping extracts underlying HTML code and, with it, data stored in a database. The scraper can then replicate entire website content elsewhere.

Web scraping is used in a variety of digital businesses that rely on data harvesting. Legitimate use cases include:

* Search engine bots crawling a site, analyzing its content and then ranking it.
* Price comparison sites deploying bots to auto-fetch prices and product descriptions for allied seller websites.
* Market research companies using scrapers to pull data from forums and social media (e.g., for sentiment analysis.)

**Algorithm:**

* Step 1: Send an HTTP request to the URL of the webpage. It responds to your request by returning the content of web pages.
* Step 2: Parse the webpage. A parser will create a tree structure of the HTML as the webpages are intertwined and nested together. A tree structure will help the bot follow the paths that we created and navigate through to get the information.
* Step 3: Using the Python library to search the parse tree.

**Conclusion:**

1. This assignment helped us to understand the concept of web crawling.
2. Successfully implemented a simple web crawler with scrapy utility library provided by python.

**Review Questions**:

#### Q1. How can data scraped from the internet be used for business intelligence, analytics, or research purposes?

#### Q2. What are the pros and cons of web scraping?

#### Q3. Why Selenium is often preferred over other tools for web scraping?

#### Q4. What are the different types of web scraping techniques that can be used?

#### Q5. What are the main challenges faced while writing a web crawler?