BRIEF REPORT ON WEBSITE INFORMATION EXTRACTOR

Prepared By:

Burra Ajay Kumar

Approach

Objective:

The goal was to create a script that gathers important details from a website and saves them into a database. This includes information like titles, descriptions, social media links, technology used, payment options, language, and category of the site.

Steps Taken:

Import Required Libraries:

- Imported necessary libraries including requests, BeautifulSoup, Selenium, MySQL.connector, re, and langcodes.

Initialize Web Driver:

- Used Selenium with ChromeDriver to load web pages. Configured it to run in headless mode for efficiency.

URL Handling:

- Ensured that the provided URL is valid and prepended 'https://' if not already present.

Data Extraction:

- Meta Information: Extracted meta title and description.
- Social Media Links: Identified and collected links to major social media platforms.
- Tech Stack: Detected MVC frameworks, CMS platforms, JavaScript libraries, CSS frameworks, and backend technologies used on the website.
- Payment Gateways: Identified payment gateways mentioned on the website using regex patterns.
- Website Language: Extracted the language code from the HTML tag and converted it to the full language name.
- Website Category: Implemented a keyword-based scoring system to categorize the website based on its content.

Save to Database:

- Connected to a MySQL database and saved the extracted data into a predefined table.

Error Handling:

- Included exception handling to manage potential issues during web scraping and database operations.

Challenges Encountered:

1. Dynamic Content:

- Some websites load content after the initial page load using JavaScript, making it harder to extract data.
- **Solution:** Used Selenium to fully render pages before analyzing them.

2. Different Website Structures:

- Websites can have different HTML structures, which makes it challenging to consistently extract data.
- **Solution:** Developed flexible parsing logic to handle diverse HTML layouts.

3. Database Integration:

- Ensured data was safely inserted into MySQL, handling security concerns and preventing data loss.
- **Solution:** Used secure methods like parameterized queries to interact with the database.

4. Accurate Technology Detection:

- Identifying the correct technologies used by websites requires careful analysis.
- **Solution:** Used keyword-based methods and regex to accurately identify technologies from scripts and links.

5. Categorizing Websites:

- Assigning websites to the correct category based on their content was a complex task.
- Solution: Implemented a scoring system that analyzed keywords across different sections of each website.

Conclusion:

The script effectively extracts essential details from websites and stores them in a database, despite challenges like dynamic content and varied HTML structures, Implemented solutions to ensure reliable data extraction. Future improvements could focus on handling CAPTCHAs, refining technology detection accuracy, and expanding the categorization system.