**Web Scrapping**

**Assignment Questions**

**Q1. What is Web Scraping? Why is it Used? Give three areas where Web Scraping is used to get data.**

Web scraping is the automated process of extracting data from websites. It involves using software tools or scripts to fetch and parse the HTML code of web pages, extracting the desired information and structuring it in a usable format, such as a spreadsheet or a database.

Web scraping is used for various purposes, including:

1. Data Collection: Web scraping allows organizations and individuals to gather large amounts of data from websites efficiently. This data can be used for research, analysis, market intelligence, competitive analysis, price comparison, and more. By automating the data collection process, web scraping saves time and effort compared to manual data extraction.

2. Content Aggregation: Web scraping is often used to aggregate and consolidate data from multiple sources into a single location. For example, news aggregators gather headlines and articles from various news websites, displaying them on their platforms. Similarly, job aggregators collect job listings from different job boards to provide a centralized database of available opportunities.

3. Business Intelligence: Web scraping enables businesses to monitor and track information related to their industry, competitors, customers, and market trends. By scraping data from websites, companies can analyze customer reviews, social media sentiment, pricing information, product details, and other relevant data to make informed business decisions.

**Q2. What are the different methods used for Web Scraping?**

There are several methods and techniques used for web scraping, depending on the complexity of the website and the desired data. Here are some common methods:

1. Manual Copy-Pasting: The most basic method involves manually copying and pasting the required data from a web page into a spreadsheet or text editor. This approach is suitable for scraping small amounts of data or when the data is easily identifiable and doesn't require automation.

2. Regular Expressions (Regex): Regular expressions are powerful patterns used to match and extract specific data from HTML or text. It involves writing custom patterns that match the desired data and using them to extract the relevant information. Regex can be useful for simple data extraction tasks, but it becomes more challenging for complex web pages with changing structures.

3. HTML Parsing: HTML parsing involves using libraries or tools to parse the HTML structure of a web page and extract specific elements or data. Popular libraries like BeautifulSoup (Python) and Jsoup (Java) provide convenient methods to navigate and extract data from HTML documents based on CSS selectors or XPath expressions.

**Q3. What is Beautiful Soup? Why is it used?**

Beautiful Soup is widely used for the following reasons:

1. HTML Parsing: Beautiful Soup handles the intricacies of parsing HTML or XML documents, making it easier to navigate and extract data. It can handle malformed or poorly structured HTML and provides methods to search for elements based on various criteria like tag names, attributes, class names, etc.

2. Easy to Use: Beautiful Soup has a user-friendly and intuitive API that simplifies the process of extracting data from web pages. It abstracts away much of the complexity involved in parsing and navigating the HTML structure, allowing developers to focus on writing concise and readable code.

3. Navigation and Search: Beautiful Soup provides powerful methods for traversing and searching the HTML tree structure. It supports CSS selectors and allows developers to locate elements based on their tag names, attributes, class names, or even text content. This makes it flexible and adaptable to different HTML structures.

Overall, Beautiful Soup simplifies the process of web scraping and data extraction from HTML or XML documents. It provides a straightforward and Pythonic way to parse and navigate web page structures, making it a popular choice among developers for their scraping projects.

**Q4. Why is flask used in this Web Scraping project?**

Flask is a web framework for Python that is commonly used in web scraping projects for several reasons:

1. Building Web Applications: Flask allows you to build web applications or APIs around your web scraping project. It provides a lightweight and flexible framework to handle HTTP requests, define routes, and serve responses. By using Flask, you can create a user interface to interact with your web scraping functionality, display the scraped data, and provide a seamless experience for users.

2. Request Handling: Web scraping often involves making HTTP requests to retrieve web pages or interact with APIs. Flask provides tools to handle incoming requests and process the responses. You can use Flask's routing capabilities to define the endpoints that will trigger your web scraping logic, and Flask's request object to access request data like form inputs, query parameters, or headers.

3. Data Storage and Presentation: Flask integrates well with various data storage solutions, including relational databases like MySQL or PostgreSQL, NoSQL databases like MongoDB, or file-based systems. You can store the scraped data in a database or file system using Flask's database extensions or file handling capabilities. Additionally, Flask enables you to present the scraped data in a user-friendly manner by rendering HTML templates, generating dynamic web pages, or providing JSON/XML responses for API endpoints.

Overall, Flask provides a convenient and flexible framework for developing web scraping projects. It facilitates handling HTTP requests, managing data storage and presentation, implementing authentication, and integrating with other libraries. By using Flask, you can build a robust and user-friendly web scraping application or API.

**Q5. Write the names of AWS services used in this project. Also, explain the use of each service.**

In a web scraping project deployed on AWS, several services can be utilized depending on the specific requirements and architecture. Here are some AWS services that can be used and their respective uses in the project:

1. Amazon EC2 (Elastic Compute Cloud): EC2 provides virtual servers in the cloud. In a web scraping project, EC2 instances can be used to run the web scraping code or host the Flask application. It offers scalable compute capacity and allows you to configure the instance specifications based on your project's needs.

2. AWS Lambda: Lambda is a serverless compute service that allows running code without provisioning or managing servers. It can be used in a web scraping project to execute specific tasks, such as triggering the scraping process at scheduled intervals or handling asynchronous processing of scraping jobs.

3. Amazon S3 (Simple Storage Service): S3 is an object storage service that provides scalable storage for data, files, and other assets. In a web scraping project, S3 can be used to store the scraped data, such as HTML files, CSV files, or images. It offers durability, availability, and easy integration with other AWS services.

4. Amazon RDS (Relational Database Service): RDS is a managed database service that simplifies the setup, operation, and scaling of relational databases. In a web scraping project, RDS can be used to store structured data extracted from web pages. You can choose database engines like MySQL, PostgreSQL, or others based on your requirements.

It's important to note that the specific services used in a web scraping project on AWS may vary depending on the project requirements, data processing needs, and overall architecture design.