assignment=17

July 27, 2023

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

Web scraping is the process of extracting data from websites by using automated programs or scripts. It involves retrieving the HTML code of a website and then parsing it to extract relevant information, such as text, images, or links.

Web scraping is used for various purposes, such as data mining, lead generation, market research, and content aggregation. By automating the process of collecting data from the web, it enables organizations and individuals to obtain large amounts of information quickly and easily, without having to manually copy and paste it from websites.

Here are three areas where web scraping is commonly used to obtain data:

E-commerce: Web scraping is widely used in the e-commerce industry to collect data about products, prices, and reviews from different websites. This helps retailers to keep track of their competitors' prices, monitor customer reviews, and analyze market trends to make better business decisions.

Social media: Web scraping is also used to collect data from social media platforms, such as Twitter and Facebook, to monitor public sentiment about a particular topic, brand, or event. This data can be used for sentiment analysis, market research, and to inform social media marketing campaigns.

Research: Web scraping is used in academic and scientific research to collect data from various sources on the web. This data can be used to conduct surveys, analyze trends, and support research projects in various fields, such as social sciences, economics, and public health.

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

There are various methods and tools used for web scraping, depending on the complexity of the task and the type of data being collected. Here are some of the most common methods:

Manual web scraping: This involves manually copying and pasting data from web pages into a spreadsheet or other software. While this method is simple, it is time-consuming and inefficient for large amounts of data.

Automated web scraping: This involves using software or tools to automate the process of extracting data from web pages. Some popular tools include Scrapy, BeautifulSoup, and Selenium. Automated web scraping can be more efficient than manual scraping, but it requires some programming knowledge.

API web scraping: Some websites offer APIs (application programming interfaces) that allow developers to access and retrieve data from their databases. This method is often more efficient and reliable than web scraping, but it requires an API key and some programming knowledge.

DOM parsing: This method involves parsing the Document Object Model (DOM) of a web page to extract specific elements or data. It can be used with tools like jQuery and XPath to locate and extract data from specific HTML tags and attributes.

Headless browsing: This method involves using a browser to simulate human behavior on a website, such as clicking buttons or filling out forms, to collect data. This can be done using tools like Puppeteer and PhantomJS. Headless browsing is useful for scraping dynamic websites that require user interaction.

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

Beautiful Soup is a popular Python library that is used for web scraping purposes. It is a tool for parsing HTML and XML documents, and it is particularly useful for extracting data from web pages.

Beautiful Soup provides a simple and intuitive way to navigate and search through HTML documents, and it can be used to extract specific data from web pages based on its HTML tags, attributes, and text content. It can also handle poorly formatted HTML and fix any errors or inconsistencies in the document structure.

Some of the key features of Beautiful Soup include:

Easy parsing: Beautiful Soup provides a simple and intuitive interface for parsing HTML and XML documents. It can handle nested tags, malformed HTML, and other common issues that can arise when scraping web pages.

Powerful search capabilities: Beautiful Soup allows you to search for specific tags, attributes, and text content within a document, making it easy to extract the data you need.

Integration with other libraries: Beautiful Soup can be easily integrated with other Python libraries, such as Requests for making HTTP requests and Pandas for data analysis.

Support for different parsers: Beautiful Soup supports different parsers, including the built-in Python parser, lxml, and html5lib. This makes it flexible and adaptable to different scraping tasks.

Overall, Beautiful Soup is a powerful and flexible library for web scraping, and it can help you to efficiently extract data from web pages for various purposes, such as research, data analysis, and content aggregation.

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

Flask is a popular Python web framework that is often used for building web applications and APIs. It is lightweight, flexible, and easy to use, making it a good choice for web scraping projects that require a simple web interface for interacting with the scraped data.

In a web scraping project, Flask can be used to build a simple web application that allows users to enter search parameters or URLs, and then display the scraped data in a user-friendly format.

Flask can also handle routing, templating, and other web-related tasks, making it easy to build a functional and responsive web interface for your scraping project.

Here are some specific reasons why Flask might be used in a web scraping project:

Easy to use: Flask is a simple and intuitive framework, making it easy for developers to get started and build web applications quickly.

Flexible: Flask is a flexible framework that can be customized and extended to meet the specific needs of your web scraping project.

Lightweight: Flask is a lightweight framework that is easy to deploy and can run on small servers or cloud-based services.

Integrates well with other Python libraries: Flask can be easily integrated with other Python libraries, such as Beautiful Soup or Scrapy, making it a good choice for web scraping projects that require data extraction from multiple sources.

Overall, Flask can be a useful tool in a web scraping project because it provides a simple and flexible way to build a web interface for interacting with the scraped data. It is a lightweight and easy-to-use framework that can be customized and extended to meet the specific needs of your scraping project.

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

CodePipeline: AWS CodePipeline is a fully managed continuous delivery service that helps us automate our software release process. With CodePipeline, we can build, test, and deploy our code every time there is a change, based on the release model of our choice.

CodePipeline allows us to create a pipeline that consists of a series of stages, each of which represents a step in our software release process. Each stage can have one or more actions, such as building our code, running tests, and deploying our code to a production environment.

We can integrate CodePipeline with other AWS services, such as AWS CodeBuild and AWS CodeDeploy, to automate our entire software release process. We can also use CodePipeline with third-party tools and services, such as Jenkins and GitHub, to customize our pipeline and incorporate our existing workflows.

By using CodePipeline, we can increase the speed and reliability of our software release process, reduce manual errors, and improve collaboration between development and operations teams.

Elastic Beanstalk: AWS Elastic Beanstalk is a fully managed service that makes it easy for us to deploy, manage, and scale our web applications and services. Elastic Beanstalk supports a wide range of popular programming languages, such as Java, .NET, Node.js, Python, Ruby, PHP, and Go.

With Elastic Beanstalk, we can simply upload our code and Elastic Beanstalk automatically handles the deployment, capacity provisioning, load balancing, and automatic scaling of our application. Elastic Beanstalk also provides us with a range of tools for monitoring and managing our application, including dashboards, logs, and alerts.

We can customize the environment that Elastic Beanstalk creates for our application, including the instance type, operating system, and database configuration. We can also integrate our application with other AWS services, such as Amazon RDS for databases, Amazon S3 for storage, and Amazon CloudWatch for monitoring.

Elastic Beanstalk offers us a range of deployment options, including rolling updates, blue/green deployments, and canary deployments. This enables us to choose the deployment method that best suits our application and business needs.

Overall, Elastic Beanstalk simplifies the process of deploying and managing web applications and services on AWS, allowing us to focus on writing code and building our business.

[ ]:

[ ]:

[ ]: