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

Ans - Web scraping is the automated process of extracting data from websites. It involves retrieving and parsing the HTML or other structured data of a webpage to extract the desired information.

Uses

1)Data Collection and Analysis: Web scraping allows organizations and researchers to collect large amounts of data from various sources on the internet. This data can be used for analysis, market research, monitoring competitor prices, sentiment analysis, and trend forecasting, among other purposes.

2)Content Aggregation: Web scraping enables the aggregation of content from different websites into a single platform. News aggregators, job portals, and real estate listing websites, for example, often use web scraping to gather and display information from multiple sources in a unified manner.

3)Research and Monitoring: Web scraping is valuable for academic research, particularly in fields like social sciences and data mining. It can help researchers collect data for surveys, analyse online discussions, track sentiment on social media platforms, and monitor changes in websites or public datasets.

Areas

a. E-commerce and Price Comparison: Online retailers use web scraping to monitor and analyse competitor prices, product details, and customer reviews. This data helps them make informed pricing decisions and adjust their strategies accordingly. Price comparison websites also rely on web scraping to gather and display product information and prices from various online stores.

b. Financial and Stock Market Analysis: Financial institutions and investors utilize web scraping to gather financial data, news articles, and market trends from different websites. This data is used to analyze stock performance, identify investment opportunities, and make informed trading decisions.

c. Social Media Monitoring and Sentiment Analysis: Web scraping is employed to monitor and analyze social media platforms for brand reputation management, customer sentiment analysis, and market research. By scraping data from platforms like Twitter, Facebook, or Reddit, companies can track mentions of their brand, identify emerging trends, and gauge public opinion about their products or services.

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

Ans -

1) Manual Copy-Pasting: This is the most basic form of web scraping, where users manually copy and paste data from websites into a local file or spreadsheet. While this method is simple, it is not automated and is only suitable for scraping small amounts of data.
2) Regular Expressions (Regex): Regular expressions are powerful patterns used to extract specific data from HTML or text documents. Regex can be used in combination with programming languages like Python or JavaScript to search for patterns in web pages and extract the desired information. While effective, regular expressions can be complex and brittle, and may not handle complex HTML structures well.
3) HTML Parsing: HTML parsing involves using libraries or frameworks like Beautiful Soup (Python) or Jsoup (Java) to parse and navigate through HTML documents. These libraries provide convenient methods to extract data based on HTML tags, attributes, or CSS selectors. HTML parsing is robust and flexible, as it can handle complex page structures and dynamically generated content.
4) Web Scraping Frameworks: There are several web scraping frameworks available that simplify the scraping process. These frameworks, such as Scrapy (Python) or Puppeteer (JavaScript), provide high-level abstractions and additional functionalities like handling cookies, sessions, and JavaScript rendering. They often include features for data extraction, handling pagination, and handling complex websites.
5) Headless Browsers: Headless browsers like Puppeteer, Selenium, or PhantomJS can be used for web scraping. They allow you to automate web browsing and interact with web pages programmatically. By simulating user

interactions, such as clicking buttons or filling out forms, headless browsers can access dynamically generated content or login-restricted areas to extract data.

6) API Access: Some websites provide APIs (Application Programming Interfaces) that allow developers to access and retrieve data in a structured manner. APIs provide a more structured and reliable way of obtaining data compared to web scraping. However, not all websites offer APIs, and API access may be limited or require authentication.

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

Ans – Beautiful Soup is a popular Python library used for web scraping and parsing HTML or XML documents. It provides a convenient and intuitive way to extract data from web pages by traversing the HTML structure and searching for specific elements or patterns.

1) HTML Parsing: Beautiful Soup excels at parsing and navigating through HTML documents, handling imperfect or poorly formatted HTML with ease. It provides a simple API that allows users to search, filter, and manipulate the parsed HTML tree.
2) Tag Searching: Beautiful Soup allows you to search for specific HTML tags or groups of tags using methods like .find() or .find_all(). These methods can search based on tag names, attributes, CSS classes, or other criteria, making it easy to extract specific data from web pages.
3) Powerful Navigational Methods: Beautiful Soup provides various methods for navigating the HTML tree structure, such as accessing parent, child, or sibling elements. This enables you to traverse the document and locate the desired data accurately.
4) Data Extraction: Beautiful Soup provides methods to extract the textual content, attributes, or other properties of HTML elements. This allows you to retrieve specific data from the parsed HTML, such as text within <p> tags, URLs from <a> tags, or image sources from <img> tags.
5) Handling Complex HTML Structures: Beautiful Soup can handle complex HTML structures, including nested elements, irregular indentation, or inconsistent formatting. It can adapt to different HTML variations and still extract data reliably.
6) Integration with Other Libraries: Beautiful Soup can be used in conjunction with other Python libraries and frameworks, such as requests for downloading web pages or pandas for data manipulation and analysis. This makes it a versatile tool in the web scraping workflow.

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

Ans -

1) Web Interface: Flask allows you to create a web interface or API to interact with the web scraping functionality. It provides a lightweight and easy-to-use framework for building web applications. With Flask, you can create endpoints to receive requests, display scraped data, and provide user interaction.
2) Routing and URL Handling: Flask provides routing capabilities, allowing you to define routes and handle different URLs and HTTP methods. This is useful when designing a web scraping application that needs to handle various URLs or dynamic parameters for scraping different websites or pages.
3) Templating Engine: Flask comes with a built-in templating engine (Jinja2), which makes it easier to generate HTML pages dynamically. This is helpful when you want to present the scraped data in a user-friendly and visually appealing manner. You can define templates with placeholders and use Flask to render those templates with the scraped data.
4) Request Handling: Flask simplifies handling incoming HTTP requests. It provides functionalities to handle request parameters, headers, cookies, and form data. This is essential in a web scraping project where you may need to accept user inputs or configure scraping options through HTTP requests.
5) Session Management: Flask supports session management, allowing you to store user-specific data across multiple requests. This can be useful in web scraping projects where you need to persist user preferences, maintain authentication tokens, or track the progress of scraping tasks.

6) Integration with Python Libraries: Flask seamlessly integrates with other Python libraries commonly used in web scraping projects. You can easily combine Flask with libraries like Beautiful Soup for parsing HTML, Requests for making HTTP requests, or Pandas for data manipulation and analysis.
7) Scalability and Deployment: Flask is known for its lightweight and scalable nature, making it suitable for both small-scale and large-scale web scraping projects. It can be easily deployed on various hosting platforms or cloud services, allowing your web scraping application to be accessible from anywhere.

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

Ans –

In a web scraping project hosted on AWS (Amazon Web Services), several services can be utilized. The specific services used may vary depending on the project requirements, but here are some commonly used AWS services and their purposes:

1) EC2 (Elastic Compute Cloud): EC2 provides virtual server instances in the cloud. It can be used to host the web scraping application itself, allowing you to run the scraping code and handle incoming requests. EC2 instances can be configured with the necessary software dependencies and scalability options to suit the project's needs.
2) Lambda: AWS Lambda is a serverless computing service that allows you to run code without managing or provisioning servers. It is commonly used in web scraping projects to execute the scraping code in a serverless manner. With Lambda, you can trigger scraping tasks based on events, schedule periodic scrapes, or dynamically scale the scraping function based on demand.
3) S3 (Simple Storage Service): S3 is an object storage service that provides highly scalable storage for data and files. In a web scraping project, you can use S3 to store the scraped data, logs, or other artifacts. S3 buckets can be configured to serve as a storage backend for the web scraping application or as a destination for storing the scraped data.
4) CloudWatch: AWS CloudWatch is a monitoring and observability service. It allows you to collect and monitor logs, set up alarms, and gain insights into the health and performance of your web scraping application. You can use CloudWatch to monitor the execution of scraping tasks, detect errors or performance issues, and set up automated notifications.
5) API Gateway: AWS API Gateway is a fully managed service for creating, deploying, and managing APIs. It can be used in a web scraping project to define an API endpoint for receiving scraping requests or providing access to the scraped data. API Gateway offers features such as authentication, throttling, and request transformation, allowing you to control and secure the API interactions.
6) DynamoDB: DynamoDB is a fast and scalable NoSQL database service provided by AWS. It can be used to store and manage the scraped data in a structured format. DynamoDB offers low-latency access, automatic scaling, and flexible schema options, making it suitable for storing and querying the scraped data.
7) Step Functions: AWS Step Functions is a serverless workflow orchestration service. It allows you to define and coordinate multiple steps or tasks in a web scraping workflow. Step Functions can be used to manage complex scraping processes, handle retries, parallelize tasks, or coordinate dependencies between different scraping functions or services.
8) Athena: Amazon Athena is an interactive query service that enables you to analyze data directly from S3 using SQL. It can be used in a web scraping project to run ad-hoc queries or perform data analysis on the scraped data stored in S3. Athena provides a serverless and on-demand query execution environment, eliminating the need for managing infrastructure.