Q1. What is Web Scraping? Why is it used? Three areas of application

Web Scraping:

- Web scraping is the **automated process of extracting data** from websites.
- It allows you to collect information **without manually copying** it.

Why it is used:

1. To collect **large amounts of data** from websites quickly.
2. To monitor and analyze **trends, prices, or news**.
3. To feed data into **data analysis, machine learning, or dashboards**.

Three areas where web scraping is used:

1. **E-commerce:** Scraping product prices, reviews, and stock information.
2. **Job Portals:** Collecting job listings, locations, and salary information.
3. **Finance:** Scraping stock prices, cryptocurrency trends, or news articles.

Q2. Different methods used for Web Scraping

1. **HTML Parsing**
    - Extracting information directly from HTML pages using libraries like **Beautiful Soup**.
2. **DOM Parsing**
    - Navigating the **Document Object Model (DOM)** using libraries like **lxml** or **Selenium**.

3. **API Requests**
    - Using the website’s **public API** to fetch structured data (JSON/XML).
4. **Web Automation Tools**
    - Tools like **Selenium** or **Playwright** to interact with dynamic pages (JavaScript rendered content).
5. **Regular Expressions (Regex)**
    - Pattern-based extraction of text from HTML or page source.

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

Beautiful Soup:

- A Python library used for **parsing HTML and XML documents**.
- Makes it easy to **navigate, search, and extract data** from web pages.

Why it is used:

1. Simplifies HTML parsing for beginners and professionals.
2. Allows searching by **tags, attributes, CSS selectors**.
3. Handles **messy HTML** that is not well-formed.

Example:

```
from bs4 import BeautifulSoup
import requests

url = "https://example.com"
response = requests.get(url)
soup = BeautifulSoup(response.text, 'html.parser')
```

```
# Extract title of page
print(soup.title.text)
```

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

- **Flask** is used to **create a web interface** for the web scraping project.
- Reasons:
    1. Provides a **frontend to input URLs or keywords** for scraping.
    2. Displays the **scraped results dynamically** on a web page.
    3. Makes the scraping project **accessible over the web** (can deploy on cloud).

Example use:

- A user enters a product name → Flask triggers scraping → Scraped data is shown on the webpage.

Q5. AWS Services used in this project and their purpose

AWS Service	Purpose in Web Scraping Project
AWS EC2 (Elastic Compute Cloud)	Hosts the Flask application for web access.
AWS S3 (Simple Storage Service)	Stores scraped data, files, or results for download.
AWS Lambda (optional)	Runs serverless scraping scripts at scheduled times.
AWS CloudWatch	Monitors the scraping application logs and performance.
AWS RDS / DynamoDB	Stores structured scraped data for analysis or dashboards.