WEB SCRAPING JOB LISTINGS FROM INDEED

A Data Extraction and Analysis Project

INTRODUCTION

- **What is Web Scraping?**
  Automated extraction of data from websites using tools like Python and BeautifulSoup.

- **Why Scrape Job Listings?**
  Analyze job market trends, salary data, hiring companies, and in-demand skills.

- **Indeed as a Source**
  A leading global job search engine with millions of job listings, making it an ideal platform for gathering valuable job market data.

- **Project Goal**
  Scrape job listings for specific roles (e.g., "Data Analyst") in defined locations (e.g., ”Mumbai") to gather details like job titles, company names, salaries, and job descriptions.

PROJECT GOAL OBJECTIVE

➢ Objective:

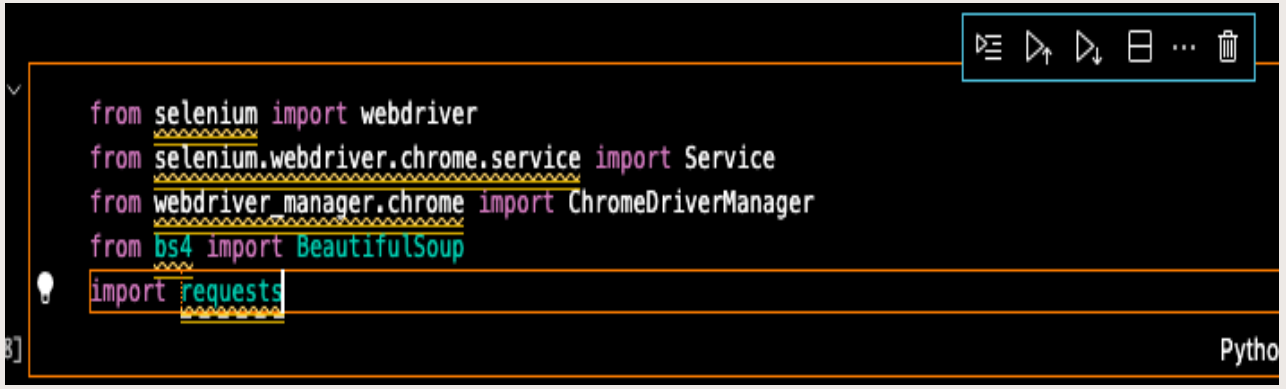
- ➢ Extract data from Indeed to understand trends in the job market.
- ➢ Focused on **Data analyst roles** in a specific location (eg. Mumbai).

➢ Key Data Points to Collect:

- ➢ **Job Titles**: Software Engineer, Data Scientist, etc.
- ➢ **Company Names**: Companies hiring in the field.
- ➢ **Salary Information**: If available, ranges and estimates.
- ➢ **Location**: City and state or remote.
- ➢ **Skills**: Key skills mentioned in the job descriptions.

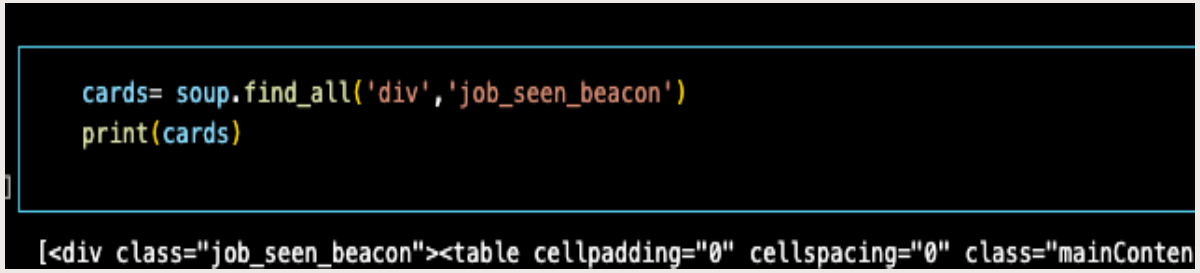
TOOLS AND TECHNOLOGIES USED

- **BeautifulSoup**: Parsing HTML content to extract relevant job data (titles, companies, locations).
    i. Provides methods like find() and find_all() to locate specific HTML tags or content.
- **Selenium**: Handling dynamic content on the Indeed website that loads through JavaScript Key Features:
    i. Automates web browser interactions (scrolling, clicking).
- **Pandas**: Used for data manipulation, cleaning, and storing scraped job data in a structured format
    i. Powerful DataFrame structure for handling large datasets.
    ii. Functions for cleaning and transforming data (e.g., handling missing values).
- **CSV**: Stores cleaned job data in a CSV file for easy export and further analysis.
    i. Lightweight format for data storage.



SCRAPING WORKFLOW

- **Sending Requests**
  - i. Used **Requests** to send HTTP requests and retrieve raw HTML pages from Indeed.
  - **ii.** **Pagination Handling**: Handled multiple pages of job listings by iterating through page numbers.
- **Parsing HTML with BeautifulSoup**
  - i. Used **BeautifulSoup** to parse the raw HTML and extract specific job-related data (e.g., titles, companies, salaries).
  - **ii.** **Data Extraction**: Targeted elements like , , and <a> to get job title, company name, and location.
- **Handling Dynamic Content with Selenium**
  - i. Used **Selenium** to handle dynamic job listings that load through JavaScript (infinite scrolling).
  - ii. Automated scrolling to load more job listings or clicked "Next" buttons



DATA CLEANING & PROCESSING

- **Data Cleaning**:
- Removed irrelevant fields (e.g., unnecessary HTML tags).
- **Handled Missing Data**: Replaced or removed incomplete job entries (e.g., missing salary or location).
- **Standardization**: Standardized job titles, locations, and salary formats.
- **Extracting Job Data**:Loop through each job listing to extract:
  - Job title (<a> tag with the title attribute).
  - Company name (tag with class 'company').
  - Location (tag with class 'location').
- **Storing Data**:Save the data into a structured DataFrame using pandas.DataFrame().
- Write the DataFrame to a CSV file using df.to_csv().

```
def get_record(card):
    atag= card.h2.a.span
    job_title= atag.get('title')
    a_tag_url= card.h2.a
    job_url = 'http://indeed.com' + a_tag_url.get('href')
    company_name =card.find('span','css-1h7lukg eu4oa1w0').text
    job_location= card.find('div','css-1restlb eu4oa1w0').text
    salary= card.find('div', 'css-18z4q2i eu4oa1w0')

    if salary:
        salary = salary.text.strip()

        # If salary contains "Full time", replace it with "Not Provide
        if "Full-time" in salary:
            salary = "Not Provided"
    else:
        salary = "Not Provided"


    record= (job_title,job_url,company_name,job_location,salary)

    return record
```

```
import pandas as pd
df = pd.DataFrame(records)
df
```

```
df.to_csv('Indeed Job Oppurtunities.csv')
```

CHALLENGES & SOLUTIONS

- **Understanding Website Structure**:
  - Identifying the correct HTML tags and elements for extracting job-related data (such as titles, locations, and salaries) was challenging due to the website's complex and sometimes inconsistent HTML structure.
  - **Solution**: Used advanced techniques like **CSS selectors** queries to pinpoint the relevant tags, ensuring accurate data extraction despite structural variations.
- **Handling Missing Data**:
  - Many job listings had missing or incomplete information, such as missing salary details,, which impacted the completeness of the dataset.
  - **Solution**: Implemented **placeholders** for missing data (e.g., "Not Provided" for missing salary) ensuring consistency and avoiding gaps in the dataset.
- **Limited Data Collection for Demonstration**:
  - To ensure the scraping process was manageable and to avoid overloading the website, the data collection was **restricted** to a small sample of job listings for demonstration purposes.

```
    # If salary contains "Full time", replace it with "Not Provided"
    if "Full-time" in salary:
        salary = "Not Provided"
else:
    salary = "Not Provided"
```

```
records[14]
```

```
('Data Analyst',
 'http://indeed.com/rc/clk?jk=9d2cfc7268a25b8c&bb=gbDQkAy6Ta
 'Sounce Retail Private Limited',
 'Grant Road, Mumbai, Maharashtra',
 '₹20,000 - ₹25,000 a month')
```

THANK YOU