**INDEED SCRAPER**

PROJECT-1

**PROJECT DESCRIPTION:**

The **Indeed Scraper** is a Python-based web scraper designed to extract job listings from the Indeed website. It gathers job details such as job title, company name, location, salary (if available), job description, and the date posted. This tool is useful for job seekers, researchers, and recruiters looking to collect job market data, analyze trends, or track specific job opportunities over time.

**TECHNOLIGIES USED:**

1. **Python:** o Primary programming language for implementing the scraper.
2. **Libraries:**
   * **BeautifulSoup:** For parsing and navigating HTML content. o **Requests:** To make HTTP requests to Indeed’s job listing pages.
   * **Selenium:** For scraping dynamically loaded content or handling JavaScript-heavy pages.
   * **Pandas:** To structure, filter, and store data in CSV, JSON, or Excel files. 
   * **Threading/Asyncio:** For multi-threaded or asynchronous scraping to handle large data volumes efficiently.
   * **Proxy Middleware:** To manage IP addresses and avoid getting blocked by the website.
3. **Data Storage:**
   * CSV, JSON, or Excel for storing scraped job data in a structured format.
4. **Browser Automation:**
   * Selenium integrated with headless browsers (e.g., ChromeDriver or GeckoDriver) for scraping pages with dynamic content.

**FEATURES:**

1. **Job Listing Search:**
   * Search and extract job listings based on keywords, location, and job type (full-time, part-time, remote).
2. **Job Details Extraction:**
   * Gather information like job title, company, location, salary, job description, and date posted.
3. **Filter and Sort Jobs:** o Filter results by date posted, salary range, or location for more refined searches.
4. **Multiple Page Scraping:**
   * Automatically scrape multiple pages of job listings to gather comprehensive data.
5. **Save Data:**
   * Store extracted job details in CSV, JSON, or Excel formats for further analysis or use.
6. **Schedule Scraping:**
   * Implement scheduled scraping to track new job postings and trends over time (for regular data collection).
7. **Company Reviews & Ratings:**
   * Scrape company ratings and reviews for job listings to help job seekers assess potential employers.

**REFERENCE:**

<https://youtu.be/bEK5vRRTqrs?si=MuZprMaax8zYVf4U> – Youtube link

<https://console.apify.com/actors> - APIFY link

<https://colab.research.google.com/drive/1VnYO_8chclmHZlXQPDrM71IUL-> - Google colab link

**CODING:**

**from apify\_client import ApifyClient**

**import pandas as pd**

**# Initialize the ApifyClient with your API token**

**client = ApifyClient("apify\_api\_UGfkd9nxsLanenISjRRKtQHZDrVJC02aLXKw")**

**# Prepare the Actor input**

**run\_input = {**

**"position": "web developer",**

**"country": "US",**

**"location": "San Francisco",**

**"maxItems": 50,**

**"parseCompanyDetails": False,**

**"saveOnlyUniqueItems": True,**

**"followApplyRedirects": False,**

**}**

**# Run the Actor and wait for it to finish**

**run = client.actor("hMvNSpz3JnHgl5jkh").call(run\_input=run\_input)**

**# Collect the results into a list of dictionaries**

**data = []**

**for item in client.dataset(run["defaultDatasetId"]).iterate\_items():**

**data.append(item)**

**# Convert the data into a DataFrame**

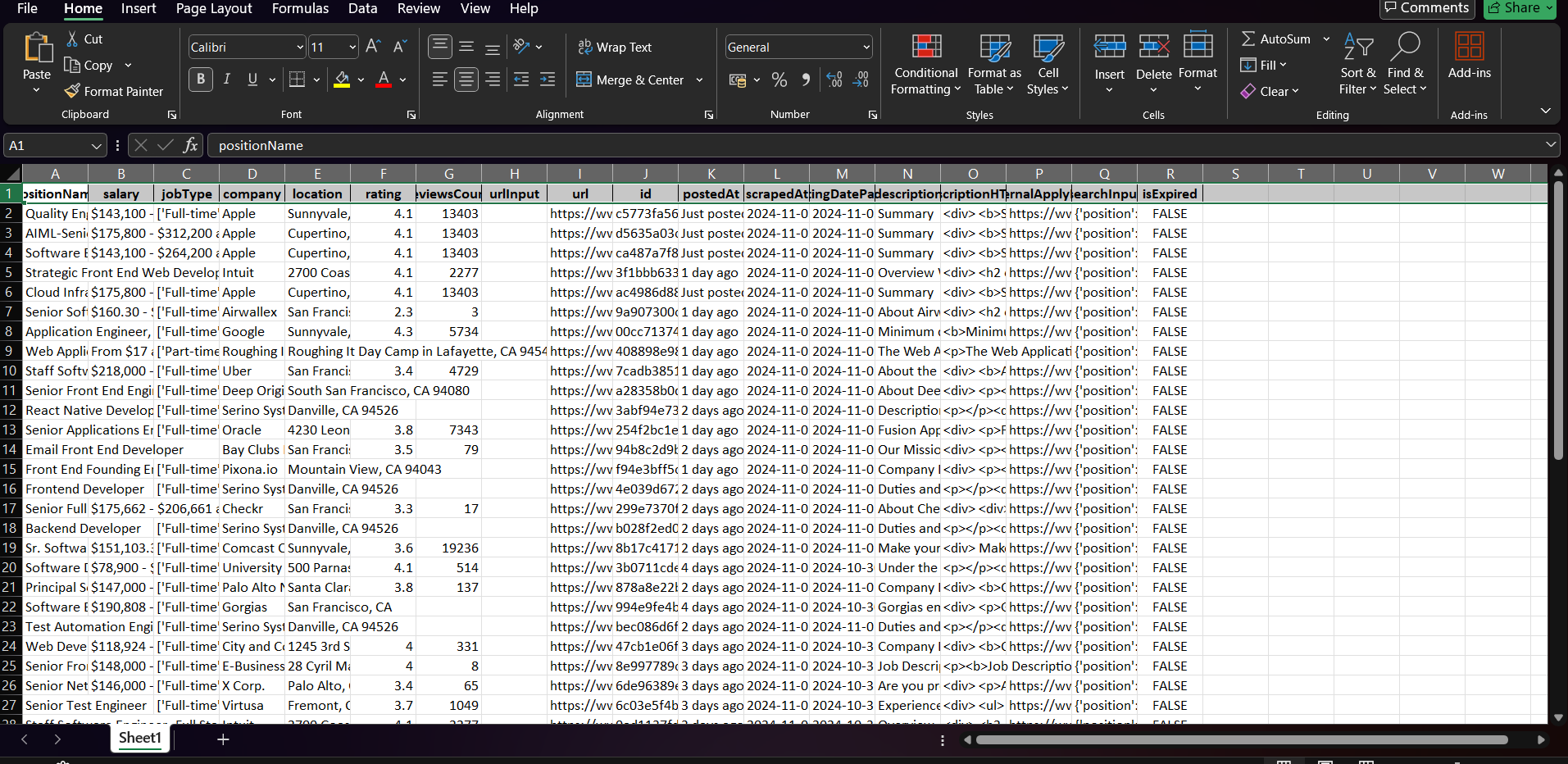
**df = pd.DataFrame(data)**

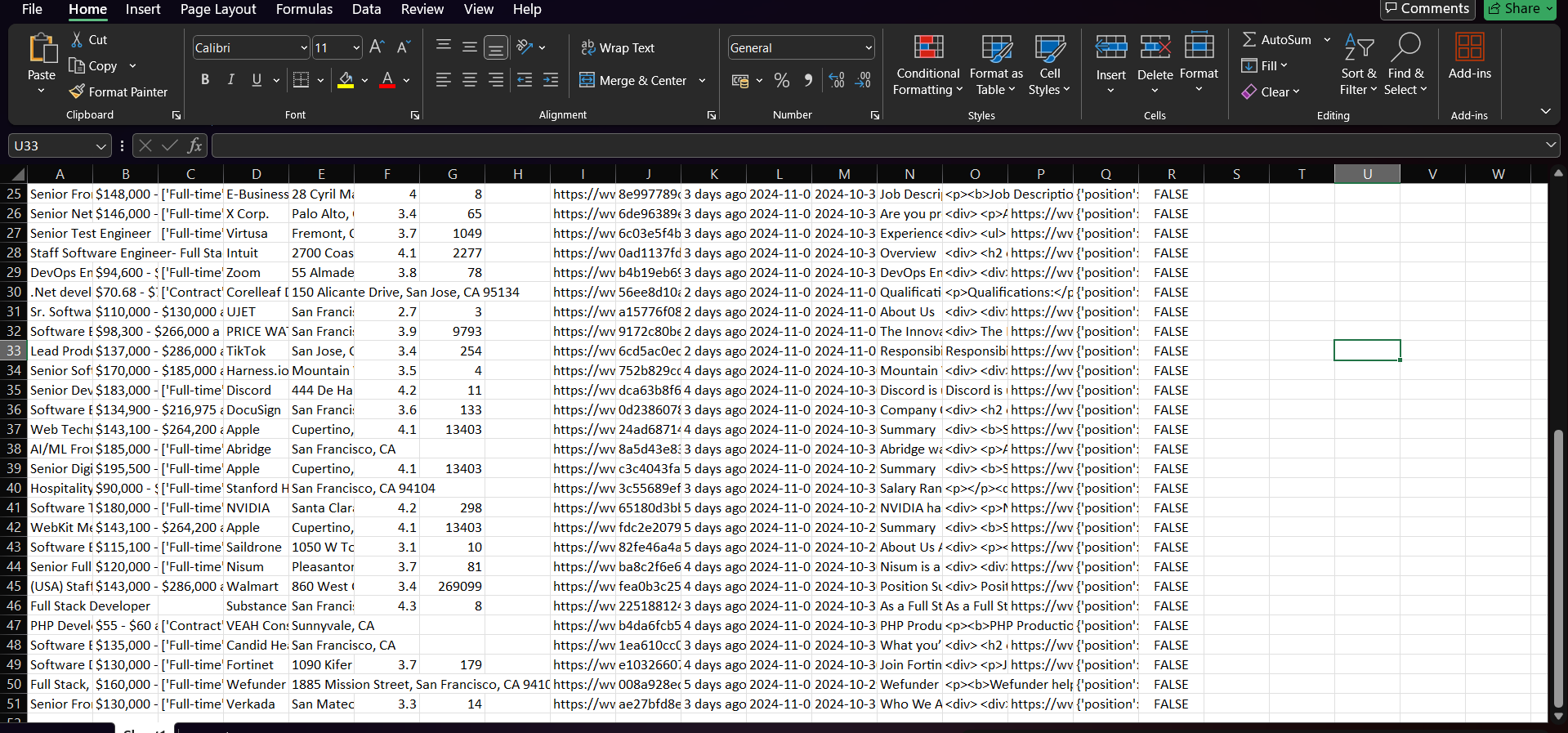
**# Save the DataFrame to an Excel file**

**df.to\_excel("scraped\_data.xlsx", index=False)**

**print("Data has been successfully saved to 'scraped\_data.xlsx'")**

**OUTPUT:**

****

****

**DONE BY :**

**ELAMPRITHIKA.M**