

job_scraper

July 9, 2025

1 Web Scraping Job Vacancies

1.1 Introduction

In this project, we'll build a web scraper to extract job listings from a popular job search platform. We'll extract job titles, companies, locations, job descriptions, and other relevant information.

Here are the main steps we'll follow in this project:

1. Setup our development environment
2. Understand the basics of web scraping
3. Analyze the website structure of our job search platform
4. Write the Python code to extract job data from our job search platform
5. Save the data to a CSV file
6. Test our web scraper and refine our code as needed

1.2 Prerequisites

Before starting this project, you should have some basic knowledge of Python programming and HTML structure. In addition, you may want to use the following packages in your Python environment:

- requests
- BeautifulSoup
- csv
- datetime

These packages should already be installed in Coursera's Jupyter Notebook environment, however if you'd like to install additional packages that are not included in this environment or are working off platform you can install additional packages using `!pip install packagename` within a notebook cell such as:

- `!pip install requests`
- `!pip install BeautifulSoup`

1.3 Step 1: Importing Required Libraries

```
[1]: import requests
    from bs4 import BeautifulSoup
    import pandas as pd
    import matplotlib.pyplot as plt
    import seaborn as sns

[2]: url = "https://remoteok.com/"
    headers = {'User-Agent': 'Mozilla/5.0'}

[5]: response = requests.get(url, headers=headers)
    soup = BeautifulSoup(response.content, "html.parser")

[6]: jobs = soup.find_all('tr', class_='job') # RemoteOK structures each job inside
    ↪ a <tr class='job'>

[7]: job_titles = []
    companies = []
    locations = []
    date_posted = []
    job_links = []
    for job in jobs:
        title = job.find('h2')
        company = job.find('h3')
        location = job.find('div', class_='location')
        link = job.get('data-href')

        if title and company:
            job_titles.append(title.text.strip())
            companies.append(company.text.strip())
            locations.append(location.text.strip() if location else "Remote")
            date_posted.append("Today") # RemoteOK doesn't show dates by default
            job_links.append("https://remoteok.com" + link)

[8]: df = pd.DataFrame({
    "Job Title": job_titles,
    "Company": companies,
    "Location": locations,
    "Date Posted": date_posted,
    "Job Link": job_links
})

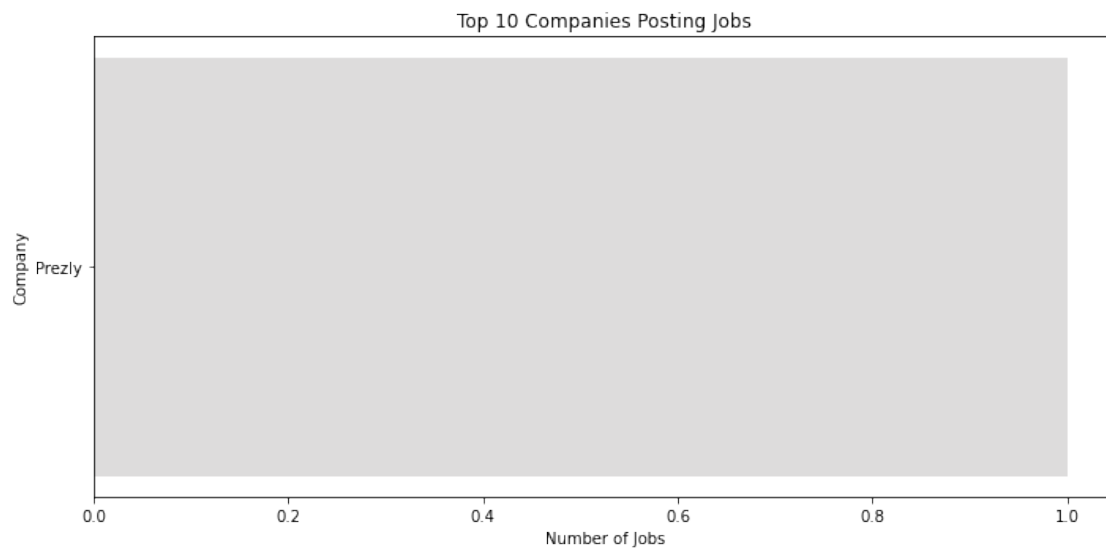
[9]: df.to_csv("job_data.csv", index=False)
    print("Scraped", len(df), "job postings and saved to 'job_data.csv'.")
```

Scraped 1 job postings and saved to 'job_data.csv'.

```
[10]: print("\nTop 5 Companies Posting Jobs:")
      print(df['Company'].value_counts().head())
```

```
Top 5 Companies Posting Jobs:
Prezly      1
Name: Company, dtype: int64
```

```
[11]: plt.figure(figsize=(10, 5))
      top_companies = df['Company'].value_counts().nlargest(10)
      sns.barplot(x=top_companies.values, y=top_companies.index, palette="coolwarm")
      plt.title("Top 10 Companies Posting Jobs")
      plt.xlabel("Number of Jobs")
      plt.ylabel("Company")
      plt.tight_layout()
      plt.show()
```



```
[ ]:
```