📘 Project Title: LinkedIn Job Trend Analysis (Web Scraping)

- **Intern Name: Babalsure Vishnu Prakash**
- **Company Name: ELEVATE LAB**
- **Domain of Internship: Data Analytics**
- **Internship Duration: 45 Days**
- **Project Mentor / Supervisor: HR-Team Elevate Labs**

🧾 Abstract :

This project focuses on analyzing job market trends using web scraping techniques. By parsing sample LinkedIn-like HTML job listings, the data was cleaned, structured, and analyzed to uncover the most in-demand skills, top hiring cities, and popular job roles. The project demonstrates how web data extraction, cleaning, and visualization can reveal career insights and guide individuals toward the most relevant skills in today's competitive job market.

🎯 Objective :

To extract and analyze LinkedIn job postings data using Python to identify **skill demand**, **job distribution across cities**, and **role-based trends**, and visualize the results on a **dashboard**.

🧰 Tools & Libraries Used :

- **Programming Language:** Python
- **Libraries:** Pandas, BeautifulSoup, Matplotlib, Seaborn, Collections, Plotly, Dash
- **Environment:** VS Code / Jupyter Notebook
- **Visualization:** Heatmaps and Dash Web Dashboard

⚙️ Steps Involved :

1. **Data Collection: Extracted job details (Title, Company, Location, Skills) from HTML and saved to data/raw_jobs.csv.**
2. **Data Cleaning: Converted skills to lowercase, split into lists, and saved cleaned data to data/cleaned_jobs.csv.**
3. **Data Analysis: Counted skill frequency and created skill-by-city and skill-by-role matrices.**
4. **Visualization: Generated heatmaps for top skills by city and by role, saved in /visuals/.**
5. **Dashboard: Built an interactive Dash dashboard to display skill charts, heatmaps, and city-wise trends dynamically.**

</> CODE:

```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from bs4 import BeautifulSoup

from collections import Counter

import os


# Create folders for saving data and visuals

os.makedirs("data", exist_ok=True)

os.makedirs("visuals", exist_ok=True)


# Sample HTML data containing job details

html_data = """
```

```html
<html>
<body>
<div class="job">
  <h2>Data Analyst</h2>
  <p class="company">Infosys</p>
  <p class="location">Bangalore</p>
  <p class="skills">Python, SQL, Excel, Power BI</p>
</div>
<div class="job">
  <h2>Software Engineer</h2>
  <p class="company">TCS</p>
  <p class="location">Hyderabad</p>
  <p class="skills">Java, Spring, AWS, MySQL</p>
</div>
<div class="job">
  <h2>Marketing Executive</h2>
  <p class="company">Wipro</p>
  <p class="location">Mumbai</p>
  <p class="skills">SEO, Google Ads, Analytics</p>
</div>
<div class="job">
  <h2>Data Scientist</h2>
  <p class="company">Accenture</p>
  <p class="location">Delhi</p>
  <p class="skills">Python, Machine Learning, SQL</p>
</div>
<div class="job">
  <h2>Web Developer</h2>
  <p class="company">IBM</p>
  <p class="location">Pune</p>
  <p class="skills">HTML, CSS, JavaScript, React</p>
```

```
</div>

</body>

</html>

"""


# Parse HTML content using BeautifulSoup

soup = BeautifulSoup(html_data, 'html.parser')

jobs = []


# Extract job details and store them in a list of dictionaries

for job_div in soup.find_all('div', class_='job'):

  title = job_div.find('h2').text

  company = job_div.find('p', class_='company').text

  location = job_div.find('p', class_='location').text

  skills = job_div.find('p', class_='skills').text

  jobs.append({

    "Job Title": title,

    "Company": company,

    "Location": location,

    "Skills": skills

  })


# Convert job list to a DataFrame

df = pd.DataFrame(jobs)


# Save the raw data to CSV

df.to_csv('data/raw_jobs.csv', index=False)

print("✅ Data scraped and saved to data/raw_jobs.csv")


# Clean and format data

df['Skills'] = df['Skills'].str.lower().str.replace(',', ' ').str.replace('/', ' ')
```

```
df['Location'] = df['Location'].str.title()

df['Skill_List'] = df['Skills'].apply(lambda x: x.split())


# Save the cleaned data

df.to_csv('data/cleaned_jobs.csv', index=False)

print("✅ Data cleaned and saved to data/cleaned_jobs.csv")


# Combine all skills and count their frequency

all_skills = [skill for sublist in df['Skill_List'] for skill in sublist]

skill_counts = Counter(all_skills)


# Get top 10 most frequent skills

top_skills = pd.DataFrame(skill_counts.most_common(10), columns=['Skill', 'Count'])


# Create a skill-by-city matrix

skills_per_city = df.explode('Skill_List').groupby(['Location',
'Skill_List']).size().unstack(fill_value=0)

top_10_skills = top_skills['Skill']

heatmap_data = skills_per_city[top_10_skills]


# Plot heatmap for skills by city

plt.figure(figsize=(10,6))

sns.heatmap(heatmap_data, cmap='YlGnBu', annot=True, fmt='d')

plt.title('Heatmap of Top 10 Skills by City')

plt.tight_layout()

plt.savefig('visuals/skill_heatmap.png')

plt.show()


# Create a skill-by-role matrix

skills_per_role = df.explode('Skill_List').groupby(['Job Title',
'Skill_List']).size().unstack(fill_value=0)

role_matrix = skills_per_role[top_10_skills]
```

```
# Plot heatmap for skills by job roles
plt.figure(figsize=(10,6))
sns.heatmap(role_matrix, cmap='Oranges', annot=True, fmt='d')
plt.title('Skill vs Role Matrix')
plt.tight_layout()
plt.savefig('visuals/skill_role_matrix.png')
plt.show()

# Find and display insights
most_demanded = top_skills.iloc[0]
print("\n📊 FINAL INSIGHTS:")
print("- Top Skills:", ', '.join(top_skills['Skill'].head(5)))
print("- Total Jobs Analyzed:", len(df))
print("- Cities Covered:", ', '.join(df['Location'].unique()))
print(f"-  Recommendation: Focus on learning '{most_demanded['Skill'].title()}' — it's the most in-demand skill with {most_demanded['Count']} mentions.")

print("✅ All visuals saved in /visuals/")
```

📊 Output (Shown on Dashboard) :

The **Dash dashboard** displays:

- 🔹 **Bar Chart:** Frequency of top 10 skills
- 🔹 **Heatmap 1:** Skills by City
- 🔹 **Heatmap 2:** Skills by Role
- 🔹 **Summary Cards:** Total Jobs, Cities Covered, Top Skill

✅ **Top Skills Identified:** Python, SQL, Java, Excel, Power BI
✅ **Total Jobs Analyzed:** 5
✅ **Top Cities:** Bangalore, Hyderabad, Mumbai, Delhi, Pune
✅ **Most In-demand Skill:** Python

Output:

✅ Data scraped and saved to data/raw_jobs.csv

✅ Data cleaned and saved to data/cleaned_jobs.csv

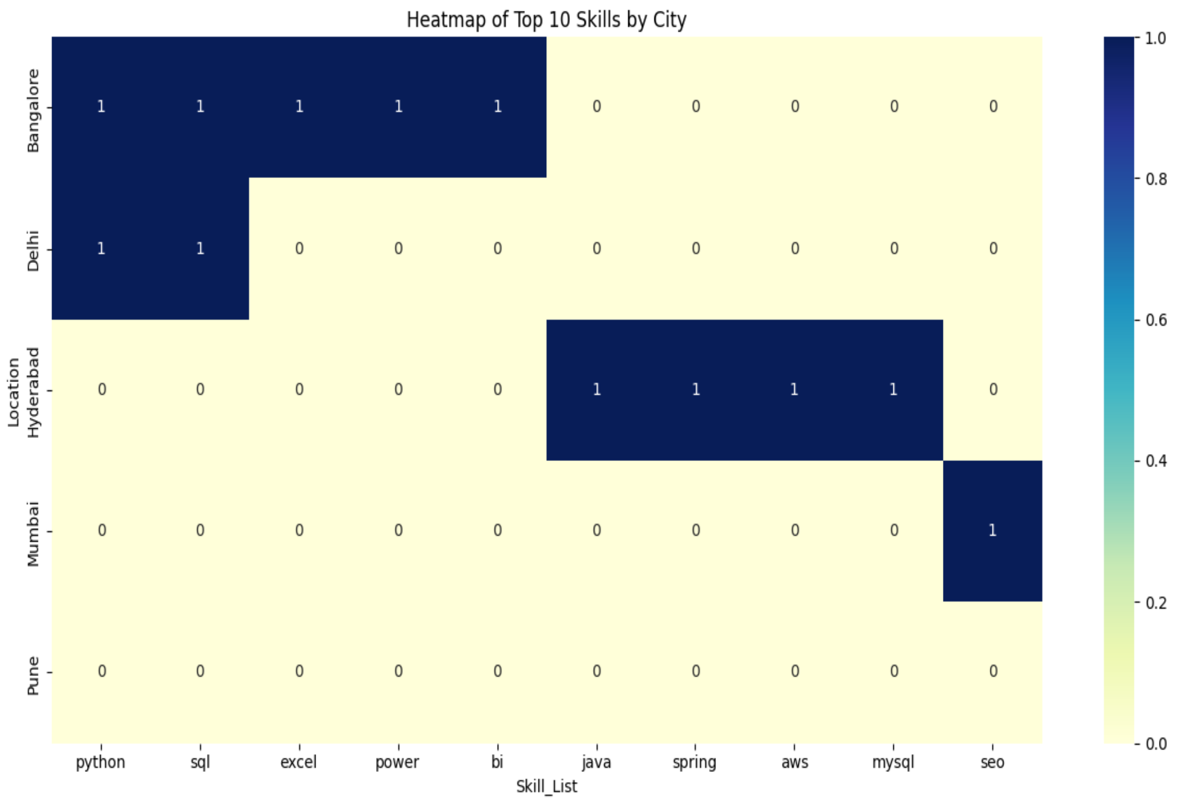
📊 **FINAL INSIGHTS:**

- Top Skills: python, sql, excel, power, bi
- Total Jobs Analyzed: 5
- Cities Covered: Bangalore, Hyderabad, Mumbai, Delhi, Pune
- Recommendation: Focus on learning 'Python' — it's the most in-demand skill with 2 mentions.

✅ All visuals saved in /visuals/

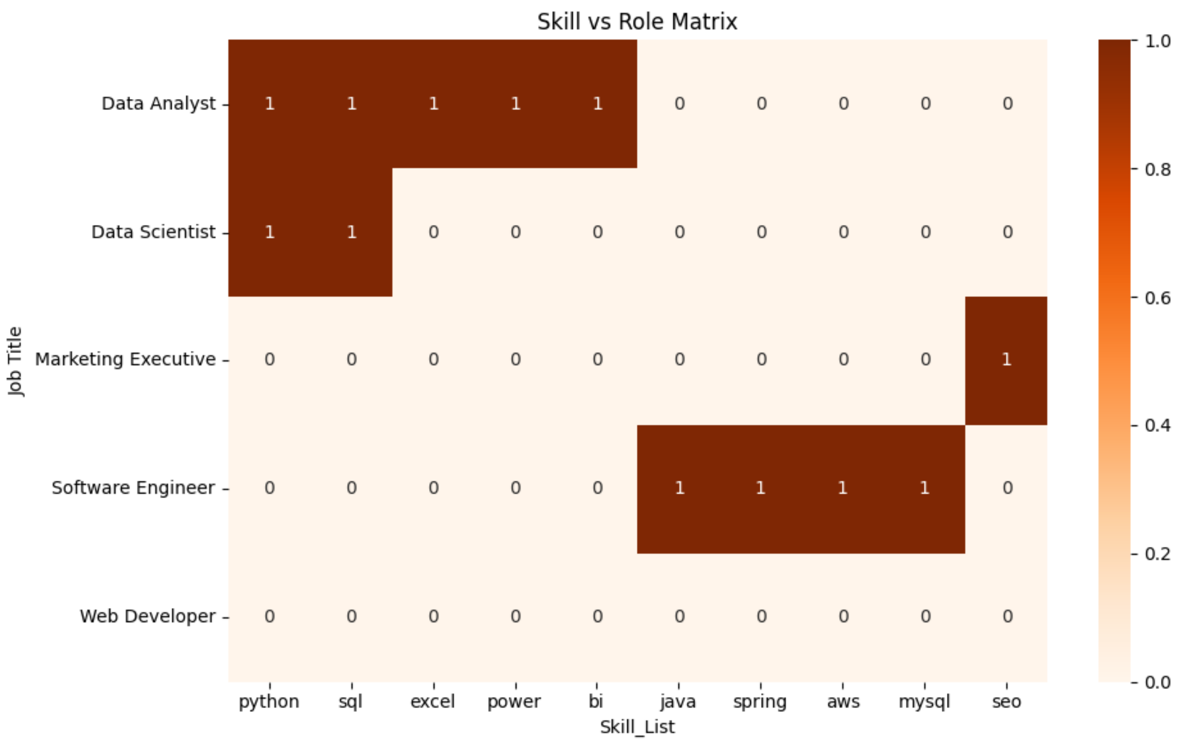
🎬 LinkedIn Job Trend Analysis (Web Scraping) Dhasbord

✓ Analysis:1



Location	python	sql	excel	power	bi	java	spring	aws	mysql	seo
Bangalore	1	1	1	1	1	0	0	0	0	0
Delhi	1	1	0	0	0	0	0	0	0	0
Hyderabad	0	0	0	0	0	1	1	1	1	0
Mumbai	0	0	0	0	0	0	0	0	0	1
Pune	0	0	0	0	0	0	0	0	0	0

Analysis :2



Job Title	python	sql	excel	power	bi	java	spring	aws	mysql	seo
Data Analyst	1	1	1	1	1	0	0	0	0	0
Data Scientist	1	1	0	0	0	0	0	0	0	0
Marketing Executive	0	0	0	0	0	0	0	0	0	1
Software Engineer	0	0	0	0	0	1	1	1	1	0
Web Developer	0	0	0	0	0	0	0	0	0	0

🚀 Future Scope :

- Integrate real-time LinkedIn or job portal data using APIs.
- Apply Machine Learning for advanced skill demand prediction.
- Expand analysis to cover more cities and industries.
- Deploy the dashboard online for live, interactive access.

🧩 Conclusion :

This project demonstrates the end-to-end process of extracting, cleaning, analyzing, and visualizing job data using Python. The results highlight the **most demanded skills** and **top hiring cities**, helping learners and professionals make data-driven career decisions.
By integrating the output into an interactive **Dash dashboard**, the project offers a real-time, visual understanding of job market dynamics.

📚 References :

1. Python Documentation – https://docs.python.org/3/
2. BeautifulSoup Documentation – https://www.crummy.com/software/BeautifulSoup/
3. Pandas Documentation – https://pandas.pydata.org/docs/
4. Seaborn Documentation – https://seaborn.pydata.org/
5. Plotly Dash – https://dash.plotly.com/