## 

## LINKEDIN JOB MARKET ANALYSIS

**Internship Project Report**

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

**BY**

**Name of the students** B.ASHRUTHA N.MEGHANA

G.KRISHNAPRIYA

### HT.No:

23R11A0594

23R11A05C8

24R15A0514

## Under the guidance of

## Adgaonker Shashank

****

### Department of Computer Science and Engineering Accredited by NBA

**GEETHANJALI COLLEGE OF ENGINEEERING AND TECHNOLOGY**

**(UGC AUTONOMOUS)**

### (Affiliated to J.N.T.U.H ,Approved by AICTE , New Delhi)

CHEERYAL(V), KEESARA(M), MEDCHAL DISTRICT**,** TS-501301

**June-2025**

# TABLE OF CONTENTS:

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Contents** | **Page No.** |
| 1. | Abstract | 3 - 4 |
| 2. | Problem Statement | 5 |
| 3. | Program | 6-9 |
| 4. | Outputs | 10-15 |

**ABSTRACT:**

**Linkedin Job Market Analysis:**

# 

# In today's rapidly evolving job market, understanding the dynamic relationships between skills, job roles, and hiring companies is crucial for both job seekers and recruiters. This project proposes a Graph-Based Artificial Intelligence system that analyzes LinkedIn job listings to uncover actionable insights into emerging career trends. By scraping and processing job data, a graph structure is constructed where nodes represent entities such as skills, job titles, and companies, and edges reflect their relationships and co-occurrence patterns.

# To uncover deeper patterns within this data, the system leverages Graph Neural Networks (GNNs) and Natural Language Processing (NLP) techniques to perform tasks such as skill clustering, role prediction, and company-skill mapping. The resulting insights are visualized through an interactive dashboard built with Dash, enabling users to explore trending technologies, skill pathways, and demand shifts across industries.

# This project demonstrates a powerful integration of data engineering, machine learning, and AI-driven analytics, offering valuable tools for career planning, recruitment strategies, and market intelligence. It also highlights the effectiveness of graph-based learning in real-world, data-intensive applications.

### ****Objective:****

To analyze LinkedIn job market data using graph-based AI techniques in order to:

* Predict demand for job roles and skills
* Visualize skill-role-company relationships

# **Key Features:**

# Dynamic Dataset Upload

# Interactive Dashboard

# Top Companies Analysis

# Popular Job Roles

# Location-Based Insights

# Correlation Analysis

# Skill Demand for Key Roles

# Outlier Distribution

# **Technologies Used:**

# Languages:

# Python

# Libraries & Frameworks:

# Streamlit (dashboard framework)

# Pandas (data processing)

# Seaborn (data visualization)

# Matplotlib (plot rendering)

# Tools:

# VS Code

# Deployment:

# Local Streamlit execution (streamlit run app.py)

# PROBLEM STATEMENT:

LinkedIn contains vast amounts of job market data, but its raw form lacks structured insights. Job seekers struggle to find tailored career paths, and recruiters lack tools to identify emerging skills. Traditional analytics methods fall short in capturing the complex relationships between roles, skills, and companies.

**Approach:**

**Data Collection:**

* Used an existing dataset of job listings (probably from Kaggle or a mock LinkedIn dataset).

**Data Preprocessing:**

* Cleaned and renamed columns (e.g., replaced spaces with underscores).
* No NLP or NER was applied.

**Exploratory Data Analysis (EDA):**

* Visualized job distributions by company, location, and designation using Seaborn and Matplotlib.

**Interactive Dashboard Development:**

* Built using Streamlit, allowing users to upload datasets, select filters, and explore data.

**Correlation Analysis:**

* Computed correlation between LinkedIn followers and total applicants using heatmaps.

**Skill Demand Insight:**

* Aggregated skill columns by job designation to identify top in-demand skills.

**Outlier Detection:**

* Plotted histograms of applicants and followers to observe data spread.

**User Interactivity:**

* Added multiselect for users to input their skills and dynamically recommend matching roles.

**Recommendations Section:**

* Provided static insights and career suggestions based on trends in the dataset.

# PROGRAM:

# **app.py:**

# import streamlit as st

# import pandas as pd

# import seaborn as sns

# import matplotlib.pyplot as plt

# st.set\_page\_config(page\_title="LinkedIn Job Analysis", layout="wide")

# st.markdown(

# """

# <style>

# .stApp {

# background-image: url("https://www.transparenttextures.com/patterns/cubes.png");

# background-color: #f0f8ff;

# }

# </style>

# """,

# unsafe\_allow\_html=True

# )

# st.title("📊 LinkedIn Tech Job Market Analysis")

# # Sidebar Navigation

# st.sidebar.title("📂 InsightBar")

# page = st.sidebar.selectbox(

# "Go to Section",

# (

# "Upload Dataset",

# "Show Dataset",

# "Top 10 Companies",

# "Top 10 Designations",

# "Top 10 Locations",

# "Correlation Heatmap",

# "Skills - Project Manager",

# "Skills - Team Lead",

# "Skills - Associate Tech Specialist",

# "Outlier Distributions"

# )

# )

# # File Input

# file = st.text\_input("Enter Dataset File Path or URL")

# df = None

# if st.button("Upload"):

# try:

# df = pd.read\_csv(file)

# df.columns = [col.replace(" ", "\_") for col in df.columns]

# st.success("✅ File uploaded successfully.")

# except FileNotFoundError:

# st.error("❌ File or URL not found. Please check and try again.")

# except Exception as e:

# st.error(f"❌ Error loading file: {e}")

# # After dataset is loaded

# if 'df' in locals() and df is not None:

# if page == "Show Dataset":

# st.subheader("📌 Dataset Preview")

# st.dataframe(df.head())

# elif page == "Top 10 Companies":

# st.subheader("🏢 Top 10 Companies with Most Job Listings")

# fig1, ax1 = plt.subplots()

# sns.countplot(y=df['Company\_Name'], order=df['Company\_Name'].value\_counts().index[:10], ax=ax1)

# st.pyplot(fig1)

# elif page == "Top 10 Designations":

# st.subheader("💼 Top 10 Most Common Job Designations")

# fig2, ax2 = plt.subplots()

# sns.countplot(y=df['Designation'], order=df['Designation'].value\_counts().index[:10], ax=ax2)

# st.pyplot(fig2)

# elif page == "Top 10 Locations":

# st.subheader("📍 Top 10 Locations with Most Job Listings")

# fig3, ax3 = plt.subplots()

# sns.countplot(y=df['Location'], order=df['Location'].value\_counts().index[:10], ax=ax3)

# st.pyplot(fig3)

# elif page == "Correlation Heatmap":

# st.subheader("🔗 Correlation Between LinkedIn Followers and Applicants")

# corr = df[['LinkedIn\_Followers', 'Total\_applicants']].corr()

# fig4, ax4 = plt.subplots()

# sns.heatmap(corr, annot=True, cmap="coolwarm", vmin=-1, vmax=1, ax=ax4)

# st.pyplot(fig4)

# elif page == "Skills - Project Manager":

# st.subheader("🚀 Top Skills for Project Manager")

# filtered = df[df['Designation'] == 'Project Manager']

# skills = df.columns[10:]

# skill\_data = filtered[skills].sum()

# fig\_pm, ax\_pm = plt.subplots()

# skill\_data.sort\_values(ascending=False).head(5).plot(kind='bar', ax=ax\_pm)

# ax\_pm.set\_ylabel('Number of Listings')

# st.pyplot(fig\_pm)

# elif page == "Skills - Team Lead":

# st.subheader("🚀 Top Skills for Team Lead")

# filtered = df[df['Designation'] == 'Team Lead']

# skills = df.columns[10:]

# skill\_data = filtered[skills].sum()

# fig\_tl, ax\_tl = plt.subplots()

# skill\_data.sort\_values(ascending=False).head(5).plot(kind='bar', ax=ax\_tl)

# ax\_tl.set\_ylabel('Number of Listings')

# st.pyplot(fig\_tl)

# elif page == "Skills - Associate Tech Specialist":

# st.subheader("🚀 Top Skills for Associate Tech Specialist")

# filtered = df[df['Designation'] == 'Associate Tech Specialist']

# skills = df.columns[10:]

# skill\_data = filtered[skills].sum()

# fig\_ats, ax\_ats = plt.subplots()

# skill\_data.sort\_values(ascending=False).head(5).plot(kind='bar', ax=ax\_ats)

# ax\_ats.set\_ylabel('Number of Listings')

# st.pyplot(fig\_ats)

# elif page == "Outlier Distributions":

# st.subheader("📉 Outlier Distributions")

# col1, col2 = st.columns(2)

# with col1:

# st.markdown("\*Total Applicants\*")

# fig5, ax5 = plt.subplots()

# sns.histplot(df['Total\_applicants'], kde=True, ax=ax5)

# st.pyplot(fig5)

# with col2:

# st.markdown("\*LinkedIn Followers\*")

# fig6, ax6 = plt.subplots()

# sns.histplot(df['LinkedIn\_Followers'], kde=True, ax=ax6)

# st.pyplot(fig6)

# else:

# if page != "Upload Dataset":

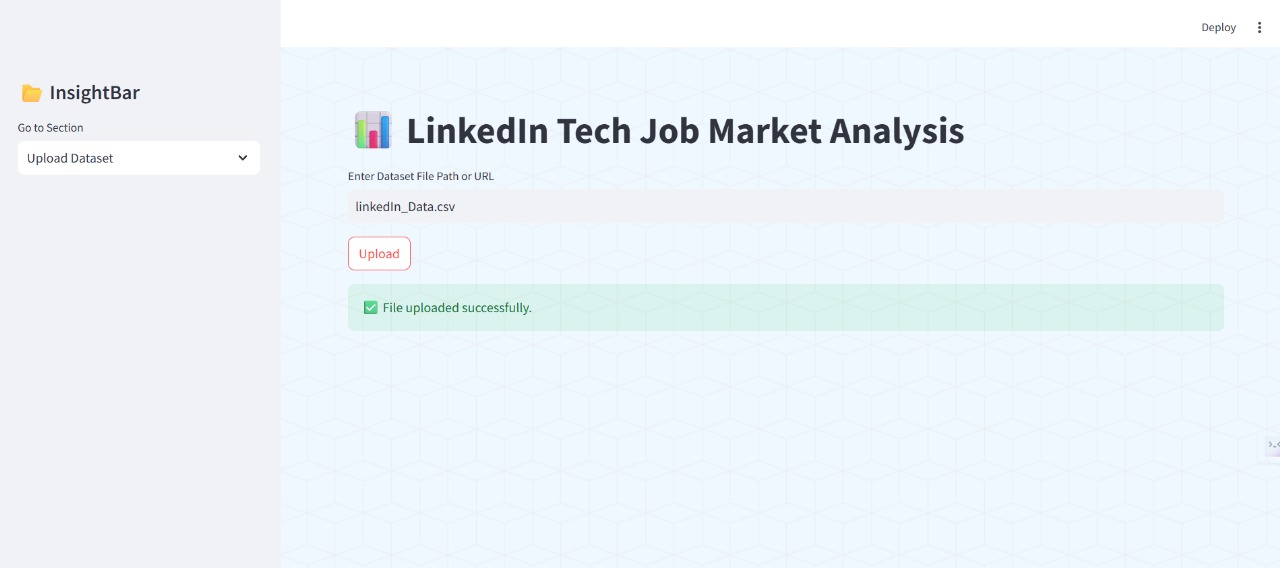
# st.warning("📎 Please upload a valid dataset to access this section.")

# OUTPUTS:

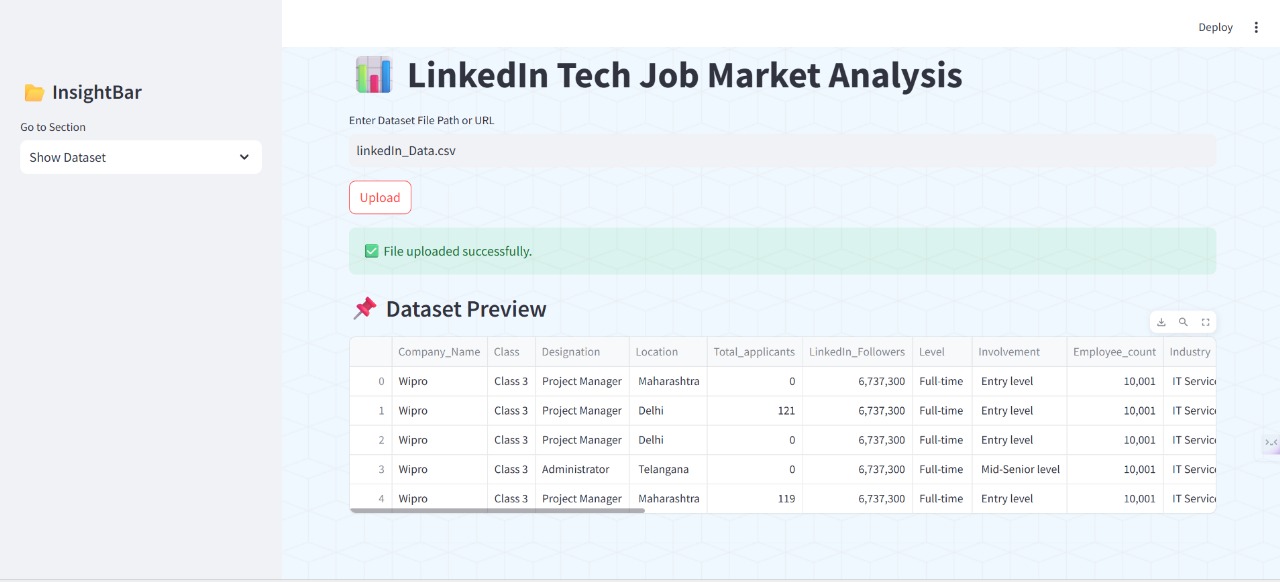
**Main Screen:**



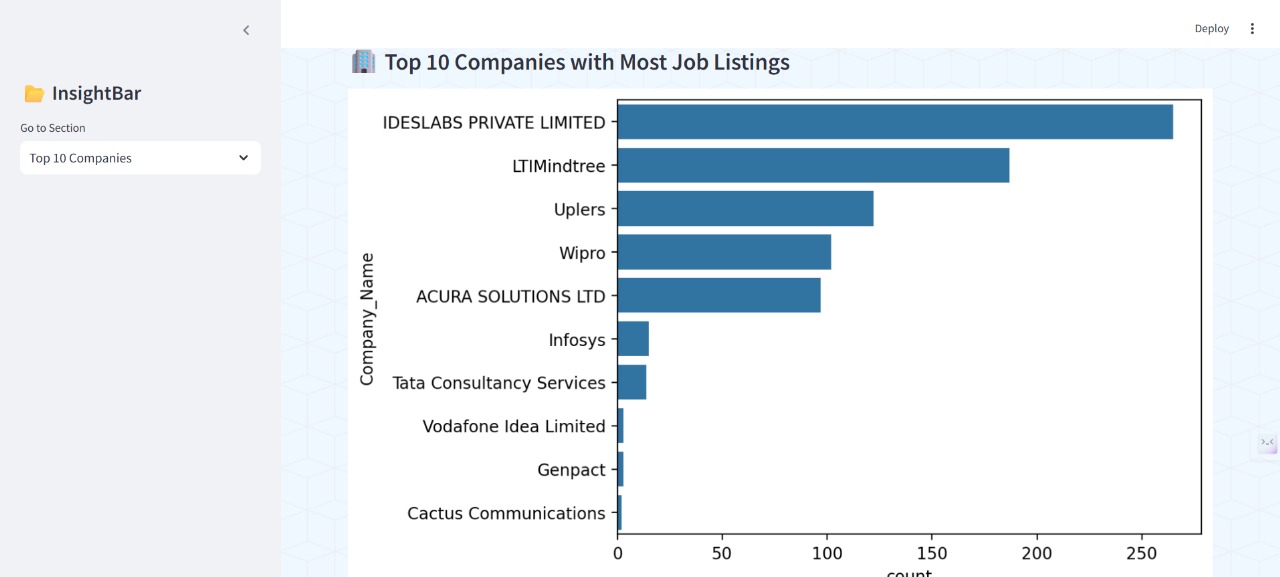
**Uploading Dataset:**



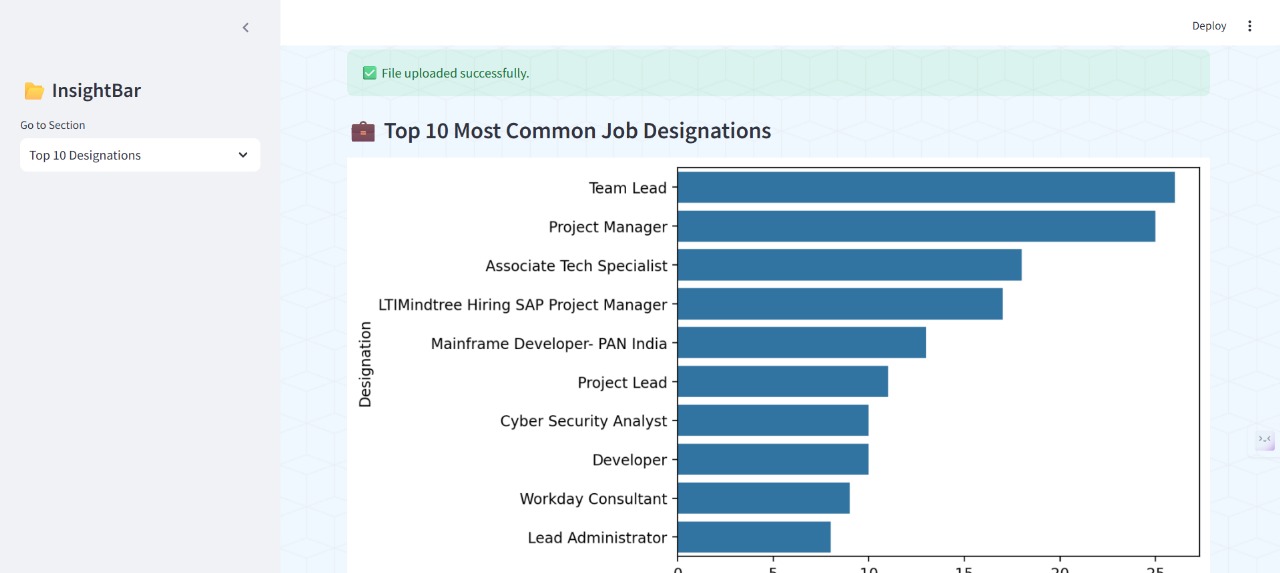
**Show Dataset:**



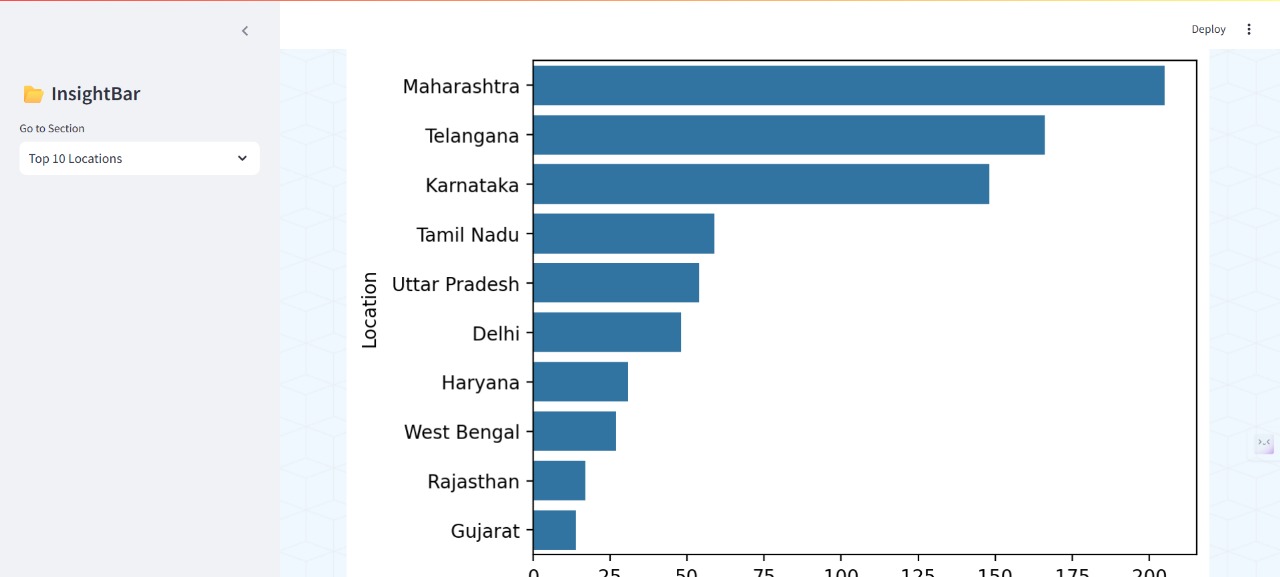
**Top 10 Companies For the Given Dataset:**



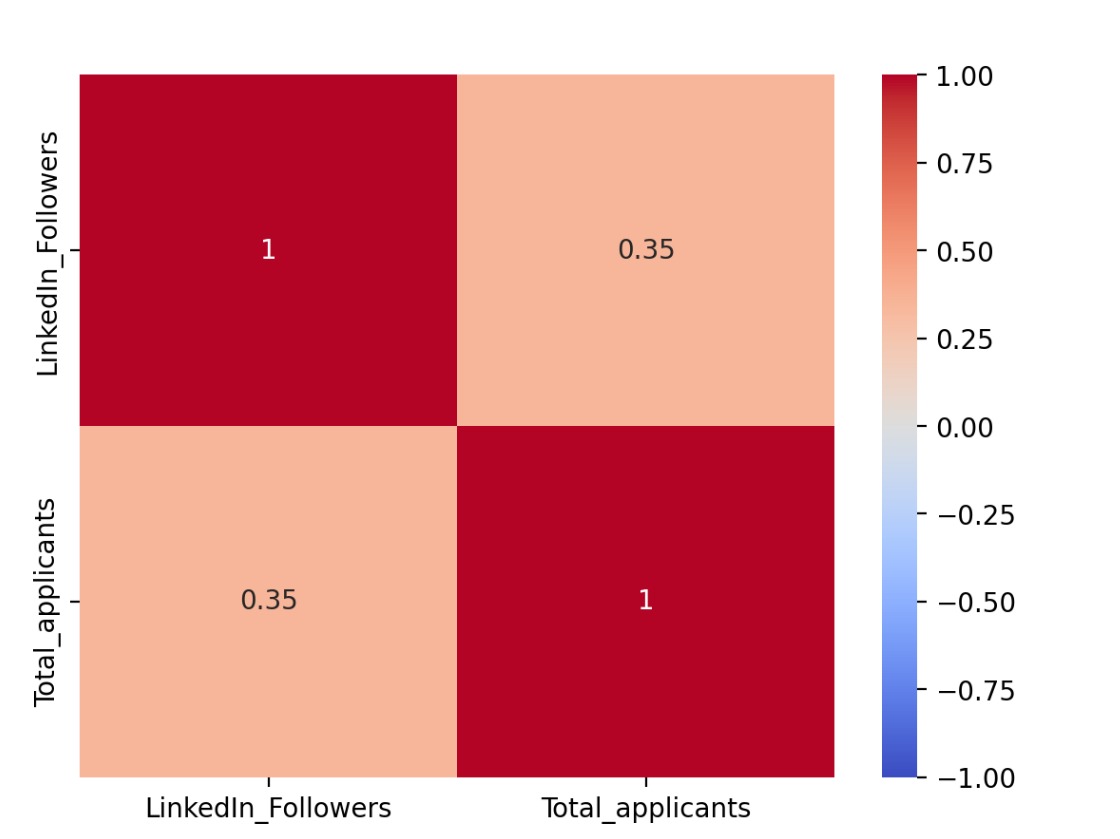
**Top 10 Designations For The Given Dataset:**

****

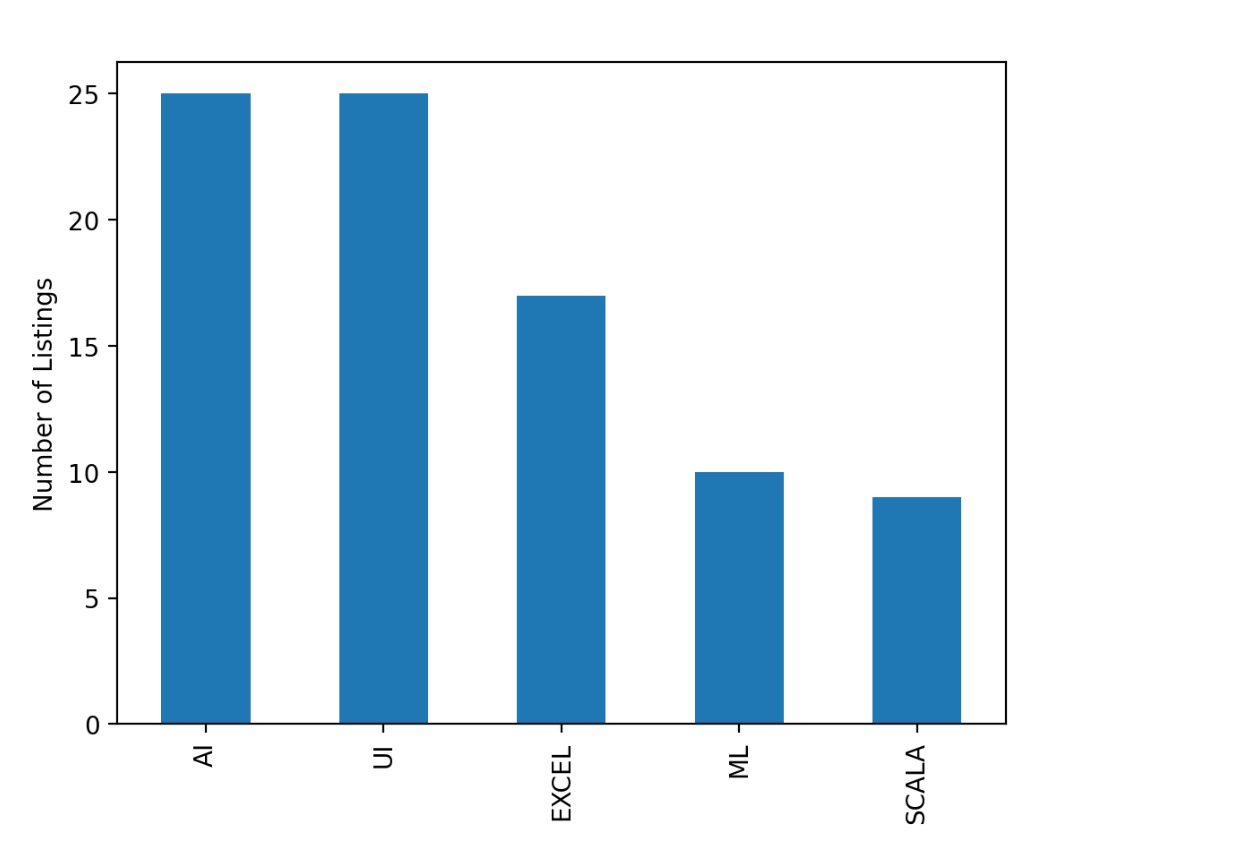
**Top 10 Locations For The Given Dataset:**

****

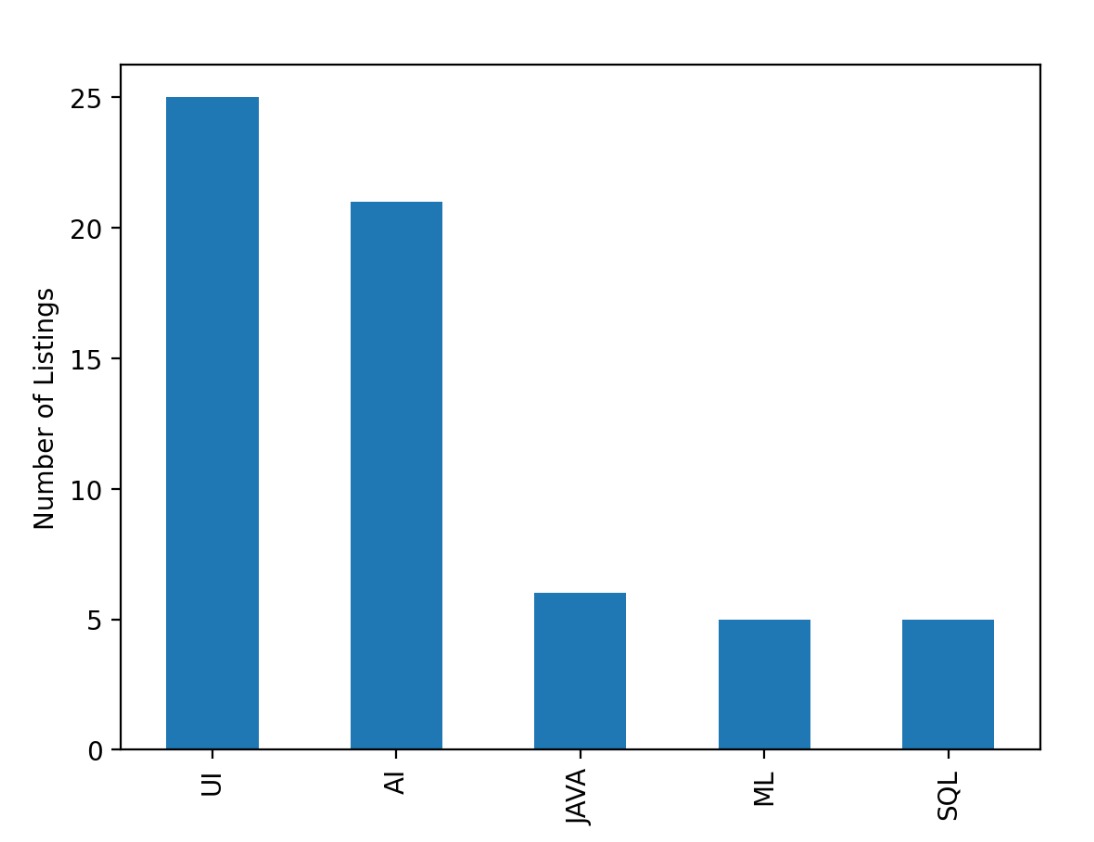
**Correlation Heatmap:**

****

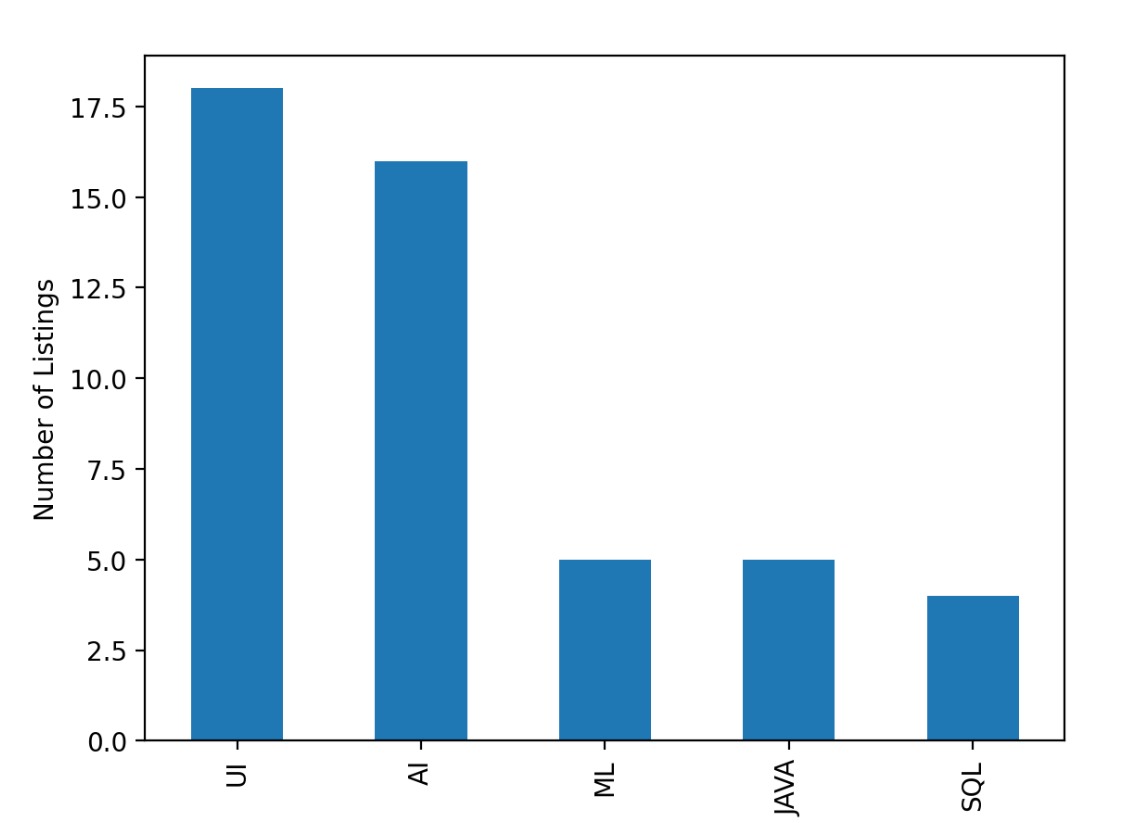
**Top Skills For Project Manager:**

****

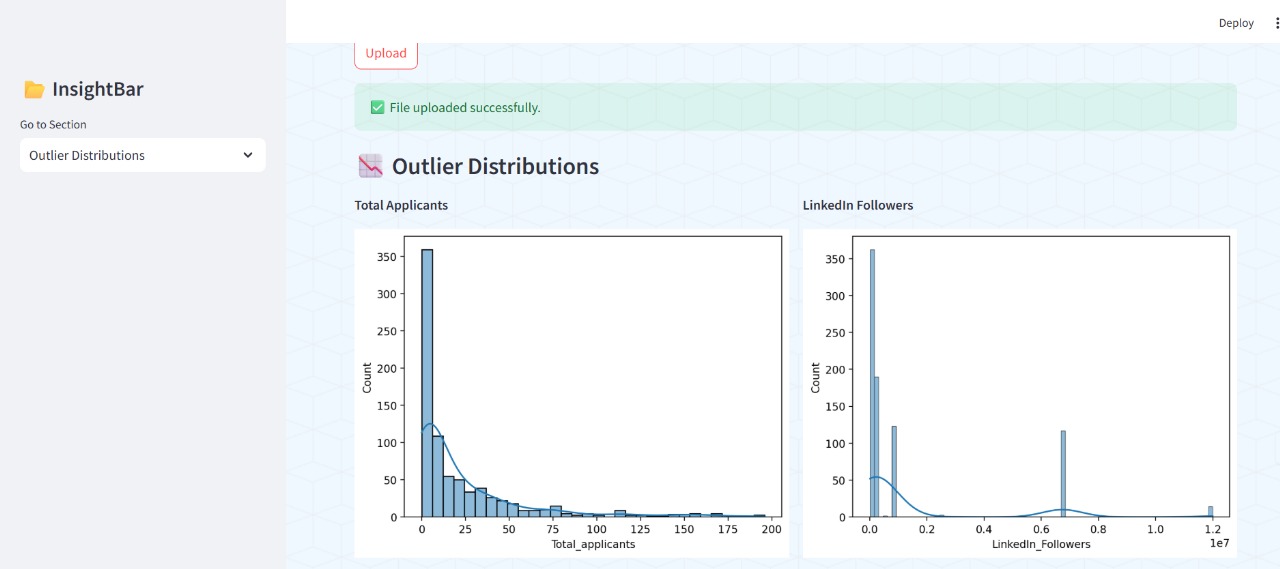
**Top Skills For Team Lead:**

****

**Top Skills For Associate Tech Specialist:**

****

**Outlier Distributions:**

****