An interim report for a dissertation that will be submitted in partial fulfilment of a University of Greenwich Master's Degree

Big Data-Driven Geospatial and Predictive Analysis of UK IT Employment Trends

Venkata Samba Siva Reddy Chilamuru

001410264

MSc Data Science

Date Proposal Submitted: 10/01/2025

Submission Date: 08/09/2025

Supervisor: Jia Wang

Topic Area: Big Data Analytics, Data Science, Geospatial Analysis, Employment Market Analysis

Keywords associated with the project: Big Data Analytics, Geospatial Data Analysis, UK IT Employment Trends, GIS for Labour Market Analysis, Employment Pattern Visualization.

MSc Modules studied that contribute towards this project: Data Visualization, Big data analysis, Geo-spatial Analysis

Abstract

This dissertation presents a data-driven approach to analyzing real-time employment trends in the UK IT sector using big data and geospatial analytics. Traditional labour market reports often lack timeliness and actionable insights. To address this gap, a fully automated system was developed using Python for web scraping, AWS S3 for cloud storage, and Power BI for interactive dashboards. The system collects job postings from LinkedIn and categorizes them by role, location, work type (remote, hybrid, on-site), and experience level. Key findings reveal that senior-level roles dominate current listings, on-site jobs are more common than remote or hybrid ones, and job postings are heavily concentrated in major cities like London, Manchester, and Birmingham. The Power BI dashboard offers interactive filtering, helping job seekers identify opportunities based on their preferences, while recruiters and policymakers can monitor workforce trends and respond accordingly. Despite its achievements, the project has limitations. Relying solely on LinkedIn may not capture the entire job market, and keyword-based classification has some accuracy constraints. Nonetheless, the system proves that real-time labour market monitoring is feasible and insightful. Future enhancements could include scraping additional platforms (Indeed, Glassdoor), applying advanced NLP techniques for better classification, integrating salary data, and adding predictive analytics for trend forecasting. The project demonstrates practical applications in employment planning, recruitment strategy, and policy development, offering a scalable model for similar analysis in other industries or regions.

Table of Contents

List of Figures

Chapter 1: Introduction

The UK's IT sector is developing rapidly and is very important to the country's economy, thanks to digital transformation, new business models, and people working remotely. Because job roles, skills needed, and types of employment are changing, real-time study of the labour market is getting more significant (Georgieff and Hyee, 2022). The reports that are currently used in the job market are often not up-to-date, too general, or don't change fast enough to help people make decisions right away.

This dissertation will help to bridge this gap by examining live job postings in LinkedIn through big data and geospatial technologies. The project is dedicated to job positions, job types (remote, work-on-a-site, or mixed jobs), expected level of experience, and region-based distribution of jobs throughout the UK between January and April 2025.

The job data is collected using a custom-built Python scraper and is cleaned, saved on AWS S3, and visualised using Power BI. The end result is the cloud-based interactive dashboard that displays IT job category vis-a-vis region, job position and job type trends. The tool assists the job seeker, employers, as well as the policymakers to make informed decisions (Jaiswal, Ievgeniia Kuzminykh and Sanjay Modgil, 2024). The dashboard indicates trends, including the most popular cities to work, job locations that awaken the most interest, and an increase in remote work. The significance of this project is the fact that it gives a dynamic picture of the labour market of IT. With this research, it becomes possible to keep track of employment patterns, see them, and anticipate them, supporting better planning among individuals, companies, and policymakers.

1.1 Background

New trends in the UK IT employment sector happen due to improvements in technology, an increase in remote work, and changes in the economy. A detailed analysis of employment pattern changes between different geographical locations is currently difficult to achieve thoroughly. This investigation uses big data analytics together with geospatial information systems (GIS) to study employment patterns of the UK IT sector.

This research combines job listing datasets with reports about industries and census statistics and LinkedIn career information to determine regional employment patterns alongside required skills and salary fluctuations throughout regions, then interactive geospatial employment pattern mapping will be created (R. Bali, 2024).

Policymakers and job seekers and businesses will receive support through the analytical framework which reveals workforce distribution and employment opportunity projections. The project satisfies MSc academic needs through the combination of real-world data-driven elements which incorporate data science techniques and geospatial analysis (Bibri and Krogstie, 2020).

The rapid evolution of the IT sector, combined with technological advancements and the rise of remote work, has created significant shifts in employment patterns. Previous research has highlighted the potential of big data and geospatial methods in urban and labor market analysis. However, current studies often lack a comprehensive approach that integrates diverse datasets and modern more advanced techniques to predict future trends in a unified framework.

1.2 Problem Statement

Although there is a lot of information about UK IT jobs online, there isn't a single system that can track or foresee employment trends in real time. People looking for work do not have a clear understanding of what jobs are in demand, where they are most popular, and what is needed for those positions (Liu et al., 2024). Employers and policy makers struggle to keep up with new developments in the job market, mainly because of more hybrid and remote work.

At this point, most research is done using old methods and does not take advantage of geospatial mapping or prediction tools. The difficulty comes from handling different and varied job data to find important trends. The study fills the gap by developing a platform that examines job ads, visualizes job trends, and predicts future changes using advanced techniques in analytics.

1.3 Significance of the Study

This research comes up with a smart way to examine UK IT job trends using big data and geospatial analysis. The information helps people in workforce development find skills and positions, employers determine trends in hiring, and policymakers use data to develop digital workforce strategies (Bone, Ehlinger and Stephany, 2023).

Unlike a regular report, the project relies on current LinkedIn data, uses geography tools, and data visualization to highlight the most promising jobs, increasing remote work, and what's expected in the future (Maghsoudi, 2023). Using the Power BI dashboard, it is possible to interact with these trends. Thanks to using advanced analytics and cloud tools, the research provides useful insights and a model that can be used for different labor markets.

1.4 Aim, Objectives and Research Questions

1.4.1 Aim

The aim of this study is to build an automated, cloud-based dashboard that uses real-time LinkedIn data to analyze and visualize UK IT job trends by role, location, and work type.

1.4.2 Objective

The goal of this dissertation is to construct a cloud dashboard that helps analyse and predict trends in the UK IT sector's job market. The study uses LinkedIn job ads to look at software engineers, data analysts, cybersecurity experts, and cloud professionals. The purpose is to change raw job information into graphics that reveal which regions, jobs, and skills are most in demand.

The system combines web scraping, geospatial mapping, and data visualisation to provide meaningful insights for job seekers, employers, and policymakers.

1.4.3 Research Questions

This research is guided by the following questions:

What are the current trends in the UK IT job market for major roles?

How can LinkedIn job data be visualised to reveal insights by location, role, and work type?

Are there noticeable regional differences in IT job postings?

What benefits can a live dashboard provide to users making job or hiring decisions?

These questions guide the research process from data collection to analysis and application.

1.5 Research Methodology

In this project, the job ads in the LinkedIn UK are extracted with the help of a Python web scraper. The scraper collects latest job offers, data to be included therein: job title, company name, location, the type of work, that is, remote, hybrid, or on-site, the level of experience required, and the day when the job post was presented. It is aimed at the acquisition of the trends in job markets. Python is used to clean and process the data that are collected. This involves eliminating records with duplicate entries, correcting inconsistent formats and addressing missing values. Status data make sure that geographical data is narrowed down so that regional analysis becomes achievable. When it is cleaned, data is stored in a cloud storage facility of AWS S3; and it provides scalable and secure access to data.

Simple statistical indicators such as counts and percentages are used to examine the trends in the type of work, places of work and the type of work. Such maps can be made using geospatial technology like GeoPandas and Folium to explain the distribution of jobs in the UK. Finally, this data is connected to Power BI where an interactive dashboard displays trends in jobs in the form of charts and graphs as well as maps. This approach will guarantee that the whole process, inclusively data gathering up to visualisation, is organised, trustworthy and enables real-time information to end scholars including job seekers, employers and administrators.

1.6 Expected Outcomes

The expected outcomes of this research are:

A fully functional automated pipeline that scrapes LinkedIn IT job listings, processes the data, and stores it in AWS S3.

An interactive Power BI dashboard that visualizes job trends by role, location, work type, and experience level.

Data analysis pipeline consisting of web scraping, cloud storage, data cleaning, and visualisation. The process can be reapplied in other industries like healthcare, finance, education, or retail in order to monitor their job market or trends.

Practical data and advice to job seekers, employers, and policymakers, including where the employment is rising, which skills are the most demanded, and how remote and hybrid employment is expanding throughout the UK will be gained.

1.8 Dissertation Structure

Chapter	Title	Description
Chapter 1	**Introduction**	Introduces the research topic, background, problem statement, objectives, significance, and the overall structure of the dissertation.
Chapter 2	**Literature Review**	Reviews existing research on employment trend analysis, geospatial analytics, and big data techniques, relevant to the IT job market.

Chapter 3	**Methodology**	Explains the data collection process, preprocessing steps, tools and technologies used (e.g., LinkedIn scraping, AWS S3, Power BI), geospatial analysis.
Chapter 4	**Results and Discussion**	Presents key findings from the analysis and visualization outputs. Discusses insights on job roles, geographic distribution, remote work trends, and future demand.
Chapter 5	**Conclusion and Future Work**	Summarizes the project outcomes, highlights limitations, and proposes directions for future research or improvement.

Chapter 2: Literature Review

2.1 Introduction

This literature review explores the theoretical and technical foundations relevant to real-time employment data analytics in the UK IT sector, with a focus on spatial analysis. The chapter draws from previous research on big data infrastructure, spatial data systems, geodemographics, and sentiment-based labor trend forecasting, in employment market analysis. The aim is to assess how past approaches influence the current study and evaluate their relevance in terms of where and how data is collected, which models are selected, and how ethical issues are considered.

2.2 Web Scraping and Cloud-Enabled Scalability for Real-Time Data Extraction

Web scraping refers to a mechanical method of extracting information on websites, which are broadly applied in various aspects such as Business Intelligence, data science, and cybersecurity. It aids in transforming non-structured data with HTML format when machine friendly ones such as JSON or XML cannot be used. The authors (Khder, 2021) says, we can use web scraping to get real-time data, whether it be facially monitoring retail prices or intelligence gathering in darknet markets to assist law enforcement. It is more accurate, consistent and efficient than manual collection of data. Scraping will need such technologies as spidering and pattern matching, and Python is regarded as one of the most comfortable languages in this case due to such tools as BeautifulSoup and Scrapy. Web scraping is closely related to big data and artificial intelligence that enhances decision-making abilities. However, ethical and legal concerns, data privacy, and adherence to the policy of websites also emerge in the author and should be addressed since this practice is becoming increasingly popular.

Web scraping has become an essential process that facilitates the induction of viable information among the vast volumes of unorganized data generated on the web sites. A geometric increase in internet users, as cited in the said paper has driven up the demands on scalable scraping technology capable of processing a large amount of data. The author (Paras et al., 2024) presents significant challenges of mass scraping, such as CAPTCHA problem, space restrictions, computational loads and the feasibility of the data integrity. In its turn, an elastic solution grounded on a cloud architecture with Amazon Web Services (AWS) as a cloud provider, specifically EC2 and DynamoDB will be proposed, as it will enable an efficient resources management, and scaling. Selenium excels because of its browser automation functionality and it is useful when dealing with sites where dynamic interactions have to be carried out. The performance and scalability are assessed in the study and this places the solution as more efficient than the existing alternatives. The combination of both web scraping and cloud infrastructure is indicative of an expanded phenomenon in the data-driven economy in which reliable, scalable, and realistic extraction methods are required.

Recent studies have also emphasized the new trend of cloud-based platforms as critical in handling increasing amounts of information and inverting real-time intelligence. (Dutta, 2019) addresses the benefits of the cloud data systems in assisting business reporting and analytics, focusing on how cloud systems such as AWS could easily manage the volume of data when integrating with visualisation tools, such as Microsoft Power BI. These principles, which were first used in an enterprise, also apply to the area of public data, i.e., in studying the job market. The paper by (Remala and Raju 2023) also discusses the flexibility of cloud computing to work with all kinds of data, emphasizing the significance of automatization, scalability, and cost-efficiency of contemporary data workflows. Their focus on selecting the appropriate infrastructure to handle particular loads justifies the application of service such as the AWS S3 to store and the AWS Lambda to automate as in the case of this study. When used together with Power BI, this cloud-based infrastructure would facilitate the automation of the collection, processing and visualisation of current IT job statistics in LinkedIn, providing the reader with current labour market intelligence.

2.3 Power BI for Visualization & Driving Data-Informed Decisions

Power BI has become a prime data visualisation software with which raw data can be converted into interactive and friendly dashboards. (Mifta, 2022) has also illustrated how the Power BI can be used when visualising some graduation trends during COVID-19, i.e. how one can convert Excel-related data into the lightweight CSV files and, as a result, visualise it to the environment where such data can be interpreted by institutional decision-makers. In the same way, (Deshmukh et al., 2023) created such a platform with Power BI to follow and show the indicators of teacher performance on a platform that provides the stakeholders of educational institutions, will demonstrate the value of real-time information delivery. In their study, they found that the Power BI has an easy-to-use interface and interactive visuals that greatly enhance accessibility, interpretation, and decision-making is far better than manual, manual-based reporting. The results justify the Power BI application in any area where there are non-technical users who require prompt analysis of a complicated data. Power BI is employed for the formulation of an interactive dashboard in the present study that shows the trends of the UK IT jobs based on the data scraped by LinkedIn by assisting users to analyse job distributions by role, location, work type, experience level.

(Park et al., 2021) emphasize that proper data visualisation may indeed enhance decision-making, as less mental effort will be needed to understand its complicated content. In their review, they discovered that visual aids had a positive effect on understanding, perception and attitude, particularly when the information were presented in a manner that accrued credibility and relevance to what was happening in the viewer. They did not allocate their services to labour markets insights, though the findings can be used to understand the labour market, where the users, which include job seekers, employers, and policymakers, find it significant to have simplified visualised results of dynamic job trends. In this project, LinkedIn job data are presented in an easily accessible form using Power BI and will allow users to take an insight into current job roles in the IT field, regional opportunities, and hiring trends, without performing a manual analysis of raw data. This method contributes to more and quicker informed choice in accordance with Park et al. finding that data visualisation can be improved when non-technical individuals facilitate the usability of a complicated dataset.

2.4 Urban Planning and Employment Trend Forecasting

As digital interactions increasingly shape social and economic behaviours, researchers are leveraging web-based data to explore geographic patterns and processes. Web scraping has emerged as a valuable technique for online data acquisition, enabling the collection of near-real-time, geolocated information in a cost-effective manner. This method is particularly useful for studying dynamic geographic phenomena such as rental markets, gentrification, entrepreneurial ecosystems, and spatial planning. Despite its utility, web scraping presents distinct challenges. Legally and ethically, concerns arise around intellectual property rights, informed consent, geo-privacy, and adherence to website terms of service. These concerns also intersect with broader open science practices. Methodologically, technical and statistical issues include data incompleteness, inconsistency, bias, and limited historical depth(Brenning and Henn, 2023). Additionally, geospatial analysis demands accurate automated extraction and geolocation of place names or addresses through techniques such as geoparsing and geocoding. A case study on Leipzig's apartment rental market demonstrates both the potential and limitations of this approach. Overall, web scraping offers significant promise for geographic research, provided its ethical and technical challenges are rigorously addressed.

The pandemic and lockdown policies have substantially shocked labour markets, which explains the importance of near real-time data to track swift developments. This is the subject of one study, which topics and geo-inferences topic modelling of job vacancy data published reed.co.uk to study sectoral and geographical variation. The analysis shows that the number of job ads decreased dramatically at the beginning of the lockdown by 60-70 percent in the first weeks and recovered only partially by the end of the year, creating a cumulative shortage of job ads of more than 40 percent. The effects were sector specific, where hospitality and graduate opportunities significantly dropped, and care work and nursing vacancies rose. There were also shifts in the salary distributions and full-time permanent positions. The paper (Arthur, 2021) also presents Jobtrender.com, an online job data query system, showing how a duplicate approach can be used to monitor the labour market in real-time in times of crisis.

This paper confirms the importance of transport connectivity in relation to increasing access to jobs and addressing spatial equity issues. Based on the Columbus, Ohio metropolitan area, the study uses GIS to compute job accessibility by walking, public transport, and car, and employs transport-based spatial autoregressive (SAR) models to account for spatial autocorrelation (SA) of block groups. Such models include built-environment and socio-economic variables to demonstrate the spatial clustering of job accessibility and its variations by mode of transport. The authors of the study conclude that local physical factors have a strong influence on walking-accessed employment opportunities, whereas public transport accessibility is greater in neighbourhoods with more households without a car (Wang and Chen, 2015). It is observed that there is a spatial mismatch between jobs accessible by public transport and clusters of the Asian population, possibly due to car-oriented patterns of suburban development. Additionally, regions with more single-parent families are at a significant disadvantage regarding general employment opportunities. The findings imply that the integration of land-use and transport planning, car ownership schemes, and social support measures are necessary to address job accessibility and transport-related inequalities.

This paper will illustrate the usefulness of georeferenced data in labour market studies based on a literature review and some real-life examples. The authors describe the process of geocoding administrative data on employment, which includes wage and socioeconomic data on almost the whole German workforce between 2000 and 2017. The data were summarised into 1-square-kilometre grid cells to increase the accessibility and analytical accuracy, encompassing a large variety of labour market and sociodemographic indicators. Such a detailed spatial resolution makes it possible to analyse intra-urban labour market processes in depth in all German cities above 100,000 inhabitants. The study by (Ostermann et al., 2022) emphasises the potential of georeferenced data in revealing localised labour market inequalities, which will be of use to spatially aware economic policy and planning.

The paper by (Netrdová and Nosek, 2020) analyses spatial unemployment differentiation in Czechia between 2002 and 2019 based on very disaggregated spatial and temporal data (municipalities). The main objective was to investigate the spatial aspect and trend of unemployment by using the global and local spatial autocorrelation. The main methodological impact of the study is that it utilizes spatial weight matrices depending on the travel accessibility in real time using the Czech road network, and compares it with the conventional distance-based spatial weights. In spite of the divergence in methodology, the two methods produced comparable spatial unemployment patterns (Netrdová and Nosek, 2020). The results show that the spatial pattern of unemployment has remained rather constant over time with merely a slow change in differentiation between larger regional (macro) and smaller (micro) scales.

In recent years, more businesses in IT, finance, healthcare, retail and logistics have depended on technical graduates for big data analytics. Many organizations are turning to big data and relying on data science, machine learning and AI experts to improve how they work and make decisions. Still, there are difficulties such as not enough qualified workers and a need for graduates to learn soft skills along with their professional skills. Many studies highlight that healthcare informatics; logistics optimization and data governance are new fields emerging because of big data's influence on different sectors. Author (Admas Abtew and Assefa, 2023) states that due to the labor market is changing, schools and policies should be updated to prepare students for the skills they will need. Overall, studies point out that technical graduates must continue to adapt and innovate to find success in big data analytics and help their organizations grow and develop the economy.

Often, employers require people to have a mix of skills to complete their tasks well, as seen by the similarities in skills listed in job postings. A recent study constructs a network of skills using 65 million job ads from the UK between 2016 and 2022 and uses a graph-based technique to identify groups of related skills. They show a network structure, where some skills are related to many others, yet all skills are still linked together. Because of their industries, each region has job opportunities that fit its workforce. As time goes on, businesses are looking for employees with various skills from different areas which suggests that jobs are changing (Liu et al., 2024). It is important to note that data-based skill groupings differ from those by experts, hinting that experts might miss some important connections between skills in real life. It emphasizes that huge amounts of data can help measure current skills needed in the labor market and guide plans for developing the workforce.

2.3 Conclusion

The chapter looked at important studies and instruments for analyzing current job data in the UK IT industry. It demonstrated that analyzing data, using various techniques and working with large sets of information can help understand what is happening in the job market. These dataset and tools will be very helpful to delivers useful advice for the job seekers and employers for the next part of the process.

Chapter 3: Methodology

3.1 Introduction

This chapter summarises the approach to use in exploring recent trends in IT employment in United Kingdom based on LinkedIn data. It uses web scraping, cloud-based storage and automation, and visualisation tools to convert raw job listing to structured insights. Individual stages, such as data collection and cleaning, data storage, and dashboard development, were to facilitate the project goals of uncovering demand of IT roles and job type and work level distribution, as well as geographical employment patterns. The approach provides a systematic methodology that requires the scraper to collect real-time data, quality is maximized through preprocessing routines, the data is stored securely with the AWS S3 service, and the interactive dashboards are provided through Power BI analysis.

3.2 Data Collection

A Python web scraper was the first step in the project to gather IT job listings from LinkedIn UK in real time. This class was made to collect information automatically from LinkedIn (Tzimas et al., 2024). The scraper had been programmed to find predefined categories of IT jobs including Cyber Security, Data Analyst, Software engineer, Devops engineer, Cloud engineer and machine learning engineer. In both positions, the scraper found information such as:

- Job title
- Company name
- Location (later changed to coordinates based in cities)
- Type of work (Remote, hybrid, on-site)
- Proficiency, entry, mid, senior.
- Date posted and date scraped
- Job description and job link

In order to update information on a timely basis, it was possible to gather only the postings made within the 3 days and to remove old postings. The obtained data were converted into records and auto-uploaded to AWS S3 where they are centrally and securely stored as CSV files with timestamps. It did this so that the dataset was scalable and could be processed and analyzed later.

Feature	Description
Number of Records	18,101 job listings
Number of Attributes	11 distinct attributes
Unique Job Titles	Over 800 unique job roles

3.3 Dataset Review

Real-time IT job listings from LinkedIn, focusing on jobs in the UK was scrapped initially. The set of data collected included 2,669 vacancies and 11 different qualities, comprising more than 800 different job titles. In both records, a blend of categorical (e.g., work type, experience level) and textual (e.g., job description), as well as geospatial (latitude/longitude based on city names) attributes was included (Liu et al., 2024). The characteristics allowed the dataset to be analyzed in multi-dimensional form, providing an understanding of role demand, work flexibility, and geographic distribution.

Despite the abundance of data in LinkedIn, inconsistencies did occur, including differences in names of locations, the absence of data-fields, and duplications. Before data analysis, data cleaning and standardisation steps were performed to ascertain reliability. This was through standardisation of city names (e.g. a post such as London, England, United Kingdom - London), missing value, and duplication of the same posting (Kazi et al., 2022). Consequently, the data was set up to be properly visualised and combined with Power BI charts.

3.4 Resources

Cloud Infrastructure:

- Using AWS S3 for secure, scalable storage of scraped LinkedIn job datasets in CSV format. It provides a safe and scalable environment, where one can easily access to analysis and visualisation.
- AWS Lambda is used to automate how the data is collected and uploaded in a serverless environment. When new job data gets scraped, a Lambda functions comes in to automatically transfer the file to the appropriate folder in s3. This eliminates the manual uploads and commitment of servers. The use of lambda was selected due to the fact that it is applicable to serverless automation, low cost and it can be scaled as required.

This cloud solution also guarantees efficiency, reliability, cost-effectiveness of the data pipeline and it allows real-time updates to the Power BI dashboard.

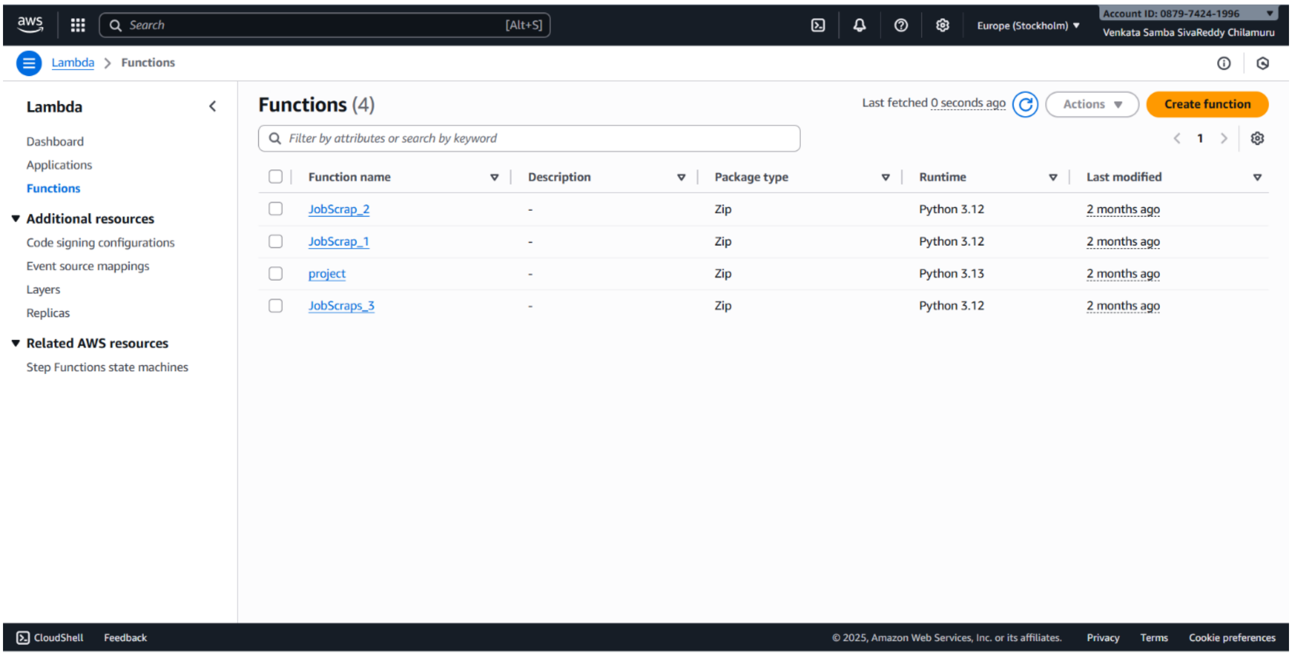


Figure 1: Lambda

As illustrated in Figure 1, data collection and uploading process is automated with the help of AWS Lambda in the project. To update the data in real-time, Lambda functions were created which would be scheduled to execute every day, to scrap the data in job, process the job data, and store it in AWS S3 to facilitate a smooth and serverless workflow lining the flow of data.

Data Collection and Processing:

- Python serves as the core language for the scraping pipeline due to its extensive libraries and ease of use. This is deployed in this because it is easy to implement, versatile and has good library backing.
- BeautifulSoup and Requests are used to scrape LinkedIn job listings. These libraries are efficient for navigating and extracting content from HTML pages, which is required to access job titles, companies, locations, and descriptions from publicly available LinkedIn job cards.
- Regular Expressions (re) are used to clean and standardise job location fields (e.g., city names) and to extract key patterns from job descriptions, such as identifying experience levels (e.g., “Senior Level”, “Entry Level”) and work types (e.g., remote, hybrid). This improves data quality and ensures consistent analysis.

- Built-in Python modules such as csv, datetime, and io.StringIO are used for formatting the data and creating in-memory CSV files before upload. These libraries make it easier to manage file creation, handle timestamps, and avoid saving temporary files locally, which aligns well with a cloud-first setup.

Data Storage and Management:

- The Boto3 library, which is AWS's official SDK for Python, is used to connect the script with AWS S3. It allows the job data to be uploaded programmatically and ensures that files are saved with timestamped filenames for easy version control. Boto3 was selected because it provides secure, direct, and reliable integration between Python and AWS, enabling the entire process to run automatically without manual file handling.

Visualization and Reporting:

- Power BI is used as the main visualisation tool because it supports direct integration with AWS S3 and offers powerful interactive visualisation features. It allows the dataset to be updated daily without the need to reimport or reprocess data manually. Power BI was chosen for its ability to create dashboards that display job distributions by role, location, company, work type, and experience level through a combination of bar charts, pie charts, and maps. This helps users understand job trends and make data-driven decisions.

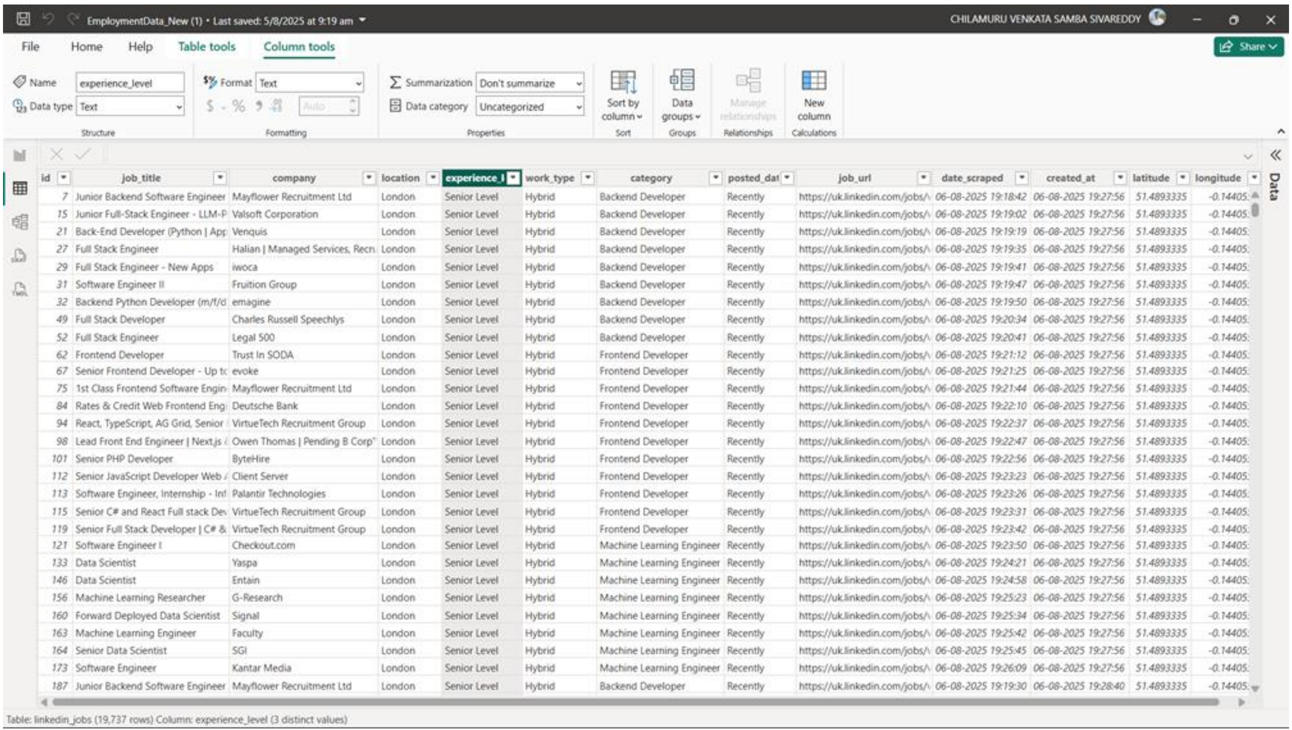


Figure 2: Dataset Overview

3.5 Data Pre-Processing

The retrieved raw job data on LinkedIn necessitates great cleaning and transformation to be analyzed. The steps that are used are as follows:

Location Standardization: Cities and regions can be spelled and standardized by breaking off suffixes such as "Greater", "Area" or "United Kingdom" to make geographic mapping consistent.

Job Description Cleaning: HTML tags and redundant whitespace are removed from job descriptions to improve readability (Tzimas et al., 2024).

Experience Level Detection: Rules of keywords matching that assign each job to their levels: Entry-Level, Mid-Level, and Senior-Level according to a description text (Ex: entry-“graduate”; senior-“lead”).

Work Type Categorization: On-site, remote, and hybrid jobs are labeled using search-based phrases such as: to be a remote worker, or work at home, or work remotely, or hybrid work, and so forth.

Date Normalization: The date in which the post was made shall be formatted in the format of a standard datetime (YYYY-MM-DD HH:MM:SS) so that the analysis is possible over time.

Deduplication and Null Handling: Duplicates are eliminated and any records that have critical fields that are missing in the listings are deleted to guarantee quality of data (Mohammad et al., 2023).

It will be structured into a processed dataset that can be used in Power BI dashboards as well. Such characteristics as city, job role, date of posting, and experience level are major inputs that are used in clustering and forecasting models.

3.6 Cloud Storage

Once cleaned, the dataset was written to an in-memory CSV file and uploaded to Amazon Web Services (AWS) Simple Storage Service (S3) using the boto3 SDK since the scrapped dataset count is high. The dataset (Approximately 2,600 job listings gathered throughout a 4-month period) can technically fit in the local storage because it is not that large but to be able to automate the process and make it more efficient, cloud storage via AWS S3 has been elected. The script to scrape is set to run once a day, at midnight, scraping fresh LinkedIn employment information. The dataset would have to be stored locally, which means it will need to be managed manually and require script execution, which is more prone to delays and mistakes. With the combination of AWS Lambda and S3, the whole process is automated and no manual operation is required: data is scraped, cleaned and automatically written to S3 every night at 12AM. Timestamps allow each file to be saved with a date and time in the name, and this will enable more convenient monitoring of daily changes, comparison of trends over time, and data version control (Ahmed Zaki et al., 2022). AWS S3 can also be accessed securely, expands and is easily integrated with Power BI, the connection enables the user to access the cloud and preview live job patterns. To conclude, local storage could be used, but with the help of AWS, it is possible to achieve stable daily automation, the ability to scale in the future and elimination of the manual process.

3.7 Data Visualization

The processed and organised data was linked to power BI to develop dynamic and interactive IT job trends in UK (Li et al., 2022). Power BI has been chosen because it can be easily integrated into AWS S3, showcased in real-time mode and offers advanced dashboarding options. It was aimed at turning some rather unstructured jobs postings in LinkedIn into a series of viable and understandable images to facilitate decision-making by various stakeholders.

3.7.1 Dashboard Design

The dashboard aimed to provide insights in many aspects, including:

Experience Levels - aren't generalised, but demonstrate the share of entry, mid, and senior positions.

Work Type Distribution - Distribution of remote, hybrid and on-site postings.

Job Roles - bar charts which show the most sought IT jobs.

Employers - comparative graphs with information on the companies that are the most active in hire and which type of work arrangement they prefer.

Geospatial Mapping - a mapping tool that uses longitude and latitude to determine locations where employment is likely to occur in UK.

3.7.2 Interactivity and Filtering

The dashboard was designed to help improve usability by adding dynamic date range, job category, experience level, type of work, and location filters. The implementation of a filter automatically updated all charts and maps so using it the user was able to drill down into particular scenarios (Example: entry-level positions in London, or hybrid jobs in Manchester). The nature of interactivity will guarantee the ability of the dashboard to meet the varied needs of job seekers, recruiters and policymakers.

3.7.3 Statistical Summaries

Along with visual exploration, basic statistical summaries, including frequency counts, proportions, and cross-tabulations, were also supported by Power BI. E.g. it determined the proportion of postings by work type or the number of listings per company. These overviews constituted the basis on the graphics.

3.7.4 Geospatial Tools

The visualisation strategy required the aspect of geographic distribution. Based on the latitude and longitude numbers received in the preprocessing step, Power BI created map plots showing job concentrations by city. The size of the bubbles documented the size of postings, and it was easy to pick out large IT hubs like London, Manchester, Birmingham, and Leeds.

3.8 Legal, Social and Ethical Issues

3.8.1 Legal Issues

The aim of this project is to collect job postings available on LinkedIn's public pages. Generally, LinkedIn's rules do not allow automated data collection without permission, but what is done here is only for educational research and not for profit. No developers are allowed to use LinkedIn's official APIs, as they are managed by their own set of agreements. It is made to respect the policies set in robots.txt and controls the rate of requests to prevent the server from getting overloaded or from violating fair use rules. Just non-personal details are collected, including job title, the name of the company, job description, location, and the date the job was posted. Nothing about a user's personal details is collected or retained. AWS S3 is used to store data securely and uses the guidelines from the UK Data Protection Act (DPA) and the General Data Protection Regulation (GDPR) (Qadir et al., 2016).

3.8.2 Social Issues

Even though the project looks at labour market trends, it does not interact with individuals. The knowledge gained is meant to support job seekers, researchers, and policymakers in their general planning and analysis of the workforce. Information in the project is shared with everyone in the same way, and it does not cause bias, discrimination, or surveillance. It tries to give everyone a fair and unbiased way to use data for better job decisions.

3.8.3 Ethical Issues

There are no people involved in the study, no interviews are carried out, and no personal information is collected, which means no ethical approval is needed. Still, the data is used in an ethical manner to ensure it is safe. The data obtained is made up of public job postings and does not involve trying to access anyone's private data. The project makes sure that the data is only used for research, protected properly, and will not lead to harm or exploitation of people. Also, the data is processed so that it does not reveal any indirect identifiers, mainly by hiding general geographic data. Overall, the study keeps academic standards and values the privacy of individuals and organizations online.

3.9 Project plan

Task	**Duration**
1. Requirement Analysis & Objective Setting	1 week
2. Literature Review	8 weeks
3. Dataset Design & Schema Planning	2 weeks
4. LinkedIn Scraping Script Development	3 weeks
5. Data Cleaning & Preprocessing	3 weeks
6. AWS S3 Setup & Data Storage Integration	2 weeks
7. Automation via AWS Lambda (optional)	2 weeks
8. Exploratory Data Analysis (EDA)	2 weeks
9. Visualization with Power BI	3 weeks
10. Drafting Methodology & Results Chapters	4 weeks
11. Evaluation, Testing & Dashboard Demonstration	6 weeks
12. Final Documentation & Proofreading	2 weeks

3.10 Conclusion

This chapter explained the ways and tools used to create a real-time system for processing UK IT job data. It is started by using a Python scraper on LinkedIn and after that, the data was sorted out. Data security and management were carried out in AWS S3, and then the data was analyzed using

Power BI. Besides, the chapter made clear that it is important to respect legal, social, and ethical rules when working with data. All in all, the process depends on proper ways to review shifts in jobs and decisions based on data.

Chapter 4: Results and Discussion

4.1 Introduction

This chapter explains the complete procedure of obtaining and studying real-time IT job data from LinkedIn UK with a Python web scraper. It consists of cleaning up data, putting the data in AWS S3, and scheduling the process with AWS Lambda. The information collected is turned into graphs and charts that show which job types are in demand, where employers are looking, the experience needed, and the main areas for employment, and all of this information is accessible through a dashboard.

4.2 Implementation

4.2.1 Data Collection

This study used a Python web scraper created especially to collect job listings from LinkedIn UK in real time. The script was built to help gather data on IT jobs such as “Cyber Security”, “Software Engineer”, “Data Analyst”, “Cloud Engineer”, and many others. The aim was to find the most up-to-date job openings to observe the trends in employment.

The process started by setting up a list of job roles in advance. For each role, the script accessed LinkedIn job search pages using HTTP requests. The HTML content was then parsed using the BeautifulSoup library to extract job-specific information such as job title, company name, location, posting date, job description, and job link.

To ensure relevance, the script filtered out old postings and retained only those published within the last three days. It also derived insights like experience level (entry, mid, senior) and work type (remote, hybrid, on-site) from the job descriptions using keyword-based logic. Cleaned job location data was simplified to city-level for accurate mapping.

Once all jobs were collected and structured into a list, the data was written to an in-memory CSV file and uploaded to Amazon S3 using the Boto3 library. This created a centralized, secure, and scalable repository for further processing and visualization.

```python
import json
import boto3
import requests
from bs4 import BeautifulSoup
import csv
from datetime import datetime
import time
import re
from io import StringIO
import os
import psycopg2
from psycopg2.extras import execute_batch
from time import sleep
from typing import Tuple, Optional
```

Figure 3: Importing Libraries

The code begins by importing essential Python libraries as shown in Figure 3 that make the web scraping and cloud storage functionality possible. json is used for working with JSON data, boto3 is the AWS SDK for Python and allows the code to interact with Amazon S3 for file storage, and requests lets the scraper send HTTP requests to web pages. BeautifulSoup from the bs4 library helps parse the HTML content retrieved from LinkedIn job listings. The csv module is used to write scraped data into CSV format. Modules like datetime, time, and re help with date-time processing, pausing between requests, and cleaning text data using regular expressions. StringIO allows creating an in-memory file object (instead of writing a file on disk), and os is used to access environment variables such as the name of the S3 bucket. Also, psycopg2 (along with execute_batch) is brought in to store job information to a PostgreSQL database, the typing library adds functionality to describe structured data types such as tuples and optional values.

4.2.2 Pre-processing

Data preprocessing was a crucial step in transforming raw job listings into clean, consistent, and analysis-ready data (Tzimas et al., 2024). The data scraped from LinkedIn included diverse formats and text inconsistencies that needed to be addressed before further analysis and visualisation. The data pre-processing process starts by cleaning and then the location standardization which is mentioned below in detailed manner.

Three variations of the code represented in Figure 4 define a single class called LinkedInRecentITJobsScraper. Within this class, the programme initially establishes headers that causes it to behave similarly to a standard web browser when the web pages are opened. This facilitates its access of job information without being blocked. Each code then has a list of job roles, although in each case the list of roles differs. Code 1 specialises in data and security related professions like Cyber Security, Data Analyst and Data Scientist. Code 2 concentrates on developer and artificial intelligence roles or positions including Backend Developer, Frontend developer, and Machine Learning Engineer. Code 3 embraces jobs relating to engineering and systems, including Software Engineer, DevOps Engineer, Cloud Engineer, and Full Stack Developer.

Database details (such as name, user, password, host, and port) are also established in the code to enable the programme to connect to and store job information in a database. And, lastly, is the small memory (cache) to hold the location data temporarily, to avoid the programme doing the same work multiple times.

```python
class LinkedInRecentITJobsScraper:
    def __init__(self):
        self.headers = {
            'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36',
            'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
            'Accept-Language': 'en-US,en;q=0.5',
            'Connection': 'keep-alive',
        }

        self.it_jobs = [
            "Cyber Security",
            "Data Analyst",
            "Data Scientist"
        ]

        # Add database configuration
        self.db_config = {
            'dbname': os.environ['DB_NAME'],
            'user': os.environ['DB_USER'],
            'password': os.environ['DB_PASSWORD'],
            'host': os.environ['DB_HOST'],
            'port': os.environ['DB_PORT']
        }

        self.geocoding_cache = {}  # Simple in-memory cache
```

```python
class LinkedInRecentITJobsScraper:
    def __init__(self):
        self.headers = {
            'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36',
            'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
            'Accept-Language': 'en-US,en;q=0.5',
            'Connection': 'keep-alive',
        }

        self.it_jobs = [
            "Backend Developer",
            "Frontend Developer",
            "Machine Learning Engineer"
        ]

        # Add database configuration
        self.db_config = {
            'dbname': os.environ['DB_NAME'],
            'user': os.environ['DB_USER'],
            'password': os.environ['DB_PASSWORD'],
            'host': os.environ['DB_HOST'],
            'port': os.environ['DB_PORT']
        }

        self.geocoding_cache = {}  # Simple in-memory cache
```

```python
class LinkedInRecentITJobsScraper:
    def __init__(self):
        self.headers = {
            'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36',
            'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
            'Accept-Language': 'en-US,en;q=0.5',
            'Connection': 'keep-alive',
        }

        self.it_jobs = [
            "Software Engineer",
            "DevOps Engineer",
            "Cloud Engineer",
            "Full Stack Developer"
        ]

        # Add database configuration
        self.db_config = {
            'dbname': os.environ['DB_NAME'],
            'user': os.environ['DB_USER'],
            'password': os.environ['DB_PASSWORD'],
            'host': os.environ['DB_HOST'],
            'port': os.environ['DB_PORT']
        }

        self.geocoding_cache = {}  # Simple in-memory cache
```

Figure 4: Initialization of LinkedInRecentITJobsScraper Class

After that three valuable functions are added in the code in Figure 5 to aid in cleaning and categorising the job details. The first function, cleanlocation, would be utilised to shorten and clear the job location. Frequently, a job advertisement contains lengthy text e.g. London, England, United Kingdom or Manchester Area, England United Kingdom. This option eliminates additional terms such as ", United

Kingdom" or Greater and leaves the name of the city. Take the case of London, England, United Kingdom, being reduced to London.

The second, determineexperiencelevel, examines the words included in the job description and arrives at the level of the position. When there is mentioning of such words as a senior, a lead, a principal, a manager or head of, they are considered a Senior Level job. When terms such as junior, entry, graduate, or trainee are used, then the job is indicated as Entry Level. When these are not known, the job default becomes Mid Level. The third function isremote that verifies whether the job can be done at home or not. Descriptions including remote, work from home, or wfh means that the job is Remote. When, the word hybrid is discovered, it will show the job as Hybrid (part office, partly remote). When they are all absent, the job is termed as On-site.

```python
def clean_location(self, location):
    """Clean location string to extract only city name"""
    location = re.sub(r', England, United Kingdom$', '', location)
    location = re.sub(r', United Kingdom$', '', location)
    location = re.sub(r', UK$', '', location)
    location = re.sub(r' Area, United Kingdom$', '', location)
    location = re.sub(r' Area$', '', location)
    location = re.sub(r'Greater ', '', location)

    if ',' in location:
        location = location.split(',')[0].strip()

    return location

def determine_experience_level(self, job_description):
    """Determine if the job is entry-level, mid-level, or senior"""
    text = job_description.lower()

    if any(word in text for word in ['senior', 'lead', 'principal', 'manager', 'head of']):
        return 'Senior Level'
    elif any(word in text for word in ['junior', 'entry', 'graduate', 'trainee']):
        return 'Entry Level'
    else:
        return 'Mid Level'

def is_remote(self, job_description):
    """Check if the job is remote"""
    text = job_description.lower()
    if any(word in text for word in ['remote', 'work from home', 'wfh', 'hybrid']):
        if 'hybrid' in text:
            return 'Hybrid'
        return 'Remote'
    return 'On-site'
```

Figure 5: Functions for Cleaning Location, Identifying Experience Level, and Checking Work Mode

Figure 6 shows that the code comes with two functions that aid in determining the level of recentness of a job and the location of the job.

The first of these functions, isrecentjob, queries whether the job posting is recent, i.e. whether it was posted in the past three days or not. It examines the date of the posting text. Unless the text contains the terms hrs ago or date ago, days ago, 2days ago, 3days ago, the job is considered recent and is considered true. When the job is more than three days old, it returns False. This can then narrow down job postings to the latest postings only.

The second one, get coordinates provides the precise latitude and longitude of a job site. First, it verifies whether the location has been cleared; in this case, returns None. Then it searches for a location by looking within a cache memory to determine whether the location coordinates had been previously stored, thus, saving time. Otherwise, it makes a request to the Nominatim service (an OpenStreetMap component) to find a location. In order to increase accuracy of the results, it appends United Kingdom to the search query. Nominatim caches the latitude and longitude and returns them to Nominatim was able to find the place. Otherwise, it would save None to the cache and would not provide coordinates. The role also waits a second between the requests to obey the rules of the service and do not overload the service.

```python
def is_recent_job(self, posted_date_text):
    """Check if the job was posted recently (within last 3 days)"""
    text = posted_date_text.lower()

    if 'hours ago' in text or 'hour ago' in text:
        return True
    elif 'day ago' in text or '1 day ago' in text:
        return True
    elif '2 days ago' in text:
        return True
    elif '3 days ago' in text:
        return True

    return False

def get_coordinates(self, location: str) -> Tuple[Optional[float], Optional[float]]:
    """Get latitude and longitude for a location using Nominatim (OpenStreetMap)"""
    if not location:
        return None, None

    # Check cache first
    if location in self.geocoding_cache:
        return self.geocoding_cache[location]

    try:
        # Add UK to the location query to improve accuracy
        search_location = f"{location}, United Kingdom"

        # Using Nominatim API
        url = "https://nominatim.openstreetmap.org/search"
        headers = {
            'User-Agent': 'LinkedInJobScraper/1.0',  # Required by Nominatim
            'Accept-Language': 'en-US,en;q=0.9',
        }
        params = {
            'q': search_location,
            'format': 'json',
            'limit': 1
        }

        response = requests.get(url, headers=headers, params=params)
        sleep(1)  # Respect rate limit - 1 request per second

        if response.status_code == 200:
            results = response.json()
            if results:
                lat = float(results[0]['lat'])
                lon = float(results[0]['lon'])
                # Cache the result
                self.geocoding_cache[location] = (lat, lon)
                return lat, lon

        # Cache negative result
        self.geocoding_cache[location] = (None, None)
        return None, None

    except Exception as e:
        print(f"Error geocoding location '{location}': {str(e)}")
        return None, None
```

Figure 6: Functions for Checking Recent Job Postings and Mapping Job Locations

The code in Figure 7 defines the scrapelinkedinjobs part of the programme, which is the primary part of the programme. This role gathers the latest IT job advertising through LinkedIn UK in regard to the job descriptions mentioned above.

The following step involves making an empty list named alljobs where all the job information will be stored. The programme then sequentially goes through each job title (such as Data Scientist, Backend Developers, etc.) in turn. Each job title is searched through one page of LinkedIn job results. The URL is formulated as the job title and location (United Kingdom) and the results are asked of LinkedIn. Assuming that the site loads properly, the code will read this page with BeautifulSoup and search for job postings (job cards).

In every job posted, the code verifies the date of posting. When the job is not new (older than 3 days old), it is not taken. When the job is new, the programme gathers helpful information including:

- Job Title (e.g. Software engineer)
- Company Name (job is posted therein)
- Location (cleansed with the previous cleanlocation function)
- URL of the posting (job title)
- Job Description (detailed descriptions of job)
- Experience Level This is determined as part of the determineexperiencelevel function.
- Work Type (Remote, Hybrid or On-site, isremote functionality)
- Posted Date and Date Scraped (when the code gleaned the job).

The jobs are then placed in orderly fashion in the all jobs list. The function also does not waste time browsing through older jobs so that it would cheque two pages in a row with no new jobs, and then come to a halt. Between requests a short break (time.sleep) is put as to not load-down LinkedIn servers too heavily. Lastly, the function gives the list of all matters that have been gathered about the job and can be later saved or analysed.

In general, Figure 7 demonstrates that the actual gathering of the job postings at LinkedIn, followed by the sifting of only the recent ones and clean presentation of the data in terms of company, location, experience, and worktype can definitely happen.

```python
def scrape_linkedin_jobs(self):
    """Scrape recent IT jobs from LinkedIn UK"""
    all_jobs = []
    max_pages_per_category = 1  # Reduced for Lambda execution time constraints

    for job_title in self.it_jobs:
        print(f"\nScraping recent jobs for: {job_title}")
        page = 0
        consecutive_old_jobs = 0

        while page < max_pages_per_category:
            try:
                encoded_title = requests.utils.quote(job_title)
                url = f"https://www.linkedin.com/jobs/search/?keywords={encoded_title}&location=United%20Kingdom&start={page*25}&f_TPR=r86400"
                print(f"Scraping page {page + 1}...")

                response = requests.get(url, headers=self.headers)
                if response.status_code != 200:
                    print(f"Failed to fetch page {page + 1}. Status code: {response.status_code}")
                    break

                soup = BeautifulSoup(response.text, 'html.parser')
                job_cards = soup.find_all('div', class_='base-card')

                if not job_cards:
                    print("No more jobs found on this page")
                    break

                recent_jobs_found = False

                for job in job_cards:
                    try:
                        posted_date = job.find('time', class_='job-search-card__listdate')
                        if posted_date:
                            posted_date_text = posted_date.text.strip()
                            if not self.is_recent_job(posted_date_text):
                                continue

                        recent_jobs_found = True

                        title = job.find('h3', class_='base-search-card__title').text.strip()
                        company = job.find('h4', class_='base-search-card__subtitle').text.strip()
                        raw_location = job.find('span', class_='job-search-card__location').text.strip()
                        location = self.clean_location(raw_location)
                        job_url = job.find('a', class_='base-card__full-link')['href']

                        job_response = requests.get(job_url, headers=self.headers)
                        job_soup = BeautifulSoup(job_response.text, 'html.parser')
                        job_description = job_soup.find('div', class_='show-more-less-html__markup').text.strip() if job_soup.find('div', class_='sh

                        experience_level = self.determine_experience_level(job_description)
                        work_type = self.is_remote(job_description)

                        all_jobs.append({
                            'Job Title': title,
                            'Company': company,
                            'Location': location,
                            'Experience Level': experience_level,
                            'Work Type': work_type,
                            'Category': job_title,
                            'Posted Date': posted_date_text if posted_date else 'Recently',
                            'Job URL': job_url,
                            'Date Scraped': datetime.now().strftime("%Y-%m-%d %H:%M:%S")
                        })

                    except Exception as e:
                        print(f"Error parsing job: {str(e)}")
                        continue

                if not recent_jobs_found:
                    consecutive_old_jobs += 1
                    if consecutive_old_jobs >= 2:
                        break
                else:
                    consecutive_old_jobs = 0

                page += 1
                time.sleep(1)  # Reduced sleep time for Lambda

            except Exception as e:
                print(f"Error fetching page: {str(e)}")
                break

    return all_jobs
```

Figure 7: Function for Scraping Recent IT Jobs from LinkedIn UK

Figure 8 defines the code that actually does the work of saving all the job data collected to Amazon S3 as a CSV file; this is the savetos3 code.

The first line of the function cheques whether there are any jobs to be saved. Whenever there is nothing in the list, it just prints a message and exits. In case there are jobs it initially creates a CSV file in memory with the usage of String IO. It maps the column names of the CSV where Job Title, Company, Location, Latitude, Longitude, Experience Level, Work Type, Category, Posted Date, Job

URL, and Date Scraped are indicated. The code pre-saves latitude and longitude associated to each job using the previously discussed getcoordinates function. It produces a new list of jobs having these coordinates set. To prevent the overloading of geocoding service, a small delay (time.sleep) is inserted between the location queries.

The code then writes jobs data in the CSV memory. Then, it networks to the Amazon S3 by the boto3 library. It uses the bucket name in the environment variables and creates a unique file name with a time stamp such as linkedinrecentitjobs20250906173000.csv. Lastly, it puts the CSV file on S3 in the correct content type (text/csv).

When it is saved successfully, it will print out the number of jobs saved as well as the complete S3 path. When any mistake occurs in this process, it captures the mistake and builds a message. All in all, Figure 6 indicates that the code stores the job data collected safely on the cloud with coordinates used to represent each job, so it is easily analysed or reported afterwards.

```python
def save_to_s3(self, jobs):
    """Save jobs to CSV file in S3"""
    if not jobs:
        print("No jobs to save")
        return

    try:
        # Create CSV in memory
        csv_buffer = StringIO()
        fieldnames = ['Job Title', 'Company', 'Location', 'Latitude', 'Longitude',
                      'Experience Level', 'Work Type', 'Category', 'Posted Date',
                      'Job URL', 'Date Scraped']

        # Add coordinates to jobs data
        jobs_with_coordinates = []
        for job in jobs:
            job_copy = job.copy()
            lat, lng = self.get_coordinates(job['Location'])
            job_copy['Latitude'] = lat
            job_copy['Longitude'] = lng
            jobs_with_coordinates.append(job_copy)
            time.sleep(0.1)  # Rate limiting

        writer = csv.DictWriter(csv_buffer, fieldnames=fieldnames)
        writer.writeheader()
        writer.writerows(jobs_with_coordinates)

        # Upload to S3
        s3 = boto3.client('s3')
        bucket_name = os.environ['S3_BUCKET_NAME']
        timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
        file_name = f'linkedin_recent_it_jobs_{timestamp}.csv'

        s3.put_object(
            Bucket=bucket_name,
            Key=file_name,
            Body=csv_buffer.getvalue(),
            ContentType='text/csv'
        )

        print(f"\nSaved {len(jobs)} jobs to S3: {bucket_name}/{file_name}")
        return f"s3://{bucket_name}/{file_name}"

    except Exception as e:
        print(f"Error saving to S3: {str(e)}")
        return None
```

Figure 8: Function to Save LinkedIn Job Data as CSV in Amazon S3

The code specifies the savetopostgres function in Figure 9 that helps to store all the obtained job information in a PostgreSQL database. The first step is to cheque whether there are any jobs to save. When the list is not full, it writes out a message and terminates. When jobs are availed it asks postgreSQL database with the credentials that are contained in the programme (self.dbconfig).

The code then makes sure the table linkedinjobs exists, and creates it as needed. This table will contain all the job details in the form of columns include Job Title, Company and Location, Latitude, Longitude, Level of experience, Type of work, as well as the category, date posted, job URL and date scraped. To store the coordinates of any job Latitude and Longitude will be added. The Job URL column is further restricted in order to avoid duplications. The code then gets job data ready to be inserted into the table. On every job, it computes the latitude and longitude based on the get_coordinates function and produces data that can be stored by PostgreSQL. The additional small time delay (time.sleep) prevents the API limits.

Lastly, the programme inserts all the jobs into the database using batch insert which can easily insert many rows and finally makes the transaction. In case of any errors it will rollback the transaction and save the database. Once it is completed, the database connexion is terminated. In general, Figure 9 illustrates that the code is used to save structured job data into a PostgreSQL database (including coordinates and all pertinent details), which is readily queryable and analysed subsequently.

```python
def save_to_postgres(self, jobs):
    """Save jobs to PostgreSQL database"""
    if not jobs:
        print("No jobs to save to database")
        return

    conn = None
    cur = None
    try:
        print("Connecting to database...")
        conn = psycopg2.connect(**self.db_config)
        cur = conn.cursor()

        # Modified table creation to include latitude and longitude
        create_table_query = """
        CREATE TABLE IF NOT EXISTS linkedin_jobs (
            id SERIAL PRIMARY KEY,
            job_title VARCHAR(255),
            company VARCHAR(255),
            location VARCHAR(255),
            latitude DECIMAL(10, 8),
            longitude DECIMAL(11, 8),
            experience_level VARCHAR(50),
            work_type VARCHAR(50),
            category VARCHAR(100),
            posted_date VARCHAR(100),
            job_url TEXT,
            date_scraped TIMESTAMP,
            created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
        );
        """
        cur.execute(create_table_query)

        # Add unique constraint if it doesn't exist
        try:
            cur.execute("""
                ALTER TABLE linkedin_jobs
                ADD CONSTRAINT unique_job_url UNIQUE (job_url);
            """)
        except psycopg2.errors.DuplicateTable:
            conn.rollback()

        conn.commit()

        insert_query = """
            INSERT INTO linkedin_jobs
            (job_title, company, location, latitude, longitude, experience_level,
             category, posted_date, job_url, date_scraped)
            VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
            ON CONFLICT (job_url) DO NOTHING
        """

        # Modify job data to include coordinates
        job_data = []
        for job in jobs:
            lat, lng = self.get_coordinates(job['Location'])
            job_data.append((
                job['Job Title'],
                job['Company'],
                job['Location'],
                lat,
                lng,
                job['Experience Level'],
                job['Work Type'],
                job['Category'],
                job['Posted Date'],
                job['Job URL'],
                datetime.strptime(job['Date Scraped'], "%Y-%m-%d %H:%M:%S")
            ))
            # Add a small delay to avoid hitting API rate limits
            time.sleep(0.1)

        execute_batch(cur, insert_query, job_data)
        conn.commit()

        print(f"\nSaved {len(jobs)} jobs to PostgreSQL database")

    except Exception as e:
        print(f"Error saving to PostgreSQL: {str(e)}")
        if conn:
            conn.rollback()
    finally:
        if cur:
            cur.close()
        if conn:
            conn.close()
```

Figure 9: Function to Save LinkedIn Job Data into PostgreSQL Database

In Figure 10, the code corresponds to the definition of the lambdahandler function, which is the starting point to execute the LinkedIn job scraper in AWS Lambda. This coordinates the whole process of scraping, storing and summarising job data. The Lambda function initially makes an instance of the LinkedInRecentITJobsScraper class when the Lambda itself is initiated. It then invokes the scrapelinkedinjobs method to find all the IT job posts published by LinkedIn UK in the recent past.

Once the jobs are gathered it stores the data in two fashions:

S3 Storage: It generates a CSV file containing all the job information (including the coordinates) and uploads it to an Amazon S3-bucket using savetos3 function.

PostgreSQL Database: It has the functionality of storing all information about each job in the PostgreSQL database using saveto postgres function which can be later interrogated and analysed.

The role also develops a summary of the jobs, tallying the total number of jobs gathered and categorising them (such as Data Scientist, Backend Developer, etc.). It constitutes S3 file name and the execution time. Lastly, the Lambda function sends a 200-status-response and the summary in the event that everything is properly brought up. In case of an error anywhere it catches the exception and responds with a status code of 500 containing the error message. In general, Figure 8 demonstrates that the Lambda function manages the overall scraping operation, stores the data in both a database and cloud storage, and gives an overview of the results.

```python
def lambda_handler(event, context):
    try:
        print("Starting LinkedIn job scraper...")
        scraper = LinkedInRecentITJobsScraper()
        jobs = scraper.scrape_linkedin_jobs()

        # Save to S3
        s3_path = scraper.save_to_s3(jobs)

        # Save to PostgreSQL
        scraper.save_to_postgres(jobs)

        # Prepare summary
        categories = {}
        for job in jobs:
            category = job['Category']
            if category not in categories:
                categories[category] = 0
            categories[category] += 1

        summary = {
            'total_jobs': len(jobs),
            'jobs_by_category': categories,
            'output_file': s3_path,
            'timestamp': datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        }

        return {
            'statusCode': 200,
            'body': json.dumps(summary)
        }

    except Exception as e:
        print(f"Error in lambda execution: {str(e)}")
        return {
            'statusCode': 500,
            'body': json.dumps({
                'error': str(e)
            })
        }
```

Figure 10: Lambda Handler Function for LinkedIn Job Scraper

4.3 Results

After data collection and data pre-processing, next data visualization results and analysis phase begins. The primary goal of visualizing the job scraping results is to make complex data easy to understand and identify meaningful patterns (Wang et al., 2024). By presenting the job data in charts and graphs, it becomes easier to analyze trends such as which IT roles are in high demand, where jobs are located, the preferred experience levels, and the flexibility of work (remote/on-site). This analysis supports better decision-making for job seekers, career planners, and researchers interested in the UK IT job market.

The interactive power bi dashboard is the principal analytical device in the present project, which converts the unprocessed LinkedIn job-posting data into valuable information. This is because it links the clean dataset in AWS S3 and updates real-time visualisations as new jobs are scraped. The information can be sorted by date, type of job, level of experience, type of work and also the location and all the charts and maps on the dashboard will be updated accordingly. This interactive filtering allows the user to dig deep into individual roles (e.g., Data Scientist), locations (e.g. London) or job types (e.g., remote) and instantly visualize how the employment trends shift. The dashboard provides flexibility, interactivity, and accessibility to various stakeholders to the analysis by integrating experience, work arrangements, employer activity, and geographic distribution.

Employment Trends by Experience and Work Type

The sample is heavily skewed towards long time professionals. As shown in Figure 12: Employment Trends by Work Type, top-end jobs, with the majority of postings, constitute the higher end of the job market, with middle-end positions, and entry-level jobs comprising only a minority. This implies that employers prefer professionals who are able to work within the company without much training, and there is even the likelihood that graduates or other professionals in their early years may lack opportunities. This imbalanced state of affairs can potentially lead to long-term issues with talent pipelines when junior positions remain low-represented.

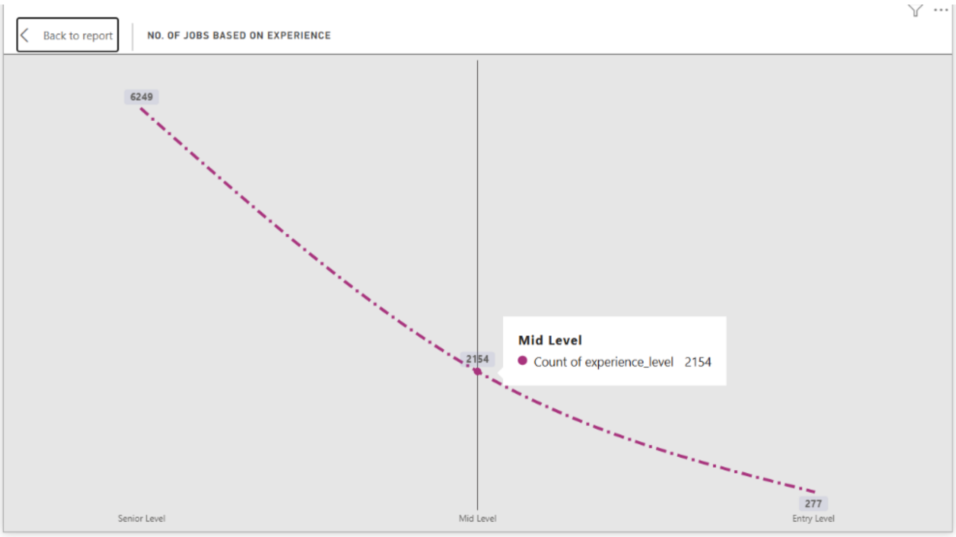


Figure 11: Employment trends by experience

Concerning work arrangements, the figure below reveals that on-site positions continue to dominate (47.5%), hybrid job opportunities (30%), and fully remote opportunities are the least prevalent (22.5%). This implies that despite the increased popularity of flexible working after the COVID-19 pandemic, employment in the UK IT industry is still predominantly office-based. This will demonstrate the flexible approach taken by job seekers in regards to location interests, and employers might need to consider hybrid and remote practices as a way of staying competitive in the talent market.

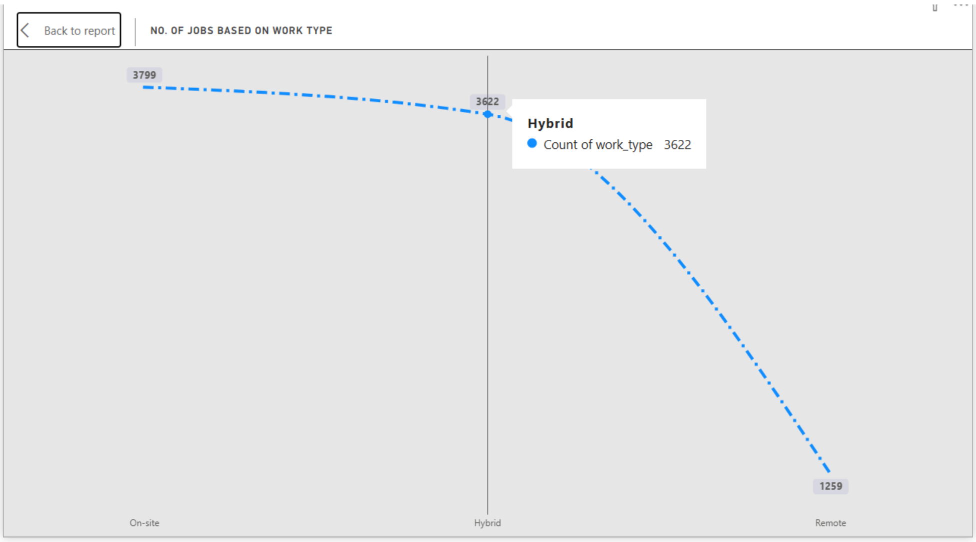


Figure 12: Employment Trends by Work Type

Hiring by Job Role and Employer

The analysis also indicates there is an important role- and company-wise variance. As Figure: Job Title Chart shows, Full Stack Engineers, Data Scientists and Machine Learning Engineers are some of the most sought-after positions. This fits within the industry trends of focusing on software development, artificial intelligence, and data-driven decision-making.

Likewise, in Figure 13, postings by the company and work type are shown. JD Sports Fashion proves to be the busiest recruiter, with most of the roles it advertises being on-site, whereas GardPass Consulting is shown to be a more balanced recruiter, with both on-site and remote jobs being equally represented. Hays is moderately flexible with both hybrid and on-site jobs. These differences indicate, as some employers remain with the traditional in-office format, some are changing to hybrid or remote formats. This provides numerous insights to job seekers about the employer policies and also helps recruiters and HR teams to evaluate themselves against others.

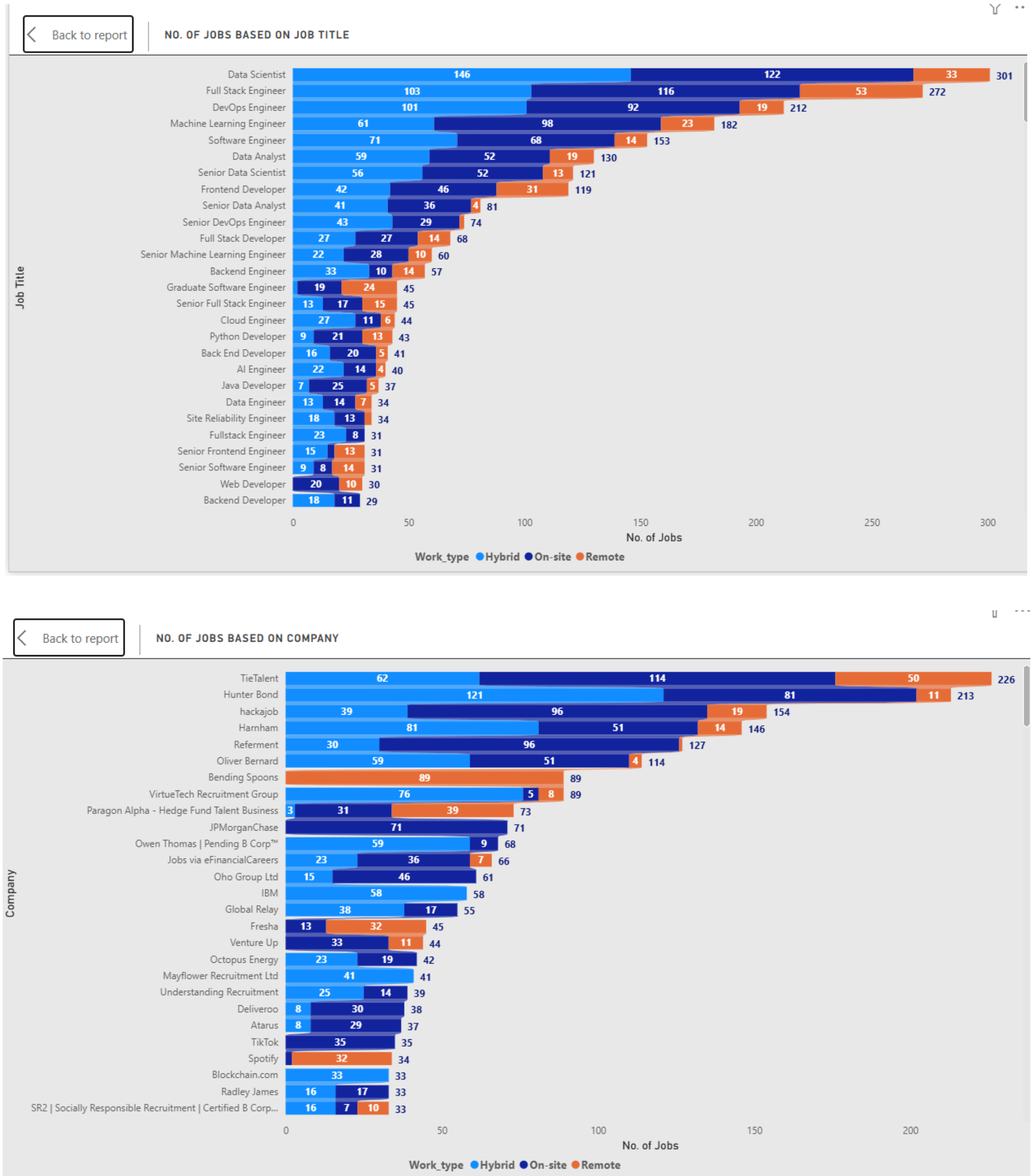


Figure 13: Hiring by Job Role and Employer

Regional and Geospatial Distribution

Another critical finding is the geographical concentration of IT jobs. As Figure 14 map illustrates, London is the core of IT job opportunities, followed by Manchester, Birmingham, and Leeds. The greater bubble size in London represents a much greater concentration of opportunities than elsewhere. This is indicative of the fact the city has become the UKs leading technology hub, with its agglomeration of financial, digital, and consulting services.

What cities like this have taught policymakers is the issue of regional disparity in employment opportunities that persists. Job seekers, on the other hand, can leverage this understanding to balance the trade-offs of moving into major hubs vs. pursuing smaller but expanding regional markets.

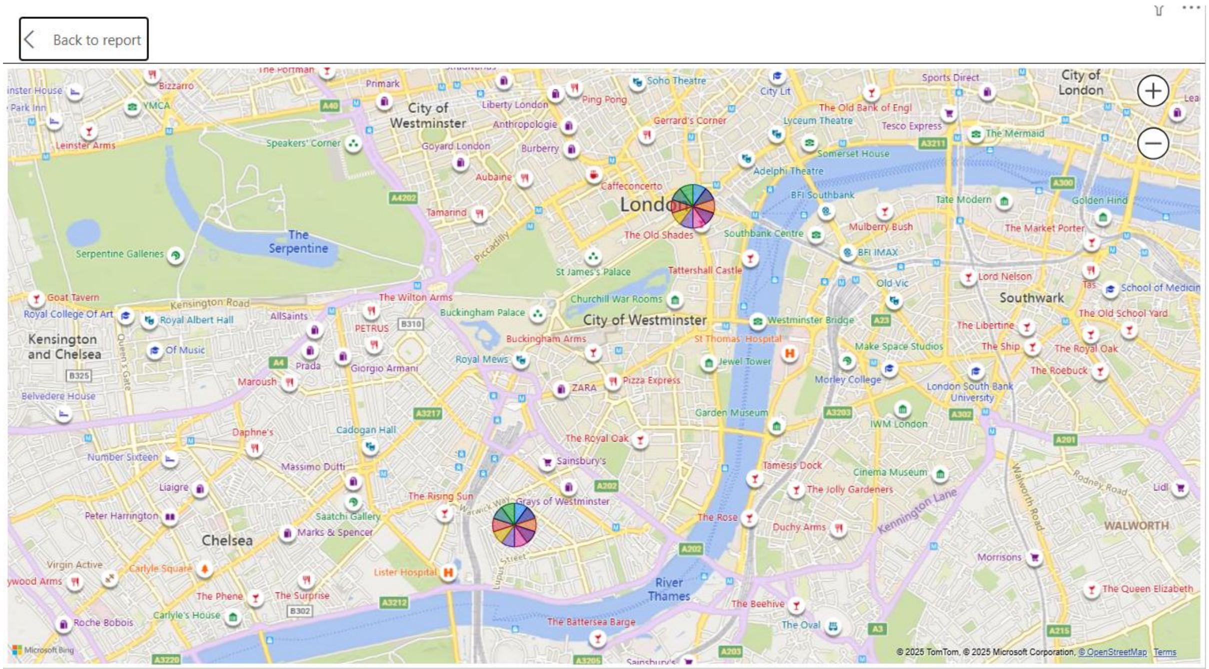


Figure 14: Regional and Geospatial Distribution

Power BI Dashboard Overview

The dashboard allows one to see the UK IT job market in detail by mixing different types of graphs and charts. The main feature is a clear map showing places of work using geographic coordinates, so that users know where the biggest hiring centers like London, Manchester, and Birmingham are, along with areas with fewer job opportunities (Kollydas, 2025).

This Figure 15, represents an interactive Power BI dashboard created to show data about the LinkedIn job market in the UK IT industry. The dashboard is designed to provide several different views which users can use to explore the employment patterns by experience, type of work, job title, company, and location. Filter options like date range, category, level of experience, the type of work I want to do, and location, aid in the dynamism of search results, allowing more flexibility in studying the trends in different circumstances. The visualisations at the top-left display the number of jobs by experience level and type of work. The findings show that there are more senior-level jobs, then there are mid-level jobs, and finally, entry-level jobs are fewer. Regarding the type of work, on-site positions are the most prevalent and hybrid jobs are not uncommon, as compared to distant opportunities, which are quite scarce.

The charts on the right hand side of the top are job offerings by job title, with some variance between them, including Full Stack Engineer, Data Scientist and Machine Learning Engineer. Such insights are used to determine which specialisations are more demanded. In a similar fashion, the bottom-right chart will show information on hiring by the company, giving relational information on which organisations are hiring the most, as well as their interest in hybrid, on-site, or remote hiring. The dashboard considers the job distributions in London with a geospatial map in the centre of the dashboard. This allows a user to visualise geographic concentrations of opportunities with larger bubbles signifying higher number of postings in certain areas.

In general, the dashboard brings together various views, such as skills, geography, and employers, into one interface. Its data interaction filters enable various stakeholders (job seekers, recruiters, and policymakers) to delve into specific information like employers adopting flexible work practices, understanding job demands in a particular position or location, or finding employment hotspots in cities.

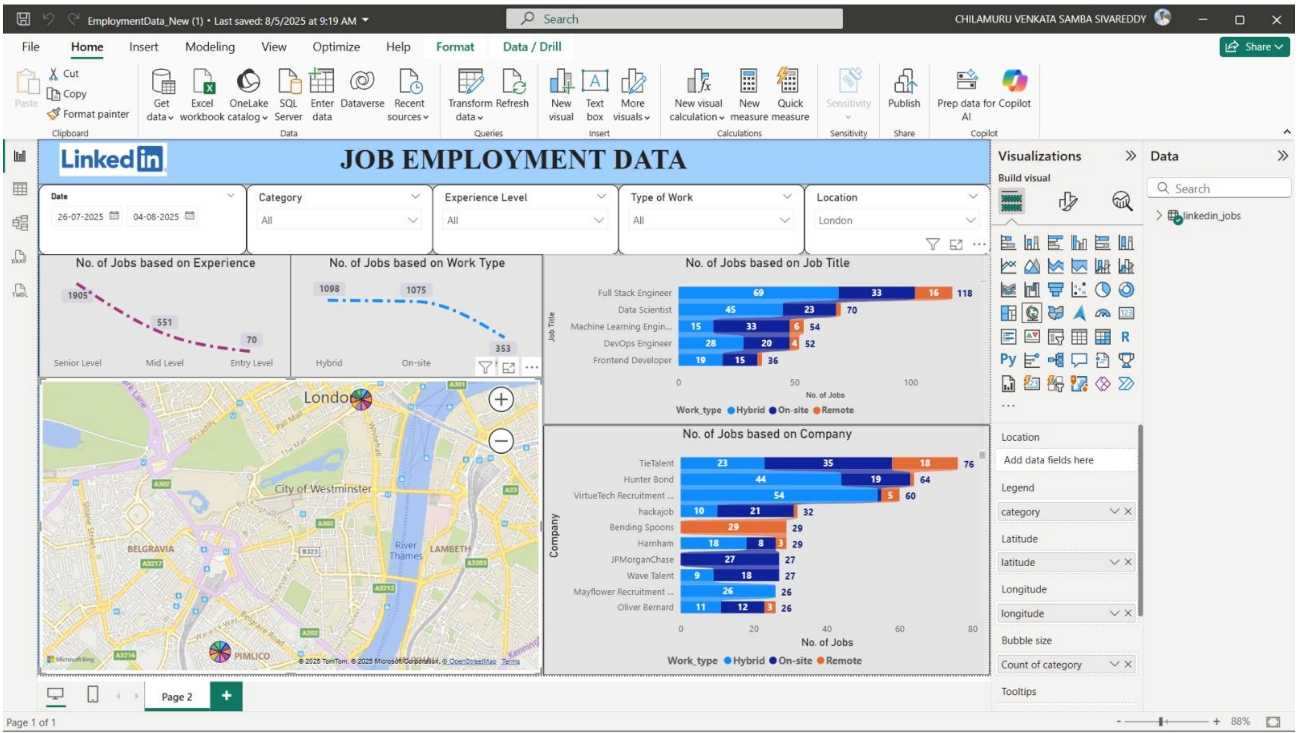


Figure 15: Interactive Dashboard

Beneficiaries of the Dashboard

This dashboard serves a diverse set of users, each drawing value from the insights in ways aligned with their goals. Job seekers are perhaps the most direct beneficiaries. With this tool, they can tailor their job search based on preferred work style (remote, hybrid, on-site), level of experience, and specific job categories. For example, a graduate looking for entry-level data analyst roles in remote settings can instantly identify where those opportunities exist and which companies are hiring. This ensures their job search is efficient, targeted, and informed by real-time trends.

Recruiters and HR teams gain a competitive edge through the dashboard's employer comparison features (Liu et al., 2024). By analyzing the roles offered by rival firms and the regions where demand is highest, they can adjust their strategies, whether by expanding job posts to underrepresented areas or offering flexible work options to attract talent. The insights play a role in deciding on hiring, creating a brand, and organizing employees.

These people also gain value from using the data. The dashboard makes it possible to watch job demand, discover where people are working, and keep track of changes in flexible and traditional job roles. Such trends can assist in creating policies for skill development, training, and providing employment in different regions. Basically, the dashboard allows everyone, whether a single job hunter or a company strategist, to use proven facts to make decisions about the UK IT job market.

4.4 Conclusion

The implementation effectively automated the collection and analysis of UK IT job postings, converting unstructured LinkedIn data into structured insights. By scraping, cleaning, and categorizing the data, it enabled visualizations that highlighted key trends like the dominance of on-site roles, high demand for senior talent, and job concentration in cities such as London and Manchester. The interactive Power BI dashboard provides a valuable tool for job seekers, recruiters, and policymakers to explore the IT job market in real time, showcasing the power of automation and visualization in tracking employment trends.

Chapter 5: Conclusion and Future Work

5.1 Project Summary and achievements

The objective of this project was to build a system that constantly collects, processes, and displays information on IT jobs in the UK by using LinkedIn as the main data source. Python web scraping, cloud storage with Amazon S3, a PostgreSQL database, and Power BI dashboards were used in the technical framework to set up an automated system. It managed to collect key information such as title, location, experience level, type of work, and posting date from different IT positions like cybersecurity, data analysis, and software engineering.

The project brings about various kinds of achievements. The pipeline would collect the job data effectively, and with the help of AWS Lambda, would automatically schedule the scraping and uploading so that fresh listings of jobs would be continuously scraped in real time with no human help. The visualisation layer enabled stakeholders to interact with the dataset, filtering by job category, region, or work preference (e.g., remote vs. on-site). Analytical insights were also achieved such as identifying leading recruiters, experience level trends, and regional hiring patterns (Hacıoğlu-Hoke, Känzig and Surico, 2021). Despite working within LinkedIn's access limitations, the scraper was able to deliver relevant and recent job listings. In addition to AWS S3, PostgreSQL was added to enable the storage of job data in a queryable format and organised data to allow a larger amount of analytical work to be performed in the future.

Furthermore, the project demonstrates a proof-of-concept for data-driven labour market monitoring, an area often dependent on manual reports or outdated government datasets. By automating the flow of job data from source to dashboard, this study validates the feasibility of real-time market analysis. While scope limitations exist, the outcomes confirm that even a single-platform pipeline can reveal meaningful employment insights that can be expanded further.

5.2 Limitations of the Study

Despite the strengths and practical value of this study, several limitations must be acknowledged that may have influenced the scope and accuracy of the results. First and foremost, the data source, LinkedIn job postings, represents only a segment of the broader job market. Many companies post jobs on other platforms (e.g., Indeed, Glassdoor, company websites), and therefore, the dataset might not capture the complete picture of the UK's IT employment landscape. Additionally, LinkedIn listings often depend on user-submitted content, which can vary in structure, detail, and accuracy.

Some job descriptions may lack crucial information such as work type, location specificity, or experience requirements, resulting in gaps or inaccuracies in classification.

The location data can be inaccurate because of its geospatial aspect. Even though regular expressions were applied to process the job listings, some locations were not geocoded with precision. Locations that were not specific, like "United Kingdom" or "Multiple Locations," were either taken out or turned into one big point, which might have influenced how the data was displayed in the visualisation.

This way of classifying people by their experience level was based on keywords, which is efficient but may overlook the details of various jobs. Even though a job titled “Software Engineer” can refer to many seniority levels, it may not be properly labeled because of the use or lack of certain keywords. Also, because the algorithm used just job descriptions, it did not notice the different terms and experience levels used within each company.

The work had to stop before some goals could be achieved because of technical and time constraints. Following AWS Lambda’s rules, only a few pages were scraped in each category, so some important listings might not have been included. Even so, these limits do not lessen the main results, yet they do suggest that one should be careful when applying them to the whole job market.

5.3 Lessons Learned

There were several important technical and strategy-related lessons that came up while working on the project. It became clear that checking and cleaning data as it is gathered is very important. Because the information on LinkedIn is not always organized, the job locations and types could be very different. Therefore, to make these fields more regular, usage of regular expressions and conditional rules is required.

One more thing learned was how useful it is to write code that can be used and reused. A class was created for the scraper, and it contains helper functions for processing, cleaning, and classifying data. Because of this structure, it was much easier to make updates and find errors, even when things on the website changed.

Managing how many API calls could be made and updating dynamic content was another tough task. Since LinkedIn does not give access to job data through open APIs, scraping was done by parsing HTML and carefully limiting the number of requests. Therefore, it became necessary to run thorough tests, stick to specific intervals, and build in reliable safety features.

Finally, making the data interactive on a dashboard clearly showed how powerful visualization is. By formatting the insights in a simple way, the data could be easily accessed by any user.

5.4 Future Work

After this study, making a few enhancements could greatly boost how the dashboard functions and how it is used. Adding Indeed, Reed, and Glassdoor to your research would help you see the job market more evenly and prevent bias from LinkedIn. Using named entity recognition or topic modeling could make the classification of work type or experience level more accurate by examining job descriptions in a more meaningful way (Baumgartner et al., 2022). Job density can be shown more clearly with postcode-level mapping or heat maps. If the tool included information from outside, such as average salaries in each city, it would benefit users. Adding time-series forecasting to the dashboard would let it show predictions. Analyzing old job postings can help machine learning models foresee the demand for jobs in the future, making it easier to plan or train workers (Green and Heywood, 2023). To sum up, role-based access in a web or mobile application can boost its usability. Adding custom views, interactive features, instant scraping, and alerts would make the dashboard more useful and lively for all users.

5.5 Practical Implications / Real-World Impact

The findings of this study matter outside of the classroom and are very important for practical use. The interactive dashboard allows job seekers to find job openings by checking their preferred location, years of experience, and type of work (Copper, 2021). It enables users to find plenty of possibilities in certain areas or remote jobs that don’t need them to relocate. Seeing the current job market in real time allows individuals to change their applications and match their career goals (Ledezma, 2021).

Using this tool, recruiters and employers can analyze the job market, check their rival companies’ actions, and modify how they recruit people. Should a company become aware that competitors are providing more remote or hybrid chances, it may reform its job policies to secure the best employees.

It is helpful for policy makers and workforce planners to have prompt and data-based information about IT employment. The information shown on the dashboard can direct investments in skills, technology, or jobs. For instance, finding areas with very few job openings may point to the need for more support (Xu and Liu, 2023).

The dashboard may also assist schools in deciding on their curriculum. Being aware of the most important roles and the expertise needed for them allows educational institutions to support the industry’s needs. So, the project supports people in choosing their careers and also helps build the workforce and embrace digital changes for the future.

5.6 Conclusion

This project successfully built a real-time, cloud-based system to monitor and visualize UK IT job trends using LinkedIn data. By integrating Python-based scraping, AWS storage, and Power BI dashboards, it provided clear insights into employment patterns based on job role, geography, and work preferences. The analysis revealed a high demand for senior-level roles, a continued preference for on-site work, and strong job concentration in major urban centers. Such information is useful for people who are job hunting, employers, and those involved in making decisions for the future. Even though the project achieved its main objectives, it had problems because it depended on one data set and used keywords for classification. Still, the results prove that the system is effective and relevant. More work should involve using additional data, improving how text is classified, and adding features that can predict results. All in all, the project proves that by using big data and geospatial analytics, we can get useful and valuable information about the labour market and use it to make decisions in different industries.

References

Admas Abtew and Assefa, A. (2023). A Review on the impact of big data analytics on the employment of technical graduates in the IT industry. Big Data.

Ahmed Zaki, A.M., Mohamed Fathy, A.M., Carnevale, M. and Giberti, H. (2022). Application of Realtime Robotics platform to execute unstructured industrial tasks involving industrial robots, cobots, and human operators. *Procedia Computer Science*, 200, pp.1359–1367. doi:https://doi.org/10.1016/j.procs.2022.01.337.

Ather, D., Singh, G., Chaudhary, N., Verma, P., Kler, R. and Gupta, A. (2024). Analyzing Trends, Skills Demand, and Salary Prediction in the AI and ML Job Market. *2024 International Conference on Intelligent & Innovative Practices in Engineering & Management (IIPEM)*, [online] pp.1–8. doi:https://doi.org/10.1109/iipem62726.2024.10925738.

Bali Ronit, Sharma Anukansha and Mala Shuchi (2024). Modeling the Geospatial Trend Changes in Jobs and Layoffs by Performing Sentiment Analysis on Twitter Data. *International Journal of Performability Engineering*, 20(2), pp.120–120. doi:https://doi.org/10.23940/ijpe.24.02.p7.120130.

Baumgartner, H.M., Granillo, M., Schulkin, J. and Berridge, K.C. (2022). Corticotropin releasing factor (CRF) systems: Promoting cocaine pursuit without distress via incentive motivation. *PLOS ONE*, 17(5), p.e0267345. doi:https://doi.org/10.1371/journal.pone.0267345.

Bibri, S.E. and Krogstie, J. (2020). The emerging data–driven Smart City and its innovative applied solutions for sustainability: the cases of London and Barcelona . *Energy Informatics*, 3(1). doi:https://doi.org/10.1186/s42162-020-00108-6.

Bone, M., Ehlinger, E. and Stephany, F. (2023). *Skills or Degree? The Rise of Skill-Based Hiring for AI and Green Jobs*. [online] arXiv.org. Available at: https://arxiv.org/abs/2312.11942?.

Brenning, A. and Henn, S. (2023). Web scraping: a promising tool for geographic data acquisition. *arXiv (Cornell University)*. doi:https://doi.org/10.48550/arxiv.2305.19893.

Copper, C. (2021). Explorations in Library Design: *Art Documentation: Journal of the Art Libraries Society of North America*, 40(2), pp.249–267. doi:https://doi.org/10.1086/716737.

Dalton, C.M. and Thatcher, J. (2015). Inflated Granularity: Spatial Big Dataa and Geodemographics. *SSRN Electronic Journal*. doi:https://doi.org/10.2139/ssrn.2544638.

Dritsas, E. and Trigka, M. (2025). Remote Sensing and Geospatial Analysis in the Big Data Era: A Survey. Remote Sensing, [online] 17(3), pp.550–550. doi:https://doi.org/10.3390/rs17030550.

Georgieff, A. and Hyee, R. (2022). Artificial Intelligence and Employment: New Cross-Country Evidence. *Frontiers in Artificial Intelligence*, 5(832736). doi:https://doi.org/10.3389/frai.2022.832736.

Green, C.P. and Heywood, J.S. (2023). Performance pay, work hours and employee health in the UK. *Labour Economics*, [online] 84, p.102387. doi:https://doi.org/10.1016/j.labeco.2023.102387.

Grus, Ł., Castelein, W., Crompvoets, J., Overduin, T., Loenen, B. van, Groenestijn, A. van, Rajabifard, A. and Bregt, A.K. (2011). An assessment view to evaluate whether Spatial Data Infrastructures meet their goals. *Computers, Environment and Urban Systems*, 35(3), pp.217–229. doi:https://doi.org/10.1016/j.compenvurbsys.2010.09.004.

G Weir-Smith and Ahmed, F. (2019). Unemployment in South Africa: Building a Spatio-temporal Understanding. *South African Journal of Geomatics*, [online] 2(3), pp.218–230. Available at: https://www.ajol.info/index.php/sajg/article/view/106979

Hacıoğlu-Hoke, S., Känzig, D.R. and Surico, P. (2021). The distributional impact of the pandemic. *European Economic Review*, 134, p.103680. doi:https://doi.org/10.1016/j.euroecorev.2021.103680.

Jaiswal, K., Ievgeniia Kuzminykh and Sanjay Modgil (2024). Understanding the skills gap between higher education and industry in the UK in artificial intelligence sector. *Industry and Higher Education*. doi:https://doi.org/10.1177/09504222241280441.

Kazi, A., Farooq, M., Fatima, Z., Hina, S. and Abid, H. (2022). Analysis of LinkedIn Jobs for Finding High Demand Job Trends Using Text Processing Techniques 1. *IJCSNS International Journal of Computer Science and Network Security*, [online] 22(10), p.223. doi:https://doi.org/10.22937/IJCSNS.2022.22.10.29.

Kollydas, K. (2025). *What Online Job Posting Data Reveals About How Flexible and Remote Work Vary Across the UK – Local Policy Innovation Partnership Hub*. [online] Bham.ac.uk.

Ledezma, I. (2021). Product-market integration with endogenous firm heterogeneity. *Oxford Economic Papers*. doi:https://doi.org/10.1093/oep/gpab001.

Li, C., Wang, J., Wang, W. and Shi, H. (2022). RF-Based on Feature Fusion and Convolutional Neural Network Classification of UAVs. *8th International Conference on Computer and Communications (ICCC)*. [online] doi:https://doi.org/10.1109/iccc56324.2022.10065895.

Liu, Z., Clarke, J.M., Rohenkohl, B. and Barahona, M. (2024). *Patterns of co-occurrent skills in UK job adverts*. [online] arXiv.org. Available at: https://arxiv.org/abs/2406.03139.

Maghsoudi, M. (2023). *Uncovering the Skillsets Required in Computer Science Jobs Using Social Network Analysis*. [online] arXiv.org. Available at: https://arxiv.org/abs/2308.08582?.

Mohammad Sultan Mahmud, Joshua Zhexue Huang, Ruby, R. and Wu, K. (2023). An ensemble method for estimating the number of clusters in a big data set using multiple random samples. *Journal of Big Data*, 10(1). doi:https://doi.org/10.1186/s40537-023-00709-4.

Netrdová, P. and Nosek, V. (2020). Spatial Dimension of Unemployment: Space-Time Analysis Using Real-Time Accessibility in Czechia. *ISPRS International Journal of Geo-Information*, [online] 9(6), p.401. doi:https://doi.org/10.3390/ijgi9060401.

Ostermann, K., Eppelsheimer, J., Gläser, N., Haller, P. and Oertel, M. (2022). Geodata in labor market research: trends, potentials and perspectives. Journal for Labour Market Research, 56(1). doi:https://doi.org/10.1186/s12651-022-00310-x.

Qadir, J., Ali, A., ur Rasool, R., Zwitter, A., Sathiaseelan, A. and Crowcroft, J. (2016). Crisis analytics: big data-driven crisis response. *Journal of International Humanitarian Action*, 1(1). doi:https://doi.org/10.1186/s41018-016-0013-9.

Satpute, B.S., Yadav, R. and Yadav, P.K. (2023). Machine Learnig Approach for Prediction of Employee Salary using Demographic Information with Experience. *4th IEEE Global Conference for Advancement in Technology (GCAT)*, [online] pp.1–5. doi:https://doi.org/10.1109/gcat59970.2023.10353537.

Tawil, A.-R., Mohamed, M., Schmoor, X., Vlachos, K. and Haidar, D. (2023). Trends and Challenges Towards an Effective Data-Driven Decision Making in UK SMEs: Case Studies and Lessons Learnt from the Analysis of 85 SMEs. arXiv.org.

Tzimas, G., Zotos, N., Mourelatos, E., Giotopoulos, Konstantinos C and Zervas, P. (2024). From Data to Insight: Transforming Online Job Postings into Labor-Market Intelligence. *Information*, [online] 15(8), p.496. doi:https://doi.org/10.3390/info15080496.

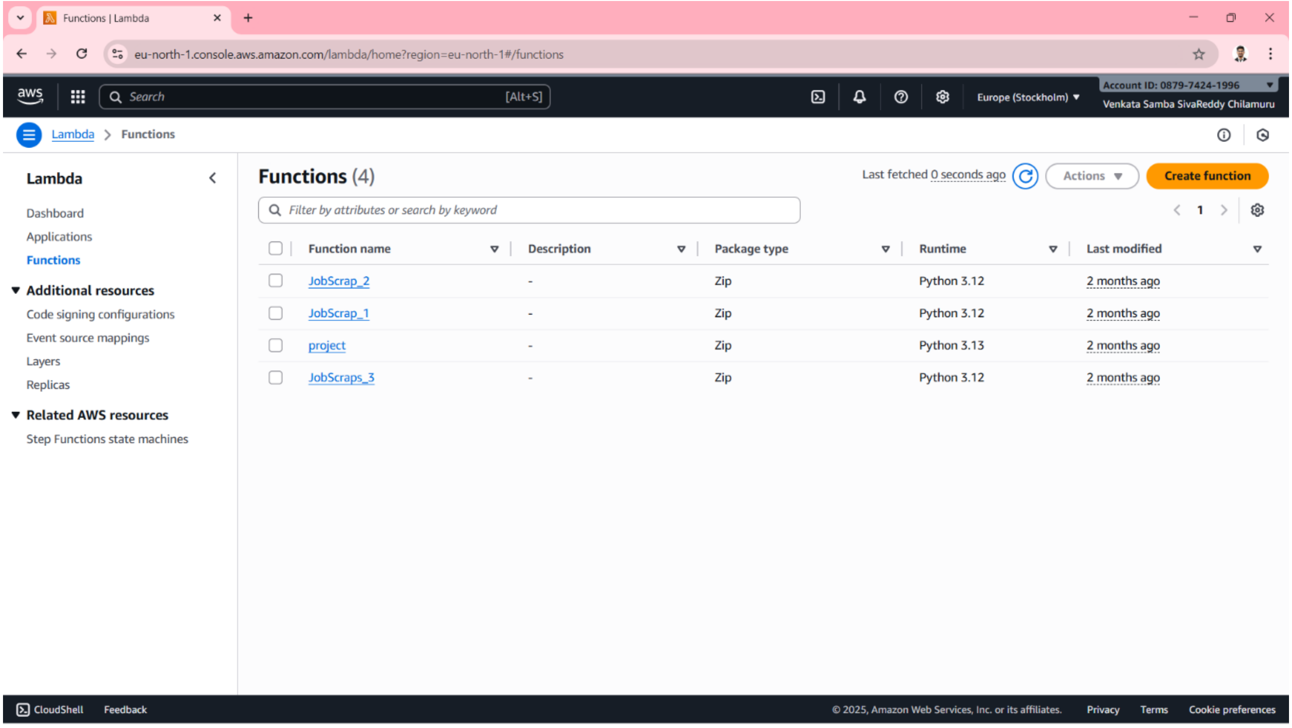
Vandenbroucke, D., Dessers, E., Joep Crompvoets, Bregt, A.K. and Jos Van Orshoven (2013). A methodology to assess the performance of spatial data infrastructures in the context of work processes. *Computers Environment and Urban Systems*, 38, pp.58–66. doi:https://doi.org/10.1016/j.compenvurbsys.2012.12.001.

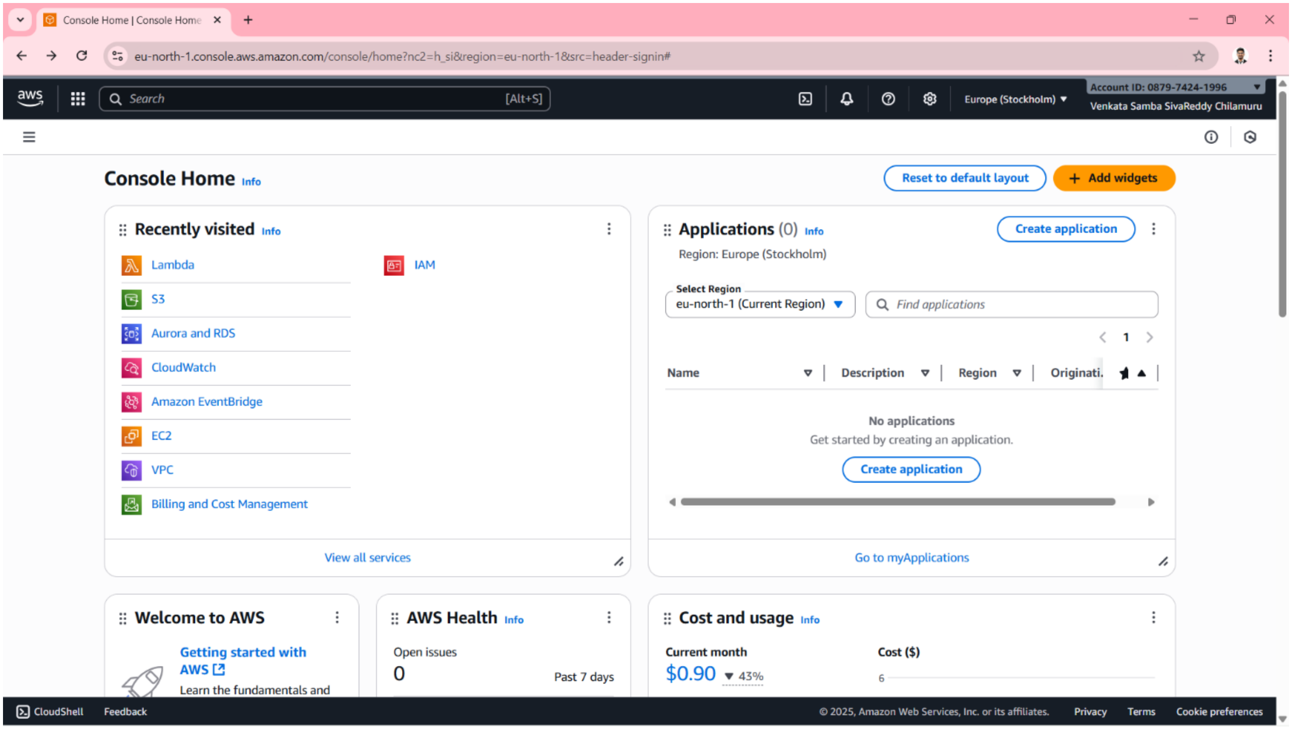
Wang, R., Chen, Q., Wang, Y., Xiong, L. and Shen, B. (2024). JobViz: Skill-driven visual exploration of job advertisements. *Visual Informatics*, [online] 8(3), pp.18–28. doi:https://doi.org/10.1016/j.visinf.2024.07.001.

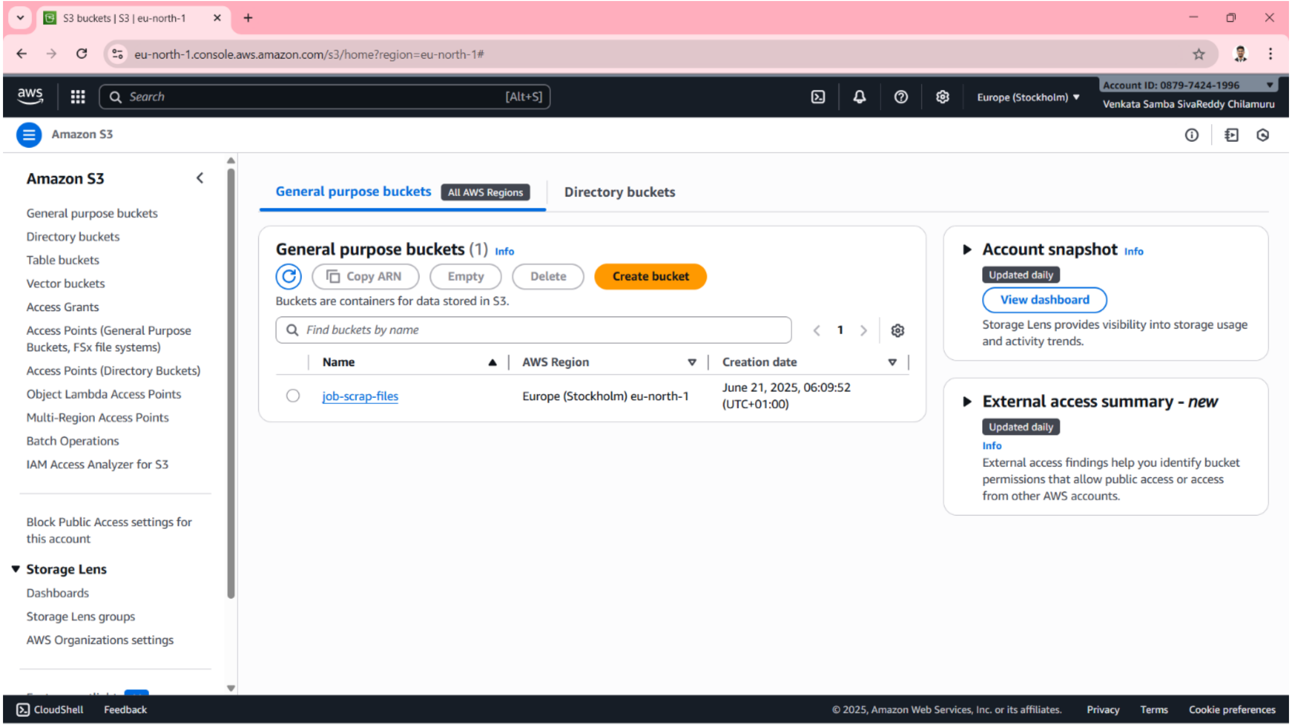
Wang, C.-H. and Chen, N. (2015). A GIS-based spatial statistical approach to modeling job accessibility by transportation mode: case study of Columbus, Ohio. *Journal of Transport Geography*, [online] 45, pp.1–11. doi:https://doi.org/10.1016/j.jtrangeo.2015.03.015.

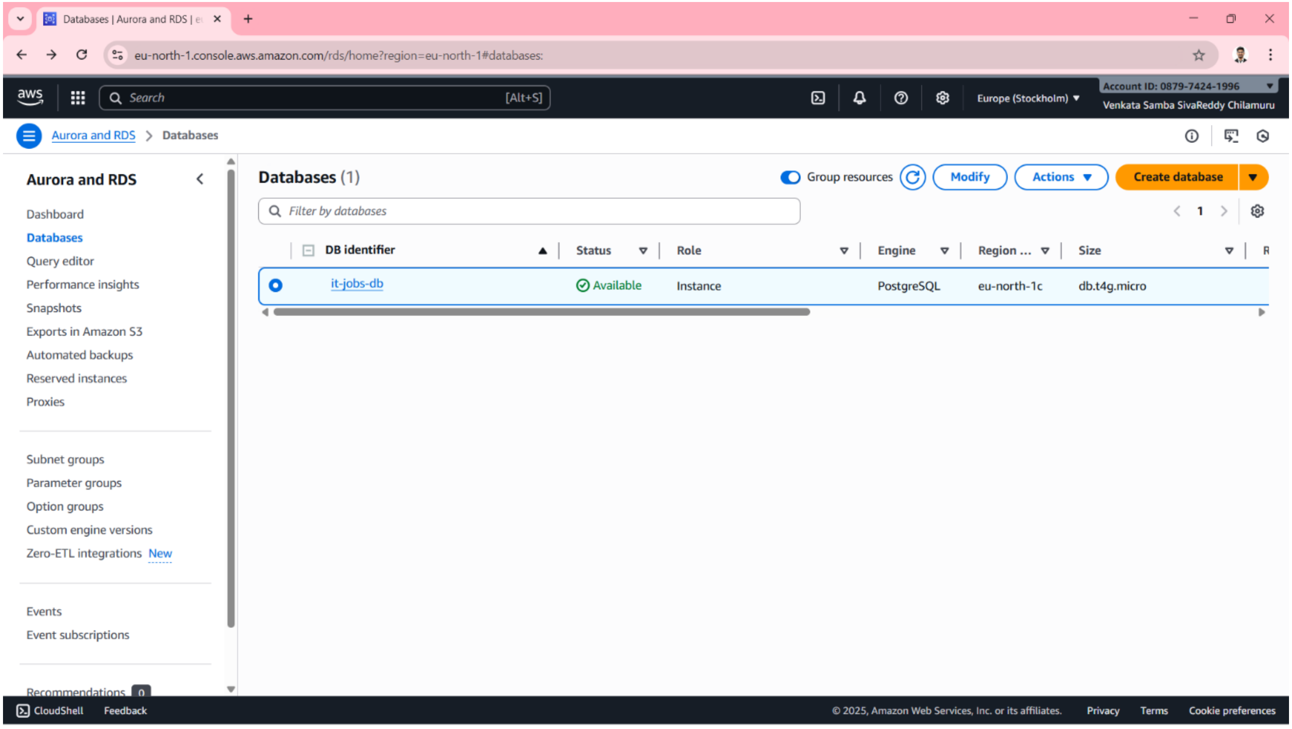
Wu, J., Gan, W., Chao, H.-C. and Yu, P.S. (2024). Geospatial Big Data: Survey and Challenges. arXiv.org.

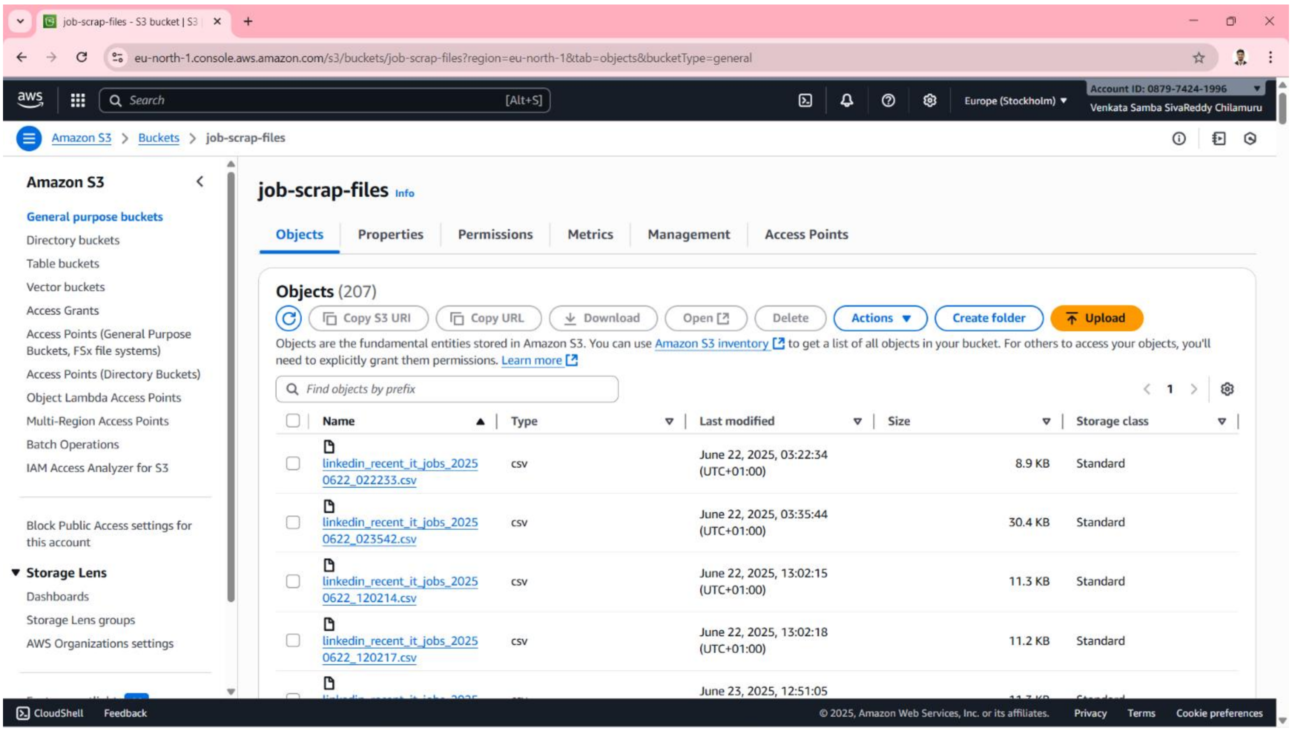
Xu, C. and Liu, X. (2023). The Economic Value of Language in China: How Important is Mandarin Proficiency in the Chinese Labor Market? A Bounding Approach. *Labour Economics*, 84, pp.102393–102393. doi:https://doi.org/10.1016/j.labeco.2023.102393.

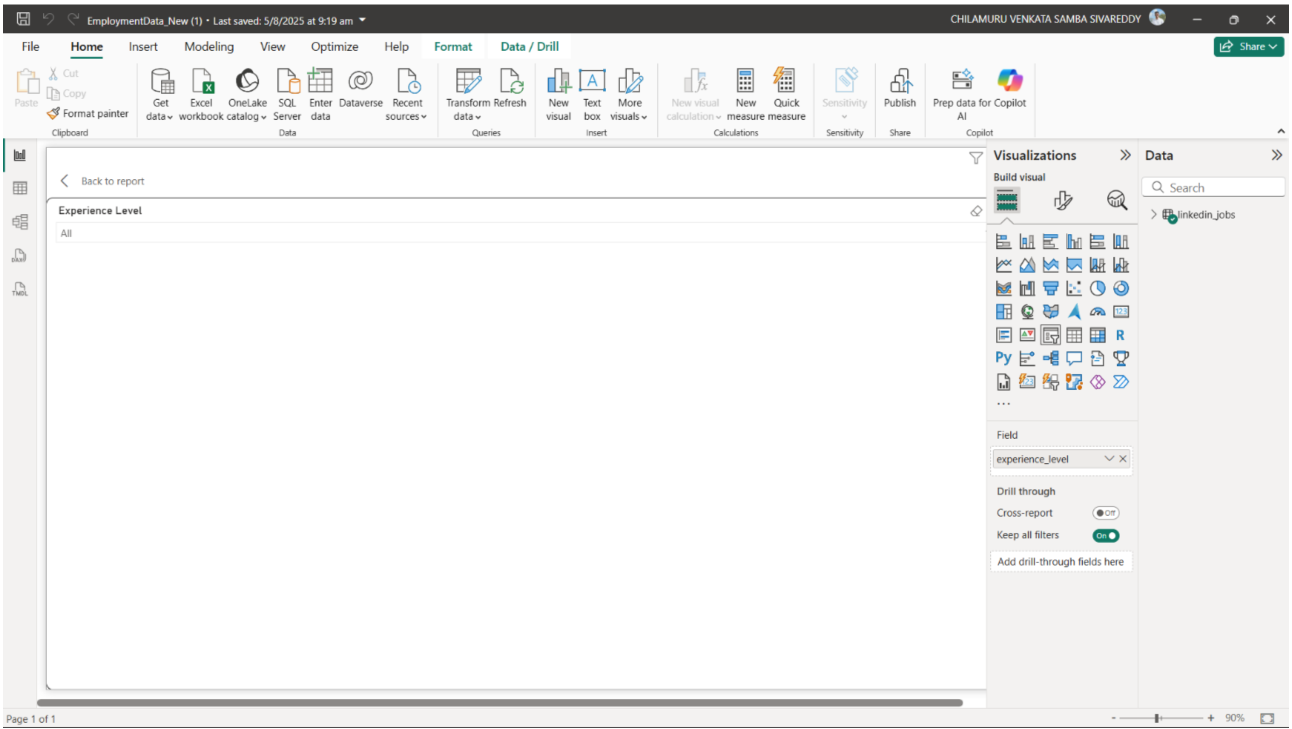

Functions | Lambda
eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/functions
Lambda > Functions
Functions (4)
Last fetched 0 seconds ago
Actions
Create function
Function name
Description
Package type
Runtime
Last modified
JobScrap_2 - Zip Python 3.12 2 months ago
JobScrap_1 - Zip Python 3.12 2 months ago
project - Zip Python 3.13 2 months ago
JobScraps_3 - Zip Python 3.12 2 months ago
Dashboard
Applications
Functions
Additional resources
Code signing configurations
Event source mappings
Layers
Replicas
Related AWS resources
Step Functions state machines

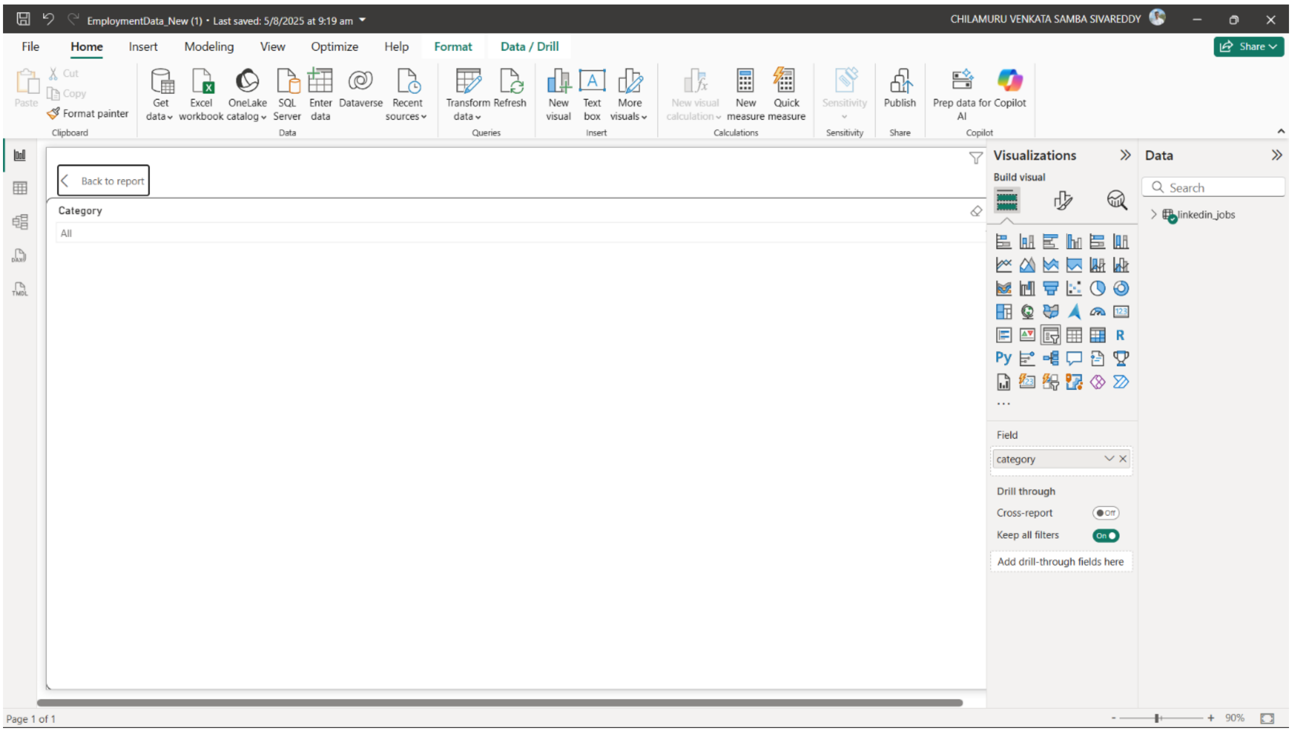

Console Home | Console Home
eu-north-1.console.aws.amazon.com/console/home?nc2=h_si&region=eu-north-1&src=header-signin#
Console Home
Reset to default layout
Add widgets
Recently visited
Lambda
S3
Aurora and RDS
CloudWatch
Amazon EventBridge
EC2
VPC
Billing and Cost Management
IAM
View all services
Applications (0)
Region: Europe (Stockholm)
Create application
No applications
Get started by creating an application.
Go to myApplications
Welcome to AWS
Getting started with AWS
AWS Health
Open issues
0
Past 7 days
Cost and usage
Current month
$0.90
43%

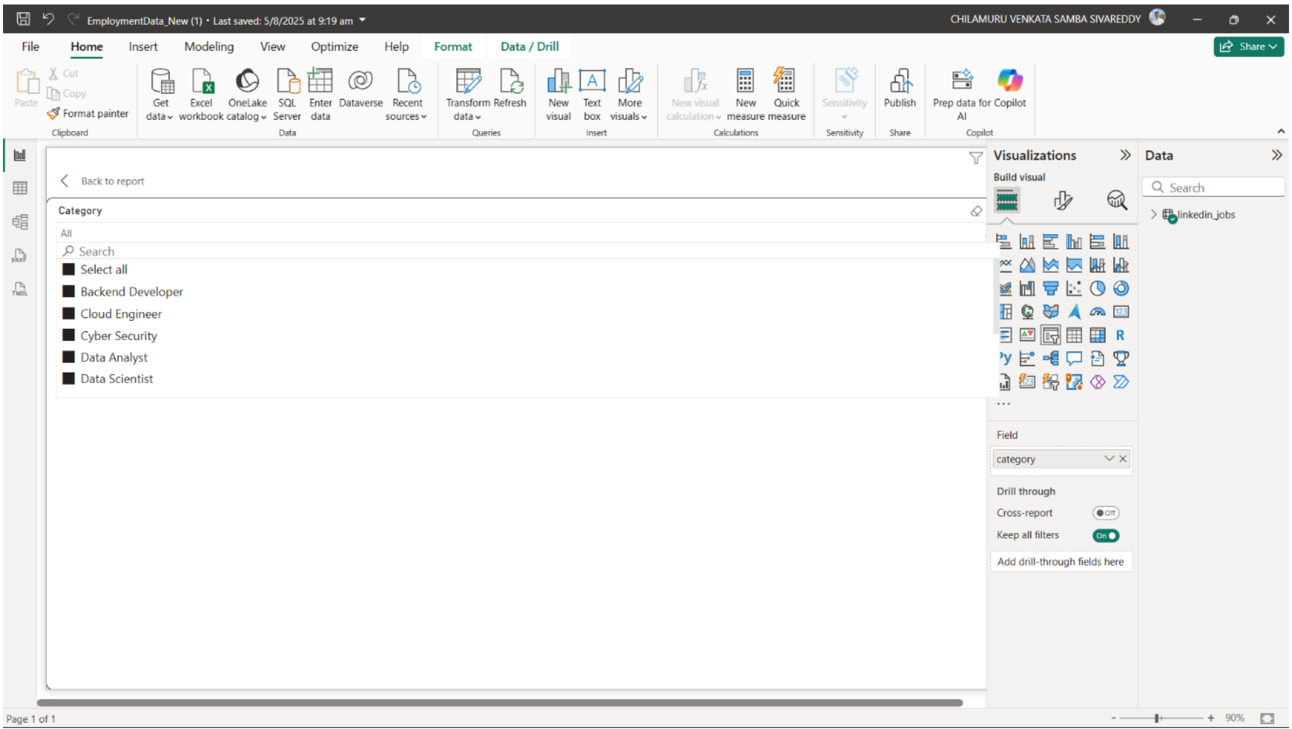

S3 buckets | S3 | eu-north-1
eu-north-1.console.aws.amazon.com/s3/home?region=eu-north-1#
Europe (Stockholm)
Account ID: 0879-7424-1996
Venkata Samba SivaReddy Chilamuru
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
General purpose buckets (1) Info
Copy ARN
Empty
Delete
Create bucket
Buckets are containers for data stored in S3.
Find buckets by name
Name
AWS Region
Creation date
job-scrap-files
Europe (Stockholm) eu-north-1
June 21, 2025, 06:09:52 (UTC+01:00)
Account snapshot Info
Updated daily
View dashboard
Storage Lens provides visibility into storage usage and activity trends.
External access summary - new
Updated daily
Info
External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

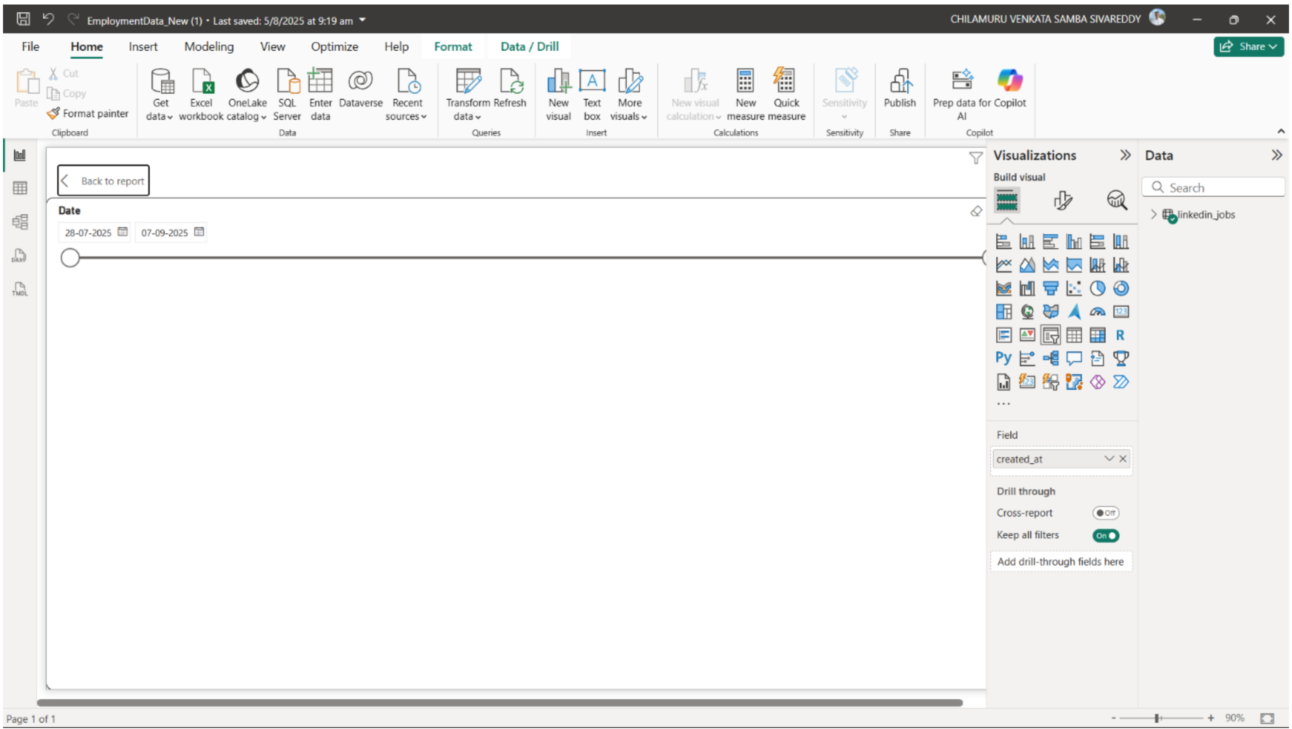

Databases | Aurora and RDS
eu-north-1.console.aws.amazon.com/rds/home?region=eu-north-1#databases:
Europe (Stockholm)
Account ID: 0879-7424-1996
Venkata Samba SivaReddy Chilamuru
Aurora and RDS > Databases
Aurora and RDS
Dashboard
Databases
Query editor
Performance insights
Snapshots
Exports in Amazon S3
Automated backups
Reserved instances
Proxies
Subnet groups
Parameter groups
Option groups
Custom engine versions
Zero-ETL integrations New
Events
Event subscriptions
Recommendations 0
Databases (1)
Group resources
Modify
Actions
Create database
Filter by databases
DB identifier
Status
Role
Engine
Region ...
Size
it-jobs-db
Available
Instance
PostgreSQL
eu-north-1c
db.t4g.micro
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

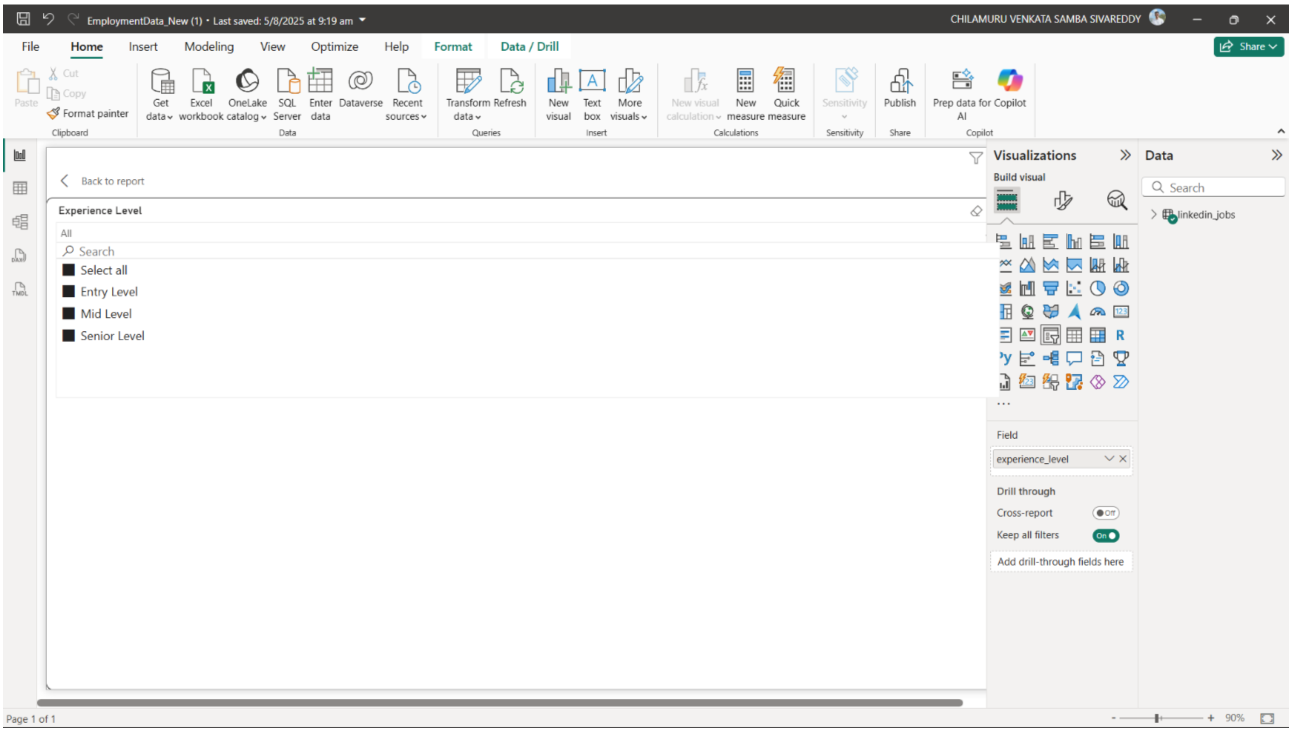


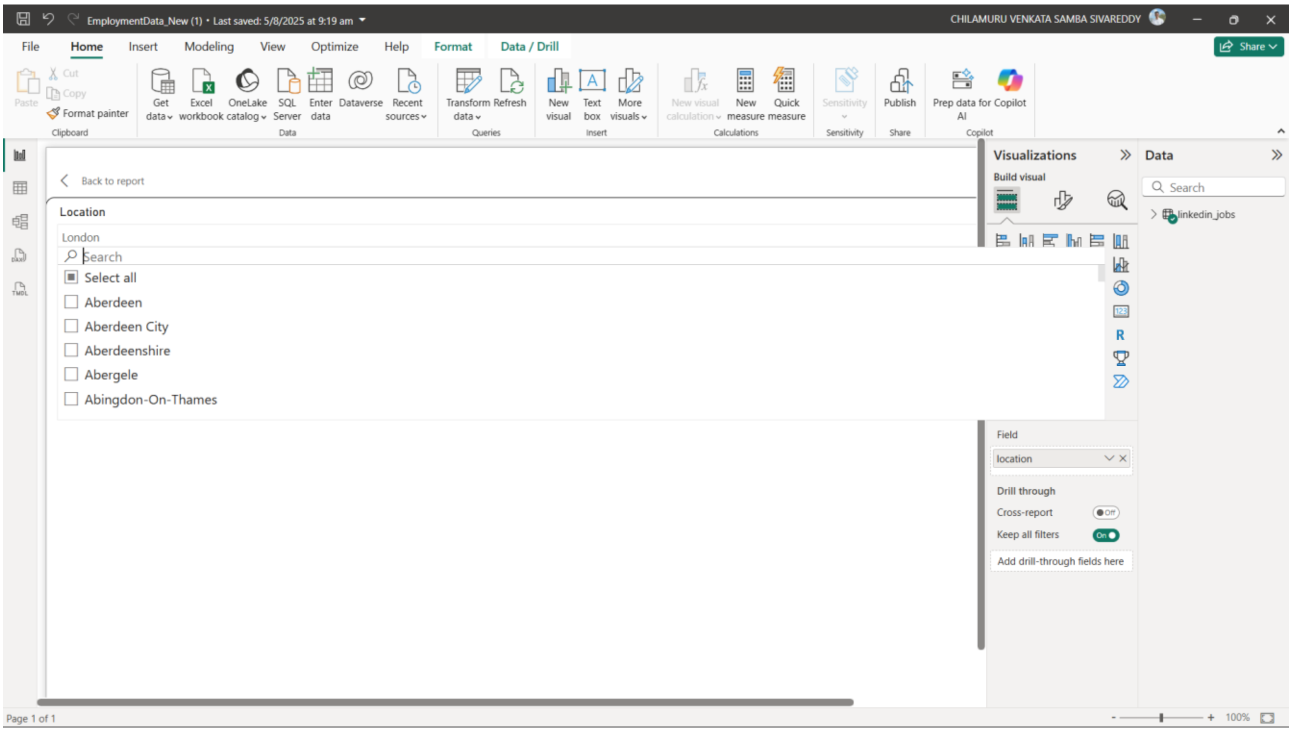


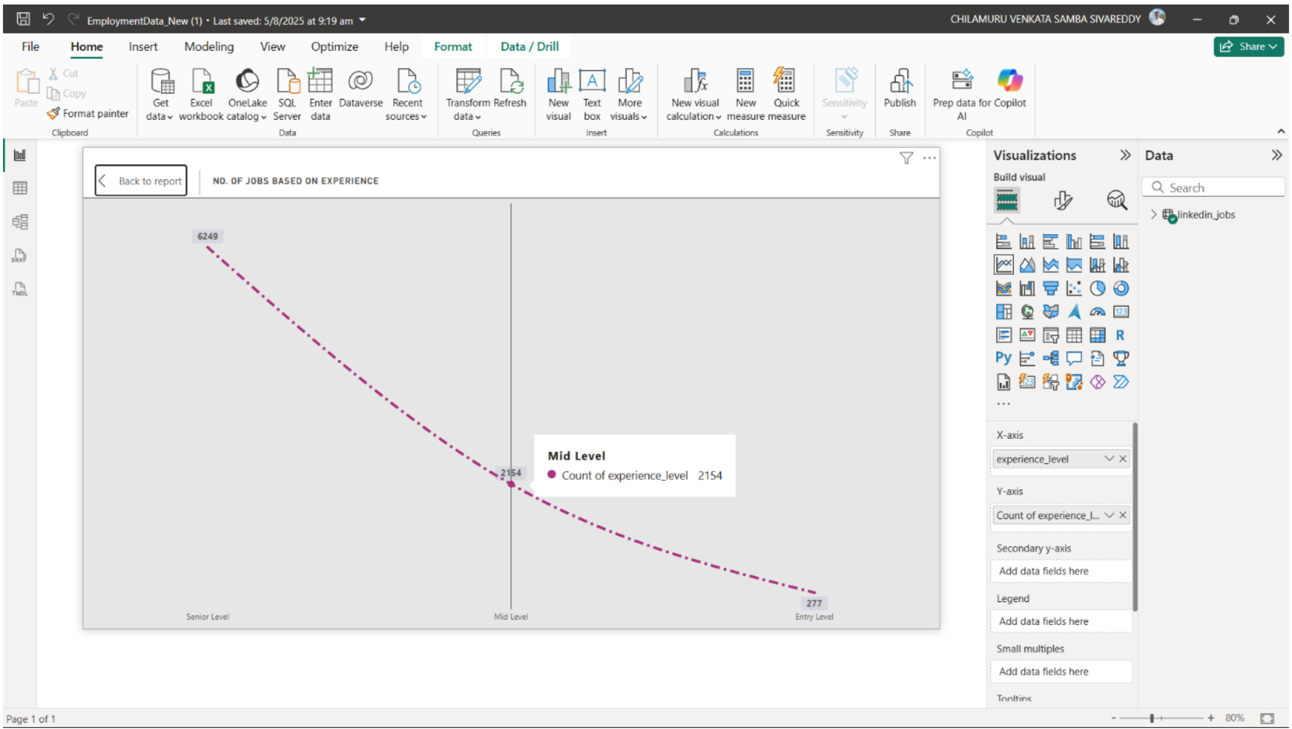

EmploymentData_New (1) • Last saved: 5/8/2025 at 9:19 am
CHILAMURU VENKATA SAMBA SIVAREDDY
File Home Insert Modeling View Optimize Help Format Data / Drill
Back to report
Category
All
Visualizations
Build visual
Data
Search
linkedin_jobs
Field
category
Drill through
Cross-report
Keep all filters
Add drill-through fields here
Page 1 of 1

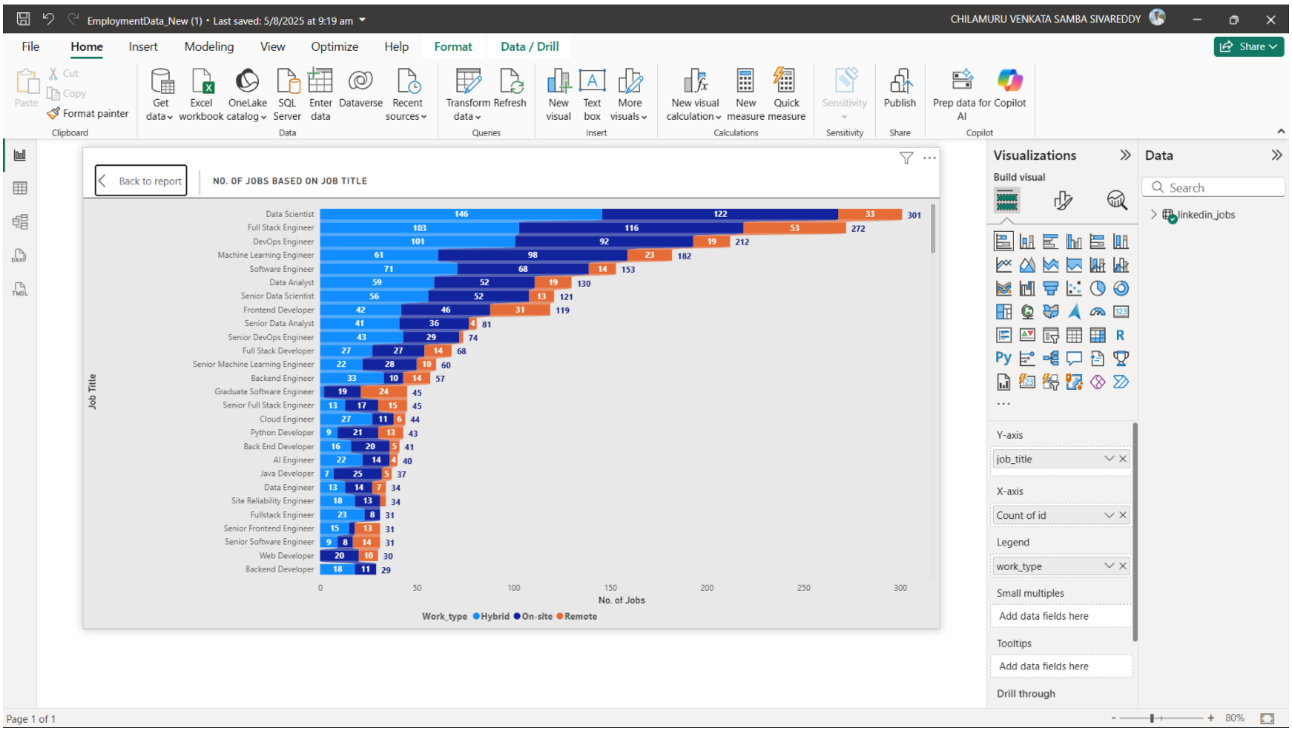

EmploymentData_New (1) • Last saved: 5/8/2025 at 9:19 am
CHILAMURU VENKATA SAMBA SIVAREDDY
File Home Insert Modeling View Optimize Help Format Data / Drill
Back to report
Category
All
Search
Select all
Backend Developer
Cloud Engineer
Cyber Security
Data Analyst
Data Scientist
Visualizations
Build visual
Data
Search
linkedin_jobs
Field
category
Drill through
Cross-report
Keep all filters
Add drill-through fields here
Page 1 of 1


EmploymentData_New (1) • Last saved: 5/8/2025 at 9:19 am
CHILAMURU VENKATA SAMBA SIVAREDDY
File
Home
Insert
Modeling
View
Optimize
Help
Format
Data / Drill
Share
Visualizations
Build visual
Data
Search
linkedin_jobs
Back to report
Date
28-07-2025
07-09-2025
Field
created_at
Drill through
Cross-report
Keep all filters
Add drill-through fields here
Page 1 of 1

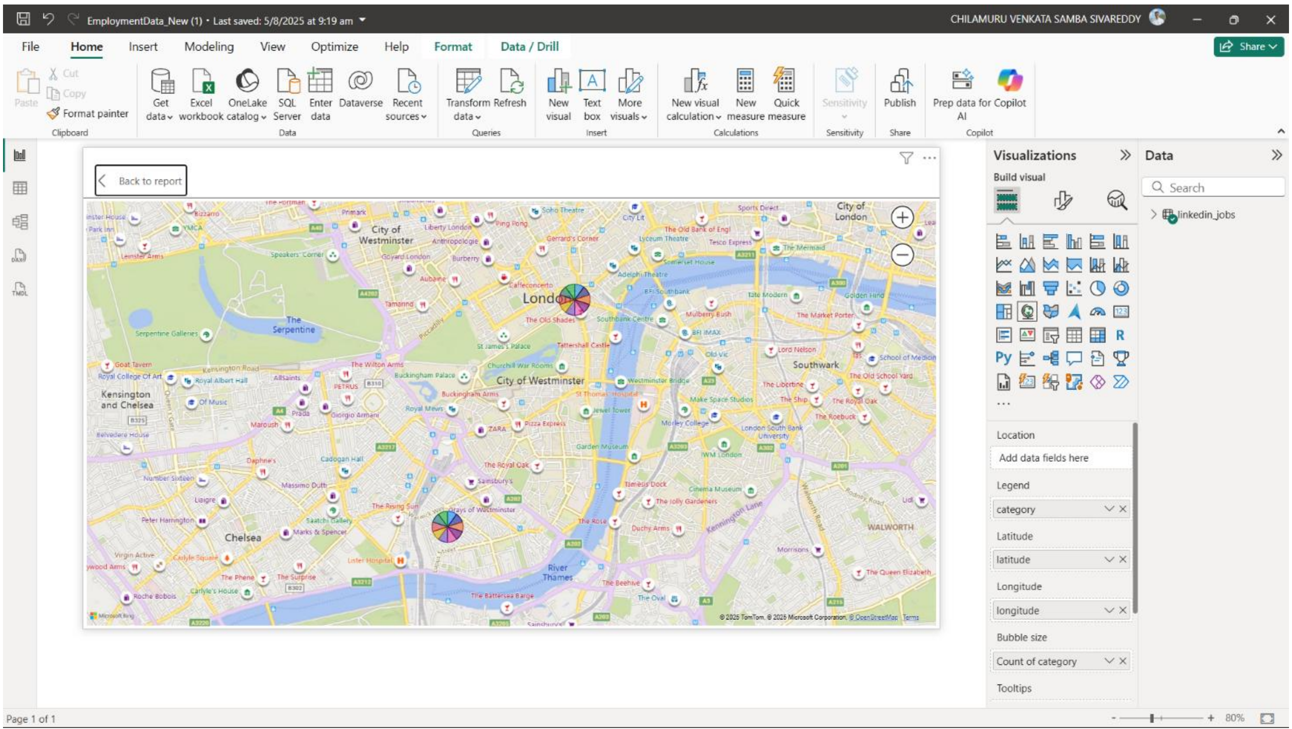

EmploymentData_New (1) • Last saved: 5/8/2025 at 9:19 am
CHILAMURU VENKATA SAMBA SIVAREDDY
File
Home
Insert
Modeling
View
Optimize
Help
Format
Data / Drill
Share
Visualizations
Build visual
Data
Search
linkedin_jobs
Back to report
Experience Level
All
Search
Select all
Entry Level
Mid Level
Senior Level
Field
experience_level
Drill through
Cross-report
Keep all filters
Add drill-through fields here
Page 1 of 1



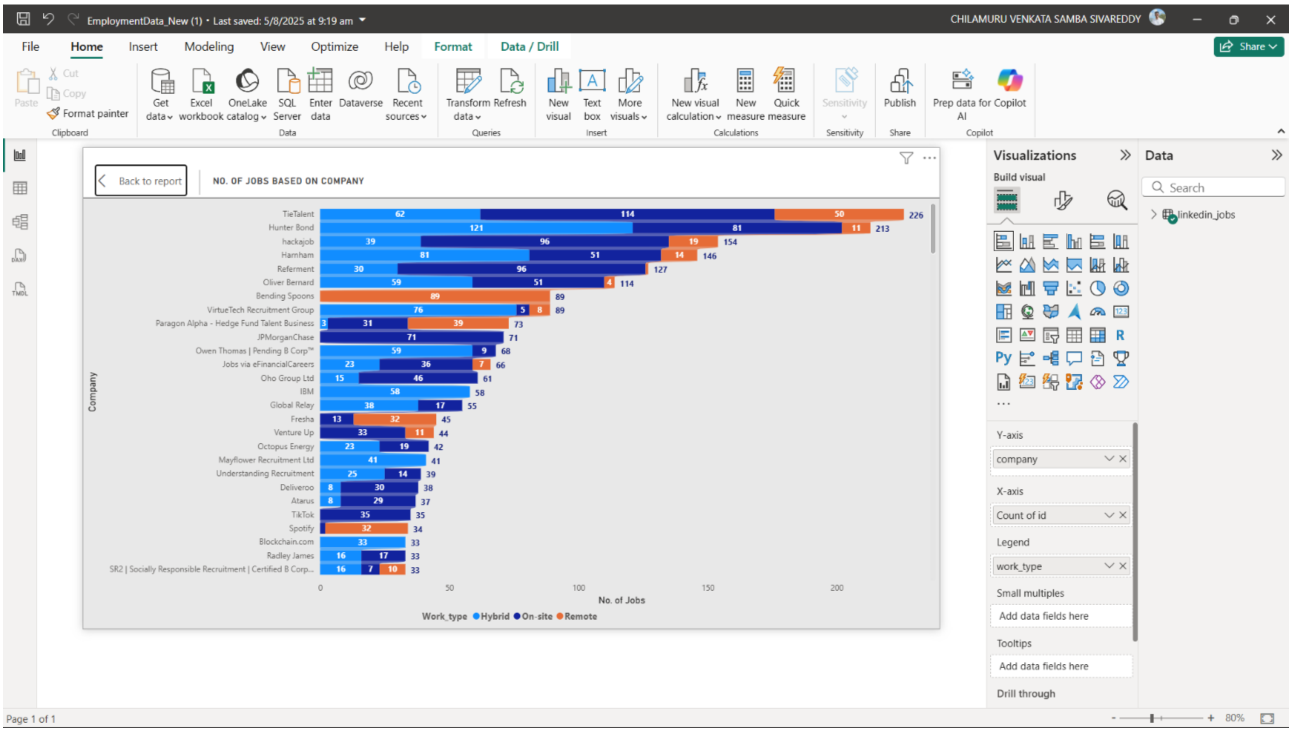

NO. OF JOBS BASED ON EXPERIENCE
6249
2154
277
Senior Level
Mid Level
Entry Level
Mid Level
Count of experience_level 2154

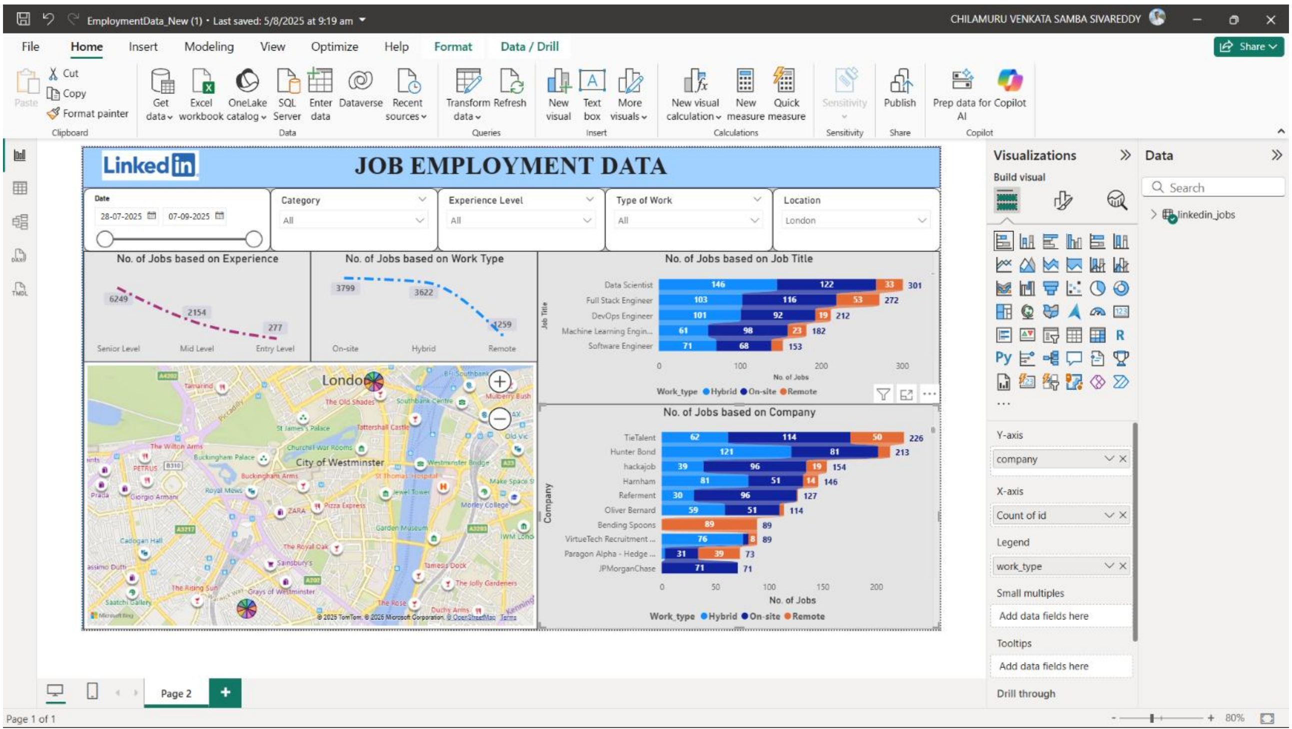


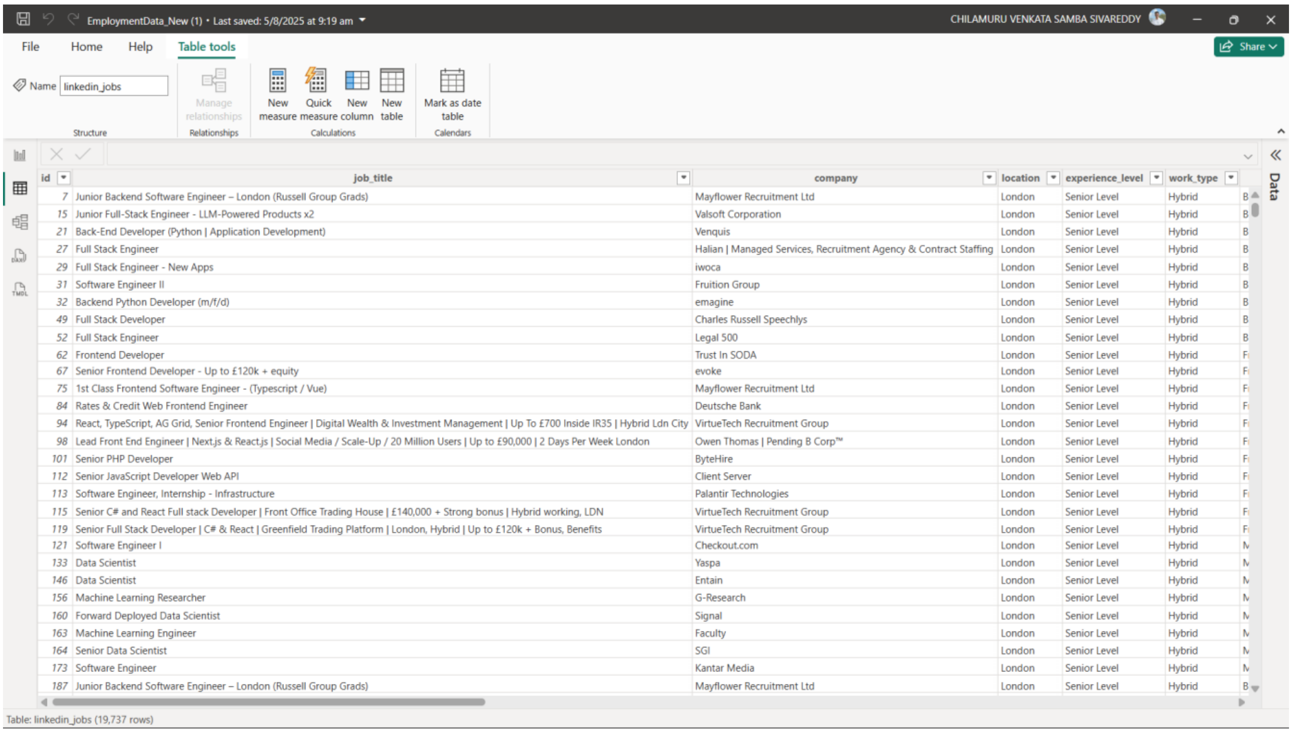
EmploymentData_New (1) • Last saved: 5/8/2025 at 9:19 am
CHILAMURU VENKATA SAMBA SIVAREDDY
File
Home
Insert
Modeling
View
Optimize
Help
Format
Data / Drill
Back to report
NO. OF JOBS BASED ON JOB TITLE
Job Title
No. of Jobs
Work_type ●Hybrid ●On-site ●Remote
Visualizations
Build visual
Data
linkedin_jobs
Y-axis
job_title
X-axis
Count of id
Legend
work_type
Small multiples
Tooltips
Drill through
Page 1 of 1

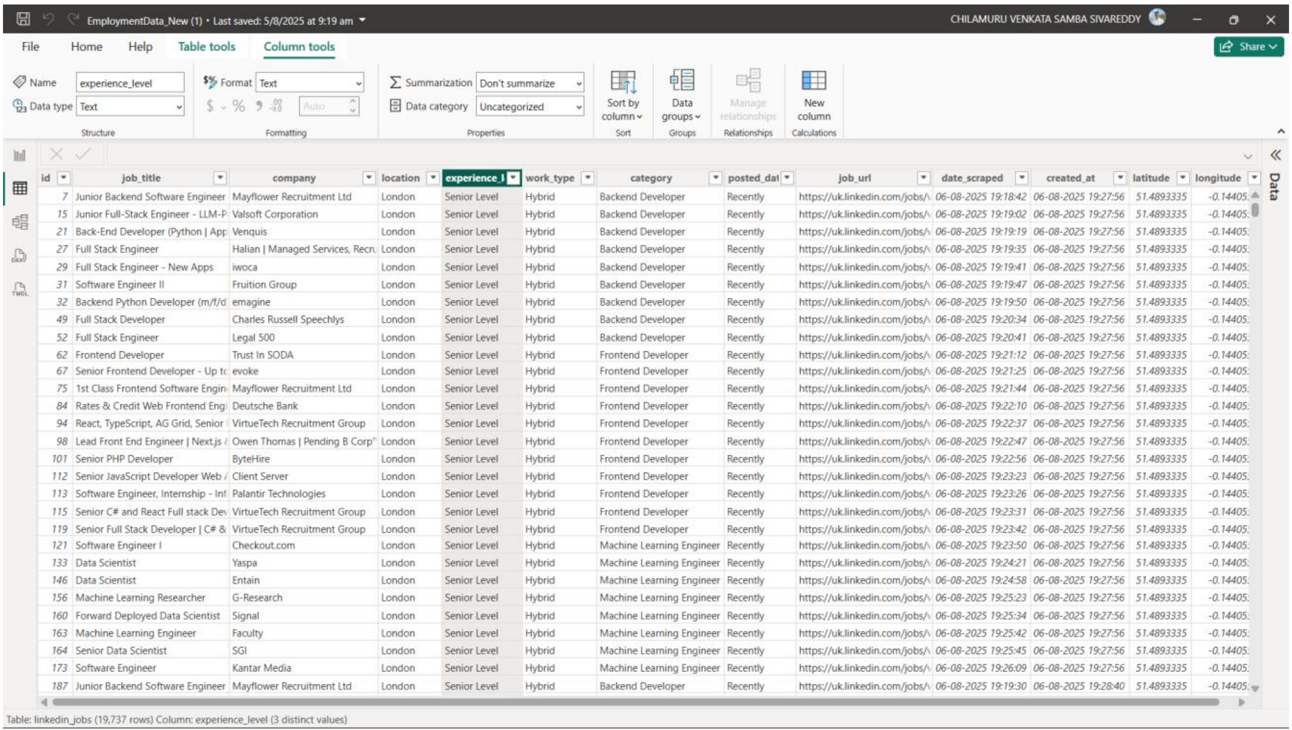


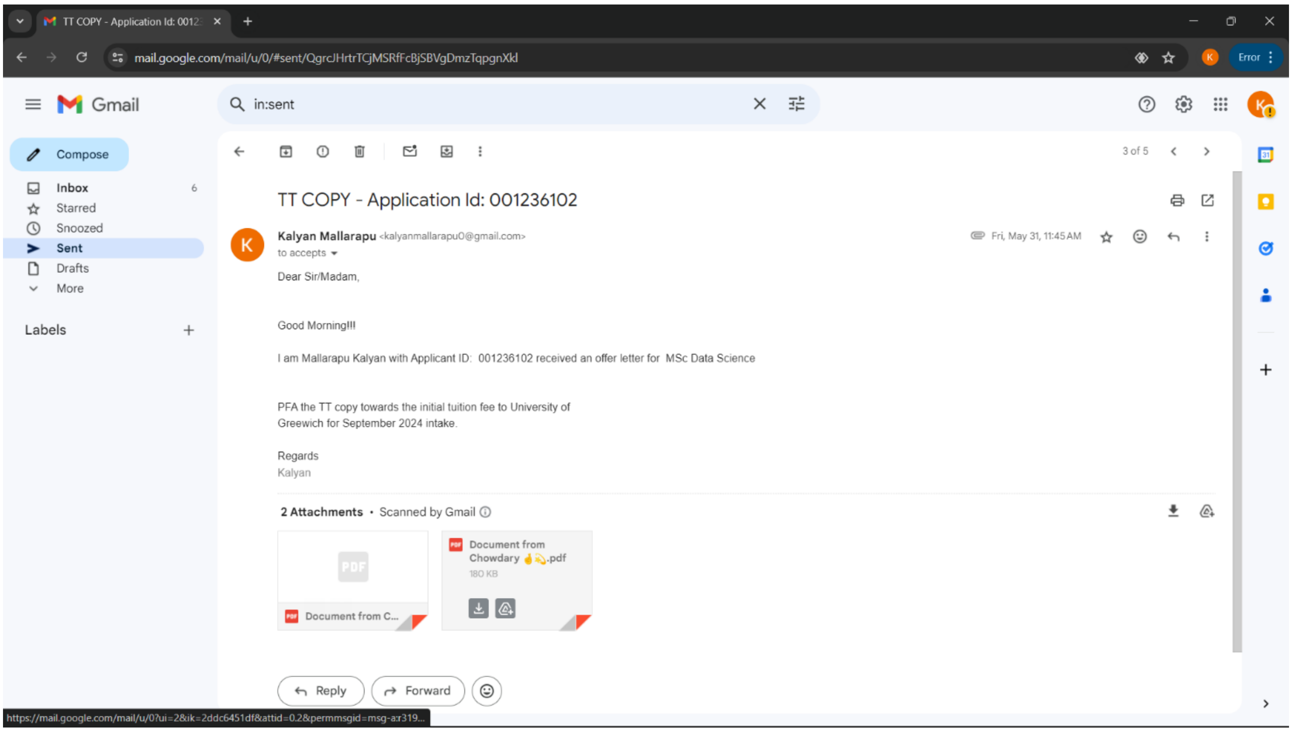
EmploymentData_New (1) • Last saved: 5/8/2025 at 9:19 am
CHILAMURU VENKATA SAMBA SIVAREDDY
Back to report
Visualizations
Build visual
Data
linkedin_jobs
Location
Legend
category
Latitude
latitude
Longitude
longitude
Bubble size
Count of category
Tooltips
Page 1 of 1


EmploymentData_New (1) • Last saved: 5/8/2025 at 9:19 am
NO. OF JOBS BASED ON COMPANY
Back to report
Company
No. of Jobs
Work_type Hybrid On-site Remote
Visualizations
Data
linkedin_jobs


JOB EMPLOYMENT DATA
Category
Experience Level
Type of Work
Location
London
No. of Jobs based on Experience
No. of Jobs based on Work Type
No. of Jobs based on Job Title
No. of Jobs based on Company
Page 2


EmploymentData_New (1) • Last saved: 5/8/2025 at 9:19 am
CHILAMURU VENKATA SAMBA SIVAREDDY
File
Home
Help
Table tools
Name
linkedin_jobs
Table: linkedin_jobs (19,737 rows)


EmploymentData_New (1) • Last saved: 5/8/2025 at 9:19 am
CHILAMURU VENKATA SAMBA SIVAREDDY
File
Home
Help
Table tools
Column tools
Name
experience_level
Table: linkedin_jobs (19,737 rows) Column: experience_level (3 distinct values)



TT COPY - Application Id: 001236102

Kalyan Mallarapu <kalyanmallarapu0@gmail.com>
to accepts

Fri, May 31, 11:45 AM

Dear Sir/Madam,

Good Morning!!!

I am Mallarapu Kalyan with Applicant ID: 001236102 received an offer letter for MSc Data Science

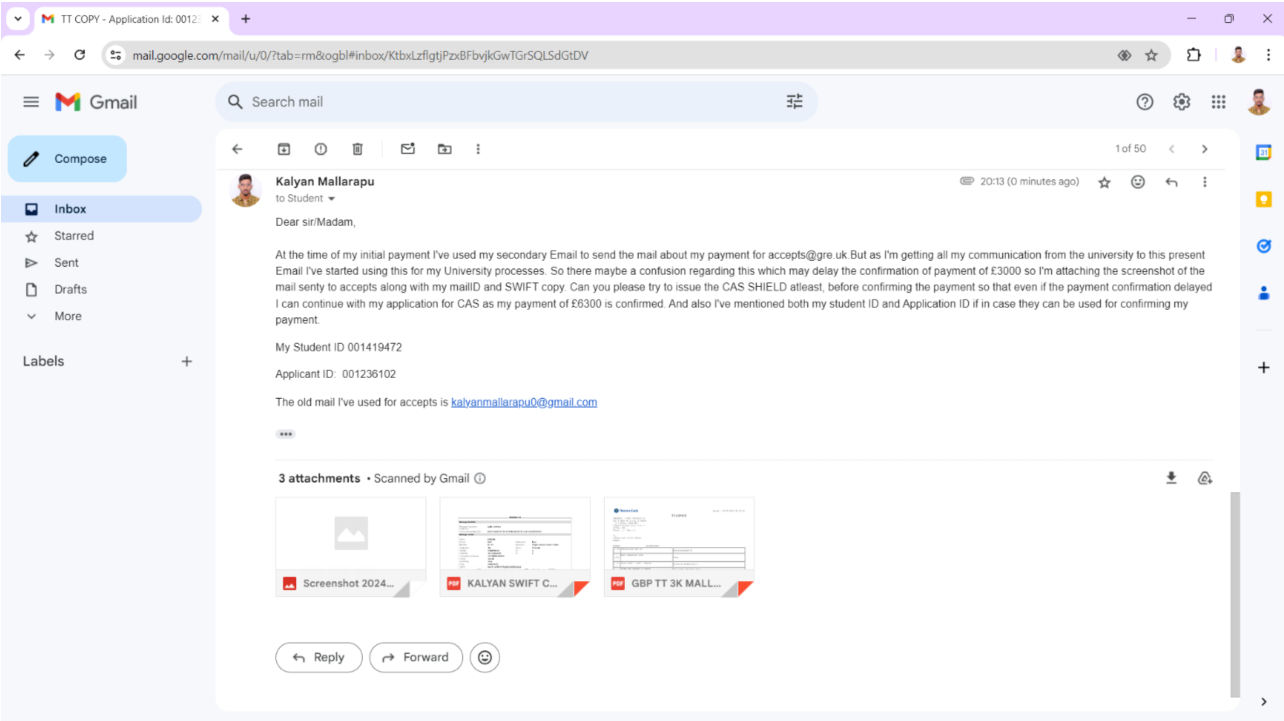
PFA the TT copy towards the initial tuition fee to University of Greewich for September 2024 intake.

Regards
Kalyan

2 Attachments • Scanned by Gmail

Document from C...

Document from Chowdary 👌💫.pdf
180 KB



Kalyan Mallarapu
to Student

20:13 (0 minutes ago)

Dear sir/Madam,

At the time of my initial payment I've used my secondary Email to send the mail about my payment for accepts@gre.uk But as I'm getting all my communication from the university to this present Email I've started using this for my University processes. So there maybe a confusion regarding this which may delay the confirmation of payment of £3000 so I'm attaching the screenshot of the mail senty to accepts along with my mailID and SWIFT copy. Can you please try to issue the CAS SHIELD atleast, before confirming the payment so that even if the payment confirmation delayed I can continue with my application for CAS as my payment of £6300 is confirmed. And also I've mentioned both my student ID and Application ID if in case they can be used for confirming my payment.

My Student ID 001419472

Applicant ID: 001236102

The old mail I've used for accepts is kalyanmallarapu0@gmail.com

3 attachments • Scanned by Gmail

Screenshot 2024...

KALYAN SWIFT C...

GBP TT 3K MALL...