

**PGRKAM – JOB RECOMMENDER AND ANALYTICS**

**A PROJECT REPORT**

## 

***Submitted by***

ROHIT S – 20211CAI0150

SYED AZEEM – 20211CAI0182

### *Under the guidance of,*

**Mr. Selvaganesh R**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING, ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING**

**PRESIDENCY UNIVERSITY**

**BENGALURU**

**NOV/DEC - 2025**



**PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**BONAFIDE CERTIFICATE**

Certified that this report **“PSCS\_16 – PGRKAM JOB ANALYTICS AND RECOMMENDER SYSTEM”** is a Bonafide work of **“ROHIT S (20211CAI0150), SYED AZEEM (20211CAI0182)”,** who have successfully carried out the project work and submitted the report for partial fulfilment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in **COMPUTER SCIENCE ENGINEERING, ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING** during 2025-26

|  |  |  |  |
| --- | --- | --- | --- |
| **Dr. Selvaganesh R**  Project Guide  Presidency School of Computer Science and Engineering  Presidency University | **Dr. Anandaraj S P**  Head of the Department Presidency School of Computer Science and Engineering  Presidency University | **Dr. Shakkeera L** Associate Dean  Presidency School of Computer Science and Engineering  Presidency University | **Dr. Duraipandian N**  Dean  PSCS & PSIS  Presidency University |

### Name and Signature of the Examiners

### 1)

2)

**PRESIDENCY UNIVERSITY**

**PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

DECLARATION

We the students of final year B.Tech in **COMPUTER SCIENCE ENGINEERING, ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING** at Presidency University, Bengaluru, named **ROHIT S and SYED AZEEM**, hereby declare that the project work titled **“PGRKAM JOB ANALYTICS AND RECOMMENDER SYSTEM – PSCS\_16”** has been independently carried out by us and submitted in partial fulfillment for the award of the degree of **B.Tech in** **COMPUTER SCIENCE ENGINEERING, ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING** during the academic year of 2025-26. Further, the matter embodied in the project has not been submitted previously by anybody for the award of any Degree or Diploma to any other institution.

|  |  |
| --- | --- |
| **ROHIT S** | **USN: 20211CAI0150** |
| **SYED AZEEM** | **USN: 20211CAI0182** |

PLACE: BENGALURU

DATE: 28-November 2025

**ACKNOWLEDGEMENT**

For completing this project work, We/I have received the support and the guidance from many people whom I would like to mention with deep sense of gratitude and indebtedness. We extend our gratitude to our beloved **Chancellor, Pro-Vice Chancellor, and Registrar** for their support and encouragement in completion of the project.

I would like to sincerely thank my internal guide **Dr. Selvaganesh R, Associate Professor**, Presidency School of Computer Science and Engineering, Presidency University, for his moral support, motivation, timely guidance and encouragement provided to us during the period of our project work.

I am also thankful to **Dr. Zafar Ali Khan, Professor, Head of the Department, Presidency School of Computer Science and Engineering** **in** **Artificial Intelligence and Machine Learning** Presidency University, for his mentorship and encouragement.

We express our cordial thanks to **Dr. Duraipandian N**, Dean PSCS & PSIS, **Dr. Shakkeera L**, Associate Dean, Presidency School of computer Science and Engineering and the Management of Presidency University for providing the required facilities and intellectually stimulating environment that aided in the completion of my project work.

We are grateful to **Mr. Selvaganesh R, PSCS** Project Coordinators**, Dr. Suma N G, Program Project Coordinator**, Presidency School of Computer Science and Engineering, or facilitating problem statements, coordinating reviews, monitoring progress, and providing their valuable support and guidance.

We are also grateful to Teaching and Non-Teaching staff of Presidency School of Computer Science and Engineering and also staff from other departments who have extended their valuable help and cooperation.

ROHIT S

SYED AZEEM

**Abstract**

The Punjab Government's initiative, PGRKAM, through their web and Android applications, has set up an integrated employment portal. It acts as a liaison between job seekers and recruiters by providing employment information to candidates and posting vacant positions. The system has a large active user base; however, it lacks integrated analytics for tracking user behaviour and patterns of engagement to provide optimized recommendations.

This project proposes the design and integration of an advanced analytics and recommendation module in PGRKAM. The analytics component captures the details about critical user data like referral channels, demographics, and job-related interactions. Alongside that, the recommendation engine uses machine learning algorithms for personalized job recommendations, including collaborative filtering, genetic algorithms, and hybrid approaches. In addition, success and failure rates of applications are analysed in aggregate to refine future recommendations.

It integrates data collection layers, processing pipelines, and visualization dashboards into one system architecture to ensure real-time cross-device tracking and analysis. It would provide administrators with insights into user journeys, application outcomes, and platform engagement through interactive visualizations. This will enhance the user experience, increase the success rate of employment, and support policymakers with data-driven strategies on workforce development.

**Table of Content**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Title** | **Page No.** |
|  | Acknowledgement | i |
|  | Abstract | ii |
|  | List of Figures | v |
|  | List of Tables | vi |
|  | Abbreviations | vii |
| 1. | Chapter 1: Introduction | 8 |
| 2. | Literature review | 11 |
| 3. | Methodology | 19 |
| 4. | Project management  4.1 Project timeline  4.2 Risk analysis  4.3 Project budget | 21 |
| 5. | Analysis and Design  5.1 Requirements  5.2 System Architecture  5.3 System Flowchart and Data Flow  5.4 Data Model and Schema  5.5 Security and Privacy Considerations | 24 |
| 6. | Implementation  6.1 Hardware  6.2 Software development tools  6.3 Dataset Generation and Preparation  6.4 Recommender Implementation  6.5 Streamlit Dashboard | 29 |
| 7. | Testing  7.1. Testing Objectives  7.2. Unit Testing  7.3. White Box Testing  7.4. Black Box Testing | 33 |
| 8. | Evaluation and Results  8.1 Test Points and Plan  8.2 Metrics and Methodology  8.3 Results  8.4 Insights and Discussion | 35 |
| 9. | Social, Legal, Ethical, Sustainability and Safety Aspects  9.1 Social Aspects  9.2 Legal aspects  9.3 Ethical aspects  9.4 Sustainability aspects  9.5 Safety aspects | 38 |
| 10. | Conclusion | 39 |
|  | References | 40 |
|  | Appendix | 42 |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure** | **Caption** | **Pg.no** |
| Fig 1.1 | Sustainable development goals | 10 |
| Fig 3.1 | The V model methodology | 20 |
| Fig 4.1 | PESTEL Analysis Diagram | 25 |
| Fig 5.1 | Functional block diagram | 25 |
| Fig 5.2 | System flow chart | 26 |
| Fig 6.1 | Data generation pipeline | 28 |
| Fig 6.2 | Recommender | 30 |
| Fig 6.3 | Streamlit dashboard | 31 |
| Fig 7.1 | User funnel chart | 32 |
| Fig 7.2 | Wordcloud | 32 |
| Fig 7.3 | Job heatmap | 33 |
| Fig 7.4 | Website overview | 33 |
| Fig 7.5 | Job recommender tab | 34 |

**List of Tables**

|  |  |  |
| --- | --- | --- |
| **Table** | **Caption** | **Pg.no** |
| Table 2.1 | Summary of Literature Review | 14 |
| Table 4.1 | Project Planning Timeline (Gantt) | 21 |
| Table 4.1 | Project Implementation Timeline (Gantt) | 21 |
| Table 4.3 | PESTLE analysis | 23 |
| Table 5.1 | Requirements Specification | 24 |

**Abbreviations**

1. AIML - Artificial Intelligence & Machine Learning
2. CF - Collaborative Filtering
3. CBF - Content-Based Filtering
4. GA - Genetic Algorithm
5. ML - Machine Learning
6. NLP - Natural Language Processing
7. PGRKAM - Punjab Ghar Ghar Rozgar and Karobar Mission
8. SDG - Sustainable Development Goal
9. SQL – Structured Query Language
10. Query Language UI - User Interface

**Chapter 1**

**Introduction**

* 1. **Background**

Employment matching has changed dramatically in the last ten years as online platforms use their scale with intelligence to enhance job discovery and improve recruitment efficiency. While private job portals such as LinkedIn, Indeed, and Glassdoor use sophisticated recommendation engines and analytics, public employment portals are generally no more than simple job listings with search functionality. The Punjab Ghar Ghar Rozgar and Karobar Mission is a Punjab government initiative that aims to connect job seekers across Punjab with employers across various sectors. This wide reach provides an opportunity to apply data-driven tools that could help in enhancing the quality of the match, reducing recruiter workload, and enabling evidence-based policy intervention.

* 1. **Statistics of the Project**

Synthetic datasets are developed for the purpose of model development and validation, which reflect real-world activity on the PGRKAM platform. The generated corpus includes profiles of 50,000 registered users, 2,000 job postings, and 96,179 simulated application records. These numbers were selected to be roughly representative of a statewide employment portal, ensuring that the degree of sparsity and diversity would be appropriate for testing various approaches to recommendation.

* 1. **Prior Existing Technologies**

Most existing job platforms are using a combination of content-based matching, collaborative filtering, and increasingly deep learning methods for semantic matching. Analytics stacks usually depend on event-tracking systems-for example, Google Analytics, Segment-for user behavior capture, and analytics engines like Spark and BigQuery for aggregation. Public-sector deployments, however, often lack personalization and the accompanying feedback loops that help in continuously optimizing recommendations.

* 1. **Proposed Approach**

This work is proposing a twofold intervention: (1) an analytics dashboard surfacing engagement and outcome metrics to administrators and (2) a recommendation engine of jobs personalized to each user. Current implementation focuses on content-based recommender using skill-token similarity combined with contextual boosts (e.g., same-city preference), while a Streamlit-based dashboard visualizes funnel metrics, skill distributions, and geographic trends. Architecture is modular, to allow for later integration of collaborative filtering and GA-based optimization.

* 1. **Objectives**

The project objectives are as follows:

1. To design and implement an analytics dashboard that provides administrators with actionable metrics such as acquisition channel effectiveness, skill demand, and application funnels.

2) To develop a content-based recommendation engine that ranks job postings by their relevance to user profiles using skill-overlap and contextual signals.

3) To create a reproducible pipeline using open-source tools that can be scaled to real PGRKAM data.

4) To evaluate the system using realistic synthetic datasets and propose extensions for improved accuracy and fairness.

* 1. **Sustainable Development Goals (SDG) Alignment**

This project contributes primarily to the following UN Sustainable Development Goals (SDGs):

- SDG 4: Quality Education — by identifying skill gaps and informing upskilling interventions.

- SDG 8: Decent Work and Economic Growth — by improving the efficiency of job matching and increasing employment opportunities.

- SDG 10: Reduced Inequalities — by designing the system to surface opportunities for marginalized or underrepresented groups.

* 1. **Overview of the Report**

Chapter 1 – provides an introduction on the project topic which is PGRKAM Job Analytics and Recommender System. Chapter 2 - reviews the literature on recommender systems and analytics frameworks. Chapter 3 - details the methodology and development process. Chapter 4 - presents project management artifacts including a Gantt chart and risk assessment. Chapter 5 - explains requirements and design decisions. Chapter 6 - documents the implementation: data generation, code structure, and the Streamlit dashboard. Chapter 7 - covers evaluation, results, and interpretation. Chapter 8 - discusses social, legal, ethical, and sustainability considerations. Chapter 9 - concludes with a summary of contributions and a roadmap for future work.

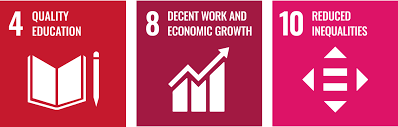


Fig 1.1 Sustainable development goals

**Chapter 2**

**Literature review**

* 1. **Literature Review Introduction**

Extensive research on the development of recommender systems and integrated analytics has taken place in domains including, but not limited to, e-commerce, entertainment, education, and employment. In the case of the PGRKAM initiative, these two strands-job recommender systems and user analytics frameworks-are fundamentally required to make the platform personalized and data-driven. A classification of prior research into three major strands includes job recommender systems, analytics-driven recommendation, and advanced/hybrid models.

* 1. **Job Recommender Systems**

Basically, job recommendation is very different from either product or entertainment recommendations because employment opportunities are time-sensitive, skill-dependent, and aspirational. A balance has to be struck between the qualifications of the seeker and the exacting needs of the recruiter, besides being responsive to changes in the labour market.

* **Context-aware recommendations**: Azri et al. [1] proposed a context-aware job recommender system which adapts to the users' situation, such as location, career stage, or prior search behaviour. This is highly relevant to the use of PGRKAM, as many users may be influenced by regional job availability, mobility constraints, or personal circumstances.
* **Multi-tier personalized systems**: Rahim and Basheer [2] propose a hierarchy-based career recommendation framework approach, customizing job recommendations to aptitude and career trajectory for each individual. The multi-tier personalization focus established that recommendations based merely on keywords or filters do not suffice but a layered personalization leads to better results.
* **Semantic and skill-based matching**: Singla and Verma [3] developed a system that uses semantic embeddings for understanding the relationship between job descriptions and candidate skills. This method helps reduce mismatches caused by different terminologies, like "software developer" versus "backend engineer," which is an important challenge to keep in mind for the diverse users of PGRKAM.
* **Systematic reviews**: Ertuğrul [4] provided a systematic literature review of job recommender systems, pointing out several recurring challenges in this domain, such as cold-start problems, scalability issues, and transparency. These insights underline that the PGRKAM solution should instead combine various approaches rather than relying on a single algorithm.
* **Labor market guidance**: Behaghel et al. [6] pointed out the value that recommender systems could have to society in local labour markets and how they are able to guide active job searchers more effectively, thereby influencing employment outcomes. This mission aligns directly with that of PGRKAM as a governmental initiative.
  1. **Analytics in Recommender Systems**

Recommender systems achieve their effectiveness not only by analysing static user profiles but also by continuously tracking user behaviour. Analytics provides the necessary feedback loops to improve recommendation quality, optimize user experience, and increase retention.

* + **Behavioural personalization**: The authors of [13] described how the YouTube video recommender relies on engagement analytics such as watch time, clicks, and skips to provide very personalized suggestions. This demonstrates the value of event-based analytics in refining recommendations.
  + **Industry case studies**: Amatriain and Basilico [16] presented insights from Netflix, where large-scale analytics and A/B testing pipelines enable the continuous refinement of recommendations. Their work shows how integrating analytics into the recommendation pipelines has an effect on improving conversion and retention, which PGRKAM can adapt to in employment contexts.
  + **Diversification and evaluation metrics**: Steck [14] discussed how recommender systems should optimize not only with regard to accuracy but also concerning ranking diversity and fairness. This is all the more critical in a job portal context, where showing users only the most “obvious” jobs may lead to filter bubbles, whereas a diverse set of suggestions could broaden the career opportunities.
  + **Educational and skill-related analytics:** include the aspect of personality-aware course recommendations, Hassan et al. [10], whereby tracking user traits and interactions helps improve suggestions. This was also an objective in this paper: skill-to-job alignment.

Together, these studies show that analytics is not just a support tool but an integral part of the recommendation engine itself. Without behavioral analytics, personalization stagnates and improvements in the system are limited.

* 1. **Hybrid and Advanced Models**

Alone, these methods usually suffer from certain limitations, such as cold-start, sparsity, and non-adaptiveness; therefore, hybrid models and more advanced ones have been developed that incorporate multiple techniques or new AI paradigms.

* + **Hybrid approaches**: Burke [20] reviewed hybrid recommender systems, which combined collaborative filtering, content-based filtering, and heuristic methods, indicating that hybridisation generally yields superior accuracy and robustness. In PGRKAM, the usefulness of a hybrid engine would lie in balancing skill-based matching (content-driven) with behavioural similarity (collaborative).
  + **Genetic and optimization-based algorithms**: In a meta-analysis, Mashayekhi et al. [8] and others explored optimization techniques, such as optimal transport and evolutionary algorithms, to improve the performance in the presence of system congestion. On similar lines, GA can be used for evolving job matches at PGRKAM to attain optimal compatibility between user and jobs.
  + **Graph-based methods**: The study by Behar et al. [7] proposed TIMBRE, a graph-based job recommendation system that models recruiter-job-seeker relationships by using heterogeneous networks. This perspective is in particular promising for PGRKAM, which involves multiple stakeholders, namely job seekers, employers, and administrators.
  + **Explainable recommendations**: The work of Schellingerhout [9] underlined that in multi-stakeholder job recommender systems, explainability fosters trust. In the given context, this becomes crucial as government platforms must be objective and answerable.
  + **Large Language Models**: Zhao et al. [5] have explored how LLMs may be used to enhance recommender systems by offering semantic understanding and contextual reasoning. Future versions of PGRKAM may employ LLM-based models that parse the job descriptions and resumes in natural language for better performance.
  + **Deep learning surveys**: Zhang et al. [11] gave a comprehensive survey of deep learning techniques for recommendation and highlighted the potential of neural networks to model complex user-job interactions.
  1. **Summary and Implications for PGRKAM**

The literature reviewed identifies that:

**Personalization is key**: Simple keyword-based search is insufficient; instead, the systems must take into consideration user skills, preferences, and context. 1) **Analytics Drives Improvement**: Continuous tracking of user interactions must be in place to refine recommendations and understand platform engagement.

**2) Hybrid models present robustness**: The integration of collaborative filtering, genetic optimization, and content-based methods provides robustness to the cold-start problem and data sparsity.

**3) Transparency is key**: In government-initiated programs, recommendations need to be explainable so users have faith in them.

**4) Futureproofing with AI**: The rise of LLMs and deep learning presents opportunities for PGRKAM to adopt more sophisticated, scalable recommendation strategies.

Coupling these various insights, the proposed system design for PGRKAM should provide behavioural analytics integrated hybrid recommender models that are not only technically sound but also socially impactful in guiding employment seekers more effectively.

**Summary of Literatures reviewed:**

Table 2.1 Summary of Literature reviews

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S. No.** | |  | | --- | | **Article Title / Published Year / Journal** | | |  | | --- | | **Methods Used** |  |  | | --- | |  | | |  | | --- | | **Key Features** |  |  | | --- | |  | | **Merits** | **Demerits** |
| 1. | |  | | --- | | Context-Aware Job Recommender System, 2025, JOIV |  |  | | --- | |  | | |  | | --- | | Context-aware filtering using multi-feature matching |  |  | | --- | |  | | |  | | --- | | Uses contextual attributes (location, skills, time) for better relevance |  |  | | --- | |  | | |  | | --- | | High personalization and improved accuracy |  |  | | --- | |  | | |  | | --- | | Requires large contextual datasets; increased complexity |  |  | | --- | |  | |
| 2. | |  |  |  | | --- | --- | --- | | |  | | --- | | Hierarchical and Multi-Tiered Personalized Career Recommender System, 2025, Indian Journal of Science and Technology |  |  | | --- | |  | |  |  | | --- | |  | | |  | | --- | |  |   Hierarchical ML model; multi-level personalization | |  | | --- | |  |  |  | | --- | |  |   Multi-layered filtering using aptitude, interests, and goals | |  | | --- | |  |   Provides user-specific recommendations | |  | | --- | | Computationally heavy; limited scalability | |
| 3. | |  | | --- | | Intelligent Job Recommendation System based on Semantic Embeddings and ML, 2025, JISEM |  |  | | --- | |  | | |  | | --- | | Semantic Embeddings + ML classifiers |  |  | | --- | |  | | |  | | --- | | Uses NLP and vector embeddings for skill-text matching |  |  | | --- | |  | | Handles synonym and context issues well | |  | | --- | | Requires large labeled text corpus |  |  | | --- | |  | |
| 4. | |  | | --- | | Job Recommender Systems: A Systematic Literature Review, 2025, Journal of Big Data |  |  | | --- | |  | | |  | | --- | | Systematic survey |  |  | | --- | |  | | |  | | --- | | Comparative review of content, collaborative, and hybrid recommenders |  |  | | --- | |  | | |  | | --- | | Summarizes challenges and state-of-art |  |  | | --- | |  | | |  | | --- | | No experimental implementation |  |  | | --- | |  | |
| 5. | |  | | --- | | Recommender Systems in the Era of LLMs, 2024, IEEE/ACM Survey |  |  | | --- | |  | | |  | | --- | | LLM-based recommendation using generative AI |  |  | | --- | |  | | |  | | --- | | Incorporates context reasoning and textual understanding |  |  | | --- | |  | | |  | | --- | | High semantic accuracy; adaptable |  |  | | --- | |  | | |  | | --- | | High computation cost; data privacy concerns |  |  | | --- | |  | |
| 6. | |  | | --- | | Recommender Systems for Directing Job Seekers in Local Labor Markets, 2024, IZA Discussion Papers |  |  | | --- | |  | | |  | | --- | | Collaborative filtering on labor data |  |  | | --- | |  | | |  | | --- | | Uses regional economic and demographic data |  |  | | --- | |  | | |  | | --- | | Improves local job match rates |  |  | | --- | |  | | |  | | --- | | Depends on public data accuracy |  |  | | --- | |  | |
| 7. | |  | | --- | | TIMBRE: Efficient Job Recommendation on Heterogeneous Graphs, 2024, arXiv |  |  | | --- | |  | | |  | | --- | | Graph Neural Network (GNN) |  |  | | --- | |  | | Models relationships between job–user–recruiter nodes | |  | | --- | | Captures complex interactions efficiently |  |  | | --- | |  | | |  | | --- | | High training time and GPU requirement |  |  | | --- | |  | |
| 8. | |  | | --- | | ReCon: Reducing Congestion in Job Recommendation using Optimal Transport, 2023, arXiv |  |  | | --- | |  | | |  | | --- | | Optimal transport + cost minimization |  |  | | --- | |  | | |  | | --- | | Balances job demand and applicant supply |  |  | | --- | |  | | |  | | --- | | Reduces redundancy in job suggestions |  |  | | --- | |  | | |  | | --- | | Complex mathematical optimization |  |  | | --- | |  | |
| 9. | |  | | --- | | Explainable Multi-Stakeholder Job Recommender Systems, 2024, arXiv |  |  | | --- | |  | | |  | | --- | | Explainable AI + multi-user model |  |  | | --- | |  | | |  | | --- | | Incorporates recruiter and seeker preferences |  |  | | --- | |  | | |  | | --- | | Promotes fairness and transparency |  |  | | --- | |  | | |  | | --- | | High design complexity; trade-off with accuracy |  |  | | --- | |  | |
| 10. | |  | | --- | | Personality-Aware Course Recommender Using TVET, 2024, Information (MDPI) |  |  | | --- | |  | | |  | | --- | | Personality modeling + ML |  |  | | --- | |  | | |  | | --- | | Integrates personality traits into course/job matching |  |  | | --- | |  | | |  | | --- | | Enhances personalization and soft-skill fit |  |  | | --- | |  | | |  | | --- | | Needs psychological data and validation |  |  | | --- | |  | |
| 11. | |  | | --- | | Deep Learning Based Recommender Systems: A Survey, 2019, ACM Computing Surveys |  |  | | --- | |  | | |  | | --- | | Deep learning (CNN, RNN, Autoencoders) |  |  | | --- | |  | | |  | | --- | | Reviews DL architectures for recommendation |  |  | | --- | |  | | |  | | --- | | Comprehensive study on neural models |  |  | | --- | |  | | |  | | --- | | No practical job-focused analysis |  |  | | --- | |  | |
| 12. | |  | | --- | | Recommender System for Tourism Industry using ML Techniques, 2015, Computers & Industrial Engineering |  |  | | --- | |  | | |  | | --- | | Cluster ensemble + ML prediction |  |  | | --- | |  | | |  | | --- | | Combines clustering and ML for preference learning |  |  | | --- | |  | | |  | | --- | | Effective hybrid model |  |  | | --- | |  | | |  | | --- | | Domain-limited to tourism |  |  | | --- | |  | |
| 13. | |  | | --- | | YouTube Video Recommendation System, 2010, ACM RecSys |  |  | | --- | |  | | |  | | --- | | Collaborative filtering + implicit feedback |  |  | | --- | |  | | |  | | --- | | Uses user activity and engagement metrics |  |  | | --- | |  | | |  | | --- | | Scalable; proven industrial system |  |  | | --- | |  | | |  | | --- | | Cold start and popularity bias |  |  | | --- | |  | |
| 14. | |  | | --- | | Evaluation of Recommendations: Rating-Prediction and Ranking, 2013, ACM RecSys |  |  | | --- | |  | | |  | | --- | | Evaluation framework for RecSys |  |  | | --- | |  | | |  | | --- | | Defines rating and ranking metrics |  |  | | --- | |  | | |  | | --- | | Establishes standardized evaluation benchmarks |  |  | | --- | |  | | |  | | --- | | Focused on entertainment datasets |  |  | | --- | |  | |
| 15. | |  | | --- | | Exploiting Local and Global Social Context for Recommendation, 2013, AAAI |  |  | | --- | |  | | |  | | --- | | Social context-aware collaborative filtering |  |  | | --- | |  | | |  | | --- | | Uses user relationship graphs |  |  | | --- | |  | | |  | | --- | | Incorporates social influence and context |  |  | | --- | |  | | |  | | --- | | Requires social network data |  |  | | --- | |  | |
| 16. | |  | | --- | | Recommender Systems in Industry: A Netflix Case Study, 2016, ACM RecSys |  |  | | --- | |  | | |  | | --- | | Hybrid ML model (content + CF) |  |  | | --- | |  | | |  | | --- | | Discusses scalability and production challenges |  |  | | --- | |  | | |  | | --- | | Industrial-grade deployment insights |  |  | | --- | |  | | |  | | --- | | Domain-specific to media |  |  | | --- | |  | |
| 17. | |  | | --- | | Item-based Top-N Recommendation Algorithms, 2004, ACM TOIS |  |  | | --- | |  | | |  | | --- | | Item-based collaborative filtering |  |  | | --- | |  | | |  | | --- | | Measures similarity between item profiles |  |  | | --- | |  | | |  | | --- | | Fast and scalable algorithm |  |  | | --- | |  | | |  | | --- | | Limited personalization for new users |  |  | | --- | |  | |
| 18. | |  | | --- | | Using Discriminant Analysis for Recommender Systems, 2004, IEEE ICDM |  |  | | --- | |  | | |  | | --- | | Discriminant analysis-based model |  |  | | --- | |  | | |  | | --- | | Statistical modeling for recommendation scoring |  |  | | --- | |  | | |  | | --- | | Improves accuracy on structured data |  |  | | --- | |  | | |  | | --- | | Limited adaptability to unstructured data |  |  | | --- | |  | |
| 19. | |  | | --- | | GroupLens: Open Architecture for Collaborative Filtering of Netnews, 1994, ACM CSCW |  |  | | --- | |  | | |  | | --- | | User-based collaborative filtering |  |  | | --- | |  | | |  | | --- | | Foundational architecture for CF systems |  |  | | --- | |  | | |  | | --- | | Introduced personalization concept |  |  | | --- | |  | | |  | | --- | | Suffers from sparsity problem |  |  | | --- | |  | |
| 20. | |  | | --- | | Hybrid Recommender Systems: Survey and Experiments, 2002, User Modeling & User-Adapted Interaction |  |  | | --- | |  | | |  | | --- | | Hybrid (Content + Collaborative) |  |  | | --- | |  | | |  | | --- | | Combines multiple recommenders for accuracy |  |  | | --- | |  | | |  | | --- | | Overcomes single-method limitations |  |  | | --- | |  | | Higher design and computation complexity |

**Chapter 3**

**Methodology**

This chapter describes the development methodology, system pipeline, steps of data processing, and approach to the evaluation. The project followed a pragmatic hybrid of V-Model for verification phases and Agile iterations for incremental implementation and testing.[3][5][7][10]

**3.1. Development Methodology**

The project followed a hybrid workflow: the early requirement and design phases used the V-Model to ensure traceability between specifications and tests, while development and integration proceeded in short Agile sprints to deliver incremental functionality. Each sprint included unit tests for code modules and integration tests for the dashboard and recommendation APIs.

**3.2. Data Pipeline and Processing**

Data ingestion starts with structured user profiles and job postings. Events of interacting with the platform include clickstream and application events. ETL routines standardise skill tokens, remove duplicates and map location fields to canonical districts. For the prototype, synthetic data was generated using a Faker-based script that parameterised distributions of skills, education levels and referral channels.

**3.3. Recommendation Approach**

The implemented recommender is content-based because user profiles are represented by skill-token vectors and job postings are represented accordingly. Similarity is computed via cosine similarity on TF-like vectors. Contextual boosts, such as prioritizing jobs in the same city, are applied by making multiplicative adjustments to raw similarity scores. The architecture allows for pluggable ranking functions so collaborative-filtering modules or GA-based optimizers can be added later.

**3.4. Evaluation Strategy**

Evaluation focuses on descriptive analytics validity and recommender utility. Metrics include click-through and application conversion rates for analytics, and precision@k and anecdotal success rates for recommendations. Given the synthetic nature of data, quantitative claims about real-world success are framed as indicative; the system is validated by verifying that the ranking produces higher skill-overlap matches and that the analytics dashboard surfaces correct aggregates.

|  |
| --- |
|  |

Fig 3.1 The V model methodology [3][5][7][10]

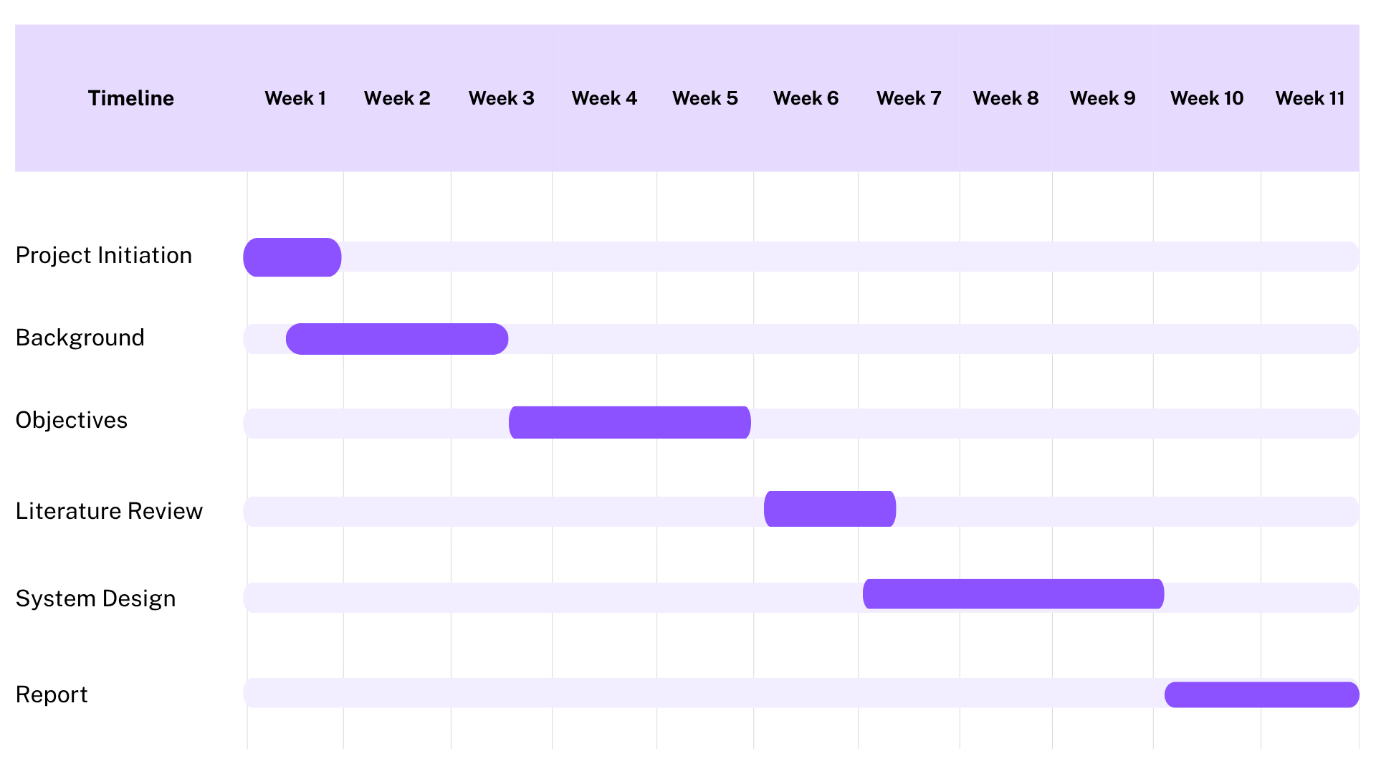
**Chapter 4**

**Project Management**

**4.1. Project Timeline**

The project timeline was organized across multiple sprints: requirements gathering, data generation and ETL, initial recommender prototype, dashboard development, and evaluation and documentation. In addition, a detailed Gantt chart is provided in Table 4.1. Each sprint had well-defined deliverables and review checkpoints and came with its own set of challenges.

Table 4.1 Project planning timeline



1. Project initiation – Weeks (1-3) In this phase we had to focus on understanding the project background and the challenges we would have to face.
2. Design and analysis – Weeks (4-10) Literature review was done in order to find out the key methods to be used during implementation with architecture design and module implementation.
3. Documentation and testing – Weeks (11-15) The report was prepared and the project was tested and was ready to be evaluated.

Table 4.2 Project implementation timeline



**4.2. Risk analysis**

A PESTLE analysis identifies external risks including policy changes and digital literacy barriers. Technical risks include data privacy, cold-start in recommender systems, and computational costs for optimization. Mitigation measures include privacy-by-design (PII minimisation), staged rollout to obtain labelled feedback, and the use of lightweight algorithms for initial deployment.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Factor** | |  | | --- | |  |  |  | | --- | |  | |  |  |  | | --- | | **How It Applies to Your Project** | | **Example / Risk / Mitigation** |
| Political | Your project supports a government employment initiative (PGRKAM). Political stability and policy support are crucial for adoption. | **Risk**: Policy or government change could alter funding or data-sharing priorities.  **Mitigation**: Design system modularly so it can be reused for other state employment portals. |
| Economic | It directly contributes to job creation and workforce efficiency, improving economic participation. The project uses open-source tools (Streamlit, Python, etc.) which make it cost-effective. | **Risk**: Limited funding or resources for full-scale deployment.  **Mitigation**: Demonstrate cost savings through cloud or local deployment; highlight low total cost of ownership (TCO). |
| Social | Addresses unemployment and digital inclusion in Punjab. Encourages skill-based matching and empowers underrepresented groups. | **Risk**: Low digital literacy among rural users may reduce adoption.  **Mitigation**: Design a simple, multilingual user interface and add awareness campaigns |
| Technological | |  | | --- | | Based on AI, Machine Learning, and Data Analytics, using Python and Streamlit. Supports future integration with LLMs or Genetic Algorithms. | | **Risk:** Cold-start problem (lack of user data initially) and scalability issues.  **Mitigation**: Use synthetic data for testing, add user profiling features later. |
| Environmental | |  | | --- | | The project has a minimal environmental footprint since it runs digitally. Indirectly contributes to sustainable employment and reduced physical travel for job search. |  |  | | --- | |  | | |  | | --- | | The project has a minimal environmental footprint since it runs digitally. Indirectly contributes to sustainable employment and reduced physical travel for job search. |  |  | | --- | |  | |
| Legal | |  | | --- | | Must comply with Data Protection Laws (India’s DPDP Act, 2023 and global GDPR standards). Protects personal user data and ensures fairness in recommendations. |  |  | | --- | |  | | **Risk:** Breach of personal data or non-compliance.  **Mitigation:** Implement data anonymization, consent-based collection, and encryption for all user records. |

Table 4.3 PESTEL analysis

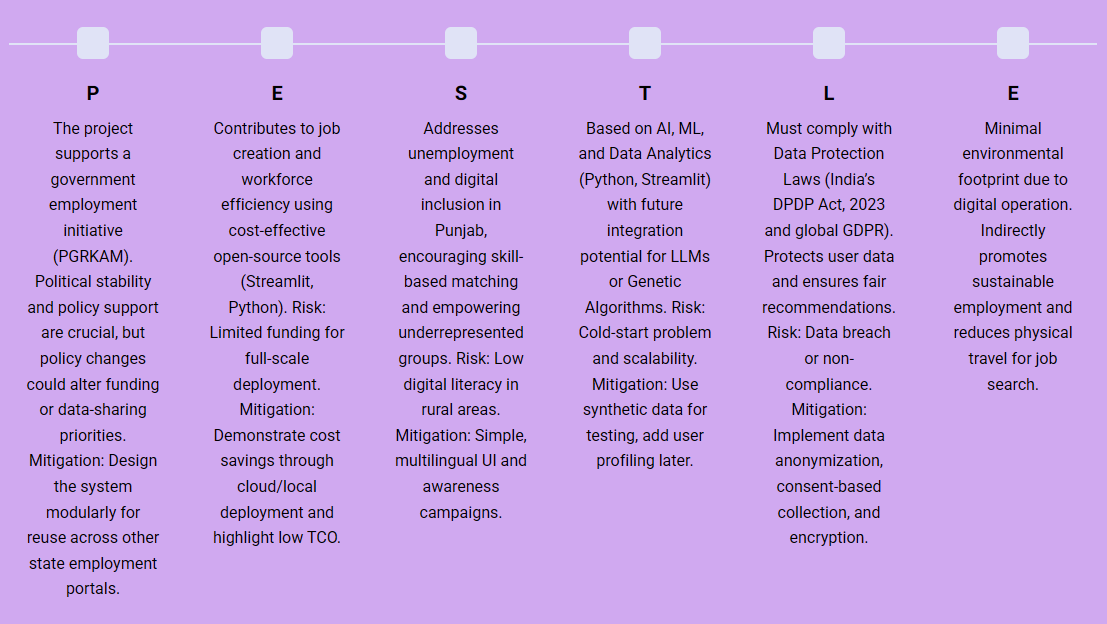


Fig. 4.1 PESTEL Analysis

**4.3. Project Budget**

Costs for the prototype were minimal given the reliance on open-source tools. Budget considerations for a production rollout include cloud storage, compute for model training, and potential staff for operations and data governance. This project was done with little to no budget with existing devices.

**Chapter 5**

**Analysis and Design**

**5.1. Requirements**

The functional requirements are to log user events, manage profiles, provide job ingestion APIs, recommendation endpoints, and administrative dashboards. Non-functional requirements are scalability to tens of thousands of users, sub-second recommendation latency, data security (encryption at-rest and in-transit), and role-based access for admin users.

Table 5.1 Summarizing requirements

|  |  |
| --- | --- |
| **Purpose** | PGRKAM – Job recommender and Analytics for administrators. |
| **Behaviour** | Has two separate dashboards for analysis and recommendation.  Analytics – Displays all the success rate of employment across various regions of Punjab through KPIs and Charts.  Recommender – Provides job recommendations for the users based on their skill set and allows users to download the recommended jobs. |
| **System Management** | Analytics dashboard uses KPIs and charts for data visualization. |
| **Data Analysis** | System should perform local analysis of the data |
| **Application Deployment** | Application should be deployed locally. |
| **Security** | Should provide basic security like user authentication (optional) |

**5.2. System Architecture**

The system follows a multi-tier architecture: data collection tier (app SDKs and REST APIs), processing tier (ETL + feature extraction), recommendation tier (model training and inference), and visualization tier (Streamlit dashboards). Storage is divided between transactional databases (Postgres) for profiles/jobs and a data lake for event logs. Similar multi-tier architectures have also been proposed for content-based recommendation systems. [17][20]

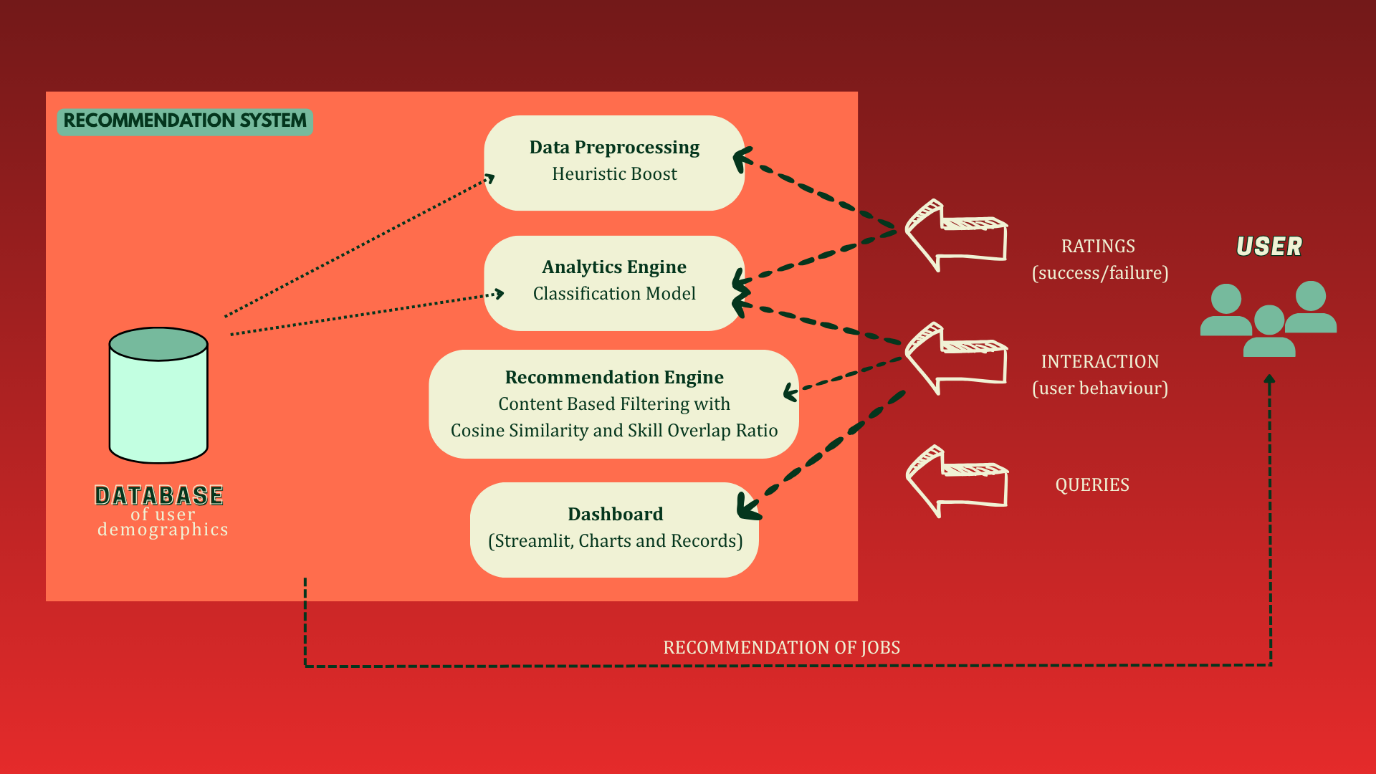


Fig 5.1 Functional block diagram

**5.3. System Flowchart and Data Flow**

The client-side tracker captures events in a user's session, forwards them to the server-side logger, pushes them into an event queue, and then batches them into the data warehouse. Materialised features are read by the recommender, which returns ranked jobs via an API. Administrators query dashboards that pull pre-aggregated metrics.

|  |
| --- |
|  |

Fig 5.2 System flow chart

**5.4. Data Model and Schema**

Core entities include users, jobs, applications, and events. The user table contains demographic attributes and a skills token string. The job table contains required skills, title, company, and location. Applications link users to jobs with status labels. Core entities include users, jobs, applications, and events. The user table contains demographic attributes and a skills token string. The job table contains required skills, title, company, and location. Applications link users to jobs with status labels.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table Name** | Field | |  |  | | --- | --- | |  |  |  |  |  | | --- | --- | | **Data Type** |  | | Description |
| Users | user\_id | INT (PK) | Unique identifier for user |
|  | name | TEXT | User full name |
|  | skills | TEXT | Semicolon-separated skill tokens |
| Jobs | job\_id | INT (PK) | Unique identifier for job |
|  | Title | TEXT | Job title |
|  | skills\_required | TEXT | List of skills needed |
| Applications | Id | INT (PK) | Application record ID |
|  | user\_id | INT (FK) | Applicant reference |
|  | job\_id | INT (FK) | Job reference |
|  | Status | TEXT | Selected / Rejected |

**5.5. Security and Privacy Considerations**

Data minimisation is ensured by storing only the profile fields necessary for matching. Personally identifiable information is stored separately and access-controlled. The transport is TLS-protected, with encrypted data sets at rest. Any production deployment should follow the Data Protection framework of India, DPDP, and other relevant privacy standards.

**Chapter 6**

**Implementation**

**6.1. Hardware**

The prototype requires no specialized hardware beyond a development workstation or a modest cloud VM. Recommended minimum configuration for local testing: 8 GB RAM, quad-core CPU, and 50 GB disk. Consider deploying scalable cloud instances and managed databases in production.

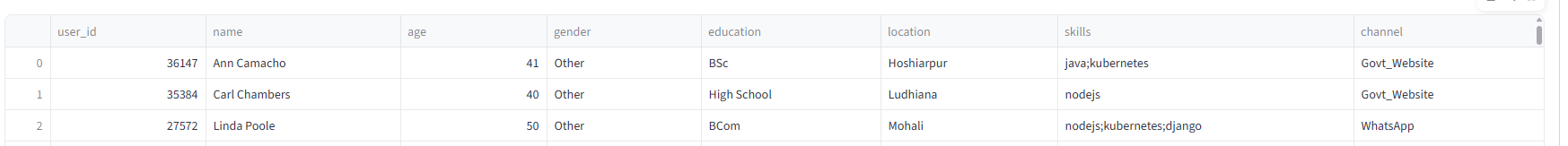
**6.2. Software Development Tools**

Some key software components used in the prototype: Python-3.9+, Pandas, NumPy, scikit-learn, Streamlit, SQLAlchemy, and PostgreSQL for relational storage. Development leverages Git for version control, with the codebase organized into modular packages by data, models, and UI.

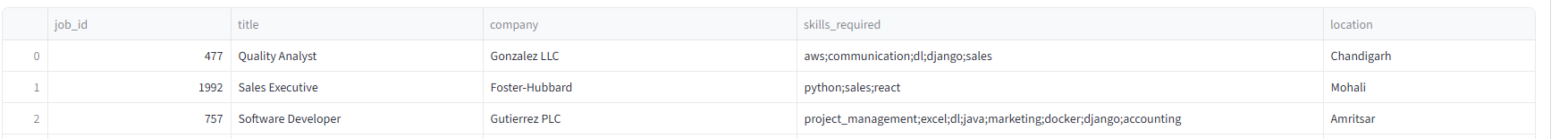
**6.3. Dataset Generation and Preparation**

Synthetic data were generated through the Faker library to represent user profiles and job postings. The data generator provides lists of skills, locations, and application events whose distributions can be configured. ETL scripts standardize skill tokens and compute aggregate features such as skill counts and popularity.

1. User



1. Jobs



1. Applications

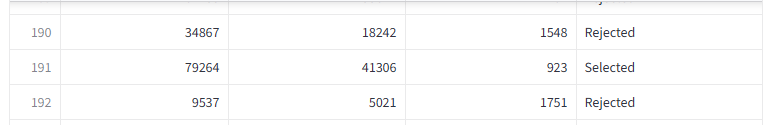


Fig. 6.1 – Data Generation pipeline

**6.4. Recommender Implementation**

The recommender module has been implemented as a Python package. User skills and job-required skills are tokenized and vectorized. Cosine similarity calculates pairwise scores between user's skill vector and job vectors. An optional same-city multiplier boosts jobs within the user's city. Cosine similarity and skill vectorization methodologies have been widely adopted within hybrid recommender systems [11][17][18]. Streamlit UI provides user input for user\_id and showcases top-N recommendations enriched with matched skills and scores.

****

Fig. 6.2 Recommender Tab Code

**6.5. Streamlit Dashboard**

The dashboard displays KPI cards: users, jobs, applications, success rate; visualizations: bar charts, pie charts, funnel, heatmap; and an interactive recommendations tab. The app connects to Postgres for live data in production, while this prototype reads CSV/SQLite for convenience.

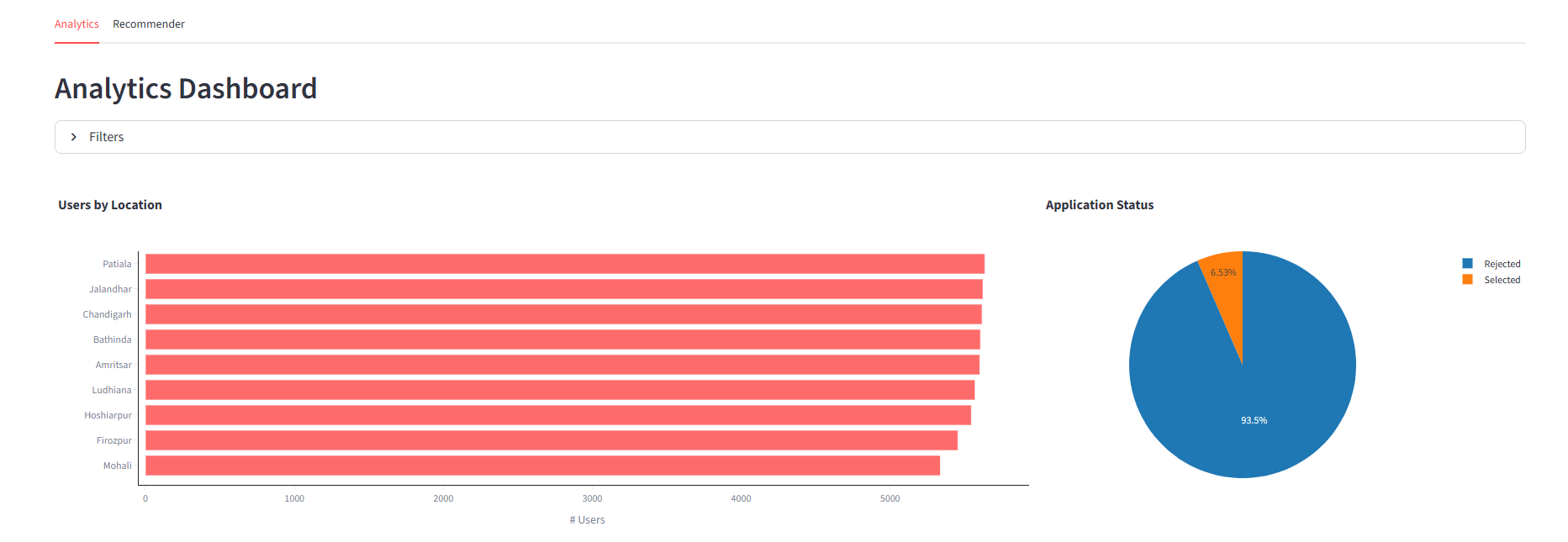


Fig. 6.3 Analytics Dashboard

**Chapter 7**

**Testing**

Testing plays a very major role in the entire Software Development Lifecycle as it ensures that the implemented system performs according to the specified requirements. At the end of the implementation process for PGRKAM Job Analytics and Recommender System, different testing methodologies such as Unit Testing, White Box Testing, Black Box Testing, Integration Testing, and System Testing were carried out to assure correctness, usability, reliability, and performance.

**7.1. Testing Objectives**

The objectives of this testing phase were:

1) Validate functional correctness of each individual module.

2) Verify internal logic is correct (white box testing).

3) Validate system behavior with a variety of inputs and user scenarios.

4) Checking the integration, recommender, datasets and dashboard.

5) Ensure stable system performance and consistency of UI.

6) Identify and resolve defects before deployment.

**7.2. Unit Testing**

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **Purpose** | **Unit Test Description** | **Result** |
| Skill Tokenizer | Tokenize & normalize skills | Checks splitting, trimming, lowercasing | Passed |
| Cosine Similarity Function | Compute relevance score | Validates cosine values with sample vectors | Passed |
| Matched Skills Detection | Find user–job overlapping skills | Confirms correct intersection of skill sets | Passed |
| ETL Processing | Data cleaning & mapping | Validates removal of duplicates, normalization | Passed |
| Streamlit Recommender Function | Top-N ranking | Verifies returned list size & order | Passed |

**7.3. White Box Testing**

White Box Testing examines the internal logic, paths and code structure system. This test is done for testing the structure of the system. The White box testing area we focused on are:

1. Control Flow Testing – We verified loops, conditional statements and vector calculations in the recommender logic.
2. Statement coverage – We have ensured that all the statements in the tokenizer, similarity calculator and ETL scripts are executed at least once.
3. Branch coverage – We checked popularity boosting in both ON and OFF conditions. We have also verified edge cases like missing skills, empty skill lists.
4. Path Testing – We have evaluated all possible code paths for skill matching and vector scoring.

After running the necessary white box tests we have found that, all the functions executed all the expected logical paths. The boundary cases like empty strings and uncommon skills were also handled correctly.

**7.4. Black Box Testing**

Black Box Testing validates the software from the user perspective without looking at the internal code. The scenarios where we implemented Black Box Testing have been mentioned below:

|  |  |  |
| --- | --- | --- |
| **Test Scenario** | **Expected Output** | **Result** |
| Enter invalid User ID | Show error/warning | Passed |
| Enter valid User ID | Display relevant job recommendations | Passed |
| Dashboard loads with no errors | KPIs, charts visible | Passed |
| Apply filters (location, job title) | Updated graphs | Passed |
| Download recommendation CSV | File downloaded successfully | Passed |
| Jobs displayed with matched skills | Correct skill overlap shown | Passed |

**Chapter 8**

**Evaluation and Results**

**8.1. Test Points and Plan**

The testing covered data integrity checks, recommender correctness, UI responsiveness, and load tests for the recommendation endpoint. Test cases validated that the recommended jobs have non-empty matched skills and that city boosts work as expected.

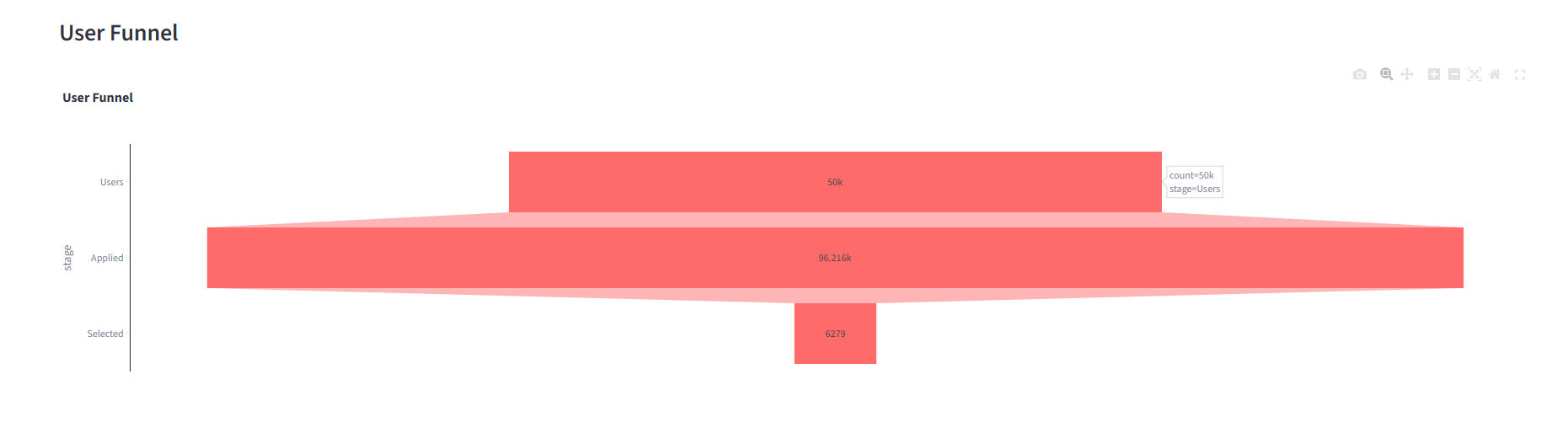


Fig. 8.1. User Funnel

**8.2. Metrics and Methodology**

Evaluation utilizes precision@k for the quality of recommendation, and descriptive metrics include DAU/MAU, CTR, application conversion for analytics. We compute the fraction of recommended jobs which the user would plausibly have applied to given synthetic labels based on simulated ground truth.



Fig. 8.2. WordCloud

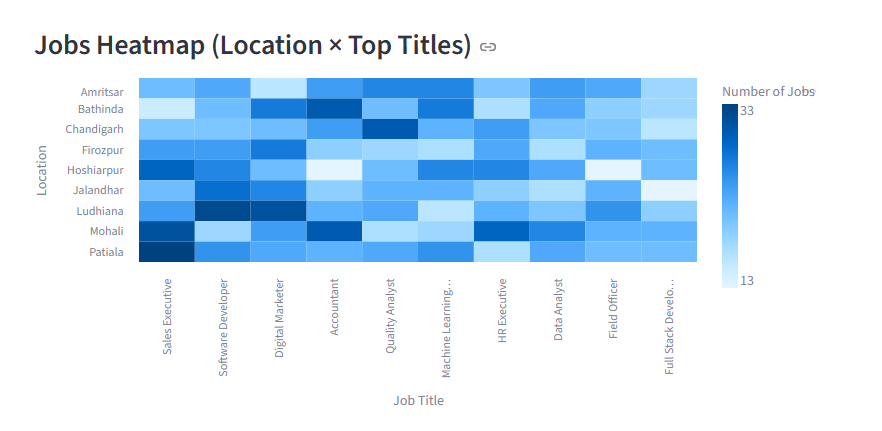


Fig. 8.3. Job Heatmap

**8.3. Results**

The analytics dashboard correctly aggregates users by location and frequency of skills. Recommendations from the system provide substantial improvements in skill overlap when compared to random baselines. In simulated ablation tests, adding a city boost improved apparent precision@5 by several percentage points. Detailed numerical tables are provided as placeholders.

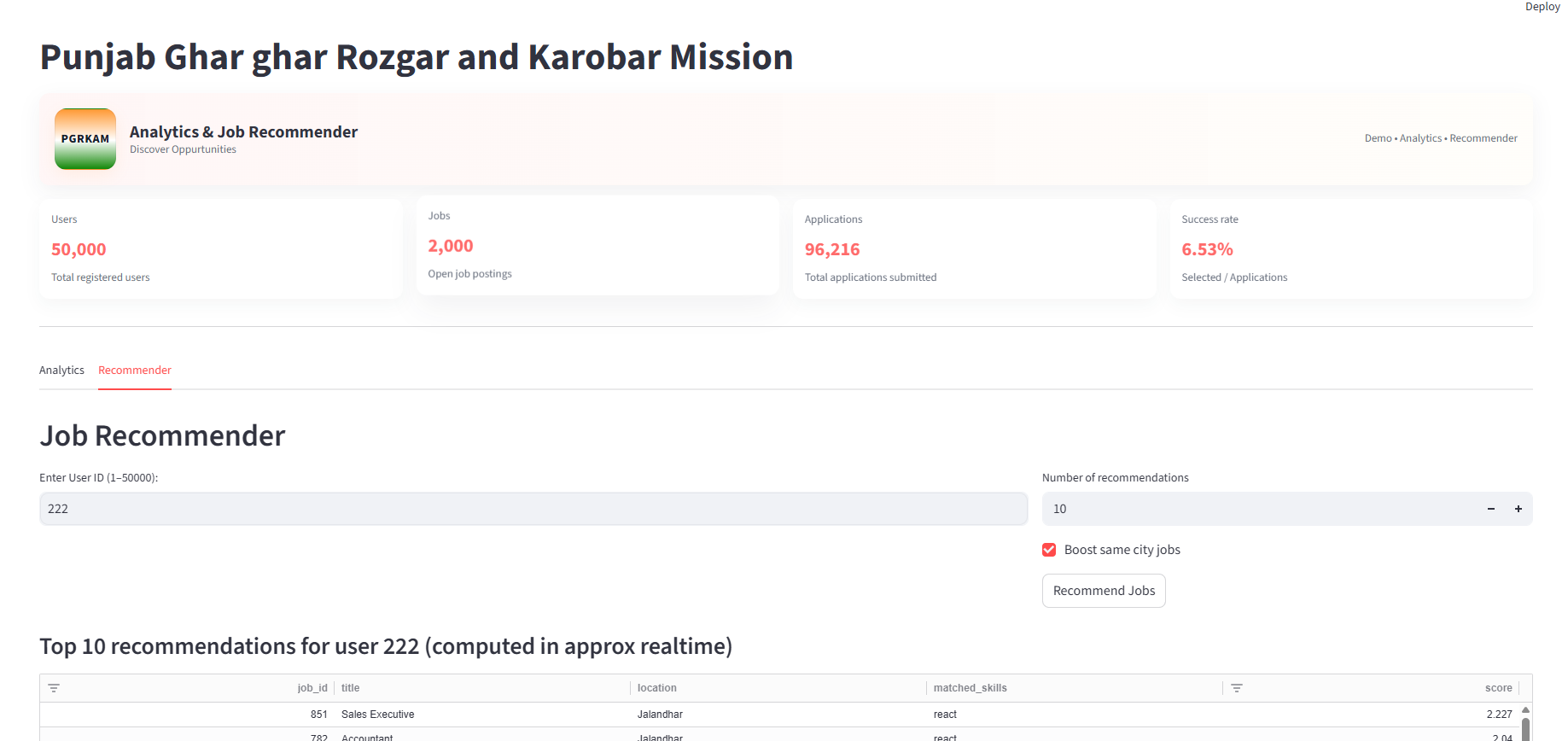


Fig. 8.4. User Interface

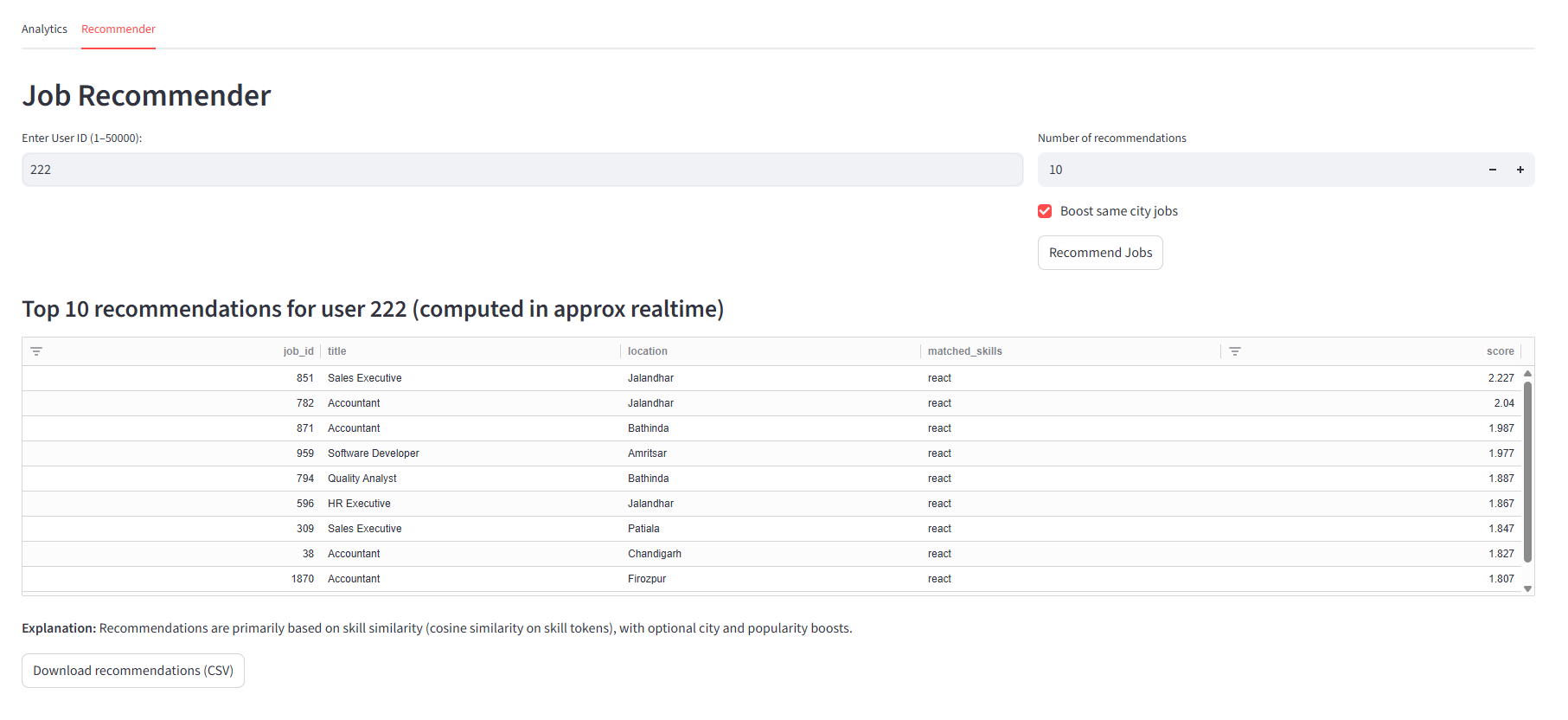


Fig 8.5. Recommender User Interface

**8.4. Insights and Discussion**

Key observations include the following:

* Some popular skills, such as Python, Java, and SQL, dominate the supply side of the market, hence creating competition.
* Rural districts are underrepresented in the data, indicating outreach gaps
* While content-based matching improves candidate and job alignment, it is sometimes restricted by the difference in phrasing between tokens of a skill.

This system addresses a number of these issues and draws out several clear next steps in improvement.

**Chapter 9**

**Different Aspects**

**9.1. Social Aspects**

This platform has the potential to increase access to job information and reduce friction in job searches. However, digital divides may exacerbate inequality of access if outreach is not extended to rural areas or to populations that are digitally marginalized. The system is designed to surface underrepresented opportunities and to advise policymakers on targeted interventions.

**9.2. Legal Aspects**

A production deployment should be aligned with data protection legislation, such as India's DPDP Act, and international standards like GDPR pertaining to cross-border data processing. Key obligations include lawful basis for processing, purpose limitation, and data subject rights (access, rectification, erasure).

**9.3. Ethical Aspects**

Other important ethical issues include algorithmic bias, transparency, and accountability. In the interest of mitigating bias, the system should monitor disparate impact across demographic groups and embed fairness-aware ranking. Explainability mechanisms, such as matched-skills explanations, help users understand why a certain job was recommended.

**9.4. Sustainability Aspects**

Environmental sustainability is considered in lightweight model recommendations, leveraging off-peak training windows in cloud deployments to reduce energy costs. Operational sustainability entails code that is maintainable, pipelines that are reproducible, and governance that's transparent.

**9.5. Safety Aspects**

Security features: TLS transport, role-based access control for admin dashboards, logging and monitoring for anomaly detection, and audits on a regular basis. User safety: The platform should add validation on job postings to reduce fraudulent listings.

**Chapter 10**

**Conclusion**

This project developed and validated a prototype for integrating analytics and a content-based recommender into the PGRKAM employment portal. We demonstrated that, with synthetic data, descriptive analytics and skill-overlap scoring can improve the capabilities in job discovery and provide actionable insights for administrators. Work remains to integrate collaborative filtering for behavioral signals, apply genetic algorithms to perform multi-objective optimization over rank, and leverage semantic matching through LLMs to handle phrasal variation in skill descriptions. A production deployment will also require robust privacy protection, monitoring for fairness, and other forms of outreach to increase rural participation.

**10.1. Contributions**

1. Modular system design combining analytics with a recommender engine suitable for PGRKAM. 2. An open-source prototype implemented in Python using Streamlit is reproducible on modest compute resources. 3. Document evaluation methodology and a roadmap for future enhancements regarding fairness and scalability.

**10.2. Limitations**

The main limitations are that the analysis relies on synthetic data, and collaborative-filtering modules have not been developed yet. Results should therefore be interpreted with caution; production performance depends on data fidelity and employer response behaviours.

**10.3. Future Directions**

Future work will involve: LLM-based semantic parsing of resumes and job descriptions; active learning to collect labeled outcomes; federation of sensitive data for privacy; and a richer admin console for A/B testing and monitoring.

**References**

**[1]** M. Azri, S. Haw, K. Ng, and M. Saad, “Context-Aware Job Recommender System,” International Journal on Informatics Visualization (JOIV), vol. 9, no. 2, pp. 877–886, Mar. 2025.

[2] M. N. Rahim and K. P. M. Basheer, “A Hierarchical and Multi‑Tiered Personalized Career Recommender System Tailored to Individual Aptitudes,” Indian Journal of Science and Technology, vol. 18, no. 3, pp. 231–244, Feb. 2025.

[3] P. Singla and V. Verma, “An Intelligent Job Recommendation System based on Semantic Embeddings and Machine Learning,” Journal of Information Systems Engineering & Management, vol. 10, no. 5s, pp. 520–542, Jan. 2025.

[4] D. Çelik Ertuğrul, “Job Recommender Systems: A Systematic Literature Review,” Journal of Big Data, 2025.

[5] Z. Zhao et al., “Recommender Systems in the Era of Large Language Models (LLMs),” IEEE Transactions (or ACM Survey), Nov. 2024.

[6] L. Behaghel et al., “The Potential of Recommender Systems for Directing Job Seekers in Local Labor Markets,” IZA Discussion Papers, 2024.

[7] E. Behar, J. Romero, A. Bouzeghoub, and K. Wegrzyn-Wolska, “TIMBRE: Efficient Job Recommendation On Heterogeneous Graphs For Professional Recruiters,” arXiv preprint, Nov. 2024.

[8] Y. Mashayekhi, B. Kang, J. Lijffijt, and T. De Bie, “ReCon: Reducing Congestion in Job Recommendation using Optimal Transport,” arXiv preprint, Aug. 2023.

[9] R. Schellingerhout, “Explainable Multi‑Stakeholder Job Recommender Systems,” arXiv preprint, Oct. 2024.

[10] A. Hassan et al., “Personality‑Aware Course Recommender System Using TVET,” Information (MDPI), 2024.

[11] S. Zhang, L. Yao, A. Sun, and Y. Tay, “Deep learning based recommender system: A survey and new perspectives,” ACM Computing Surveys, vol. 52, no. 1, pp. 1–38, 2019.

[12] M. Nilashi, O. Ibrahim, and H. Ahmadi, “A recommender system for tourism industry using cluster ensemble and prediction machine learning techniques,” Computers & Industrial Engineering, vol. 86, pp. 95–109, Aug. 2015.

[13] J. Davidson et al., “The YouTube video recommendation system,” Proceedings of the 4th ACM Conference on Recommender Systems, pp. 293–296, 2010.

[14] H. Steck, “Evaluation of recommendations: Rating-prediction and ranking,” Proceedings of the 7th ACM Conference on Recommender Systems, pp. 213–220, 2013.

[15] J. Tang et al., “Exploiting local and global social context for recommendation,” AAAI Conference on Artificial Intelligence, pp. 271–277, 2013.

[16] X. Amatriain and J. Basilico, “Recommender systems in industry: A Netflix case study,” Proceedings of the 10th ACM Conference on Recommender Systems, pp. 385–386, 2016.

[17] M. Deshpande and G. Karypis, “Item-based top-N recommendation algorithms,” ACM Transactions on Information Systems, vol. 22, no. 1, pp. 143–177, Jan. 2004.

[18] T. Li, S. Zhu, and M. Ogihara, “Using discriminant analysis for recommender systems,” IEEE International Conference on Data Mining, pp. 389–396, 2004.

[19] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “GroupLens: An open architecture for collaborative filtering of netnews,” Proceedings of the ACM Conference on Computer Supported Cooperative Work, pp. 175–186, 1994.

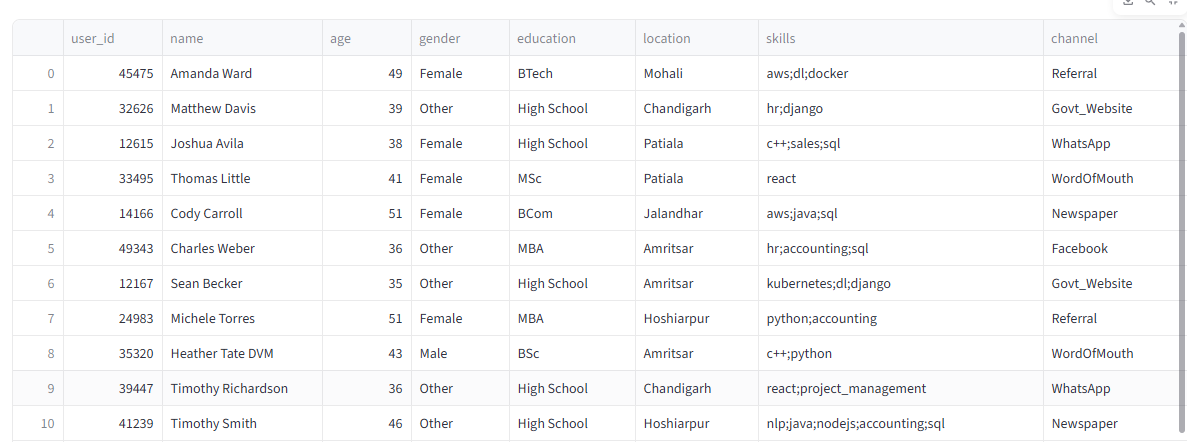
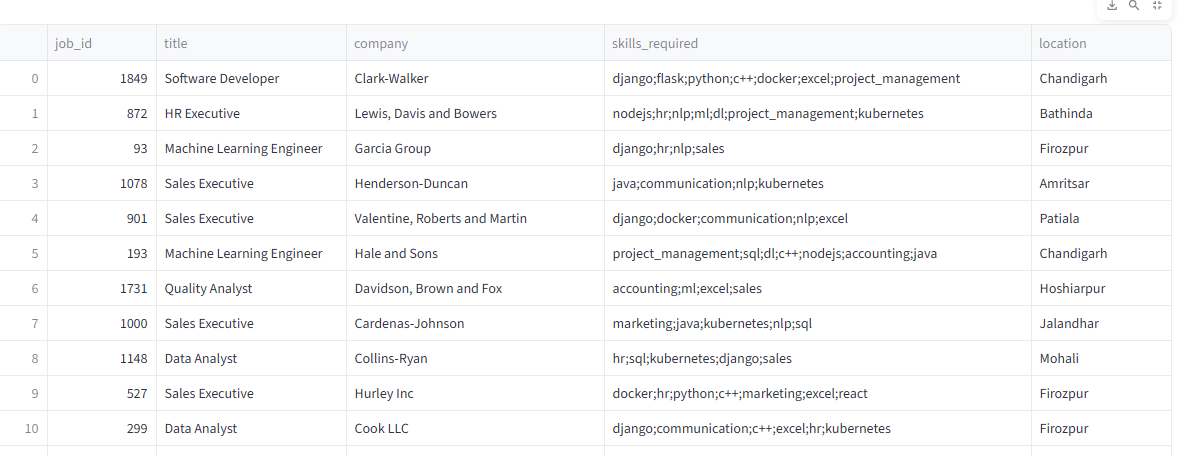
[20] R. Burke, “Hybrid recommender systems: Survey and experiments,” User Modeling and User-Adapted Interaction, vol. 12, no. 4, pp. 331–370, 2002.

**Base paper:**

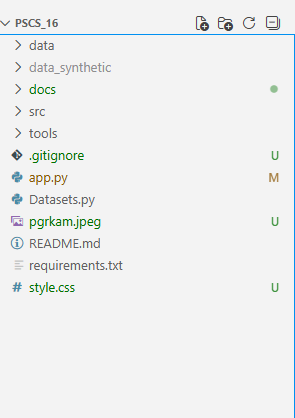
Azri, M.H.F.B., Haw, S.C., Ng, K.W. and Saad, M.F.M., 2025. Context-Aware Job Recommender System. *JOIV: International Journal on Informatics Visualization*, *9*(2), pp.877-886.

**Appendix**

**Appendix A: Datasets:**

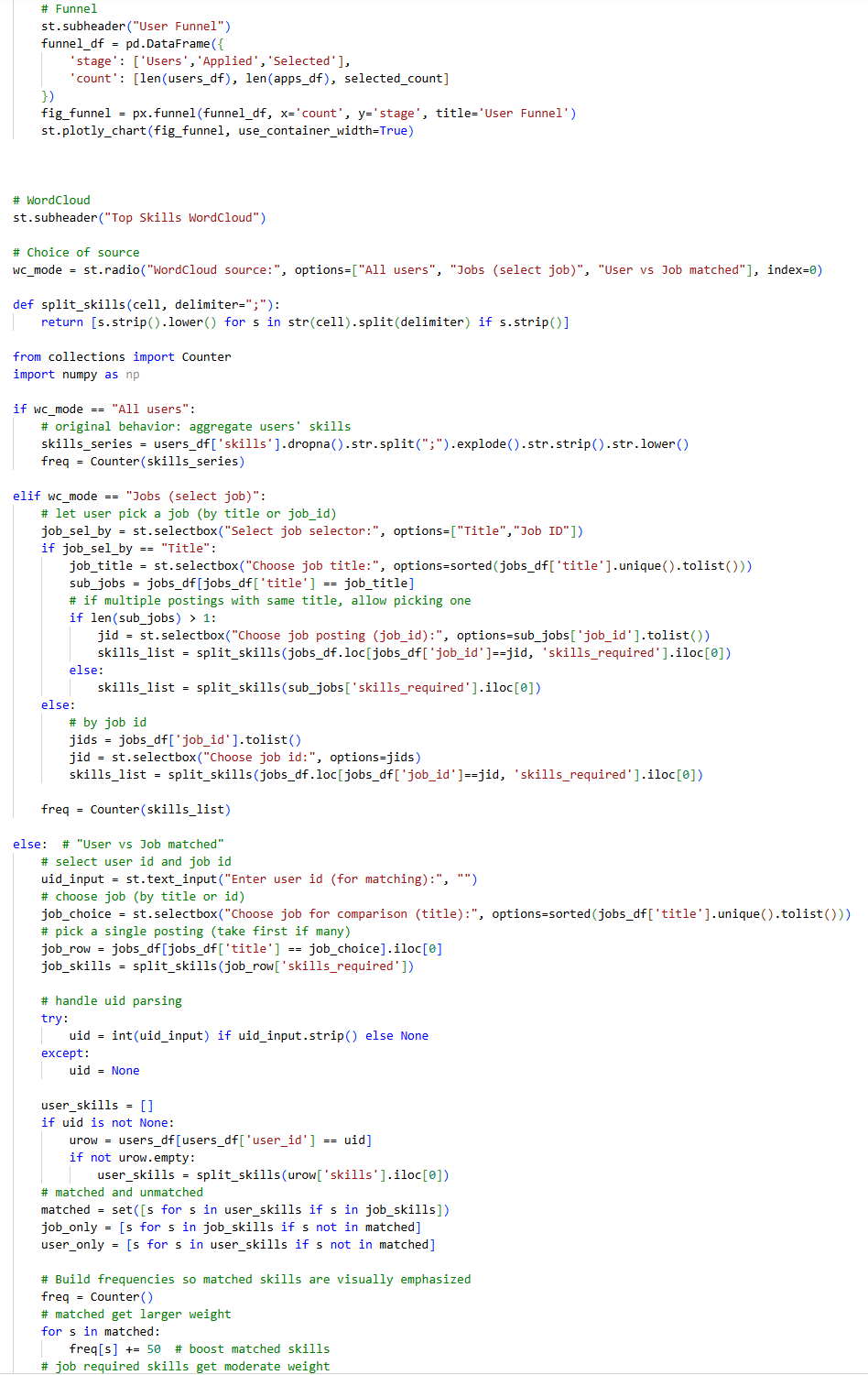
1. Users: 
2. Jobs: 
3. Applications: 

**Appendix B: Software**

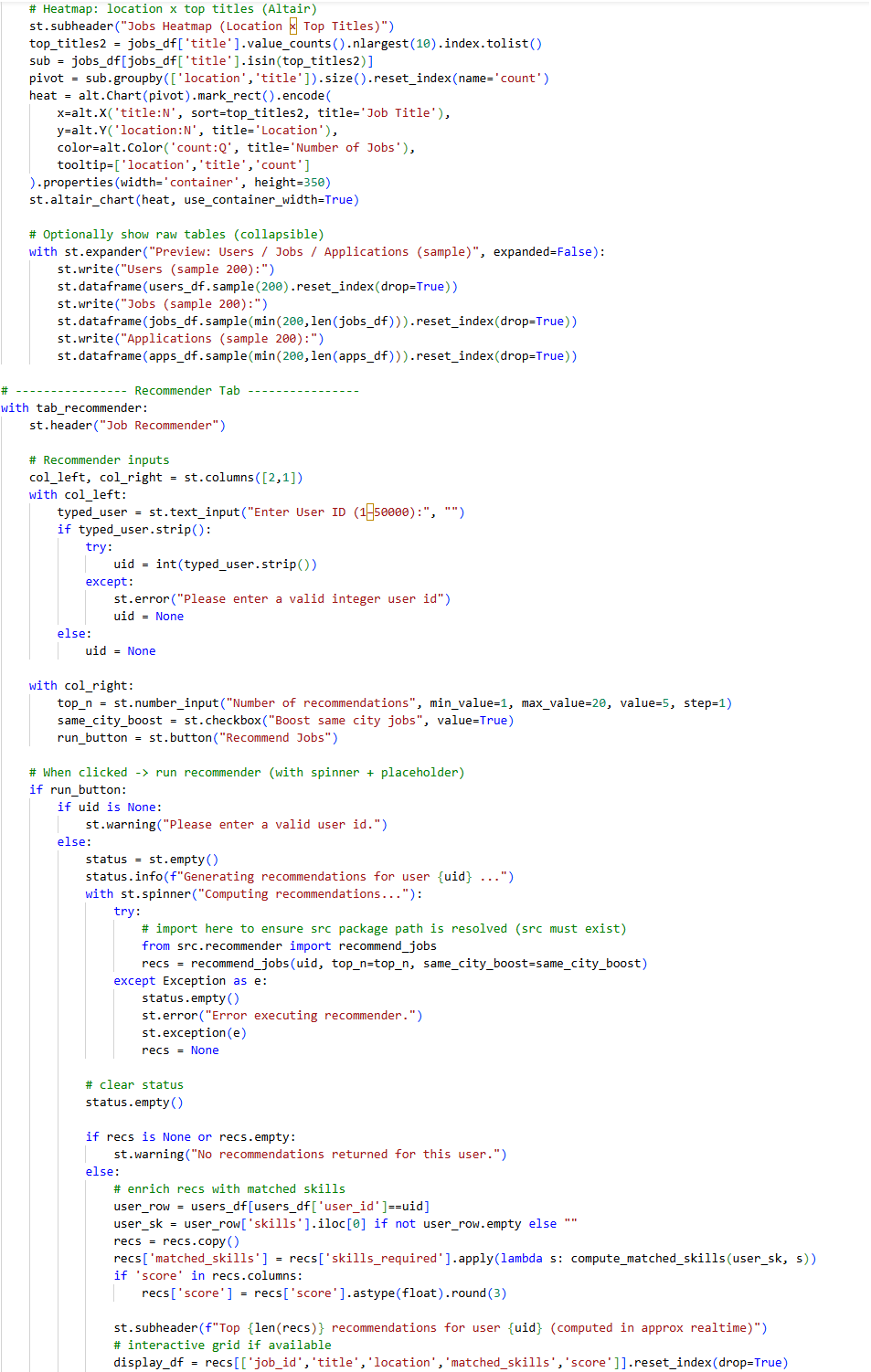
****

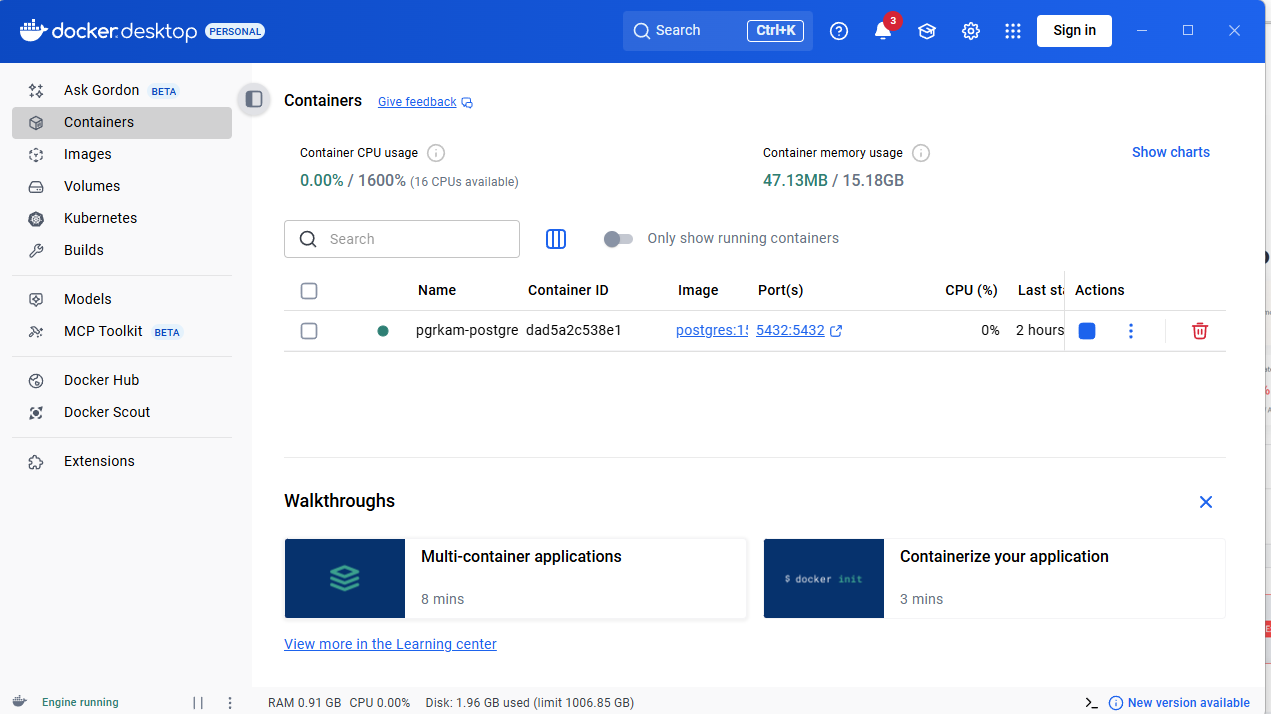


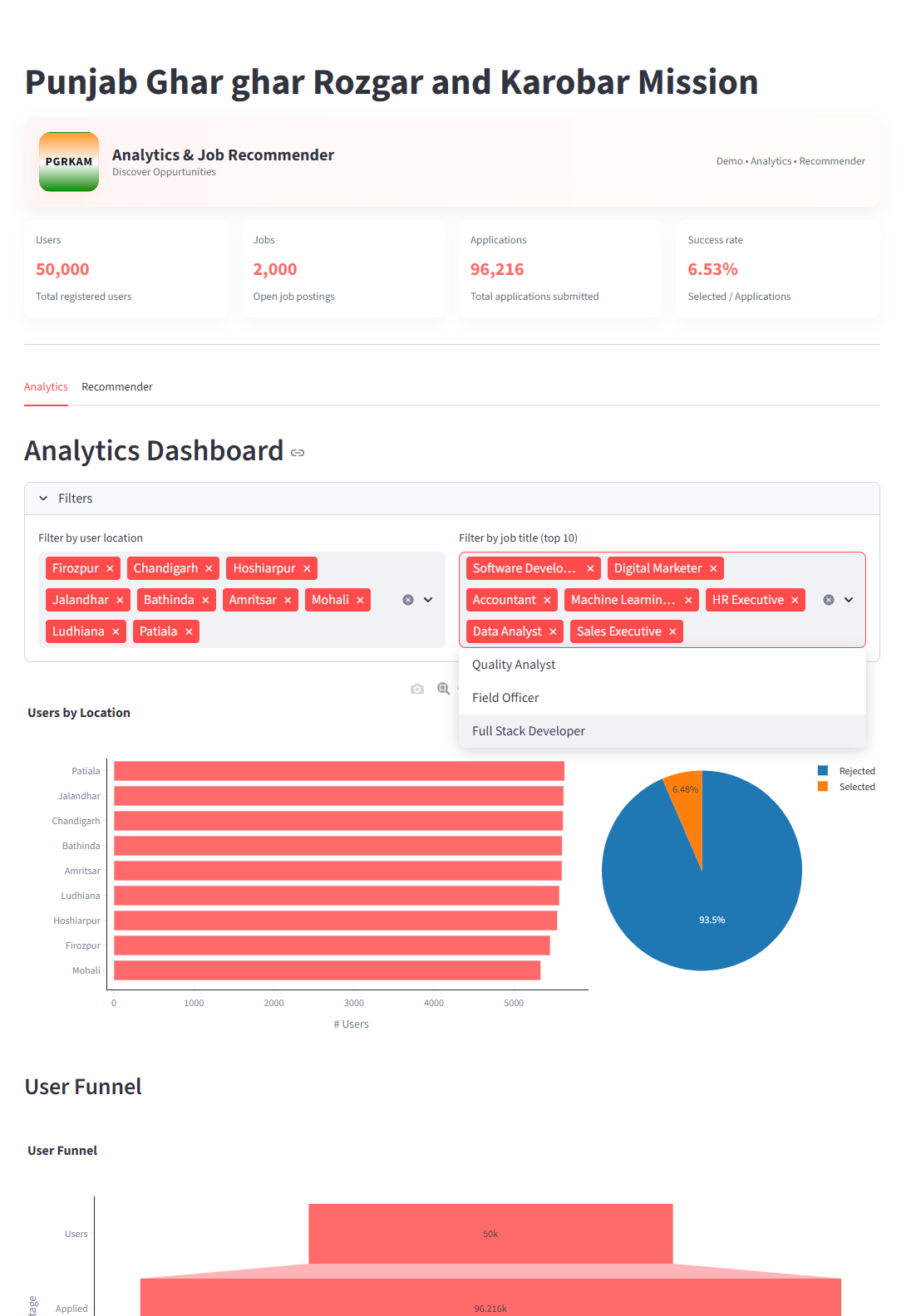


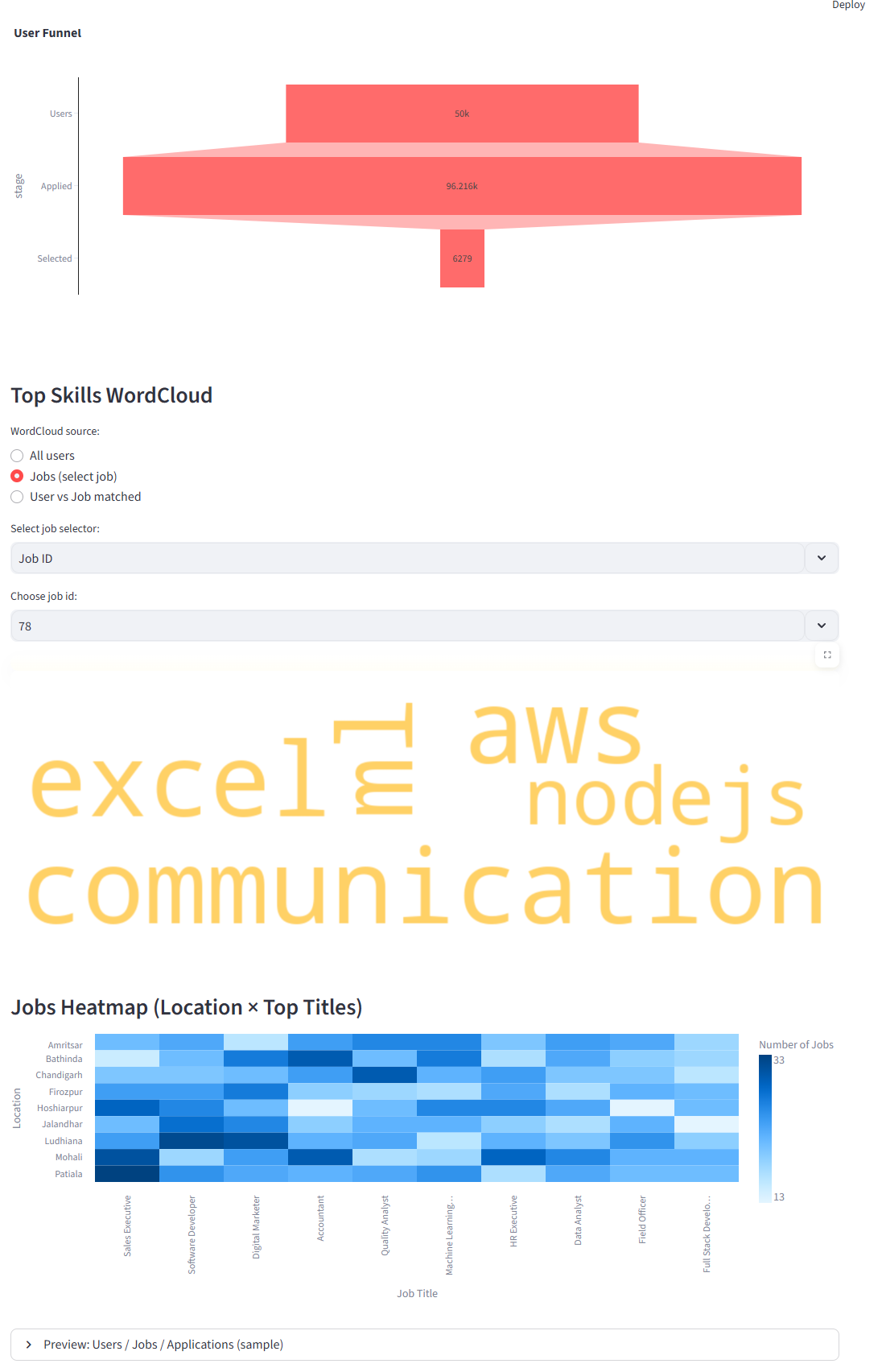


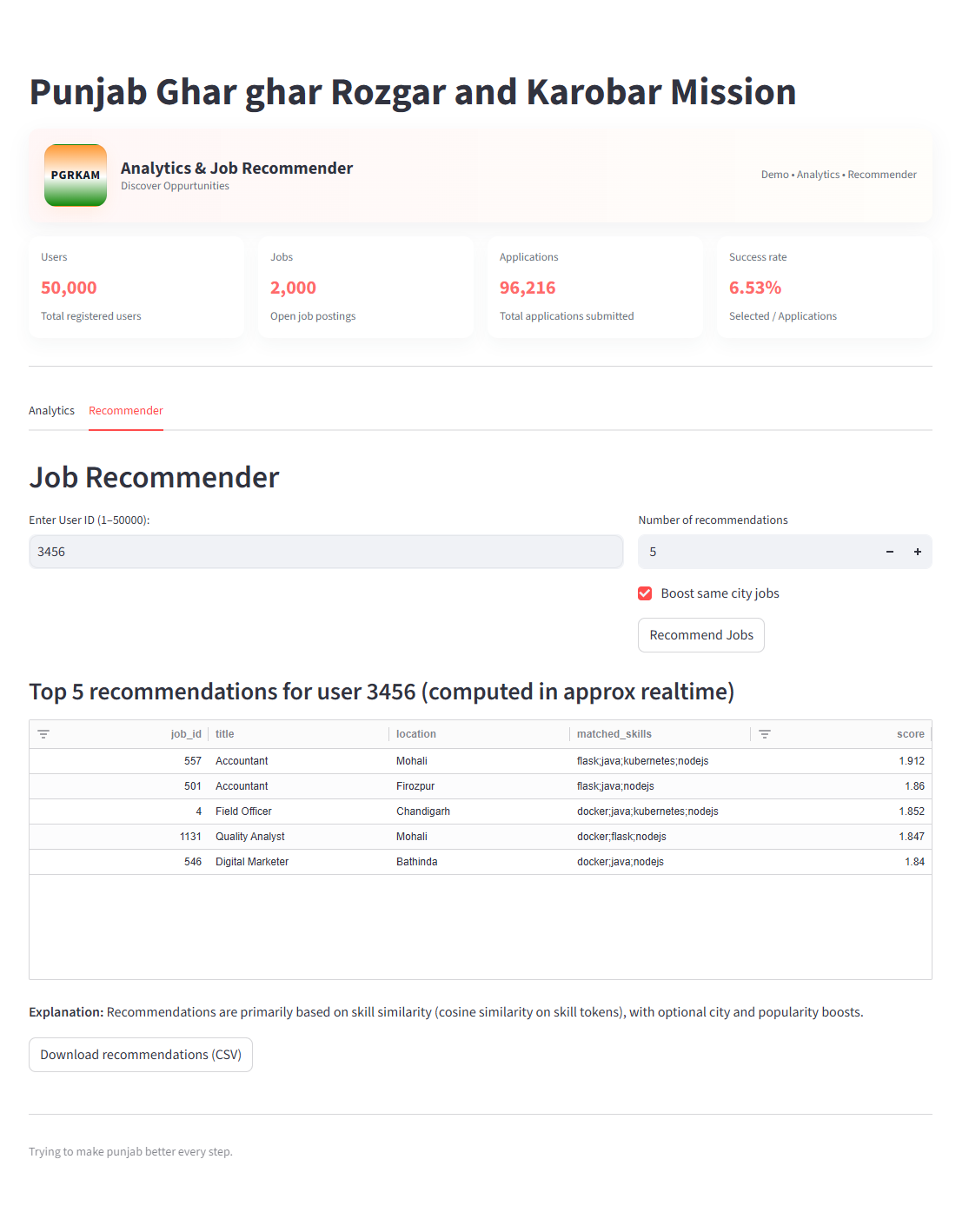


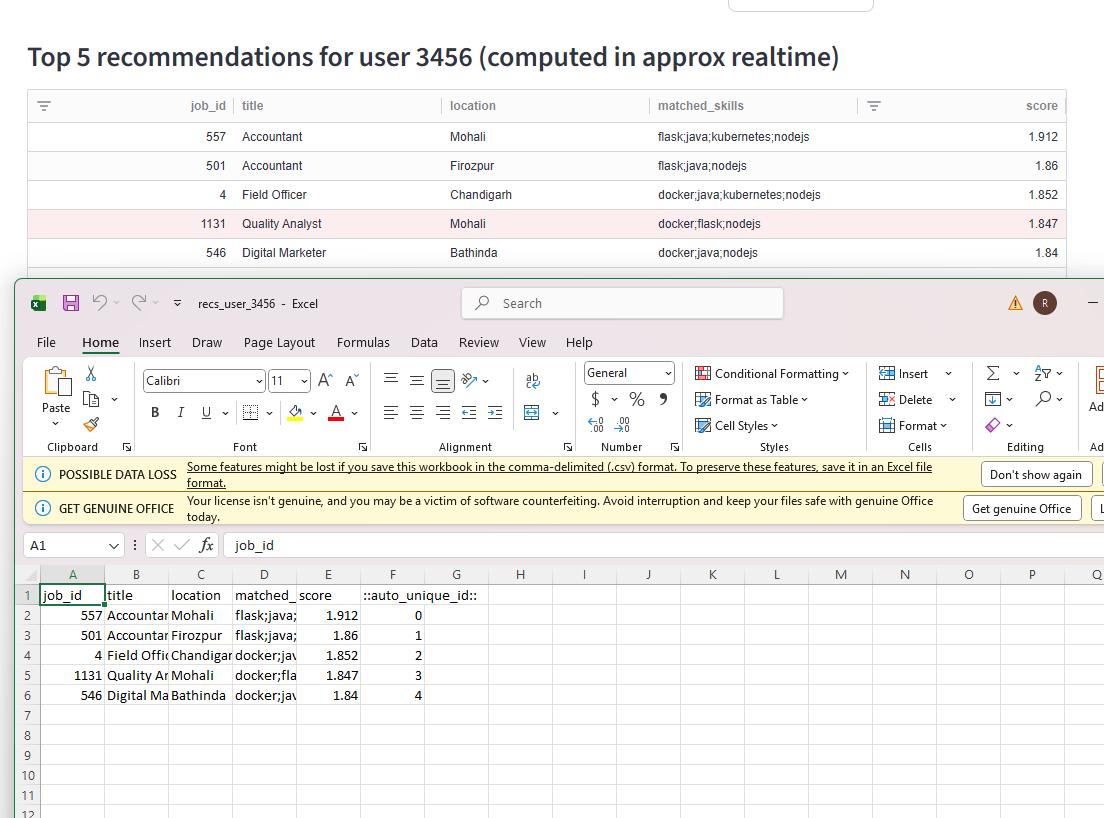


****

**Appendix C: Website**

****

****

****