URCAREER: JOB RECOMMENDATION SYSTEM

A PROJECT REPORT

Submitted by

BHARATHEESHWAR S (2116210701041)
GOKULA KRISHNA H (2116210701062)

in partial fulfillment for the course

CS19643-FOUNDATIONS OF MACHINE LEARNING

for the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE
RAJALAKSHMI NAGAR
THANDALAM
CHENNAI – 602 105

MAY 2024

RAJALAKSHMI ENGINEERING COLLEGE

CHENNAI - 602105

BONAFIDE CERTIFICATE

Certified that this Thesis titled **"URCAREER: JOB RECOMMENDATION SYSTEM** " is the bonafide work of "**BHARATHEESHWAR S (2116210701041) , GOKULA KRISHNA H (2116210701062)"** who carried out the project work for the course **CS19643-FOUNDATIONS OF MACHINE LEARNING** under my supervision.

Dr.S.Vinod Kumar
PROJECT COORDINATOR
Professor
Department of
Computer Science and Engineering
Rajalakshmi Engineering College
Rajalakshmi Nagar
Thandalam
Chennai - 602105

Submitted to Project and Viva Voce Examination for the course CS19643-Foundations of Machine Learning held on _____.

ABSTRACT

The "urcareer: Job Recommendation System" presents a groundbreaking approach to revolutionize the job search experience through the integration of advanced machine learning (ML) algorithms. In today's dynamic job market, the process of identifying suitable employment opportunities can be daunting and time-consuming, often hindered by the lack of personalized recommendations and outdated search methodologies. Traditional job search platforms typically rely on rudimentary keyword matching and simplistic filters, which may not adequately capture the nuanced preferences and skills of individual users.The "urcareer" system addresses these challenges by leveraging a hybrid approach that combines content-based and collaborative filtering ML algorithms. Content-based filtering utilizes natural language processing (NLP) techniques to analyze job descriptions and user profiles, ensuring that recommended jobs closely align with the skills, qualifications, and preferences of each user. Concurrently, collaborative filtering harnesses user interaction data to identify similarities between users and recommend jobs that have been positively received by users with comparable profiles.In summary, the "urcareer: Job Recommendation System" represents a significant advancement in job search technology, empowering users to navigate the job market with confidence and efficiency. Through the integration of hybrid content-based and collaborative filtering ML algorithms, the system provides users with tailored recommendations that match their unique skills, interests, and aspirations, ultimately facilitating more informed and satisfying career decisions.

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavour to put forth this report. Our sincere thanks to our Chairman **Thiru. S.Meganathan, B.E., F.I.E.,** our Vice Chairman **Mr. M.Abhay Shankar, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) Thangam Meganathan, M.A., M.Phil., Ph.D.,** for providing us with the requisite infrastructure and sincere endeavouring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N.Murugesan, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P.Kumar, M.E., Ph.D.,** Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We express our gratitude to our project coordinator **Dr.S.Vinod Kumar** for his encouragement and guiding us throughout the project towards successful completion of this project and to our parents, friends, all faculty members and supporting staff for their direct and indirect involvement in successful completion of the project for their encouragement and support.

BHARATHEESHWAR S (2116210701041)

GOKULA KRISHNA H (2116210701062)

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	**3**
	LIST OF FIGURES	**6**
	LIST OF ABBREVIATIONS	**7**
1.	**INTRODUCTION**	**8**
	1.1 PROBLEM STATEMENT	8
	1.2 AIM AND OBJECTIVES	9
	1.3 EXISTING SYSTEM	10
	1.4 PROPOSED SYSTEM	11
2.	**LITERATURE REVIEW**	**12**
3.	**HARDWARE AND SOFTWARE REQUIREMENTS**	**17**
4.	**SYSTEM DESIGN**	**18**
	4.1 ARCHITECTURE DIAGRAM	18
	4.2 SYSTEM FLOW DIAGRAM	19
	4.3 SEQUENCE DIAGRAM	20
	4.4 ER DIAGRAM	21
5.	**PROJECT DESCRIPTION**	**22**
	5.1. MODULES	22
6.	**SAMPLE CODING**	**25**
7.	**OUTPUTS AND FINAL RESULTS**	**31**
8.	**CONCLUSION AND FUTURE ENHANCEMENTS**	**35**
	LIST OF REFERENCES	**36**

LIST OF FIGURES

Figure No	Figure Name	Page No.
4.1	Architecture Diagram	18
4.2	System Flow Diagram	19
4.3	Sequence Diagram	20
4.4	ER Diagram	21
5.1	Searching based on user-id	31
5.2	Job Recommendation for user 1	31
5.3	Job Recommendation for user 2	32
5.4	Job Recommendation for user 3	32
5.5	Job Recommendation for user 4	33
5.6	Content-Based Filtering	33
5.7	Analytical Predictions of Jobs at Urcareer	34

LIST OF ABBREVIATIONS

ABBREVIATION	ACRONYM
HTML	Hyper-Text Markup Language
CSS	Cascading Style Sheets
API	Application Programming Interface
URL	Uniform Resource Locator
UI	User Interface
JS	JavaScript
TF - IDF	Term Frequency-Inverse Document Frequency
NLP	Natural Language Processing
ML	Machine Learning
CSV	Comma Separated Values
IDE	Integrated Development

CHAPTER 1
INTRODUCTION

1.1 PROBLEM STATEMENT

In today's rapidly evolving job market, the process of finding suitable employment opportunities is often fraught with challenges and inefficiencies. Traditional job search methods rely heavily on manual efforts and generic algorithms, resulting in a lack of personalized recommendations and suboptimal matches between job seekers and available positions. Moreover, existing job search platforms often struggle to adapt to the diverse needs and preferences of individual users, leading to frustration and dissatisfaction among job seekers.

The "urcareer: Job Recommendation System" seeks to address these shortcomings by leveraging advanced machine learning (ML) algorithms to deliver highly personalized job recommendations tailored to each user's unique skills, qualifications, and preferences. The primary challenge lies in developing a robust recommendation engine that can effectively analyze and interpret complex job descriptions and user profiles, extracting meaningful insights to facilitate accurate matching.

Additionally, the system must overcome the inherent limitations of traditional recommendation approaches by integrating a hybrid model that combines content-based and collaborative filtering techniques. This hybrid approach presents its own set of challenges, including algorithmic complexity, data integration, and scalability issues.By addressing these challenges, the system seeks to provide job seekers with a more efficient, effective, and personalized way to navigate the job market, ultimately facilitating better matches between candidates and employers.

1.2. AIM AND OBJECTIVES:

Aim:

The aim of the "urcareer: Job Recommendation System" is to provide a comprehensive and personalized solution to enhance the job search process for users. The primary objective is to develop a sophisticated recommendation system that leverages machine learning algorithms to deliver tailored job suggestions based on users' skills, qualifications, and preferences.

Objectives:

1. Implementing a hybrid approach combining content-based and collaborative filtering algorithms to ensure highly relevant job recommendations.

2. Integrating natural language processing (NLP) techniques to analyze job descriptions and user profiles effectively.

3. Developing an intuitive user interface for seamless interaction and navigation within the platform.

4. Enhancing the system's learning capabilities to adapt to users' evolving preferences and feedback.

5. Evaluating the system's performance through user feedback and iterative improvements to optimize recommendation accuracy and user satisfaction.

1.3 EXISTING SYSTEM

In the current landscape of job search platforms, the process of finding suitable employment opportunities is often characterized by inefficiencies and limitations. Traditional job search platforms primarily rely on basic keyword matching and simplistic filters, which may lead to irrelevant or inadequate job recommendations for users. These platforms often lack the ability to provide personalized recommendations that align closely with the skills, qualifications, and preferences of individual users. Additionally, they may not effectively leverage user interaction data to improve the relevance and accuracy of job suggestions over time.

Moreover, existing job search systems typically lack advanced machine learning (ML) capabilities, which are essential for delivering highly personalized recommendations. Without ML algorithms, these systems struggle to analyze and interpret complex job descriptions and user profiles, resulting in suboptimal matching between job seekers and job opportunities.

Overall, the existing job search landscape is characterized by its limitations in providing personalized and accurate job recommendations, as well as its reliance on outdated search methodologies. There is a clear need for a more sophisticated and efficient solution that leverages advanced ML algorithms to overcome these challenges and deliver superior job recommendations tailored to the unique preferences and requirements of individual users.As such, there is a pressing need for a more sophisticated and efficient job recommendation system that leverages hybrid content-based and collaborative filtering techniques to deliver highly personalized and relevant job recommendations to users.

1.4 PROPOSED SYSTEM

The proposed "urcareer: Job Recommendation System" aims to revolutionize the job search process by introducing an advanced platform that harnesses the power of machine learning (ML) algorithms. Unlike traditional job search platforms that often provide generic and impersonalized recommendations, the proposed system will offer highly tailored job suggestions based on the unique skills, preferences, and career objectives of each user.

At the core of the proposed system lies a hybrid approach that integrates content-based and collaborative filtering ML algorithms. Content-based filtering utilizes natural language processing (NLP) techniques to analyze job descriptions and user profiles, enabling the system to identify relevant job opportunities that closely match the user's qualifications and interests. Meanwhile, collaborative filtering leverages user interaction data to identify patterns and similarities between users, allowing the system to recommend jobs that have been positively received by users with similar profiles.

The "urcareer" system will feature a user-friendly interface that allows users to input their preferences and search criteria effortlessly. Users will have the option to specify their skills, experience level, desired industry, location preferences, and other relevant factors, enabling the system to generate tailored recommendations that align with their unique requirements.

Furthermore, the proposed system will continuously adapt and improve its recommendations based on user feedback and interactions. By incorporating a feedback loop mechanism, the system will learn from user behavior and preferences over time, ensuring that the recommendations remain accurate, relevant, and up-to-date.

CHAPTER 2
LITERATURE REVIEW

The field of career guidance and job recommendation has evolved significantly in recent years, driven by advancements in technology and a growing demand for personalized and data-driven solutions. Historically, career counseling has relied on traditional models such as trait-factor theory and Holland's theory of vocational choice. These models focus on matching individuals with careers based on personality traits, interests, and aptitudes. While these approaches have provided valuable insights into career decision-making processes, they often lack the flexibility and customization needed to address the diverse needs of today's workforce.Recent research has highlighted the potential of technology to enhance career guidance services. Tools such as online career assessments, virtual career fairs, and AI-powered chatbots have emerged as effective means of delivering personalized career advice and resources to individuals. For example, studies by Savickas (2019) and Sampson et al. (2020) emphasize the importance of integrating technology into career counseling to reach a wider audience and provide scalable solutions.The use of natural language processing (NLP) techniques for resume analysis has gained traction in the field of human resources and career guidance. Research by Gugnani et al. (2018) demonstrates the effectiveness of NLP in extracting key information from resumes, such as skills, experiences, and qualifications. By automating the resume screening process, NLP algorithms can save time and resources for both job seekers and employers.Machine learning (ML) algorithms offer promising opportunities for assessing individual skills and competencies in a data-driven manner. Studies by AlZoubi et al. (2021) and Sina et al. (2020) explore the use of ML models for predicting job performance and identifying skill gaps.

Personalized job recommendation systems leverage user data and machine learning algorithms to match individuals with relevant job opportunities. Research by Li et al. (2019) and Zhou et al. (2021) demonstrates the effectiveness of recommendation algorithms in improving job search outcomes and user satisfaction. These systems analyze factors such as skills, experience, and career preferences to provide tailored job recommendations that align with the individual's needs and interests.While technology has transformed the field of career guidance, several challenges remain. Issues such as data privacy, algorithmic bias, and the digital divide must be addressed to ensure equitable access to career resources and opportunities. Future research should focus on developing ethically sound and inclusive technologies that empower individuals from diverse backgrounds to make informed decisions about their careers.As technology plays an increasingly prominent role in career guidance, it is crucial to address ethical considerations to ensure fair and equitable outcomes for all users. Research by Coetzee and Ferreira (2019) highlights the importance of transparency, accountability, and privacy protection in the design and implementation of career guidance technologies. Ethical guidelines should be established to govern the collection, use, and storage of user data, particularly sensitive information such as resumes and personal preferences. Additionally, measures to mitigate algorithmic bias and ensure algorithmic fairness must be implemented to prevent discriminatory outcomes in job recommendations and skill assessments. Looking ahead, several emerging trends and technologies have the potential to shape the future of career guidance and job recommendation. Augmented reality (AR) and virtual reality (VR) applications offer immersive experiences for career exploration and skill development, allowing users to simulate real-world work environments and tasks. Blockchain technology holds promise for creating decentralized career credentialing systems, enabling individuals to securely store and share their qualifications and achievements.

In recent years, the utilization of machine learning (ML) algorithms in job recommendation systems has garnered significant attention from researchers and practitioners alike. Various studies have explored the application of ML techniques to enhance the accuracy and effectiveness of job recommendation platforms, aiming to address the challenges associated with traditional job search methodologies.Content-based filtering has emerged as a prominent approach in job recommendation systems, leveraging natural language processing (NLP) techniques to analyze job descriptions and user profiles. Researchers have investigated the effectiveness of content-based methods in generating personalized job recommendations by matching the skills, qualifications, and preferences of individual users with relevant job postings. Additionally, advancements in deep learning algorithms have facilitated the extraction of semantic information from unstructured text data, enabling more nuanced and context-aware job recommendations.Collaborative filtering represents another vital component of job recommendation systems, leveraging user interaction data to identify patterns and similarities between users. Numerous studies have explored the application of collaborative filtering techniques, such as matrix factorization and nearest neighbor algorithms, in generating recommendations based on user preferences and historical behavior. By harnessing collective user feedback, collaborative filtering algorithms can effectively identify relevant job opportunities and provide personalized recommendations to users.Hybrid approaches, which combine content-based and collaborative filtering techniques, have gained prominence in the field of job recommendation systems. These hybrid models aim to leverage the strengths of both approaches while mitigating their respective limitations. Researchers have proposed various hybridization strategies, including feature combination, model fusion, and ensemble methods, to improve recommendation accuracy and enhance user satisfaction.

The incorporation of deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), has emerged as a promising avenue for enhancing the performance of job recommendation systems. These neural network-based models can effectively capture complex patterns and relationships in job-related data, leading to more accurate and personalized recommendations. Additionally, the use of reinforcement learning algorithms has enabled the development of adaptive recommendation systems that can continuously learn and improve over time based on user feedback and interaction.Beyond algorithmic approaches, researchers have also focused on user modeling techniques to enhance the personalization and relevance of job recommendations. User profiling methods, such as demographic analysis, behavioral modeling, and skill mapping, are employed to capture the preferences, interests, and expertise of individual users. Additionally, context-aware recommendation techniques, which consider temporal, spatial, and situational factors, have been investigated to provide more contextually relevant job suggestions.

The evaluation of job recommendation systems poses unique challenges due to the subjective nature of job suitability and the absence of ground truth labels for user preferences. Researchers have proposed various evaluation metrics and methodologies to assess the effectiveness and performance of recommendation algorithms. Common evaluation metrics include precision, recall, F1-score, and ranking metrics, such as mean average precision (MAP) and normalized discounted cumulative gain (NDCG). Moreover, user-centric evaluation techniques, such as user studies and A/B testing, are employed to gather feedback and insights from real users regarding the quality and utility of job recommendations.

Furthermore, research efforts have focused on addressing the challenges of scalability, privacy, and fairness in job recommendation systems. Scalable ML algorithms and distributed computing frameworks have been developed to handle large-scale job datasets and support real-time recommendation generation. Moreover, privacy-preserving techniques, such as differential privacy and federated learning, have been explored to protect user data and ensure confidentiality in recommendation systems. Additionally, efforts have been made to mitigate algorithmic biases and promote fairness and diversity in job recommendations by employing fairness-aware ML techniques and designing inclusive recommendation algorithms.In addition to algorithmic and evaluation aspects, the literature also addresses practical considerations, such as scalability, efficiency, and privacy, in the design and implementation of job recommendation systems. Scalable ML algorithms and distributed computing frameworks are employed to handle large-scale job datasets and support real-time recommendation generation. Furthermore, privacy-preserving techniques, including differential privacy and federated learning, are investigated to protect user data and ensure confidentiality in recommendation systems.

In summary, the literature survey highlights the significant advancements and research trends in the field of job recommendation systems, with a particular focus on the utilization of ML algorithms. By leveraging content-based, collaborative, and hybrid approaches, as well as incorporating deep learning and reinforcement learning techniques, researchers aim to develop more accurate, personalized, and scalable job recommendation systems that cater to the diverse needs and preferences of users in today's competitive job market.

CHAPTER 3
HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirements:

- Desktop/Laptop Computers:
    - Processor: Intel Core i3 or equivalent
    - RAM: 4GB or higher
    - Storage: 128GB SSD or higher
    - Display: 15-inch monitor or larger
    - Processor: Intel Core i5 or equivalent
    - RAM: 8GB or higher
    - Storage: 512GB SSD or higher
    - Display: 15-inch monitor or larger

Software Requirements:

- Python 3.8 or higher
- Flask Framework
- Pandas Library
- Scikit-learn Library
- Spacy Library
- HTML/CSS/JS for Frontend Development

CHAPTER 4
SYSTEM DESIGN

4.1 ARCHITECTURE DIAGRAM

An architecture diagram is a visual representation or blueprint that illustrates the high-level structure of a system, application, or a specific component within a system.

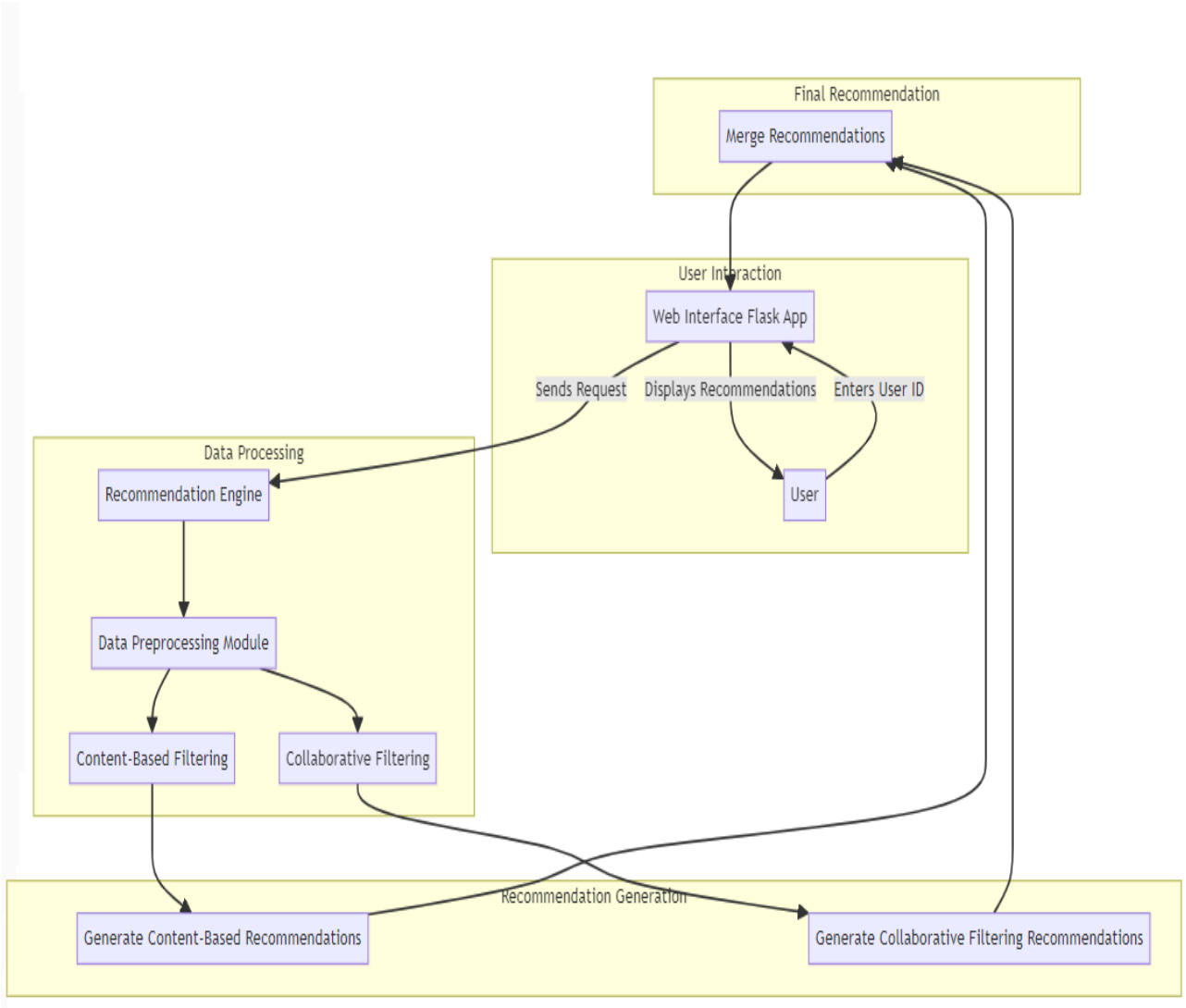


FIG. 4.1 ARCHITECTURE DIAGRAM

4.2 SYSTEM FLOW DIAGRAM

A Flow-Chart represents the flow of the application.It is a representation of a start to end of the application.

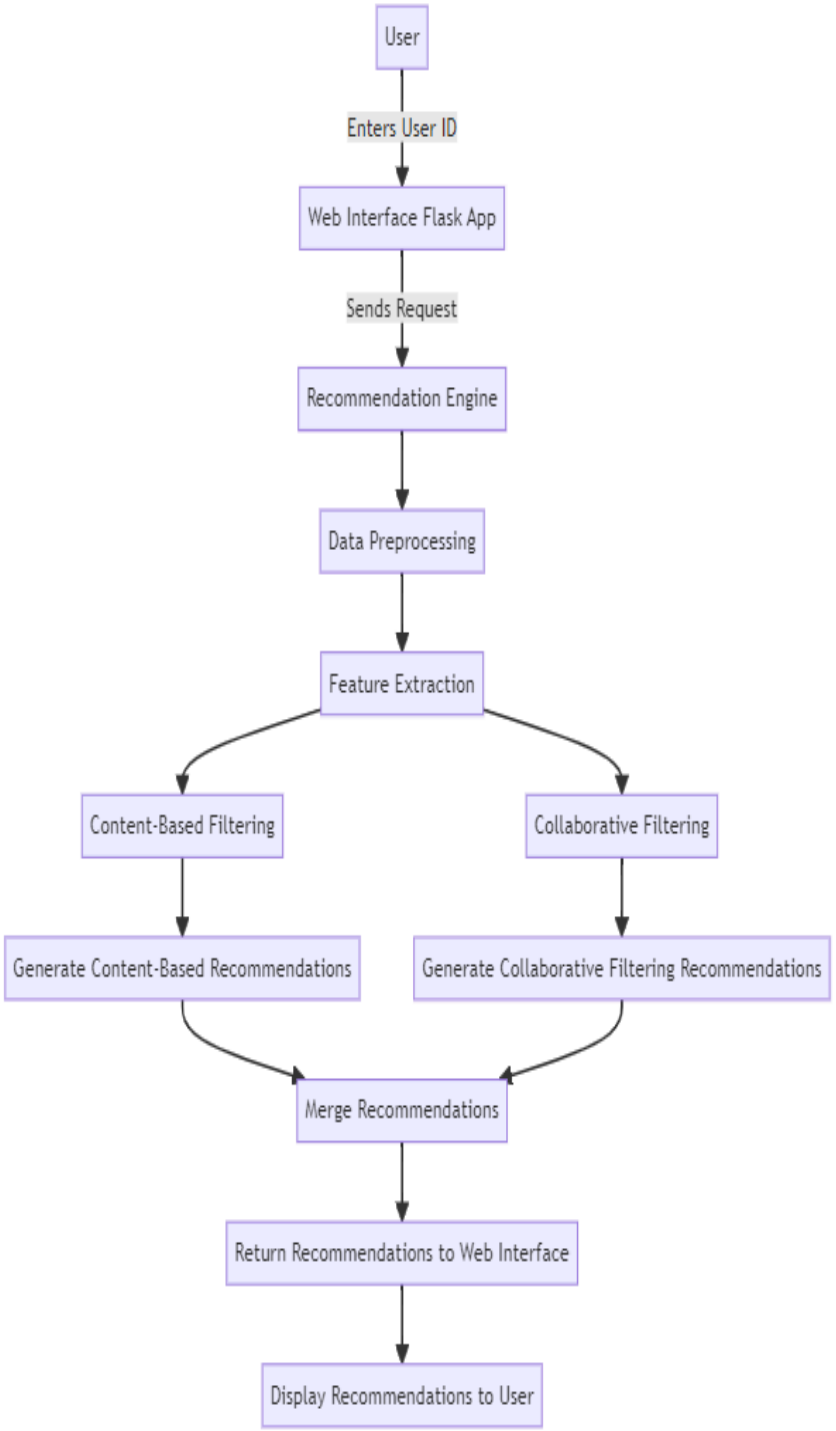


FIG. 4.2 SYSTEM FLOW DIAGRAM

4.3 SEQUENCE DIAGRAM

A sequence diagram is a type of interaction diagram in the Unified Modeling Language (UML) that illustrates the interactions between objects or components within a system in a chronological order. It provides a dynamic view of the system's behavior by depicting the sequence of messages exchanged between different entities over time.

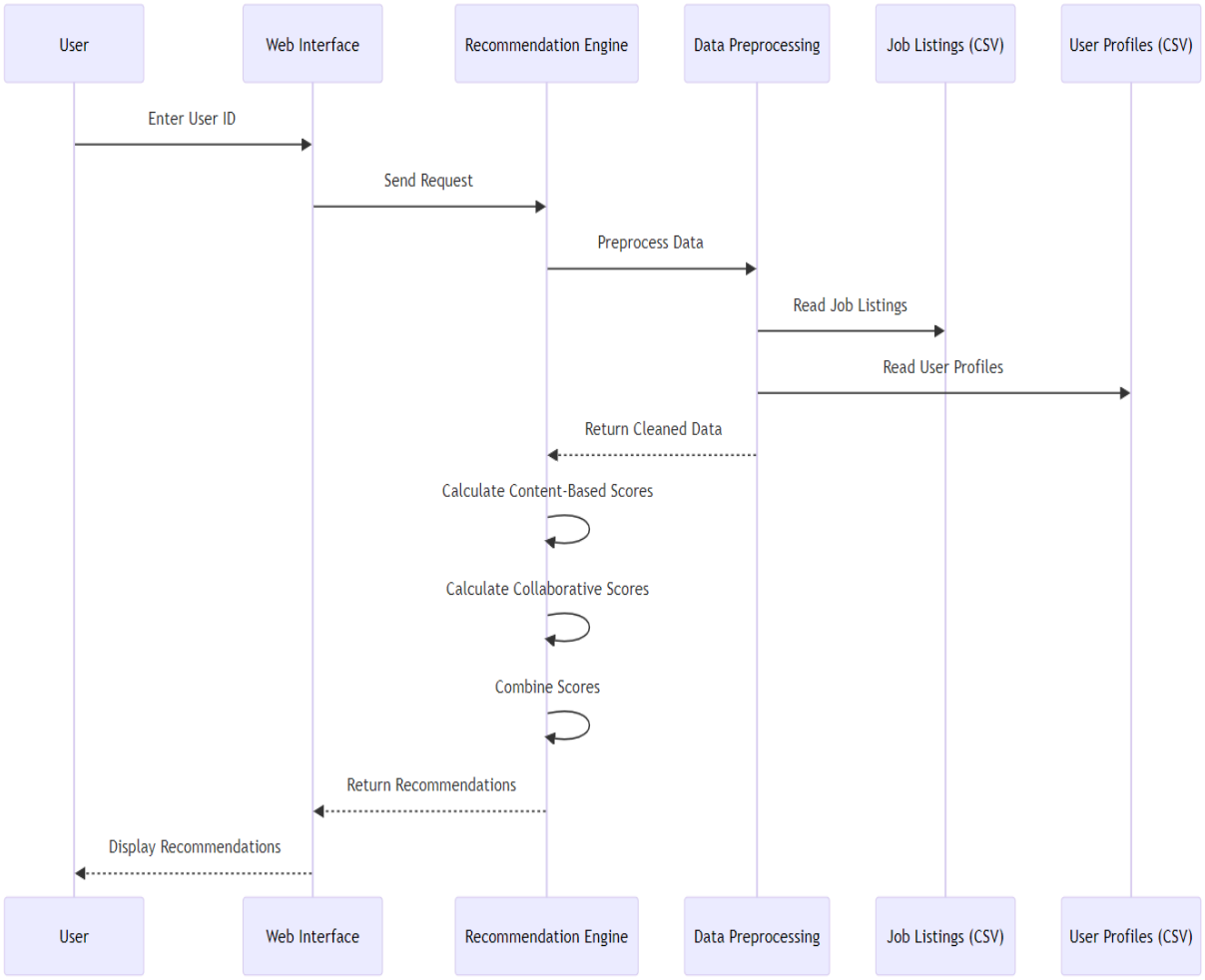


FIG. 4.3 SEQUENCE DIAGRAM

4.4 ENTITY-RELATIONSHIP DIAGRAM

An Entity-Relationship (ER) diagram is a graphical representation of the data model of a system or a business process.

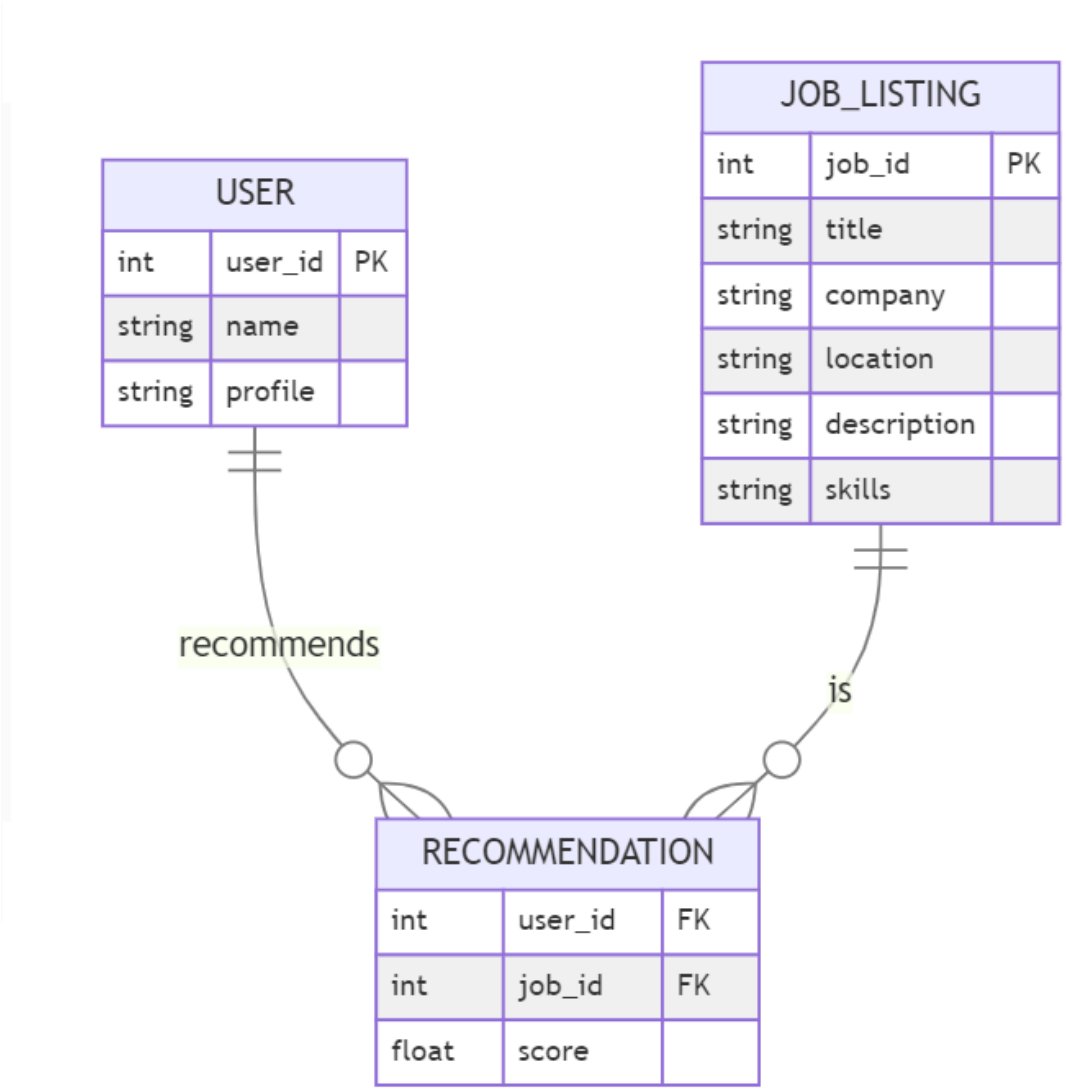


FIG. 4.4 ER DIAGRAM

CHAPTER 5
PROJECT DESCRIPTION

5.1 MODULES

1. Data Preprocessing Module:

This module is responsible for processing and cleaning the raw data obtained from job listings and user profiles. It utilizes natural language processing (NLP) techniques through libraries such as spaCy to tokenize, lemmatize, and remove stop words from job descriptions. Additionally, it implements data cleaning operations to handle missing values, incorrect formats, and outliers in the dataset. Libraries like pandas and NumPy are employed for efficient data manipulation and preprocessing operations.

2. Feature Extraction Module:

The Feature Extraction Module extracts relevant features from the preprocessed data to represent job descriptions and user profiles. It utilizes techniques such as TF-IDF (Term Frequency-Inverse Document Frequency) vectorization to convert textual data into numerical feature vectors. The module also employs libraries like scikit-learn for implementing feature extraction algorithms and generating feature matrices.Furthermore, the Feature Extraction Module may incorporate word embeddings techniques such as Word2Vec or GloVe to represent words in a continuous vector space.These embeddings capture semantic relationships between words, allowing for a more nuanced understanding of textual data.

3. Content-Based Filtering Module:

This module implements content-based recommendation algorithms to suggest jobs based on the similarity between job descriptions and user profiles. It utilizes cosine similarity or other distance metrics to measure the similarity between feature vectors representing job descriptions and user profiles. Advanced ML techniques such as word embeddings (e.g., Word2Vec) are incorporated for capturing semantic similarity between words and phrases. The module relies on libraries like scikit-learn for implementing similarity calculations and integrating NLP models.By focusing on the content of the jobs, including skills required, job titles, and descriptions, this module excels at suggesting relevant opportunities that align with the user's expertise and interests.

4. Collaborative Filtering Module:

The Collaborative Filtering Module implements collaborative filtering algorithms to recommend jobs based on user interaction data and similarities between users. It utilizes user-item matrices to capture user preferences and interactions with job listings. Techniques such as matrix factorization (e.g., Singular Value Decomposition) or deep learning models (e.g., neural collaborative filtering) are implemented to learn latent user and item representations. Libraries like TensorFlow or PyTorch are utilized for implementing complex collaborative filtering models and optimizing recommendation performance.Techniques such as matrix factorization, including Singular Value Decomposition (SVD) or Alternating Least Squares (ALS), are employed to uncover latent features representing user preferences and job characteristics.

5. Hybrid Recommendation Module:

The Hybrid Recommendation Module integrates content-based and collaborative filtering approaches to provide enhanced job recommendations. It combines the strengths of both approaches to mitigate their respective weaknesses and provide more accurate and diverse recommendations. Ensemble techniques or weighted combination strategies are implemented to fuse recommendation scores from content-based and collaborative filtering algorithms. The module utilizes libraries like NumPy for efficient array manipulation and ensemble modeling.

6. User Interface Module:

The User Interface Module develops a user-friendly interface for users to interact with the recommendation system. It implements web-based or desktop GUIs using frameworks like Flask or Django for seamless integration with backend modules. HTML, CSS, and JavaScript are utilized for designing responsive and visually appealing user interfaces. Interactive elements and real-time updates are incorporated to enhance user experience and engagement.

7. Evaluation and Testing Module:

The Evaluation and Testing Module implements evaluation metrics to assess the performance and effectiveness of the recommendation system. It utilizes metrics such as precision, recall, and mean average precision (MAP) to evaluate recommendation quality. Libraries like scikit-learn are utilized for implementing evaluation metrics and statistical analysis.

CHAPTER 6

SAMPLE CODING

urcareer.py

```
from flask import Flask, render_template, request
from flask_cors import CORS
import pandas as pd
import spacy
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import numpy as np

app = Flask(__name__)
CORS(app)

# Load and preprocess data
job_data = pd.read_csv('job_listings.csv')
user_data = pd.read_csv('user_profiles.csv')
nlp = spacy.load('en_core_web_sm')

def preprocess(text):
    doc = nlp(text)
        tokens = [token.lemma_ for token in doc if not token.is_stop and token.is_alpha]
    return ' '.join(tokens)

job_data['processed_description'] = job_data['description'].apply(preprocess)
vectorizer = TfidfVectorizer(max_features=1000)
tfidf_matrix = vectorizer.fit_transform(job_data['processed_description'])
similarity_matrix = cosine_similarity(tfidf_matrix, tfidf_matrix)
```

```
user_item_matrix = np.random.rand(user_data.shape[0], job_data.shape[0])

def hybrid_recommendations(user_id):
    content_scores = similarity_matrix.dot(user_item_matrix[user_id])
    collaborative_scores = user_item_matrix[user_id]
    combined_scores = 0.5 * content_scores + 0.5 * collaborative_scores
    return np.argsort(combined_scores)[::-1][:3]

@app.route('/recommend', methods=['POST'])
def recommend():
    if request.method == 'POST':
        user_id = int(request.form['user_id'])
        try:
                        user_name = user_data.loc[user_data['user_id'] == user_id,
'name'].values[0]
        except IndexError:
            return "User ID not found", 404
        recommended_jobs = hybrid_recommendations(user_id)
        recommended_jobs_data = job_data.iloc[recommended_jobs]
                return render_template('index.html', user={'name': user_name},
matched_jobs=recommended_jobs_data.to_dict('records'))

@app.route('/')
def index():
    return render_template('index.html')

if __name__ == '__main__':
    app.run(debug=True)
```

urcareer.ipynb

```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.preprocessing import MinMaxScaler
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

nltk.download('punkt')
nltk.download('stopwords')



jobs_df = pd.read_csv('jobs.csv')
jobs_df.head()




resumes = [
    "Experienced data scientist with expertise in Python, machine learning, and data visualization.",
   "Software engineer with proficiency in Java, C++, and cloud computing.",
    "Fresh graduate with knowledge in computer science, fabrication, and quality check.",
   "Technical support specialist with skills in troubleshooting and user support.",
     "Software tester experienced in manual testing, test engineering, and test cases.",
   # Add more resumes as needed
```

```
]

resumes_df = pd.DataFrame(resumes, columns=['Resume'])

skills_list = [
        'Python', 'machine learning', 'data visualization', 'Java', 'C++', 'cloud computing',
    'computer science', 'fabrication', 'quality check', 'technical support',
    'manual testing', 'test engineering', 'test cases', 'troubleshooting', 'user support'
]

def extract_skills(text, skills):
    tokens = word_tokenize(text.lower())
    tokens = [word for word in tokens if word not in stopwords.words('english')]
    extracted_skills = [skill for skill in skills if skill.lower() in tokens]
    return extracted_skills

resumes_df['Skills'] = resumes_df['Resume'].apply(lambda x: extract_skills(x, skills_list))
resumes_df.to_csv('extracted_skills.csv', index=False)


df = pd.read_csv('extracted_skills.csv')
df.head()


# Collaborative Filtering
interaction_data = {
    'Resume_ID': [0, 0, 1, 1, 2, 2, 3, 3, 4, 4],
```

```
    'Job_ID': [0, 2, 1, 3, 2, 4, 3, 0, 4, 1],
    'Interaction': [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
}

interaction_df = pd.DataFrame(interaction_data)
user_item_matrix = interaction_df.pivot(index='Resume_ID', columns='Job_ID', values='Interaction').fillna(0)

user_similarity = cosine_similarity(user_item_matrix)

def get_collaborative_recommendations(user_id, user_similarity, top_n=3):
    similar_users = list(enumerate(user_similarity[user_id]))
            similar_users = sorted(similar_users, key=lambda x: x[1], reverse=True)[1:top_n+1]
    recommended_jobs = set()
    for user, score in similar_users:
                user_jobs = set(interaction_df[interaction_df['Resume_ID'] == user]['Job_ID'])
        recommended_jobs = recommended_jobs.union(user_jobs)
    return recommended_jobs

collab_recommendations = get_collaborative_recommendations(0, user_similarity)
print("Collaborative Filtering Recommendations for Resume 0:", collab_recommendations)



# In[9]:
```

```
# Content-Based Filtering
jobs_df['Combined'] = jobs_df['Key Skills'] + ' ' + jobs_df['Role Category'] + ' '
+ jobs_df['Functional Area'] + ' ' + jobs_df['Industry'] + ' ' + jobs_df['Job Title']
resumes_df['All_Skills'] = resumes_df['Skills'].apply(lambda x: ' '.join(x))

tfidf_vectorizer = TfidfVectorizer(stop_words='english')
job_tfidf = tfidf_vectorizer.fit_transform(jobs_df['Combined'])
resume_tfidf = tfidf_vectorizer.transform(resumes_df['All_Skills'])

cosine_similarities = cosine_similarity(resume_tfidf, job_tfidf)

def get_content_based_recommendations(resume_id, cosine_similarities,
top_n=3):
    sim_scores = list(enumerate(cosine_similarities[resume_id]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)[:top_n]
    recommended_jobs = [job[0] for job in sim_scores]
    return recommended_jobs

content_recommendations = get_content_based_recommendations(0,
cosine_similarities)
print("Content-Based Filtering Recommendations for Resume 0:",
content_recommendations)
```

CHAPTER 7

OUTPUTS AND FINAL RESULTS

urcareer front-end:

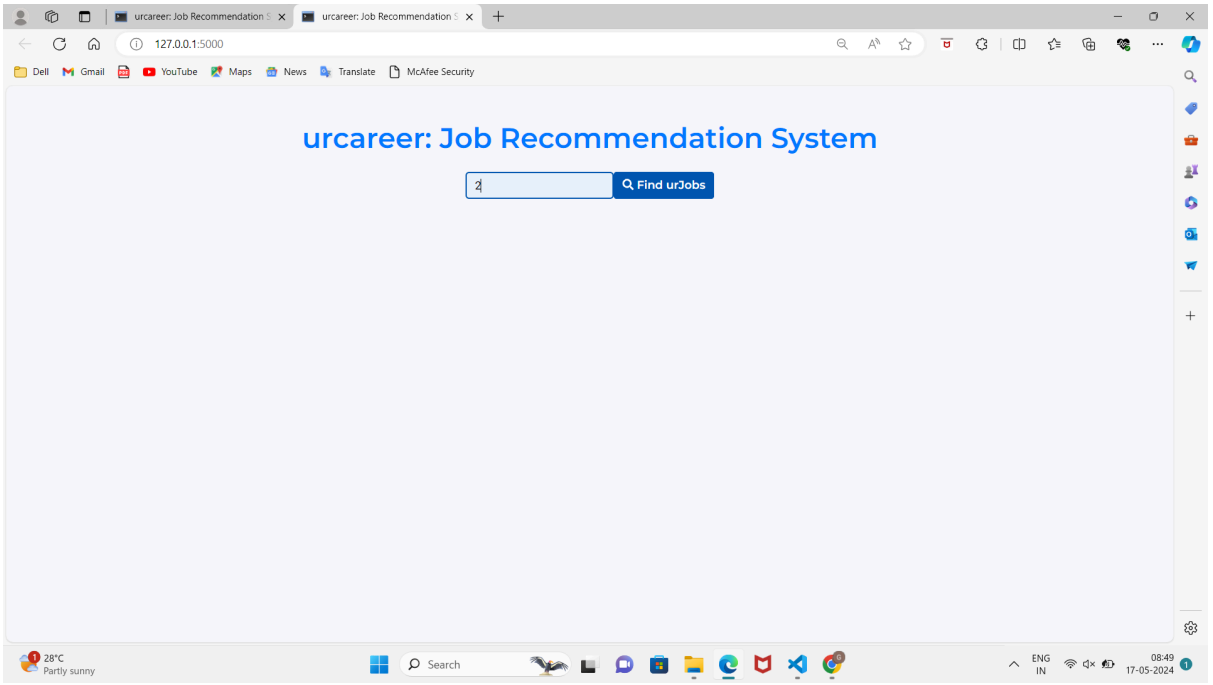


FIG. 5.1 Searching based on user-id

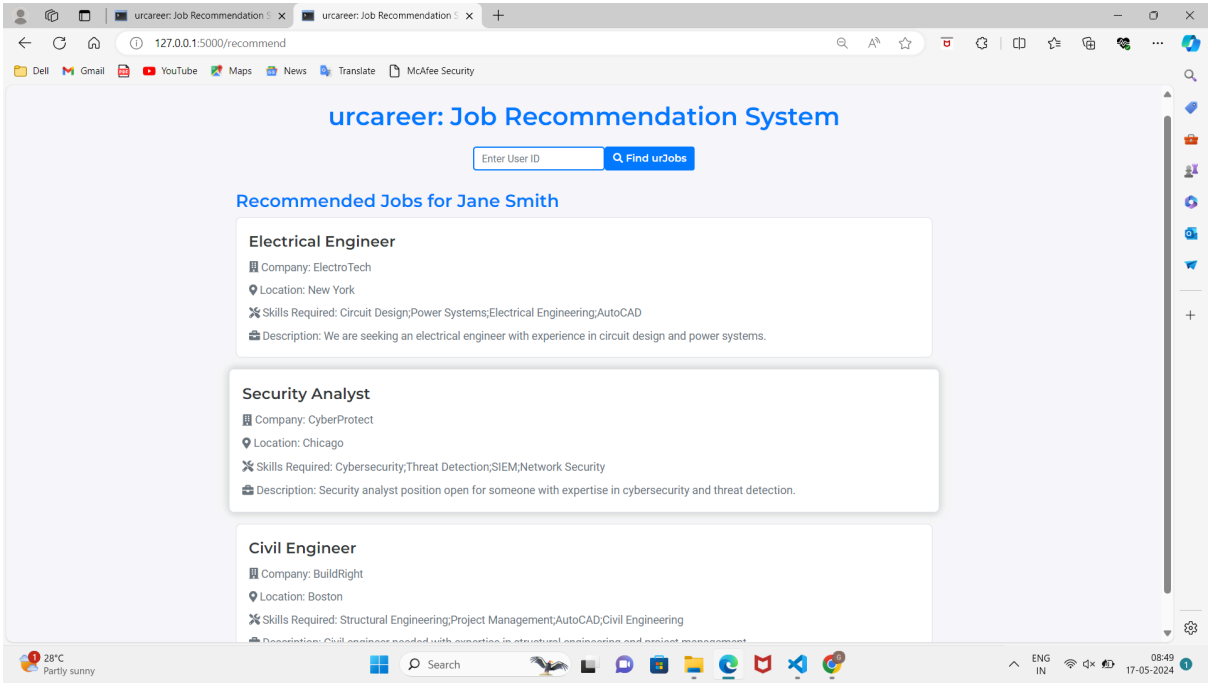


FIG. 5.2 Job Recommendations for user 1

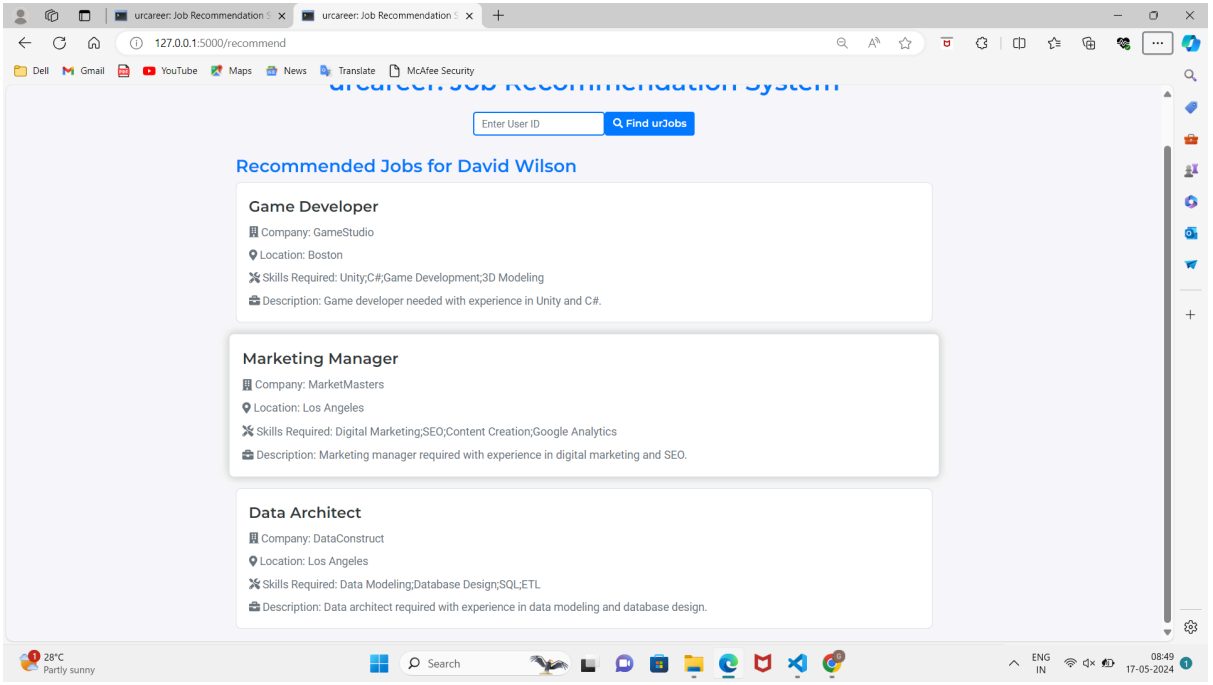


FIG. 5.3 Job Recommendations for user 2

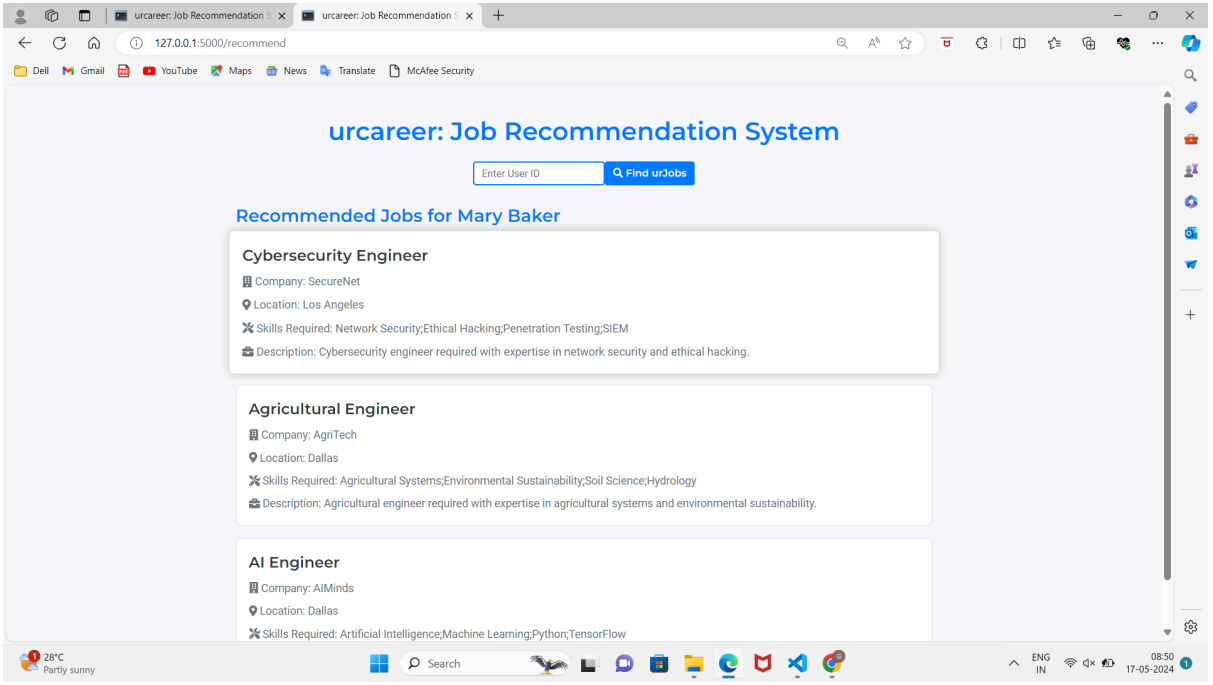


FIG. 5.4 Job Recommendations for user 3

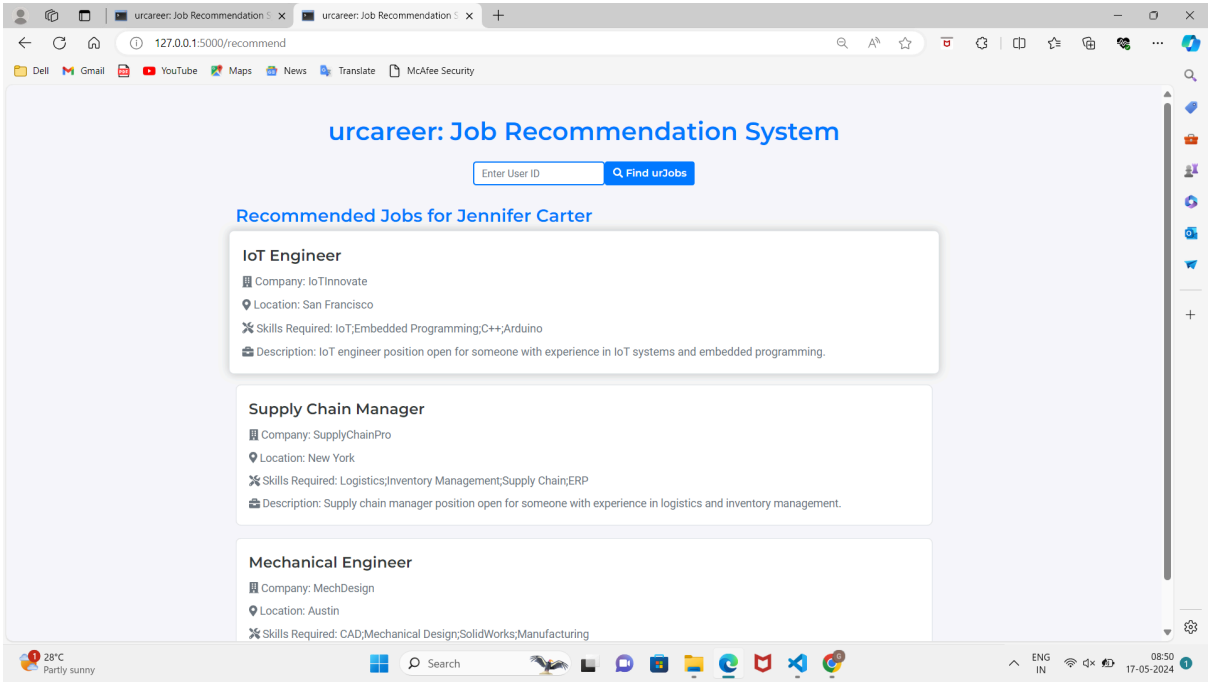


FIG. 5.5 Job Recommendations for user 4

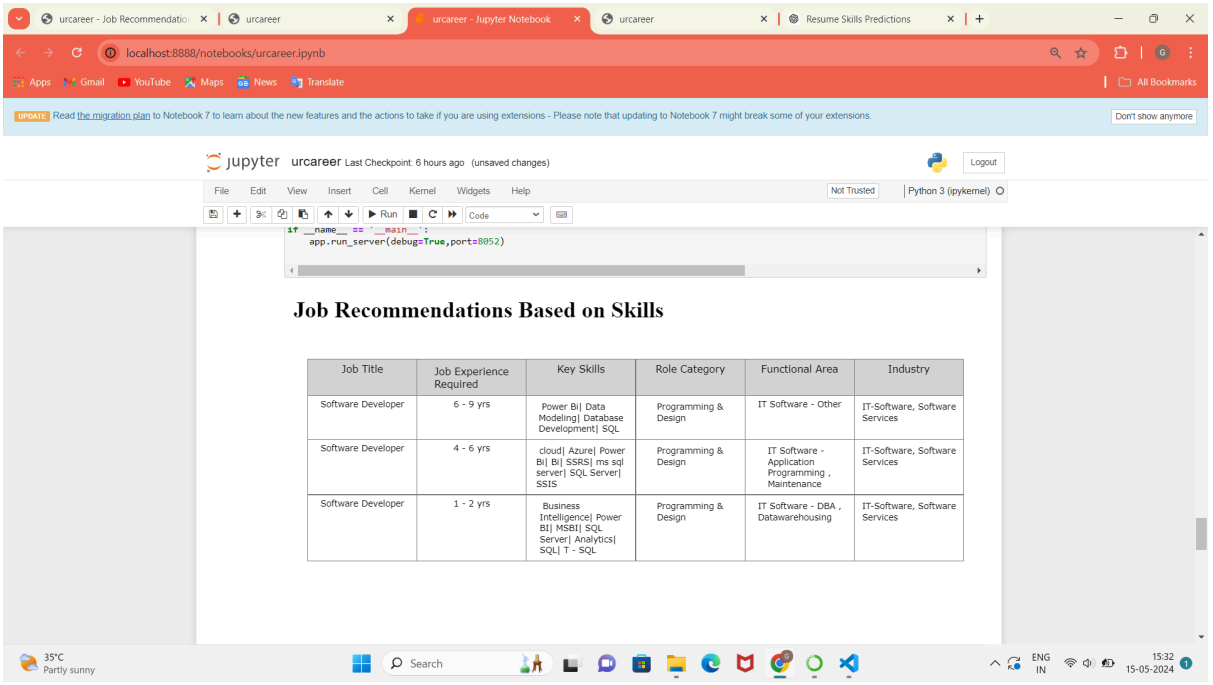
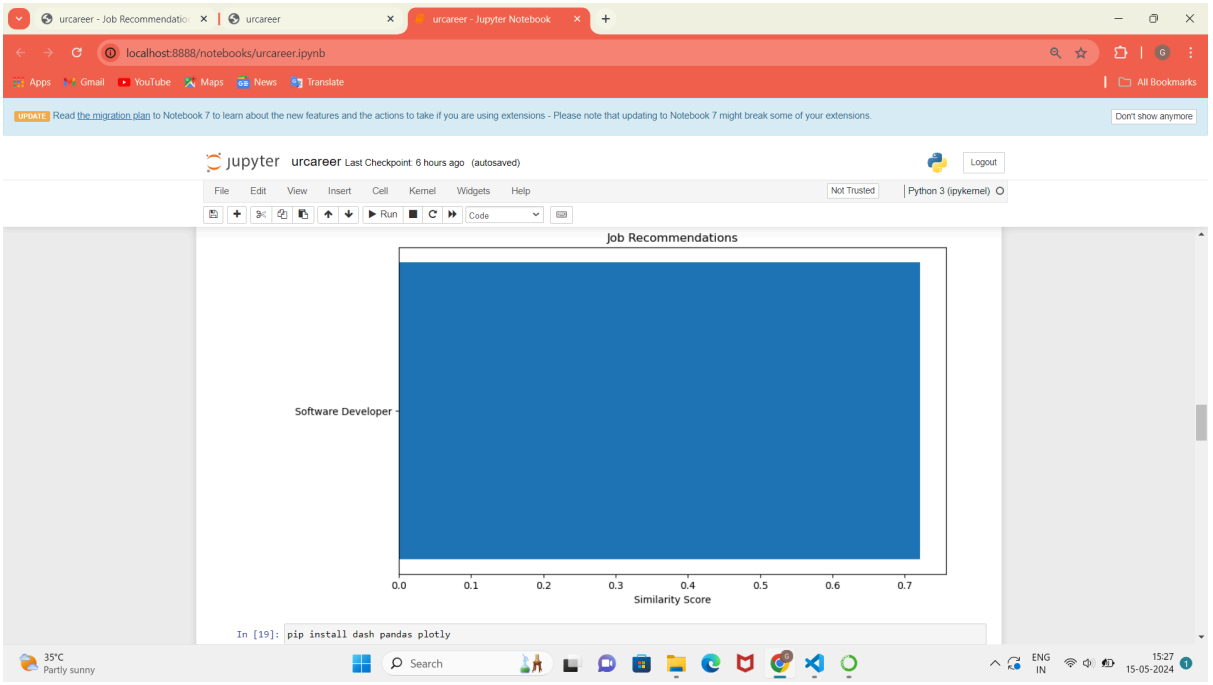


FIG. 5.6 Content-Based Filtering



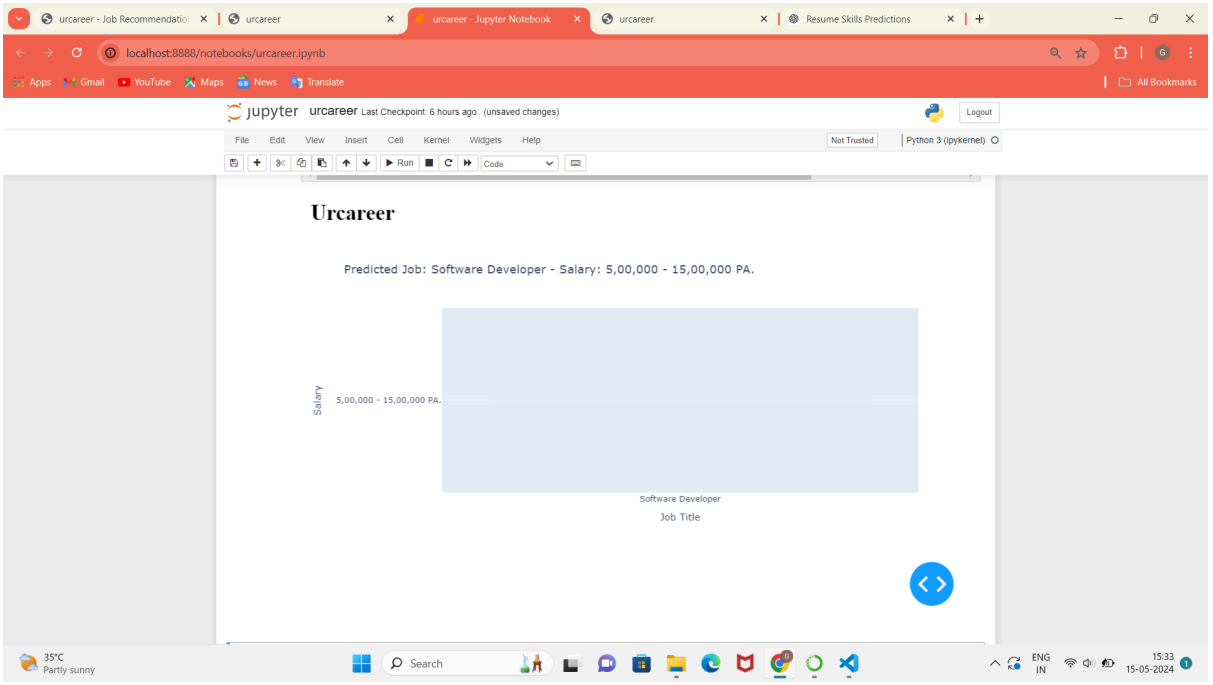


FIG. 5.7 Analytical Predictions of Jobs at urcareer

CHAPTER 8
CONCLUSION AND FUTURE ENHANCEMENTS

In conclusion, the urcareer: Job Recommendation System presents a comprehensive solution for assisting users in their job search endeavors. Through the integration of advanced machine learning algorithms, including hybrid content-based and collaborative filtering techniques, the system delivers personalized job recommendations tailored to individual preferences and expertise. The project successfully addresses the challenge of information overload in the job market by providing users with relevant and targeted job suggestions based on their profiles and interactions.

Looking ahead, several avenues for future enhancements and expansions exist to further enrich the capabilities of the system. Firstly, the integration of additional data sources, such as social media profiles or professional networks, could enhance the system's understanding of user preferences and improve recommendation accuracy. Furthermore, incorporating contextual information, such as industry trends or geographical considerations, could enable the system to offer more contextualized recommendations that align with evolving user needs and market dynamics.

Moreover, the implementation of a feedback loop mechanism, where users can provide explicit feedback on recommended jobs, could enable the system to iteratively refine its recommendations and adapt to evolving user preferences over time.Overall, the urcareer: Job Recommendation System represents a significant step forward in leveraging machine learning for personalized job recommendations, with ample opportunities for further innovation and enhancement in the future.

LIST OF REFERENCES

- Harvey & Paul Deitel & Associates, Harvey Deitel and Abbey Deitel, “Internet and World Wide Web - How ToProgram”, Fifth Edition, Pearson Education, 2011.

- Jeffrey C and Jackson, “Web Technologies A Computer Science Perspective”, Pearson Education, 2011.

- S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, Third Edition, 2015.

- Nils J. Nilsson, Artificial Intelligence: A New Synthesis (1 ed.), Morgan-Kaufmann, 1998. ISBN 978- 1558605350.

- Stephen Marsland, “Machine Learning – An Algorithmic Perspective”, Second Edition, Chapman and Hall/CRC Machine Learning and Pattern Recognition Series, 2014.

- Peter Flach, “Machine Learning: The Art and Science of Algorithms that Make Sense of Data‖”, First Edition,Cambridge University Press, 2012.

- Tom M Mitchell, “Machine Learning”, First Edition, McGraw Hill Education, 2013.

- Trevor Hastie, Robert Tibshirani and Jerome Friedman, “The Elements of Statistical Learning (ESL)”, 2nd edition,Springer, 2016. ISBN 978-0387848570.