Faculty of Computer Science and Technology

CAREER NAVIGATOR: EMPOWERING GRADUATES THROUGH DATA INSIGHTS

Najwa Watiqah binti Saifuddin

Bachelor of Computer Science with Honours (Information Systems)

2024

UNIVERSITI MALAYSIA SARAWAK

THESIS STATUS ENDORSEMENT FORM

TITLE Career Navigator: Empowering Graduates Through Data

ACADEMIC SESSION: 2024/2025

NAJWA WATIQAH BINTI SAIFUDDIN

(CAPITAL LETTERS)

hereby agree that this Thesis* shall be kept at the Centre for Academic Information Services, Universiti Malaysia Sarawak, subject to the following terms and conditions:

1. The Thesis is solely owned by Universiti Malaysia Sarawak
2. The Centre for Academic Information Services is given full rights to produce copies for educational purposes only
3. The Centre for Academic Information Services is given full rights to do digitization in order to develop local content database
4. The Centre for Academic Information Services is given full rights to produce copies of this Thesis as part of its exchange item program between Higher Learning Institutions [or for the purpose of interlibrary loan between HLI]
5. ** Please tick (√)

☐	CONFIDENTIAL	(Contains classified information bounded by the OFFICIAL SECRETS ACT 1972)
☐	RESTRICTED	(Contains restricted information as dictated by the body or organization where the research was conducted)
☑	UNRESTRICTED	

Validated by

(AUTHOR'S SIGNATURE)

(SUPERVISOR'S SIGNATURE)

Permanent Address

KAMPUNG SUNGAI UD,
96300 DALAT, SARAWAK

Date: 17/01/2025

Date: _____

Note * Thesis refers to PhD, Master, and Bachelor Degree
** For Confidential or Restricted materials, please attach relevant documents from relevant organizations / authorities

CAREER NAVIGATOR: EMPOWERING GRADUATES THROUGH DATA INSIGHTS

NAJWA WATIQAH BINTI SAIFUDDIN

This project is submitted in partial fulfilment of the requirements for the degree of Bachelor of Computer Science with Honours (Information Systems)

Faculty of Computer Science and Information Technology
UNIVERSITI MALAYSIA SARAWAK
2024

NAVIGATOR KERJAYA: MEMPERKASA GRADUAN MELALUI DATA INSIGHT

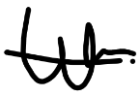
NAJWA WATIQAH BINTI SAIFUDDIN

Projek ini merupakan salah satu keperluan untuk Ijazah Sarjana Muda Sains Komputer dengan Kepujian (Sistem Maklumat)

Fakulti Sains Komputer dan Teknologi Maklumat
UNIVERSITI MALAYSIA SARAWAK
2024

DECLARATION

I hereby declare that this project is my original work. I have not copied from any other student's work or from any other sources except where due reference or acknowledgement is not made explicitly in the text, nor has any part had been written for me by another person.



…….

(NAJWA WATIQAH BINTI SAIFUDDIN) 17 JANUARY 2025

Matric No: 80329

ACKNOWLEDGEMENT

First and foremost, I would like to express my deepest gratitude to Allah, whose blessings and guidance have been my constant source of strength and perseverance throughout this journey. Without His divine support, completing this work would not have been possible.

I extend my heartfelt appreciation to my supervisor, Dr. Mohammad bin Hossin, for his invaluable guidance, encouragement, and patience throughout this process. His expertise and thoughtful advice have been instrumental in shaping this work and in helping me grow academically and personally. I am deeply grateful for your unwavering support and mentorship.

To my family, I owe everything your unconditional love, encouragement, and belief in me have been the foundation of my strength. Thank you for standing by me through every challenge, celebrating my successes, and offering unwavering support when times were tough. I could not have done this without you.

To my friends, thank you for being my pillars of encouragement and positivity. Your words of motivation and the moments of laughter we shared brought light to this journey. I am truly grateful for your companionship and support, which have made this achievement even more meaningful. Thank you all for being part of this milestone in my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Career Navigator: Empowering Graduates Through Data Insights addresses the persistent challenge of unemployment and underemployment among Computer Science (CS) and Information Technology (IT) graduates. These challenges arise from the mismatch between academic training and the rapidly evolving demands of the technology sector. Current career guidance systems often rely on generalized recommendations that fail to consider the unique qualifications, technical skills, and career aspirations of CS and IT students, leading to job mismatches and dissatisfaction. This project develops a web-based recommender system that employs a content-based filtering algorithm with a customizable threshold mechanism to provide tailored job recommendations for CS and IT graduates. User data, including academic background, technical skills, and personality traits, are matched with job postings retrieved from dynamic databases. The system incorporates a personality assessment to enhance job alignment, categorizing users as introvert, extrovert, or ambivert. The methodology involves phases such as data collection, preprocessing, feature extraction, algorithm development, recommendation generation, and iterative refinement through user feedback. The findings demonstrate that integrating academic, technical, and psychological attributes into job recommendation systems significantly enhances their accuracy and relevance. The system empowers graduates to explore personalised career paths, bridging the gap between their qualifications and the demands of the tech industry. By addressing gaps in traditional systems, such as lack of personalization and localization, this project offers a scalable solution for improving the employability of Malaysian CS and IT graduates. This study contributes to the field by providing a data-driven, user-centric approach to career guidance, addressing the unique needs of CS and IT graduates. It highlights the importance of combining academic qualifications, technical expertise, and personality insights to deliver precise and actionable job recommendations. The proposed system represents a significant step toward improving job search experiences and aligning education with employment in the fast-evolving technology landscape.

ABSTRAK

Career Navigator: Empowering Graduates Through Data Insights menangani isu pengangguran dan kekurangan pekerjaan dalam kalangan graduan Sains Komputer (CS) dan Teknologi Maklumat (IT). Ketidakpadanan antara latihan akademik dan keperluan industri teknologi yang sentiasa berubah menyebabkan banyak graduan menghadapi cabaran mencari pekerjaan yang sesuai. Sistem panduan kerjaya sedia ada sering memberikan cadangan umum yang tidak memperibadikan kelayakan unik dan kemahiran teknikal graduan, menyebabkan ketidaksesuaian pekerjaan. Projek ini membangunkan sistem cadangan kerjaya berasaskan web menggunakan algoritma content-based filtering dengan mekanisme ambang boleh laras untuk menyediakan cadangan pekerjaan yang diperibadikan. Data pengguna, termasuk latar belakang akademik, kemahiran teknikal, dan ciri personaliti, dipadankan dengan jawatan kosong daripada pangkalan data dinamik. Penilaian personaliti juga disertakan untuk mengkategorikan pengguna sebagai introvert, ekstrovert, atau ambivert, menjadikan cadangan lebih relevan. Sistem ini direka dengan metodologi berstruktur, termasuk pengumpulan data, prapemprosesan, pengekstrakan ciri, pembangunan algoritma, dan penambahbaikan berdasarkan maklum balas pengguna. Hasilnya menunjukkan bahawa gabungan atribut akademik, teknikal, dan psikologi meningkatkan ketepatan cadangan pekerjaan secara signifikan. Graduan dapat meneroka laluan kerjaya yang lebih sesuai dengan aspirasi mereka, menjembatani jurang antara pendidikan dan keperluan pekerjaan dalam sektor teknologi. Sistem ini menawarkan penyelesaian yang berskala, diperibadikan, dan relevan untuk pasaran pekerjaan Malaysia, meningkatkan kebolehkerjaan graduan CS dan IT. Kajian ini menunjukkan bahawa panduan kerjaya yang mengintegrasikan data akademik, kemahiran teknikal, dan psikologi dapat memperbaiki pengalaman pencarian pekerjaan dalam landskap teknologi yang sentiasa berkembang.

CHAPTER 1 INTRODUCTION

1.1 Preliminary

Graduate unemployment has become a significant global concern, with many graduates, particularly those from Computer Science (CS) and Information Technology (IT) backgrounds, struggling to secure jobs that align with their academic qualifications, technical skills, and career aspirations. The challenge lies in the growing gap between the skills gained during formal education and the rapidly evolving demands of the tech labour market. CS and IT graduates often face difficulties in finding suitable employment, leading to underemployment and dissatisfaction in their careers. This issue is especially prevalent in the technology industry, where rapid advancements in areas such as artificial intelligence, cybersecurity, and cloud computing constantly reshape the required skills and qualifications (Štimac & Tanasić, 2023). Furthermore, navigating an ever-changing job market remains a complex task for these graduates, who often lack personalised guidance and tools to match their specialized skills and interests with the right career opportunities (Rahhal et al., 2022).

Traditional career counselling systems have attempted to address these issues, but they are often limited by their static nature and broad classification methods. These systems typically offer generalized advice and fail to consider the specific qualifications, technical proficiencies, and personal preferences of CS and IT graduates. For example, many platforms rely on simple keyword-based matching algorithms that do not account for the comprehensive technical and personal profiles of users, resulting in mismatched job recommendations that fail to align with the unique skills, aspirations, and long-term career goals of these students (Karnewar et al., 2024). Consequently, many CS and IT graduates find themselves in roles that neither match

their expertise nor offer meaningful career growth, perpetuating underemployment and dissatisfaction.

To address these challenges, this study introduces *Career Navigator: Empowering Graduates Through Data Insights*, a personalised career guidance system designed to enhance the job search experience for CS and IT graduates. The project aims to bridge the gap between academic qualifications, technical skills, and the demands of the job market. By leveraging a content-based filtering algorithm enhanced with a threshold mechanism, Career Navigator provides tailored job recommendations that align with the specialized profiles of CS and IT graduates. Users input key data points, such as academic background, technical skills (e.g., programming, web development, machine learning), career preferences, and personality traits, allowing the system to generate personalised job suggestions that are both relevant and actionable.

This chapter outlines the research problem, methodology, and contributions of the study, while also explaining the structure of the report. The system's approach focuses on practical, data-driven insights, aiming to improve the alignment between CS and IT graduates' aspirations and industry demands. Ultimately, Career Navigator seeks to empower graduates to make informed career decisions, enhance their employability, and navigate the dynamic job market with greater confidence and precision.

1.2 Problem Statement

Graduate unemployment and underemployment are pressing global concerns, particularly affecting graduates from Computer Science (CS) and Information Technology (IT) backgrounds. These challenges stem from a significant disconnect between the technical skills taught in education and the rapidly evolving demands of the technology industry. Studies

indicate that many CS and IT graduates lack the specific, up-to-date competencies required by industries, leading to job mismatches and dissatisfaction in their careers (Rahhal et al., 2022). This issue is especially prevalent in fields such as data science, cybersecurity, and artificial intelligence, where technological advancements continuously reshape required skills and qualifications (Štimac & Tanasić, 2023).

Traditional career guidance systems have attempted to address these issues but often fall short due to their generalized, one-size-fits-all approach. These systems fail to consider the unique qualifications, technical proficiencies, and evolving job roles specific to CS and IT graduates (Sam, 2020). Additionally, they struggle to adapt to emerging industries and roles requiring expertise in areas like machine learning, cloud computing, and blockchain development (Štimac & Tanasić, 2023). As a result, many CS and IT graduates remain unaware of diverse career opportunities, limiting their employment prospects and career satisfaction (Al-Dhari & Al-Alawi, 2023).

Moreover, existing systems rely on outdated, static algorithms, such as keyword-based matching, that fail to address the complexity of technical preferences, skill gaps, and the fast-changing nature of the tech job market. This lack of personalization exacerbates underemployment, with CS and IT graduates often ending up in roles that neither align with their expertise nor offer meaningful career growth (Karnewar et al., 2024).

To bridge this gap, there is an urgent need for a personalised, data-driven career guidance system specifically tailored to the needs of CS and IT graduates. Such a system would integrate technical skills, academic achievements, and personal preferences with the dynamic demands of the tech job market. By providing tailored, actionable career recommendations, this

approach empowers graduates to explore meaningful career paths and improves their chances of securing employment in specialized fields (Guntupalli et al., 2024).

1.3 Aims and Objectives

The aim of this project is to develop a personalised career guidance system for graduating Computer Science (CS) and Information Technology (IT) students, enhancing their job search experience and aligning career opportunities with their qualifications and aspirations.

The objectives of the project are:

1. To develop a web-based system that provides tailored job recommendations based on academic background, technical skills, and personality traits.
2. To implement a content-based matching algorithm that aligns student profiles with relevant tech industry roles, considering academic achievements, skills, industry trends, and user feedback.
3. To improve the job search process by enabling users to adjust threshold preferences for more personalised and relevant recommendations.

1.4 Scope

The scope of this project, *Career Navigator: Empowering Graduates Through Data Insights*, is to develop a personalised career guidance system tailored specifically for graduating Computer Science (CS) and Information Technology (IT) students. The system focuses on providing tailored job recommendations by analysing user data such as academic background, technical skills, and personality traits. Using a content-based filtering algorithm with a threshold mechanism, the project aims to match students with relevant job opportunities within the tech industry. To ensure feasibility, the system relies on structured datasets from user inputs

and job market data, without incorporating real-time job updates or advanced machine learning models. The primary goal is to deliver a functional, data-driven platform that bridges the gap between education and employment for CS and IT graduates, with potential for future expansion based on user feedback and system performance.

1.5 Brief Methodology

This project employs a quantitative, data-driven approach to develop a personalised career guidance system tailored for Computer Science (CS) and Information Technology (IT) students. The system uses a content-based filtering algorithm with a threshold mechanism to match user profiles with relevant job opportunities. User data, including academic background (e.g., CGPA, certifications, and courses), technical skills (e.g., programming, data analysis, machine learning), and career preferences, is collected through a structured input process. Additionally, a personality assessment categorizes users as introverted, extroverted, or ambivert, enhancing the relevance of job recommendations.

Collected data undergoes preprocessing to ensure accuracy and consistency. This step involves cleaning incomplete entries, handling missing values, and standardizing formats. Key features from user profiles and job postings, such as required skills and qualifications, are extracted to enable precise matching between users and job opportunities (Siswipraptini et al., 2022).

The content-based filtering algorithm calculates similarity scores by comparing user profiles with job descriptions. A threshold mechanism filters out irrelevant recommendations, ensuring only highly relevant job opportunities are presented to users. The system generates ranked job suggestions, with the most aligned roles appearing at the top. Users can adjust

threshold preferences to refine results further and tailor recommendations to their specific career goals (Rahman et al., 2023).

User feedback plays a critical role in improving the system's performance. Feedback on job recommendations is used to refine the algorithm and adapt the system to changing user needs and market trends. This iterative process ensures the system remains responsive, accurate, and valuable for CS and IT graduates as they navigate the dynamic job market.

1.6 Significance of Project

The *Career Navigator: Empowering Graduates Through Data Insights* project addresses graduate unemployment among Computer Science (CS) and Information Technology (IT) students by providing personalised career guidance. Unlike traditional career counselling systems, it employs a content-based filtering algorithm with a threshold mechanism to offer tailored job recommendations based on academic qualifications, technical skills, and personality traits. This approach bridges the gap between education and the evolving tech job market, enhancing employability and career satisfaction (Sam, 2020).

The project refines career guidance frameworks by integrating personality psychology, addressing gaps in existing systems that overlook traits like introversion or extroversion. This holistic approach improves job-matching accuracy and supports meaningful career outcomes (Hakimi et al., 2011). Methodologically, the dynamic and customizable algorithm offers insights into optimizing similar systems for personalization in rapidly changing industries (Al-Dhari & Al-Alawi, 2023).

Practically, the system benefits students by reducing job mismatches, educators by aligning curricula with market demands, and policymakers by providing a scalable model for

data-driven career counselling. With immediate and long-term implications, *Career Navigator* is a practical solution for improving employability and reshaping career guidance in the tech industry (Siswipraptini et al., 2022).

1.7 Project Schedule

A well-structured project timetable is essential for the project's success. Key milestones are identified to keep the project on track and on schedule. The project takes two semesters to complete, with each component meticulously planned and monitored. A Gantt Chart is used to manage the development stages, which include data collecting, algorithm implementation, system testing, and evaluation. This tool provides a clear timeline and serves as a guideline to ensure timely progress and completion of the project.

TASK	START	END	OCTOBER 2024				NOVEMBER 2024				DECEMBER 2024				JANUARY 2025			
			WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 5	WEEK 6	WEEK 7	WEEK 8	WEEK 9	WEEK 10	WEEK 11	WEEK 12	WEEK 13	WEEK 14	WEEK 15	WEEK 16
Approved Project Title Selection	14-Oct-24	14-Oct-24																
Brief Project Proposal	28-Oct-24	9-Nov-24																
Submission of Approved Brief Proposal	9-Nov-24	9-Nov-24																
Full Research Proposal	10-Nov-24	27-Nov-24																
Submission of Full Proposal	28-Nov-24	28-Nov-24																
Chapter 1: Introduction	25-Nov-24	15-Dec-24																
Submission of Chapter 1	15-Dec-24	15-Dec-24																
Chapter 2: Literature Review	4-Dec-24	24-Dec-24																
Submission of Chapter 2	24-Dec-24	24-Dec-24																
Chapter 3: Methodology	24-Dec-24	5-Jan-25																
Submission of Chapter 3	5-Jan-25	5-Jan-25																
FYP 1 Final Report	5-Jan-25	17-Jan-25																
Final Amendment and Modification	14-Jan-25	16-Jan-25																
Submission of FYP1 Final Report	17-Jan-24	17-Jan-24																
FYP1 Presentation	23-Jan-25	25-Jan-25																

Figure 1.1 Project Schedule for FYP 1

TASK	START	END	MARCH 2025			APRIL 2025				MAY 2025					JUNE 2025			
			WEEK 0	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 5	WEEK 6	WEEK 7	WEEK 8	WEEK 9	WEEK 10	WEEK 11	WEEK 12	WEEK 13	WEEK 14	WEEK 15
Revised Structured of FYP Report	10-Mar-25	17-Mar-25																
Chapter 4: Implementation	17-Mar-25	1-May-25																
Submission of Chapter 4	1-May-25	1-May-25																
Chapter 5: Testing	2-May-25	20-May-25																
Submission of Chapter 5	20-May-25	20-May-25																
Chapter 6: Conclusion and Future Work	20-May-25	4-Jun-25																
Prepare Draft for Abstract Paper	2-Jun-25	3-Jun-25																
Submission of Chapter 6	4-Jun-25	4-Jun-25																
Final Report	5-Jun-25	20-Jun-25																
Compilation of FYP Report	5-Jun-25	16-Jun-25																
Final Amendment and Modification	16-Jun-25	20-Jun-25																
Submission of FYP Final Report	23-Jun-25	23-Jun-25																

Figure 1.2 Project Schedule for FYP2

1.8 Expected Outcome

The expected outcome of the *Career Navigator: Empowering Graduates Through Data Insights* project is a functional web-based system that provides personalised job recommendations for Computer Science (CS) and Information Technology (IT) graduates. Using a threshold-based filtering algorithm, the system aligns user profiles—including academic background, technical skills, and personality traits—with relevant job opportunities, improving employability by reducing mismatches between qualifications and job roles. This prototype will serve as a practical tool for enhancing career guidance, empowering students to make informed decisions, and supporting educators in aligning academic programs with market demands.

1.9 Report Outline

1.9.1 Chapter 1: Introduction

This chapter introduces the *Career Navigator: Empowering Graduates Through Data Insights* project, focusing on its objectives, significance, and scope. It explores the issue of graduate unemployment and underemployment among Computer Science (CS) and Information Technology (IT) students and highlights the limitations of traditional career guidance systems.

The chapter also explains how a personalised career guidance system, leveraging content-based filtering with a threshold mechanism, addresses these challenges. Key topics include the problem statement, methodology overview, and expected outcomes of the project.

1.9.2 Chapter 2: Literature Review

This chapter reviews existing career guidance systems, including traditional methods and AI-driven solutions, with an emphasis on their strengths and limitations for addressing the needs of CS and IT students. It evaluates how academic qualifications, technical skills, and personality traits are used to enhance personalised career recommendations. Additionally, the chapter explores relevant methodologies, including content-based filtering, and identifies gaps in existing systems to justify the proposed approach for the Career Navigator system.

1.9.3 Chapter 3: Methodology

This chapter describes the methodology used to develop the Career Navigator system. It details the phases of the project, including data collection, preprocessing, feature extraction, algorithm development, and recommendation generation. The chapter also discusses the content-based filtering algorithm with a threshold mechanism and highlights the role of user feedback in refining recommendations. Testing and evaluation methods are outlined to assess the system's performance and adaptability to user needs.

1.9.4 Chapter 4: Implementation

This chapter explains how the Career Navigator system was developed, including the setup of the development environment, integration of system modules, and the implementation of core functionalities such as user profiling, data preprocessing, and the recommendation

engine. It also describes the technologies used and includes screenshots of the interface to demonstrate the system's operational features.

1.9.5 Chapter 5: Evaluation, Testing and Result

This chapter presents the testing and evaluation of the Career Navigator system. It includes usability testing using the System Usability Scale (SUS), user acceptance testing (UAT), and sample recommendation outputs. The results are analysed to determine system effectiveness, usability, and alignment with user expectations.

1.9.6 Chapter 6: Conclusion and Future Works

This chapter summarizes the project outcomes, highlighting the system's success in delivering personalized job recommendations based on user profiles. It also outlines the limitations encountered and proposes future enhancements, such as integrating employer accounts, expanding job sources, and applying machine learning for improved recommendation accuracy.

CHAPTER 2 LITERATURE REVIEW

2.1 Introduction

This chapter reviews career guidance platforms, such as LinkedIn, JobStreet, and Hiredly, focusing on their methodologies, strengths, and limitations in addressing the needs of Computer Science (CS) and Information Technology (IT) graduates in Malaysia. These systems often emphasize academic and technical skills but lack personalization features, such as integrating personality traits into job recommendations. Additionally, the chapter explores the potential of content-based filtering with threshold mechanisms to enhance the precision and relevance of job matches. Key features such as academic background, technical skills, and psychological traits are analysed to identify gaps in existing systems, supporting the development of the proposed *Career Navigator* system tailored for CS and IT graduates.

2.2 Discussion on the Key Features

2.2.1 Academic Background

Academic qualifications play a vital role in job matching, especially for fresh graduates entering the workforce. Attributes such as CGPA, field of study, and certifications serve as key indicators of a candidate's suitability for specific roles, particularly in technical fields like Computer Science (CS) and Information Technology (IT). Research indicates a strong correlation between educational background and employability, especially in specialized domains. For instance, studies have shown that academic achievements are often closely linked to skill alignment, which, in turn, affects job success in technical fields (Mwita et al., 2024). Job recommendation systems often prioritize these academic attributes, matching candidates with roles that align with their qualifications. This ensures that fresh graduates are directed

toward positions that reflect their educational achievements and capabilities, fostering a better start to their careers.

2.2.2 Technical Skills

Technical skills are fundamental in job-matching systems, particularly for roles in IT and CS. These skills include programming languages, data analysis, project management, and specialized competencies such as database management and software engineering. Automated job-matching platforms typically evaluate these skills by analysing resumes, skill assessments, and direct inputs from candidates. These technical qualifications are then matched with job descriptions that require similar expertise, ensuring a strong fit between the candidate and the job. Research supports the notion that job-matching platforms that effectively evaluate technical skills increase candidates' chances of securing roles aligned with their qualifications. This improves the recruitment process by reducing mismatches and enhancing job satisfaction (Tuan et al., 2024). Effective integration of technical skills into job recommendation systems is essential, especially for fresh graduates who often have limited professional experience but possess specialized technical knowledge.

2.2.3 Psychological Traits

In addition to academic and technical skills, psychological traits significantly influence workplace dynamics and overall career satisfaction. Personality profiles, encompassing traits such as introversion, extroversion, and ambiversion, can affect an individual's suitability for a specific role or organisation. Wu et al. (2024) propose that incorporating personality assessments into job recommendation systems improves job satisfaction, boosts productivity, and decreases employee turnover. By integrating psychological characteristics, job-matching systems offer a comprehensive method for career placement, aligning a candidate's skills,

academic qualifications, and personality with appropriate job roles. This leads to more harmonious workplace environments and enhances long-term career satisfaction. This approach is particularly advantageous for recent graduates, as it aids in identifying roles that align with their personality, which is essential for acclimating to professional environments and thriving in their initial careers.

Figure 2.1 depicts a Venn diagram that elucidates the interrelations and shared attributes of introversion, ambiversion, and extroversion, offering a visual representation of their impact on career alignment. The model employs MBTI profiles and psychological insights to associate personality types with specific career paths, providing a more customised and effective recommendation approach. Furthermore, Figure 2.2 illustrates a Naïve Bayes-based personalised career path model employing the Myers-Briggs Type Indicator (MBTI) framework, as demonstrated by Siswipraptini et al. (2024). This model integrates psychological assessments into career recommendation systems, emphasising the importance of personality traits in evaluating career appropriateness. These visualisations illustrate how the amalgamation of machine learning algorithms with psychological data can improve the accuracy and relevance of career guidance, especially for recent graduates entering the workforce.

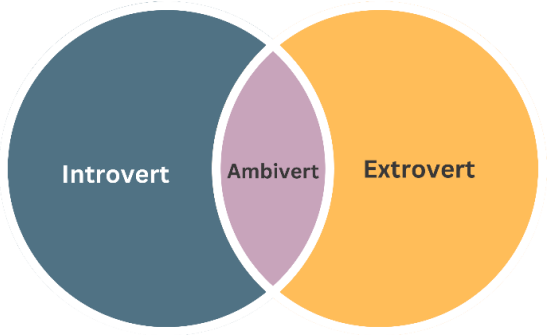


Figure 2.1 Venn Diagram Illustrating the Overlapping Characteristics of Introversion, Ambiversion, and Extroversion (Nickerson, 2024)

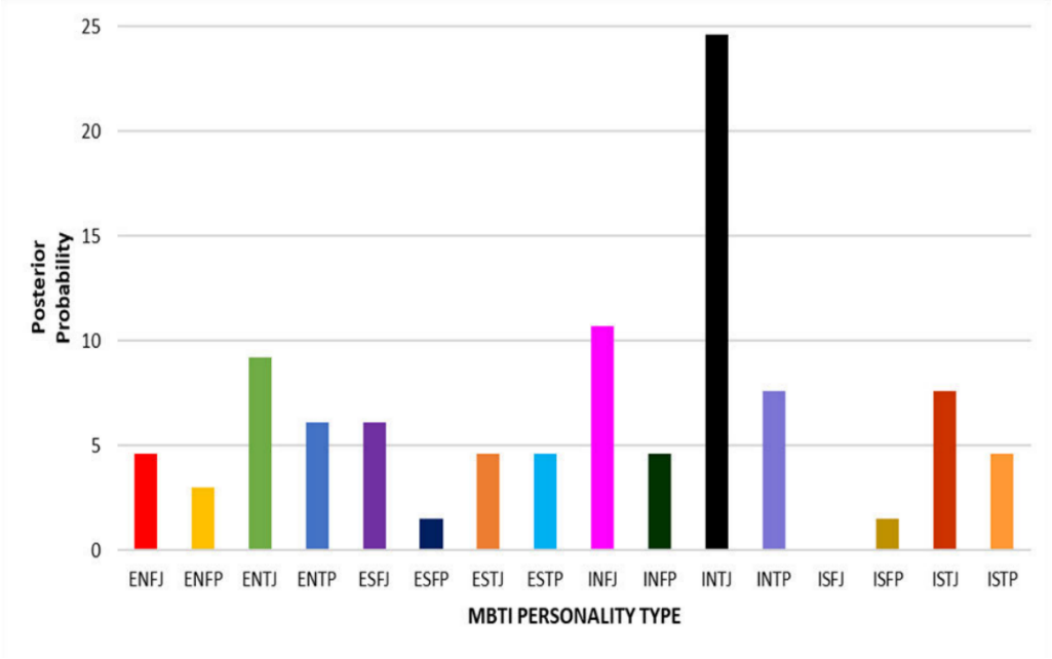


Figure 2.2 Naïve Bayes-based personalised career path model using MBTI (Siswipraptini et al., 2024).

2.2.4 Content-Based Filtering Recommender System

Content-based filtering is a highly effective method for personalised job recommendations, as it matches candidates' specific attributes—such as skills, qualifications, and experiences—with job descriptions. This approach analyses the relevance of each candidate's profile against the job requirements, ensuring that only highly compatible roles are recommended. To enhance recommendation accuracy, content-based filtering systems often use a threshold mechanism to ensure that candidates are only shown jobs that meet certain predefined criteria, such as skill proficiency or academic qualifications. Research has shown that content-based filtering is particularly efficient in narrowing down job matches based on specific attributes, thus improving job alignment and overall user satisfaction (Hawari et al., 2022). By tailoring job suggestions based on a detailed assessment of individual qualifications,

the system ensures that fresh graduates are presented with roles that closely match their capabilities and career goals, significantly improving the job search process.

Feature	Description	Examples/Indicators	Significance in Job Matching
Academic Background	Highlights a candidate's educational qualifications and achievements.	CGPA, certifications, field of study.	Ensures job recommendations align with academic achievements and career goals.
Technical Skills	Focuses on specialized skills required for technical roles, especially in CS and IT.	Programming languages, database management, software engineering.	Matches candidates to roles requiring specific expertise, reducing skill mismatches.
Psychological Traits	Incorporates personality attributes to evaluate workplace compatibility and satisfaction.	Introversion, extroversion, ambiversion (via MBTI, Big Five).	Aligns candidates to organizational culture and roles, improving long-term satisfaction (Wu et al., 2024).

Table 2.1 Key Features of Academic, Technical, and Psychological Traits

2.3 Review of Similar Existing System

2.3.1 JobStreet

JobStreet is a prominent job-matching platform in Malaysia, widely recognized for its extensive database and localized job listings. It has become a popular choice among fresh graduates due to its strong alignment with the Malaysian job market. The platform offers a wide variety of opportunities, making it highly relevant for local graduates. However, while JobStreet excels in providing accessible job listings, it faces significant limitations in addressing the unique needs of fresh graduates, particularly those from technical fields such as Computer Science (CS) and Information Technology (IT).

One strength of JobStreet is its user-friendly interface, which simplifies the job search process and makes it accessible to users who may not be familiar with complex job boards (Thali et al., 2023). Figure 2.3.1 illustrates the classification of job roles on JobStreet, showcasing its extensive range of job categories. While this broad categorization provides opportunities across diverse industries, it highlights the platform's generalized approach, which often fails to meet the specific needs of fresh graduates in technical fields such as Computer Science and Information Technology (Kamaruddin et al., 2019). Despite these strengths, JobStreet lacks the level of personalization required to meet the diverse needs of fresh graduates in dynamic technical fields. The platform's broad approach to job listings often fails to match candidates' technical skills and academic backgrounds with the most suitable roles (Alsaif et al., 2022). For example, it relies heavily on resume data and listed skills but does not consider important factors such as career aspirations, specific interests, or potential for long-term growth (Rashid et al., 2022).

Moreover, JobStreet focuses on traditional job roles and immediate job placement, which may not reflect the evolving skill sets required in fields like AI development or cloud computing (Nguyen, 2013). Fresh graduates, particularly from technical disciplines, are expected to possess a combination of technical and soft skills, such as problem-solving, communication, and adaptability. However, the platform's static matching criteria do not account for these attributes, nor does it provide tools for long-term career development (Siswipraptini et al., 2024). This limitation leaves specialized roles, such as AI Developers or Data Scientists, underserved.

These challenges highlight the need for a more personalised and adaptive job-matching platform, such as the proposed *Career Navigator*. Unlike JobStreet, the *Career Navigator* system integrates academic qualifications, technical skills, soft skills, and career aspirations into its recommendation algorithm. This approach ensures tailored recommendations for emerging tech roles and provides tools for tracking skill growth and aligning with long-term career paths (Kamaruddin et al., 2019; Alsaif et al., 2022).

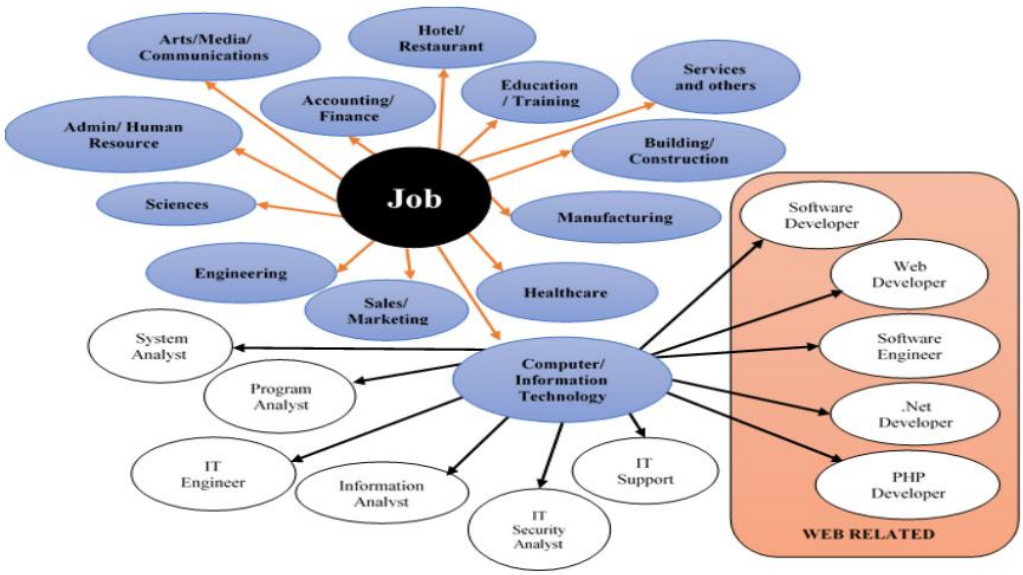


Figure 2.3 The classification of job roles on JobStreet, as shown in Kamaruddin et al. (2019)

2.3.2 LinkedIn

Global professional networking platform LinkedIn combines job recommendations with social networking. Uses content-based filtering and social network insights to recommend jobs based on skills, endorsements, and connections. LinkedIn is great for experienced professionals due to its global network and personalised job recommendations, but new graduates, especially in Malaysian technical fields like CS and IT, may find it difficult.

LinkedIn masters professional networking. Networking with organisations, industry leaders, and peers worldwide boosts professional reach and job market visibility (Kamaruddin et al., 2019). Skills endorsements and profile data help the platform match users' skills and qualifications to jobs (Rashid et al., 2022). LinkedIn offers graduates outside Malaysia international opportunities (Kenthapadi et al., 2017).

LinkedIn restricts recent graduates entering the workforce. Recent graduates without professional experience are disadvantaged by the platform's focus on endorsements and work experience (Rashid et al., 2022). The algorithm favours experienced professionals, making it difficult for recent graduates with strong technical skills but little work experience to get relevant job recommendations. LinkedIn values networking over academics and technical skills, which CS and IT graduates need for technical jobs (Kamaruddin et al., 2019). Programming or data analysis graduates without professional endorsements or project experience may be overlooked.

LinkedIn lacks Malaysian job market localisation, another drawback. Its global audience causes the platform's algorithms to overlook local industry trends, regulatory requirements, and cultural differences. CS and IT graduates in Malaysia may have trouble finding tech industry-specific jobs (Kenthapadi et al., 2017). LinkedIn lacks long-term career

development tools like growth planning, skill tracking, and path exploration. This gap is problematic because early-career professionals need guidance in the dynamic tech job market (Rashid et al., 2022).

LinkedIn's limitations necessitate a Malaysian CS and IT graduate job-matching platform. The proposed Career Navigator system prioritises academic and technical skills over work experience, which many recent graduates lack. The system will focus on localised job opportunities based on Malaysian industry trends and culture for relevance and accessibility (Kamaruddin et al., 2019). Career development tools will help graduates choose careers, plan long-term growth, and match skills to industry needs. Career Navigator addresses these gaps to give Malaysian graduates a personalised and holistic job-matching experience (Rashid et al., 2022).

2.3.3 Hiredly

Hiredly (formerly WOBB) is a job portal that emphasizes cultural fit, prioritizing candidates' personal values and company cultures over qualifications and work experience. Its innovative "visual-first" job listings create an intuitive and immersive job search experience, but this approach poses challenges for fresh graduates, particularly in technical fields like Computer Science (CS) and Information Technology (IT).

Hiredly's key strength is its focus on cultural alignment. By prioritizing workplace values and environments, it helps candidates find roles where they can thrive, improving job satisfaction and retention (Sam, 2020). Its visually appealing interface also strengthens relationships between job seekers and employers by effectively showcasing company branding and work culture, making the job search process more engaging (Siswipraptini et al., 2022).

Despite its strengths, Hiredly has notable limitations for fresh CS and IT graduates. The platform's reliance on basic career interests and resumes neglects academic qualifications, certifications, and technical skills, making it difficult to match candidates with roles that align with their expertise (Rahman et al., 2023). Additionally, the emphasis on cultural fit may overshadow technical competencies, leaving niche roles like AI Developers or Data Scientists underserved (Siswipraptini et al., 2022). Furthermore, the platform offers minimal support for long-term career development, which is essential for navigating fast-evolving tech industries (Karnewar et al., 2024).

The proposed Career Navigator system addresses these gaps by integrating academic qualifications, technical skills, and certifications into its recommendation algorithm while still considering cultural fit. It also supports long-term career growth through tools for skill tracking and alignment with industry demands, offering a balanced and personalised job-matching experience (Karnewar et al., 2024; Rahhal et al., 2022). Hiredly's "visual-first" interface, as shown in Figure 2.3.3, highlights its emphasis on cultural alignment and branding, providing a more engaging user experience (Siswipraptini et al., 2022).

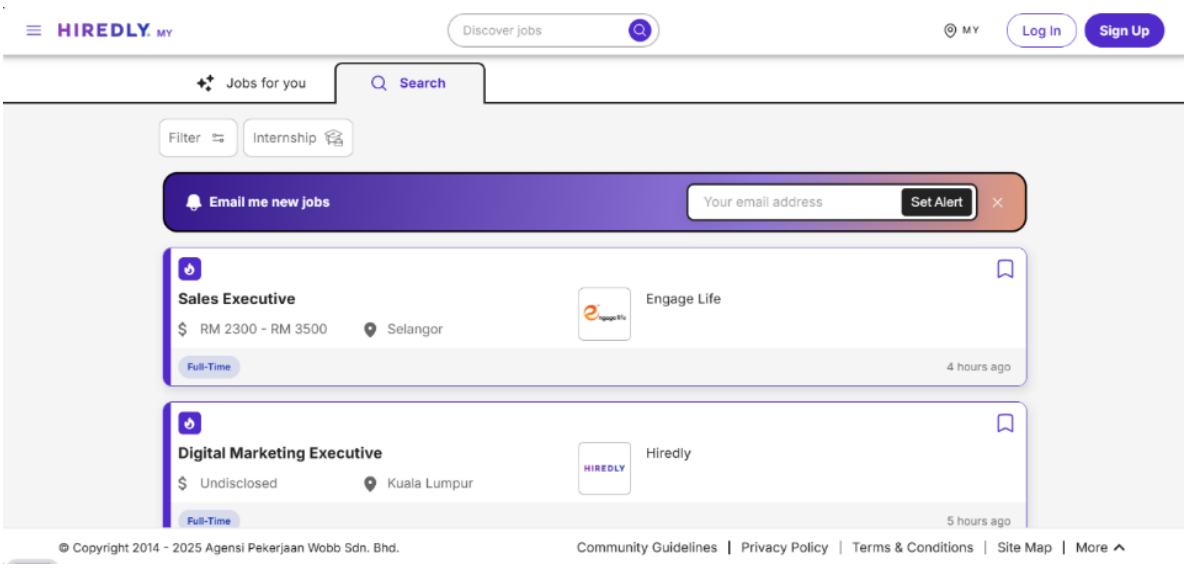


Figure 2.4 Example of Hiredly's job listing interface (Hiredly, 2023)

2.4 Comparison of Existing Systems and Proposed System

2.4.1 Comparative Analysis

The comparative analysis of existing job recommendation systems and the proposed Career Navigator system reveals key differences in their approaches.

Aspect	**Hiredly**	**JobStreet**	**LinkedIn**
Description	Hiredly, formerly WOBB, is Malaysia's job board that emphasizes candidate-employer cultural fit, featuring job listings that showcase company culture and workplace.	JobStreet, Southeast Asia's leading online employment platform, connects employers and job seekers with a vast local job database across industries.	LinkedIn is a global job listing and social networking platform that offers professional profiles, peer networking, and job listings worldwide.
Advantages	Prioritizing cultural fit, visual-first job listings, and a localized focus tailored to the Malaysian job market.	Provides an extensive job database across Southeast Asia, a user-friendly interface, and localized content tailored to the local market.	Offers global job access, professional networking, and skill endorsements to boost profile credibility.
Disadvantages	Primarily focuses on the Malaysian market,	Relies on keyword matching with limited	Emphasizes experience and

	with limited emphasis on technical skills and international opportunities.	personalization, overlooking personality traits and cultural fit.	endorsements, risking bias against fresh graduates, while potentially overwhelming users with excessive content.
Key Features	Offers cultural fit assessment tools, visual job listings with rich media, and localized support for Malaysian employers and job seekers.	Features comprehensive job listings, a resume builder, and employer insights to assist candidates.	Provides detailed professional profiles, global networking tools, and personalised job alerts.

Table 2.2 Comparative Analysis of Existing Recommendation System and Proposed System

2.4.2 Insights from Comparison

The comparison highlights Hiredly, JobStreet, and LinkedIn's unique strengths and limitations. Hiredly excels at emphasising cultural fit and providing a visually engaging job search experience, but it falls short on technical qualifications and international opportunities. JobStreet has a large database and localised support, making it extremely accessible; however, its reliance on keyword matching and lack of personality considerations limit its effectiveness for personalised recommendations. LinkedIn offers unparalleled global networking and skill

endorsements, but it prioritises experienced professionals, putting recent graduates at a disadvantage.

These findings highlight the need for a platform that combines Hiredly's cultural focus, JobStreet's extensive database, and LinkedIn's networking tools, while also filling gaps in personalisation, technical skill matching, and localised relevance. The proposed Career Navigator addresses these gaps by combining academic qualifications, technical skills, personality traits, and adjustable threshold functionality to provide a comprehensive, tailored job-matching experience for recent CS and IT graduates.

2.5 Identifying Gaps in Existing Research

A review of literature and job recommendation platforms revealed critical gaps for fresh CS and IT graduates, highlighting the need for a more personalized and comprehensive job-matching platform. Current platforms like LinkedIn and Indeed often ignore academic qualifications, technical skills, and personality traits, instead prioritizing work experience and career interests. This approach fails to accommodate CS and IT graduates, whose employability relies on technical expertise such as programming languages, database management, and software development (Siswipraptini et al., 2022; Rahhal, 2022). Additionally, these platforms lack tailored career guidance, leaving recent graduates unsupported as they enter the workforce.

Most platforms are designed for experienced professionals, often neglecting the needs of fresh graduates with strong theoretical and technical skills but limited professional experience. This work history bias makes it difficult for graduates to find jobs that align with their skills and qualifications (Sam, 2020). The lack of personalization is another significant issue. Platforms without customizable thresholds or preferences fail to deliver relevant

recommendations for fresh graduates, particularly in fast-evolving fields like AI, Data Science, and Cloud Computing (Štimac & Tanasić, 2023; Mishra & Jain, 2023).

Global platforms like LinkedIn and Indeed also fail to address Malaysian job market nuances. Their algorithms lack localization, ignoring local industry trends, cultural norms, and regulatory requirements, leaving Malaysian CS and IT graduates underserved (Rahman et al., 2023). To address these gaps, the Career Navigator system integrates academic, technical, and psychological data, allowing for personalized recommendations. Customizable thresholds enable graduates to refine their job search, while localized data connects them to industry-specific opportunities, bridging the gap between education and employment.

2.6 Summary

This chapter reviewed JobStreet, LinkedIn, and Hiredly, highlighting their strengths like broad job databases, global networking, and cultural alignment but highlighting critical gaps in personalisation, localisation, and data integration. These platforms often fail to integrate academic qualifications, technical skills, and personality traits, limiting their effectiveness for fresh computer science and IT graduates. The proposed Career Navigator system provides personalised, Malaysian-specific job recommendations to fill these gaps. The system uses academic, technical, and psychological data and customisable thresholds to improve employability, reduce job mismatches, and improve job-matching for recent graduates.

CHAPTER 3 METHODOLOGY

3.1 Introduction

This chapter discusses the methodology used to develop the Career Navigator, a web-based job recommendation system designed for CS and IT graduates. The methodology includes literature review, data preparation, system architecture, system development using the Waterfall Model, system evaluation, and documentation. The Waterfall Model guides the system development through five phases: system requirements, design, implementation, testing, and deployment. Informed by the gaps identified in Chapter 2, the methodology incorporates content-based filtering with customizable thresholds and localized data integration to provide personalized and relevant job recommendations tailored to user profiles and the Malaysian job market.

3.2 System Development Methodology

The Career Navigator system uses a structured methodology guided by the Waterfall Model to ensure systematic development and minimize risks (Kramer, M., 2018). The methodology, illustrated in Figure 3.1, includes literature review, system architecture design, system development, evaluation, and documentation. Each step is carefully planned to ensure seamless integration and alignment with the project's objectives. Insights from Chapter 2 inform the integration of academic qualifications, technical skills, and personality traits into the system's content-based filtering algorithm with customizable thresholds. The system development life cycle progresses through five sequential phases: requirements, design, implementation, testing, and deployment. Evaluation focuses on assessing quality, usability, and efficiency, while the documentation phase includes the preparation of a thesis report, research paper, and presentation slides to detail the project. By following the methodology

outlined in Figure 3.1, the system ensures personalized and localized job recommendations tailored to CS and IT graduates in Malaysia.

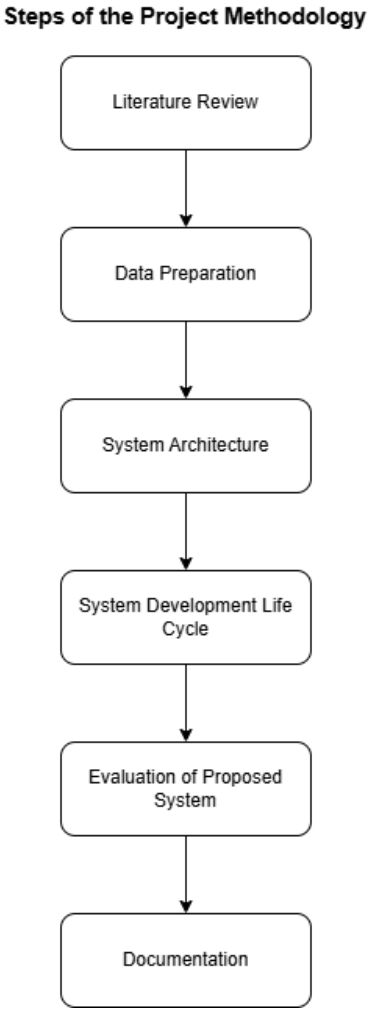


Figure 3.1 Steps of Project Methodology

3.3 Step 1: Literature Review

This step bridges the insights from Chapter 2 to the methodology adopted for developing the Career Navigator system. Chapter 2 identified significant gaps in existing job recommendation platforms, such as JobStreet, LinkedIn, and Hiredly, including insufficient personalization, limited integration of academic and psychological traits, and a lack of localization for niche technical roles. These findings directly informed the design of the Career

Navigator, which incorporates a content-based filtering algorithm, a customizable threshold mechanism, and detailed user profiling to address these issues.

The integration of academic qualifications and psychological traits is a central feature of the system's design. Research from Chapter 2 (e.g., Wu et al., 2024) underscores the importance of personality traits, such as introversion, extroversion, and ambiversion, in improving job matching accuracy and user satisfaction. This insight inspired the inclusion of a personality assessment to better align users with roles suited to their preferences and workplace compatibility. Similarly, the demonstrated success of content-based filtering in platforms like LinkedIn (Kenthapadi et al., 2017) guided its adoption as the core algorithm, enhanced by threshold adjustments to improve flexibility and customization.

Additionally, the system addresses localization gaps for Malaysian CS and IT graduates by incorporating real-time job data through APIs and focusing on industry-specific trends. This ensures that the Career Navigator delivers personalized and contextually relevant job recommendations tailored to users' academic qualifications, technical skills, and career aspirations. By addressing these challenges, the methodology builds a robust foundation for a system designed to enhance employability and career alignment for fresh graduates in the Malaysian tech sector.

3.4 Step 2: Data Preparation

Data preparation is a crucial phase in developing the Career Navigator system, ensuring that raw data is transformed into clean, actionable inputs for generating personalized job recommendations. This process, illustrated in Figure 3.2, consists of three interconnected components: data collection, data preprocessing, and feature extraction.

The first step, data collection, involves gathering information from two primary sources: user inputs and job posting databases. User data is collected through structured questionnaires designed specifically for CS and IT graduates. These questionnaires capture comprehensive details such as academic background, including CGPA, field of study, and certifications, as well as technical skills like programming languages, data analysis, machine learning, and project management. To enhance job matching precision, the questionnaire also includes a personality assessment, categorizing users as introverted, extroverted, or ambiverted. By incorporating academic, technical, and personality traits, the system can deliver recommendations tailored to both workplace preferences and individual strengths (Siswipraptini et al., 2024). For job postings, data is dynamically sourced using APIs from reputable platforms such as LinkedIn, JobStreet, and Glassdoor. These APIs provide access to real-time job descriptions, required skills, and qualifications. Labour market trends and reports further enrich the database, ensuring that the system reflects current industry demands and emerging roles (Adeagbo et al., 2019).

The second step, data preprocessing, focuses on improving the consistency and reliability of the collected data. Initially, data cleaning is performed to remove irrelevant, incomplete, or redundant information. For instance, outdated job postings, duplicate entries, and inconsistencies in formatting are addressed during this phase. Handling missing values is another critical task, where gaps in user profiles, such as missing CGPA values, are resolved using statistical techniques like median imputation. Records with significant omissions, such as incomplete job descriptions, are excluded to preserve data quality. Finally, standardization ensures uniform formatting across datasets, such as consolidating skill labels like “Java” and “Java Programming” into a single standard term. This step also ensures compatibility between

user profiles and job postings, enhancing the precision of the recommendation algorithm (Bansal & Jain, 2020; Siswipraptini et al., 2024).

The final step, feature extraction, identifies and organizes the most relevant attributes from user profiles and job postings. For user profiles, the system extracts academic achievements, technical skills (e.g., Python, cloud computing, project management), and soft skills like communication and teamwork. Personality traits, categorized as introvert, extrovert, or ambivert, further enrich the user profile, providing deeper insights into workplace compatibility. Job postings are analyzed to extract technical requirements, qualifications, certifications, and specific roles, ensuring precise alignment with user capabilities. As shown in Figure 3.2, these steps ensure the data is transformed into structured attributes, forming the foundation for the recommendation engine. Accurate feature extraction is essential for determining compatibility between users and job postings, as emphasized by Kumar et al. (2022).

By seamlessly integrating data collection, preprocessing, and feature extraction, the system transforms raw inputs into actionable attributes. These attributes allow the recommendation algorithm to compute similarity scores, matching user profiles with job postings. For example, a user proficient in Python can be effectively matched with a job requiring Python expertise. This comprehensive approach to data preparation ensures that the Career Navigator system delivers accurate, relevant, and personalized job recommendations, significantly improving user satisfaction and employability outcomes (Mulay et al., 2022).

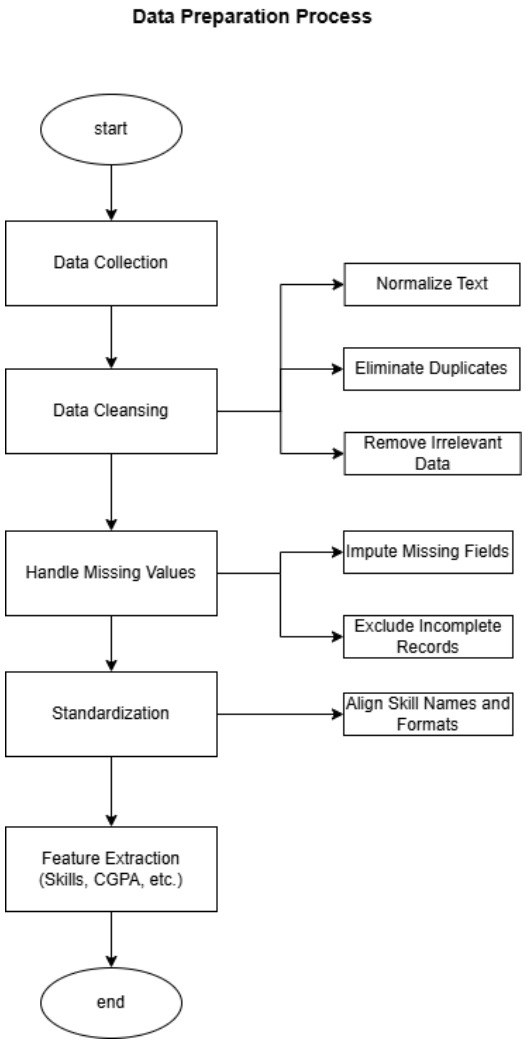


Figure 3.2 Data Preparation Process

3.5 Step 3: System Architecture

The web-based Career Navigator system is designed with a modular three-layer architecture: Input Layer, Processing Layer, and Output Layer to ensure efficient operations, scalability, and a seamless user experience. This layered approach facilitates the collection, processing, and delivery of personalized job recommendations tailored to CS and IT graduates.

The Input Layer collects data from two primary sources: user profiles and job postings. User profiles include academic background (e.g., CGPA, certifications), technical skills (e.g., programming languages, project management), and personality traits assessed through a simple

questionnaire, categorizing users as introverted, extroverted, or ambiverted. Job postings are sourced dynamically via APIs from platforms like LinkedIn and JobStreet, ensuring up-to-date details on job descriptions, required skills, and qualifications. This layer ensures accurate and reliable data collection, forming the foundation for relevant job matches (Shimpi et al., 2024).

The Processing Layer performs core system operations, including data preprocessing to clean, standardize, and handle missing values, followed by feature extraction to identify actionable attributes such as skills and qualifications. A content-based filtering algorithm computes similarity scores between user profiles and job postings. The customizable threshold mechanism allows users to set a match relevance score, filtering out less relevant job postings. For example, a user skilled in Python and SQL with a threshold of 75% would only see job matches exceeding this score. This layer is critical for ensuring accurate, timely, and relevant recommendations, as emphasized by Rahhal et al. (2022).

The Output Layer delivers job recommendations in a ranked format through an intuitive and visually appealing user interface. Job titles, company names, match percentages, and required skills are displayed, with options for users to sort or filter recommendations by location, industry, or other preferences. This layer minimizes cognitive load and enhances user satisfaction by prioritizing relevant job postings (Mulay et al., 2022).

Figure 3.3 illustrates the Career Navigator's three-layer architecture, where the Input Layer collects user and job data, the Processing Layer handles data operations and recommendation generation, and the Output Layer delivers ranked job suggestions. This structured design ensures scalability, usability, and alignment with the personalized needs of CS and IT graduates in the Malaysian job market.

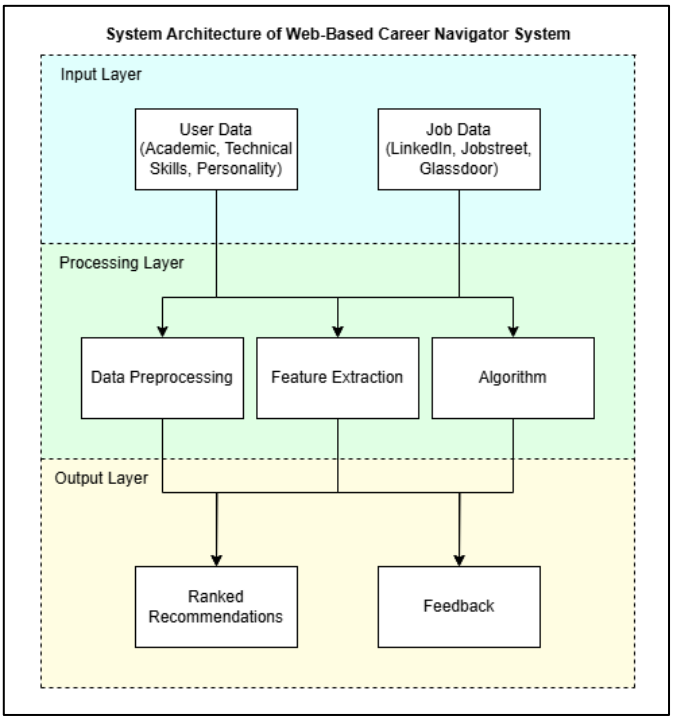


Figure 3.3 System Architecture of Web-Based Career Navigator System

3.6 Step 4: Development Approach: Waterfall Model

The Waterfall model is adopted as the methodology for the System Development Life Cycle (SDLC) in developing the web-based Career Navigator system. This model is a structured and sequential approach where each phase must be completed before the next begins, ensuring a clear and organized workflow. The Waterfall model comprises five distinct phases: system requirement, system design, system implementation, system testing, and system deployment, as illustrated in Figure 3.4. This step-by-step progression is particularly suitable for this project, as it ensures that all requirements are thoroughly addressed before moving forward, reducing risks and ensuring a systematic development process.

In the system requirement phase, the necessary components and features of the Career Navigator are identified and analysed. This includes gathering data about user profiles, such as academic qualifications, technical skills, and personality traits, alongside job postings with detailed requirements. The system design phase involves creating UML diagrams and mock-up

interfaces to outline the system's structure and user interaction flow. In the implementation phase, the system's core components, such as data processing and the content-based filtering algorithm, are developed. During the testing phase, the system undergoes rigorous functional and usability testing to ensure accuracy and reliability. Finally, in the deployment phase, the web-based Career Navigator is launched and made accessible to users, ensuring it is ready for practical use and future updates. This systematic approach ensures that the system aligns with its objectives and delivers a robust, user-centric solution.

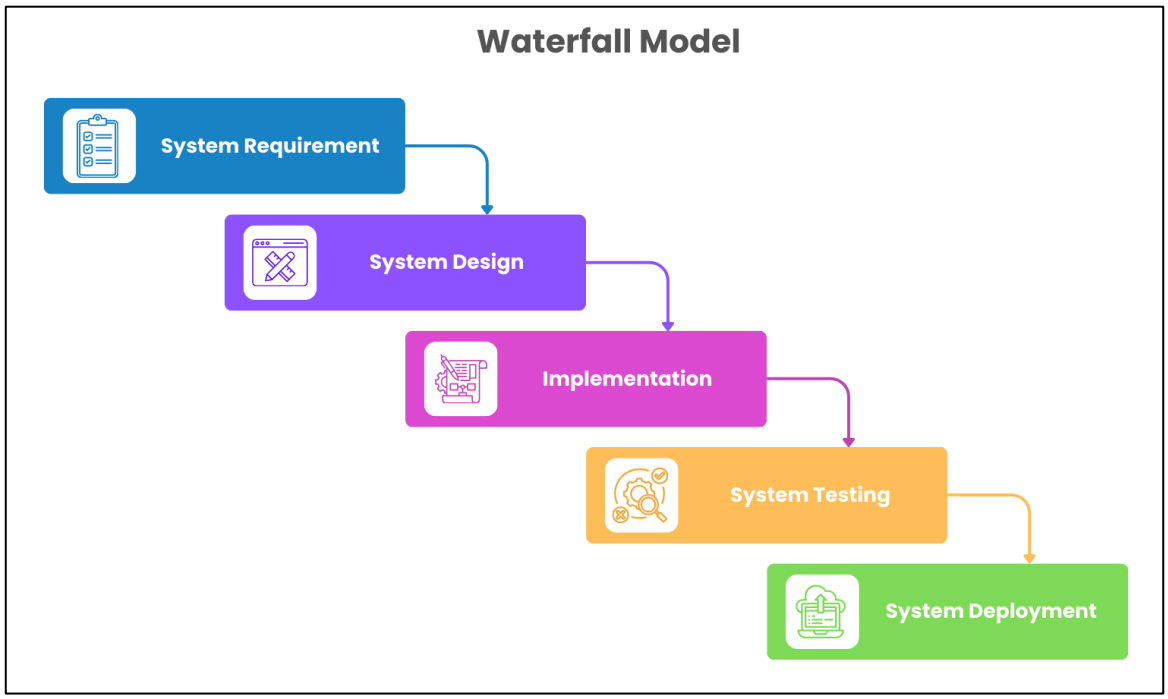


Figure 3.4 Waterfall Model

3.6.1 Phase 1: System Requirement (with Profiling)

The Requirements Analysis phase is critical for identifying the data necessary to deliver accurate and personalized job recommendations for CS and IT graduates. This phase focuses on gathering and structuring data from two key sources: user profiles and job postings, ensuring meaningful alignment between users and job opportunities.

User data is collected through a structured questionnaire, capturing attributes such as academic background (e.g., education level, CGPA, certifications), technical skills (e.g., programming languages, machine learning, project management), and personality traits. A personality assessment categorizes users as introverted, extroverted, or ambiverted, enhancing the recommendation process by aligning workplace preferences with job roles (Siswipraptini et al., 2024). Incorporating both technical and personality traits into profiling improves the precision of career recommendations, as emphasized by Rahhal et al. (2022).

Job data is dynamically updated via APIs from platforms like LinkedIn and JobStreet, ensuring access to real-time job descriptions, required skills, qualifications, and emerging roles. Labor market reports provide additional insights into trends and in-demand skills, ensuring the system reflects the latest industry demands (Adeagbo et al., 2019). This integration of real-time job data aligns with the needs of fresh graduates entering the Malaysian tech sector.

Profiling structures user and job attributes into actionable features, forming the foundation for the content-based filtering algorithm. By organizing data such as CGPA, technical skills, and personality traits for users, alongside qualifications and job roles for postings, the system calculates similarity scores effectively. This ensures tailored, relevant recommendations, addressing the unique challenges faced by fresh graduates (Kumar et al., 2022).

3.6.1.1 Tools and Technologies

The Career Navigator system employs a range of tools and technologies to ensure seamless functionality, scalability, and user engagement. Python is used for developing the recommendation algorithm due to its versatility and support for libraries like Scikit-learn, which facilitates efficient implementation of the content-based filtering mechanism (Bansal & Jain,

2020). MySQL is employed for database management, enabling reliable and scalable storage of user profiles and job postings with robust querying capabilities.

For back-end development, Flask and Django frameworks are utilized to bridge the front-end interface with the recommendation engine. Flask offers flexibility for lightweight applications, while Django ensures scalability and security, supporting future enhancements (Mulay et al., 2022). The front-end is developed using HTML, CSS, and JavaScript, providing a responsive and interactive user experience. APIs from platforms like LinkedIn and JobStreet enable real-time data collection, ensuring job recommendations reflect current market trends (Siswipraptini et al., 2022). A detailed summary of these tools and their purposes is provided in Table 3.3.

Tool/Technology	**Purpose**	**Justification**
Python	Algorithm Development	Versatile and easy to use, with extensive support for machine learning libraries like Scikit-learn, enabling efficient algorithm implementation.
MySQL	Database Management	Reliable and scalable for structured data. Used to store user profiles and job postings efficiently, with robust querying capabilities.

Flask/Django	Back-end Development	Supports scalability and rapid development. Flask offers flexibility, while Django provides built-in tools for secure and robust server-side logic.
APIs	Real-time Data Collection	Dynamically fetches job postings from platforms like LinkedIn and JobStreet, ensuring recommendations align with current market trends.
Scikit-learn	Content-Based Filtering Algorithm	Simplifies similarity score calculations and enables efficient recommendation generation through advanced data analysis tools.

Table 3.1 Summarization of their Tools, Purposes, and Justifications

3.6.2 Phase 2: System Design

The system design for the proposed Career Navigator system utilizes Diagrams.net, a web-based tool for creating graphical representations of system models. UML diagrams, including use case, class, activity, and sequence diagrams, are developed to visualize the system's architecture, interactions, and workflows across the design, implementation, and planning stages. These diagrams consist of graphical elements such as nodes and edges, which represent system components and their relationships. By providing a standardized way to represent system architecture, UML diagrams enhance communication, facilitate planning, and ensure

clarity during the design phase (Ozkaya et al., 2020). Supplementary documentation, such as use case specifications, is also included to offer a detailed understanding of the system's functionality and structure.

3.6.2.1 Use Case Diagram

The Use Case Diagram, as illustrated in Figure 3.5, depicts the interactions between both users and administrators with the web-based Career Navigator system, focusing on the system's functional requirements. It highlights the key actions each actor can perform to achieve their respective goals.

The primary use case is the Register/Login process, where users create an account or log in to access the system. Once authenticated, users can engage in Profile Management, which involves entering details about their academic background, technical skills, and personality traits. These inputs are essential for creating a comprehensive user profile that forms the basis for personalized job recommendations. Another critical use case is Search Job Recommendations, where users initiate a search for job opportunities. The system processes their profile data and dynamically retrieves job postings, providing tailored recommendations ranked by relevance. To further refine these results, users can engage in the Customize Preferences use case by adjusting thresholds and applying filters such as location, industry, or required skills. This customization ensures that the recommendations align with the user's specific career goals. Finally, the View Results use case allows users to review ranked job matches, which include detailed information such as job titles, company names, match percentages, and required skills. This functionality provides users with the insights needed to evaluate and act on job opportunities effectively.

For administrators, interaction begins with the Admin Login use case, which ensures that only authorized personnel can access administrative functionalities. Upon successful authentication, admins can perform actions such as Manage User Profiles, where they can view, update, or deactivate user information to maintain profile accuracy. They also have access to Manage Job Postings, enabling them to insert, update, or delete job-related data in the system. These administrative capabilities are critical for ensuring the overall quality, security, and trustworthiness of the platform.

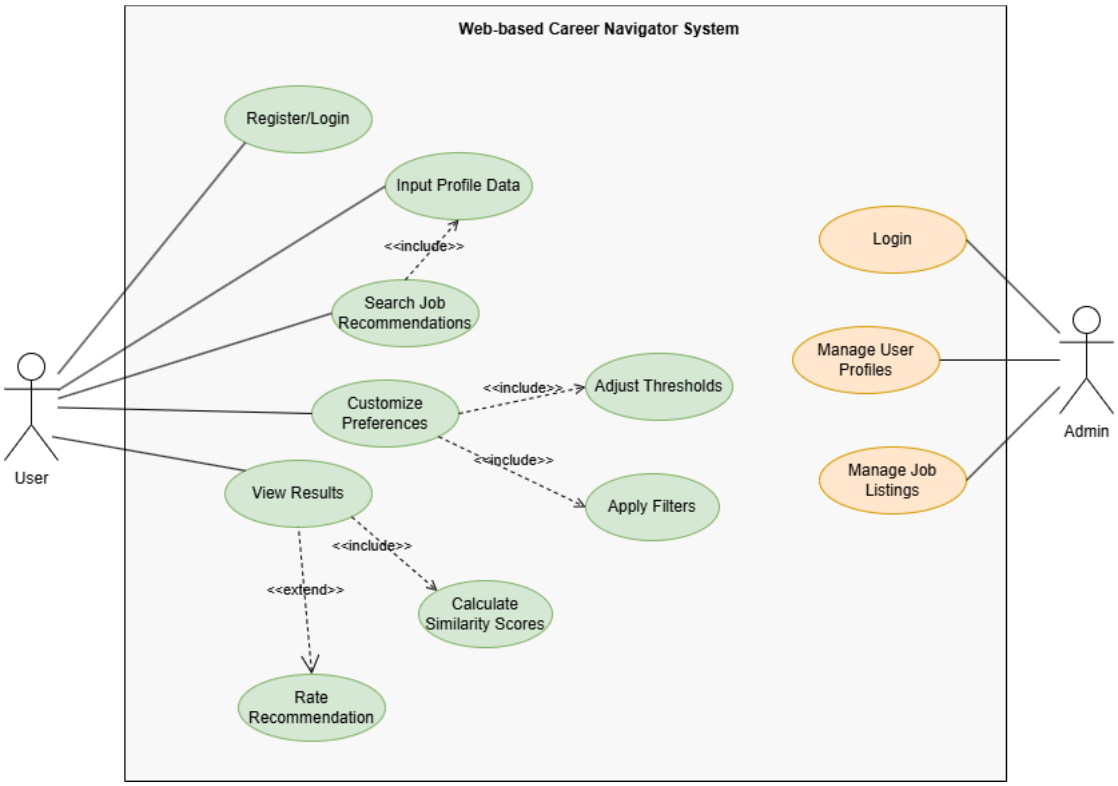


Figure 3.5 Use Case Diagram of Web-Based Career Navigator System

3.6.2.2 Class Diagram

The Class Diagram provides a structural view of the web-based Career Navigator system by representing its core entities and relationships. It highlights the data architecture and interactions between users, job postings, and the recommendations generated by the system.

The system consists of three main classes. The User class represents individuals using the system, with attributes such as userID, name, email, academicBackground, technicalSkills, and personalityTraits. These attributes are essential for creating a comprehensive user profile. The Job class represents job postings, with attributes such as jobID, title, requiredSkills, qualifications, and company. These attributes capture job details necessary for matching jobs to users. The Recommendation class represents the matches generated by the system, including attributes such as recommendationID, userID, jobID, and similarityScore, which quantify the alignment between user profiles and job postings. Additionally, the Admin class has been introduced to represent system administrators. This class includes attributes such as adminID, email, and password, and it supports methods for managing users and job postings, such as manageUserProfile() and manageJobData().

The relationships between these classes include a One-to-Many relationship between User and Recommendation, as each user can receive multiple job recommendations, and a One-to-One relationship between Job and Recommendation, as each recommendation corresponds to a single job posting. As shown in Figure 3.6, this structure ensures efficient data management and facilitates seamless interaction between system components.

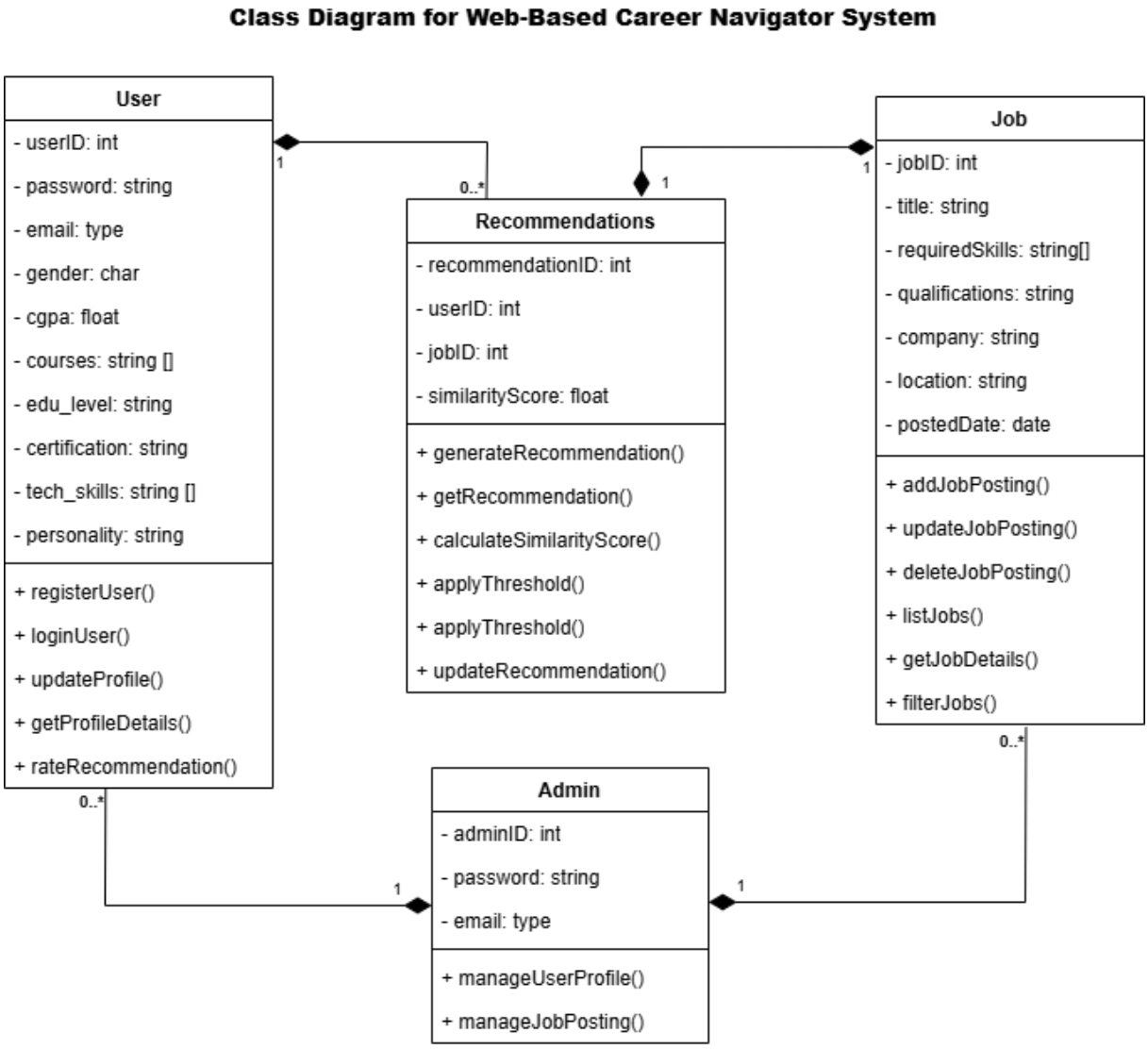


Figure 3.6 Class Diagram of Web-Based Career Navigator System

3.6.2.3 Sequence Diagram

The Sequence Diagram visualizes the dynamic interactions between the user, system components, and recommendation algorithm, highlighting how data flows through the Career Navigator system during key processes. The diagram emphasizes the real-time nature of user interactions and the system's data processing capabilities.

The process begins with User Registration and Login, where the user submits credentials, such as an email address and password. The system verifies these credentials

against stored records in the database to ensure secure and authenticated access. Once verified, the user gains access to the platform, enabling personalized job recommendations. Following authentication, the user proceeds to the Profile Submission stage, where they input critical information such as academic background, technical skills, and personality traits. The system validates and preprocesses this data, ensuring consistency through cleaning, handling missing values, and standardizing formats. This step prepares the data for analysis by the recommendation algorithm. In the Job Search process, the user initiates a query, prompting the system to dynamically retrieve job postings from external APIs and the local database. The retrieved data is processed by the content-based filtering algorithm, which calculates similarity scores between the user's profile and the job postings. These scores are ranked by relevance, and the top recommendations are displayed in a user-friendly interface. Users can then refine the results during the Adjust Threshold process by modifying similarity thresholds to either broaden or narrow the scope of recommendations. The system dynamically recalculates similarity scores and updates the ranked results accordingly.

As shown in Figure 3.7, the Sequence Diagram provides a detailed, step-by-step representation of the flow of data between the user, the system interface, the database, and the recommendation algorithm. This structured visualization ensures transparency in the system's functionality, demonstrating how the Career Navigator dynamically adapts to user input and preferences.

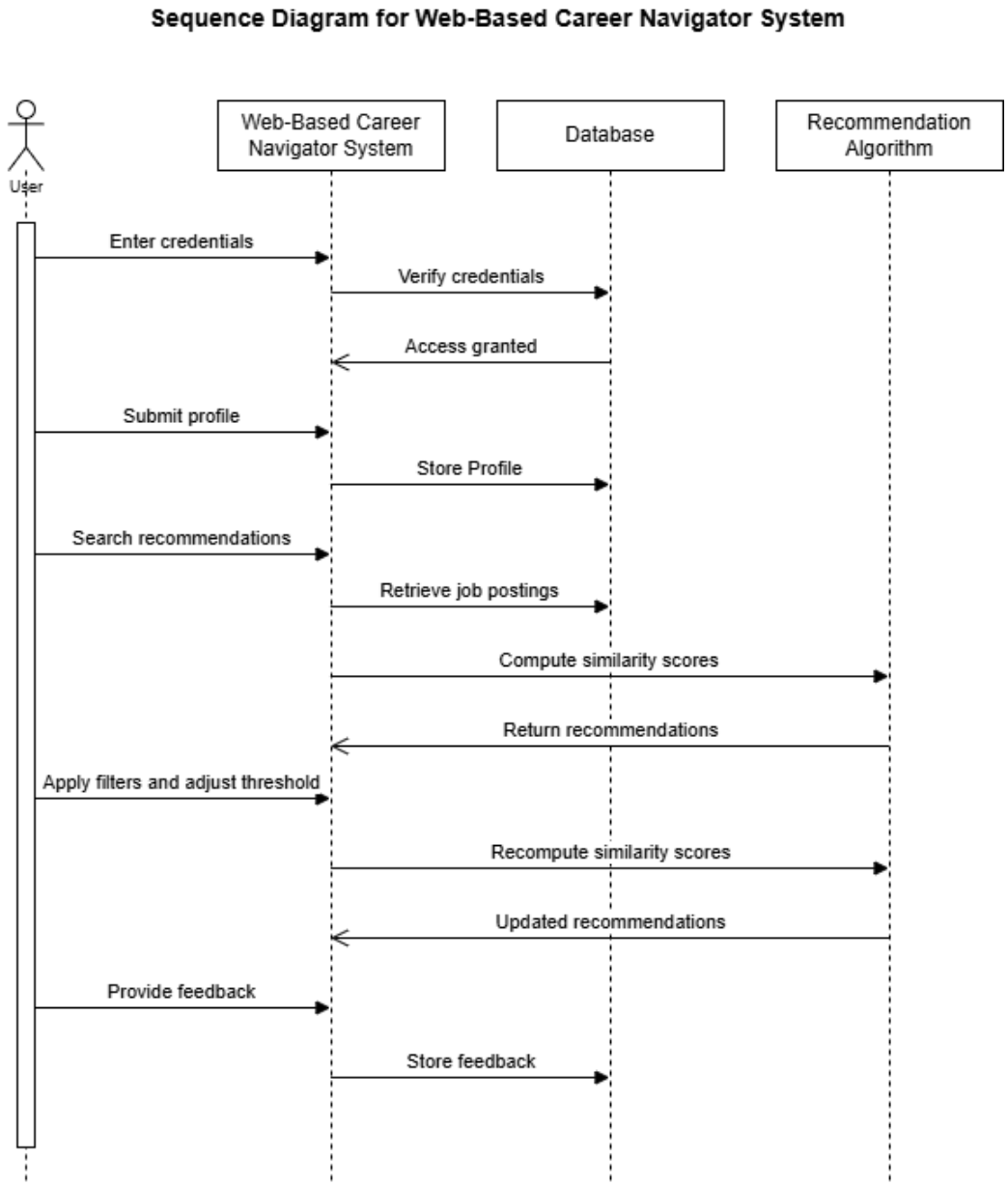


Figure 3.7 Sequence Diagram of Web-Based Career Navigator System

3.6.2.4 User Interface

The User Interface (UI) of the Career Navigator system is designed to be user-friendly, visually appealing, and responsive, ensuring seamless interaction for CS and IT graduates seeking personalized job recommendations. Figma was utilized to create wireframes and mock-ups, offering a clear visual representation of the system's structure and functionality. These pre-

designs, illustrated in Figures 3.8a to 3.8g, provide insights into the system's layout and features, aligning expectations between developers and users before implementation.

The Home Page (Figure 3.8a) serves as the first point of interaction for users, introducing the system's features and guiding them toward registration or login. The Register Page (Figure 3.8b) allows new users to create an account by providing essential information, ensuring secure and seamless access to the system. The Login Page (Figure 3.8c) facilitates secure authentication for returning users, allowing them to enter their email and password to access their personalized dashboard.

Upon successful login, users are directed to the Landing Page (Figure 3.8d), where they need to set up their profile. Figure 3.8e shows the page where users can enter their academic background, technical skills, and personality traits. Once the profile setup is complete, users are directed to the Search Page (Figure 3.8f), which allows them to specify criteria for their job searches. Results from the search are displayed on the Results Page (Figure 3.8g), which organizes job matches into a ranked format. Each result includes details such as job titles, companies, salary, and location, enabling users to evaluate opportunities efficiently.

These mock-ups (Figures 3.8a to 3.8g) demonstrate how the Career Navigator system supports intuitive navigation, efficient searching, and personalized recommendations, empowering users to focus on opportunities aligned with their qualifications and career goals.

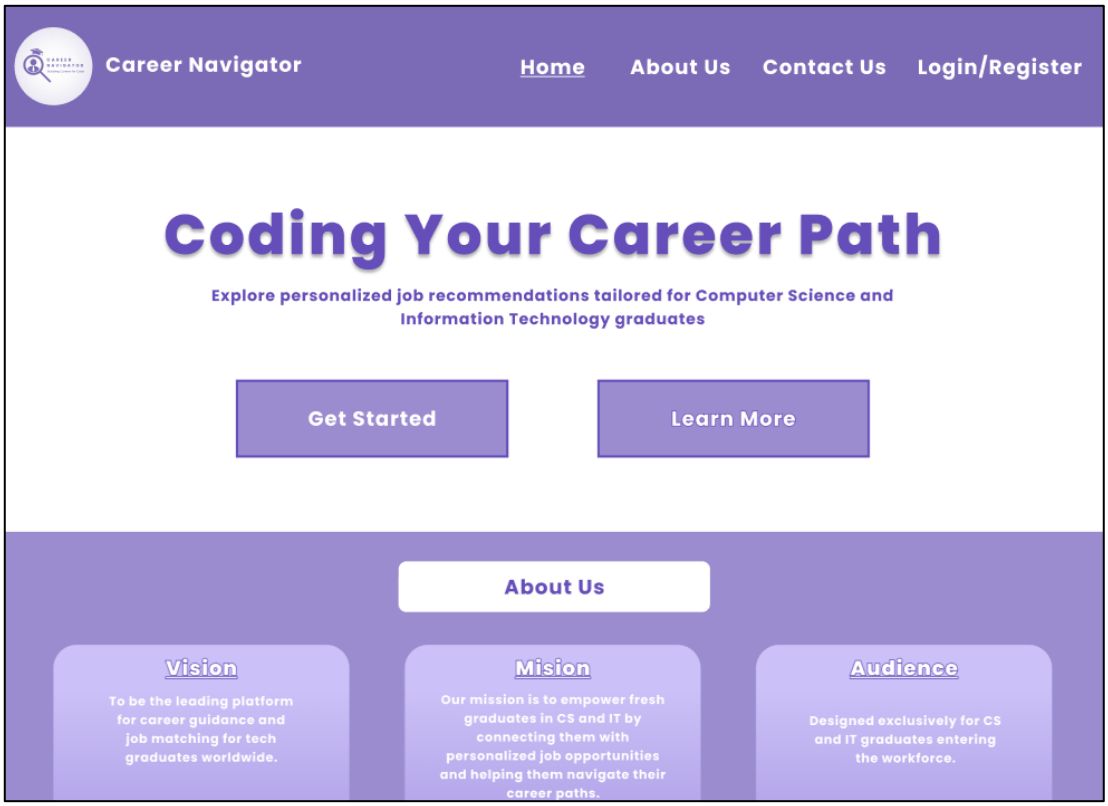


Figure 3.8a Home Page providing an introduction to the system.

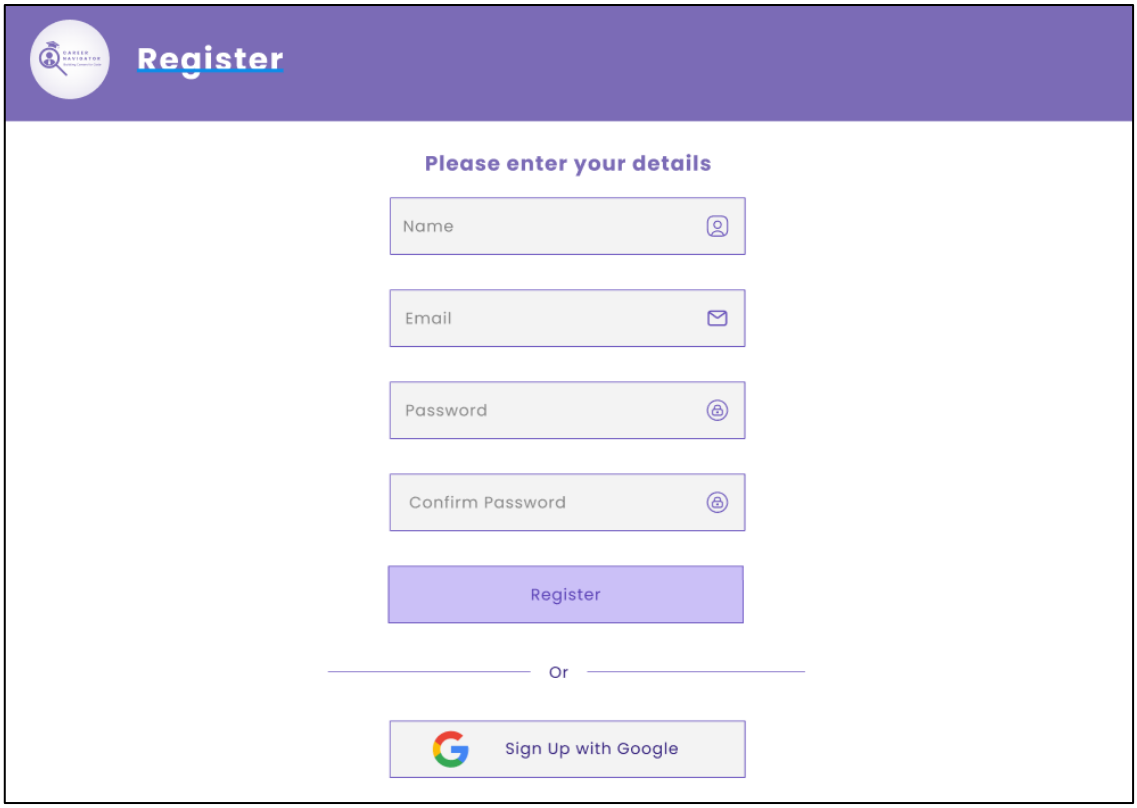


Figure 3.8b Register Page for creating a new user account.

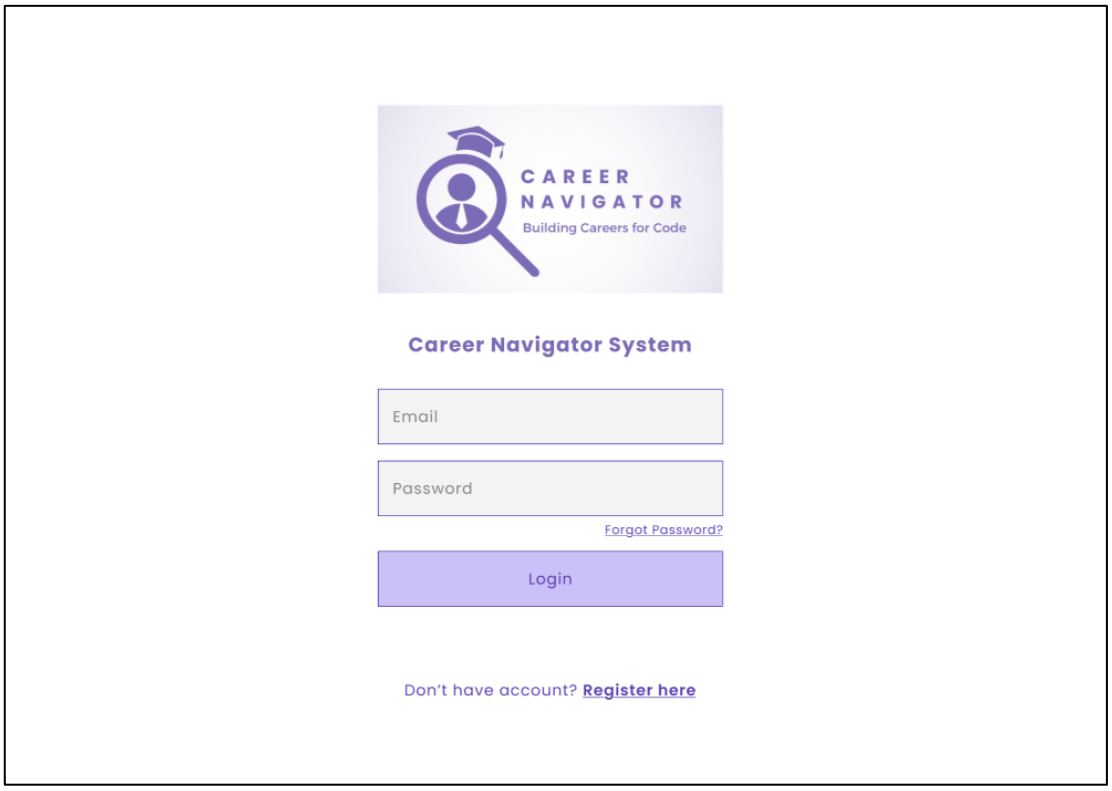


Figure 3.8c: Login Page for secure user authentication.

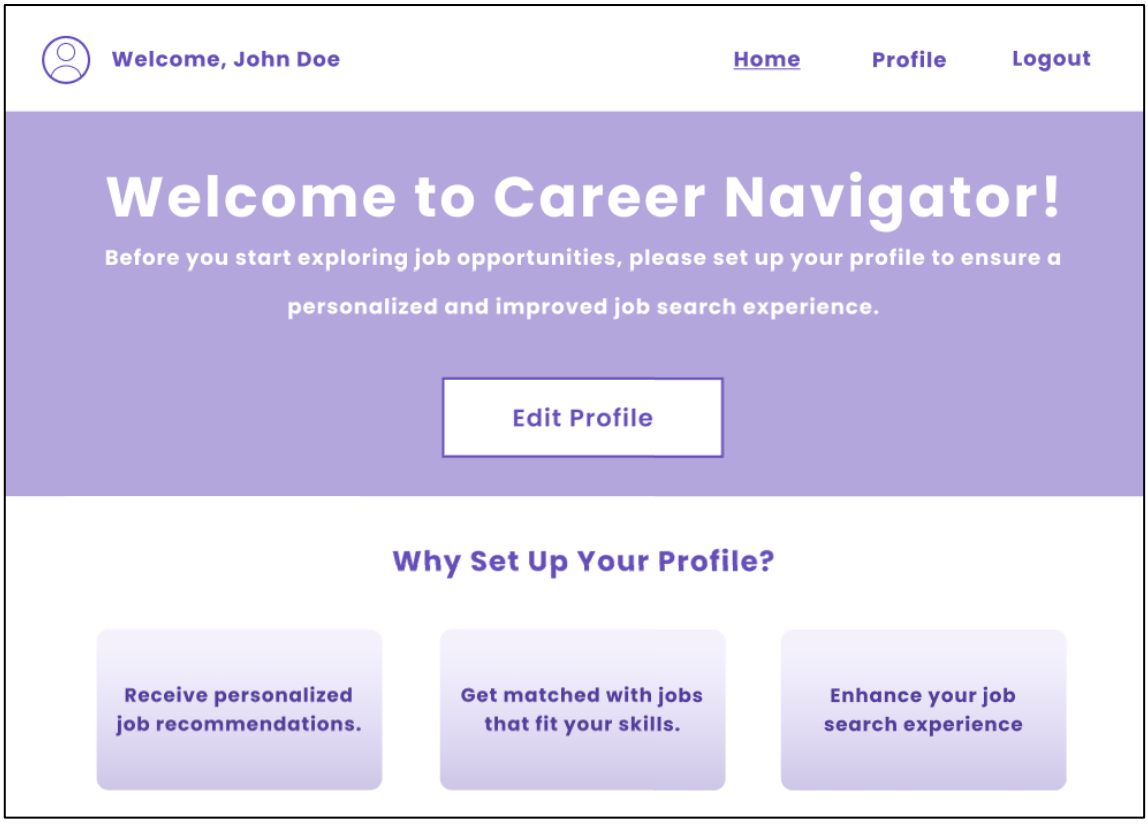


Figure 3.8d Homepage after user successfully signed in.

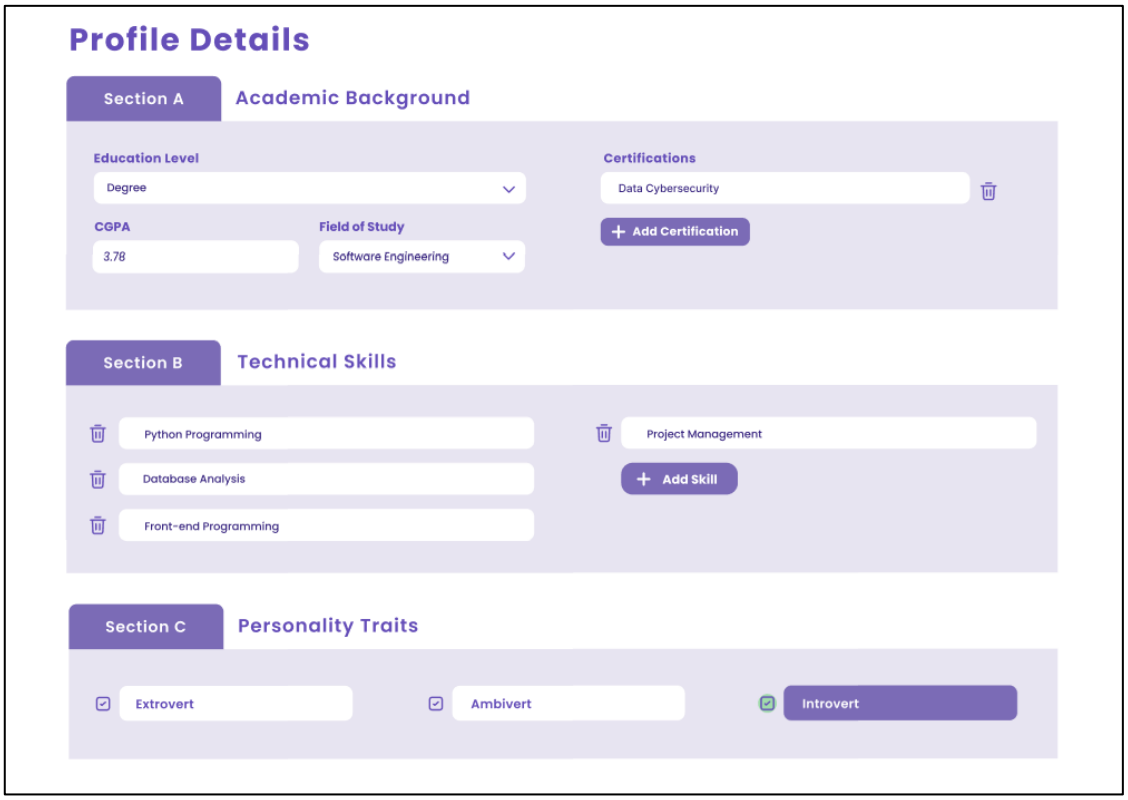


Figure 3.8e Page on where user input their details

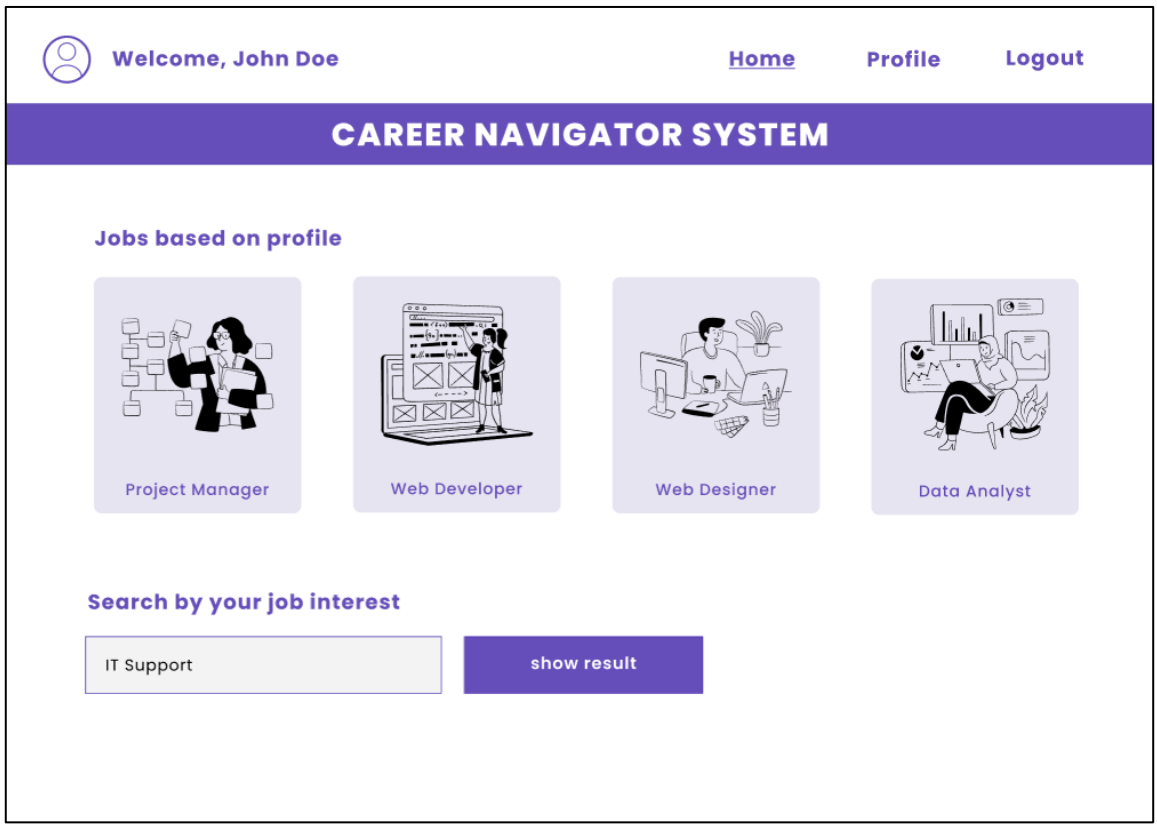


Figure 3.8f Search Page enabling users to refine job search criteria.

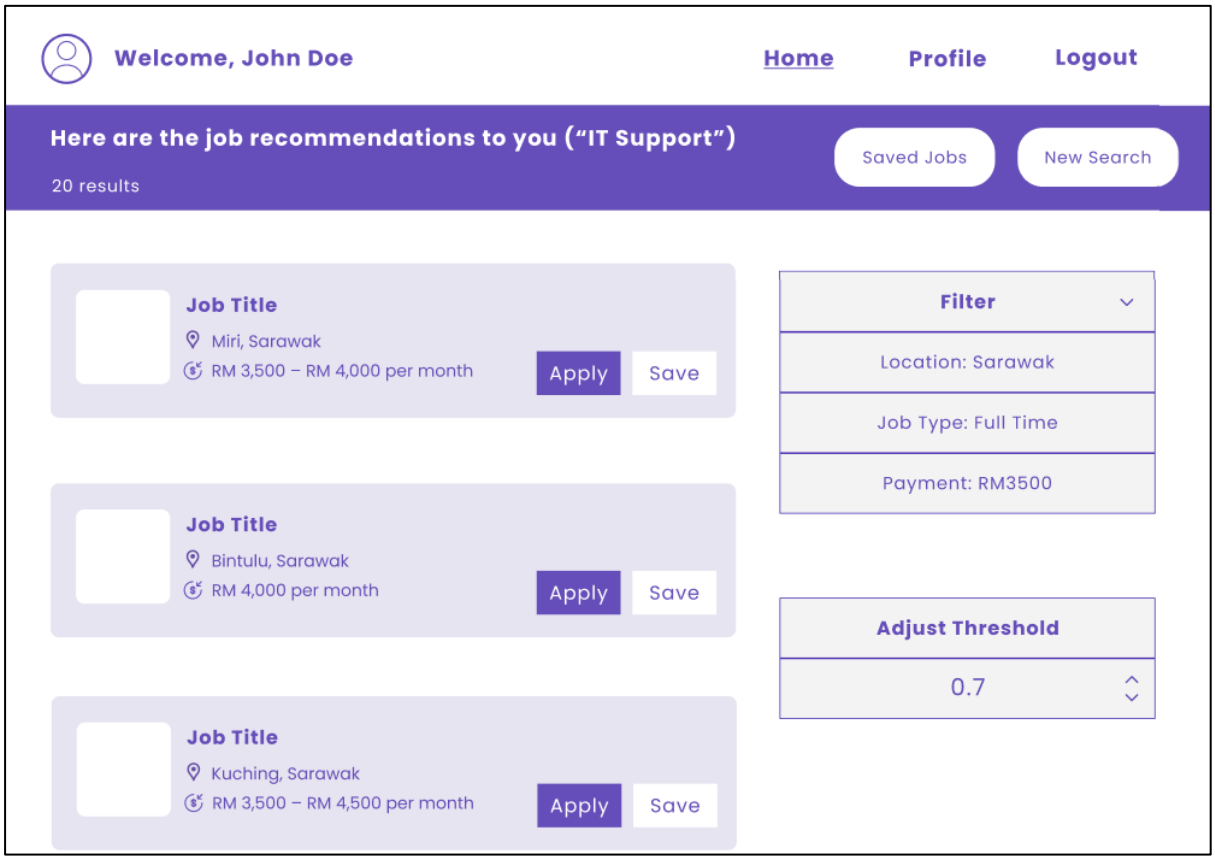


Figure 3.8g Results Page displaying ranked job recommendations.

3.6.3 Phase 3: Implementation

Web-based Career Navigator's implementation phase develops the front-end interface, back-end functionality, and database integration. In this phase, the system design is turned into a fully functional web application that gives CS and IT graduates personalized job recommendations. The front-end will use HTML, CSS, and JavaScript to create a responsive, user-friendly interface that works across devices. Flask or Django will handle server-side logic, secure user data, and recommendation engine integration on the back end. MySQL will store structured data like user profiles and job postings for real-time data queries and recommendation generation. In this phase, user registration, profile input, and basic recommendation features will be added incrementally before the actual implementation in the next semester. Later versions will add threshold mechanism and feedback integration. To

ensure functionality, accuracy, and project goals, the system will be tested after implementation. With this planned approach, the implementation phase matches the system design, creating a robust and scalable solution.

3.6.4 Phase 4: System Testing

The system testing phase of the Career Navigator ensures that all components function as intended and align with the project's objectives. Testing focuses on validating the accuracy, reliability, and usability of the system by systematically evaluating key features, including user data input, recommendation generation, and real-time job updates. Functional tests will confirm that each module operates correctly, such as verifying that user profiles are saved accurately, job recommendations are relevant, and the threshold mechanism filters results effectively. Additionally, usability testing will assess the system's interface for intuitiveness and accessibility, ensuring a seamless user experience. Testing scenarios will simulate user interactions, such as creating profiles, performing job searches, and applying filters, to identify and resolve any errors or inconsistencies. According to Mulay et al. (2022), thorough system testing is essential for ensuring the reliability and effectiveness of web-based recommendation systems. This phase prepares the system for deployment, guaranteeing it meets user expectations and delivers accurate, personalized job recommendations.

3.6.5 Phase 5: System Deployment

The system deployment phase involves making the Career Navigator accessible to users as a fully functional web application. This step will include hosting the system on a cloud-based platform, such as AWS or Heroku, to ensure scalability, security, and reliable performance. Deployment tasks will involve setting up the database, integrating the back-end and front-end components, and configuring APIs for real-time job data collection. While the actual

deployment has not yet been implemented, this phase is planned for the next semester following system development and testing. Once deployed, the system will be monitored for performance and user feedback to ensure smooth operations and adaptability to evolving user needs and industry trends.

3.6.6 Algorithm Development

The Career Navigator system's core functionality is powered by a content-based filtering algorithm, designed to match user profiles with suitable job postings based on attributes such as technical skills, academic qualifications, and personality traits. The algorithm calculates a similarity score by comparing user-provided attributes with job requirements. This score quantifies the alignment between the two, forming the basis for ranking job recommendations. For instance, if a job requires Python, Java, and SQL, and the user's profile includes Python and Java, the system assigns a similarity score reflecting the degree of overlap.

To enhance flexibility, the system incorporates a threshold mechanism that filters job recommendations based on user-defined relevance criteria. Users seeking broader options can set a lower threshold, while those targeting specific roles can raise the threshold to refine their matches. As shown in Figure 3.9, the algorithm iterates through the job database, calculates the similarity score for each posting, and adds jobs to the recommendation list only if their score meets or exceeds the user-defined threshold. This mechanism reduces cognitive overload and improves decision-making by tailoring results to user preferences (Rashid et al., 2022).

The algorithm workflow dynamically processes data by first initializing an empty recommendation list. It then evaluates each job posting in the database by calculating its similarity score against the user profile. Only postings with scores above the defined threshold are added to the list. Once all job postings are evaluated, the system returns a ranked

recommendation list based on relevance. This structured approach ensures that users receive personalized and accurate job suggestions.

Additionally, the system integrates a feedback mechanism, allowing users to rate and review job recommendations. This feedback helps refine the algorithm by adjusting attribute weights and matching logic over time. For instance, consistent negative feedback on roles requiring advanced skills may reduce the weight of those skills in future recommendations, while positive ratings for specific job types could increase their priority in subsequent matches (Bansal & Jain, 2020).

By combining similarity scores, customizable thresholds, and dynamic feedback, the Career Navigator system ensures personalized and adaptable job recommendations. This approach not only streamlines the job search process but also aligns with user preferences and industry trends, offering an efficient, user-centric career matching solution.

```
Initialize recommendation_list as an empty list


FOR each job_posting in job_database:
    similarity_score = calculate_similarity(user_profile,
job_posting)
    IF similarity_score >= user_defined_threshold:
        Add job_posting to recommendation_list
    ENDIF
ENDFOR
RETURN recommendation_list
```

Figure 3.9 Threshold Mechanism

3.7 Step 5: Evaluation of Proposed System

The evaluation of the Career Navigator system centers on gathering user feedback to assess its performance and identify areas for improvement. This process involves collecting input on the relevance, accuracy, and usefulness of job recommendations through surveys, ratings, and reviews. User feedback plays a pivotal role in refining the system's algorithm, ensuring that recommendations align more closely with user preferences. For instance, consistent negative ratings for specific recommendations prompt adjustments to the matching logic or attribute weights, improving future matches. Additionally, the threshold mechanism will be fine-tuned based on user input, allowing the system to adapt to diverse preferences by filtering job postings more effectively (Mulay et al., 2022; Rashid et al., 2022).

The system will be evaluated across three key aspects: recommendation quality, user satisfaction, and system usability. Recommendation quality focuses on how well the system matches user profiles to relevant job postings, ensuring alignment with user attributes such as technical skills and career goals (Bansal & Jain, 2020). User satisfaction assesses the perceived value of recommendations and the overall trust users place in the platform (Sindhu et al., 2023). Finally, system usability evaluates the interface design, navigation ease, and accessibility across devices, ensuring that the platform supports intuitive and efficient interactions. This iterative feedback process ensures that the system evolves dynamically to remain user-centric and responsive to changing job market trends, ultimately delivering a personalized and engaging experience for job seekers (Rahman et al., 2023).

3.8 Step 6: Documentation

The final step in developing the Career Navigator system is the preparation of comprehensive documentation, including a detailed thesis report, a research paper, and presentation slides. The thesis report will provide an in-depth account of the project, covering the background, objectives, methodology, system design, implementation, evaluation, and future work. The research paper will distill the project's key contributions and findings into a concise format suitable for academic publication, emphasizing how the system addresses gaps in job-matching platforms for CS and IT graduates. Lastly, the presentation slides will visually summarize the system's features, methodology, and results for stakeholders or academic panels, facilitating clear and impactful communication of the project's significance.

3.9 Summary

This chapter detailed the methodology for developing the web-based Career Navigator system, employing a structured approach across key phases: data collection, preprocessing, feature extraction, algorithm development, recommendation generation, and system evaluation. The system's content-based filtering and adjustable threshold mechanism deliver personalized and accurate job recommendations tailored to users' academic backgrounds, technical skills, and personality traits. User feedback plays a central role in refining the system dynamically, ensuring recommendations remain relevant to evolving user needs and industry demands. By integrating advanced filtering techniques, customizable options, and a robust feedback loop, the Career Navigator system provides a scalable, localized, and user-centric platform. This innovative approach enhances employability for CS and IT graduates, empowering them with targeted career guidance in a competitive job market.

CHAPTER 4 IMPLEMENTATION

4.1 Introduction

This chapter describes how the Career Navigator web-based application was put into use to provide graduates in computer science and information technology with individualized job recommendations. The system architecture, user profiling, data preprocessing, recommendation algorithm, and development environment are all covered. It explains how to integrate MySQL for data storage with Flask for backend development. In order to prepare for further testing, this chapter also describes how the design ideas from Chapter 3 were turned into a working system.

4.2 Hardware and Software Environment

The hardware and software environments used during the implementation phase are detailed in Table 4.1 below. These configurations provided sufficient resources and tools to develop, test, and deploy the Career Navigator system effectively.

No	Component	Specification
1	CPU	AMD Ryzen 5 5500U with Radeon Graphics 2.10 GHz
2	RAM	8.00 GB (7.38 GB usable)
3	Operating System	Windows 11 Home Single Language (64-bit)

4	Integrated Development Environment (IDE)	Visual Studio Code
5	Front-end	HTML, CSS, JavaScript
6	Backend Framework	Flask
7	Database	MySQL
8	Programming Language	Python 3.13.1
9	Web Browsers	Google Chrome

Table 4.1 Hardware and Software Environment

The system was developed on a personal laptop equipped with an AMD Ryzen 5 5500U CPU clocked at 2.10 GHz and 8 GB of RAM, running Windows 11 Home Single Language (64-bit). Table 4.1 summarizes the key hardware and software components utilized throughout the implementation, including the development environment, backend framework, database management system, front-end technologies, testing browsers, and deployment platform. The hardware and software requirements were moderate, and the laptop provided sufficient computational power and memory to efficiently support the development, testing, and deployment phases of the Career Navigator system.

4.3 Environment Setup

To successfully create and deploy the Career Navigator web-based application, a correctly configured environment is necessary, one that offers the fundamental tools and frameworks. The configuration procedure for the Flask framework, which functions as the application's backend, is

covered in length in this chapter. The integrated development environment (IDE) used throughout the project, Visual Studio Code (VS Code), is also covered. The installation and setup procedures for these crucial elements are covered in the sections below.

4.3.1 Flask

Flask is a lightweight, open-source Python web framework known for its simplicity, flexibility, and ease of integration with various Python libraries and extensions, making it a popular choice for developing web applications and APIs. The Flask framework was chosen for its lightweight nature and ease of integration with Python (Relan, 2019). Flask supports essential web development features like routing, session management, and template rendering, and can be extended with packages for database connectivity, authentication, and more.

Setting up a Flask environment typically involves installing Python, pip, Flask itself, and any necessary extensions (such as mysql-connector-python for database access), following official documentation to ensure compatibility and stability. The environment setup began with the installation of Python and the Python package manager, pip, which were prerequisites for Flask. Using the command-line interface on Windows 11, Flask was installed via pip along with necessary extensions such as *mysql-connector-python* to enable database connectivity (Chauhan et al., 2019).

Before installing Flask, the Python and pip versions were checked for compatibility and proper environment setup. Figure 4.3 shows how the pip command was used to install Flask and its required extension, *mysql-connector-python*. After installing the required packages, the Flask development server was launched locally on port 5000, confirming that the backend was properly configured and running as shown in Figure 4.4. This setup was critical because Flask was the main backend framework, providing features such as routing, session management, and

template rendering. Its modular and extensible design allowed for the seamless integration of components such as the recommendation engine and user profiling module, which together comprise the Career Navigator system's core logic.

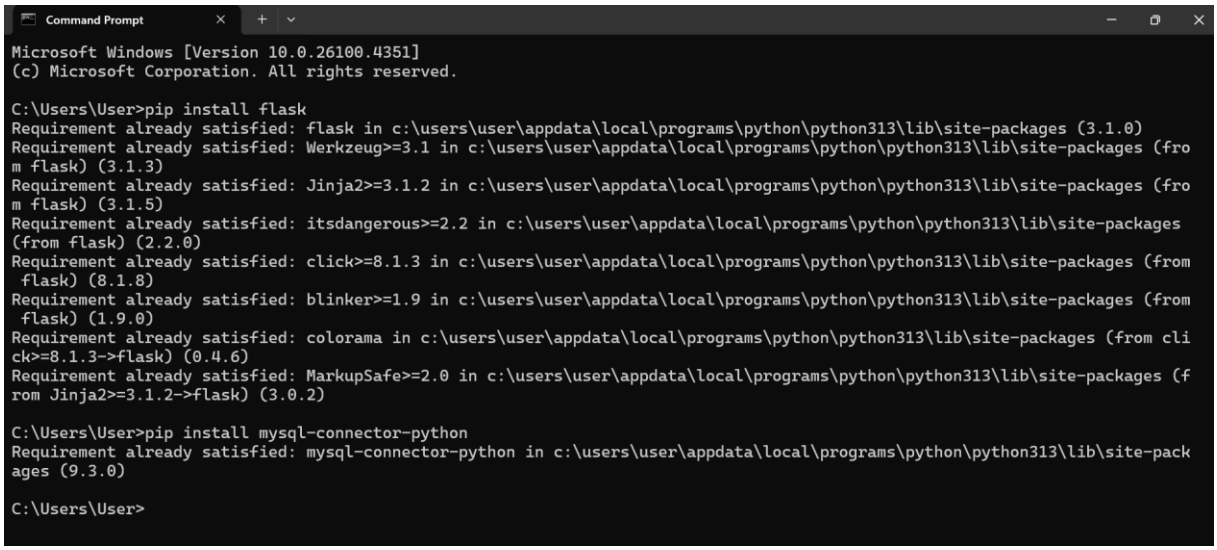


Figure 4.1 Installation of Flask and mysql-connector-python using pip

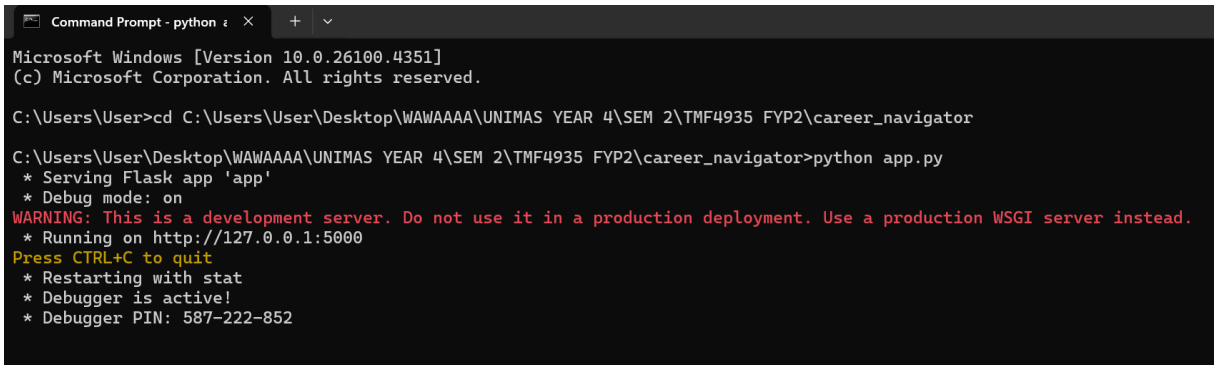


Figure 4.2 Running Flask Server

4.3.2 Visual Studio Code

Microsoft Visual Studio Code (VS Code) (Microsoft, 2022) was selected as the primary integrated development environment (IDE) due to its user-friendly interface, extensive Python

support, and versatile debugging capabilities. The latest stable version of VS Code was downloaded and installed on a Windows 11 laptop powered by an AMD Ryzen 5 5500U processor with 8GB of RAM.

To enhance development efficiency, Python and Flask extensions were added to VS Code, providing syntax highlighting, code completion, and debugging features tailored for Flask applications. Additional workspace settings were configured to enforce code formatting standards and enable integrated terminal support, allowing smooth execution of Flask commands directly within the IDE.

Figure 4.4 illustrates the folder structure of the Flask project as viewed in the Visual Studio Code interface. This organized structure includes templates for the user interface, static assets like CSS and JavaScript, and core Python modules for routing and logic. A well-organized project directory improves code maintainability and enables developers to navigate and scale the system with ease.

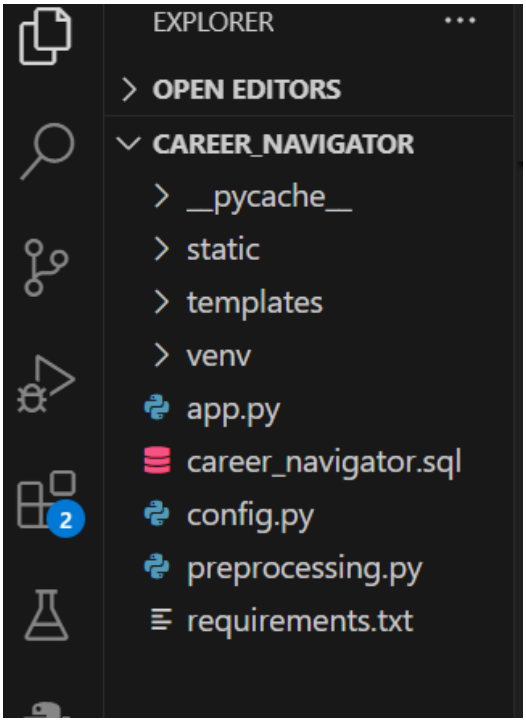


Figure 4.3 Project folder structure in Visual Studio Code

4.4 System Modules

This section presents the implementation of the key system modules that align with the methodology discussed in Chapter 3. These modules work together to deliver personalized job recommendations tailored to users' academic backgrounds, technical skills, and personality traits. Each module is designed to achieve specific project goals, from data input and processing to similarity-based job suggestion.

4.4.1 Graphical User Interface (GUI) System

The Career Navigator system's graphical user interface (GUI) is intended to provide a seamless and user-friendly experience, utilizing standard web technologies such as HTML, CSS, and JavaScript to ensure broad accessibility and responsiveness across devices. It allows users to securely register and log in, enter personal and academic background information, and interact with job recommendations generated by the system. The interface displays ranked job listings based on similarity scores and that allows users to adjust the recommendation threshold to receive more personalized suggestions. Their design is based on accessibility and usability principles such as simplicity, consistency, and adaptability to different screen sizes, ensuring that users from various backgrounds can effectively interact with the system (Daniel, 2025). This front-end implementation adheres to usability principles like simplicity, consistency, and real-time interaction, which improves user satisfaction and system effectiveness. Figure 4.8 depicts the main page of the Career Navigator web-based system.

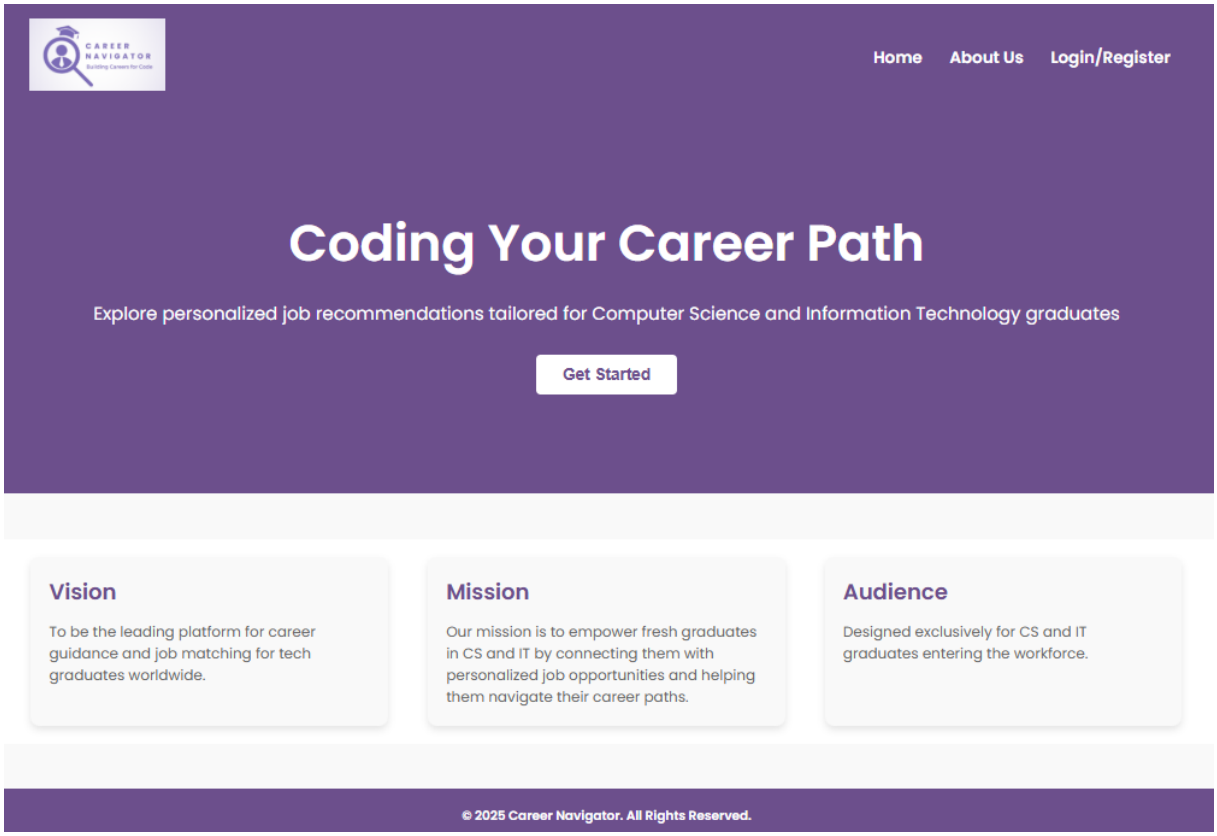


Figure 4.4 The main page of the Career Navigator Web-based System

4.4.2 User class (Profiling Module)

The Career Navigator Web-based System's User module is intended to handle comprehensive profile information such academic history, technical abilities, and personality characteristics, all of which are essential for producing precise and tailored job suggestions. To guarantee input accuracy and completeness, this module uses structured data collection via validated web forms, while user data is continuously stored in a MySQL database to maintain reliability, consistency, and scalability (D et al., 2024). This method is consistent with recognized best practices in user profile management, where detailed and precise profiles facilitate both skill-based matching and deeper behavioural compatibility via personality evaluations. Based on concepts like personality-job fit, this design enables the system to suggest

jobs that are in line with the user's skills and workplace choices. Modern recommender systems rely heavily on intelligent user profiling since it enables a deeper understanding of user goals, behavior, and traits, which in turn leads to more relevant and enjoyable user experiences. In addition, the system incorporates fundamental authentication methods to safeguard sensitive profile information throughout the user lifecycle in to maintain trust and privacy (Cesconetto et al., 2020).

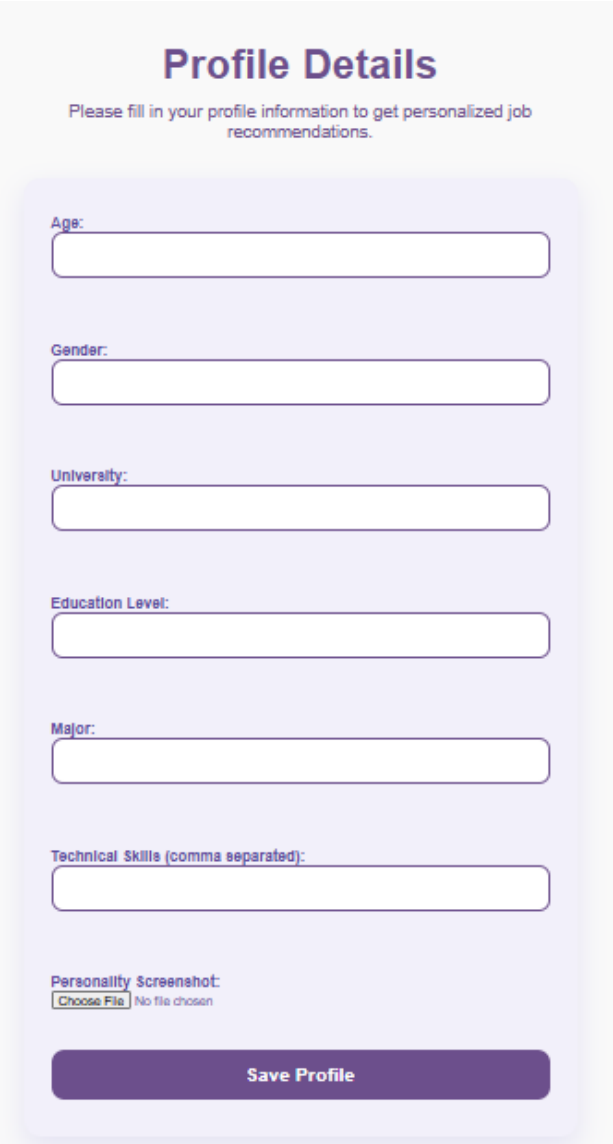


Figure 4.5 Screenshot of Profile Completion Form (Academic + Personality Section)

4.4.3 Data preprocessing and Feature Extraction

Data preprocessing is a foundational step to ensure that both user profiles and job postings are clean, standardized, and comparable, which is critical for generating accurate job recommendations. This phase involves multiple tasks such as removing duplicate or outdated job listings, filling in missing values through median imputation, and normalizing inconsistent skill terminology. These steps convert unstructured data into a consistent and structured feature space, aligning with best practices in recommendation system design (Rodriguez & Chavez, 2019).

To facilitate this, the system applies a preprocess_skills() function that transforms comma-separated skill strings from both users and job postings into lowercase sets. This ensures uniformity and enables set-based operations for skill matching. For analyzing personality-job fit, the extract_personality_matches() function performs keyword-based scanning of job descriptions, looking for indicative terms like “detail-oriented” (introvert), “team player” (ambivert), or “leadership” (extrovert). These matches are then interpreted by the personality_match() function, which determines if a particular job aligns with a user’s MBTI personality category, producing a binary match result.

The final step, feature extraction, is operationalized through the compute_similarity() function, which combines technical skill overlap and personality compatibility into a single score. It uses a Jaccard similarity measure for skills and a binary value for personality fit. The resulting similarity score is calculated as a weighted average of 70% based on technical skills and 30% on personality traits ensuring a balanced and meaningful recommendation output. This streamlined approach is illustrated in Figure 4.11, which shows the core Python functions

responsible for preprocessing and computing the similarity score between user profiles and job postings.

preprocessing.py > ...

```python
def preprocess_skills(skills_string):
    if not skills_string:
        return set()
    skills = [skill.strip().lower() for skill in skills_string.split(',')]
    return set(skills)

def compute_similarity(user_profile, job_posting):
    user_skills = preprocess_skills(user_profile.get('Tech_Skills', ''))
    job_skills = preprocess_skills(job_posting.get('required_skills', ''))

    intersection = user_skills.intersection(job_skills)
    union = user_skills.union(job_skills)
    skill_sim = len(intersection) / len(union) if union else 0

    personality_sim = 1 if personality_match(user_profile.get('MBTI', '').lower(), job_posting.get('description

    final_score = 0.7 * skill_sim + 0.3 * personality_sim
    return final_score
```

Figure 4.6 Screenshot of core preprocessing and similarity computation functions used to match user profiles with job postings based on skill overlap and personality-job fit.

4.4.4 Recommendation Engine Module (Matchmaking Module)

The recommendation engine is the core of the Career Navigator system, employing a content-based filtering algorithm enhanced with a customizable threshold mechanism. It calculates similarity scores by measuring feature overlaps between user profiles and job postings, factoring in both skill matches and personality alignment. Users are empowered to adjust the recommendation threshold, controlling the breadth and precision of suggested jobs. This flexible filtering allows users to tailor the job recommendations to their preferences, directly addressing the project's objective of personalized job matching (Rahman et al., 2014).

The threshold logic effectively balances recommendation quantity and quality, enabling an adaptive user experience.

The system integrates a content-based filtering algorithm that computes similarity using a weighted combination of technical skills (70%) and personality-job fit (30%). The implementation leverages functions such as compute_similarity() to evaluate profile-job alignment and generate ranked results. The system provides a dynamic adjustable threshold on the user interface, which enables users to fine-tune recommendation strictness. By increasing the threshold, users receive fewer but more relevant results; lowering it broadens the pool of suggested jobs. This implementation ensures that the recommendation process remains both adaptive and user-driven, fulfilling the goals of personalized, precise job matching.

4.5 Deployment

The Career Navigator system was initially deployed in a local development environment to support iterative development, debugging, and usability testing. This approach provided a stable and controlled setting for verifying core functionalities such as user registration, profile creation, and job recommendation accuracy before considering future deployment on cloud platforms (Daniel, 2025). The deployment process began on a Windows 11 machine by installing essential software components, including Python, Flask, MySQL, and supporting Python libraries such as mysql-connector-python and Werkzeug, using pip. A virtual environment was created using venv to isolate project dependencies and ensure consistent behavior across different systems (*Welcome to Flask — Flask Documentation (3.1.x)*, n.d.). Version control using Git was applied throughout development to maintain code integrity and track changes (Daniel, 2025). Database connectivity between the Flask backend and MySQL was configured using secure credentials stored in environment variables to protect sensitive

data. The Flask development server was launched locally using the command python app.py, allowing the system to run on localhost via port 5000. This made the application accessible for real-time testing through browsers such as Google Chrome and Mozilla Firefox. Extensive testing was performed across different browsers to ensure consistent interface behavior and cross-platform compatibility. Network and firewall settings were also checked to guarantee unrestricted local access during development. Overall, this local deployment phase was crucial in validating system functionality and refining the user experience before proceeding to future stages like public or cloud deployment.

4.6 Summary

In summary, this chapter described the implementation of the Career Navigator system, starting with the setup of the development environment using Flask, MySQL, and Visual Studio Code. Key modules were implemented based on the project's methodology, including the user-friendly graphical interface, the User profiling module, data preprocessing and feature extraction, and the personalized recommendation engine. Each component worked together to match users with suitable jobs based on skills and personality. The system was deployed in a local environment for testing, ensuring functionality and readiness for future enhancements or public deployment.

CHAPTER 5 EVALUATION, TESTING AND RESULT

5.1 Introduction

This chapter presents the evaluation phase of the Career Navigator system, focusing on usability testing conducted through the System Usability Scale (SUS) and supplemented by open-ended user feedback. The aim of this evaluation is to assess the system's usability, clarity, and overall effectiveness from the perspective of actual users. A total of 14 participants were recruited to complete a structured questionnaire distributed via Google Form, which consisted of demographic questions, a 10-item SUS section, and an open-ended feedback segment. The SUS uses a 5-point Likert scale, a well-established psychometric instrument for measuring user attitudes and opinions (Lewis, 2018).

5.2 User Acceptance Testing (UAT)

User Acceptance Testing (UAT) was conducted to determine whether the Career Navigator system met the functional and usability expectations from an end-user perspective. This phase focused on evaluating how users interact with the system and whether the core features performed as intended (Fitri et al., 2024). By simulating real tasks and collecting observations, UAT helped ensure the system is ready for actual usage. The testing involved both general users and admin roles. Participants were asked to complete a series of predefined actions such as registering an account, logging in, uploading personality test results (MBTI), viewing job matches, and, for the admin, managing job postings and approving uploads. These test cases were designed to reflect realistic user interactions with the system.

5.2.1 UAT Objectives

The key objectives of conducting UAT for this project were to:

1. Ensure all features are functional and behave as expected.
2. Validate that users can navigate the system easily without confusion.
3. Check the workflow for both user and admin roles, especially the MBTI upload and job recommendation process.
4. Identify any bugs or usability issues that need to be fixed before deployment.
5. Gather direct feedback from users regarding their experience.

5.2.2 UAT Test Cases

Each feature was tested individually through structured test cases. The test results are summarized in the Table 5 below. All test cases were completed successfully with no critical errors reported. Users were able to interact with the system smoothly, and all key functions responded as expected.

The successful outcome of UAT indicates that the system is stable and functional in real use scenarios. No significant issues were encountered during testing, and participants responded positively to the interface and flow of the application. Minor suggestions from users were noted for future enhancement, such as simplifying the MBTI upload process or adding more job options. Overall, UAT confirmed that the system is ready for deployment.

Module	Test Case ID	Description	Status
User Registration	TC001	Register with valid credentials	Pass
User Login	TC002	Login using valid user account	Pass
MBTI Upload	TC003	Upload MBTI test screenshot	Pass
Admin Approval	TC004	Admin views and approves uploads	Pass
Job Matchmaking	TC005	Display matching jobs based on personality	Pass
Admin Add Job	TC006	Admin adds new job postings	Pass
Admin Delete Job	TC007	Admin removes job postings	Pass

Table 5.1 Summary of UAT Results

5.3 Demographic Information

The demographic data collected from the 14 participants who completed the usability test for the Career Navigator system provides useful insight into the background of the users. This information includes age range, gender, and academic year, which are relevant in understanding potential differences in user engagement and perception of the system.

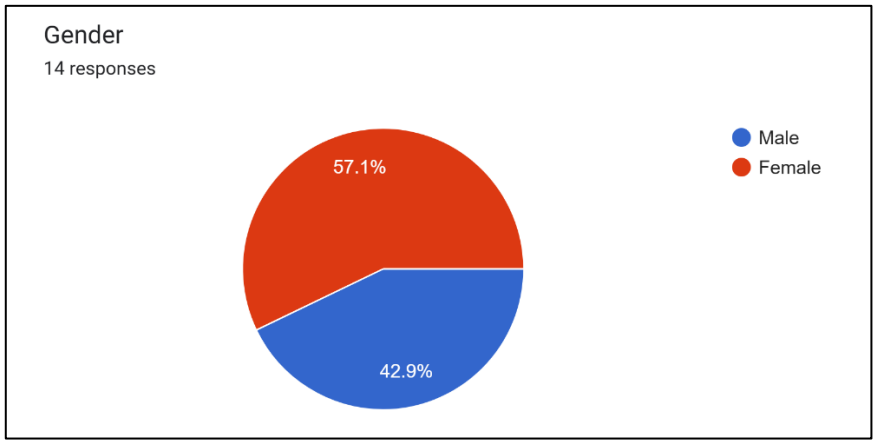


Figure 5.1 Gender Distribution of Respondents

As shown in Figure 5.1, most of the respondents were female, making up 57.1% of the sample, while 42.9% were male. This gender distribution suggests a slightly higher participation from female students, which may reflect their increased interest in career planning tools or availability during the testing period.

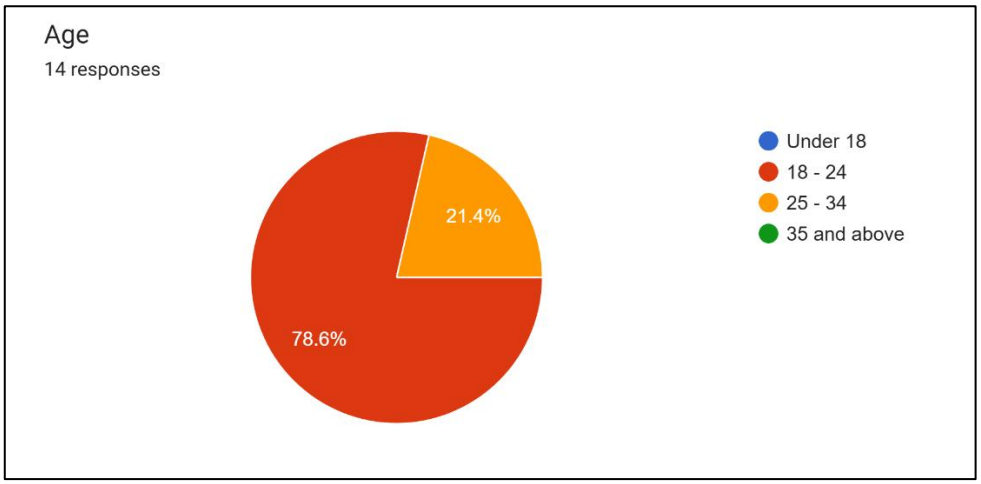


Figure 5.2 Age Group of Respondents

In terms of age, Figure 5.2 illustrates that the majority of users (78.6%) were within the 18–24 age group, which aligns with the typical undergraduate student demographic. A smaller portion, 21.4%, fell within the 25–34 age range. This distribution indicates that the system was mostly

evaluated by its intended target users—university students nearing the transition into the workforce.

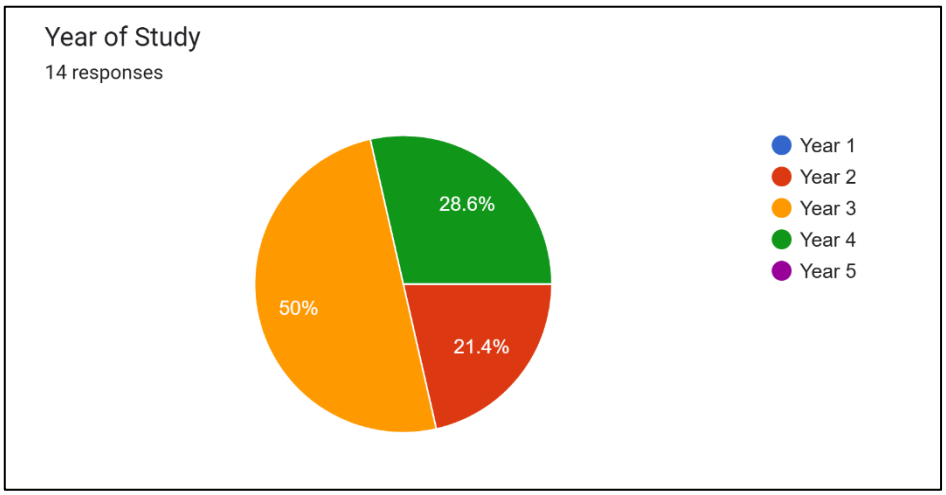


Figure 5.3 Academic Year of Respondents

Regarding academic standing, Figure 5.3 presents the breakdown of participants by year of study. The largest segment came from Year 3 students (50.0%), followed by Year 4 students (28.6%), and Year 2 students (21.4%). The concentration of users in Year 3 and Year 4 (a combined 78.6%) suggests that those closer to graduation are more inclined to engage with career support systems.

These demographic findings are valuable in interpreting the results of the usability evaluation. For example, the higher satisfaction reported by final-year students may indicate that the system aligns well with the needs of those actively preparing for employment. Their feedback is particularly meaningful, as they represent the primary audience for the Career Navigator system's core features.

5.4 System Usability Scale (SUS)

The System Usability Scale (SUS) is indeed a widely recognized and standardized questionnaire for assessing perceived usability across a broad range of products and systems, including software, websites, and digital applications. Developed in the 1980s, SUS consists of 10 items that alternate between positive and negative statements, each rated on a 5-point Likert scale, which helps reduce response bias and provides a balanced view of user experience (Lewis, 2018). In the context of career guidance systems, such as the Career Navigator, SUS has been effectively used to gather user feedback and quantify usability. For example, a recent evaluation of a university career information system using SUS found a score of 63.75, which is considered marginally acceptable and suggests that while the system is functional, there is room for improvement to enhance user satisfaction and engagement (Fitri et al., 2024). This approach not only provides actionable insights for system refinement but also supports ongoing development to better meet the needs of both end-users and stakeholders.

A total of 14 participants completed the SUS as part of the usability testing phase. Each participant's responses were scored based on the SUS methodology: for positive statements (odd-numbered), 1 is subtracted from the score; for negative statements (even-numbered), the score is subtracted from 5. The adjusted values are summed and multiplied by 2.5 to produce a final usability score out of 100. Please refer to Appendix A for a comprehensive overview of the System Usability Scale questionnaire used in this investigation.

Participant	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**	**Q7**	**Q8**	**Q9**	**Q10**	**SUS Score**
1	4	2	5	1	5	1	5	1	5	1	95.0
2	5	1	5	1	5	1	5	1	5	1	100.0
3	4	2	4	2	4	2	4	2	4	2	75.0

4	5	1	5	1	5	1	5	1	5	1	100.0
5	3	3	4	3	4	3	4	3	4	3	60.0
6	3	2	5	1	5	2	4	1	5	1	87.5
7	4	2	4	2	4	2	4	2	4	2	75.0
8	5	1	5	1	5	1	5	1	5	1	100.0
9	4	2	5	2	5	1	5	1	5	1	92.5
10	5	1	5	1	5	1	5	1	5	1	100.0
11	4	2	5	1	5	1	5	1	5	1	95.0
12	3	3	4	2	4	2	4	2	4	2	70.0
13	4	2	5	1	5	1	5	1	5	1	95.0
14	4	2	5	2	5	1	5	1	5	1	92.5

Table 5.2 SUS Score Summary for All Participants

The analysis of the System Usability Scale (SUS) responses revealed several key findings regarding the Career Navigator system's perceived usability. The average SUS score was 88.39, with individual scores ranging from 60.0 to 100.0, producing a score range of 40.0 points. According to industry benchmarks, a SUS score above 85 is considered excellent (Hyzy et al., 2022). Thus, the system's average score reflects a high level of usability and satisfaction, suggesting that the Career Navigator effectively meets the needs of its users.

Further analysis of individual question responses offers deeper insights. For ease of use, particularly items Q3 ("I thought the system was easy to use") and Q7 ("I would imagine most people would learn to use the system quickly"), 13 out of 14 participants selected either 4 or 5. This indicates users found the system intuitive and easy to learn. For system consistency (Q6), most participants disagreed with the negative statement, confirming that the system's interface and functionality were perceived as coherent and well-integrated.

Regarding user confidence (Q9), 12 participants selected the maximum score of 5, suggesting a high level of comfort and trust while using the system. Meanwhile, for complexity (Q2) and need for technical support (Q4), many of participants strongly disagreed, reinforcing the notion that the system was not only intuitive but also required minimal guidance to operate. Overall, these results affirm that participants were able to interact with the Career Navigator system effortlessly, complete essential tasks without confusion, and felt confident and satisfied with the experience.

5.4.1 Algorithm Personalization & Matching Accuracy

To further demonstrate the fulfillment of Objective 2 and Objective 3, this section evaluates the impact of the implemented content-based filtering algorithm and the personalized threshold adjustment on the recommendation outcomes.

The algorithm matches job postings with user profiles based on a weighted similarity score, calculated using a 70% weight for technical skills (via Jaccard similarity) and 30% for personality alignment (via MBTI-job description compatibility). This ensures that matches consider both the user's capabilities and workstyle preferences, aligning with the goal of delivering context-aware recommendations.

To support Objective 3, an adjustable threshold was implemented in the user interface, allowing users to dynamically refine how strict or broad the job matching process should be. During usability testing, participants adjusted the threshold and observed changes in the number and relevance of job suggestions in real time. Users found this feature especially helpful for exploring more diverse options or narrowing results to highly relevant roles.

User feedback validated the effectiveness of the algorithm and the threshold mechanism, with 92.9% of participants (13 out of 14) finding the recommendations useful and aligned with their profile input. The ability to personalize job suggestions based on user-defined thresholds reflects the system's adaptability and supports a tailored job search experience, the core of Objective 3.

These observations demonstrate that the algorithm not only functioned correctly but also significantly enhanced the user experience, making the job search process more accurate, personalized, and engaging.

5.5 Respondent's Feedback

To gain qualitative insights alongside the SUS scores, participants were asked to share their opinions about the system's usefulness, design, and potential improvements. This section summarizes the open-ended feedback provided by the 14 respondents.

Most users expressed positive sentiments about the system's design and functionality. Comments highlighted the simplicity of the interface, the usefulness of job matching based on MBTI personality types, and the value of admin filtering in ensuring data accuracy. These responses suggest that the system is perceived as both intuitive and effective in its intended purpose of supporting career exploration.

Participants also shared helpful suggestions for improvement. Several recommended integrating a feature that allows users to apply directly to job listings, which would enhance the system's practicality. Others proposed a more automated approach to detecting MBTI types,

eliminating the need to upload screenshots. There were also requests for additional features, such as allowing companies to register and post their own job advertisements.

In terms of recommendation, participants were asked, “Would you recommend this system to others?” The results are summarized in Figure 5.11. Out of 14 participants, 13 (85.7%) answered “Yes”, while 1 (14.3%) responded “Maybe.” No participants selected “No,” indicating high levels of user satisfaction and perceived value.

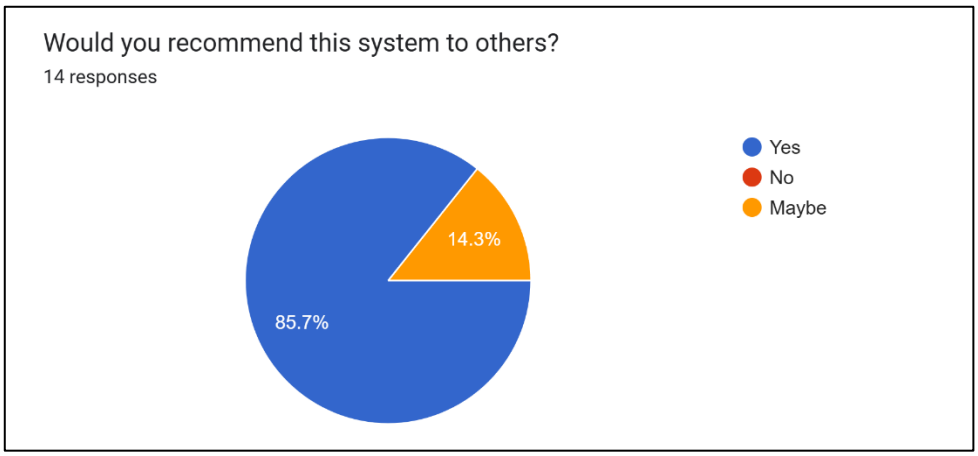


Figure 5.4 Willingness to Recommend the System – 85.7% Yes, 14.3% Maybe

This feedback reflects both the strengths of the Career Navigator system and the potential directions for enhancement. While users found the system helpful, particularly for students nearing graduation, their suggestions point toward meaningful opportunities to improve interactivity, automation, and employer involvement in future iterations.

5.6 Summary

This chapter evaluated the Career Navigator system through usability testing involving 14 participants. The System Usability Scale (SUS) results showed an average score of 88.39, indicating excellent usability. Most users found the system easy to use, consistent, and confidence-boosting. Final-year students, in particular, responded positively, showing the system's relevance for career preparation. Additionally, 85.7% of participants said they would recommend the system. Overall, the system met its usability goals and received constructive suggestions for future improvements.

CHAPTER 6 CONCLUSION AND FUTURE WORK

6.1 Introduction

This project ends in Chapter 6, which provides an overview of the successes, challenges, and future development opportunities. The project's goal was to create Career Navigator, a web-based career recommendation system that matches recent graduates in computer science and information technology with appropriate positions based on their educational background, technical proficiency, and personal characteristics. The system incorporates a content-based filtering algorithm to determine job suitability and was created with Python, Flask, MySQL, and web technologies.

6.2 Achievements

The main goals of the project are effectively achieved by the Career Navigator system. To help recent graduates in computer science and information technology find employment opportunities that complement their technical skills, educational background, and personality, a fully functional prototype was created. Users can register, enter pertinent profile information, and get job recommendations from the system using a similarity-matching algorithm. Admins manage job postings and validate personality test uploads, while Users receive career recommendations based on their personal and professional profiles. The implementation includes a dual-user role system. Users can modify the degree of similarity that must be present in the recommendations through a threshold-based mechanism, which further improves personalization. By combining HTML, CSS, JavaScript, Flask, MySQL, and Python, the project has produced a platform that shows that tailored career matching for tech graduates is feasible.

6.3 Limitations

The current version of the system still has several limitations in spite of its achievements. First off, because the system does not yet support dynamic integration with external job platforms or Application Programming Interfaces (APIs), job postings must be manually added by the administrator. This restricts job listings' scalability and relevance in real time. Second, the application feature is not yet accessible; users are unable to apply for jobs directly through the system and can only view job matches. Furthermore, personality test submissions are uploaded as images, necessitating administrators' manual verification which a process that can be subjective and inefficient. Additionally, because user testing has not yet been done, usability, performance, and functional assessments have not yet been carried out. These drawbacks draw attention to areas that need to be fixed before the system is deemed workable or appropriate for widespread implementation.

6.4 Future Works

The Career Navigator system will be improved in the future with an emphasis on scalability, automation, and interactivity. The addition of a third user role, Job Provider, which will enable businesses to register and post job openings straight to the platform, will be one of the biggest enhancements. This would give users access to a larger and more varied job pool in addition to automating updates to job listings. System utility will also be improved by adding an application feature that allows users to apply for jobs directly through the platform and submit their resumes. In terms of automation, the system might be able to extract text from personality test images by integrating Optical Character Recognition (OCR) technology, which would lessen the administrative workload. An MBTI or personality test could be integrated into the system to enhance personalization and data quality. Furthermore, the system would be able to provide current and location-relevant job recommendations if it were integrated with external

job APIs like Indeed, Glassdoor, or JobStreet. To ensure that the platform is both technically sound and user-friendly, usability testing and performance benchmarking are essential next steps for confirming the efficacy of the recommendation algorithm and overall system functionality.

6.5 Chapter Summary

In summary, this chapter has reviewed the project's major accomplishments, identified current limitations, and proposed potential future improvements. The Career Navigator system achieves its goal of providing personalized job recommendations to CS and IT graduates by utilizing a content-based filtering approach enhanced with personality profiling. To reach its full potential, future work must address its manual processes, limited user roles, and lack of interactivity. The system can be expanded into a comprehensive career platform by incorporating new features such as dynamic job posting, automated test verification, and direct application capabilities. These enhancements will help to significantly reduce graduate underemployment by providing more intelligent, responsive, and inclusive career guidance tools for the technology sector.

REFERENCES

Adeagbo, M. A., Akhigbe, B. I., & Afolabi, B. S. (2019). Towards a job recommender model: An architectural-based approach. International Journal of Advanced Trends in Computer Science and Engineering, 8(6), 3258–3264. https://doi.org/10.30534/ijatcse/2019/94862019

Al-Dhari, S., & Al-Alawi, A. I. (2023). The Application of Data Analytics to Career Choice Prediction: A Literature Review. 2023 International Conference on Cyber Management and Engineering, CyMaEn 2023, 260–265. https://doi.org/10.1109/CyMaEn57228.2023.10051101

Alsaif, S. A., Sassi Hidri, M., Eleraky, H. A., Ferjani, I., & Amami, R. (2022). Learning-Based Matched Representation System for Job Recommendation. *Computers*, *11*(11), 161. https://doi.org/10.3390/computers11110161

Bansal, K., & Jain, V. (2020). Performance Analysis of Collaborative Filtering based Recommendation System on Similarity Threshold. Proceedings of the 4th International Conference on Computing Methodologies and Communication, ICCMC 2020, 442–448. https://doi.org/10.1109/ICCMC48092.2020.ICCMC-00083

Cesconetto, J., Silva, L. A., Leithardt, R. Q. V., Caceres, M. N., Silva, L. A., & Garcia, N. M. (2020). PRIPRO:Solution for user profile control and management based on data privacy. *2022 17th Iberian Conference on Information Systems and Technologies (CISTI)*, 1–6. https://doi.org/10.23919/cisti49556.2020.9140810

Chauhan, N., Singh, M., Verma, A., Parasher, A., & Budhiraja, G. (2019). Implementation of database using python flask framework. *International Journal of Engineering and Computer Science*, 8, 24894-24899. https://doi.org/10.18535/ijecs/v8i12.4390.

D, N. M. D. A., M, N. M. N. S., A, N. M. K. G., R, N. M. G. A., & B, N. P. M. B. (2024). User Management System using Admin Panel. *International Journal of Advanced Research in Science Communication and Technology*, 33–35. https://doi.org/10.48175/ijarsct-15307

Daniel, P. (2025). NXTGEN - Intelligent Career Navigator: an AI-Powered Career Guidance System. *INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT*, *09*(04), 1–9. https://doi.org/10.55041/ijsrem46255

Fitri, L. E., Setiawan, D., & Kaswari, T. (2024). USABILITY ANALYSIS OF THE CAREER SYSTEM AT JAMBI UNIVERSITY USING SUS AND HEURISTIC EVALUATION TO IMPROVE EMPLOYER SATISFACTION. *JOURNAL OF BUSINESS STUDIES AND MANGEMENT REVIEW*, *8*(1), 114–123. https://doi.org/10.22437/jbsmr.v8i1.41607

Guntupalli, U. G. S., Pandala, M. L., Veeranki, D. T., & Kumbha, P. (2024). Carrer Compass: A Career Path Recommender using Machine Learning. 5th International Conference on Electronics and Sustainable Communication Systems, ICESC 2024 - Proceedings, 1937–1941. https://doi.org/10.1109/ICESC60852.2024.10689897

Hakimi, S., Hejazi, E., & Lavasani, M. G. (2011). The relationships between personality traits and students' academic achievement. Procedia - Social and Behavioral Sciences, 29, 836–845. https://doi.org/10.1016/j.sbspro.2011.11.312

Hawari, A. R., Munawir, M., Zulfan, Z., Satria, D. S. D., & Susmanto, S. (2022). Job recommendation System using the Content-Based Filtering method. *Proceedings of International Conference on Multidiciplinary Research*, *5*(2), 138–140. https://doi.org/10.32672/pic-mr.v5i2.5419

Hiredly. (2023). *Hiredly job portal interface*. Retrieved from Hiredly: https://hiredly.com

Hyzy, M., Bond, R., Mulvenna, M., Bai, L., Dix, A., Leigh, S., & Hunt, S. (2022). System Usability Scale Benchmarking for digital health apps: Meta-analysis. *JMIR Mhealth and Uhealth*, *10*(8), e37290. https://doi.org/10.2196/37290

Kamaruddin, N., Rahman, A. W. A., & Lawi, R. A. M. (2019). Jobseeker-industry matching system using automated keyword selection and visualization approach. *Indonesian Journal of Electrical Engineering and Computer Science*, *13*(3), 1124–1129. https://doi.org/10.11591/ijeecs.v13.i3.pp1124-1129

Karnewar, B., Purankar, S., Borkar, P., Jagtap, S., & Shirbhate, D. (2024). Navigating Futures: A Paper on Comprehensive Review of Career Guidance Portal.

Kenthapadi, K., Le, B., & Venkataraman, G. (2017). Personalized job recommendation system at LinkedIn: Practical challenges and lessons learned. *RecSys 2017 - Proceedings of the 11th ACM Conference on Recommender Systems*, 346–347. https://doi.org/10.1145/3109859.3109921

Kramer, M. (2018). Best Practices In Systems Development Lifecycle: An Analyses Based On The Waterfall Model. *Review of Business and Finance Studies*, 9, 77-84.

Kumar, N., Gupta, M., Sharma, D., & Ofori, I. (2022). Technical Job Recommendation System Using APIs and Web Crawling. Computational Intelligence and Neuroscience, 2022. https://doi.org/10.1155/2022/7797548

Lewis, J. R. (2018). The system usability scale: past, present, and future. *International Journal of Human-Computer Interaction*, *34*(7), 577–590. https://doi.org/10.1080/10447318.2018.1455307

Microsoft. (2022). *Visual Studio Code [Computer Software]*. Retrieved from https://code.visualstudio.com

Mulay, A., Sutar, S., Patel, J., Chhabria, A., & Mumbaikar, S. (2022). Job Recommendation System Using Hybrid Filtering. ITM Web of Conferences, 44, 02002. https://doi.org/10.1051/itmconf/20224402002

Mwita, K., Mwilongo, N., & Mwamboma, I. (2024). The role of soft skills, technical skills and academic performance on graduate employability. *International Journal of Research in Business and Social Science (2147- 4478)*. https://doi.org/10.20525/ijrbs.v13i5.3457.

Nguyen, T. (2013). Online Marketing: A Study in E–Service Section: Case Study: Jobstreet. com.

Nickerson, C. (2024). What is an Ambivert Personality? *Simply Psychology*. https://www.simplypsychology.org/ambivert.html

Ozkaya, M., & Erata, F. (2020). A survey on the practical use of UML for different software architecture viewpoints. *Inf. Softw. Technol.*, 121, 106275. https://doi.org/10.1016/j.infsof.2020.106275.

Rahhal, I., Carley, K., Ismail, K., & Sbihi, N. (2022). Education Path: Student orientation based on the job market needs. IEEE Global Engineering Education Conference, EDUCON, 2022-March, 1365–1373. https://doi.org/10.1109/EDUCON52537.2022.9766771

Rahman, M., Clougherty, J., Hewitt, S., & Clougherty, E. (2014). Personalized Job Matching. .

Rahman, S., Habiba, K., Roy, S., & Nur, F. N. (2023). Job Title Prediction and Recommendation System for IT Professionals. 2023 5th International Conference on Sustainable Technologies for Industry 5.0, STI 2023. https://doi.org/10.1109/STI59863.2023.10464457

Rashid, A. H. A., Mohamad, M., Masrom, S., & Selamat, A. (2022). Student Career Recommendation System Using Content-Based Filtering Method. *2022 3rd International Conference on Artificial Intelligence and Data Sciences: Championing Innovations in Artificial Intelligence and Data Sciences for Sustainable Future, AiDAS 2022 - Proceedings*, 60–65. https://doi.org/10.1109/AiDAS56890.2022.9918766

Relan, K. (2019). Beginning with Flask. *Building REST APIs with Flask*. https://doi.org/10.1007/978-1-4842-5022-8_1.

Rodriguez, L., & Chavez, E. (2019). Feature Selection for Job Matching Application using Profile Matching Model. *2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*, 263-266. https://doi.org/10.1109/CCOMS.2019.8821682.

Sam, V. (2020). Impacts of educational mismatches on job satisfaction: The case of university graduates in Cambodia. International Journal of Manpower, 41(1), 84–99. https://doi.org/10.1108/IJM-07-2018-0229

Shimpi, P., Balinge, B., Golait, T., Parthasarathi, S., Arunima, C. J., & Mali, Y. K. (2024). Job Crafter - The One-Stop Placement Portal. 2024 15th International Conference on Computing Communication and Networking Technologies, ICCCNT 2024. https://doi.org/10.1109/ICCCNT61001.2024.10725010

Siswipraptini, P. C., Warnars, H. L. H. S., Ramadhan, A., & Budiharto, W. (2022). Trends and Characteristics of Career Recommendation Systems for Fresh Graduated Students. 2022 10th International Conference on Information and Education Technology, ICIET 2022, 355–361. https://doi.org/10.1109/ICIET55102.2022.9779037

Siswipraptini, P. C., Warnars, H. L. H. S., Ramadhan, A., & Budiharto, W. (2024). Personalized Career-Path Recommendation Model for Information Technology Students in Indonesia. *IEEE Access*, *12*, 49092–49105. https://doi.org/10.1109/ACCESS.2024.3381032

Štimac, H., & Tanasić, K. B. (2023). COMPETENCIES AND SKILLS: GAP BETWEEN HIGHER EDUCATION AND LABOUR MARKET. In Economic Thought and Practice (Vol. 32, Issue 2, pp. 615–628). University of Dubronvnik. https://doi.org/10.17818/EMIP/2023/2.15

Thali, R., Mayekar, S., More, S., Barhate, S., & Selvan, S. (2023). Survey on Job Recommendation Systems using Machine Learning. *International Conference on Innovative Data Communication Technologies and Application, ICIDCA 2023 - Proceedings*, 453–457. https://doi.org/10.1109/ICIDCA56705.2023.10100122

Tuan, A. C., Dang, M. T., Nam, H., DO, Solanki, V. K., Torres, J., Crespo, R. G., & Nguyen, T. N. A. (2024). Ontology and its applications in skills matching in job recruitment. *Applied Ontology*, *19*(3), 287–306. https://doi.org/10.3233/ao-240019

Welcome to Flask — Flask Documentation (3.1.x). (n.d.). https://flask.palletsprojects.com/en/stable/

Wu, R., Zhao, X., Li, Z., & Xie, Y. (2024). The role of employee personality in employee satisfaction and turnover: insights from online employee reviews. *Personnel Review*, *53*(7), 1581–1611. https://doi.org/10.1108/pr-04-2023-0309

APPENDIX A

System Usability Scale (SUS)

	Strongly Disagree 1	2	3	4	Strongly Agree 5
1. I think I would like to use this system frequently.					
2. I found the system unnecessarily complex.					
3. I thought the system was easy to use.					
4. I think I would need the support of a technical person to use this system.					
5. I found the various functions in this system well integrated.					
6. I thought there was too much inconsistency in this system.					
7. I would imagine most people would learn to use this system very quickly.					
8. I found the system very cumbersome to use.					
9. I felt very confident using the system.					
10. I needed to learn a lot of things before I could get going with this system.					

APPENDIX B

Source Code of CareerNavigator

app.py

```
from flask import Flask, render_template, request, redirect, url_for,
     flash, session

import mysql.connector

from      werkzeug.security      import      generate_password_hash,
     check_password_hash

from preprocessing import compute_similarity


app = Flask(__name__)

app.secret_key = 'your_secret_key'  # Change to a strong secret key!


def get_db_connection():

    return mysql.connector.connect(

        host='localhost',

        user='CareerNavigator',

        password='123',

        database='career_navigator'

    )


def is_profile_complete(user):

     required_fields = ['Age', 'Gender', 'University', 'Edu_Level',
      'Major', 'Tech_Skills', 'MBTI']

     return all(user.get(field) not in (None, '', []) for field in
      required_fields)


@app.route('/')

def home():
```

```
    return render_template('index.html')   # Welcome page before
    login/register


@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        password = request.form['password']
        confirm = request.form['confirm_password']

        if password != confirm:
            flash("Passwords do not match!")
            return redirect(url_for('register'))

        hashed_password = generate_password_hash(password)

        try:
            conn = get_db_connection()
            cursor = conn.cursor(dictionary=True)

             cursor.execute("SELECT * FROM users WHERE Email = %s",
    (email,))
            user = cursor.fetchone()
            if user:
                flash("Email already registered!")
                return redirect(url_for('register'))
```

```
            cursor.execute(
                "INSERT INTO users (Name, Email, password_hash) VALUES
      (%s, %s, %s)",
                 (name, email, hashed_password)
            )
            conn.commit()
            flash("Registration successful! Please login.")
            return redirect(url_for('login'))

        except Exception as e:
            print("Error during registration:", e)
            flash("Registration failed. Please try again.")
            return redirect(url_for('register'))

        finally:
            cursor.close()
            conn.close()

    return render_template('register.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']

        try:
            conn = get_db_connection()
```

```
        cursor = conn.cursor(dictionary=True)


         cursor.execute("SELECT * FROM users WHERE Email = %s",
  (email,))
        user = cursor.fetchone()


          if user and check_password_hash(user['password_hash'],
  password):
            session['user_id'] = user['user_id']
            session['user_name'] = user['Name']
            flash("Logged in successfully!")
            return redirect(url_for('dashboard'))


        else:
            flash("Invalid email or password!")
            return redirect(url_for('login'))


    except Exception as e:
        print("Login error:", e)
        flash("Login failed. Please try again.")
        return redirect(url_for('login'))


    finally:
        cursor.close()
        conn.close()


return render_template('login.html')
```

```
@app.route('/dashboard')
def dashboard():
    if 'user_id' not in session:
        flash("Please login first.")
        return redirect(url_for('login'))

    user_name = session.get('user_name')

    conn = get_db_connection()
    cursor = conn.cursor(dictionary=True)
      cursor.execute("SELECT * FROM users WHERE user_id = %s",
      (session['user_id'],))
    user = cursor.fetchone()
    cursor.close()
    conn.close()

    if not is_profile_complete(user):
        flash("Please complete your profile to continue.")
       return render_template('dashboard.html', user_name=user_name,
     profile_complete=False)

      return render_template('dashboard.html', user_name=user_name,
     profile_complete=True)

@app.route('/profile/setup', methods=['GET', 'POST'])
def profile_setup():
    if 'user_id' not in session:
        flash("Please login first.")
        return redirect(url_for('login'))
```

```
user_id = session['user_id']
conn = get_db_connection()
cursor = conn.cursor(dictionary=True)

if request.method == 'POST':
    age = request.form['age']
    gender = request.form['gender']
    university = request.form['university']
    edu_level = request.form['edu_level']
    major = request.form['major']
    tech_skills = request.form['tech_skills']
    mbti = request.form['mbti']

    cursor.execute("""
        UPDATE users
            SET Age=%s, Gender=%s, University=%s, Edu_Level=%s,
 Major=%s, Tech_Skills=%s, MBTI=%s
        WHERE user_id=%s
    """, (age, gender, university, edu_level, major, tech_skills,
 mbti, user_id))
    conn.commit()
    flash("Profile updated successfully!")
    cursor.close()
    conn.close()
    return redirect(url_for('job_recommendations'))

 cursor.execute("SELECT * FROM users WHERE user_id = %s",
 (user_id,))
```

```
    user = cursor.fetchone()
    cursor.close()
    conn.close()
    return render_template('profile_setup.html', user=user)

@app.route('/jobs')
def job_recommendations():
    if 'user_id' not in session:
        flash("Please login first.")
        return redirect(url_for('login'))

    user_id = session['user_id']

    conn = get_db_connection()
    cursor = conn.cursor(dictionary=True)
      cursor.execute("SELECT * FROM users WHERE user_id = %s",
      (user_id,))
    user = cursor.fetchone()

    if not is_profile_complete(user):
        flash("Please complete your profile first.")
        cursor.close()
        conn.close()
        return redirect(url_for('profile_setup'))

    cursor.execute("SELECT * FROM jobs")
    jobs = cursor.fetchall()
    cursor.close()
```

```
    conn.close()


    scored_jobs = []
    for job in jobs:
        score = compute_similarity(user, job)
        scored_jobs.append({'job': job, 'score': score})


    scored_jobs.sort(key=lambda x: x['score'], reverse=True)


     return  render_template('job_recommendations.html',  user=user,
     jobs=scored_jobs)


def personality_requirements(profile_mbti, job_description):
    introvert_keywords = [
        "independent work", "analytical skills", "detail-oriented",
     "attention to detail",
       "written communication", "self-motivated", "research-focused",
     "minimal interaction", "data analysis"
    ]


    ambivert_keywords = [
             "team  player",  "flexible",  "collaborative",  "good
     communication skills", "adaptable",
           "adaptability",  "balanced  communication",  "teamwork",
     "problem-solving",
       "works well independently and in teams", "balance of social
     interaction and solitary work"
    ]


    extrovert_keywords = [
```

```
        "strong communication skills", "team-oriented", "leadership",
      "communication skills", "teamwork",
              "networking",  "customer  interaction",  "collaborative
      environment", "public speaking"
    ]

    keywords_map = {
        "introvert": introvert_keywords,
        "ambivert": ambivert_keywords,
        "extrovert": extrovert_keywords
    }

    keywords = keywords_map.get(profile_mbti.lower(), [])

    job_desc_lower = job_description.lower()

    matched_keywords = [kw for kw in keywords if kw in job_desc_lower]

     return matched_keywords if matched_keywords else ["No specific
      personality-based requirements found."]

@app.route('/search')
def job_search():
    if 'user_id' not in session:
        flash("Please login first.")
        return redirect(url_for('login'))

    query = request.args.get('q', '').lower().strip()
    threshold = float(request.args.get('threshold', 0))
```

```
location_filter = request.args.get('location', '').strip()
job_type_filter = request.args.get('job_type', '').strip()
page = int(request.args.get('page', 1))
per_page = 10

if not query:
    flash("Please enter a search keyword.")
    return redirect(url_for('dashboard'))

user_id = session['user_id']

conn = get_db_connection()
cursor = conn.cursor(dictionary=True)
  cursor.execute("SELECT * FROM users WHERE user_id = %s",
  (user_id,))
user = cursor.fetchone()

if not is_profile_complete(user):
    flash("Please complete your profile first.")
    cursor.close()
    conn.close()
    return redirect(url_for('profile_setup'))

cursor.execute("SELECT * FROM jobs")
jobs = cursor.fetchall()

# Extract unique locations and job types for filters
```

```
locations = sorted(set(job.get('location', '') for job in jobs if
 job.get('location')))

job_types = sorted(set(job.get('job_type', '') for job in jobs if
 job.get('job_type')))


filtered_jobs = []

user_mbti = user.get('MBTI', '').lower() if user else ''


for job in jobs:

    combined_text = " ".join([

        job.get('job_title', '').lower(),

        job.get('descriptions', '').lower(),

        job.get('required_skills', '').lower()

    ])

    if query in combined_text:

            if location_filter and job.get('location', '') !=
 location_filter:

            continue

            if job_type_filter and job.get('job_type', '') !=
 job_type_filter:

            continue


        score = compute_similarity(user, job)

        if score < threshold:

            continue


          personality_reqs = personality_requirements(user_mbti,
 job.get('descriptions', ''))


        job_entry = {
```

```
            'job': job,
            'score': score,
            'personality_requirements': personality_reqs
        }
        filtered_jobs.append(job_entry)

filtered_jobs.sort(key=lambda x: x['score'], reverse=True)

total = len(filtered_jobs)
start = (page - 1) * per_page
end = start + per_page
paginated_jobs = filtered_jobs[start:end]

total_pages = (total + per_page - 1) // per_page

cursor.close()
conn.close()

return render_template(
    'job_recommendations.html',
    user=user,
    jobs=paginated_jobs,
    search_term=query,
    threshold=threshold,
    location=location_filter,
    locations=locations,
    job_type=job_type_filter,
    job_types=job_types,
```

```
        page=page,
        total_pages=total_pages
    )

@app.route('/profile', methods=['GET', 'POST'])
def profile():
    if 'user_id' not in session:
        flash("Please login first.")
        return redirect(url_for('login'))

    user_id = session['user_id']
    conn = get_db_connection()
    cursor = conn.cursor(dictionary=True)

    if request.method == 'POST':
        age = request.form['age']
        gender = request.form['gender']
        university = request.form['university']
        edu_level = request.form['edu_level']
        major = request.form['major']
        tech_skills = request.form['tech_skills']
        mbti = request.form['mbti']

        cursor.execute("""
            UPDATE users
                SET Age=%s, Gender=%s, University=%s, Edu_Level=%s,
     Major=%s, Tech_Skills=%s, MBTI=%s
            WHERE user_id=%s
```

```
        """, (age, gender, university, edu_level, major, tech_skills,
      mbti, user_id))
        conn.commit()
        flash("Profile updated successfully!")
        cursor.close()
        conn.close()
        return redirect(url_for('profile'))

      cursor.execute("SELECT * FROM users WHERE user_id = %s",
      (user_id,))
    user = cursor.fetchone()
    cursor.close()
    conn.close()
    return render_template('profile.html', user=user)

@app.route('/job/<int:job_id>')
def job_details(job_id):
    if 'user_id' not in session:
        flash("Please login first.")
        return redirect(url_for('login'))

    conn = get_db_connection()
    cursor = conn.cursor(dictionary=True)
    cursor.execute("SELECT * FROM jobs WHERE job_id = %s", (job_id,))
    job = cursor.fetchone()
    cursor.close()
    conn.close()
```

```
    if not job:
        flash("Job not found.")
        return redirect(url_for('job_recommendations'))

    return render_template('job_details.html', job=job)

@app.route('/logout')
def logout():
    session.clear()
    flash("You have been logged out.")
    return redirect(url_for('home'))

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

requirements.txt

```
Flask==2.3.3
Flask-SQLAlchemy==3.1.1
Flask-Login==0.6.3
Flask-WTF==1.2.1
mysqlclient==2.2.1
Werkzeug==2.3.7
python-dotenv==1.0.0
scikit-learn==1.3.1
pandas==2.1.1
```

preprocessing.py

```
def preprocess_skills(skills_string):

    if not skills_string:

        return set()

          skills    =    [skill.strip().lower()    for    skill    in
     skills_string.split(',')]

    return set(skills)


def extract_personality_matches(job_description):

    introvert_keywords = [

         "independent work", "analytical skills", "detail-oriented",
     "attention to detail",

       "written communication", "self-motivated", "research-focused",
     "minimal interaction", "data analysis"

    ]

    ambivert_keywords = [

        "team player", "flexible", "collaborative",

          "good communication skills", "adaptable", "adaptability",
     "balanced communication", "teamwork",

         "problem-solving", "works well independently and in teams",
     "balance of social interaction and solitary work"

    ]

    extrovert_keywords = [

       "strong communication skills", "team-oriented", "leadership",
     "communication skills",

               "teamwork",   "networking",   "customer   interaction",
     "collaborative environment", "public speaking"

    ]



    desc = job_description.lower()
```

```
    introvert_matches = sum(kw in desc for kw in introvert_keywords)
    ambivert_matches = sum(kw in desc for kw in ambivert_keywords)
    extrovert_matches = sum(kw in desc for kw in extrovert_keywords)

    return {
        "introvert": introvert_matches,
        "ambivert": ambivert_matches,
        "extrovert": extrovert_matches
    }

def personality_match(user_personality, job_description):
    matches = extract_personality_matches(job_description)

    if user_personality == "introvert":
        return matches["introvert"] > 0
    elif user_personality == "extrovert":
        return matches["extrovert"] > 0
    elif user_personality == "ambivert":
        # Ambiverts match all jobs
        return True
    else:
        return False

def compute_similarity(user_profile, job_posting):
     user_skills = preprocess_skills(user_profile.get('Tech_Skills',
      ''))
```

```
    job_skills = preprocess_skills(job_posting.get('required_skills',
      ''))

    intersection = user_skills.intersection(job_skills)
    union = user_skills.union(job_skills)
    skill_sim = len(intersection) / len(union) if union else 0

    personality_sim = 1 if personality_match(user_profile.get('MBTI',
      '').lower(), job_posting.get('descriptions', '')) else 0

    final_score = 0.7 * skill_sim + 0.3 * personality_sim
    return final_score
```

styles.css

```
/* Reset default styles */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: 'Poppins', sans-serif; /* Modern clean font */
    background-color: #f9f9f9;
    color: #333;
}

/* Header and Navigation */
header {
```

```
    background-color: #6C4F8C; /* Soft purple background */
    padding: 20px 40px;
    color: white;
}

nav {
    display: flex;
    justify-content: space-between;
    align-items: center;
}

nav .logo img {
    height: 80px;
}

nav ul {
    display: flex;
    list-style-type: none;
}

nav ul li {
    margin: 0 15px;
}

nav ul li a {
    color: white;
    text-decoration: none;
    font-weight: 700;
```

```
    font-size: 18px;
    transition: color 0.3s ease;
}

nav ul li a:hover {
    text-decoration: underline;
    color: #d2c6f7;
}

header nav.header-nav {
  display: flex;
  align-items: center;
  justify-content: space-between;
  gap: 20px;
}

header nav.header-nav .logo {
  flex-shrink: 0;
}

header nav.header-nav .welcome-name {
  font-weight: 700;
  font-size: 1.2rem;
  color: white;
  white-space: nowrap;
  margin-left: 20px;
  flex-grow: 1;
  /* Align welcome message left, pushing nav to right */
```

```
}

header nav.header-nav ul {
  display: flex;
  list-style: none;
  margin-left: auto;
  gap: 15px;
}

header nav.header-nav ul li a {
  color: white;
  text-decoration: none;
  font-weight: 700;
  font-size: 18px;
  transition: color 0.3s ease;
}

header nav.header-nav ul li a:hover {
  text-decoration: underline;
  color: #d2c6f7;
}

/* Hero Section */
.hero {
    text-align: center;
    padding: 100px 20px;
    background-color: #6C4F8C;
    color: white;
```

```
}

.hero h1 {
    font-size: 56px; /* Big impactful heading */
    font-weight: 700;
    margin-bottom: 20px;
}

.hero p {
    font-size: 20px;
    margin: 20px 0;
}

.hero button {
    padding: 12px 30px;
    margin: 10px;
    background-color: white;
    color: #6C4F8C;
    border: none;
    cursor: pointer;
    font-weight: 700;
    font-size: 18px;
    border-radius: 5px;
    transition: background-color 0.3s ease;
}

.hero button:hover {
    background-color: #fff5f7;
```

```
}

/* About Section */
section.about {
    display: flex;
    justify-content: space-around;
    margin: 50px 0;
    padding: 20px;
    background-color: #fff;
}

section.about div {
    width: 30%;
    padding: 20px;
    background-color: #f9f9f9;
    border-radius: 10px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}

section.about h2 {
    font-size: 24px;
    margin-bottom: 15px;
    font-weight: 700;
    color: #6C4F8C;
}

section.about p {
    font-size: 16px;
```

```
    color: #666;
}

/* Footer Section */
footer {
    background-color: #6C4F8C;
    padding: 20px 40px;
    color: white;
    text-align: center;
    font-weight: 600;
    border-radius: 0 0 12px 12px;
    margin-top: 50px;
}

footer p {
    font-size: 14px;
}

/* Buttons */
button {
    padding: 10px 20px;
    background-color: #6C4F8C;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    font-weight: 700;
    font-size: 18px;
```

```
    transition: background-color 0.3s ease;
}

button:hover {
    background-color: #5a3e7b;
}

/* Container for login and register forms */
.container {
    width: 100%;
    max-width: 450px;
    margin: 60px auto;
    padding: 40px 30px;
    background-color: #f2f0fa;
    border-radius: 15px;
    box-shadow: 0 8px 20px rgba(107, 93, 211, 0.3);
    text-align: center;
}

/* Logo */
.container img.logo {
    max-width: 120px;
    margin-bottom: 25px;
}

/* Heading */
.container h2 {
    color: #6C4F8C;
```

```
    font-weight: 700;
    font-size: 28px;
    margin-bottom: 30px;
}

/* Input fields for both login and register */
.container input[type="text"],
.container input[type="email"],
.container input[type="password"] {
    width: 100%;
    padding: 14px 18px;
    margin-bottom: 20px;
    border: 2px solid #6C4F8C;
    border-radius: 10px;
    font-size: 16px;
    color: #4a3e87;
    font-weight: 500;
    transition: border-color 0.3s ease;
}

.container input::placeholder {
    color: #a8a0d2;
}

.container input:focus {
    outline: none;
    border-color: #5040a0;
}
```

```
/* Forgot password link */
.container .forgot-password {
    display: block;
    text-align: right;
    font-size: 14px;
    margin-bottom: 30px;
    color: #6C4F8C;
    font-weight: 600;
    text-decoration: none;
}

.container .forgot-password:hover {
    text-decoration: underline;
}

/* Login button */
.container button.login-btn,
.container button.register-btn {
    width: 100%;
    background-color: #6C4F8C;
    color: white;
    font-weight: 700;
    padding: 15px;
    font-size: 18px;
    border: none;
    border-radius: 10px;
    cursor: pointer;
```

```
    transition: background-color 0.3s ease;
}

.container button.login-btn:hover,
.container button.register-btn:hover {
    background-color: #5040a0;
}

/* Register prompt */
.container .register-prompt {
    margin-top: 25px;
    font-size: 16px;
    color: #6C4F8C;
}

.container .register-prompt a {
    font-weight: 700;
    color: #5040a0;
    text-decoration: none;
}

.container .register-prompt a:hover {
    text-decoration: underline;
}

.login-prompt {
    margin-top: 25px;
    font-size: 16px;
```

```
    color: #6C4F8C;
    font-weight: 600;
    text-align: center;
}

.login-prompt a {
    font-weight: 700;
    color: #5040a0;
    text-decoration: none;
}

.login-prompt a:hover {
    text-decoration: underline;
}

/* OR separator for register */
.container .or-separator {
    margin: 30px 0;
    display: flex;
    align-items: center;
    font-weight: 700;
    color: #6C4F8C;
    font-size: 14px;
}

.container .or-separator::before,
.container .or-separator::after {
    content: "";
```

```
    flex-grow: 1;
    height: 1px;
    background-color: #d0ccef;
    margin: 0 10px;
    border-radius: 2px;
}

/* Google signup button */
.container button.google-btn {
    width: 100%;
    background-color: white;
    color: #6C4F8C;
    font-weight: 700;
    padding: 12px 0;
    font-size: 16px;
    border: 2px solid #6C4F8C;
    border-radius: 10px;
    cursor: pointer;
    display: flex;
    align-items: center;
    justify-content: center;
    gap: 12px;
    transition: background-color 0.3s ease, color 0.3s ease;
}

.container button.google-btn img {
    height: 22px;
    width: 22px;
```

```
}

.container button.google-btn:hover {
    background-color: #6C4F8C;
    color: white;
}

/* Shared container */
.dashboard-container {
  max-width: 900px;
  margin: 80px auto 80px;
  padding: 0 30px;
  text-align: center;
  color: #5a3e7b;
}

/* Welcome title */
.welcome-title {
  font-size: 2.8rem;
  font-weight: 700;
  margin-bottom: 40px;
}

/* Welcome card (for incomplete profile) */
.welcome-card {
  background: white;
  border-radius: 18px;
  padding: 60px 40px;
```

```
  box-shadow: 0 20px 40px rgba(107, 93, 211, 0.15);
  display: flex;
  flex-direction: column;
  gap: 24px;
  align-items: center;
}

.welcome-card p.intro-text {
  font-size: 1.25rem;
  font-weight: 500;
  color: #6C4F8C;
  margin-bottom: 30px;
}

.why-setup {
  text-align: left;
  max-width: 480px;
  color: #5a4b9c;
  margin: 0 auto 40px;
}

.why-setup h2 {
  font-size: 1.8rem;
  font-weight: 700;
  margin-bottom: 16px;
  color: #6C4F8C;
}
```

```
.why-setup ul {
  list-style-type: disc;
  padding-left: 20px;
  font-size: 1.1rem;
  font-weight: 500;
  line-height: 1.5;
}

.why-setup ul li {
  margin-bottom: 10px;
}

.btn-primary.btn-xl {
  font-size: 1.5rem;
  padding: 20px 48px;
  border-radius: 12px;
  font-weight: 800;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.btn-primary.btn-xl:hover {
  background-color: #5040a0;
}

/* Complete profile section */
.complete-profile-content {
  display: flex;
```

```
  flex-direction: column;
  align-items: center;
  gap: 40px;
}

/* Featured image */
.feature-image {
  max-width: 320px;
  width: 100%;
  border-radius: 16px;
  box-shadow: 0 15px 35px rgba(107, 93, 211, 0.2);
}

/* Search form */
.search-form {
  display: flex;
  justify-content: center;
  gap: 12px;
  width: 100%;
  max-width: 600px;
}

.search-form input[type="text"] {
  flex-grow: 1;
  padding: 14px 18px;
  font-size: 1.1rem;
  border-radius: 12px;
  border: 2px solid #6C4F8C;
```

```
  font-weight: 500;
  color: #4a3e87;
  transition: border-color 0.3s ease;
}

.search-form input[type="text"]:focus {
  outline: none;
  border-color: #5040a0;
}

.search-form button {
  background-color: #6C4F8C;
  color: white;
  border: none;
  padding: 14px 32px;
  font-size: 1.1rem;
  font-weight: 700;
  border-radius: 12px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.search-form button:hover {
  background-color: #5040a0;
}

/* Recommended jobs */
.recommended-jobs {
```

```
  width: 100%;
  max-width: 700px;
  text-align: left;
  color: #3B328C;
}

.recommended-jobs h2 {
  font-size: 2rem;
  font-weight: 700;
  margin-bottom: 24px;
  color: #6C4F8C;
}

.job-card {
  background-color: #f2f0fa;
  border-radius: 15px;
  padding: 20px 30px;
  margin-bottom: 20px;
  display: flex;
  justify-content: space-between;
  align-items: center;
  gap: 25px;
}

.job-info h3 {
  color: #6C4F8C;
  font-weight: 700;
  margin-bottom: 8px;
```

```
  font-size: 1.1rem;
}

.job-info p {
  margin: 4px 0;
  font-size: 1rem;
  color: #5a4b9c;
}

.btn-outline {
  background-color: white;
  border: 2px solid #6C4F8C;
  color: #6C4F8C;
  padding: 10px 22px;
  border-radius: 8px;
  font-weight: 700;
  font-size: 1rem;
  cursor: pointer;
  transition: background-color 0.3s ease, color 0.3s ease;
}

.btn-outline:hover {
  background-color: #6C4F8C;
  color: white;
}

/* Profile setup page container */
.profile-setup-container {
```

```
  max-width: 650px;
  margin: 60px auto 80px;
  padding: 0 25px;
  color: #3B328C;
}

/* Profile header */
.profile-header {
  text-align: center;
  margin-bottom: 40px;
}

.profile-header h1 {
  font-size: 2.8rem;
  font-weight: 700;
  margin-bottom: 12px;
  color: #6C4F8C;
}

.profile-header p {
  font-size: 1.2rem;
  font-weight: 500;
  color: #6C4F8C;
}

/* Form styles */
.profile-form-section form {
  background-color: #f2f0fa;
```

```
  padding: 40px 30px;
  border-radius: 18px;
  box-shadow: 0 12px 30px rgba(107, 93, 211, 0.15);
}

.profile-form-section label {
  display: block;
  font-weight: 600;
  font-size: 1rem;
  color: #5a4b9c;
  margin-bottom: 8px;
}

.profile-form-section input[type="text"],
.profile-form-section select {
  width: 100%;
  padding: 14px 18px;
  margin-bottom: 25px;
  border: 2px solid #6C4F8C;
  border-radius: 12px;
  font-weight: 500;
  font-size: 1rem;
  color: #4a3e87;
  transition: border-color 0.3s ease;
}

.profile-form-section input::placeholder,
.profile-form-section select {
```

```
  color: #a8a0d2;
}

.profile-form-section input:focus,
.profile-form-section select:focus {
  outline: none;
  border-color: #5040a0;
}

/* Submit button */
.profile-form-section button.btn-primary {
  width: 100%;
  padding: 16px 0;
  font-size: 1.3rem;
  border-radius: 14px;
  font-weight: 700;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.profile-form-section button.btn-primary:hover {
  background-color: #5040a0;
}

.system-header {
  font-size: 2.8rem;
  font-weight: 700;
  color: #6C4F8C;
```

```
  margin-bottom: 40px;
  text-align: center;
}

/* Recommended jobs simple list */
.recommended-jobs {
  max-width: 600px;
  margin: 0 auto 40px;
  text-align: left;
  color: #3B328C;
}

.recommended-jobs h2 {
  font-size: 2rem;
  font-weight: 700;
  margin-bottom: 24px;
  color: #6C4F8C;
}

.job-list {
  list-style-type: disc;
  padding-left: 20px;
  font-size: 1.2rem;
  font-weight: 600;
  color: #5a4b9c;
}

.job-list li {
```

```
  margin-bottom: 10px;
}

/* Search form for job input */
.search-form {
  display: flex;
  justify-content: center;
  gap: 12px;
  max-width: 600px;
  margin: 0 auto;
}

.search-form input[type="text"] {
  flex-grow: 1;
  padding: 14px 18px;
  font-size: 1.1rem;
  border-radius: 12px;
  border: 2px solid #6C4F8C;
  font-weight: 500;
  color: #4a3e87;
  transition: border-color 0.3s ease;
}

.search-form input[type="text"]:focus {
  outline: none;
  border-color: #5040a0;
}
```

```
.search-form button {
  background-color: #6C4F8C;
  color: white;
  border: none;
  padding: 14px 32px;
  font-size: 1.1rem;
  font-weight: 700;
  border-radius: 12px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.search-form button:hover {
  background-color: #5040a0;
}

.job-card-simple {
  background-color: #f9f7ff;
  border-radius: 12px;
  padding: 20px 25px;
  margin-bottom: 16px;
  display: flex;
  justify-content: space-between;
  align-items: center;
  box-shadow: 0 8px 20px rgba(107, 93, 211, 0.1);
}

.job-main-info h3 {
```

```
  color: #6C4F8C;
  font-weight: 700;
  margin-bottom: 8px;
  font-size: 1.25rem;
}

.job-main-info p {
  color: #5a4b9c;
  margin: 4px 0;
  font-weight: 600;
}

.btn-outline {
  background-color: white;
  border: 2px solid #6C4F8C;
  color: #6C4F8C;
  padding: 10px 26px;
  border-radius: 10px;
  font-weight: 700;
  font-size: 1rem;
  cursor: pointer;
  transition: background-color 0.3s ease, color 0.3s ease;
}

.btn-outline:hover {
  background-color: #6C4F8C;
  color: white;
}
```

```
.pagination {
  display: flex;
  justify-content: center;
  gap: 12px;
  margin-top: 30px;
  flex-wrap: wrap;
}

.page-link {
  padding: 8px 14px;
  border-radius: 8px;
  border: 2px solid #6C4F8C;
  color: #6C4F8C;
  font-weight: 700;
  text-decoration: none;
  cursor: pointer;
  user-select: none;
  transition: background-color 0.3s ease, color 0.3s ease;
}

.page-link:hover {
  background-color: #6C4F8C;
  color: white;
}

.page-link.active {
  background-color: #6C4F8C;
```

```
  color: white;
  pointer-events: none;
}

.filter-panel {
  flex: 1;
  background-color: #f9f7ff;
  padding: 25px 20px;
  border-radius: 15px;
  box-shadow: 0 8px 20px rgba(107, 93, 211, 0.15);
  color: #6C4F8C;
  font-weight: 600;
  max-width: 300px;
}

.filter-panel h3 {
  margin-bottom: 25px;
  font-weight: 700;
  font-size: 1.5rem;
  color: #6C4F8C;
}

.filter-panel label {
  display: block;
  margin-bottom: 10px;
  font-weight: 600;
}
```

```
.filter-panel input[type="number"] {
  width: 100%;
  padding: 10px 14px;
  font-size: 1rem;
  border-radius: 10px;
  border: 2px solid #6C4F8C;
  margin-bottom: 25px;
  font-weight: 500;
  color: #4a3e87;
  transition: border-color 0.3s ease;
}

.filter-panel input[type="number"]:focus {
  outline: none;
  border-color: #5040a0;
}

.filter-panel button.btn-primary.btn-sm {
  width: 100%;
  padding: 12px 0;
  font-size: 1.1rem;
  border-radius: 12px;
  font-weight: 700;
  cursor: pointer;
  background-color: #6C4F8C;
  color: white;
  border: none;
  transition: background-color 0.3s ease;
```

```
}

.filter-panel button.btn-primary.btn-sm:hover {
  background-color: #5040a0;
}
 .filter-panel select {
  width: 100%;
  padding: 10px 14px;
  font-size: 1rem;
  border-radius: 10px;
  border: 2px solid #6C4F8C;
  margin-bottom: 25px;
  font-weight: 500;
  color: #4a3e87;
  transition: border-color 0.3s ease;
  cursor: pointer;
}

.filter-panel select:focus {
  outline: none;
  border-color: #5040a0;
}

.content-wrapper {
  display: flex;
  gap: 40px;
  justify-content: space-between;
  flex-wrap: wrap;
```

```
}

.job-list-section {
  flex: 2 1 600px;
  min-width: 320px;
}

.job-card-detailed {
  background-color: #f9f7ff;
  border-radius: 15px;
  padding: 22px 28px;
  margin-bottom: 18px;
  display: flex;
  justify-content: space-between;
  align-items: flex-start;
  box-shadow: 0 8px 20px rgba(107, 93, 211, 0.15);
}

.job-info h3 {
  font-size: 1.8rem;
  font-weight: 800;
  color: #6C4F8C;
  margin-bottom: 10px;
  text-transform: uppercase;
}

.job-info p {
  font-weight: 600;
```

```
  color: #5a4b9c;
  margin: 6px 0;
}

.personality-req-list {
  list-style-type: disc;
  padding-left: 20px;
  margin-top: 6px;
}

.personality-req-list li {
  margin-bottom: 6px;
  font-weight: 500;
  color: #4a3e87;
}

.btn-outline {
  background-color: white;
  border: 2px solid #6C4F8C;
  color: #6C4F8C;
  padding: 10px 26px;
  border-radius: 10px;
  font-weight: 700;
  font-size: 1rem;
  cursor: pointer;
  transition: background-color 0.3s ease, color 0.3s ease;
  align-self: center;
  height: fit-content;
```

```
}

.btn-outline:hover {
  background-color: #6C4F8C;
  color: white;
}

/* Pagination */
.pagination {
  display: flex;
  justify-content: center;
  gap: 12px;
  margin-top: 30px;
  flex-wrap: wrap;
}

.page-link {
  padding: 8px 14px;
  border-radius: 8px;
  border: 2px solid #6C4F8C;
  color: #6C4F8C;
  font-weight: 700;
  text-decoration: none;
  cursor: pointer;
  user-select: none;
  transition: background-color 0.3s ease, color 0.3s ease;
}
```

```
.page-link:hover {
  background-color: #6C4F8C;
  color: white;
}

.page-link.active {
  background-color: #6C4F8C;
  color: white;
  pointer-events: none;
}

/* Filter panel */
.filter-panel {
  flex: 1;
  background-color: #f9f7ff;
  padding: 25px 20px;
  border-radius: 15px;
  box-shadow: 0 8px 20px rgba(107, 93, 211, 0.15);
  color: #6C4F8C;
  font-weight: 600;
  max-width: 300px;
}

.filter-panel h3 {
  margin-bottom: 25px;
  font-weight: 700;
  font-size: 1.5rem;
  color: #6C4F8C;
```

```
}

.filter-panel label {
  display: block;
  margin-bottom: 10px;
  font-weight: 600;
}

.filter-panel input[type="number"],
.filter-panel select {
  width: 100%;
  padding: 10px 14px;
  font-size: 1rem;
  border-radius: 10px;
  border: 2px solid #6C4F8C;
  margin-bottom: 25px;
  font-weight: 500;
  color: #4a3e87;
  transition: border-color 0.3s ease;
  cursor: pointer;
}

.filter-panel input[type="number"]:focus,
.filter-panel select:focus {
  outline: none;
  border-color: #5040a0;
}
```

```
.filter-panel button.btn-primary.btn-sm {
  width: 100%;
  padding: 12px 0;
  font-size: 1.1rem;
  border-radius: 12px;
  font-weight: 700;
  cursor: pointer;
  background-color: #6C4F8C;
  color: white;
  border: none;
  transition: background-color 0.3s ease;
}

.filter-panel button.btn-primary.btn-sm:hover {
  background-color: #5040a0;
}

/* Content wrapper */
.content-wrapper {
  display: flex;
  gap: 40px;
  justify-content: space-between;
  flex-wrap: wrap;
}

.job-list-section {
  flex: 2 1 600px;
  min-width: 320px;
```

```
}

@media (max-width: 960px) {
  .content-wrapper {
    flex-direction: column;
  }
  .filter-panel {
    max-width: 100%;
    margin-top: 30px;
  }
  .job-list-section {
    width: 100%;
  }
}

@media (max-width: 960px) {
  .content-wrapper {
    flex-direction: column;
  }
  .filter-panel {
    max-width: 100%;
    margin-top: 30px;
  }
  .job-list-section {
    width: 100%;
  }
}
```

```
/* Responsive */
@media (max-width: 480px) {
  .profile-setup-container {
    padding: 0 15px;
    margin: 40px auto 60px;
  }
  .profile-header h1 {
    font-size: 2.2rem;
  }
  .profile-header p {
    font-size: 1rem;
  }
  .profile-form-section form {
    padding: 30px 20px;
  }
  .profile-form-section button.btn-primary {
    font-size: 1.1rem;
  }
}

/* Responsive */
@media (max-width: 768px) {
  .search-form {
    flex-direction: column;
    gap: 18px;
  }
  .search-form button {
    width: 100%;
```

```
  }
  .job-card {
    flex-direction: column;
    align-items: flex-start;
  }
  .job-card button {
    margin-top: 12px;
    width: 100%;
  }
}

@media (max-width: 480px) {
  .dashboard-container {
    margin: 40px 15px 40px;
  }
  .welcome-title {
    font-size: 2rem;
  }
  .recommended-jobs h2 {
    font-size: 1.5rem;
  }
}

/* Responsive */
@media (max-width: 480px) {
    .container {
        padding: 30px 20px;
        margin: 40px 10px;
```

```
    }
    .container h2 {
        font-size: 24px;
    }
    .container button.login-btn,
    .container button.register-btn,
    .container button.google-btn {
        font-size: 16px;
        padding: 12px;
    }
}
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
      <meta name="viewport" content="width=device-width, initial-
scale=1" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Career Navigator</title>
          <link rel="stylesheet" href="{{ url_for('static',
filename='css/styles.css') }}" />
                                                              <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&
display=swap" rel="stylesheet" />
</head>
<body>
    <header>
```

```
        <nav>
            <div class="logo">
                                    <img   src="{{   url_for('static',
filename='images/logo1.png') }}" alt="Career Navigator Logo" />
            </div>
            <ul>
                <li><a href="{{ url_for('home') }}">Home</a></li>
                <li><a href="#">About Us</a></li>
                                    <li><a   href="{{   url_for('login')
}}">Login/Register</a></li>
            </ul>
        </nav>
    </header>

    <section class="hero">
        <h1>Coding Your Career Path</h1>
          <p>Explore personalized job recommendations tailored for
Computer Science and Information Technology graduates</p>
        <button onclick="window.location.href='{{ url_for('register')
}}'">Get Started</button>
    </section>

    <section class="about">
        <div class="vision">
            <h2>Vision</h2>
            <p>To be the leading platform for career guidance and job
matching for tech graduates worldwide.</p>
        </div>
        <div class="mission">
            <h2>Mission</h2>
```

```
            <p>Our mission is to empower fresh graduates in CS and IT by connecting them with personalized job opportunities and helping them navigate their career paths.</p>

        </div>

        <div class="audience">

            <h2>Audience</h2>

             <p>Designed exclusively for CS and IT graduates entering the workforce.</p>

        </div>

    </section>


    <footer>

        <p>&copy; 2025 Career Navigator. All Rights Reserved.</p>

    </footer>

</body>

</html>
```

job_recommendations.html

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

  <title>Job Recommendations - Career Navigator</title>

      <link     rel="stylesheet"     href="{{     url_for('static', filename='css/styles.css') }}" />

                                                                  <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap" rel="stylesheet" />

</head>

<body>
```

```
<header>
  <nav>
    <div class="logo">
         <img  src="{{  url_for('static',  filename='logo.png')  }}"
alt="Career Navigator Logo" />
    </div>
    <ul>
      <li><a href="{{ url_for('dashboard') }}">Home</a></li>
      <li><a href="{{ url_for('profile') }}">Profile</a></li>
      <li><a href="{{ url_for('logout') }}">Logout</a></li>
    </ul>
  </nav>
</header>

<main class="job-recommendations-container">
  <h2 class="page-title">
    Here are the job recommendations for you ({{ search_term or 'your
interests' }})
  </h2>

  <div class="content-wrapper">
    <!-- Job list -->
    <section class="job-list-section">
      {% if jobs %}
      <ul class="job-list">
        {% for entry in jobs %}
        <li class="job-card-detailed">
          <div class="job-info">
```

```
            <h3>{{ entry.job.job_title.upper() }}</h3>
            <p><strong>Location:</strong> {{ entry.job.location }}</p>
            <p><strong>Salary:</strong> {{ entry.job.salary or 'N/A' }}</p>
            <p><strong>Requirements:</strong></p>
            <ul class="personality-req-list">
              {% for req in entry.personality_requirements %}
              <li>{{ req }}</li>
              {% endfor %}
            </ul>
          </div>
                        <button onclick="window.location.href='{{ url_for('job_details', job_id=entry.job.job_id) }}'" class="btn-outline">View Job</button>
        </li>
        {% endfor %}
      </ul>

      <!-- Pagination -->
      <nav class="pagination">
        {% if page > 1 %}
          <a href="{{ url_for('job_search', q=search_term, page=page-1, threshold=threshold, location=location, job_type=job_type) }}" class="page-link">Prev</a>
        {% endif %}
        {% for p in range(1, total_pages + 1) %}
            <a href="{{ url_for('job_search', q=search_term, page=p, threshold=threshold, location=location, job_type=job_type) }}"
             class="page-link {% if p == page %}active{% endif %}">{{ p }}</a>
```

```
        {% endfor %}

        {% if page < total_pages %}

                <a  href="{{  url_for('job_search',  q=search_term,
page=page+1,          threshold=threshold,          location=location,
job_type=job_type) }}" class="page-link">Next</a>

        {% endif %}

      </nav>

      {% else %}

      <p>No suitable jobs found matching your profile.</p>

      {% endif %}

    </section>


    <!-- Filter panel -->

    <aside class="filter-panel">

      <h3>Adjust Filters</h3>

      <form method="GET" action="{{ url_for('job_search') }}">

        <input type="hidden" name="q" value="{{ search_term }}">


        <label for="threshold">Threshold:</label>

       <input type="number" id="threshold" name="threshold" value="{{
threshold or 0 }}" min="0" max="100" step="1" />


        <label for="location">Location:</label>

        <select id="location" name="location">

         <option value="" {% if not location %}selected{% endif %}>All
Locations</option>

          {% for loc in locations %}

         <option value="{{ loc }}" {% if location == loc %}selected{%
endif %}>{{ loc }}</option>

          {% endfor %}
```

```
        </select>


        <label for="job_type">Job Type:</label>

        <select id="job_type" name="job_type">

         <option value="" {% if not job_type %}selected{% endif %}>All
Types</option>

          {% for jt in job_types %}

          <option value="{{ jt }}" {% if job_type == jt %}selected{%
endif %}>{{ jt }}</option>

          {% endfor %}

        </select>


                  <button   type="submit"   class="btn-primary   btn-
sm">Apply</button>

      </form>

    </aside>

  </div>

</main>

</body>

</html>
```