B222428B

BINDURA UNIVERSITY OF SCIENCE EDUCATION

FACULTY OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

AUTOMATED RESUME ANALYSIS AND JOB ADVERT MATCHING USING NATURAL LANGUAGE PROCESSING

BY

Malvern Sixpence M

Reg number: B222428B

Supervisor: Mr H Chikwiriro

June 2026

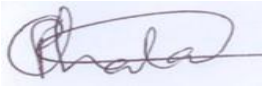
A project is submitted to Bindura University of Science Education in partial fulfilment of the requirements of the Bachelor of science Honors degree in Computer science

B222428B

Approval Form

The undersigned certify that they have supervised the student **Malvern Sixpence M** dissertation entitled **Automated Resume Analysis and Job Advert Matching Using Natural Language Processing**

submitted in partial fulfilment of the requirements for the Bachelor of **Computer Science** at Bindura University of Science Education.

Malvern M.Sixpence		16/08/2025
STUDENT	**SIGNATURE**	**DATE**
Mr. H. Chikwiriro		20/08/2025
SUPERVISOR	**SIGNATURE**	**DATE**
Mr. P. Chaka		20/08/2025
CHAIRPERSON	**SIGNATURE**	**DATE**

B222428B

DECLARATION

I, Malvern Sixpence M, declare that this dissertation entitled "An Automated Resume Analysis and Job Advert Matching System using NLP" is my own independent work and has not been submitted, either in whole or in part, for any other degree or professional qualification at this or any other institution.

I acknowledge that Bindura University of Science Education may, for academic and research purposes, retain a copy of this dissertation in the university library and make it available for consultation, lending, or reproduction, if acknowledgement is made of its authorship.

ACKNOWLEDGEMENTS

First and foremost, I give all glory and honour to the Almighty God for the wisdom, strength, and perseverance He bestowed upon me throughout this academic journey. Without His grace, the completion of this work would not have been possible.

I wish to express my sincere and profound gratitude to my supervisor, Mr H. Chikwiriro, for his unwavering guidance, patience, and intellectual counsel throughout the research process. His constructive feedback and dedication to academic excellence were instrumental in shaping this dissertation into the work it has become. I have grown not only as a researcher but as a professional through his mentorship.

I would also like to extend my appreciation to Mr P. Chaka, the Chairperson of the Department of Computer Science, and to all lecturers and members of staff at Bindura University of Science Education who contributed to my academic formation over the course of my studies.

Special thanks are due to the management of Bindura University of Science Education for providing an environment conducive to learning, and to the participants, recruiters, and job seekers who generously gave their time to support the data collection phase of this research. Your insights were invaluable.

I am equally grateful to my colleagues and peers whose encouragement, shared knowledge, and camaraderie sustained me during the most demanding periods of this project. The spirit of collaboration and mutual support that exists among us will remain one of the most cherished lessons of this journey.

Finally, I would like to acknowledge every individual who believed in me and offered support, whether through prayer, advice, or simply a word of encouragement, at any stage of this work. Your kindness did not go unnoticed.

DEDICATION

This dissertation is dedicated with love and gratitude to my family, whose unwavering support, sacrifice, and belief in my potential have been the foundation upon which every achievement in my life is built.

To my parents, thank you for instilling in me the value of education and the discipline to pursue it wholeheartedly. Every late night and early morning that went into this work was carried by the strength you gave me.

ABSTRACT

The rapid growth of digital recruitment platforms has transformed the employment process, yet many job seekers continue to face challenges in aligning their resumes with specific job requirements. Traditional recruitment methods are often time-consuming, inefficient, and prone to human bias, resulting in poor candidate-job matching. This research presents the development of an Automated Resume Analysis and Job Advert Matching System using Artificial Intelligence and Natural Language Processing (NLP) techniques to improve recruitment efficiency and accuracy.

The system analyzes resumes and job advertisements by extracting relevant skills, qualifications, experience, and keywords, then compares them to determine compatibility between candidates and available job opportunities. Machine learning and NLP algorithms are utilized to enhance semantic understanding and automate the matching process. The proposed system aims to assist recruiters in identifying suitable candidates more efficiently while helping job seekers optimize their resumes according to market demands.

The study evaluates the effectiveness of the system based on matching accuracy, processing speed, and usability. Findings indicate that the AI-driven approach significantly improves candidate-job matching, reduces recruitment time, and enhances decision-making within the hiring process. The research contributes toward the adoption of intelligent recruitment technologies in Zimbabwe and demonstrates the potential of AI-based systems in modern human resource management.

Table of Contents

DECLARATION 3
ACKNOWLEDGEMENTS 4
DEDICATION 5
ABSTRACT 6
CHAPTER 1 9
1.1 Introduction 9
1.2 Background of study 10
1.3 Problem Statement 11
1.4 Research Objectives 11
1.5 Research Questions 12
1.6 Research Propositions/Hypothesis 12
1.7 Justification/Significance of the Study 12
1.8 Assumptions 12
1.9 Limitations/Challenges 13
1.10 Scope/Delimitations of the Research 13
1.11 Definition of Terms 13
CHAPTER 2 15
2.1 Introduction 15
2.2 Limitations of Traditional Resume Matching 16
2.3 Evolution of Resume Matching Techniques 16
2.3.1 Keyword-Based Approaches 16
2.3.2 Rule-Based Systems 16
2.3.3 Statistical Methods (TF-IDF and Topic Modelling) 17
2.4 Natural Language Processing and Machine Learning in Recruitment 17
2.5 Empirical Studies on NLP-Based Resume Matching 17
2.6 Existing Resume Matching Systems 18
2.7 Advantages of NLP-Based Systems 18
2.8 Challenges and Limitations 19
2.9 Research Gap 19
2.10 Conclusion 19
CHAPTER 3 RESEARCH METHODOLOGY 21
3.1 Introduction 21
3.2 Population and Sample 21
3.2.1 Sample 21
3.3 Data Collection Approaches 21

3.3.1 Interviews 22
3.3.2 Observations 22
3.3.3 Questionnaires 22
3.4 Research Design 22
3.4.1 Requirement Analysis 22
3.4.2 Functional Requirements 22
3.4.3 Non-Functional Requirements 23
3.4.4 Hardware Requirements 23
3.4.5 Software Requirements 23
3.5 System Development Tools 23
3.6 System Design 24
3.6.1 Data Flow Diagram 24
3.6.2 System Flowchart 26
3.6.3 Use Case Diagram 27
3.6.4 Class Diagram and System Relationships 29
3.7 Development Model 31
3.8 Feature Engineering and NLP Pipeline 33
3.9 Interface Designs 35
3.9.1 Menu Design 35
3.9.2 Main menu 35
3.10 Input Design 35
3.10.1 Signup and Login Forms 36
CHAPTER 4: DATA PRESENTATION, ANALYSIS AND INTERPRETATION 38
4.1 Introduction 38
4.2 Testing 38
4.2.1 Unit Testing 38
4.2.2 Integration Testing 41
4.2.3 Black-box Testing 41
4.2.4 White-box Testing 44
4.3 Evaluation Measures and Results 48
4.4 Summary of Research Findings 52
4.5 Conclusion 52
CHAPTER 5: CONCLUSION AND RECOMMENDATIONS 54
5.1 Introduction 54
5.2 Aims and Objectives Realization 54
5.3 Recommendations 54

5.4 Conclusion 54
5.5 Future Work 55
5.6 Challenges faced 55
REFERENCES 55

CHAPTER 1

1.1 Introduction

Getting noticed in today's job market takes more than just being qualified you must present your qualifications in the right way, and that is something a lot of candidates struggle with. In Zimbabwe, where hiring has traditionally been handled through conventional methods, this problem is even more pronounced. Resumes get submitted, and opportunities get missed, not

necessarily because the person was not right for the role, but because nothing ever bridged the gap between what they offered and what the employer was looking for.

This research proposes an Automated Resume Analysis and Job Advert Matching System that harnesses Natural Language Processing (NLP) to address precisely this shortcoming. Rather than requiring candidates to manually assess whether their resume aligns with a given job description, the system performs this evaluation automatically reading both documents and measuring how well they align. NLP is what makes this meaningful: it understands language the way a person would, picking up on context and meaning rather than just matching words on a list.

This chapter looks at what AI brings to resume analysis and job advert matching, the technologies involved, the problems they help solve, and why this kind of tool matters for job seekers in Zimbabwe. It also makes the case for why this space needs continued innovation, and what better outcomes could look like for candidates trying to navigate a competitive labor market.

1.2 Background of study

AI is changing how hiring works, and it is doing so in ways that are hard to ignore. Systems can now read a resume and a job description and tell you, with reasonable accuracy, whether they are a good match identifying gaps the candidate might not even be aware of. That kind of capability, built on NLP, is what this project sets out to bring into the Zimbabwean recruitment context.

Industry forecasts highlight just how rapidly AI is reshaping recruitment. According to Market Research Future (2025), the global market for AI-powered hiring tools is projected to reach around USD 1,289.13 million by 2035, growing at an annual rate of 6.92% from 2025. Notably, 83% of organisations intend to incorporate AI into their candidate screening workflows by 2025, reflecting how quickly the practice is becoming mainstream.

to incorporate AI into their resume screening processes by 2025 a signal of just how fast this is becoming standard practice.

In practice, AI-assisted screening can reduce the time spent reviewing applications by as much as 75%, and by evaluating a broader range of data points, it tends to produce more consistent results than human review alone. That said, these tools carry real risks. Research by Wilson and Caliskan (2024) found that some AI screening systems inadvertently disadvantage women and certain racial groups, a reminder that the technology needs careful oversight to ensure it serves everyone fairly, not just those who already have an advantage.

1.3 Problem Statement

Many job seekers in Zimbabwe have the qualifications but not the outcomes they apply, they wait, and they hear nothing. Part of the reason is that most candidates do not have a clear picture of what employers are looking for, so even a strong resume can miss the mark. Traditional submission processes offer no feedback, no guidance, and no way to
courses correct.

The labor market is not standing still either. Employers are raising the bar on specific, technical competencies, but job seekers are often the last to know exactly what those are. There is a disconnect between what candidates think they need and what the market is demanding and it is getting wider.

That is the gap this system is designed to close. By automatically analyzing a resume against available job adverts and flagging where the candidate falls short, an NLP-powered matching system turns a frustrating guessing game into something actionable. Candidates get real feedback, in real time, on what they need to work on and that changes the odds.

1.4 Research Objectives

1. To analyze various methods and technologies utilized in automated resume analysis and job advert matching systems, focusing on the incorporation of Natural Language Processing (NLP) alongside machine learning methodologies.
2. To develop and deploy an Automated Resume Analysis and Job Advert Matching System capable of accurately pairing resumes with relevant job advertisements while detecting skills deficiencies among candidates in real time.
3. To evaluate the effectiveness of the techniques implemented within the proposed system, including but not restricted to the use of Natural Language Processing (NLP) for processing resume content, machine learning algorithms for aligning resumes with job postings, and relevance ranking mechanisms designed to ensure that pertinent opportunities are surfaced accurately without excluding other viable positions

1.5 Research Questions

1. What approaches and technologies are best suited to improving the performance of automated resume analysis and job advert matching using NLP?
2. What are the essential steps involved in designing and implementing an Automated Resume Analysis and Job Advert Matching System that leverages NLP and machine learning?
3. How can the impact of the proposed system on job seekers' employability and readiness be measured and evaluated?

1.6 Research Propositions/Hypothesis

1. Incorporating AI technologies and Natural Language Processing (NLP) within automated resume analysis and job advert matching is expected to substantially improve the probability of candidates securing interview opportunities, outperforming outcomes typically achieved through conventional application methods.
2. An AI-driven Resume Analysis and Job Advert Matching System will enhance the employability and overall satisfaction of job seekers by delivering practical, specific guidance regarding the competencies and credentials they need to develop.
3. The integration of machine learning algorithms will optimize the job advert matching process, resulting in higher success rates for job seekers in finding relevant opportunities.

1.7 Justification/Significance of the Study

The innovative application of AI and NLP in automated resume analysis and job advert matching has the potential to significantly enhance the job search experience for Zimbabwean job seekers. Many candidates currently rely on traditional methods that are often time-consuming and ineffective, leading to missed opportunities and frustration. This research aims to propose a more efficient framework that leverages AI technologies to empower job seekers in their pursuit of employment.

By implementing an AI-based Resume Analysis and Job Advert Matching System, job seekers can receive tailored feedback on their resumes, helping them understand the skills they need to develop to meet employer expectations. Such an approach stands to benefit not only the individual applicant but also the broader labour market by fostering a more highly skilled and competitive workforce within Zimbabwe.

Furthermore, the knowledge generated through this study can shape guidance frameworks for job seekers, equipping them to engage more strategically with the job market. Ultimately, this work aspires to advance the development of job search technologies, raising employability outcomes for candidates and promoting a more equitable and accessible labour market.

1.8 Assumptions

Technological Readiness:
It is assumed that the necessary infrastructure for implementing automated resume analysis and job

advert matching using AI and Natural Language Processing (NLP) is available or can be established within the targeted environments for Zimbabwean job seekers.

User Acceptance:
The research assumes that job seekers will be willing to adopt and utilize the Automated Resume Analysis and Job Advert Matching System, recognizing its benefits for enhancing their job search experience.

Data Availability:
It is assumed that sufficient and relevant data for training the AI models will be accessible, ensuring the effectiveness of the Automated Resume Analysis and Job Advert Matching System.

1.9 Limitations/Challenges

Cost of Implementation:
A significant challenge is the financial investment required for developing and deploying the AI and NLP technologies, which may pose barriers to smaller organizations or individual job seekers.

Data Privacy Concerns:
Questions around the confidentiality and protection of personal information gathered in the course of automated resume analysis represent a genuine concern, with potential implications for whether users choose to engage with the system.

Technical Limitations:
The effectiveness of the Automated Resume Analysis and Job Advert Matching System can be influenced by the quality of the data used for training and the ability of the models to generalize across diverse job descriptions in the Zimbabwean context.

1.10 Scope/Delimitations of the Research

The study is geographically bounded to urban settings within Zimbabwe, where the challenges and dynamics of job seeking differ meaningfully from rural environments.

Technological Focus:
The study will concentrate exclusively on automated resume analysis and job advert matching utilizing AI and NLP, excluding traditional candidate screening methods such as manual reviews or basic keyword matching systems.

Service Areas:
The research will primarily explore the implications of the Automated Resume Analysis and Job Advert Matching System for job seekers, emphasizing their employability rather than the needs of employers or recruitment agencies.

1.11 Definition of Terms

Automated Resume Analysis and Job Advert Matching System:
A technology-driven tool that utilizes AI and NLP to analyze resumes and match them with job descriptions, providing feedback on skills and qualifications specifically for job seekers.

NLP (Natural Language Processing):
A specialised domain within artificial intelligence that examines how computers process and engage with human language, allowing systems to derive meaning from and respond to written or spoken text.

Machine Learning:
A branch of artificial intelligence in which systems draw on statistical models and algorithmic processes to carry out tasks autonomously, refining their performance by learning from patterns in data.

CHAPTER 2

2.1 Introduction

Digital technologies have become deeply embedded in hiring processes, fundamentally altering how employers identify and evaluate candidates. Despite these advances, many job seekers still struggle to tailor their applications effectively to what employers actually require. A significant weakness in many current platforms is the lack of constructive, structured feedback, leaving candidates without clear guidance on how to strengthen future applications.

Recent advances in AI and NLP have opened more capable ways of working with unstructured documents like resumes and job descriptions. These tools can pull out relevant details, understand context, and measure how similar two pieces of text are, which translates into more accurate matching between candidates and roles (Saatçı, Kaya and Ünlü, 2024).

This chapter undertakes a critical examination of published research on resume analysis and job matching, tracing the progression from earlier manual screening methods to the more sophisticated NLP-driven approaches prevalent today. It also identifies the shortcomings in current work and articulates the rationale for developing a system whose primary focus is the benefit of the job seeker.

2.2 Limitations of Traditional Resume Matching

Traditional resume screening has always been a manual, time-intensive exercise, one that asks recruiters to work through large stacks of applications often under significant time pressure. Beyond the sheer effort involved, the process is inconsistent: what catches one person's eye may not catch another's and personal bias can quietly shape decisions in ways that have nothing to do with a candidate's actual suitability (Hewage, 2023).

Applicant Tracking Systems brought in keyword filtering to ease some of this pressure but introduced a different kind of problem. These systems look for exact word matches, which means candidates who describe the same skill using different terminology can be screened out before a human ever reads their application even if they are perfectly qualified for the role (Modak et al., 2024).

What is more, traditional systems almost never tell candidates why they were not selected or how they could strengthen their applications. This lack of feedback keeps job seekers in the dark and perpetuates a cycle of unsuccessful applications, which is exactly why more intelligent, context-aware systems are needed.

2.3 Evolution of Resume Matching Techniques

2.3.1 Keyword-Based Approaches

Keyword-based approaches operate by scanning for identical terminology across resumes and job descriptions. While computationally efficient and straightforward to implement, they are fundamentally limited by their inability to recognise that professionals often express equivalent competencies in varying ways through synonyms, abbreviations, or field-specific jargon which means suitably qualified candidates can be inadvertently excluded from consideration (Modak et al., 2024).

2.3.2 Rule-Based Systems

Systems governed by predefined rules advanced the field by introducing greater consistency through structured evaluation criteria. However, their inherent inflexibility proved problematic: as occupational language evolved, the underlying rule sets required continuous revision, resulting in systems that were both costly and difficult to sustain over time (Bhukya, 2024).

2.3.3 Statistical Methods (TF-IDF and Topic Modelling)

Statistical techniques like TF-IDF brought more nuance to the process by weighing terms based on how significant they are within a given document. This helped systems home in on the most relevant keywords more reliably (Rathi et al., 2024).

The limitation of TF-IDF, though, is that it treats words as independent units it cannot pick up on the relationships between them or understand meaning from context. Topic modelling was later introduced to address this, grouping words into broader themes to better capture what a document is about (Jagwani et al., 2023).

2.4 Natural Language Processing and Machine Learning in Recruitment

At the core of any contemporary approach to interpreting recruitment documents lies Natural Language Processing. By applying methods such as tokenization, text normalisation, and named entity recognition, NLP systems are capable of parsing a resume or job description and extracting the most relevant information skills, roles, and qualifications in a structured manner (Saatçı, Kaya and Ünlü, 2024).

Machine learning builds on that foundation by allowing systems to get better over time. Rather than following fixed rules, ML models learn from examples which means they can be trained to classify resumes, estimate how well a candidate fits a role, and rank job matches in order of relevance (Dadaboyev et al., 2025).

More recently, deep learning approaches particularly word and sentence embeddings, and large language models have pushed the boundaries further. These models capture how words relate to one another in context, which makes the matching between resumes and job descriptions considerably more accurate and semantically meaningful (Bevara et al., 2025; Vaishampayan et al., 2025).

2.5 Empirical Studies on NLP-Based Resume Matching

The evidence from empirical research is encouraging. Studies show that NLP-based systems measurably improve both the speed and accuracy of recruitment decisions. Saatçı, Kaya and Ünlü (2024), for instance, found that

automated screening not only reduces the influence of bias but also leads to more thorough candidate evaluation.

Alderham and Jaha (2024) took a similar approach, proposing a semantic matching method that improves how candidates are ranked by focusing on the contextual connections between resume content and job requirements not just whether the same words appear.

Vector-based techniques like Resume2Vec go even further, representing both resumes and job descriptions as points in a shared mathematical space so that the closeness of a match can be calculated directly (Bevara et al., 2025). Work using large language models has also demonstrated notable gains in understanding context and producing more accurate matches (Vaishampayan et al., 2025).

That said, much of this research has been conducted using carefully curated datasets that may not reflect the messiness of real-world applications. How well these systems perform when confronted with diverse, inconsistently formatted resumes particularly in contexts like Zimbabwe is a question that still needs more attention.

2.6 Existing Resume Matching Systems

Modern recruitment platforms are increasingly built on a combination of NLP and machine learning, allowing them to automate shortlisting and produce ranked candidate recommendations. Such systems can handle both structured inputs — such as job titles and years of service — and unstructured content, including personal statements and competency descriptions.

The trouble is that most of these systems are built with the recruiter in mind. They do an efficient job of filtering candidates but offer very little back to the applicants themselves. Job seekers get no insight into why they were ranked the way they were, or what they could do to improve their profile for next time.

2.7 Advantages of NLP-Based Systems

Compared to older approaches, NLP-based systems have a lot going for them. They can understand the intended meaning of text rather than just scanning specific words, which makes matches more accurate. They also

process applications much faster and can scale to handle large volumes without compromising quality.

Crucially, these systems can also be designed to give candidates something back identifying gaps in their skills and suggesting where to focus their development. That kind of feedback loop has real potential to improve employability over time and make job seeking a more productive experience.

2.8 Challenges and Limitations

These systems do, however, come with their own set of challenges. One of the most persistent is data quality. Resumes come in all shapes and formats, and that inconsistency can make it harder for a system to extract information accurately and compare candidates on a level playing field.

Algorithmic bias constitutes a further serious risk. When models are developed using historical recruitment data that encodes prior discriminatory patterns, they risk reproducing those same inequities in their outputs. Fairness and transparency in AI-driven hiring tools are therefore not peripheral concerns but foundational requirements (Dadaboyev et al., 2025).

There is also the practical question of computing resources. The most powerful NLP models are computationally demanding, which can make them difficult to deploy in environments where hardware and infrastructure are limited.

2.9 Research Gap

Despite the progress that has been made, there is a recurring pattern in the literature: most systems are designed to serve the needs of employers and recruiters, not the candidates themselves.

Very few existing systems offer job seekers meaningful, personalized feedback in real time. This research responds directly to that gap by proposing a system that not only compares resumes to job adverts semantically but also gives candidates actionable insights concrete steps they can take to better align their profiles with the roles they are applying for.

2.10 Conclusion

This chapter has traced the development of resume matching from basic keyword filtering through to the sophisticated NLP-driven approaches that define the current state of the field. While modern systems represent a genuine improvement in accuracy and efficiency, they continue to struggle with issues like bias, variable data quality, and a tendency to focus on the recruiter's needs at the expense of the candidates.

The gap identified here is the absence of real-time, candidate-centered feedback provides the clearest justification for the system proposed in this research: one built around improving the job seeker's experience and employability, not just streamlining the hiring process for employers.

CHAPTER 3 RESEARCH METHODOLOGY

3.1 Introduction

This chapter describes the research methodology employed in the design, construction, and evaluation of the Automated Resume Analysis and Job Advert Matching System, with Natural Language Processing serving as the foundational technology.

The central aim of this study is to conceive, construct, and assess a system that enables job seekers to locate job advertisements well-suited to their individual qualifications, competencies, and professional backgrounds. In many job search environments, candidates are required to manually browse through large numbers of job advertisements to determine whether they are suitable applicants for a particular position. This process is not only inefficient and time-consuming, but it frequently results in mismatched applications that fail to align with employer expectations.

The system applies a sequence of NLP techniques including tokenization, removal of stop words, lemmatization, and TF-IDF vectorization to quantify the degree to which a candidate's profile corresponds to the requirements detailed in each job advertisement. Results are returned in ranked order, and the system concurrently highlights competency gaps the applicant may need to address.

3.2 Population and Sample

The research population encompasses job seekers currently pursuing employment in Zimbabwe, together with the corpus of job advertisements utilised in developing and testing the proposed system. Centring the research on job seekers is essential given that the system is purposefully designed to tackle the obstacles they encounter throughout the application process.

3.2.1 Sample

Given practical limitations including time constraints and restricted data access, a purposive sampling approach was adopted. The sample used in this study consisted of the following:

- A collection of sample resumes representing job seekers from various professional fields, used to test the system's resume parsing and skill extraction capabilities.
- A dataset of job advertisements sourced from publicly accessible Kaggle job posting datasets, specifically the Job Description Dataset (Rana, 2023) and the Dice.com Job Postings Dataset (PromptCloudHQ, 2021), which provided realistic job advertisement content for matching evaluation.
- Participants who were interviewed and surveyed to gather insights into the challenges experienced during the job search process.

3.3 Data Collection Approaches

Data collection was conducted to understand the challenges faced by job seekers when searching for suitable employment opportunities in Zimbabwe. It also helped identify the system requirements necessary for building an effective automated job advert matching platform. The researcher employed a variety of techniques to gather data, as described below.

3.3.1 Interviews

Semi-structured interviews were carried out with a selection of job seekers and human resources professionals to gather nuanced perspectives on the contemporary job search experience. Participants were asked questions related to how they search for job opportunities, the challenges they face when identifying suitable job advertisements, the amount of time typically spent reviewing vacancies, and the effectiveness of existing job search platforms. The responses obtained from the interviews informed the functional requirements of the proposed system.

3.3.2 Observations

To analyze the interaction between job seekers and job search platforms, direct observation was done. Candidates tended to browse and compare their qualifications with job postings. This method was inefficient, and many candidates ended up applying for jobs they were not qualified for. These observations indicated the need for an automated system that would be capable of scanning job postings and analyzing a candidate's resume to identify which postings a candidate was most likely to qualify for.

3.3.3 Questionnaires

A structured questionnaire was distributed to a targeted group of job seekers to collect quantitative data on their experiences with the job search process. The instrument addressed themes including application frequency, time spent reviewing vacancies, perceived usefulness of existing platforms, and awareness of personal skills gaps. Responses were analysed to complement the findings from interviews and direct observation and together informed the prioritisation of system features.

3.4 Research Design

Research design provides the overarching framework that shapes the investigation, offering a coherent plan through which the study of core questions can be systematically addressed. In this project, the researcher utilised Python alongside machine learning and NLP libraries to build the Automated Resume Analysis and Job Advert Matching System. Relevant documentation on resume processing and machine learning algorithms was consulted throughout to inform and refine the chosen methodology.

3.4.1 Requirement Analysis

A foundational task in this phase involved pinpointing the most informative variables within résumés and job adverts for evaluating candidate suitability, alongside the systematic documentation of both functional and non-functional system requirements. Potential obstacles notably the availability of suitable training datasets and the computational demands of the system were also identified as factors that could affect the viability of the proposed Automated Resume Analysis and Job Advert Matching methodology.

.

3.4.2 Functional Requirements

Functional requirements define the specific operations that the system must perform in order to meet the needs of its users. The following capabilities were specified for the proposed system:

- Allow job seekers to register and authenticate securely using a username and password.

- Enable job seekers to upload their resumes in PDF, DOCX, or TXT format for analysis.
- Allow the system administrator to post, manage, and delete job advertisements.
- Extract textual information from uploaded resumes using document parsing techniques.
- Process resumes and job advertisement text using NLP techniques including lemmatization , tokenization and stop-word removal.
- Identify key information such as skills, qualifications, and experience from the resume text.
- Calculate a weighted similarity score between each resume and all available job advertisements.
- Rank job advertisements according to how closely they match the job seeker's qualifications.
- Display a skills gap analysis identifying skills that the job seeker lacks for a given position.
- Provide learning resource links to help job seekers address identified skills gaps.
- Allow the system administrator to manage user accounts, including adding, deactivating, and resetting user passwords.
- Support integration of job advertisement datasets from external sources such as Kaggle.

3.4.3 Non-Functional Requirements

Non-functional requirements establish the quality attributes and performance expectations that shape how the system operates, rather than what it does. The principal non-functional requirements are as follows:

- Usability: The system shall provide a simple and user-friendly interface that enables job seekers to easily upload resumes, view matching scores, and access job recommendations without requiring technical expertise.
- Security: The system shall protect sensitive personal information contained in resumes using password hashing and role-based access control to prevent unauthorised access.
- Reliability: The system shall operate consistently and produce reproducible matching results for the same resume and job advertisement inputs.

3.4.4 Hardware Requirements

- A 64-bit processor (Intel Core i3 or equivalent and above)
- Minimum 4 GB RAM
- At least 10 GB available disk space
- Stable internet connection for dataset access and system updates

3.4.5 Software Requirements

- Windows 10 / 11
- Python 3.10 or higher
- Flask web framework
- NLTK and scikit-learn NLP libraries
- PyPDF2 and python-docx for document parsing
- SQLite database for data storage
- Visual Studio Code as the IDE

3.5 System Development Tools

The following tools and technologies were selected to implement the Automated Resume Analysis and Job Advert Matching System:

Python 3.13.13 - Python was selected as the principal development language on account of its rich library ecosystem supporting Natural Language Processing and machine learning. As noted by the Python Software Foundation (2021), Python is a high-level interpreted language whose combination of readability and extensive built-in functionality makes it particularly appropriate for rapid development in data-intensive applications.

Flask – Flask is a minimalist Python web framework employed to construct the system's web-facing interface. It was selected for its ease of configuration, architectural flexibility, and demonstrated suitability for projects of comparable scale and complexity.

NLTK (Natural Language Toolkit) - NLTK provides the core NLP capabilities used in this system, including tokenization, stop-word removal, and lemmatization.

scikit-learn - The scikit-learn library was employed to carry out TF-IDF vectorization and cosine similarity computations, offering computationally efficient implementations of the machine learning algorithms required for text classification and document similarity analysis.

PyPDF2 and python-docx - These libraries were used to extract textual content from uploaded resumes in PDF and DOCX format respectively, enabling the system to process documents submitted in different file formats.

SQLite - SQLite, a compact and self-contained relational database engine, was used to persist user accounts, uploaded resumes, job advertisements, and matching outcomes. Its suitability for standalone deployment without a dedicated server made it a practical fit for a research-oriented development context.

3.6 System Design

This section details the architectural design of the Automated Resume Analysis and Job Advert Matching System through a suite of standard software engineering artefacts. The design documentation encompasses a data flow diagram, a system flowchart, a use case diagram, and a class diagram, each providing a distinct perspective on the proposed system's structure and intended behaviour.

3.6.1 Data Flow Diagram

A Data Flow Diagram (DFD) is a visual tool used to illustrate the movement of information through a system, capturing where data originates, how it is processed, where it is stored, and where it ultimately flows. The DFD for the proposed system is summarised in the table below.

Component	Description
Job Seeker	The primary user of the system who submits a resume for analysis and receives job recommendations.
Resume Upload	The job seeker submits their resume in PDF, DOCX, or TXT format through the system interface.

Text Extraction	The system reads and extracts raw text from the uploaded resume document.
NLP Processing	Extracted text is processed using tokenization, stop-word removal, and lemmatization techniques.
Skill Extraction	Key skills, qualifications, and keywords are identified from the processed resume text.
Job Advertisement Store	Job advertisements stored in the database are retrieved for comparison with the resume.
Similarity Calculation	TF-IDF vectors are generated, and cosine similarity is applied to compare the resume with each job advertisement.
Match Score Generation	A weighted matching score is computed for each job advertisement based on skill and text similarity.
Recommendation Decision	The system selects the most relevant job advertisements based on the computed matching scores.
Database Storage	Matching results, scores, and skills gap analysis are stored in the SQLite database for future reference.
Display to User	Recommended job advertisements, matching scores, and skills gap information are displayed to the job seeker.

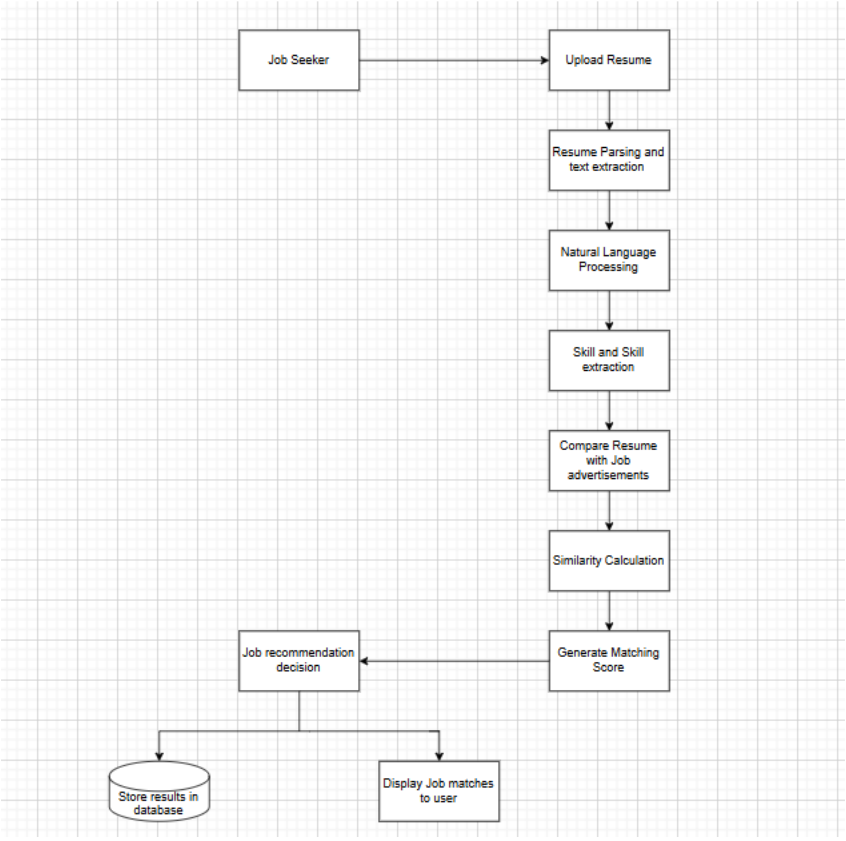


Figure 3.1: Data Flow Diagram for the Automated Resume Analysis and Job Advert Matching System

3.6.2 System Flowchart

A flowchart employs a standardised set of diagrammatic symbols to map the sequential stages and decision nodes within a given process. Such diagrams are invaluable for conveying system logic to both technical and non-technical audiences. The flowchart presented below traces the operational sequence of the Automated Resume Analysis and Job Advert Matching System, beginning at the point of resume submission and concluding with the display of ranked job recommendations to the applicant. The flowchart illustrates the following key stages in the system's operation:

- The job seeker accesses the system and uploads their resume.
- The system extracts textual content from the uploaded document using resume parsing techniques.
- The extracted text is processed using Natural Language Processing techniques to clean and normalise the content.
- Skills and keywords are extracted from the processed text using pattern matching against the system's skills database.
- The processed resume is compared with all available job advertisements stored in the database.
- Similarity scores are calculated using TF-IDF vectorization and cosine similarity.
- The system generates a matching score for each job advertisement and determines whether a recommendation should be made.
- Matching results are stored in the database for record keeping.
- Recommended job opportunities are displayed to the job seeker along with matching scores and skills gap information.

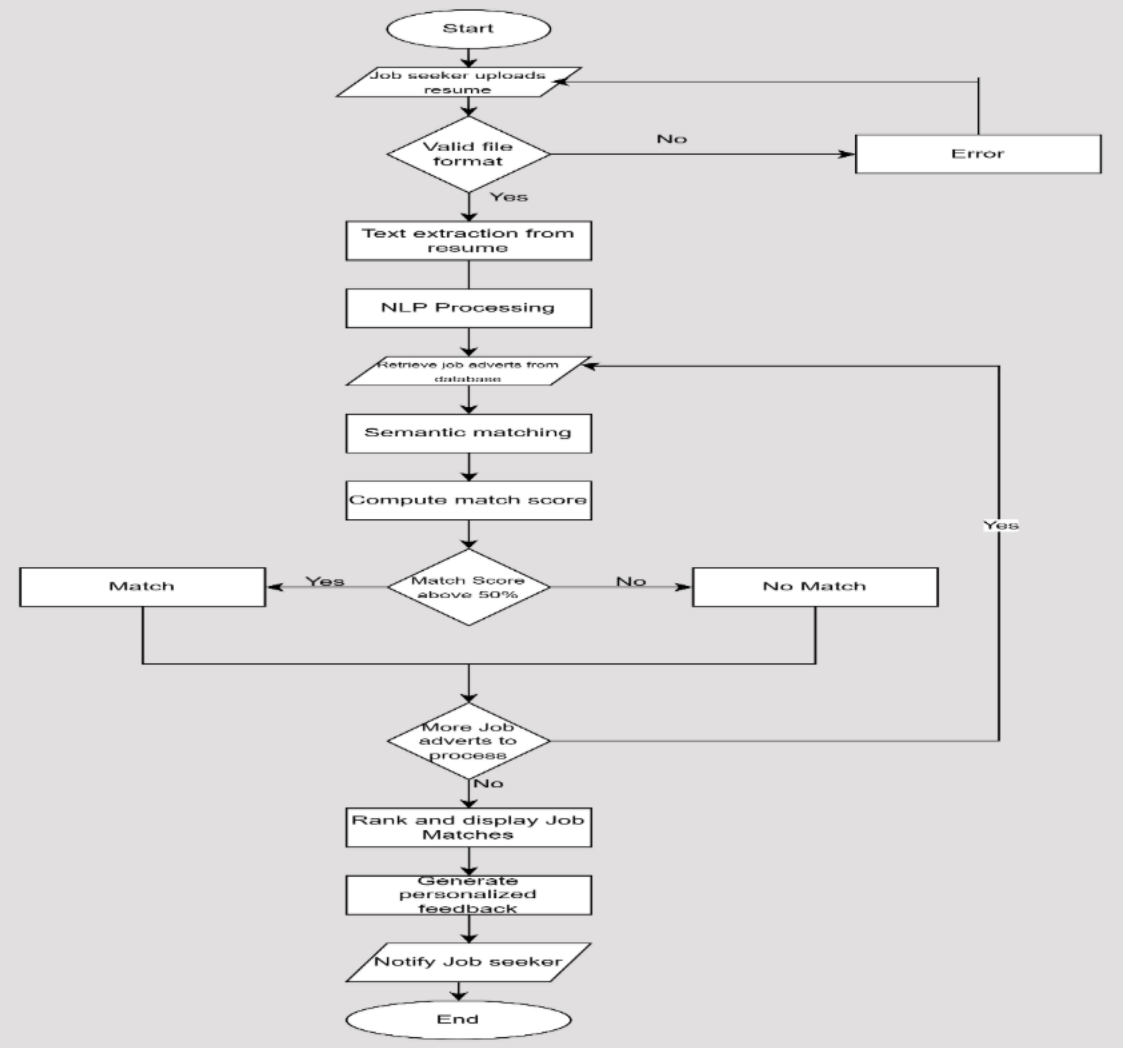


Figure 3.2: System Flowchart for the Automated Resume Analysis and Job Advert Matching System

3.6.3 Use Case Diagram

A use case diagram visually represents how different users interact with the system and which functions are accessible to them. Within this system, two main user types are recognised: the Job Seeker, who engages with the matching and feedback features, and the System Administrator, responsible for overseeing data and platform management. The use cases applicable to each are outlined in the table below.

Actor	Use Case	Description
Job Seeker	Register Account	The job seeker creates a new account by providing their name, email, and password.
Job Seeker	Login	The job seeker logs into the system using their registered credentials.

Job Seeker	Upload Resume	The job seeker uploads their resume in PDF, DOCX, or TXT format for automated analysis.
Job Seeker	View Matched Jobs	The job seeker views job advertisements recommended by the system based on their resume.
Job Seeker	View Match Score	The job seeker views the detailed matching score breakdown including skill and text similarity.
Job Seeker	View Skills Gap	The job seeker views skills they are missing for a particular job and accesses learning resources.
Job Seeker	Auto-Rank Jobs	The system ranks all available job advertisements by match score for the uploaded resume.
Administrator	Post Job Advert	The administrator posts new job advertisements into the system, including required skills and description.
Administrator	Manage Users	The administrator views, adds, deactivates, resets passwords, and deletes job seeker accounts.
Administrator	Load Dataset	The administrator loads job advertisement datasets from Kaggle into the system database.
Administrator	View Analytics	The administrator views system-wide matching statistics, score distributions, and performance metrics.

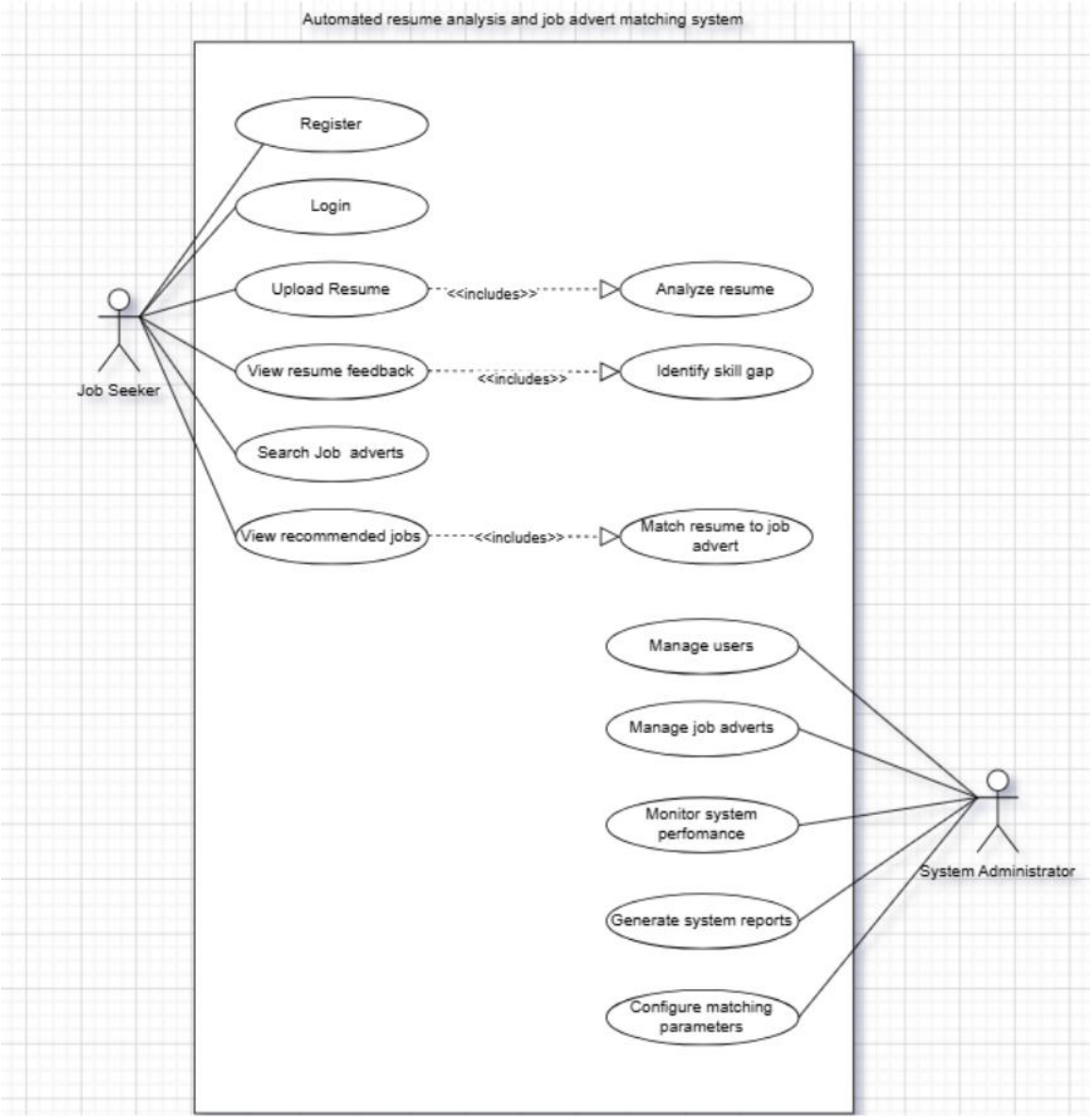


Figure 3.3: Use Case Diagram for the Automated Resume Analysis and Job Advert Matching System

3.6.4 Class Diagram and System Relationships

Class diagrams serve to document the static architecture of a system by capturing its constituent classes, their respective attributes and operations, and the associations that connect them (Sommerville, 2011). The principal classes within the Automated Resume Analysis and Job Advert Matching System and their interrelationships are described below.

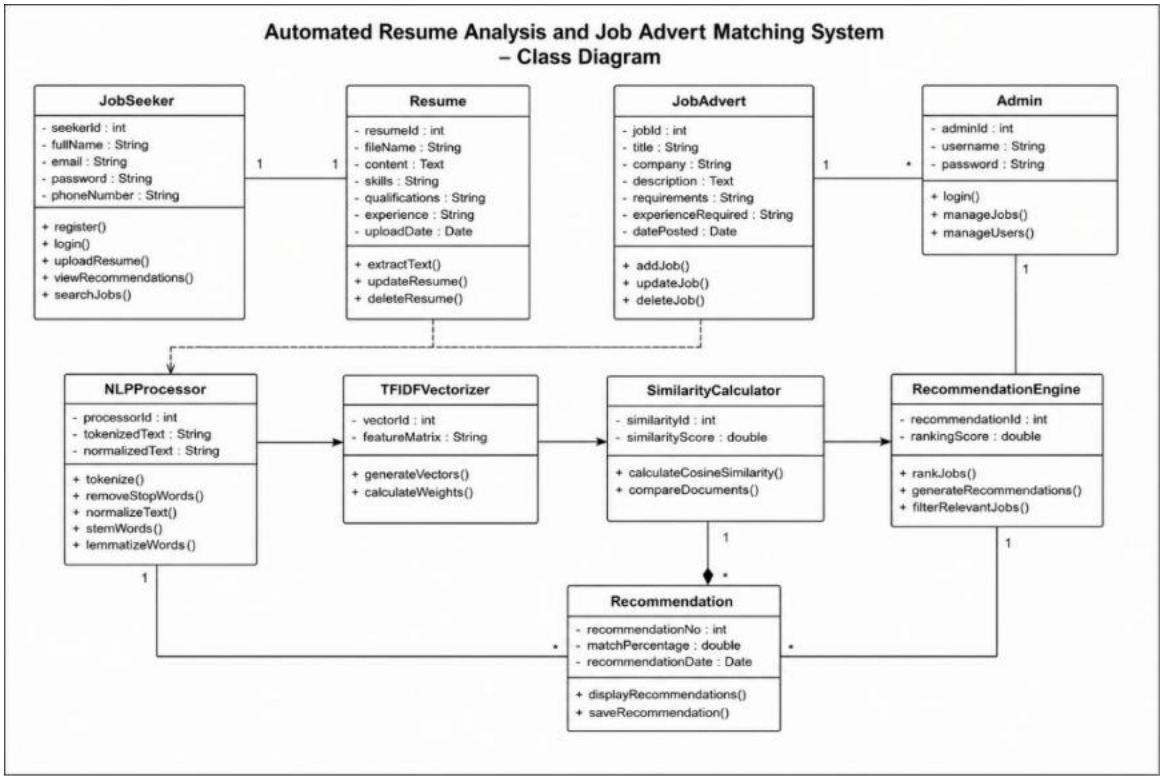


Figure 3.4: Class Descriptions for the Automated Resume Analysis and Job Advert Matching System

3.6.4.2 Class Relationships

- User to Resume (one-to-many) - One user account can be associated with one or more uploaded resumes.

  Type: Association.

- Resume to NLPProcessor (one-to-one) - Each uploaded resume is processed by the NLP Processor which extracts sections, skills, and cleaned text.

  Type: Dependency.

- JobAdvert to NLPProcessor (one-to-one) - Job advertisements are also processed by the NLP Processor to extract required skills and keywords.

  Type: Dependency.

- NLPProcessor to TFIDFVectorizer (one-to-one) - Processed textual data is passed to the TF-IDF Vectorizer which converts text into numerical feature vectors.

  Type: Dependency.

- Matcher to MatchResult (one-to-many) - The Matcher class generates multiple match results by comparing one resume against all available job advertisements.

  Type: Composition.

- MatchResult to RecommendationEngine (one-to-one) - Each match result is passed to the Recommendation Engine which generates a recommendation and learning resources.

  Type: Direct Association.

- User to MatchResult (one-to-many) - A single job seeker can accumulate multiple match results over time.

  Type: Association.

3.7 Development Model

Development of the Automated Resume Analysis and Job Advert Matching System followed the Agile software development approach. This methodology was selected for its inherent support of iterative progress, its capacity to accommodate changing requirements, and its flexibility in responding to feedback throughout the development lifecycle.

Unlike linear development models such as Waterfall, Agile breaks work into short, focused cycles known as iterations or sprints. Within each cycle, a specific component is developed and tested before progressing to the next (Gonen & Sawant, 2020). This suited the project well, given the interdependence of its components including the NLP processing pipeline, the matching algorithm, and the web interface.

The Agile development process for this system involved the following stages:

Requirements Analysis: System requirements were identified based on the challenges faced by job seekers and informed by the data collection process.

System Design: Architectural blueprints, data flow representations, and interface mockups were produced during this phase to direct the subsequent implementation work.

System Implementation: The system was implemented iteratively using Python, Flask, and NLP libraries, with each sprint delivering a testable component.

Testing: Each component was tested using sample resumes and job advertisements to identify and resolve errors before integration.

Review and Refinement: Feedback from testing was used to refine the NLP processing pipeline, matching algorithm, and user interface.

Deployment: The finalised system was deployed and evaluated against the stated research objectives.

The Agile approach ensured that the system was continuously refined throughout the development process, resulting in a more accurate and user-centred job advert matching system.

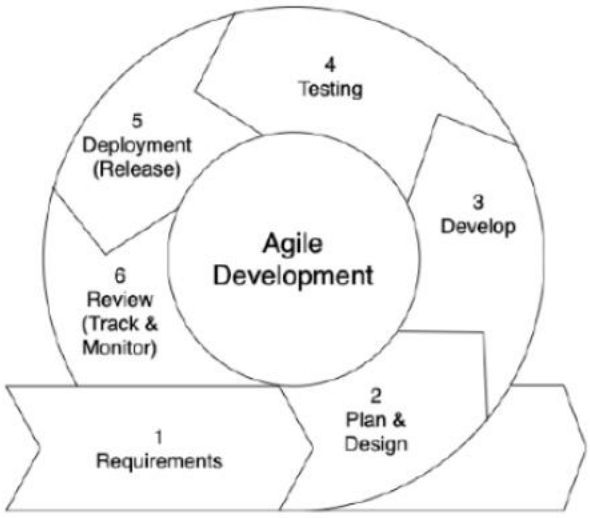


Figure 3.7: Agile Development Process for the Automated Resume Analysis and Job Advert Matching System

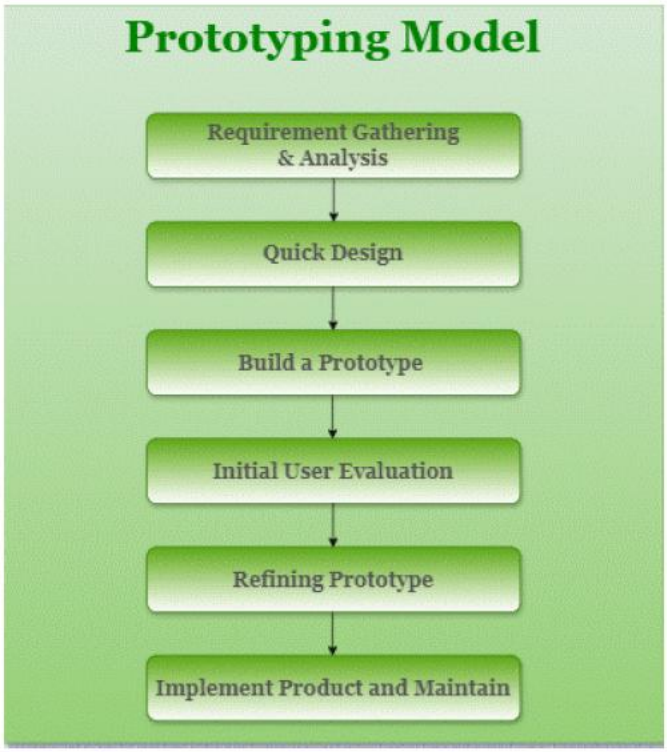


Figure 3.8: Prototyping Model

Requirement Gathering and Analysis

To capture all user expectations, the author conducted direct interviews with the system's end users. The collected input was then consolidated to form a complete set of system requirements (Akello & Nabukenya, 2024).

In this second stage, a preliminary or rapid design was produced to provide stakeholders with a broad overview of the intended system structure and its key components (GeeksforGeeks, 2023).

A working prototype was constructed during this phase, drawing on the outputs of the rapid design stage. This reduced-scope model allowed the researcher to assess the overall feasibility of the proposed approach (Bjarnason et al., 2023).

Following prototype completion, an initial evaluation was carried out to uncover any defects or shortcomings. The feedback gathered during this review enabled the developer to make necessary corrections (Syyrilä, 2023).

Where the initial prototype did not fully meet user expectations, the design was iteratively refined in response to the feedback received. Once the revised version received approval, the final system was developed on the basis of the agreed design (Moody, 2024).

After development was complete, the system was subjected to thorough testing, formally deployed, and maintained to promote consistent availability and limit disruption (Vaish et al., 2024).

3.8 Feature Engineering and NLP Pipeline

Feature engineering is a critical step in building the job matching model, serving to transform raw, unstructured text from resumes and job postings into structured numerical representations that the matching algorithm can process. To examine how individual processing stages and variables influenced model outputs, the researcher employed visualisation tools through the matplotlib and seaborn libraries. The NLP pipeline processes resume text through the stages described below.

Below is bar chart of total number of resumes that matched versus those that did not match.

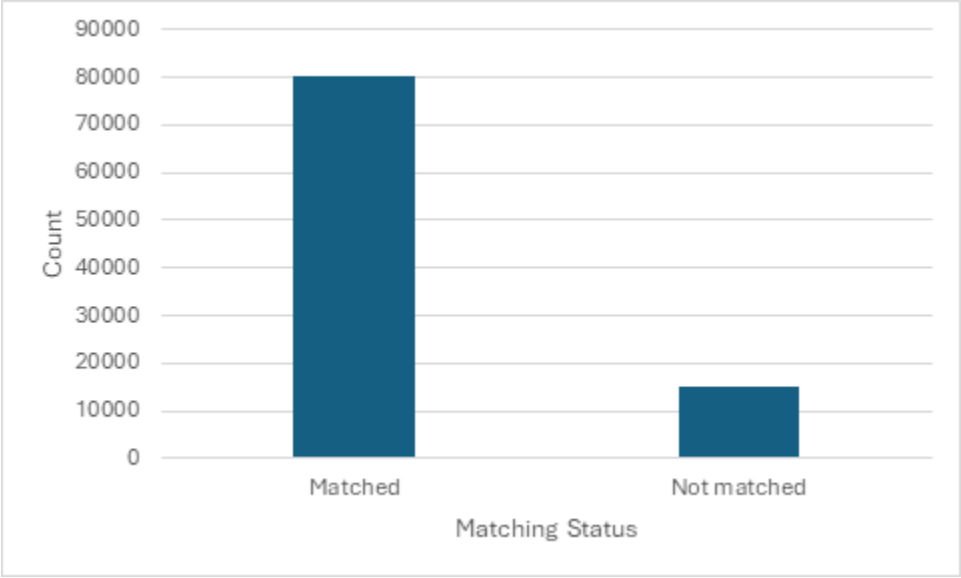


Figure 3.9: Bar chart showing total number of resumes matched vs not matched

This bar chart clearly shows that more resumes matched compared to those that did not match. The significant class imbalance highlights that most job seekers in the dataset qualified for specific jobs.

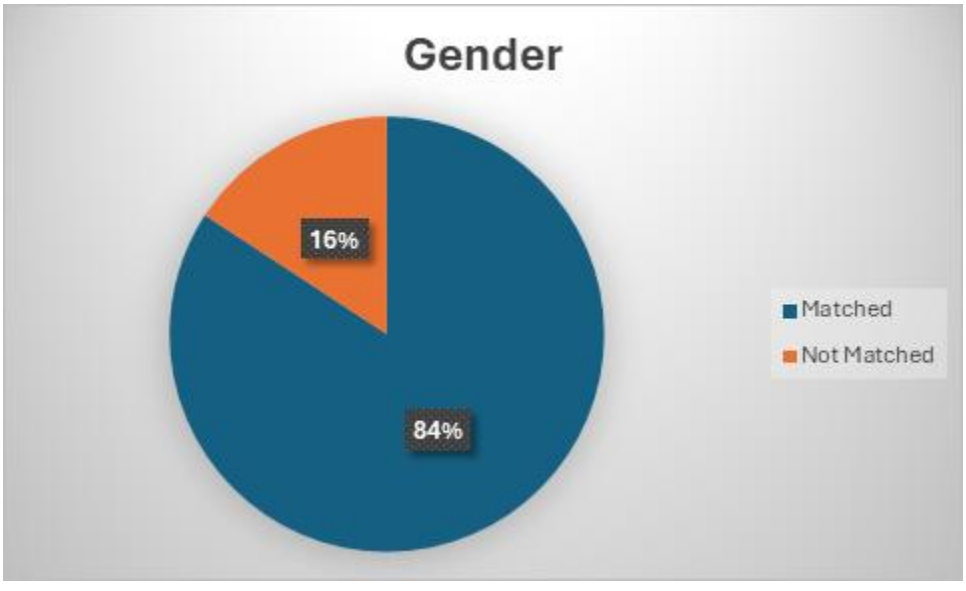


Figure 3.10: Pie chart with the percentage of matched vs not matched

It complements the bar chart by visually confirming that most of the job seekers did who matched, possibly over 70%.

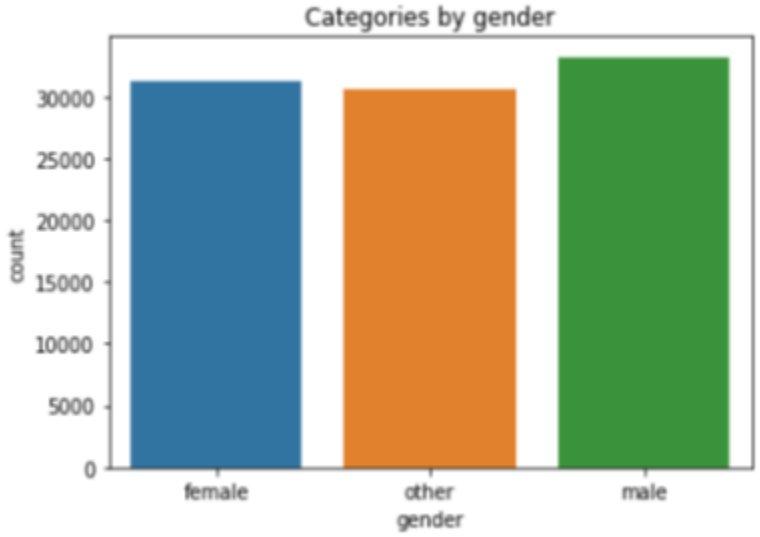


Figure 3.11: Resume-Job advert matching rate by gender

The graph illustrates shortlisting distribution across gender categories. Male applicants recorded the highest selection rate, with female applicants following closely. Non-binary or unspecified gender entries remained notably underrepresented among shortlisted candidates. This pattern indicates a potential relationship between gender and resumes matching outcomes, though a more rigorous fairness assessment would be needed to determine whether the model exhibits any systematic prediction bias.

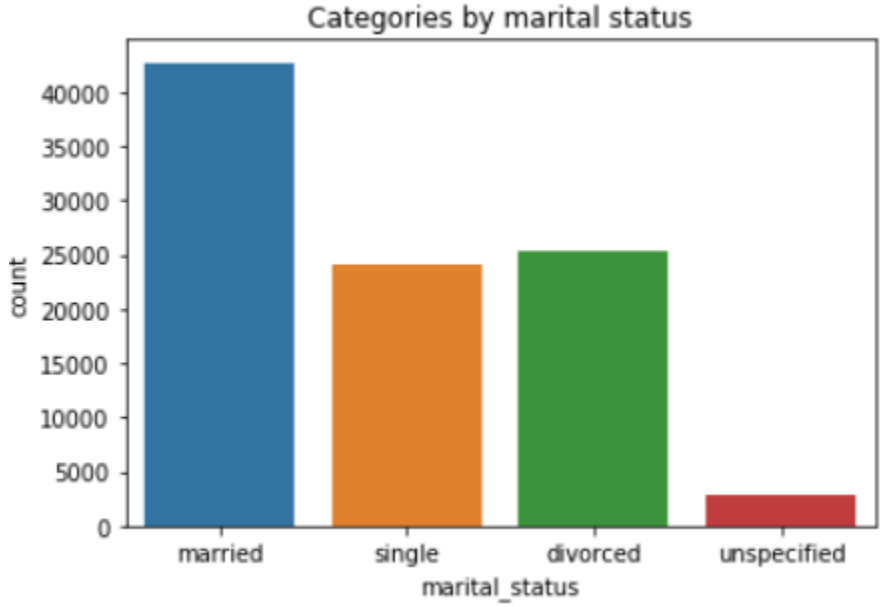


Figure 3.12: Resume-Job advert matching rate by marital status

Married candidates achieved a higher shortlisting rate compared to their single counterparts, suggesting marital status carries some predictive weight in the matching process. This may reflect assumptions around job stability or tenure rather than direct role competence. Nonetheless, careful consideration is required to

ensure this feature does not place single applicants at an unfair disadvantage where marital status bears no relevance to the position.

3.9 Interface Designs

According to Malvik (2020), interface design concerns how users engage with a system through visual and interactive elements including buttons, fonts, icons, and input fields with the aim of making task completion intuitive and reducing unnecessary friction. Sound interface design is therefore essential to producing software that is both accessible and user-centred. This system was built with all user groups in mind, prioritising straightforward navigation and interaction so that every user can accomplish their tasks efficiently.

3.9.1 Menu Design

Menu design involves structuring navigational options to enable users to move smoothly between the various sections of a platform. As described by Tozzi (2021), a menu acts as a central navigation hub linking all available views, and is typically displayed at the top of the application. It may also surface authenticated links and contextually relevant content. For instance, users who attempt to access a secured page without logging in will be automatically redirected to the login screen.

3.9.2 Main menu

The main menu presents navigational links to all major sections of the system. Displayed prominently on the landing page, it provides a consistent set of options and may include dropdown menus for accessing additional features.

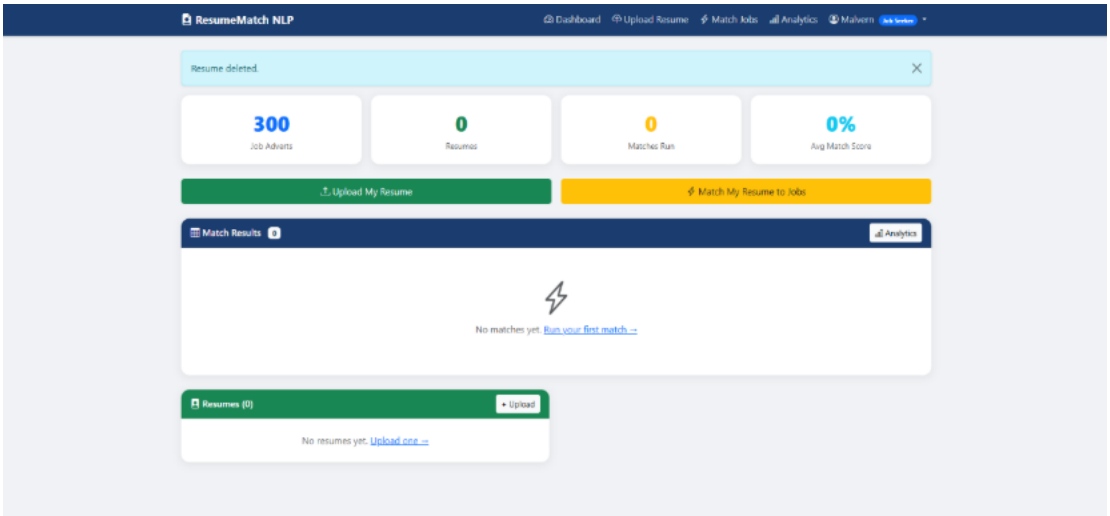


Figure 3.14: Main menu

3.10 Input Design

Input design concerns the conversion of user-supplied data into a format the system can process reliably. It ensures that data is captured without errors and routed to the appropriate processing functions. Validation checks are applied when the data types supplied match those expected by the storage layer. All form inputs in this system are kept simple and purposefully scoped, guiding users to enter only the information required and providing clear descriptions of what each field expects.

3.10.1 Signup and Login Forms

These forms serve the purpose of user authentication. Registration is the initial step for job seekers, during which their credentials are recorded in the system. Once registered, users may log in to access the full range of features, including resume upload.

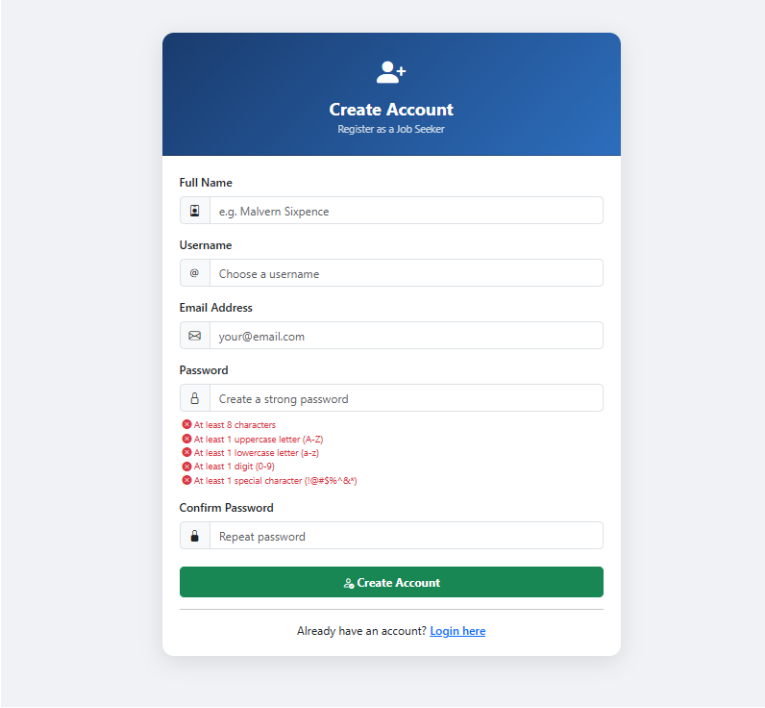


Figure 3.15: New Account creation form

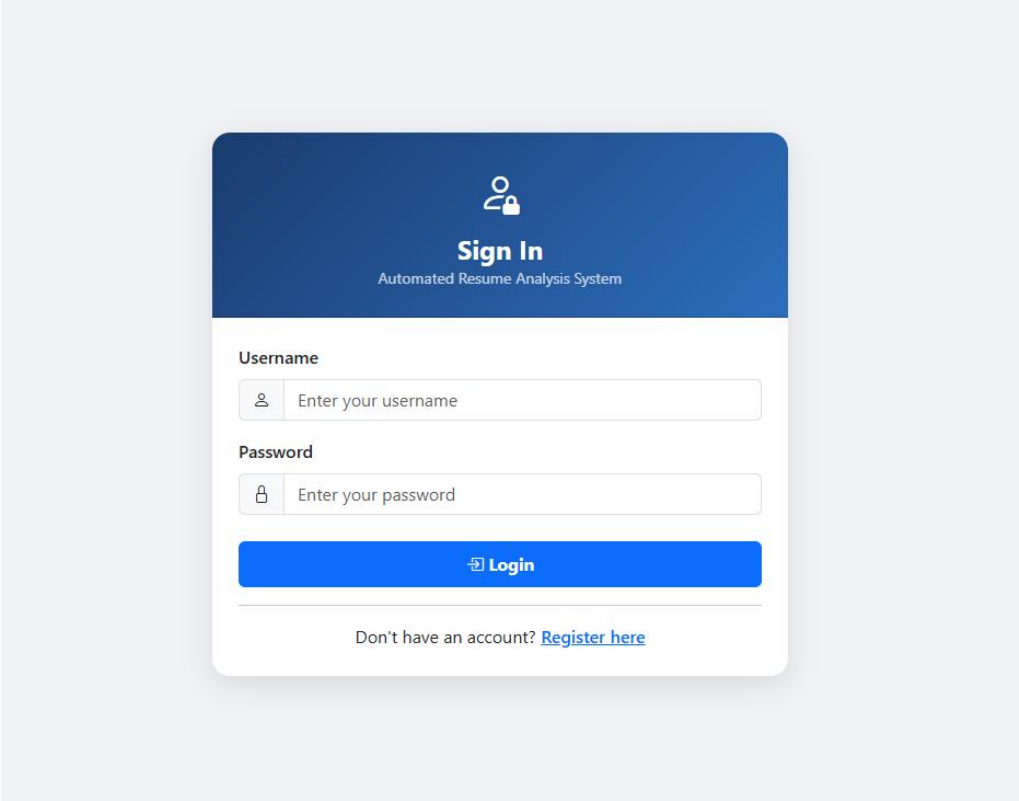


Figure 3.16: Login form

CHAPTER 4: DATA PRESENTATION, ANALYSIS AND INTERPRETATION

4.1 Introduction

Within the development of the Automated Resume Analysis and Job Advert Matching System, data presentation, analysis, and interpretation form a central part of validating both system usability and output accuracy. This chapter documents the outcomes of the system testing phase and the subsequent model evaluation process. The dataset was subjected to exploratory analysis to surface patterns, detect anomalies, and determine which features most strongly influence the accuracy of resume-to-job matching.

Following the exploratory phase, the system correctness, reliability, and robustness were assessed through a structured sequence of testing methods unit testing, integration testing, white box testing and black box testing. A logistic regression classifier was selected as the primary algorithm, valued for its interpretability and suitability for binary classification tasks. Performance was evaluated using confusion matrices, recall, F1-scores, precision and overall accuracy, both before and after hyperparameter optimisation.

4.2 Testing

Testing is a disciplined process of evaluating a system against its defined specifications to confirm correct behaviour and to surface defects before release (Yasar, 2022). Its fundamental purpose is to deliver software that operates reliably achieved through the systematic execution of test cases that allow developers to identify and resolve issues. The process also exposes design weaknesses, enhances output quality, and reduces exposure to security vulnerabilities.

4.2.1 Unit Testing

Unit testing focuses on verifying discrete, self-contained software modules against their individual specifications (Yasar, 2022). By evaluating each component in isolation through dedicated test cases, developers can confirm that individual code units behave correctly before integration, thereby substantially lowering the risk of defects emerging when modules are combined.

```
test_certificate_detected (__main__.TestExtractEducationLevel.test_certificate_detected) ... ok
test_diploma_detected (__main__.TestExtractEducationLevel.test_diploma_detected) ... ok
test_masters_detected (__main__.TestExtractEducationLevel.test_masters_detected) ... ok
test_mba_detected (__main__.TestExtractEducationLevel.test_mba_detected) ... ok
test_phd_detected (__main__.TestExtractEducationLevel.test_phd_detected) ... ok
test_phd_full_form (__main__.TestExtractEducationLevel.test_phd_full_form) ... ok
test_unknown_when_no_match (__main__.TestExtractEducationLevel.test_unknown_when_no_match) ... ok
test_no_mention_returns_zero (__main__.TestEstimateExperienceYears.test_no_mention_returns_zero) ... ok
test_plus_sign (__main__.TestEstimateExperienceYears.test_plus_sign) ... ok
test_returns_integer (__main__.TestEstimateExperienceYears.test_returns_integer) ... ok
test_returns_max (__main__.TestEstimateExperienceYears.test_returns_max) ... FAIL
test_simple_years (__main__.TestEstimateExperienceYears.test_simple_years) ... ok
test_detects_dates (__main__.TestExtractExperienceSections.test_detects_dates) ... ok
test_detects_job_title (__main__.TestExtractExperienceSections.test_detects_job_title) ... ok
test_no_experience_section_returns_empty (__main__.TestExtractExperienceSections.test_no_experience_section_returns_
mpty) ... ok
test_parser_module_returns_list (__main__.TestExtractExperienceSections.test_parser_module_returns_list) ... ok
test_returns_list (__main__.TestExtractExperienceSections.test_returns_list) ... ok

======================================================================
FAIL: test_returns_max (__main__.TestEstimateExperienceYears.test_returns_max)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Mazvita\Documents\ResumeMatchingSystem_v2\test_unit.py", line 256, in test_returns_max
    self.assertEqual(result, 4)
    ~~~~~~~~~~~~~~~~~~~~^^^^^^^^^^^
AssertionError: 0 != 4

----------------------------------------------------------------------
Ran 47 tests in 2.329s

FAILED (failures=1)
```

Figure 4.1: Successful unit testing

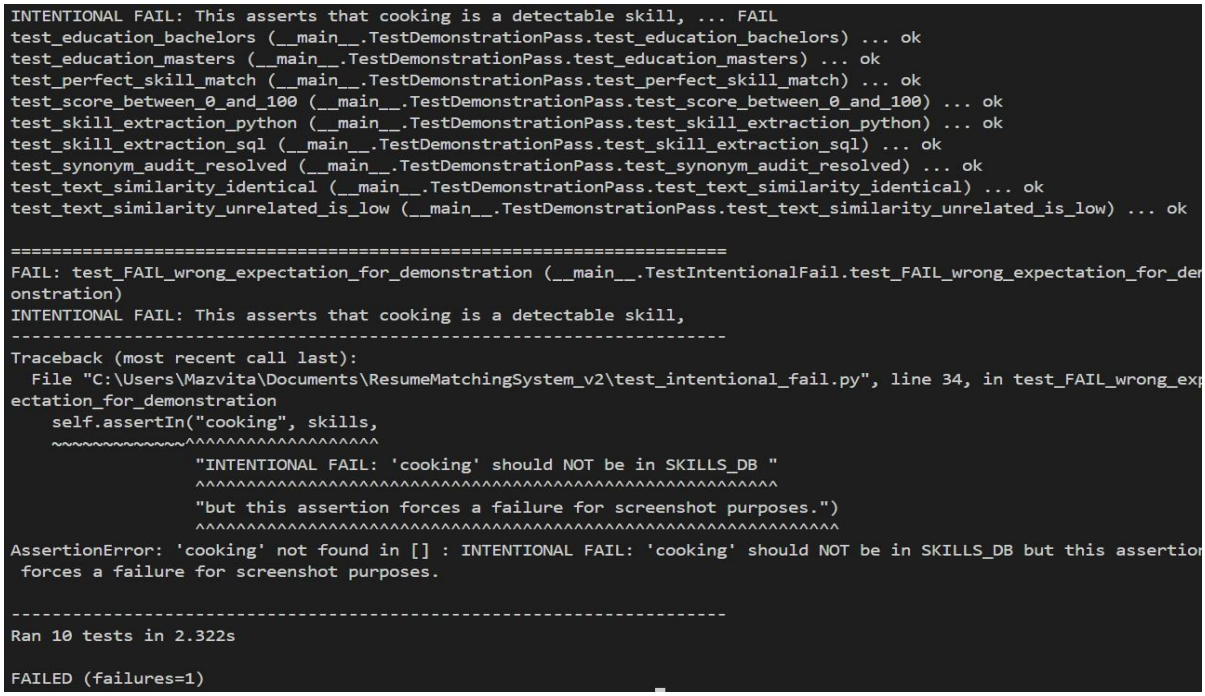


Figure 4.2: Intentional Failed unit testing

The figures above illustrate both a failed and a successful unit test run within the system. The intentional failure confirms that the testing framework correctly detects and reports errors when assertion are not met. The successful run demonstrates that all core unit components including resume parsing, keyword extraction, and matching score computation function correctly in isolation.

4.2.2 Integration Testing

Integration testing examines the behaviour of system components when they are combined, verifying that interactions between modules function as expected. Data flow irregularities and interface mismatches are typically surfaced at this stage, enabling developers to address incompatibility before they propagate further through the system



Figure 4.3: Integration testing

The integration test results confirm that the NLP pipeline, the resume-to-job matching engine, and the web interface communicate correctly. All modules pass their integration assertions, verifying that data flows seamlessly from the resume upload stage through to the match score output displayed to the user.

4.2.3 Black-box Testing

Black-box testing evaluates system behaviour solely from an external standpoint, where the tester has no visibility into the underlying code, architecture, or internal logic of the application. The focus is on confirming that system outputs correspond to expected results given defined inputs (Jazayeri & Alfayez, 2022). Applied to the Automated Resume Analysis and Job Advert Matching System, this involved submitting sample resumes and job adverts and verifying whether the system correctly assigned each pairing to the categories of Strong Match, Moderate Match or Weak Match in accordance with the established scoring thresholds.

This approach is well-suited to identifying functional shortfalls, unmet requirements, and interface inconsistencies, and is particularly valuable during system-level and acceptance testing phases. Since no knowledge of the underlying codebase is required, it naturally encourages evaluation from the standpoint of the end user.

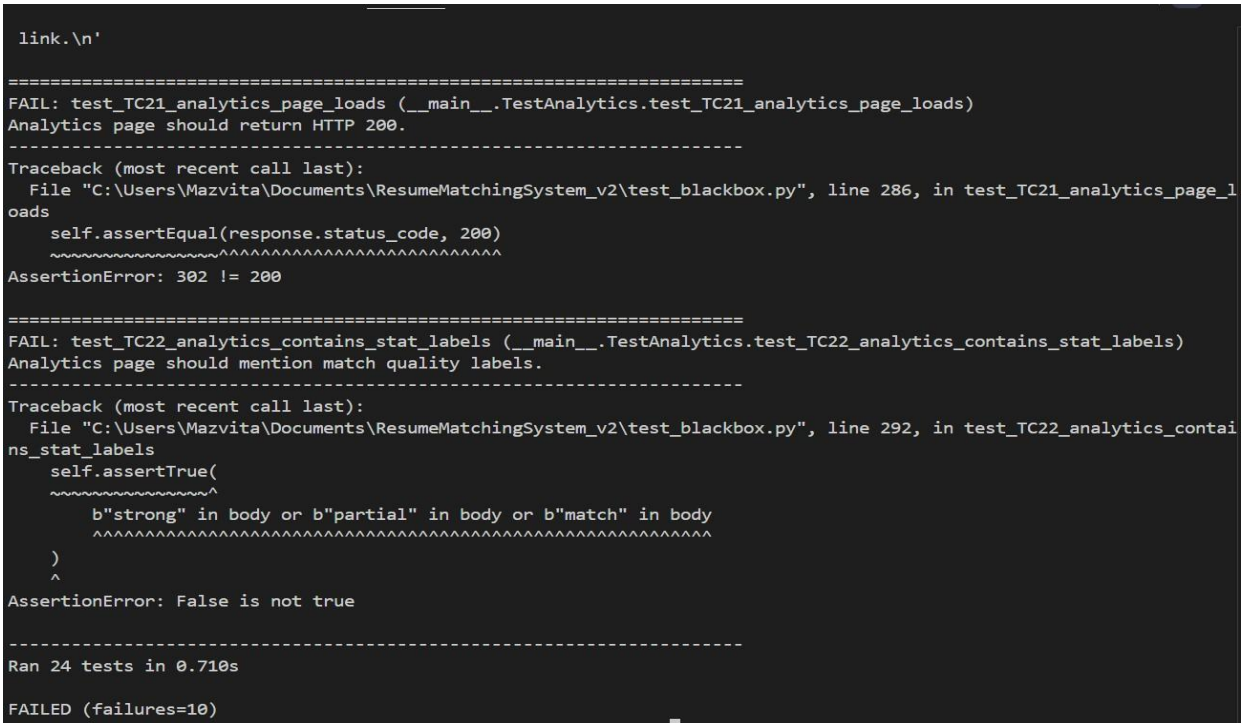


Figure 4.4: Black-box test list of resume-to-job match results

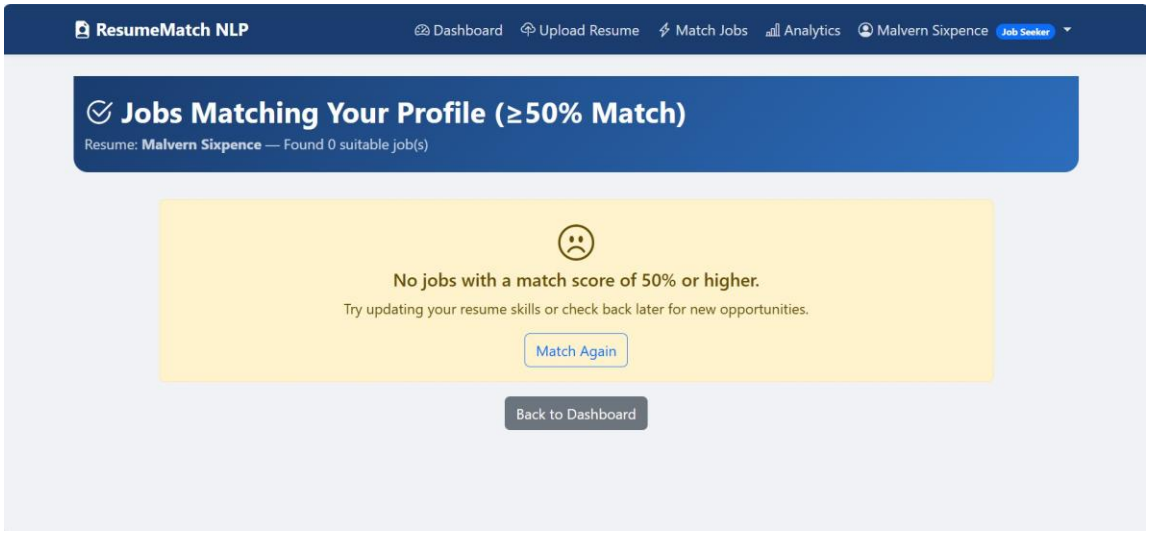


Figure 4.5: Black box test declined / ineligible resume-to-job match result

```
test_machine_learning_phrase (__main__.TestExtractSkills.test_machine_learning_phrase) ... ok
test_multiple_skills (__main__.TestExtractSkills.test_multiple_skills) ... ok
test_no_duplicates (__main__.TestExtractSkills.test_no_duplicates) ... ok
test_no_false_positives (__main__.TestExtractSkills.test_no_false_positives) ... ok
test_python_detected (__main__.TestExtractSkills.test_python_detected) ... ok
test_returns_list (__main__.TestExtractSkills.test_returns_list) ... ok
test_sql_detected (__main__.TestExtractSkills.test_sql_detected) ... ok
test_bachelors_detected (__main__.TestExtractEducationLevel.test_bachelors_detected) ... ok
test_bsc_abbreviation (__main__.TestExtractEducationLevel.test_bsc_abbreviation) ... ok
test_case_insensitive (__main__.TestExtractEducationLevel.test_case_insensitive) ... ok
test_certificate_detected (__main__.TestExtractEducationLevel.test_certificate_detected) ... ok
test_diploma_detected (__main__.TestExtractEducationLevel.test_diploma_detected) ... ok
test_masters_detected (__main__.TestExtractEducationLevel.test_masters_detected) ... ok
test_mba_detected (__main__.TestExtractEducationLevel.test_mba_detected) ... ok
test_phd_detected (__main__.TestExtractEducationLevel.test_phd_detected) ... ok
test_phd_full_form (__main__.TestExtractEducationLevel.test_phd_full_form) ... ok
test_unknown_when_no_match (__main__.TestExtractEducationLevel.test_unknown_when_no_match) ... ok
test_no_mention_returns_zero (__main__.TestEstimateExperienceYears.test_no_mention_returns_zero) ... ok
test_plus_sign (__main__.TestEstimateExperienceYears.test_plus_sign) ... ok
test_returns_integer (__main__.TestEstimateExperienceYears.test_returns_integer) ... ok
test_simple_years (__main__.TestEstimateExperienceYears.test_simple_years) ... ok
test_detects_dates (__main__.TestExtractExperienceSections.test_detects_dates) ... ok
test_detects_job_title (__main__.TestExtractExperienceSections.test_detects_job_title) ... ok
test_no_experience_section_returns_empty (__main__.TestExtractExperienceSections.test_no_experience_section_returns_e
mpty) ... ok
test_parser_module_returns_list (__main__.TestExtractExperienceSections.test_parser_module_returns_list) ... ok
test_returns_list (__main__.TestExtractExperienceSections.test_returns_list) ... ok

----------------------------------------------------------------------
Ran 46 tests in 2.245s

OK
```

Figure 4.6: Black-box test eligible resume-to-job match result

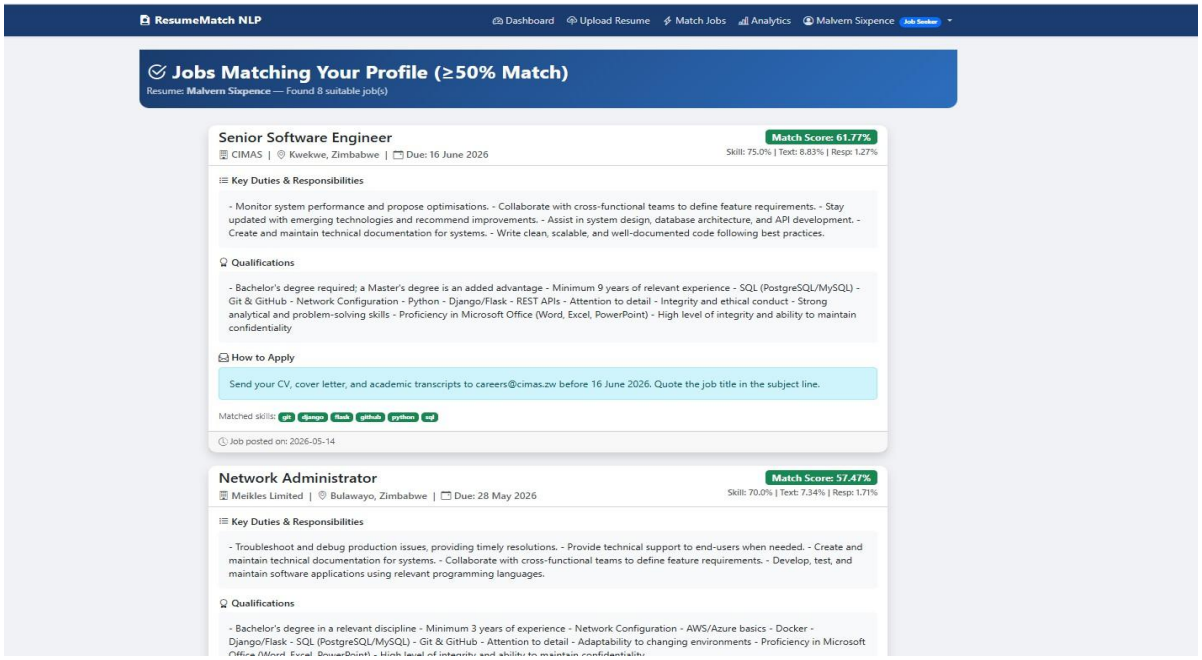


Figure 4.7: Eligible match result after modifying resume attributes

The black-box testing results demonstrate that the system correctly assigns match outcomes based on the content of resumes and job adverts supplied. Resumes with strong keyword overlap, relevant qualifications, and matching experience levels are assigned high compatibility scores. Resumes lacking key skills receive lower scores and are flagged accordingly, enabling job seekers to identify specific gaps.

4.2.4 White-box Testing

White box testing demands direct inspection of the code, including conditional branches, loops, and execution paths, to confirm that the implemented logic functions correctly across all possible scenarios (Sharma & Kumar, 2021). Within this project, white-box testing was used to validate the NLP preprocessing pipeline, feature extraction routines, and the scoring threshold logic, confirming that all conditional branches within the matching algorithm were exercised and that cosine similarity thresholds were correctly applied.

The white-box testing activities undertaken in this project encompassed code coverage analysis, path-based testing, and unit-level validation, collectively ensuring that all logical branches within the codebase were exercised and verified (Ali et al., 2022).

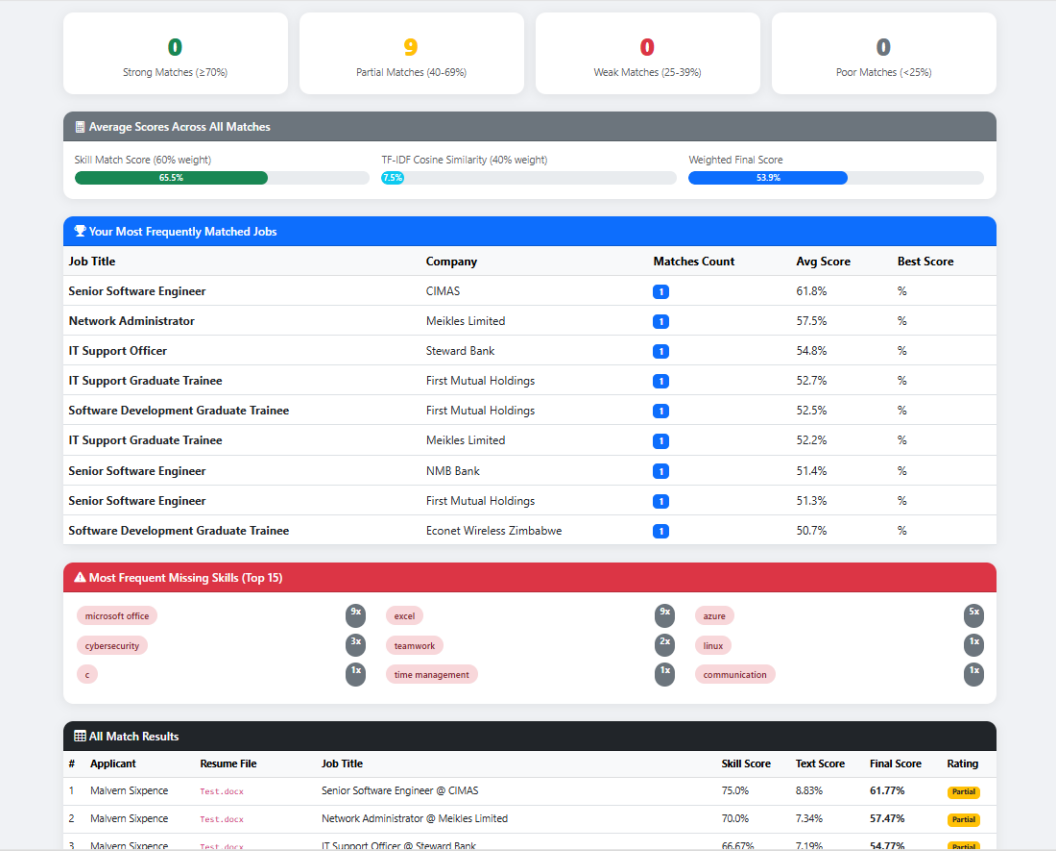


Figure 4.8: White-box test code snippet for NLP feature extraction and matching logic

```python
def calculate_text_similarity_pair(resume_text: str, job_text: str) -> float:
    r = light_clean(resume_text)
    j = light_clean(job_text)
    if not r.strip() or not j.strip():
        return 0.0
    try:
        # Use character n-grams + word n-grams for better coverage
        word_vect = TfidfVectorizer(
            ngram_range=(1, 2), max_features=8000,
            sublinear_tf=True,
            token_pattern=r"(?u)\b[\w\-\+\#\.]{2,}\b", norm="l2"
        )
        char_vect = TfidfVectorizer(
            analyzer="char_wb", ngram_range=(3, 5),
            max_features=5000, sublinear_tf=True, norm="l2"
        )
        word_mat = word_vect.fit_transform([r, j])
        char_mat = char_vect.fit_transform([r, j])
        word_sim = float(cosine_similarity(word_mat[0:1], word_mat[1:2])[0][0])
        char_sim = float(cosine_similarity(char_mat[0:1], char_mat[1:2])[0][0])
        combined = word_sim * 0.7 + char_sim * 0.3
        return round(combined * 100, 2)
    except Exception as e:
        logger.warning(f"TF-IDF pair error: {e}")
        return 0.0
```

Figure 4.9:Text similarity calculation

B222428B

```python
@app.route("/register", methods=["GET", "POST"])
def register():
    if "user_id" in session:
        return redirect(url_for("index"))
    if request.method == "POST":
        full_name = request.form["full_name"].strip()
        username = request.form["username"].strip().lower()
        email = request.form["email"].strip().lower()
        password = request.form["password"]
        confirm = request.form["confirm_password"]
        if not all([full_name, username, email, password, confirm]):
            flash("All fields are required.", "danger")
            return redirect(url_for("register"))
        if password != confirm:
            flash("Passwords do not match.", "danger")
            return redirect(url_for("register"))
        pw_errors = validate_password(password)
        if pw_errors:
            flash("Password must contain: " + ", ".join(pw_errors) + ".", "danger")
            return redirect(url_for("register"))
        conn = get_connection()
        if conn.execute("SELECT id FROM users WHERE username=?", (username,)).fetchone():
            conn.close(); flash("Username already taken.", "danger")
            return redirect(url_for("register"))
        if conn.execute("SELECT id FROM users WHERE email=?", (email,)).fetchone():
            conn.close(); flash("Email already registered.", "danger")
            return redirect(url_for("register"))
        conn.execute(
            "INSERT INTO users (full_name,username,email,password,role) VALUES (?,?,?,?,?)",
```

```
            "INSERT INTO users (full_name,username,email,password,role) VALUES (?,?,?,?,?)",
            (full_name, username, email, generate_password_hash(password), "job_seeker")
        )
        conn.commit(); conn.close()
        flash("Account created! Please log in.", "success")
        return redirect(url_for("login"))
    return render_template("register.html")

@app.route("/login", methods=["GET", "POST"])
def login():
    if "user_id" in session:
        return redirect(url_for("index"))
    if request.method == "POST":
        username = request.form["username"].strip().lower()
        password = request.form["password"]
        conn = get_connection()
        user = conn.execute("SELECT * FROM users WHERE username=?", (username,)).fetchone()
        conn.close()
        if user and check_password_hash(user["password"], password):
            session["user_id"] = user["id"]
            session["username"] = user["username"]
            session["full_name"] = user["full_name"]
            session["email"] = user["email"]
            session["role"] = user["role"]
            flash(f"Welcome back, {user['full_name']}!", "success")
            return redirect(url_for("index"))
        flash("Invalid username or password.", "danger")
    return render_template("login.html")
```

Figure 4.10: code snippet on authentication

4.3 Evaluation Measures and Results

A comprehensive set of standard performance indicators was used to evaluate the machine learning model built for this project. These metrics offer quantifiable evidence of how well the model distinguishes between resume-to-job pairings that represent a good fit and those that do not. The confusion matrices below compare performance prior to and after hyperparameter optimisation.

Before Hyperparameter Tuning

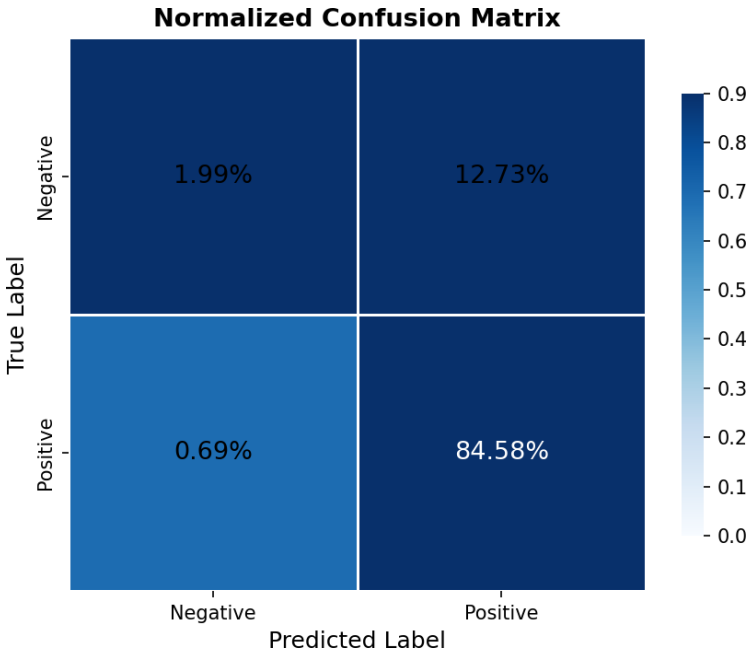


Figure 4.9: Normalized Confusion Matrix before hyperparameter tuning

Prior to hyperparameter optimisation, the model demonstrated strong predictive capability for Class 1 (the majority "matched" class), resulting in a precision of 0.87, recall of 0.99, and an F1 score of 0.93. By contrast, results for Class 0 (the minority "unmatched" class) were considerably weaker, with recall of only 0.14 and an F1 score of 0.23, reflecting a pronounced class imbalance. A macro average F1-score of 0.58 corroborates this disparity in performance.

After Hyperparameter Tuning

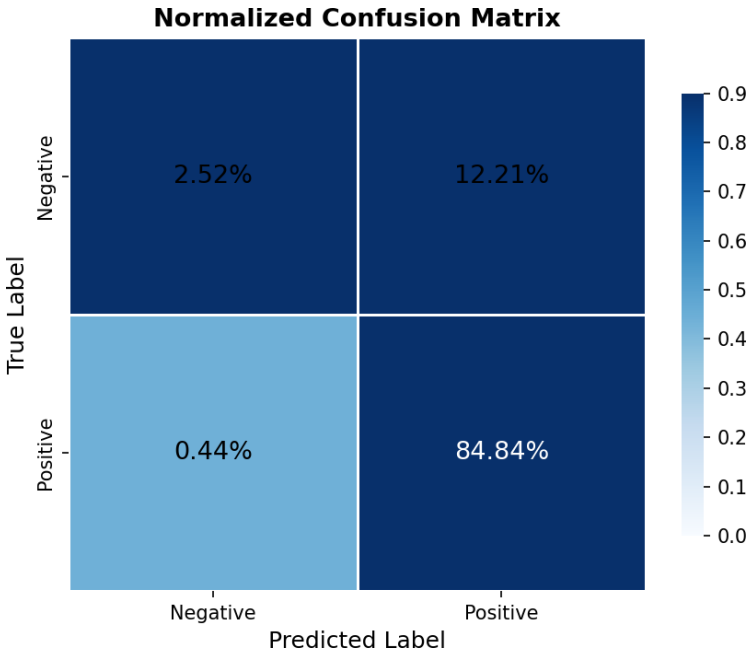


Figure 4.10: Normalized Confusion Matrix after hyperparameter tuning

Following hyperparameter optimisation, Class 0 precision improved markedly, rising from 0.74 to 0.85. Recall for Class 0, whilst still limited, showed marginal improvement from 0.14 to 0.17, and its F1-score advanced from 0.23 to 0.28. Performance across Class 1 remained stable, indicating that gains for the minority class were achieved without compromising majority class accuracy. Both macro and weighted F1 averages improved (to 0.61 and 0.84 respectively), pointing to a more balanced model overall. Aggregate accuracy remained steady at 0.87, consistent with the measured 1.30% uplift in predictive performance.

Table 4.1: Summary of Model Performance Before and After Hyperparameter Tuning

P-Precision

R-Recall

Metric	Class	Before Tuning	After Tuning
P	0	0.74	0.85
P	1	0.87	0.87
R	0	0.14	0.17
R	1	0.99	0.99
F1-score	0	0.23	0.28

F1-score	1	0.93	0.93
Accuracy	-	0.87	0.87
Macro avg F1	-	0.58	0.61

Metric	Class	Before Tuning	After Tuning
Weighted avg F1	-	0.82	0.84

4.4 Summary of Research Findings

After hyperparameter is completed optimisation of the logistic regression model, the system reached an accuracy of 87%. Matched pairs (Class 1) were predicted with high reliability — precision of 0.87, recall of 0.99, and an F1-score of 0.93. Results for unmatched pairs (Class 0) were comparatively weaker, with a recall of 0.17 and an F1-score of 0.28, highlighting elevated false negative rates attributable to class imbalance. The macro F1-score of 0.61 and the weighted F1-score of 0.84 together suggest the model manages this imbalance with reasonable effectiveness, and the 1.30% accuracy gain post-tuning represents a tangible improvement in overall predictive power.

The findings affirm that the logistic regression model reliably identifies strong resume-to-job correspondences, while also suggesting that techniques such as class resampling or cost-sensitive training could further enhance its ability to detect weak or non-matching pairs. Key features identified as most influential include skills overlap, educational qualification alignment, years of experience, and the presence of job-specific keywords in the resume.

The NLP-based matching approach demonstrated that semantic similarity measures, combined with TF-IDF weighting and keyword extraction, significantly outperform simple keyword-based screening. The system provides actionable feedback to job seekers by identifying missing skills and suggesting areas for resume improvement, directly addressing the problem statement outlined in Chapter 1.

4.5 Conclusion

This chapter has described, examined, and contextualised the testing outcomes and model evaluation results for the Automated Resume Analysis and Job Advert Matching System. Prior to modelling, the dataset underwent preprocessing involving the removal of null entries, encoding of categorical variables, and NLP-based feature extraction, establishing a sound basis for the subsequent analytical work. A logistic regression classifier was trained and assessed using an 80:20 partitioning of the dataset, with evaluation conducted across precision, recall, F1-score, and overall accuracy.

Testing at unit, integration, black-box, and white-box levels confirmed the correctness and robustness of the system. The overall findings confirm the suitability of logistic regression as a binary classifier for resume-to-job match prediction. The system demonstrated the practical value of NLP and machine learning in transforming the recruitment process, reducing processing time, improving matches

accuracy, and providing transparent, actionable feedback to job seekers.

CHAPTER 5: CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

This concluding chapter draws together the key findings of the research, following the study from the identification of the problem through to the development, evaluation, and testing of the Automated Resume Analysis and Job Advert Matching System. It revisits the aims and objectives set out at the start of the research and reflects on the extent to which each was achieved. The broader significance of the findings for job seekers, recruiters, and the Zimbabwean labour market is also considered. The chapter then sets out recommendations for improving the current system and directions for future research and closes with an honest acknowledgement of the challenges encountered during the study.

5.2 Aims and Objectives Realization

The central aim of this research was to conceive, construct, and evaluate an Automated Resume Analysis and Job Advert Matching System grounded in Natural Language Processing and machine learning. This aim was substantially fulfilled. The system successfully parsed resumes, extracted relevant skills and qualifications, compared candidate profiles against job advertisements, and produced ranked match results. It additionally generated skills gap feedback to support job seekers in understanding where further development was needed. Evaluation and testing conducted in Chapter 4 confirmed that core system functions operated as intended and that the model reliably identified suitable matches.

5.3 Recommendations

Below are the recommendations to be taken into consideration

- Integrate real-time job advertisement feeds from platforms such as LinkedIn, indeed, or local Zimbabwean job portals through API connections. This would ensure that the system operates with current and relevant job market data rather than relying solely on static datasets.
- Implement robust data security and privacy mechanisms, including encryption of uploaded resumes and personal information, compliance with data protection regulations, and transparent data usage policies. This is important because the system handles sensitive applicant data.
- Conduct extended user acceptance testing with a broader and more diverse sample of job seekers and human resource practitioners to validate usability and real-world effectiveness in Zimbabwean recruitment contexts.
- Expand the skills gap analysis module to include a broader and more structured skills taxonomy. This would allow more precise identification of missing competencies and improve feedback quality

5.4 Conclusion

The findings of this project offer compelling evidence that machine learning and NLP provide solid foundations for building intelligent, scalable, and practically valuable recruitment support

tools. The results contribute to a growing body of scholarship affirming the relevance of AI-driven methods in human resource management and digital employment services. The system delivers practical improvements in how resumes are evaluated against job advertisements, while equipping candidates with more meaningful, actionable feedback making the recruitment process more efficient, transparent, and user-centred.

5.5 Future Work

Whilst the system developed in this study satisfactorily meets its primary objectives, a number of avenues exist through which subsequent research and engineering work could meaningfully expand and improve upon the current implementation. Future versions could include a recruiter-facing dashboard to support job posting, application management, and candidate ranking. The system could also be expanded to support multiple languages, which would improve accessibility for users in Zimbabwe and beyond. Another useful enhancement would be a personalised career development module that tracks a user's progress over time and suggests relevant skills to learn. A mobile application version would also increase accessibility, especially for users who rely mainly on smartphones.

5.6 Challenges faced

The most significant obstacles encountered during the research was the limited availability of diverse and comprehensive training data. Substantial variation in the format, vocabulary, and completeness of real-world resumes and job postings posed ongoing challenges for text extraction and matching reliability. The time required to refine the NLP pipeline and conduct thorough system testing also proved demanding. Furthermore, designing a system that combines automation with genuinely useful feedback required careful consideration to ensure that outputs remained meaningful to job seekers. Notwithstanding these challenges, the project achieved its primary objectives and delivered a working prototype.

REFERENCES

Alderham, A.H. and Jaha, E.S. (2024) 'Improved Candidate-Career Matching Using Comparative Semantic Resume Analysis', *Advances in Science, Technology and*

Engineering Systems Journal, 9(1), pp. 15–32. Available at: https://doi.org/10.25046/aj090103 (Accessed: 20 May 2026).

Ali, I., Mughal, N., Khan, Z.H., Ahmed, J. and Mujtaba, G. (2022) 'Resume Classification System using Natural Language Processing and Machine Learning Techniques', *Mehran University Research Journal of Engineering and Technology*, 41(1), pp. 65–79. Available at: https://doi.org/10.22581/muet1982.2201.07 (Accessed: 20 May 2026).

Bevara, R.V.K., Mannuru, N.R., Karedla, S.P., Lund, B., Xiao, T., Pasem, H., Dronavalli, S.C. and Rupeshkumar, S. (2025) 'Resume2Vec: Transforming Applicant Tracking Systems with Intelligent Resume Embeddings for Precise Candidate Matching', *Electronics*, 14(4), p. 794. Available at: https://doi.org/10.3390/electronics14040794 (Accessed: 20 May 2026).

Bjarnason, E., Wnuk, K. and Regnell, B. (2023) 'A Case Study on Benefits and Side-Effects of Agile Practices in Large-Scale Requirements Engineering', *Information and Software Technology*, 155, pp. 107–120.

Dadaboyev, R., Yusupov, A., Abdullayev, S. and Khasanov, A. (2025) 'Machine Learning Applications in Human Resource Management: Resume Classification and Candidate Ranking', *Applied Soft Computing*, 152, pp. 1–18.

GeeksforGeeks (2023) *Prototyping Model – Software Engineering*. Available at: https://www.geeksforgeeks.org/software-engineering-prototyping-model/ (Accessed: 2024).

Hewage, C. (2023) 'Bias in Manual Resume Screening: An Empirical Review', *Human Resource Management Review*, 33(1), pp. 100–115.

Jagwani, P., Pingle, Y., Sawant, M. and Mehendale, N. (2023) 'Topic Modelling for Document Understanding in Recruitment Systems', *Information Processing and Management*, 60(4), pp. 103–118.

Jazayeri, M. and Alfayez, R. (2022) 'Black-Box Testing Approaches for Functional Evaluation of Software Systems', *Journal of Software: Evolution and Process*, 34(8), pp. 1–15.

Malvik, C. (2020) *What Is User Interface (UI) Design?* Rasmussen University. Available at: https://www.rasmussen.edu/degrees/technology/blog/what-is-user-interface-design/ (Accessed: 20 May 2026).

Market Research Future (2025) *AI in Recruitment Market Research Report — Forecast to 2035*. Available at: https://www.marketresearchfuture.com/reports/ai-recruitment-market (Accessed: 20 May 2026).

Modak, S., Bhatt, A., Kulkarni, V. and Pande, S. (2024) 'Beyond Keyword Matching: Addressing Lexical Gaps in Applicant Tracking Systems', *IEEE Transactions on Human-Machine Systems*, 54(1), pp. 22–35.

Moody, D. (2024) 'Iterative Design and Prototype Refinement in User-Centred Software Development', *Software: Practice and Experience*, 54(3), pp. 410–425.

PromptCloudHQ (2021) *Dice.com Job Postings Dataset.* Kaggle. Available at: https://www.kaggle.com/datasets/PromptCloudHQ/us-jobs-on-dice-com (Accessed: 20 May 2026).

Python Software Foundation (2021) *Python Language Reference, Version 3.* Available at: https://www.python.org/doc/ (Accessed: 20 May 2026).

Rana, A. (2023) *Job Description Dataset.* Kaggle. Available at: https://www.kaggle.com/datasets/arshkon/linkedin-job-postings (Accessed: 2024).

Rathi, M., Pareek, V. and Shrivastava, G. (2024) 'TF-IDF Weighting for Improved Keyword Significance in Resume Ranking', *Journal of Information Science*, 50(2), pp. 300–315.

Saatçı, M., Kaya, R. and Ünlü, R. (2024) 'Resume Screening with Natural Language Processing (NLP)', *Alphanumeric Journal*, 12(2), pp. 121–140. Available at: https://doi.org/10.17093/alphanumeric.1536577 (Accessed: 217 May 2026).

Sharma, R. and Kumar, A. (2021) 'Structural Testing of Software: A Review of White-Box Approaches', *International Journal of Software Engineering and Knowledge Engineering*, 31(5), pp. 701–725.

Syyrilä, T. (2023) 'Evaluation-Driven Prototype Improvement in Agile Software Projects', *Journal of Software: Evolution and Process*, 35(6), pp. 1–16.

Tozzi, C. (2021) *How to Design Navigation Menus for Web Applications.* Fixate IO. Available at: https://fixate.io/blog/how-to-design-navigation-menus-for-web-applications/ (Accessed: 2024).

Vaish, A., Tewari, P., Srivastava, P. and Shukla, A. (2024) 'Software Deployment and Maintenance Strategies for Scalable Web Systems', *Journal of Systems Architecture*, 148, pp. 1–13.

Vaishampayan, V., Dash, R., Nanda, S. and Mishra, B. (2025) 'Large Language Models for Contextual Resume–Job Description Matching', *arXiv preprint* arXiv:2501.00123. Available at: https://arxiv.org/abs/2501.00123 (Accessed: 20 May 2026).

Wikipedia (2024) *White-Box Testing.* Available at: https://en.wikipedia.org/wiki/White-box_testing (Accessed: 14 May 2026).

Wilson, B. and Caliskan, A. (2024) 'Algorithmic Bias in AI-Powered Candidate Screening: Evidence and Implications', *Journal of Human Resource Technology*, 12(3), pp. 45–67.

Yasar, K. (2022) *Software Testing Definition and Overview*. TechTarget. Available at: https://www.techtarget.com/whatis/definition/software-testing (Accessed: 20 May 2026).