A Project Report on

Smart Application Tracking System: Utilizing Generative AI for Efficient Resume Matching

Submitted in partial fulfillment for award of

Bachelor of Technology

Degree

in

Computer Science and Engineering

By

V. Venkata Anitha (Y21ACS587) **Sk. Raheeman (L22ACS606)**

R. Venkata Siva Prasad (Y21ACS552) **U. Mukhesh Ramana (Y21ACS581)**

Under the guidance of

Dr. P. Pardhasaradhi

Professor

Department of Computer Science and Engineering

Bapatla Engineering College

(Autonomous)

(Affiliated to Acharya Nagarjuna University)

BAPATLA – 522 102, Andhra Pradesh, INDIA

2024-2025

Department of
Computer Science and Engineering

CERTIFICATE

This is to certify that the project report entitled **Smart Application Tracking System: Utilizing Generative AI for Efficient Resume Matching** that is being submitted by V. Venkata Anitha (Y21ACS587), Sk. Raheeman (L22ACS606), R. Venkata Siva Prasad (Y21ACS552), U. Mukhesh Ramana (Y21ACS581) in partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science & Engineering to the Acharya Nagarjuna University is a record of bonafide work carried out by them under our guidance and supervision.

Date:

Signature of the Guide
Dr. P. Pardhasaradhi
Professor

Signature of the HOD
Dr. M. Rajesh Babu
Assoc. Prof. & Head

DECLARATION

We declare that this project work is composed by ourselves, that the work contained herein is our own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

V. Venkata Anitha (Y21ACS587)

SK. Raheeman (L22ACS606)

R. Venkata Siva Prasad (Y21ACS552)

U. Mukhesh Ramana (Y21ACS581)

Acknowledgement

We sincerely thank the following distinguished personalities who have given their advice and support for successful completion of the work.

We are deeply indebted to our most respected guide **Dr. P. Pardhasaradhi**, Professor, Department of CSE, for his valuable and inspiring guidance, comments, suggestions and encouragement.

We extend our sincere thanks to **Dr. M. Rajesh Babu**, Assoc. Prof. & Head of the Dept. for extending his cooperation and providing the required resources.

We would like to thank our beloved Principal **Dr.N.Rama Devi** for providing the online resources and other facilities to carry out this work.

We would like to express our sincere thanks to our project coordinator **Dr. P. Pardhasaradhi,** Prof. Dept. of CSE for his helpful suggestions in presenting this document.

We extend our sincere thanks to all other teaching faculty and non-teaching staff of the department, who helped directly or indirectly for their cooperation and encouragement.

V. Venkata Anitha (Y21ACS587)

SK. Raheeman (L22ACS606)

R. Venkata Siva Prasad (Y21ACS552)

U. Mukhesh Ramana (Y21ACS581)

Table of Contents

List of Figures

List of Tables

Abstract

The Smart Application Tracking System is a web-based application built on Streamlit that leverages generative AI to automate and enhance the resume matching process. This system extracts textual information from PDF-format resumes and compares them against detailed job descriptions. Utilizing advanced natural language processing and text mining techniques, the application calculates an ATS match score by evaluating keyword presence, experience relevance, and skills alignment. The system not only streamlines the recruitment process by reducing the need for manual resume screening but also provides actionable recommendations for both job seekers and HR professionals. By integrating Google's Generative AI with a robust analytical framework, the Smart Application Tracking System empowers users with data-driven insights to improve candidate selection and optimize career development.

Keywords:

Generative AI, Streamlit, PDF Extraction, Natural Language Processing, Text Mining, Resume Matching, ATS Score.

1 Introduction

In today's competitive job market, both job seekers and HR professionals face significant challenges in bridging the gap between candidate potential and job requirements. Many talented individuals struggle to secure high-quality positions due to resumes that fail to capture their true potential or align with specific job criteria. On the other side, HR departments often grapple with the overwhelming task of manually sifting through hundreds of resumes, which can lead to missed opportunities for both employers and candidates.

Our project, the Smart Application Tracking System, addresses these challenges by harnessing the power of generative AI combined with natural language processing and text mining. This innovative system automates the resume screening process by extracting key information from PDF-format resumes and comparing it against detailed job descriptions. It calculates an ATS match score based on critical factors such as keyword presence, experience relevance, and skills alignment, thereby providing a more objective and efficient method of candidate evaluation.

The integration of generative AI enables the system not only to score resumes but also to generate actionable insights and recommendations for improvement. This dual functionality benefits job seekers by guiding them on how to enhance their resumes, while also supporting HR professionals by streamlining the recruitment process and ensuring that top candidates are identified with greater accuracy.

By leveraging advanced algorithms and a user-friendly interface built on Streamlit, the Smart Application Tracking System transforms the traditional recruitment process into a more dynamic, data-driven experience. This system is designed to empower candidates to present themselves in the best possible light and to

assist HR teams in making informed hiring decisions, ultimately contributing to a more effective match between job opportunities and candidate skills.

2 Problem Definition

In today's dynamic job market, both job seekers and employers face significant challenges in effectively aligning candidate qualifications with job requirements. Job seekers often struggle to craft resumes that accurately reflect their skills and experiences in a way that resonates with modern Applicant Tracking Systems (ATS). As a result, even highly qualified candidates may be overlooked due to misalignment between their resume content and the specific job criteria.

For employers, the manual screening of large volumes of resumes is time-consuming and prone to human error. Traditional ATS solutions, while helpful, often rely on basic keyword matching and do not account for the nuanced details of a candidate’s background. This leads to inefficiencies where potentially suitable candidates are either missed or inaccurately ranked, resulting in a prolonged hiring process and increased recruitment costs.

Furthermore, the feedback provided to job seekers is typically generic, offering little insight into how their resume could be improved to better meet the needs of modern recruitment systems. This lack of personalized guidance not only hampers the candidate’s ability to enhance their applications but also contributes to a cycle of suboptimal job matching.

The Smart Application Tracking System addresses these challenges by integrating generative AI with advanced natural language processing and text mining techniques. By automating resume analysis and providing detailed, personalized recommendations, the system aims to improve candidate screening accuracy and empower job seekers to present their qualifications effectively.

3 Literature Survey

The employment process is being revolutionized in the modern world by the emergence of technology like the Internet. More than fifty thousand Internet employment portals invite candidates to upload their resumes. The likelihood of your application being chosen among thousands of others is quite low. Using this clever resume analyzer could significantly increase your chances of being added to the candidate list. It provides advice on resume writing techniques.

The CV RDF Ontology was created by John G. Breslin and Uldis Bojars. It is a model of the CV that utilizes the RDF data model. The Resume RDF describes resume-related information using a wide range of classes and attributes. To better explain the material, Uldis Bojars supplemented the FOAF with additional details. In 2002 and 2003, Turney and Littmann considered the semantic orientation or evaluation features of words from a corpus of 100 billion words. They provided a schema for guessing and also considered the semantic relevance of other words identified as a paradigm.

Using the concept of linked data, Ujjal Marjit et al. [1] proposed an additional method for recovering CV information. This makes it possible for the Internet to find and exchange many kinds of information.

Zhi Xiang Jing et al. created an online Chinese resume parser [2] that used a rule-based statistical algorithm to extract data from the CV. By working on block analysis of CV documents based on pattern matching and multi-level information identification, Zhang Chuang et al. [3] created the largest CV analysis system.

The following system was designed by Elik et al. [4] to translate CV to the ontological framework. Turkish and Turkish analysis English CV simplified in a model.

Di Wu and associates [5] Resultant Improved resume information extraction with WordNet Calculation of Similarity and the ontology concept.

Top resume analyzed using techniques such as natural language processing. With the help of natural language processing, only the text data is extracted from the CV and the strength of the applicant profile is displayed as a percentage. Additional attributes such as B. Percentage of candidate skills as a result of a resume review, depending on the candidate's education, qualifications, courses, and work experience. However, there is no provision on the website that it applies only to job seekers. Posting a particular job may give recruiters within your organization the opportunity to provide a ranking of candidates that match your immediate job-related skills. Therefore, there are many other such web applications, most of which provide similar functionality.

K. Sri Surya, Reguri Sharanya, Afrah Zilani, and Dr. CH. Niranjan [1] presented a Smart Applicant Tracking System using Gen AI, integrating generative AI with traditional ATS methods to enhance candidate screening. Their work builds on previous studies, including the online Chinese resume parser by Zhi Xiang Jing et al.

The Cluster-based Ranking Index (CBR), which evaluates resumes to identify the top candidates, was introduced by Mayuri Verma as a way to enhance the hiring process. [6] The development of a recommender system received very little attention in the past. Nonetheless, a great deal of study on recommender systems has been identified. The first recommendation system was created by Resnick et al. [7].
Otaibi et al. examined the use of recruitment services, outlined the steps that should be taken in each organization's hiring process, discussed the advantages of using electronic recruitment portals, what factors should be considered when selecting candidates, and a host of other crucial hiring strategies. A text mining study employing Natural

Language Processing (NLP) and clustering was presented by R. Janani et al. At the extraction point, text mining methods were used to boost efficiency. A study titled "Toward an Ontology-Based Information Extraction System that Matches Work with CV" was published by D. Cerrick and colleagues. In this case, the Ontology Knowledge Base transforms a resume written in plain text into an Ontology form (OKB). The completion rate is determined by this system based on factors including education and work experience. Marinovsky and others to use an Expectation-Maximization (EM) algorithm to draft an employment proposal that takes into account the job description provided by the company as well as the candidate's resume. Golecetal suggests using the fuzzy approach to assess a candidate's job description relevancy. The rapidly expanding discipline of text mining, which is described as statistical machine learning how to interpret unstructured raw materials, is also explained by text analysis.

For additional classification, classification, adequate training, and training, you can convert data into structured data. Excellent information for functionality [8]. technology. Text mining, also known as Natural Language Processing (NLP), involves pattern identification, information extraction, data mining, and parsing. Even if there are several other websites for resume analysis, our website is unique in its way. In our we have two interfaces one is the admin interface and the other is the user interface. In the admin interface, you view the candidates who have used our website in an excel. Recruiters can put the user's resume on the candidate list based on their resume score and the skills required for a particular position. In this way, it helps the recruiters. On the other hand, you have a user interface where you can upload your resume and get the resume score and the recommendations required to boost your resume score.

In summary, while previous research has contributed significantly to the fields of resume parsing and candidate matching, the integration of generative AI represents a

promising frontier. By leveraging these modern techniques, the Smart Application Tracking System offers a more nuanced and effective solution to the challenges faced by both job seekers and employers in today's competitive job market.

4 System Analysis

System analysis is a critical phase in the software development lifecycle where the requirements, constraints, and objectives of a project are carefully examined and documented.

4.1 Objective

The purpose of this project is to develop an AI-powered solution to enhance the hiring process for both job seekers and recruiters. With many candidates struggling to get shortlisted due to ineffective resume optimization, this system aims to provide personalized recommendations based on their resumes. It analyzes resumes against job descriptions, identifies missing keywords, suggests necessary skills and certifications, and improves ATS compatibility. By offering real-time feedback and automated resume enhancements, the system increases candidates' chances of securing job opportunities in reputable companies while assisting recruiters in efficiently shortlisting the most suitable applicants.

4.2 Existing System

The existing system for resume screening consists of several methods, each with its own limitations. Manual resume screening is a traditional approach where recruiters evaluate resumes by reviewing qualifications, experience, and skills. However, this method is highly time-consuming, prone to bias, and inconsistent in evaluation. Keyword-based ATS screening is another common approach used by applicant tracking systems like Taleo and Workday, which match resumes with predefined keywords. While this method automates filtering, it fails to understand context, skill synonyms, and career gaps, leading to inaccurate shortlisting. Additionally, online resume score checkers

such as Enhancv and Resume Worded provide AI-driven feedback but come with significant drawbacks. Enhancv offers AI-based resume insights but is expensive for full features, while Resume Worded provides template-based feedback that is often generic and not role-specific. Furthermore, many of these tools require user logins and do not compare resumes against job descriptions, making them less effective in real-world hiring scenarios. The limitations of these existing systems highlight the need for a smarter, AI-powered resume analysis tool that offers contextual understanding, personalized recommendations, and improved shortlisting accuracy.

4.3 Limitations of Existing Methods

a. The Traditional resume screening methods, including manual screening and keyword-based ATS systems, rely on rigid criteria and lack contextual understanding, often leading to inaccurate candidate shortlisting.

b. Most ATS systems only filter resumes based on predefined keywords and do not provide personalized recommendations to improve the resume or increase job matching accuracy.

c. Online resume score checkers offer generic, template-based feedback and do not analyze resumes against specific job descriptions, making them less effective for job seekers looking for tailored improvements.

d. Many AI-powered resume tools require user logins and paid subscriptions, limiting accessibility for job seekers who need free and effective resume evaluation solutions.

e. These are the few drawbacks of the existing papers and making it difficult for recruiters to identify unique talent among applicants.

4.4 Proposed System

A significant number of Python modules are used in Smart Application Tracking operation, which facilitates the application's management and functionality. Streamlit, PyPdf2, OS , Flask, google.generativeAI, are a few of the noteworthy modules. The roles of the various components are as follows: Streamlit is more structured and simplicity-focused, and it is used to quickly construct online apps for data science and machine learning. Thus, the smart application tracking system uses Streamlit.

The main purpose of building this project is to create a smart technology for the corporate hiring world. Nowadays thousands of people are unemployed and facing problems in finding jobs. We aim to provide a smart system that can give the perfect recommendation based on the resume of the user, The Smart Application Tracking System is designed to revolutionize the hiring process by leveraging AI-powered resume analysis to enhance the job application experience for both job seekers and recruiters. The system efficiently compares a candidate’s resume with a job description, providing a match percentage score and suggestions for missing keywords, ensuring a better alignment with job requirements.

This system integrates Google's Gemini AI, NLP (Natural Language Processing), and text mining techniques to analyze resumes, extract relevant data, and generate personalized feedback. Unlike traditional ATS software that merely filters resumes based on predefined keywords, our system provides intelligent recommendations, helping job seekers improve their resumes and increase their chances of getting shortlisted.

The web application, built using Flask and Streamlit, features a user-friendly interface where users can upload their resumes in PDF format. The system then

performs resume parsing, keyword extraction, and job description comparison to identify gaps and suggest improvements. Additionally, it includes an automated resume enhancement feature that updates the resume to achieve at least an 80% match with the job description. Users also have the option to download the updated resume for submission.

For recruiters, the system provides an HR candidate evaluation dashboard, which visually represents resume scores and applicant rankings based on qualifications, experience, and skill match. The system also enables recruiters to efficiently sort and shortlist candidates, thereby reducing manual screening time and hiring bias.

By implementing this AI-driven Smart Application Tracking System, we aim to streamline the hiring process, improve resume effectiveness, and enhance job application success rates for candidates in a highly competitive job market.

For example, if the recruiter receives hundreds of applications then manually checking them could be a very cumbersome task through a ATS the process could be done smoothly.

To identify most qualified candidates for a particular position for any job opening. To shorten recruitment time and reduce bias during the selection process, with a view to ranking candidates based on various aspects of their resume. Recruiters can efficiently identify the best talent for their job openings. This project is a step forward in making the job application process smarter, faster, and more effective.

4.4.1 Features of Proposed System

The Smart Application Tracking System aims to evolve continuously to meet the needs of job seekers and HR professionals. Future enhancements will include

advanced machine learning algorithms for deeper resume analysis and real-time feedback mechanisms that guide users in optimizing their resumes. The system will provide personalized career path recommendations based on user profiles and industry trends, and it will integrate with popular job portals for seamless application processes. An AI-powered interview preparation module will simulate interview scenarios, while an advanced analytics dashboard in the HR portal will offer insights into recruitment metrics. Enhanced privacy features will allow users to manage their data preferences, and a mobile version of the application will increase accessibility. Additionally, community and networking features will enable users to connect with industry professionals, and a feedback loop will ensure continuous improvement of the system's recommendations and user experience.

4.4.2 Feasibility Study

The feasibility analysis begins by defining the project goals and generating possible solutions to provide an indication of the new system’s potential. This phase requires creativity and imagination to explore new ways of doing things and generate ideas.

4.4.3 Technical Feasibility

The application is developed as a web application and requires specific tools and technologies. The main technologies used are generative AI, natural language processing (NLP), implemented in Python programming language. Adobe XD is used for design purposes, and Star UML is used for creating diagrams. The Visual Studio IDE is utilized for development.

4.4.4 Financial Feasibility

The application is freely available for all users, without any charges for usage. There are no monetary services associated with the application, allowing every user to access it freely.

4.4.5 Operational Feasibility

Operational feasibility measures how well the proposed system solves problems and takes advantage of opportunities identified during scope definition. The Smart Application Tracking System effectively addresses identified challenges and meets all requirements. Core features like resume-job description analysis, keyword suggestions, ATS scoring, and HR evaluation function seamlessly with Google's Gemini AI integration.

Admins can manage user data, while users can upload, update, and remove resumes, ensuring control and privacy. The automated resume enhancement feature optimizes match scores, improving job shortlisting chances. The system operates efficiently, making it a feasible solution for job application optimization.

4.5 Hardware Requirements

The following are the system requirements to develop Smart Application Tracking System.

a. Processor: Intel Core i5

b. Hard Disk: Minimum 100GB

c. RAM: Minimum 8GB

d. Internet Connection: Stable connection (10 Mbps)

4.6 Software Requirements

The following are the software used in the development of the app.

a. Operating System: Windows 10/11, macOS, or Linux (Ubuntu 20.04+)

4.6.1 Python Programming Language

Python is advanced, higher level and easy to learn programming language, we have chosen Python to develop Desktop GUI, We will create a desktop application using Python. In python everything is related to the Image processing is very easy to implement. The documentation and community of the python is very big, so we will get the help in development from the community.

4.6.2 Flask

Flask is a lightweight, Python-based web framework known for its simplicity, flexibility, and scalability. We chose Flask for developing our Smart Application Tracking System as it provides a minimal yet powerful structure for building web applications. With built-in support for routing, request handling, and template rendering, Flask allows efficient integration with AI models, databases, and third-party APIs. Its extensive documentation and strong community support make development smoother, ensuring we can easily troubleshoot and enhance the system.

4.6.3 Streamlit

Streamlit is a lightweight, Python-based framework designed for building interactive web applications with minimal code. We chose Streamlit for our Smart Application Tracking System as it allows for quick prototyping and seamless UI integration. With its simple syntax and real-time updates, Streamlit makes it easy to

display AI-driven insights, user inputs, and analytical results. Its strong community support and extensive documentation ensure efficient development and troubleshooting.

4.6.4 GitHub

It is free and open source backup platform for your running project. It is maintaining by GIT. It's very easy to upload your work on free. There is no storage limitation. You can do development in group and separately also. It is very easy to maintain the Information Technology work with other teammates also.

4.6.5 Visual Studio

Visual Studio Code, commonly referred to as VS Code, is a renowned and widely used source code editor developed by Microsoft. It stands out for its versatility, lightweight design, and rich ecosystem of extensions. Available across Windows, macOS, and Linux, VS Code's cross-platform compatibility has made it a favored choice among developers from various backgrounds. Notably, its lightweight nature ensures swift startup times and efficient use of system resources, even on less powerful machines.

What sets VS Code apart is its extensibility. It boasts a vast collection of extensions, enabling developers to tailor the editor to their specific needs. These extensions span different programming languages, frameworks, and tools, enhancing functionality and personalizing the coding environment. Despite being a code editor, VS Code offers integrated development environment (IDE) features, including code navigation, debugging, version control integration, and an integrated terminal.

Intelligent code editing features such as IntelliSense (code completion), code

refactoring, and real-time error checking assist developers in writing clean, error-free code. The built-in Git support simplifies version control management and fosters collaboration. It also provides debugging capabilities for multiple programming languages, allowing users to set breakpoints, inspect variables, and debug their code efficiently.

VS Code's integrated terminal is a time-saving feature that enables running commands and scripts without the need to switch to a separate terminal window. Its multi-language support, encompassing syntax highlighting, code formatting, caters to a diverse range of developers. Furthermore, it offers extensive customization options, including a variety of themes and appearance adjustments. With an active and supportive community, numerous resources, tutorials, and forums are readily available to assist users with questions and challenges.

Importantly, Visual Studio Code is both free and open-source, making it accessible to developers of all levels. Its lightweight design and powerful feature set have contributed to its widespread adoption, earning it a strong reputation in the coding community as a flexible and highly capable code editor.

5 System Requirement Study

A system requirement study, also known as a requirements analysis or requirements elicitation, is a crucial phase in the software development lifecycle where the needs and expectations of stakeholders are gathered, documented, and analyzed to define the scope and objectives of a project.

5.1 User Characteristics

In our system, there will be two types of users

a. Admin: - In this system, the Admin (HR/Recruiter) can access the HR Portal by entering a secure password. The admin can upload multiple candidate resumes along with a job description, view the ATS match score for each resume, see missing keywords, shortlist the best candidates, and download a detailed PDF report of the shortlisted candidates.
b. Normal User: - Normal users (Job Seekers) can access the web application to upload their resume and a job description. The system analyzes the resume, provides an ATS match score, highlights missing keywords, profile summary and gives improvement suggestions.

5.2 System Main Module

It typically acts as the entry point for the application, coordinating the interaction between various modules, components, and layers within the system.

a. Resume Upload and Parsing: – Users (job seekers or HR) can upload resumes in PDF format. These resumes are then parsed using text extraction techniques to retrieve relevant content such as skills, experience, education, and more for further analysis.

b. Job Description Input: – Users can either upload or type in a job description. This job description acts as the base against which the resume will be analyzed to determine compatibility.

c. ATS Score Matching: – Once the resume and job description are processed, this module uses Google's Gemini AI to analyze both and generate an ATS (Application Tracking System) match score. This score indicates how well the resume aligns with the job description.

d. Keyword Extraction and Suggestions: – The system identifies missing keywords in the resume when compared to the job description. It then provides keyword suggestions, profile summary and improvement tips to increase the resume's match score.

e. AI Resume Improvement: – This module offers users the ability to automatically enhance their resume by integrating the missing keywords and restructuring content to achieve a match score of at least 80%.

f. Candidate Shortlisting and Reporting (Admin Module): – HR/Admin users can upload multiple resumes with a single job description, view individual match scores and missing keywords, and shortlist the most suitable Candidates. A final PDF report of shortlisted candidates can also be generated and downloaded.

5.3 Functional Requirement

a. User needs to visit the Smart Application Tracking System web application.
b. User uploads their resume (PDF format) into the system.
c. User provides or uploads the job description for comparison.

d. The system extracts text content from the uploaded resume and job description.
e. The system uses Google's Gemini AI to analyze and compare the resume with the job description.
f. The system calculates and displays the ATS match score indicating the compatibility between the resume and the job role.
g. The system identifies and lists the missing keywords from the resume based on the job description.
h. The system suggests improvements and relevant keywords and profile summary to enhance the resume.
i. HR/Admin users can upload multiple resumes and a single job description to evaluate and shortlist the best candidates.
j. The system generates a downloadable PDF report of shortlisted candidates with match scores.

5.4 Non-Functional Requirement

a. Accessibility: This can be used by any one because it is basic in the structure and usage.
b. Efficiency: This project efficiency as a first trial is not great by numbers but with more learning and hard work the app can be perfect and very efficient.
c. Scalability: This is now only on computers, so it requires a good internet connection to work properly, still it will take a time to do pre-processing.
d. Security: We are not providing any Login/Registration for user, still user can upload the resume, resume will be deleted from the system, Still user data will be secured with us for future training purpose.

5.5 Project Planning

We have built this project planning in Jira.

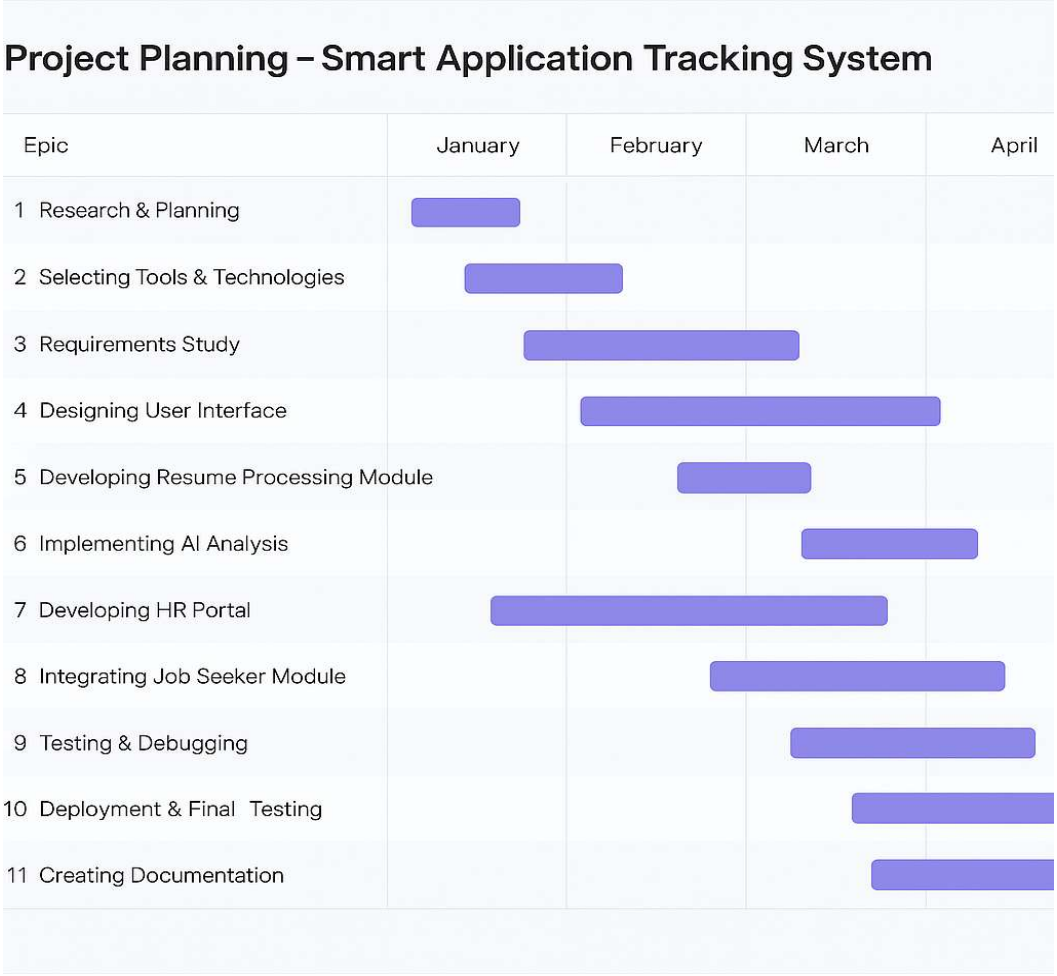

Figure 5.1 Project Planning

6 Methodology

In this chapter we are going to learn about working of Smart Application Tracking System.

6.1 Working of Smart Application Tracking System

1. We are going to see how actually our system is working behind, we have divided our work in separate tasks, let's understand each steps of it.

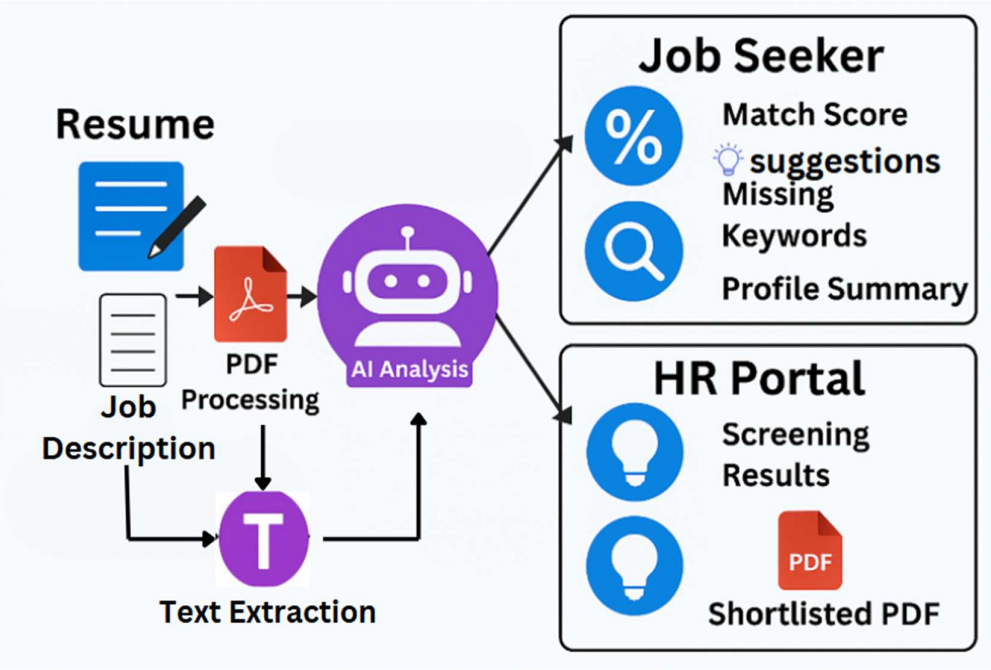

Figure 6.1 Working of Smart Application Tracking System

6.1.1 PDF Extracting

PDF Extracting is a module that automatically retrieves the user's resume, provided that the resume is in PDF format. This module extracts the user's data from the resume.

6.1.2 Text Extracting

Text Extracting is a module that fetches the text information from the resume and job description. This text data is used for language processing in further tasks, such as recommendations and fetching the user's personal information.

6.1.3 AI-Powered Analysis

The core functionality of the system lies in this phase, where AI and NLP algorithms analyze the extracted text. In the Job Seeker interface, the system computes the match score between the resume and the job description, helping the user understand how well their resume aligns with the targeted job. It highlights missing keywords that may be important for passing an ATS, generates a comprehensive profile summary from the user's resume, and provides improvement suggestions to enhance resume relevance and readability. In the HR Portal, the system takes multiple resumes and screens them against a single job description. It ranks the resumes in hierarchical order based on the calculated match percentage and generates a downloadable PDF containing only the shortlisted candidates for easy access by HR professionals.

6.1.4 Resume Optimization

The system also helps job seekers improve their resumes through targeted optimization. It identifies missing or weak sections and suggests keywords or changes to increase the resume's match score. In many cases, the resume can be automatically enhanced to achieve at least an 80% compatibility score with the job description.

6.1.5 Insights and Analytics

A new For HR professionals, the system provides valuable insights by analyzing trends in candidate data, screening patterns, and matching scores. Visualizations such as charts and summaries help in interpreting the data efficiently, aiding in quicker and more informed recruitment decisions.

7 System Design

In this chapter, we will learn about the system designs, diagram and DFDs.

7.1 UML Diagram

UML is an acronym that stands for Unified Modeling Language. Simply put, UML is a modern approach to modeling and documenting software. It is based on diagrammatic representations of software components. As the old proverb says: “a picture is worth a thousand words”. By using visual representations, we are able to better understand possible flaws or errors in software or business processes.

Mainly, UML has been used as a general-purpose modeling language in the field of software engineering. However, it has now found its way into the documentation of several business processes or workflows. For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts. They provide both a more standardized way of modeling workflows as well as a wider range of features to improve readability and efficacy.

Any complex system is best understood by making some kind of diagrams or pictures. These diagrams have a better impact on our understanding. There are two broad categories of diagrams and they are again divided into subcategories

a. Structural Diagrams

b. Behavioral Diagrams

7.1.1 Structural Diagrams

The Structural diagrams represent the static aspect of the system. These static aspects represent those parts of a diagram, which forms the main structure and therefore stable. These static parts are represented by classes, interfaces, objects, components, and nodes. The four structural diagrams are

a. Class diagram
b. Object diagram
c. Component diagram
d. Deployment diagram

7.1.2 Behavioral Diagrams

Any System can have two aspects, static and dynamic. So, a model is considered as complete when both the aspects are fully covered. Behavioral diagrams basically capture the dynamic aspects of a system. UML has the following five types of behavioral diagrams.

a. Use case diagram
b. Sequence diagram
c. Collaboration diagram
d. Statechart diagram
e. Activity diagram

7.2 Use Case Diagram

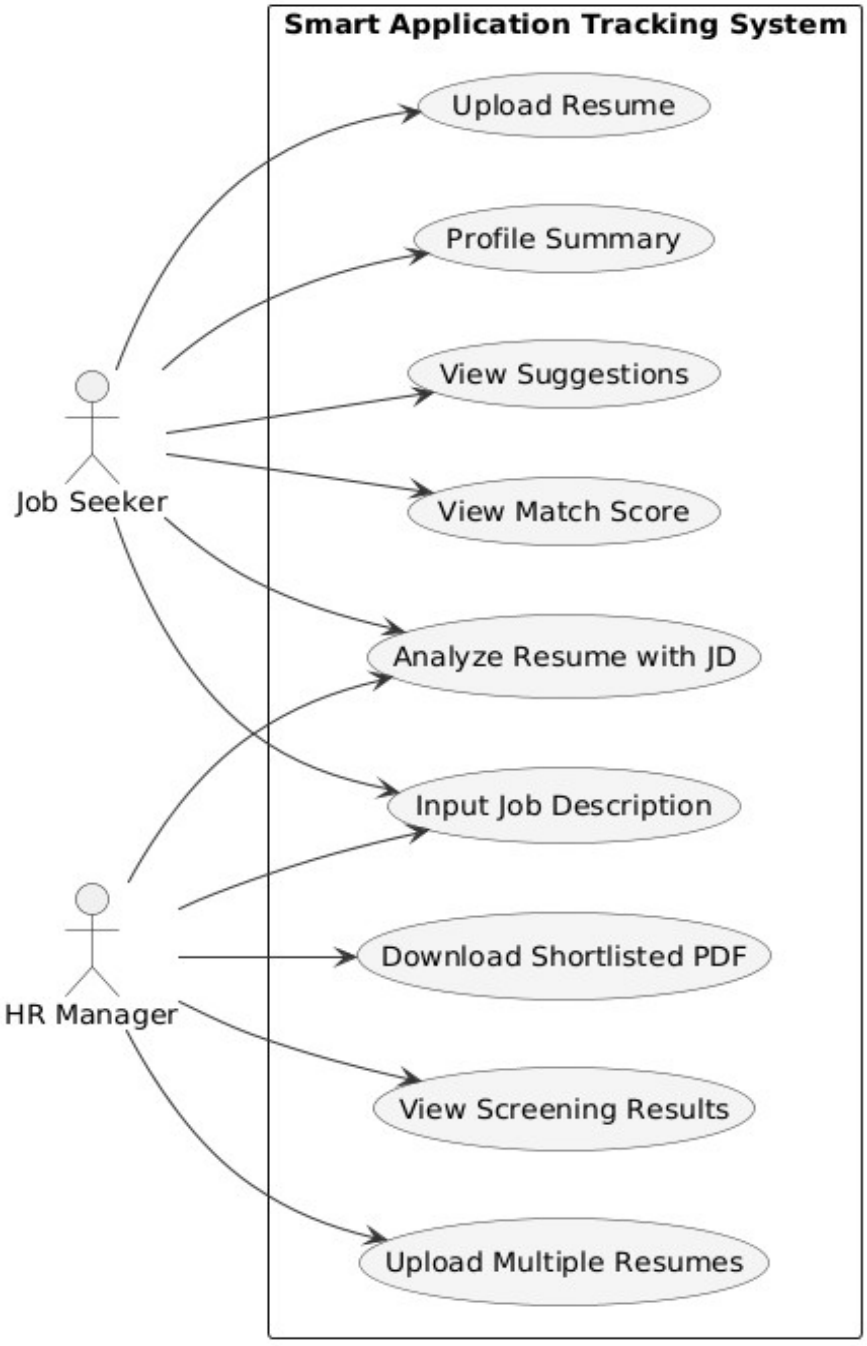

Figure 7.1 Use Case Diagram

a. In this diagram, we seen what the functionality Job Seeker and HR Manager can use. We can see job seeker and HR manager have the different access of the functionality.
b. A use case diagram at the simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.
c. While a use case itself might drill into a lot of detail about every possibility, a use-case diagram can help provide a high-level view of the system. It has been said before that "Use case diagrams are the blueprints for your system". They provide a simplified and graphical representation of what the system must actually do.
d. Due to their simplistic nature, use case diagrams can be a good communication tool for stakeholders. The drawings attempt to mimic the real world and provide a view for the stakeholder to understand how the system is going to be designed.
e. The purpose of the use case diagram is simply to provide the high-level view of the system and convey the requirements in laypeople's terms for the stakeholders.
f. Additional diagrams and documentation can be used to provide a complete functional and technical view of the system.
g. Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities,

use cases are prepared and actors are identified.

h. When the initial task is complete, use case diagrams are modelled to present the outside view. In brief, the purpose of use case diagrams can be said to be as follows –

i. Used to gather the requirements of a system.

ii. Used to get an outside view of a system.

iii. Identify the external and internal factors influencing the system.

iv. Shows the interactions among the requirements are actors.

7.3 Class Diagram

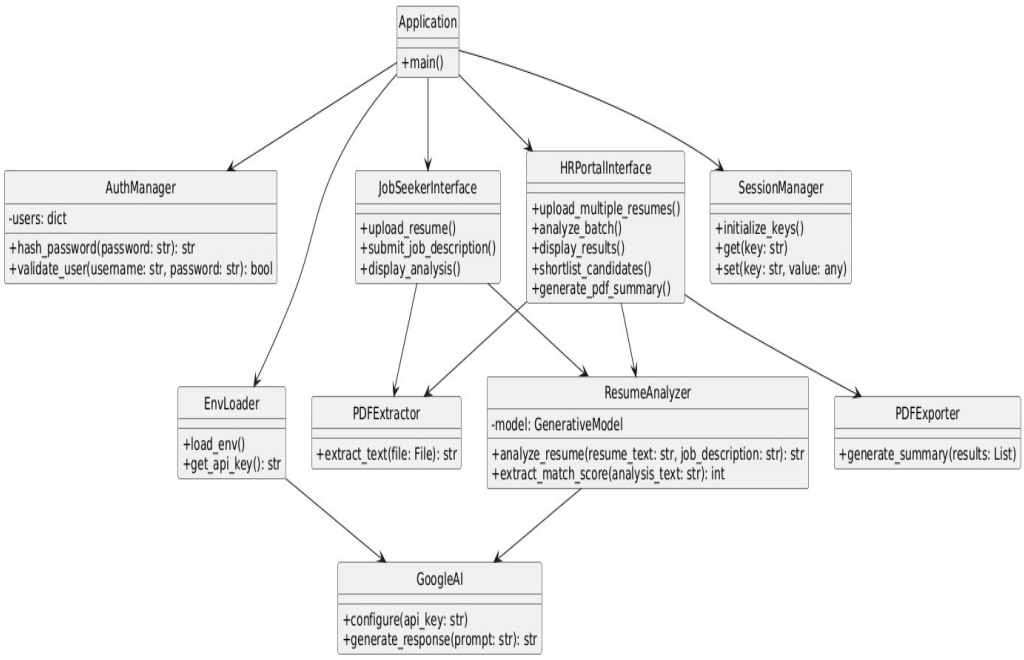

Figure 7.2 Class Diagram

a. In this diagram, we seen different classes and functions according to the functionality. We can see how system is associated with the user and admin functionality.
b. A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.
c. The Class Diagram is the main building block of object- oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling translating the models into programming code.
d. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.
e. In the diagram, classes are represented with boxes that contain three compartments:
    i. The top compartment contains the name of the class. It is printed in bold and centered, and the first letter is capitalized.
    ii. The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.
    iii. The bottom compartment contains the operations the class can execute. They are also left- aligned and the first letter is lowercase.

7.4 Sequence Diagram

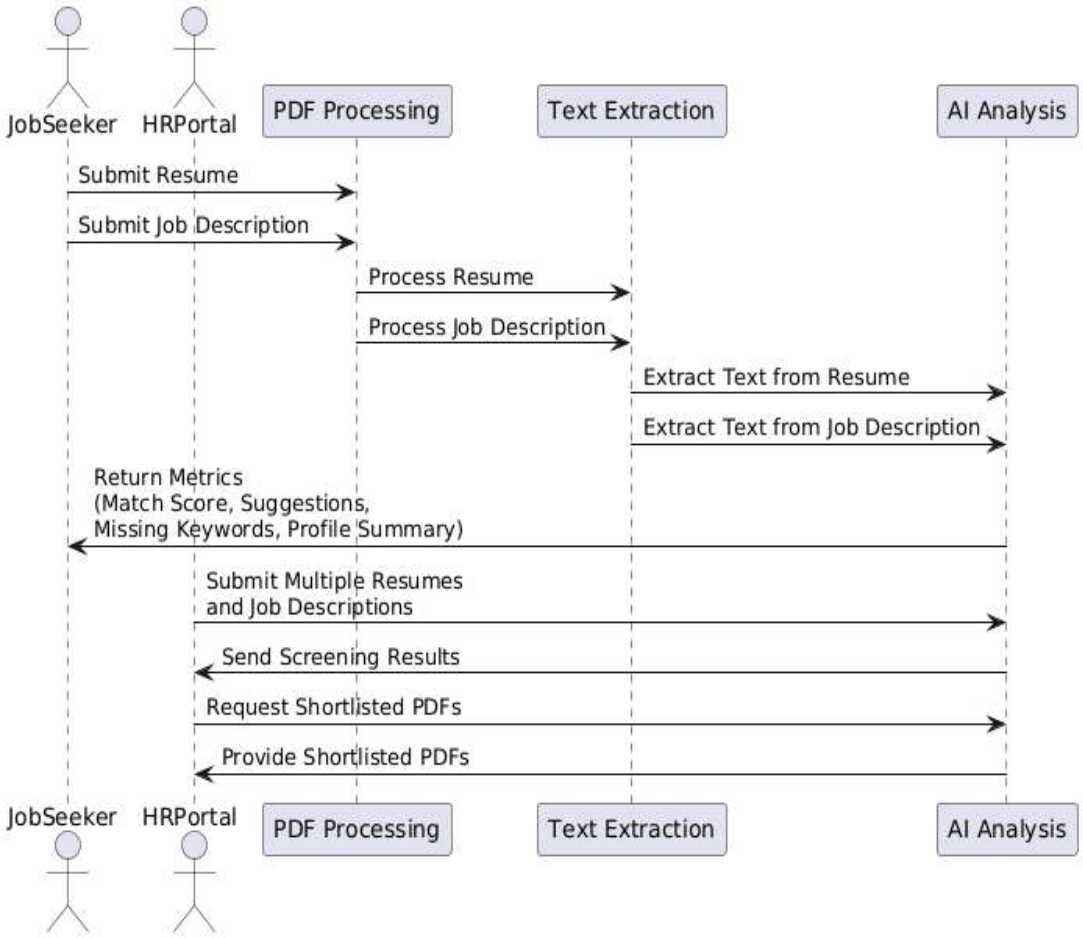

Figure 7.3 Sequence Diagram

a. In this diagram, we have seen the step-by-step procedure of Resume Processing. It is showing the sequence from Upload resume to get the Recommendations.

b. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

c. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development.

d. A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows. the messages exchanged between them, in the order in which they occur.

e. This allows the specification of simple runtime scenarios in a graphical manner.

f. Messages, written with horizontal arrows with the message name written above them, display interaction. Solid arrowheads represent synchronous calls, open arrowheads represent asynchronous messages, and dashed lines represent reply messages.

g. Asynchronous calls are present in multithreaded applications, event- driven applications and in message-oriented middleware.

h. Activation boxes, or method-call boxes, are opaque rectangles drawn on top of lifelines to represent that processes are being performed in response to the message.

7.5 Activity Diagram

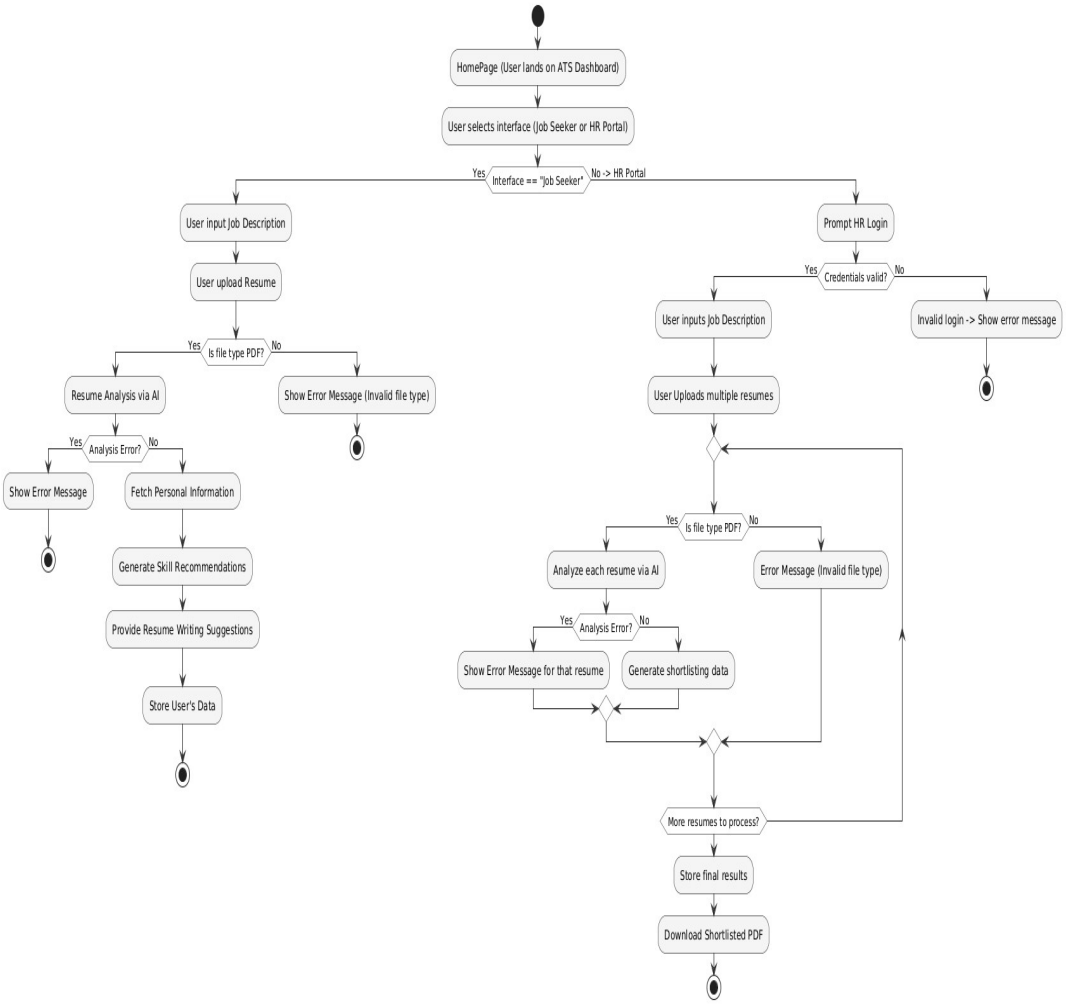

Figure 7.4 Activity Diagram

a. In this diagram, we can see that the activity of Smart Resume Analyzer, if everything works well, we will get the output, otherwise it will be error.

8 System Implementation

System implementation, in the context of a project, refers to the process of turning the design and specifications of the system into a working and operational reality. It involves translating the theoretical concepts and plans outlined in the project requirements and design phases into tangible software or hardware components.

8.1 Coding

The coding is the process of transforming the design of a system into a computer language format. This coding phase of software development is concerned with software translating design specification into the source code. It is necessary to write source code & internal documentation so that conformance of the code to its specification can be easily verified. Coding is done by the coder or programmers who are independent people than the designer. The goal is not to reduce the effort and cost of the coding phase, but to cut to the cost of a later stage.

8.2 Module Specification

In our system there will be two modules, Job Seeker & HR Portal. Admin will be only one. Let see what user & admin can do.

Table 8.1 Module Specification

Job Seeker	**HR Portal**
Upload the Resume and Job description	Do Login
Check the match score	Upload the resumes and job description
Check the missing keywords	Check the screening results
Check the profile summary and improvement suggestions	Check the downloaded pdf of screening results

8.3 Sample coding

The code which is mentioned below it is just main logic of Job seeker interface and HR Portal interface. This code is only just functional part, not the full part with the backend.

a. Job Seeker Interface: This code is shows the main logic of the job seeker interface and shows how it work.

```
def job_seeker_interface():
    st.title("Job Seeker Dashboard")  # Title for the Job Seeker dashboard
    st.subheader("Boost Your ATS Score")  # Subtitle for the dashboard

    # Session state initialization for storing analysis results
    session_defaults = {
        'analysis': None,
        'resume_file': None,
        'jd_text': ""
    }
    for key, val in session_defaults.items():
        if key not in st.session_state:  # Initialize session state if not already set
            st.session_state[key] = val

    # Input fields for job description and resume upload
    jd = st.text_area("Paste Job Description",
                      height=200,
                      placeholder="Enter job description here...",
                      value=st.session_state.jd_text)  # Text area for job description

    resume = st.file_uploader("Upload Resume (PDF only)",
                              type="pdf",
                              help="Max file size: 5MB")  # File uploader for resume

    if st.button("Analyze Resume"):  # Button to trigger analysis
        if resume and jd.strip():  # Check if both resume and job description are provided
            with st.spinner("Analyzing your resume..."):  # Show loading spinner
                text_content = extract_text_from_pdf(resume)  # Extract text from the upload
```

Figure 8.1 Job Seeker sample code

b. HR Portal Interface: This code is used for HR Portal interface, it will shows the functions of the HR Portal interface.

```
def hr_portal_interface():
    st.title("HR Dashboard")  # Title for the HR dashboard
    st.subheader("Candidate Evaluation System")  # Subtitle for the HR dashboard

    if 'hr_results' not in st.session_state:
        st.session_state.hr_results = []  # Initialize session state for HR results
    if 'analyzed_resumes' not in st.session_state:
        st.session_state.analyzed_resumes = {}  # Initialize cache for analyzed resumes

    jd = st.text_area("Job Description", height=200,
                      placeholder="Paste job requirements here...",
                      key="hr_jd")  # Text area for job description in HR portal

    uploaded_files = st.file_uploader("Upload Candidate Resumes",
                                      type="pdf",
                                      accept_multiple_files=True,
                                      help="Upload PDF resumes for screening")  # File uploader for mult

    if st.button("Start Screening"):  # Button to start screening resumes
        if not jd.strip() or not uploaded_files:  # Check if both JD and resumes are provided
            st.warning("Please provide both Job Description and Resumes")  # Notify user to provide in
            return

        st.session_state.hr_results = []  # Reset HR results for new screening
        progress_bar = st.progress(0)  # Initialize progress bar

        for idx, file in enumerate(uploaded_files):  # Loop through each uploaded file
            try:
```

Figure 8.2 HR Portal sample code

8.3.1 Code URL

The GitHub repository hosts code and data for smart application tracking system. This is about to analyze the resume of any person, based on resume and job description. Our smart system will give the match score, missing keywords, suggestions to the user. It is very easy to use and smart application, in this application user just need to upload the resume and JD. In HR Portal also used for screening the resumes and able to download the screening results as pdf format.

https://github.com/AnithaVarikallu/Smart-Application-Tracking-System.git

9 Testing

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. In simple words, testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

9.1 Purpose of Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests; each test type addresses a specific testing requirement. Testing and test design are parts of quality assurance that should also focus on bug prevention. A prevented bug is better than a detected and corrected bug. Testing consumes at least half of the time and work required to produce a functional program. History reveals that even well written programs still have 1-3 bugs per hundred statements. Testing is the process of executing a program with the aim of finding errors. To make our software perform well it should be error-free. If testing is done successfully, it will remove all the errors from the software.

9.2 Testing Strategies

In order to uncover the errors, present in different phases we have the concept of Level of testing. The Software testing has a prescribed order with the following list of software testing categories arranged in chronological order for our project testing

and to generate test cases for output. These are the steps taken to fully test new software in preparation for marketing it:

9.3 Types of testing

a. **Unit testing** performed on each module or block of code during development. Unit Testing is normally done by the programmer who writes the code.

b. **Integration testing** done before, during and after integration of a new module into the main software package. This involves testing of each individual code module. One piece of software can contain several modules which are often created by several different programmers. It is crucial to test each module's effect on the entire program model.

c. **System testing** done by a professional testing agent on the completed software product before it is introduced to the market.

d. **Acceptance testing** - beta testing of the product done by the actual end users.

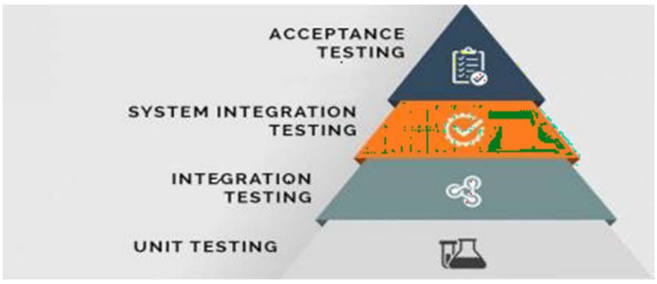

Figure 9.1 Types of Testing

9.3.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software i.e., the module. Using the detailed design and the process specifications testing is done to uncover errors within the boundary of the module. All modules must be successful in the unit test before the start of the integration testing begins. It has been seen that each activity class runs after its development using unit testing.

Unit testing is performed at the first stage of testing as it is performed first of all other testing processes. Unit testing is also known as white box testing. So it's generally performed by developers. In our Project Unit testing is commonly automated but may still be performed manually. Software Engineering does not favor one over the other but automation is preferred. A manual approach to unit testing may employ a step- by-step instructional document. The three Modules of Fake Face Detection GANs, Phase I & Phase II individually tested in terms of detection of defects. Used to test each one of those functions, behavior is tested.

Here the Unit Testing Techniques are mainly categorized into three parts which are Black box testing that involves testing of user interface along with input and output, White box testing that involves testing the functional behavior of the software application and Gray box testing that is used to execute test suites, test methods, test cases and performing risk analysis.

The Code coverage techniques used in our project under Unit Testing are listed below:

a. Statement Coverage
b. Decision Coverage
c. Branch Coverage

d. Condition Coverage

e. Finite State Machine Coverage

In this project, the unit testing was performed on each module separately to find any defects in the code. The testing performed was manual walkthrough in the code of each module whether the functional call was correct or not, whether the attributes of functions called were accurate or not. By unit testing we found that without proper execution of each individual block of code we cannot further execute the project. Each module should give an accurate result and then should be moved towards the next testing process. A piece of code cannot work if the file mentioned was not found in mentioned directory, by Unit Testing we can find such faults. Through unit testing of this project we found that each individual module or block of code was working without any faults.

9.3.2 Integration Testing

Integration testing is the testing where multiple modules are tested to verify if different pieces of the modules are working together as per expectation or not. In case of Integration testing multiple modules get integrated and are tested as a single module so testers focus more on integrated functionality rather on internal design of the application.

In our Project the Integration testing is performed after all modules get integrated are done with unit tested i.e. Integration testing is done after unit testing and before System testing.

Although each of our software module is unit tested, defects still exist for various reasons like below

1. A Module, understanding and programming logic may differ. To verify the software modules, work in unity.
2. At the time of module development, there are wide chances of change in requirements by the Users. These new requirements may not be unit tested.
3. Interfaces of the software modules with the dataset are sometimes erroneous.
4. External Hardware interfaces, if any, could be erroneous
5. Exception handling causes issues in execution sometimes.

Integration Test Case differs from other test cases in the sense it focuses mainly on the interfaces & flow of data/information between the modules. Here priority is to be given for the integrating links rather than the unit functions which are already tested.

Through integration testing we made sure each module was integrated properly and there were no errors in integrating the modules. The information was passed smoothly from one module to other modules without missing any data.

9.3.3 System Testing

System Testing is a level of testing that validates the complete and fully integrated software product. The purpose of a system test is to evaluate the end-to-end system specifications. Usually, the software is only one element of a larger computer-based system. Ultimately, the software is interfaced with other software/hardware systems. System Testing is actually a

series of different tests whose sole purpose is to exercise the full computer- based system.

In our project System Testing involves testing the software code for following

a. Testing the fully integrated modules including external peripherals in order to check how components interact with one another and with the system as a whole. This is to support an End-to-End testing scenario.
b. Verified with thorough testing of every input in individual modules to check for desired outputs.
c. Tested for user's experience and expectations with the integrated modules and System Application.

Here the entire software system is tested. The reference document for this process is a requirements document and the goal was to see if software meets its requirements.

Below we have listed types of system testing in our project typically used:

a. **Usability Testing-** it mainly focuses on the user's ease to use the application, flexibility in handling controls and ability of the system to meet its objectives of our project.
b. **Functional Testing –** to achieve functional completeness testing, Functional Testing involves trying to think of any possible missing functions. We might make a list of additional functionalities that a product could have to improve during functional testing.

9.4 Test Cases

The table highlights key test cases for the Smart Application Tracking System. It covers resume analysis, login, error handling, and UI validation. Each scenario shows expected output and confirms successful functionality. Overall, the system performs reliably across various user interactions.

Test Case	Input	Expected Output	Result
Valid Resume Analysis	PDF with matching job desc	85% match, suggestions	Pass
Missing Keywords	PDF lacking keywords	45% match, missing keywords	Pass
No Resume	Only job desc	Prompt for resume	Pass
Invalid PDF	Non-PDF file	File type error	Pass
Valid Multiple Resumes	Multiple resumes + job desc	Sorted screening results	Pass
Shortlist Resumes	Select top N resumes	Downloadable shortlisted PDF	Pass
No Resumes Uploaded	Only job desc	Prompt for resumes	Pass
Successful Login	Valid creds	Login success	Pass
Invalid Credentials	Invalid creds	Login error	Pass
API Error Handling	Simulate 429 error	Quota warning	Pass
Non-Readable PDF	Image-only PDF	Text extract fail	Pass
CSS Rendering	App load	Styled UI	Pass
Match % Extraction	Text with % values	Extract % correctly	Pass

Figure 9.2 Test Case Diagram

10 **Result Screenshots**

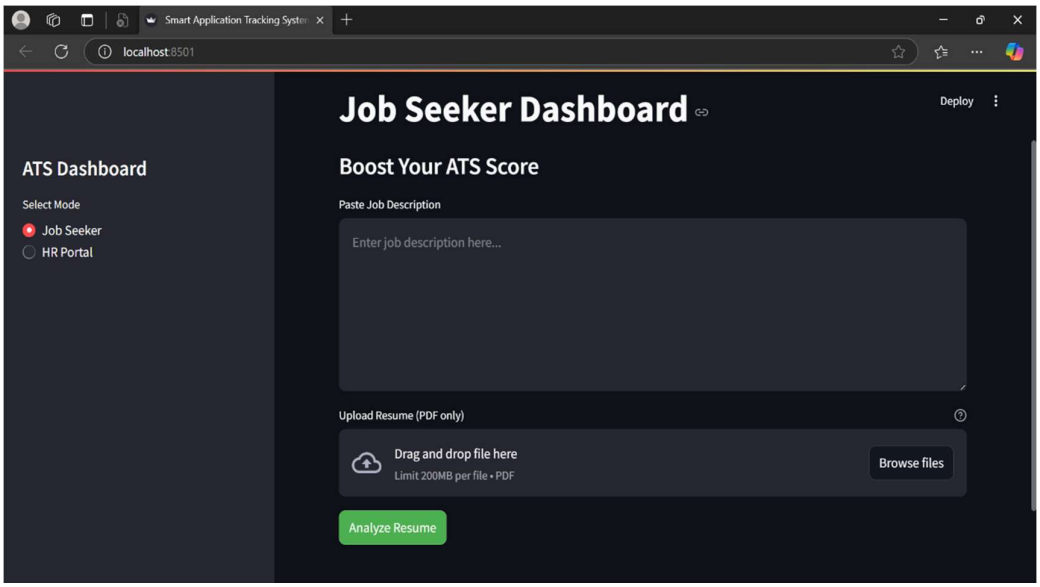

Figure 10.1 Landing Web Page of the Smart Application Tracking System

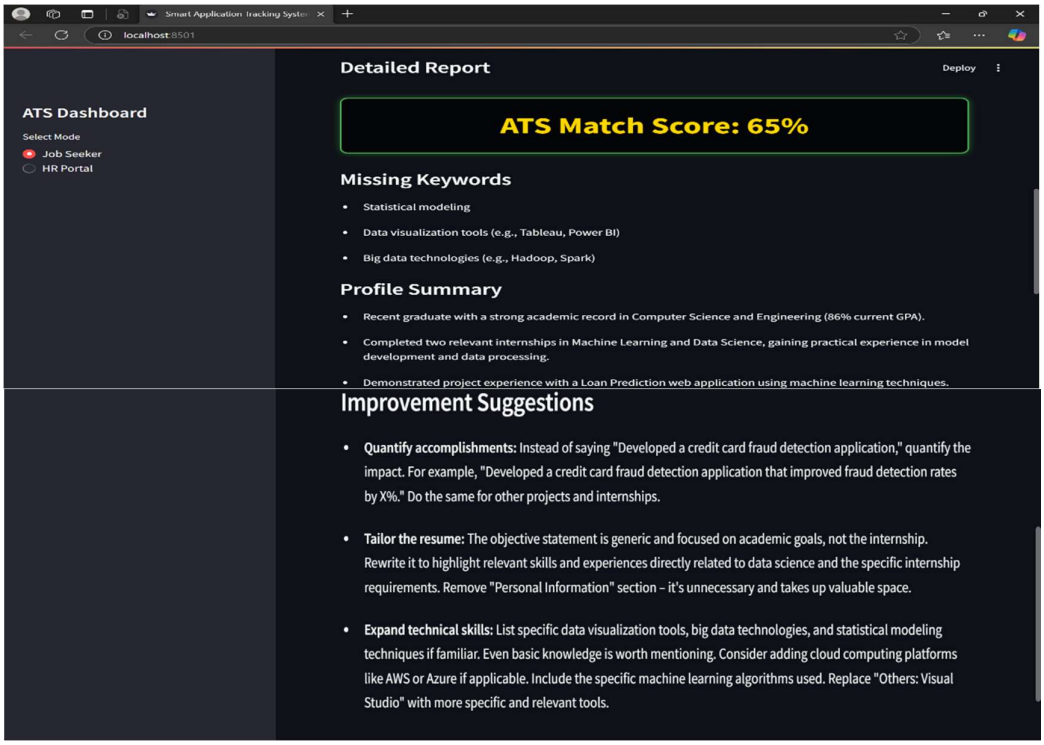

Figure 10.2 Web page for Results of resume after selecting Job Seeker Mode

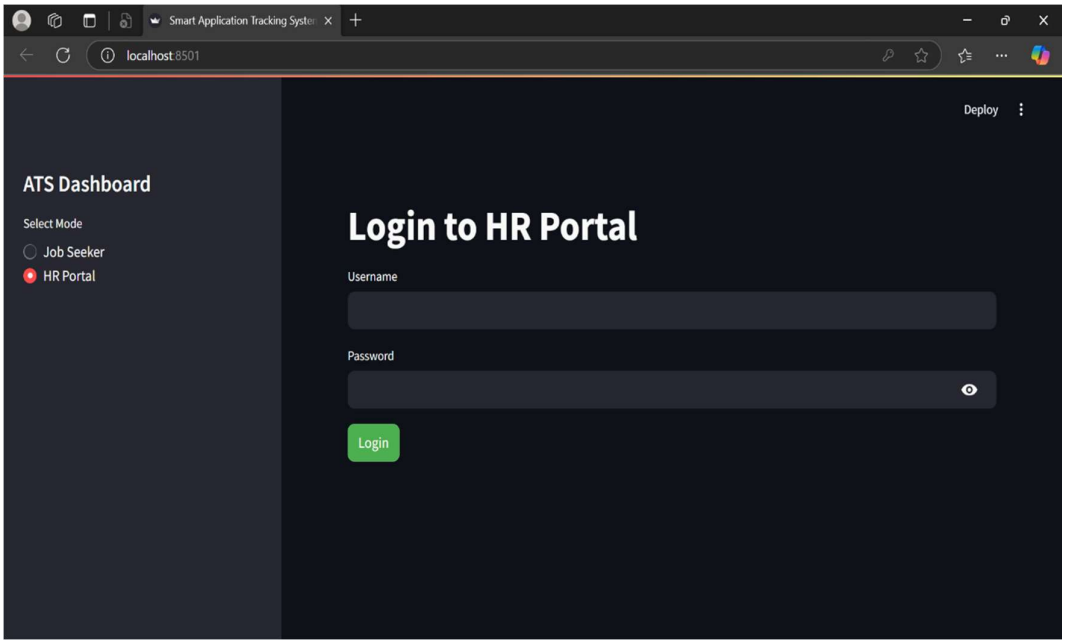

Figure 10.3 Login page of HR Portal Mode

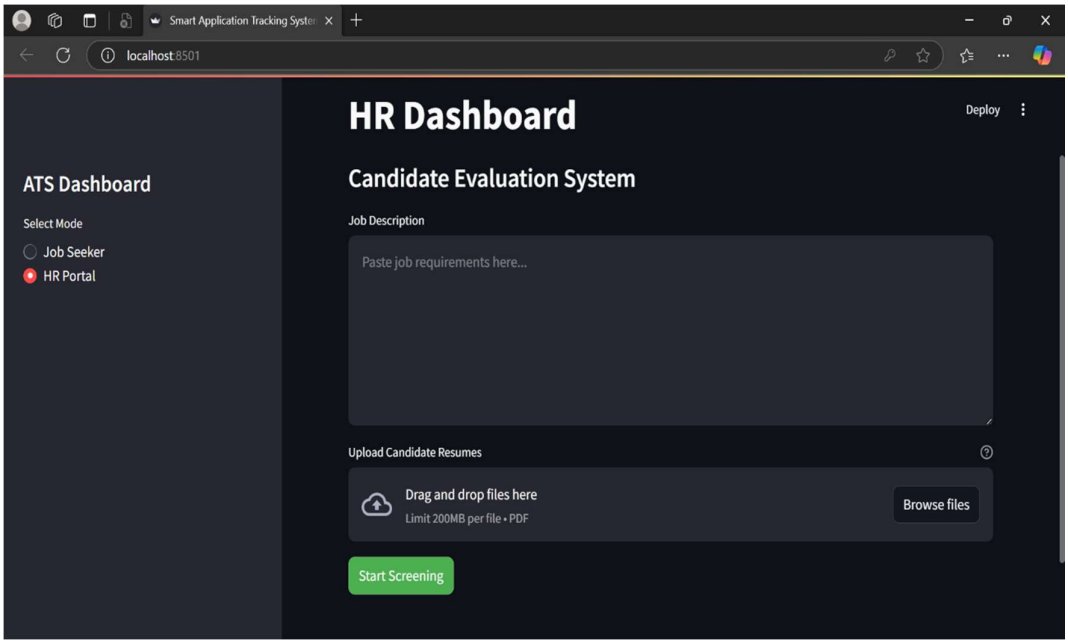

Figure 10.4 HR Portal interface

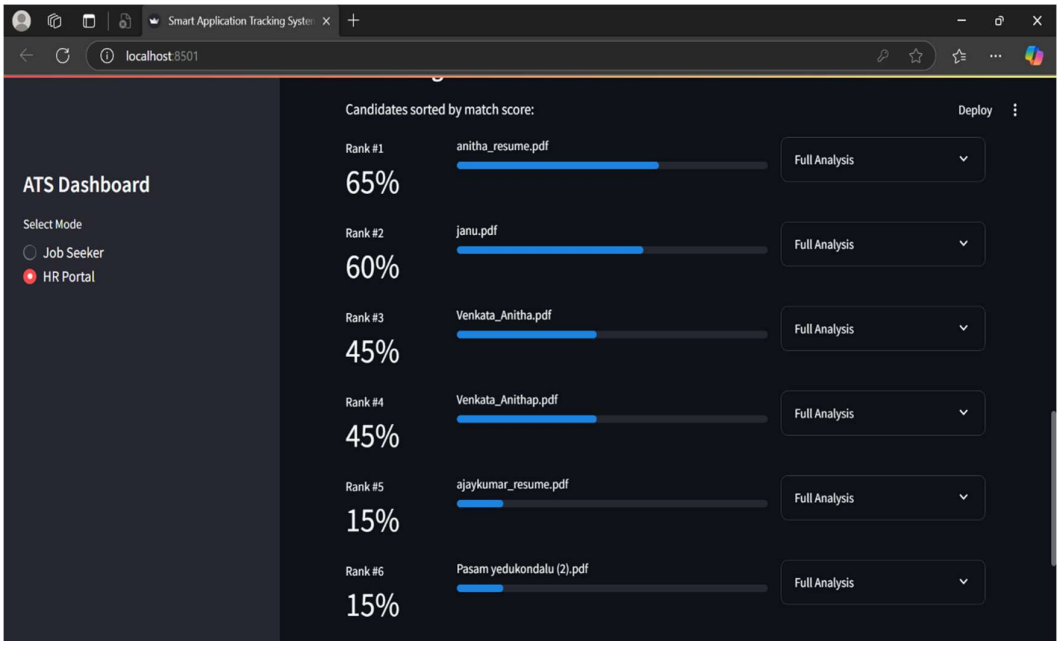

Figure 10.5 Screening Results of the Resumes in HR Portal Interface

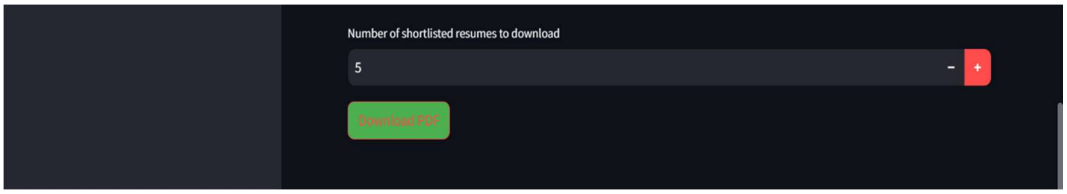

Figure 10.6 Number of resumes to be shortlisted download button

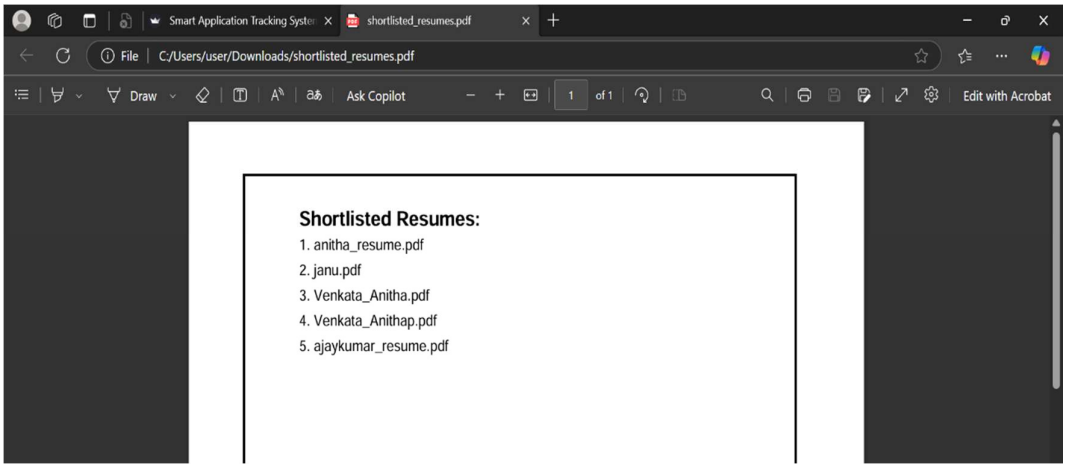

Figure 10.7 Shortlisted Resumes PDF

11 Risks and Challenges

a. Reliance on Keyword-based Analysis: The Smart Application Tracking System heavily depends on AI-driven keyword matching to analyze resumes against job descriptions. While effective for identifying skill alignment, this approach may oversimplify candidate evaluation. Candidates with deep expertise may be misclassified if they do not explicitly use the expected keywords in their resumes. This could lead to missed opportunities for qualified applicants and reduce the system's accuracy in identifying true talent.

b. Limited Applicability to Fresher: Another challenge associated with the smart application tracking system is its restricted scope, as it is currently only designed to cater to fresher. This limitation can hinder its usefulness for organizations seeking candidates with experience beyond entry-level positions. By exclusively targeting fresher, the application may not adequately address the hiring needs of companies looking to fill mid-level or senior roles. This restriction can limit the application's market appeal and potential user base, posing challenges for its adoption and long-term viability.

c. API Dependency and Service Downtime: The application relies on external APIs such as Google Gemini for AI analysis. Any disruption in API availability, quota exhaustion, or connectivity issues may affect the system's performance and lead to temporary unavailability of resume analysis features. This dependency can reduce reliability, especially during high-demand periods.

d. User Trust and Adoption Barriers: A significant challenge for any new technology or application is gaining user adoption and establishing trust among

potential users. Users, especially recruiters and job seekers unfamiliar with AI tools, may hesitate to trust automated analysis. Concerns about accuracy, transparency, and control over results can hinder adoption. Building user confidence through clear explanations, consistent performance, and user-friendly interfaces is essential for system success.

e. Data Privacy and Security Concerns: Handling resumes and job descriptions involves sensitive user data. Ensuring that files are securely uploaded, processed, and not misused is a critical challenge. Any breach or lack of transparency could harm user trust and legal compliance.

12 Conclusions

The purpose of this project was to build a Smart Application Tracking System that leverages Natural Language Processing and AI to bridge the gap between job seekers and recruiters. Throughout this project, we explored various tools and technologies including Streamlit for UI development, Google Gemini AI for resume and job description analysis. We developed modules for resume scoring, keyword analysis, and automatic resume enhancement, along with HR features like shortlisting and exporting candidate data.

This project allowed us to dive deep into the workings of an ATS and understand how real-world recruitment software functions. We gained hands-on experience in integrating AI APIs, managing resume data, and providing meaningful insights through visual analytics using Plotly. It also helped us understand user behavior and design a user-friendly interface for both job seekers and recruiters.

Additionally, we improved our teamwork and project planning skills by dividing the work efficiently and completing all milestones within the project timeline. We also focused on non-technical aspects such as requirement gathering, writing documentation, drawing architecture and flow diagrams, and conducting proper testing.

Overall, this project helped us strengthen our technical skills in Python, NLP, data visualization, API integration, and full-stack development. It also taught us how to approach real-life problems with innovative solutions, making this a highly valuable and enriching learning experience.

13 Future Enhancements

a. Currently web applications are deployed locally, our future aim is to deploy them on the internet.

b. In the future, we will add more formats of resumes. Currently, the system only supports PDF format for uploading resumes.

c. Currently, the system operates using the free version of the API key, which imposes limitations on the number of resumes that can be processed due to quota restrictions. In future iterations, the application aims to adopt the paid version of the API, allowing for higher throughput, faster processing, and unrestricted access to advanced features. This will significantly enhance the system's scalability and performance for enterprise use.

d. We have achieved good accuracy in fetching user data, but sometimes the displayed data fetched from the PDF is incorrect. We will improve this in the future.

e. In the current system, matching score, missing keywords and resume improvement suggestions are displayed. As a future enhancement, the application will offer an automated PDF generation and download feature. After analyzing and optimizing the resume with relevant keywords, the system will automatically generate and provide the improved resume in PDF format, streamlining the user experience and saving time for job seekers.

f. In the mobile view of the application, there are occasional UI lags. We will address this issue soon.

14 References

[1] K. Sri Surya, Reguri Sharanya, Afrah Zilani, Dr. CH. Niranjan, "Smart Applicant Tracking System using Gen AI", 2024 *International Journal for Innovative Engineering and Management Research*, 13(4), 505-516.

[2] Mr. Chirag Dariyani, Gurmeet Singh Chhabra, Harsh Patel, Indrajit Chhabra. "An Automated Resume Screening Using Natural Language Processing And Similarity." *Ethics and Information Technology*.

[3] Shubham Bhor, Vivek Gupta, Vishak Nair, Harish Shinde, Mansi Kulkarni. "A Resume Parser Using Natural Language Processing Technique." *International Journal of Research in Engineering and Science (IJRES)*.

[4] Satyak Sanyal, Souvik Hazra, Neelanjan Ghosh, Soumyashree Adhikary. "Resume Parser with Natural Language Processing." *2017 IJESC*.

[5] Pradeep Roy, Sarabjeet Chaudhary, Rocky Bhatia. "A Machine Learning Approach for Automation of Resume Recommendation System." *ICCIDS 2019*.

[6] Streamlit Documentation. *Documentation from Streamlit Developer Community*.

[7] OpenCV Documentation. *Documentation from OpenCV Developer Community*.

[8] Pillow Documentation. *Official documentation from PIL Developer Community*.

[9] NLTK Documentation. *Official documentation from NLTK Developer Community*.

[10] PyResParser Documentation. *Official documentation from PyResParser Developer Community*.