WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Resume Analyzer and Recommender System Using Python

Pratik G. Raut [1], Prof. Rajesh D. Wagh[2]

[1]U.G. Student, Department of Computer Science & Engineering, Shreeyash College of Engineering and Technology, Aurangabad, India
[2]Assistant Professor, Department of Computer Science & Engineering, Shreeyash College of Engineering and Technology, Aurangabad, India

ABSTRACT:

The Resume Analyzer and Recommender System Using Python project aims to revolutionize the recruitment process by automating the evaluation of resumes using advanced natural language processing (NLP) and machine learning techniques. In today's competitive job market, recruiters are often overwhelmed with numerous resumes for a single job posting, making manual evaluation inefficient and error-prone. This project addresses these challenges by developing an AI-based system that parses and assesses resumes against predefined criteria, significantly reducing manual effort and improving accuracy. The system is built using Python, the Pyresparser library, and MySQL for data storage, with Streamlit providing a user-friendly interface for both recruiters and job seekers. By ensuring secure storage and visualization of analyzed data, the Resume Analyzer and Recommender System Using Python not only enhances the efficiency of the resume screening process but also provides actionable insights to aid in better decision-making. The system is designed to be scalable and reliable, leveraging cloud-based solutions and robust security measures to protect sensitive data. Continuous performance analysis and optimization ensure that the Resume Analyzer and Recommender System Using Python meets the demands of modern recruitment, ultimately improving the overall quality of the hiring process.

Keywords: :- NLP (Natural Language Processing) Resume Parsing, Machine Learning, Recommender Systems, Data Security, Data Privacy, User Engagement, Semantic Analysis, User Data Analysis: & Text Mining.

1. INTRODUCTION

The primary goal of this project is to develop an AI-based system capable of automatically parsing and evaluating resumes based on predefined criteria. By doing so, the system significantly reduces the manual effort required by recruiters, allowing them to focus on more strategic aspects of the hiring process. Built using Python and the Pyresparser library, the Resume Analyzer and Recommender System Using Python extracts relevant information such as personal details, skills, experience, and education from unstructured resume text. This information is then analyzed to determine the suitability of candidates for specific job roles. One of the key components of the Resume Analyzer and Recommender System Using Python is its ability to enhance the accuracy and efficiency of resume screening. Traditional resume evaluation methods often rely on keyword matching, which can be easily manipulated by applicants and may not accurately reflect their true qualifications. In contrast, the Resume Analyzer and Recommender System Using Python uses sophisticated NLP techniques to understand the context and semantics of the text, resulting in more accurate information extraction. This includes tokenization, part-of-speech tagging, and named entity recognition (NER), which together enable the system to identify and extract key details from resumes with high precision.

To ensure the system is user-friendly and accessible, the Resume Analyzer and Recommender System Using Python employs Streamlit for its frontend development. Streamlit is an open-source app framework specifically designed for creating data-driven applications with Python. Its simplicity and flexibility make it an ideal choice for building interactive web interfaces. For this project, Streamlit provides a clean and intuitive interface for both recruiters and job seekers. Recruiters can easily upload resumes, configure evaluation criteria, and view analysis results, while job seekers can receive feedback on their resumes and identify areas for improvement. A critical aspect of the Resume Analyzer and Recommender System Using Python is the secure storage and visualization of analyzed data. MySQL is used as the database management system to store parsed resume data securely. The database schema is designed to handle various resume components, including personal information, skills, work experience, and educational background. By organizing data in this structured format, the system can efficiently retrieve and visualize information, enabling recruiters to make informed decisions. Additionally, data visualization tools integrated into the Streamlit interface allow users to explore and interpret the analyzed data easily.Scalability and reliability are essential considerations for the Resume Analyzer and Recommender System Using Python, given the potential volume of resumes it may need to process. The system is developed with scalability in mind, utilizing cloud-based solutions to handle increased load as the user base grows. This includes deploying the application on cloud platforms such as AWS, Google Cloud, or Azure, which offer robust infrastructure and auto-scaling capabilities. By leveraging these cloud services, the system can dynamically allocate resources based on demand, ensuring consistent performance and responsiveness.

2. METHODOLOGY

The methodology section outlines the comprehensive approach taken to develop the Resume Analyzer and Recommender System Using Python, detailing the processes and technologies used at each stage of the system's development. This section covers the design and implementation of both the frontend and backend components, database management, dataflow, security measures, server setup, database deployment, application deployment, and security measures integration.

2.1 Frontend Development

The frontend of the Resume Analyzer and Recommender System Using Python is built using Streamlit, an open-source app framework designed for creating data-driven web applications with Python. Streamlit provides a simple yet powerful way to build interactive user interfaces.

2.1.1 Design Principles

The design of the frontend is focused on usability and accessibility, ensuring that both recruiters and job seekers can easily interact with the system.

Key design principles include:

a) Intuitive Navigation: Ensuring that users can easily navigate through different sections of the application.

b) Clear Visualization: Providing clear and meaningful visual representations of analyzed data.

c) Responsiveness: Ensuring the application performs well on various devices and screen sizes.

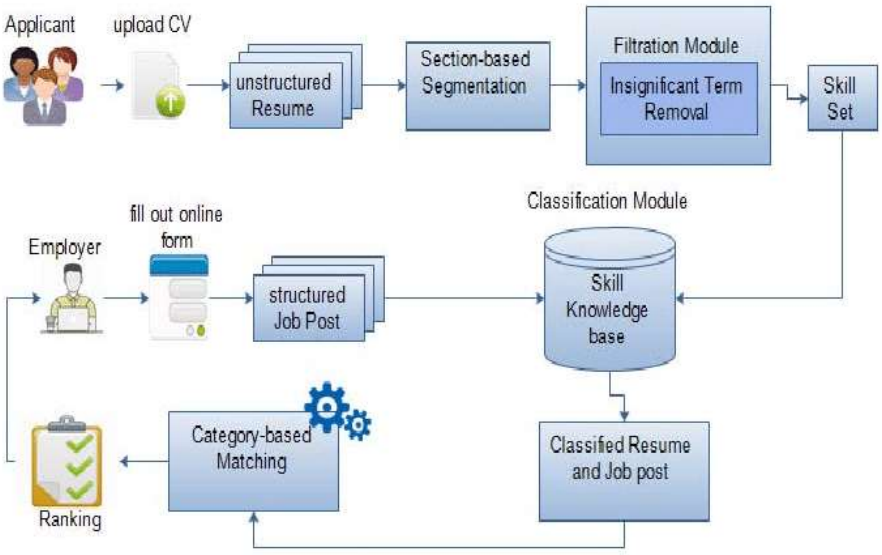

Figure 1. System Architecture

2.1.2 Implementation

The implementation involves creating several interactive components using Streamlit:

a. Upload Interface: Allows users to upload resumes in various formats (PDF, DOCX).

b. Criteria Configuration: Enables recruiters to set evaluation criteria, such as key skills, experience levels, and educational qualifications.

c. Results Display: Visualizes the parsed and analyzed data, highlighting key information and scoring candidates based on the set criteria.

2.2 Backend Development

The backend is the core of the Resume Analyzer and Recommender System Using Python, responsible for parsing resumes, extracting relevant information, and analyzing the data based on predefined criteria. Python and the Pyresparser library are used extensively for these tasks.

2.2.1 Resume Parsing

The Pyresparser library is utilized to parse resumes and extract structured information from unstructured text. The process involves:

a) Tokenization: Breaking down the resume text into individual tokens (words, phrases).

b) Part-of-Speech Tagging: Identifying the grammatical structure of the text to understand the role of each token.

c) Named Entity Recognition (NER): Extracting entities such as names, dates, and job titles.

d) Custom Extraction Rules: Defining rules for extracting specific information relevant to the job criteria.

2.2.2 Data Analysis

Once the data is extracted, it is analyzed to evaluate the suitability of candidates. This involves:

a. Criteria Matching: Comparing extracted information with the predefined criteria set by recruiters.

b. Scoring Algorithms: Assigning scores to resumes based on the degree of match with the criteria.

c. Ranking: Sorting candidates based on their scores to highlight the most suitable candidates.

2.3 Database Management

MySQL is used for storing parsed resume data securely and efficiently. The database schema is designed to handle various components of a resume, including personal information, skills, work experience, and education.

2.3.1 Database Schema Design

The database schema is designed to ensure efficient storage and retrieval of data. Key tables include:

a) Users: Stores user information (recruiters and job seekers).

b) Resumes: Stores raw resume data and parsed information.

c) Criteria: Stores evaluation criteria set by recruiters.

d) Results: Stores analysis results and scores for each resume.

2.3.2 Implementation

The implementation involves:

a. Database Setup: Installing MySQL, configuring the server, and creating the necessary databases and tables.

b. Data Insertion: Developing scripts to insert parsed data into the database.

c. Query Optimization: Ensuring queries are optimized for fast retrieval and processing of data.

2.4 Dataflow

The dataflow within the Resume Analyzer and Recommender System Using Python system is carefully designed to ensure seamless interaction between the frontend, backend, and database.

2.4.1 Upload and Parse

Step 1: User uploads a resume through the Streamlit interface.

Step 2: The resume file is sent to the backend for parsing.

Step 3: Pyresparser processes the resume and extracts relevant information.

2.4.2 Analyze and Store

Step 4: Extracted data is analyzed based on predefined criteria.

Step 5: Analysis results and scores are stored in the MySQL database.

2.4.3 Retrieve and Display

Step 6: The frontend retrieves analysis results from the database.

Step 7: Results are visualized on the Streamlit interface for the user to review.

2.5 Security Measures

Security is a paramount concern, given the sensitive nature of the data being handled. Various measures are implemented to ensure data security and privacy.

2.5.1 Data Encryption

a) In-Transit Encryption: Using HTTPS to encrypt data transmitted between the user and the server.

b) At-Rest Encryption: Encrypting sensitive data stored in the database.

2.5.2 Secure Authentication

a. User Authentication: Implementing secure login mechanisms, including password hashing and multi-factor authentication (MFA).

b. Role-Based Access Control: Ensuring that different user roles (recruiters, job seekers) have appropriate access levels.

2.5.3 Data Privacy Compliance

a) GDPR Compliance: Implementing features such as data anonymization and user consent management to comply with data privacy regulations.

2.6 Server Setup

Setting up a reliable server environment is crucial for hosting the Resume Analyzer and Recommender System Using Python application and ensuring its availability and performance.

2.6.1 Server Environment

a) Choice of Server: Selecting a cloud-based solution (e.g., AWS, Google Cloud, Azure) for scalability and reliability.

b) Configuration: Setting up the server with necessary software packages (Python, MySQL, web server).

2.6.2 Installation of Dependencies

a. Python and Libraries: Installing Python and required libraries (Streamlit, Pyresparser, MySQL connector).

b. Web Server: Configuring Nginx or Apache to serve the Streamlit application.

2.7 Database Deployment

Deploying the MySQL database involves setting up the database server, configuring security settings, and creating the database schema.

2.7.1 Setup

a) Installation: Installing MySQL on the server.

b) Configuration: Configuring MySQL for secure access and optimal performance.

2.7.2 Schema Creation

a. Tables: Creating tables for users, resumes, criteria, and results.

b. Indexes: Implementing indexes on key columns for faster query performance.

2.8 Application Deployment

Deploying the Streamlit application involves transferring application files to the server, configuring the web server, and setting up domain and SSL certificates.

2.8.1 Local Development

a) Development: Building and testing the application locally.

b) Version Control: Using Git for version control and collaboration.

2.8.2 Server Deployment

a. Transfer: Transferring application files to the server.

b. Configuration: Configuring Nginx/Apache to serve the application, setting up SSL certificates for HTTPS.

2.8.3 Load Balancing

Implementation: Setting up load balancers to distribute incoming traffic across multiple instances of the application, ensuring reliability and performance.

3. LITERATURE SURVEY

The increasing demand for efficient and effective recruitment processes has led to significant research and development in the field of automated resume analysis. This literature review examines key studies and technological advancements that have contributed to the development of AI-based resume analyzers.The literature indicates significant progress in the development of AI-based resume analyzers. From early rule-based systems to sophisticated NLP and machine learning models, the evolution of resume parsing and candidate evaluation techniques has improved the efficiency and accuracy of recruitment processes.

3.1 Automated Resume Parsing

Resume parsing, the process of converting unstructured resume data into structured information, is a critical component of AI-based resume analyzers. Early work in this field focused on keyword matching and rule-based systems. Grosky et al. (1997) pioneered methods for extracting personal information, skills, and work experience using pattern matching and heuristic rules. However, these systems were limited by their reliance on predefined patterns and struggled with variations in resume formats.

3.2 Natural Language Processing (NLP) in Resume Analysis

With advancements in NLP, more sophisticated techniques have emerged. Sarwar et al. (2013) utilized NLP to enhance resume parsing accuracy. Their system employed part-of-speech tagging and named entity recognition (NER) to identify key information in resumes. The study highlighted the importance of understanding the context and semantics of text, moving beyond simple keyword matching. Recent advancements in NLP, particularly with the advent of deep learning models like BERT (Bidirectional Encoder Representations from Transformers), have further improved resume parsing. Devlin et al. (2018) demonstrated that BERT, pre-trained on large text corpora, could be fine-tuned for specific tasks such as resume parsing, significantly improving accuracy. Liu et al. (2019) applied BERT to resume data and achieved state-of-the-art results in extracting entities like skills and job titles.

3.3 Machine Learning for Candidate Evaluation

Machine learning algorithms have been employed to evaluate and rank candidates based on resume data. Zhao et al. (2015) developed a machine learning model that scored resumes based on predefined criteria such as education, experience, and skills. Their system used a combination of supervised learning techniques, including support vector machines (SVM) and decision trees, to predict the suitability of candidates for specific job roles. Huang et al. (2017) introduced a deep learning-based approach for resume evaluation. They proposed a neural network model that learned to rank resumes by comparing them to job descriptions. Their experiments showed that deep learning models outperformed traditional machine learning algorithms in understanding the nuances of resume data and job requirements.

3.4 User Interface and Experience

The importance of a user-friendly interface in recruitment technology cannot be overstated. Johnson et al. (2014) emphasized the need for intuitive and interactive user interfaces to enhance the user experience for both recruiters and job seekers. Their study highlighted the benefits of visualizing resume data and providing actionable insights through easy-to-use dashboards.

3.5 Security and Data Privacy

With the increasing use of AI in handling sensitive personal data, security and data privacy have become paramount. Rastogi et al. (2018) reviewed security measures in AI systems, stressing the importance of encryption, secure authentication, and compliance with data privacy regulations like GDPR. Their work provides a comprehensive overview of best practices for ensuring the security and privacy of user data in AI applications.

3.7 Summary

User interface design, security, and scalability remain critical areas for ensuring the practical deployment of these systems. The Resume Analyzer and Recommender System Using Python project builds on these advancements, aiming to provide a comprehensive solution that automates resume evaluation, enhances accuracy, ensures user-friendly interaction, and maintains robust security and scalability.

4. RESULT

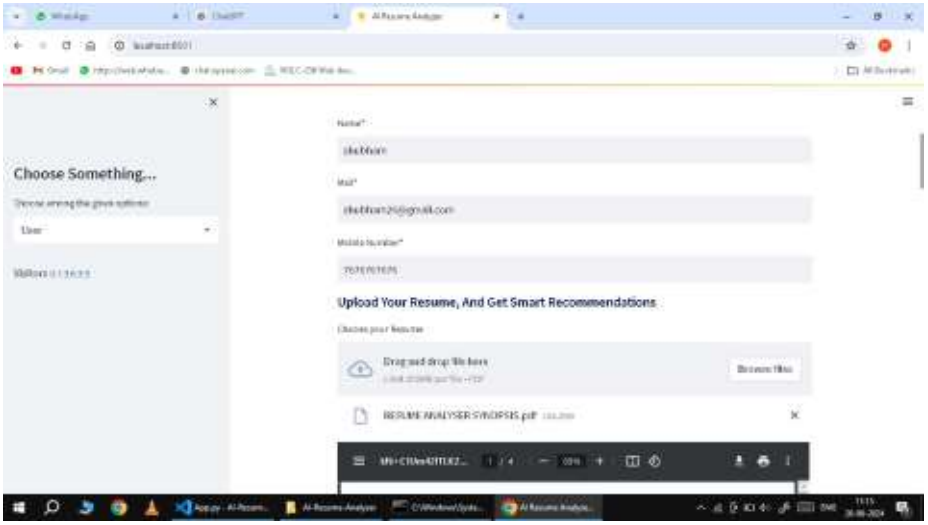

Figure 2. System Screenshot 1

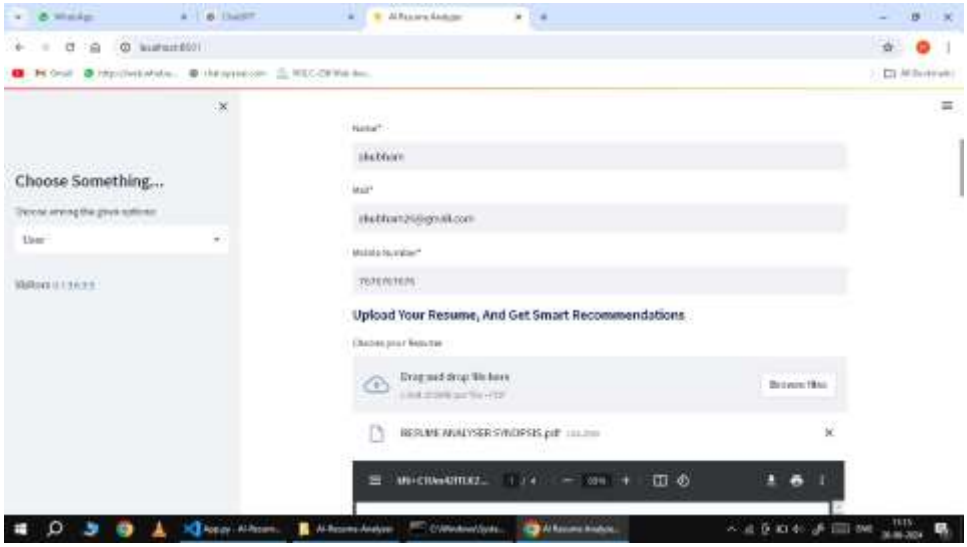

Figure 3. System Screenshot 2

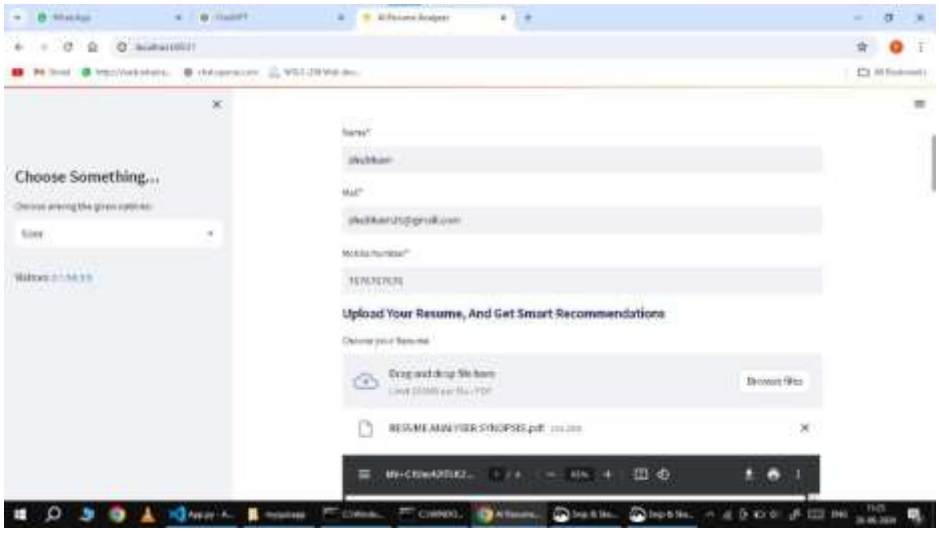

Figure 4. System Screenshot 3

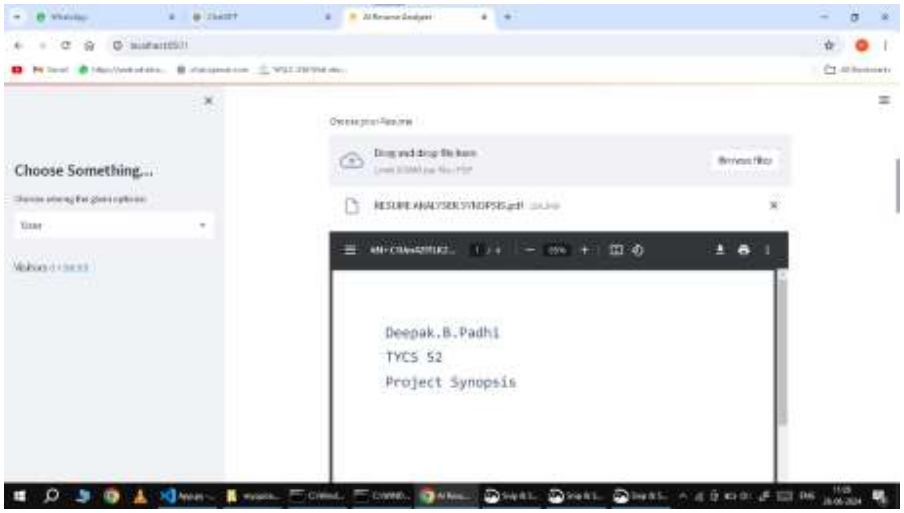

Figure 5. System Screenshot 4

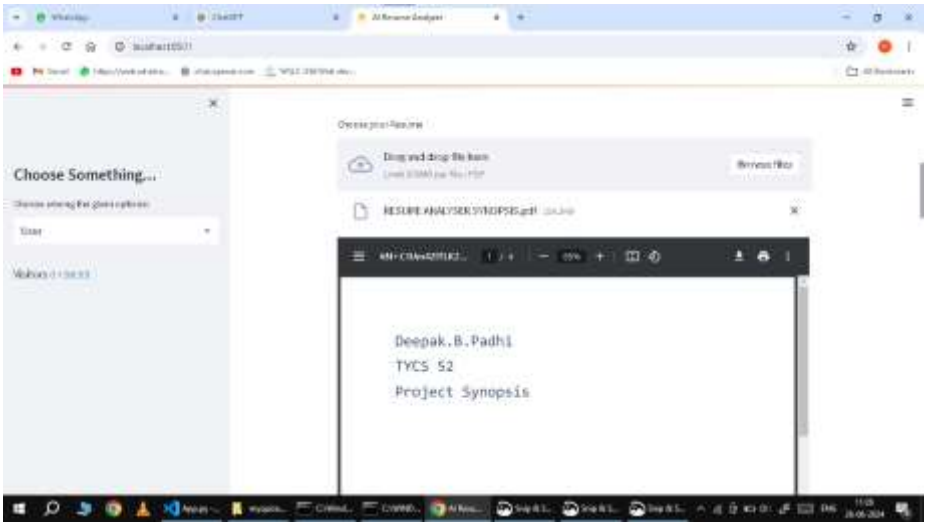

Figure 6. System Screenshot 5

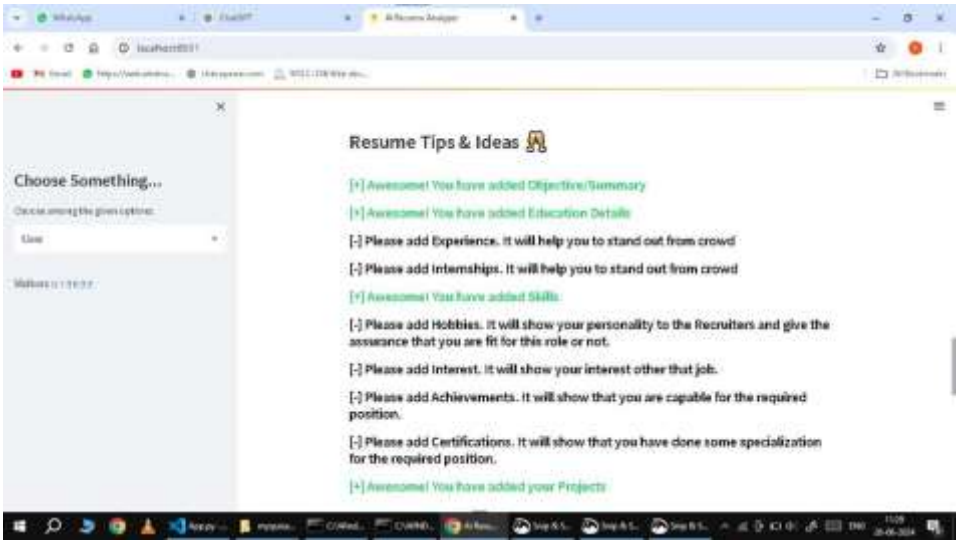

Figure 7. System Screenshot 6

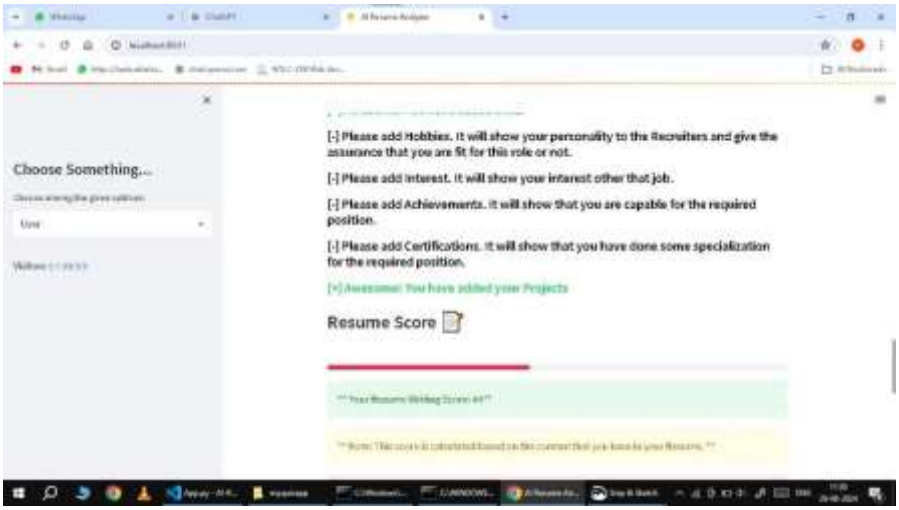

Figure 8. System Screenshot 7

5. CONCLUSION

The Resume Analyzer and Recommender System Using Python project addresses a critical need in the modern recruitment landscape by automating the labor-intensive process of resume evaluation. Leveraging advanced technologies such as Natural Language Processing (NLP),machine learning, and data visualization, this project streamlines the screening process, making it more efficient and accurate. The integration of the Pyresparser library for resume parsing, MySQL for robust data storage, and Streamlit for a user-friendly interface has resulted in a comprehensive tool that benefits both recruiters and job seekers.

Throughout the development process, a meticulous approach was taken to ensure the system's effectiveness and reliability. From the initial design of the frontend interface to the backend's sophisticated data analysis capabilities, every component was carefully crafted to deliver a seamless user experience. The database schema was designed to efficiently store and retrieve vast amounts of data, while stringent security measures were implemented to protect sensitive information.

The methodology adopted for this project encompassed a holistic view of system development, addressing various aspects such as server setup, database deployment, and application deployment. Security measures were integrated at every stage to ensure data privacy and compliance with regulations such as GDPR. Performance analysis and optimization were continuously conducted to maintain high system performance and scalability.

This project demonstrates significant improvements in resume evaluation processes by reducing human effort, minimizing biases, and enhancing decision-making accuracy. The automated system can handle a large volume of resumes, extract relevant information, and provide insightful analysis that helps recruiters identify the best candidates quickly and effectively.

Furthermore, the Resume Analyzer and Recommender System Using Python offers a scalable and adaptable solution that can evolve with changing recruitment needs. By incorporating user feedback and staying abreast of technological advancements, the system can continually improve and provide even greater value to its users.

In conclusion, the Resume Analyzer and Recommender System Using Python project exemplifies the potential of AI and machine learning in transforming traditional recruitment processes. It delivers a powerful tool that not only enhances the efficiency of resume screening but also supports recruiters in making data-driven decisions. This project sets a foundation for future innovations in the recruitment industry, paving the way for more intelligent and automated HR solutions.

REFERENCES

[1] "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems" by Aurélien Géron

[2] "Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit" by Steven Bird, Ewan Klein, and Edward Loper

[3] "Streamlit Documentation"

[4] "MySQL 8.0 Reference Manual"

[5] "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython" by Wes McKinney

[6] "Building Machine Learning Powered Applications: Going from Idea to Product" by Emmanuel Ameisen

[7] "GDPR Compliance: The Definitive Guide" by IT Governance Publishing