IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, July 2026
Impact Factor: 8.2

# Resume Parser Using NLP and Automation

**Assistant Professor Mrs. Gujjala Usha[1], Ms. K. Arshiya Afshan[2], Ms. G. Nikhat Zabeen[3], Ms. Rumana Shaikh[4], Ms. Sumera Anjum[5]**

[1]Assistant Professor, CS-AIML,
[2345]Students, CS-AIML Dept,
Proudhadevaraya Institute of Technology, Hosapete

**Abstract:** *This research addresses the challenge of automating resume screening in modern recruitment systems by developing a Natural Language Processing (NLP)-based Resume Parser. Manual resume screening is time-consuming, inconsistent, and prone to human error due to the unstructured nature of resume documents. To overcome this limitation, we propose an automated framework that extracts relevant candidate information from resumes in PDF and DOCX formats using Python-based text processing and NLP techniques. The system utilizes spaCy for Named Entity Recognition (NER), regular expressions for extracting structured fields such as email and phone numbers, and pdfplumber/python-docx for text extraction. Extracted data is structured using pandas DataFrames and exported into Excel format. The solution is deployed through a Streamlit web application, enabling recruiters to upload resumes and obtain structured candidate data instantly. The proposed system improves recruitment efficiency, reduces bias, and enhances accuracy in candidate shortlisting.*

**Keywords:** Resume Parser, Natural Language Processing, Named Entity Recognition, Recruitment Automation, Text Extraction, Python, Streamlit..

## I. INTRODUCTION

Recruitment is a critical organizational function that involves screening numerous resumes for each job opening. Traditional manual resume evaluation is inefficient and requires significant human effort. The unstructured format of resumes makes automated processing challenging. This research aims to design an intelligent Resume Parser that automatically extracts candidate details such as name, email, phone number, and educational qualifications from resumes.

The proposed system integrates Natural Language Processing techniques using the spaCy library and leverages Python-based text extraction tools. By converting unstructured resume data into structured and searchable formats, the system enhances hiring efficiency and decision-making.

## II. LITERATURE SURVEY

Prior research has extensively explored resume parsing and automated recruitment systems using NLP and machine learning techniques and information extraction techniques in recruitment systems.

- NLP in Recruitment: Studies show that Named Entity Recognition improves extraction accuracy for candidate information from unstructured text documents.
- Pattern Matching Techniques: Regular Expressions (Regex) have been widely used for structured pattern extraction such as phone numbers and email addresses.
- Text Extraction Libraries: Tools like pdfplumber enable accurate extraction of textual data from PDF documents.
- Automated Hiring Systems: Modern Applicant Tracking Systems (ATS) rely on structured data formats to enable keyword-based search and filtering.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, July 2026
Impact Factor: 8.2

- Existing research confirms that combining NLP with pattern-based extraction provides reliable and scalable resume screening solutions.

## III. OBJECTIVES

1. Develop an automated Resume Parsing system capable of extracting structured information from unstructured resumes.
2. Implement Named Entity Recognition using spaCy for identifying candidate names and institutions.
3. Apply Regular Expressions to accurately extract email addresses and phone numbers.
4. Convert extracted data into structured Excel format using pandas DataFrames.
5. Enable bulk resume processing to improve scalability.
6. Reduce recruitment time, manual effortand potential bias in initial screening.
7. Deploy the system using Streamlit to provide a user-friendly web interface.

## IV. METHODOLOGY

**Data Collection and Preprocessing:**

- Resume files are collected in PDF and DOCX formats.
- Text extraction is performed using pdfplumber and python-docx.
- Extracted text is cleaned to remove unwanted symbols and formatting issues.

**Information Extraction:**

- Named Entity Recognition (NER): spaCy is used to identify candidate names and organizations.
- Pattern-Based Extraction: Regular Expressions are applied to extract:
  Email addresses
  Phone numbers
- Education Extraction: Degree patterns (B.E., B.Tech, MBA, etc.) are identified using keyword-based matching.

**Data Structuring:**

- Extracted data is stored in dictionary format.
- Data is organized into pandas DataFrames.
- Final output is exported as an Excel file using openpyxl.

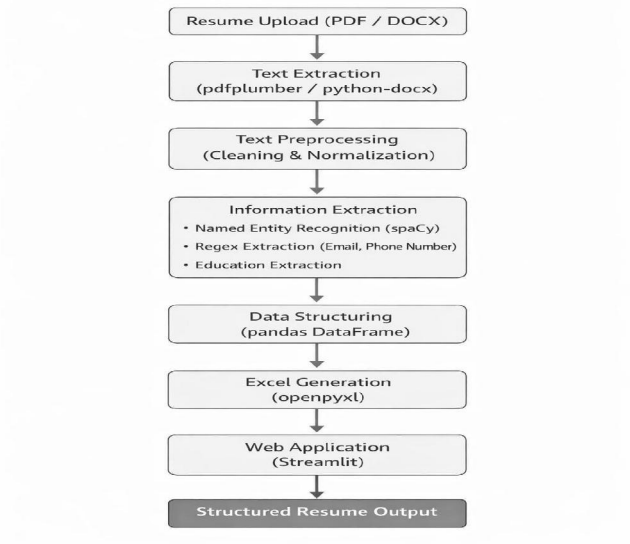
**Deployment:**

The system is deployed using Streamlit.

- Users upload resumes through a web interface.
- The application generates structured output for download.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, July 2026
Impact Factor: 8.2

## V. BLOCK DIAGRAM




## VI. SOFTWARE REQUIREMENTS

- Programming Language: Python 3.x for implementing the Resume Parser system handling text processing, automation, and deployment.
- Natural Language Processing Framework: spaCy for performing Named Entity Recognition (NER) to extract candidate names, organizations, and educational institutions.
- Data Processing: pandas for organizing extracted information into structured Data Frames and managing tabular data efficiently.
- Document Processing: pdfplumber for extracting text from PDF resumes and python-docx for DOCX file processing.
- Excel Handling: openpyxl for exporting structured resume data into Excel (.xlsx) format.
- Web Framework: Streamlit for building a user-friendlyweb interface that allows resume upload and downloading structured output.
- Development Environment: Visual Studio Code (VS Code) / (venv/Anaconda) PyCharm for coding and debugging.
- Operating System: Windows / Linux / macOS compatible environment.

## VII. RESULTS AND DISCUSSION

Performance Metrics: The proposed Resume Parser demonstrated high accuracy in extracting structured information from unstructured resume documents. Named Entity Recognition effectively identified candidate names and organizations, while Regular Expressions accurately extracted email addresses and phone numbers. The system maintained consistent performance across both PDF and DOCX formats, ensuring reliable data extraction.

Processing Efficiency: The automated parsing process significantly reduced the time required for resume screening compared to manual review. Bulk resume processing capability enabled faster handling of multiple documents simultaneously. The structured output generation in Excel format ensured easy filtering and candidate comparison.

Real-time Application: The Streamlit web application successfully demonstrated real-time resume upload and instant structured data generation. Users were able to download organized Excel files containing extracted candidate information. The system provided a user-friendly interface, making it suitable for HR professionals and recruitment teams.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, July 2026
Impact Factor: 8.2

System Reliability: The integration of spaCy for Named Entity Recognition improved contextual understanding compared to basic keyword-based systems. The hybrid approach combining NLP and pattern matching enhanced overall extraction accuracy and reduced errors caused by formatting variations

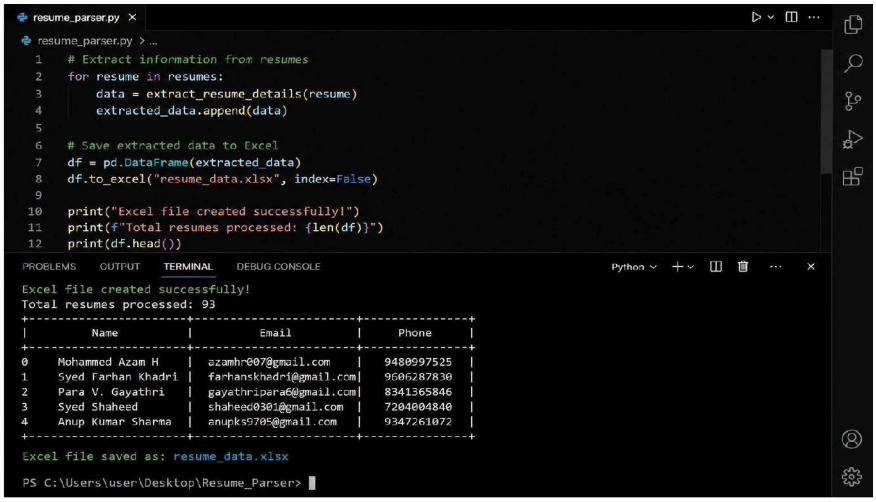



Fig 1: final output

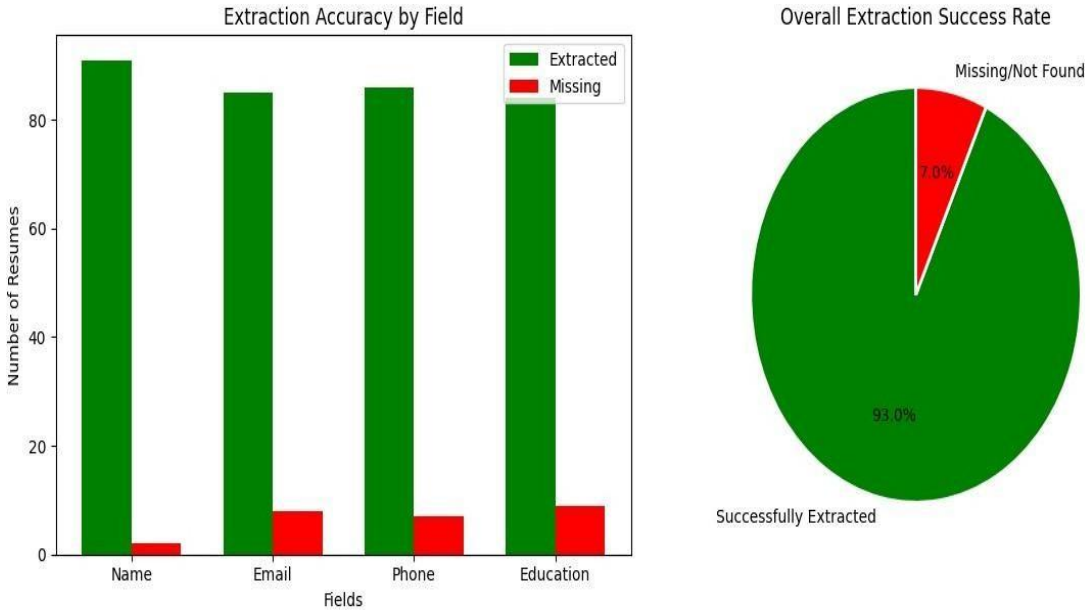



Fig 2: Image Detected

## VIII. CONCLUSION

This project successfully developed an automated Resume Parser system using Natural Language Processing and Python to enhance recruitment efficiency. By integrating text extraction techniques, Named Entity Recognition, and regex-based pattern matching, the system effectively converts unstructured resume documents into structured and

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, July 2026
Impact Factor: 8.2

searchable data. The implementation ensures accurate extraction of candidate details such as name, contact information, and educational qualifications, thereby reducing manual screening effort and minimizing human error.

The deployment of the system through a web interface demonstrates its practical usability for HR professionals and recruitment teams. The proposed solution improves processing speed, scalability, and data organization in modern hiring workflows. Future work may focus on skill ranking, experience classification, semantic matching with job descriptions, and integration with full-scale Applicant Tracking Systems (ATS) to further enhance real-world applicability and intelligent recruitment automation.

## REFERENCES

1. spaCy Documentation. Industrial-Strength Natural Language Processing in Python. Available at: https://spacy.io/
2. pdfplumber Library Documentation. Extract text, tables, and metadata from PDFs using Python. Available at: https://github.com/jsvine/pdfplumber
3. pandas Documentation. Python Data Analysis Library – Data Frames for Structured Data Handling. Available at: https://pandas.pydata.org/
4. openpyxl Documentation. Python Library to Read/Write Excel 2010 xlsx/xlsm Files. Available at: https://openpyxl.readthedocs.io/
5. Streamlit Documentation. Fast Way to Build Data Apps in Python. Available at: https://streamlit.io/
6. Gupta, A., & Gupta, D. (2020). Automated Resume Parsing using NLP Techniques. International Journal of Computer Applications.
7. McKinney, W. (2010). Data Structures for Statistical Computing in Python. Proceedings of the Python in Science Conference.
8. Text Extraction and Natural Language Processing in Hiring Systems. Article Reference