Project Report

Project Name: Smart Resume Parser
Intern Name: Renuga Devi.M

Introduction

Smart Resume Parser is a web app that extracts details from resumes (PDFs) and compares them with job descriptions, helping HR teams screen faster and more consistently.

Abstract

The app finds key fields like name, email, phone, and skills, then calculates a match score. It presents results in a table with search, filter, and export features, making resume management efficient and organized.

Tools & Technologies Used

- Python
- Flask / Streamlit
- PyMuPDF
- pandas
- Regex
- HTML, Bootstrap

Steps Involved

1. Setup project and install libraries
2. Parse PDFs to extract text
3. Use regex to find details
4. Calculate match score with job description
5. Build user interface to upload and view data
6. Add extra features: export CSV, dark mode, pagination

Conclusion

This project helped me combine Python programming and text processing to create a useful real-world tool. I gained practical experience in web development, data extraction, and user interface design. Overall, it improved my problem-solving skills and confidence in building complete, deployable applications.

Project Name: PDF to Audiobook Converter
Intern Name: Renuga Devi.M

Introduction

PDF to Audiobook Converter is a web app that helps users convert text from PDF files into spoken audio files. This tool supports multiple languages and makes documents accessible for visually impaired users or for listening on the go.

Abstract

The project extracts text from uploaded PDFs and uses text-to-speech technology to generate an audio file (MP3). Users can choose the language, adjust speed, and download the resulting audio, making reading easier and more flexible.

Tools & Technologies Used

- Python
- Streamlit
- PyMuPDF
- gTTS (Google Text-to-Speech)
- pandas
- HTML, CSS

Steps Involved

1. Setup and install required Python libraries
2. Build a user-friendly interface using Streamlit
3. Extract text content from uploaded PDF files
4. Use gTTS to convert text into audio in selected language
5. Provide options to adjust speed and download the audio file
6. Add extra features like language selection and save history

Conclusion

Through this project, I learned how to integrate text extraction and text-to-speech technologies into a user-friendly web application. It helped me understand the practical use of Python libraries for building accessible tools, and taught me how to design features like multi-language support and download options. Overall, this experience strengthened my coding skills and confidence to develop solutions that help people in everyday life.