**PROJECT REPORT**

**DYMRA Tech (Week 2) Internship Project**

Name: Somya Vats [[github]](https://github.com/vatssomya)

Type: Individual Project

Submitted: 12/06/2025

**Project Selected:**

AI-Powered Resume Ranker & Candidate Shortlister (RecruitGPT)

Problem: HR teams spend hours filtering resumes manually.

Goal: Build a system that parses resumes and job descriptions, ranks candidates, and auto-matches top profiles.

• Skills: NLP (BERT, Spacy), cosine similarity, TF-IDF

• Bonus: Build a web interface, score diversity or soft skills from text

**Overview**

This mini-project was undertaken as part of Week 2 of the internship program at DYMRA Tech. The objective was to apply AI/ML concepts in a real-world setting, following the internal workflow and building something demo-ready within 5–6 days. The themes included AI/NLP, automation, and lightweight frontends.

**Initial Development: Resume Scanner (Phase 1)**

To begin understanding the system and familiarize myself with the development pipeline, I first created a **basic Resume Scanner** web application. This tool provided a quick summary of an uploaded resume and attempted to predict the best-fit role based on extracted text.

[GITHUB](https://github.com/vatssomya/Resume-Scanner)

[Video Demo](https://drive.google.com/drive/folders/1C9OH14gm68KyKVC68QUTSqQnNRbqVtHQ?usp=sharing)

**Key Features:**

* Extracted text from .pdf, .docx, and .txt resumes
* Used TF-IDF vectorization and cosine similarity with predefined role descriptions
* Rendered predictions through a simple web interface built using Flask
* Designed for quick validation of the resume parsing and classification pipeline

This phase helped me understand the input flow, text preprocessing, integration with Flask, and resume parsing libraries. It also provided foundational functionality that I later extended into a more advanced candidate ranking system.

**Main Development: RecruitGPT – Resume Ranker & Shortlister (Phase 2)**

After successfully testing the resume scanner, I began working on the main problem statement provided:

**Problem Statement:**  
HR teams spend hours filtering resumes manually.  
**Goal:** Build a system that parses resumes and job descriptions, ranks candidates, and auto-matches top profiles.

[GITHUB](https://github.com/vatssomya/RecruitGPT)

[Video Demo](https://drive.google.com/drive/folders/1vk3kcsfcdmMylLvnhz5errH0MltloftI?usp=drive_link)

**Objectives:**

* Parse multiple resumes and a given job description
* Score and rank candidates based on relevance
* Provide a visual UI to upload and view ranked results

**Technical Stack**

* **Backend:** Python, Flask
* **NLP Techniques:**
  + Text extraction via PyMuPDF and python-docx
  + Preprocessing with SpaCy
  + TF-IDF Vectorizer for resume-job matching
  + Cosine Similarity for ranking
  + (Optional BERT-based matching planned for future improvement)
* **Frontend:** HTML, CSS, Flask integration
* **Utilities:** Logging, error handling, file sanitization, timestamp-based upload handling
* **Training Data:** Kaggle resume datasets for common role descriptions (used for similarity comparisons)

**Functional Workflow**

1. **Upload Phase:**
   * Users upload multiple resumes (PDF/DOCX/TXT)
   * A job description is entered manually
2. **Text Extraction & Preprocessing:**
   * Text is extracted using format-specific extractors
   * NLP preprocessing applied (lowercasing, stopword removal, lemmatization)
3. **Similarity Matching:**
   * TF-IDF vectors created for each resume and the job description
   * Cosine similarity computed to rank candidates
4. **Output & Display:**
   * Top-ranked candidates shown in descending order
   * Resume snippets and skill previews included in UI
   * Summary stats like total resumes processed, match score, and prediction logs displayed

**User Interface & Experience**

The frontend was enhanced in the final stage of development to match the clarity and accessibility expected from real-world tools. Key improvements:

* Responsive, clean UI with pink-blue-lilac theme
* Progress bar styling for match scores
* Expandable resume text previews
* Upload validations and feedback messages
* Print-friendly styling for final reports

**Challenges Faced**

* Text extraction inconsistencies across different resume formats
* Handling of large resume files and content size
* Balancing between speed and accuracy in TF-IDF-based comparisons
* Designing a user-friendly interface within the short time window

**Outcome**

* Successfully built a working resume parsing and candidate ranking system
* Extended the original scanner idea into a ranked, visual platform
* Gained hands-on exposure to real-world NLP workflows, Flask routing, and frontend refinement
* Understood how to design, test, and deploy mini AI tools in fast-paced timelines

**Conclusion**

This individual project gave me a complete development cycle experience – from concept to deployment. Starting with a basic scanner helped build the groundwork, which I then scaled into **RecruitGPT**, aligning directly with the problem statement provided by DYMRA Tech. Through this process, I was able to combine my learning of NLP, Flask, and web technologies to create an industry-relevant, demo-ready tool.