.

---

# 📄 AI Resume Ranker Using NLP and Streamlit

---

## 🧠 Objective:

To develop a smart resume ranking system that takes a Job Description (JD) and multiple resumes, and ranks them based on how closely they match the JD using NLP techniques and cosine similarity.

---

## 🧰 Tools & Technologies Used:

- Python
- pdfplumber (for PDF text extraction)
- Scikit-learn (`TfidfVectorizer`, `cosine_similarity`)
- Streamlit (for the user interface)
- Google Colab (for backend testing)

---

## 📁 Folder Structure:

```
AI_RESUME_RANKER/
├── data/                      ← (optional, for offline resumes)
├── ranked_output.csv          ← output from backend code
├── resume_ranker.ipynb        ← Colab notebook for backend
├── streamlit_app/
│   └── app.py                 ← final working Streamlit app
├── README.md
└── project_report.pdf
```

---

## ⚙️ How It Works:

1. The user pastes a job description.
2. Uploads one or more PDF resumes.
3. The system:
    - Extracts text from each resume
    - Converts JD + resumes into TF-IDF vectors
    - Calculates similarity using cosine similarity
    - Ranks resumes based on Match Percentage (%)
4. The Streamlit UI displays a ranking table.

---

## 📊 Sample Output:

| Resume | Match (%) |
|---|---|
| rizwan_resume.pdf | 84.73 |
| sabiha_resume.pdf | 72.15 |

---

## 📦 Deliverables:

- Jupyter Notebook: `resume_ranker.ipynb`
- Streamlit Web App: `app.py`
- CSV Output File: `ranked_output.csv`
- GitHub Repository: with `README.md`
- Project Report: `project_report.pdf`

---

## ✅ Conclusion:

This project solves a real-world HR challenge by automating resume screening using NLP. The system is simple, fast, and scalable. With its clean Streamlit interface, it can be used by recruiters to shortlist candidates efficiently without needing to read every resume manually.