**Final - Report**

**Priya Poosaala**

**Rollno: 26**

**1.Project Overview**

The AI Skill Gap Analyzer is a sophisticated application designed to bridge the gap between job seekers' skills and job market requirements. It employs advanced natural language processing and machine learning techniques to analyze resumes and job descriptions, providing detailed skill comparisons, gap analysis, and personalized learning recommendations. The application features a modern, intuitive interface built with Streamlit, offering interactive visualizations and comprehensive reporting capabilities.

**2.Objectives**

* Automate skill extraction from resumes, job descriptions, or any unstructured text.
* Enable semantic skill analysis using BERT embeddings for similarity and matching.
* Visualize skill distributions and relationships with interactive charts).
* Export results in CSV, JSON, and text formats for further analysis or reporting.
* Provide a user-friendly, interactive UI with Streamlit, supporting real-time feedback and multi-tab navigation.

**3.Data for Skill Gap Analyzer**

The system processes the following data:

* **Resume:** Uploaded by the user (candidate) as plain text (TXT) or **Job Description (JD):** Provided by employers or users in text format.

**4. Implementation Details**

The system implements a sequential six-stage workflow, with each tab building upon the results of previous stages:

Workflow Pipeline:

1. 📄 Upload Documents Tab → Document ingestion and text extraction
2. 🔍 Extract Skills Tab → Skill identification and categorization
3. 📊 Analysis Tab → Semantic similarity and gap analysis
4. 📈 Visualization Tab → Interactive charts and insights
5. 🎯 Score & Report Tab → Overall scoring and export
6. 📚 Learning Path Tab → Personalized development roadmap

**5. Text Processing Flow**

1. **Text Input:** Users paste or upload resume and job description text.
2. **Preprocessing:** Text is cleaned, tokenized, and processed with spaCy, including custom stop word handling.
3. **Skill Extraction**.
4. **Skill Normalization**: Abbreviations and synonyms are expanded for consistent matching.
5. **Categorization:** Extracted skills are mapped to predefined categories.
6. **Visualization:** Results are presented with interactive charts and tables.
7. **Export:** Users can download results in various formats.

**6. Features Implemented**

* **Semantic skill embeddings** and similarity analysis using Sentence-BERT.
* **Interactive visualizations:** Pie, bar, similarity matrix.
* **Export options:** CSV, JSON, and text reports.
* **User interface:** Multi-tab Streamlit app with real-time feedback and session state management.

**7.Statistics Computation**

The analyzer computes and displays:

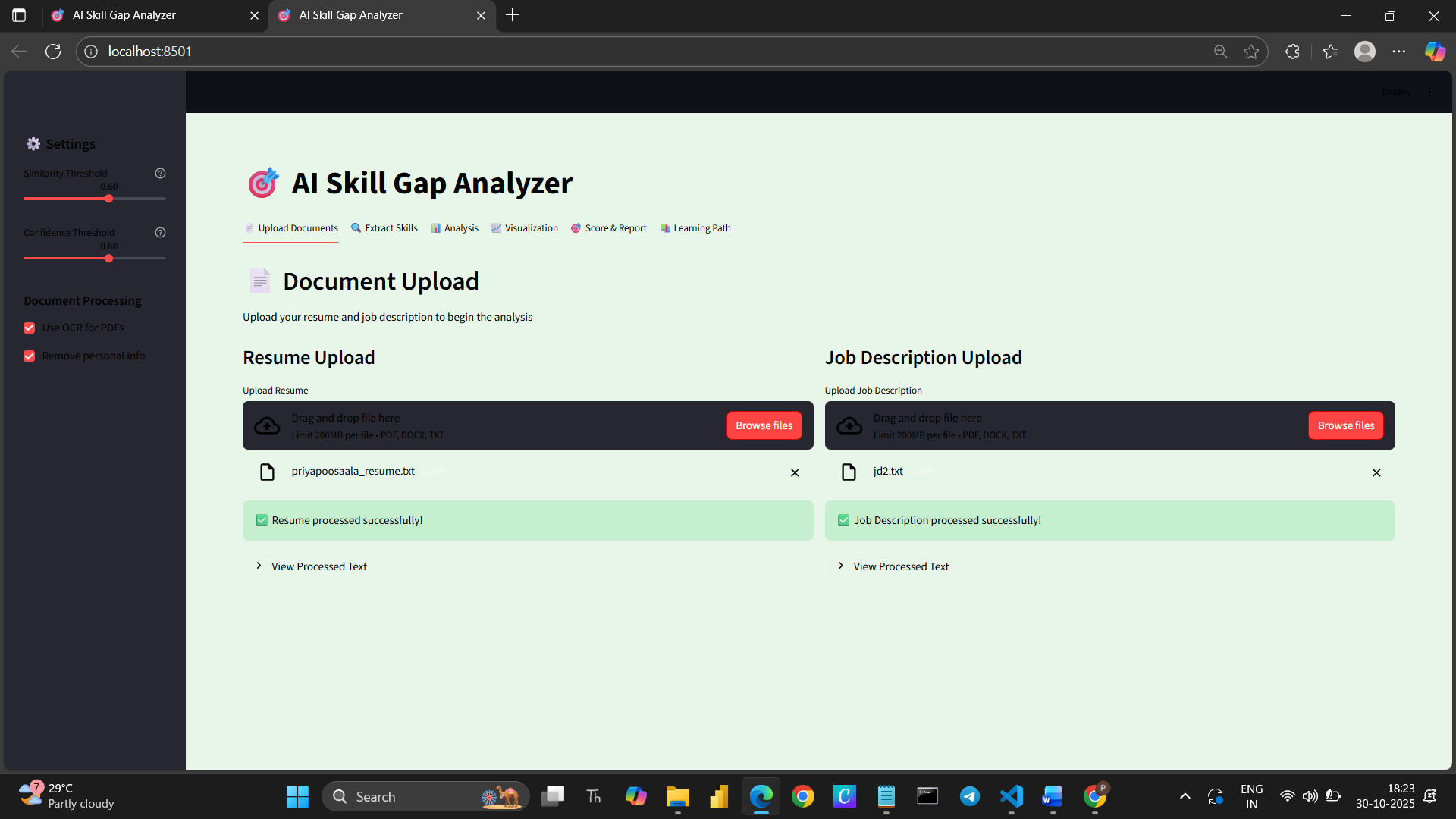
* **Total Skills Extracted**
* **Technical Skills Count**

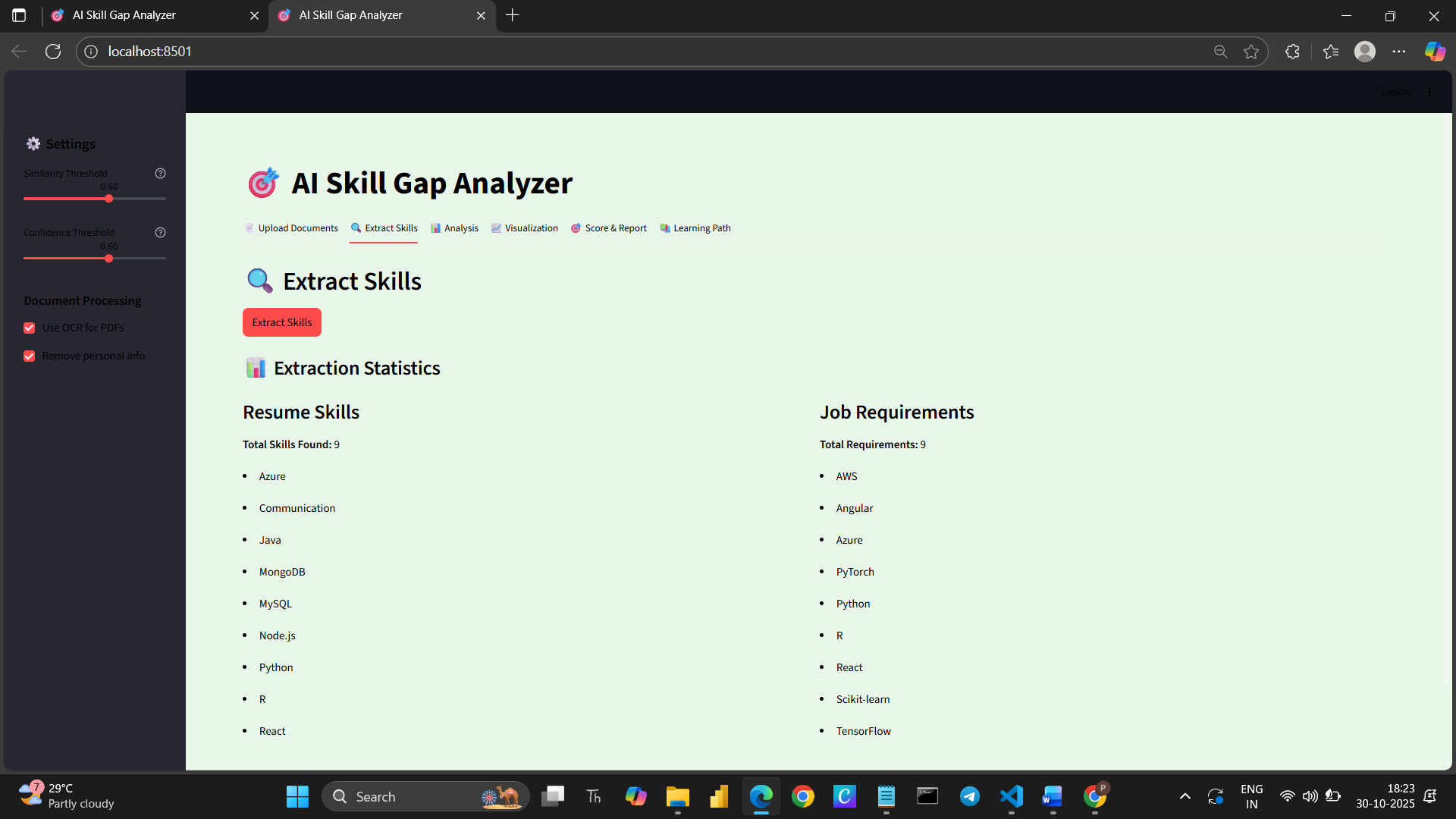
**Extraction Method Statistics** (number of skills found by each method)

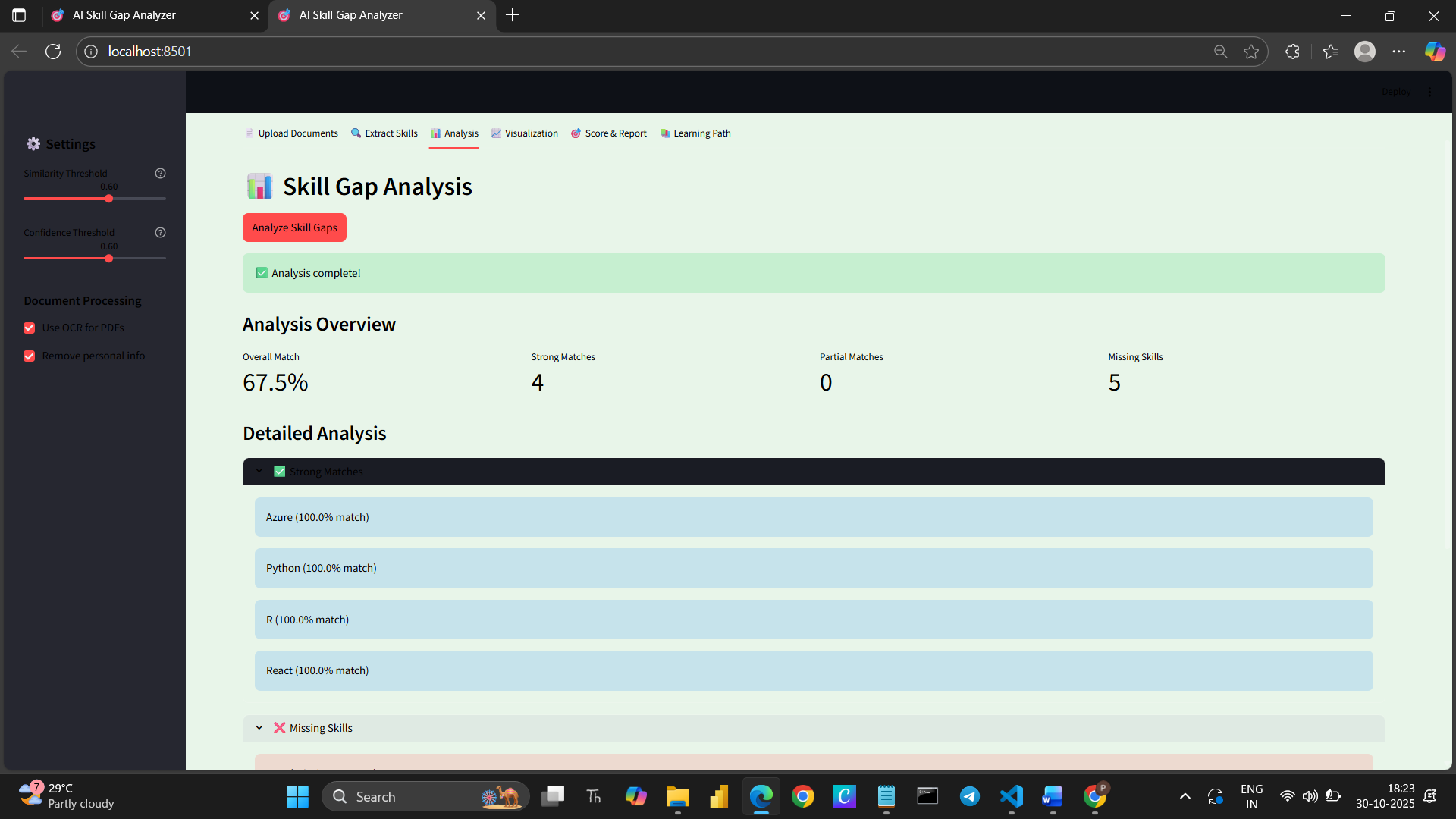
* **Skill Gap Analysis** (planned for future milestones)

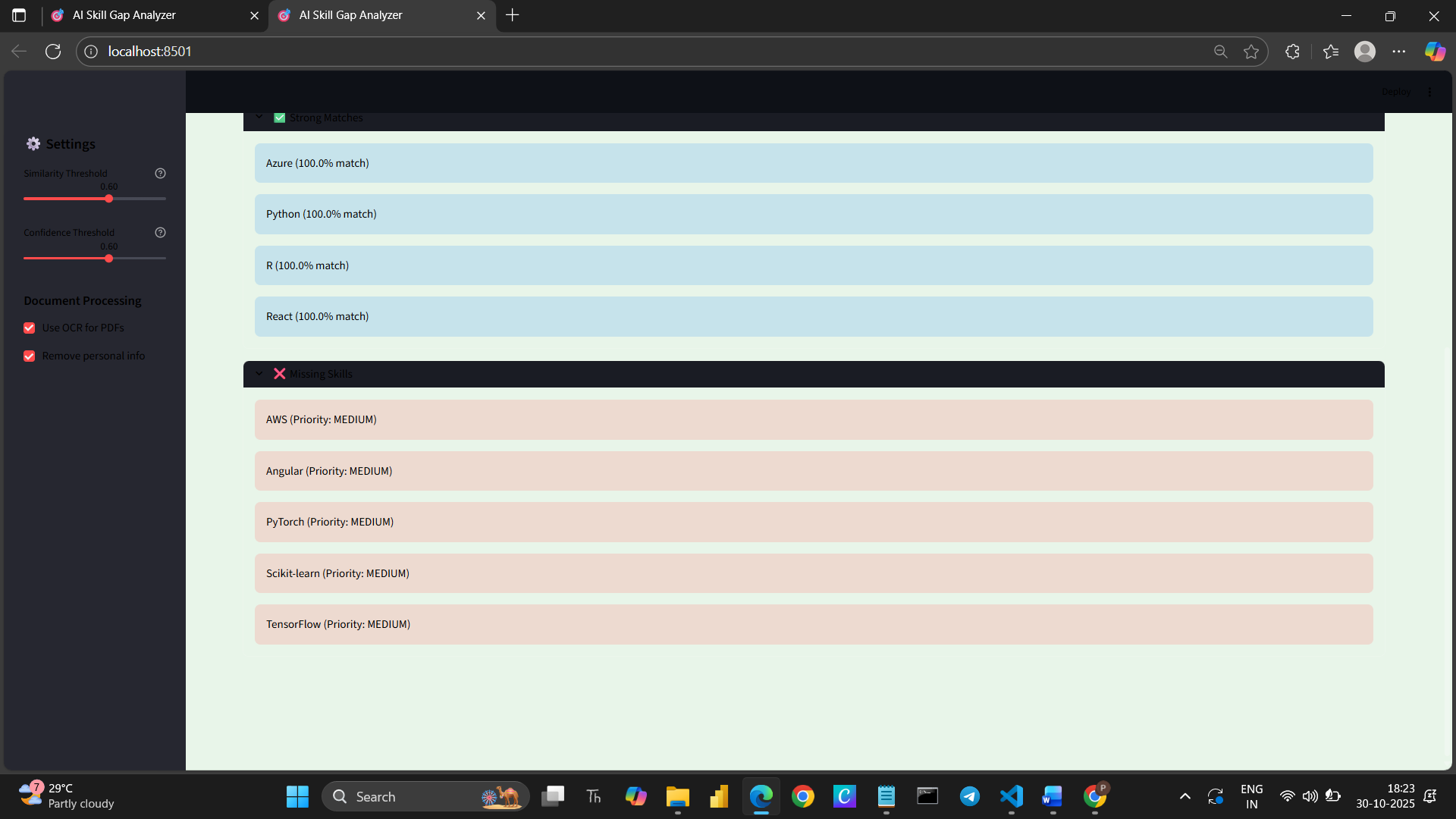
These statistics are presented as metrics, tables, and visualizations for easy interpretation.

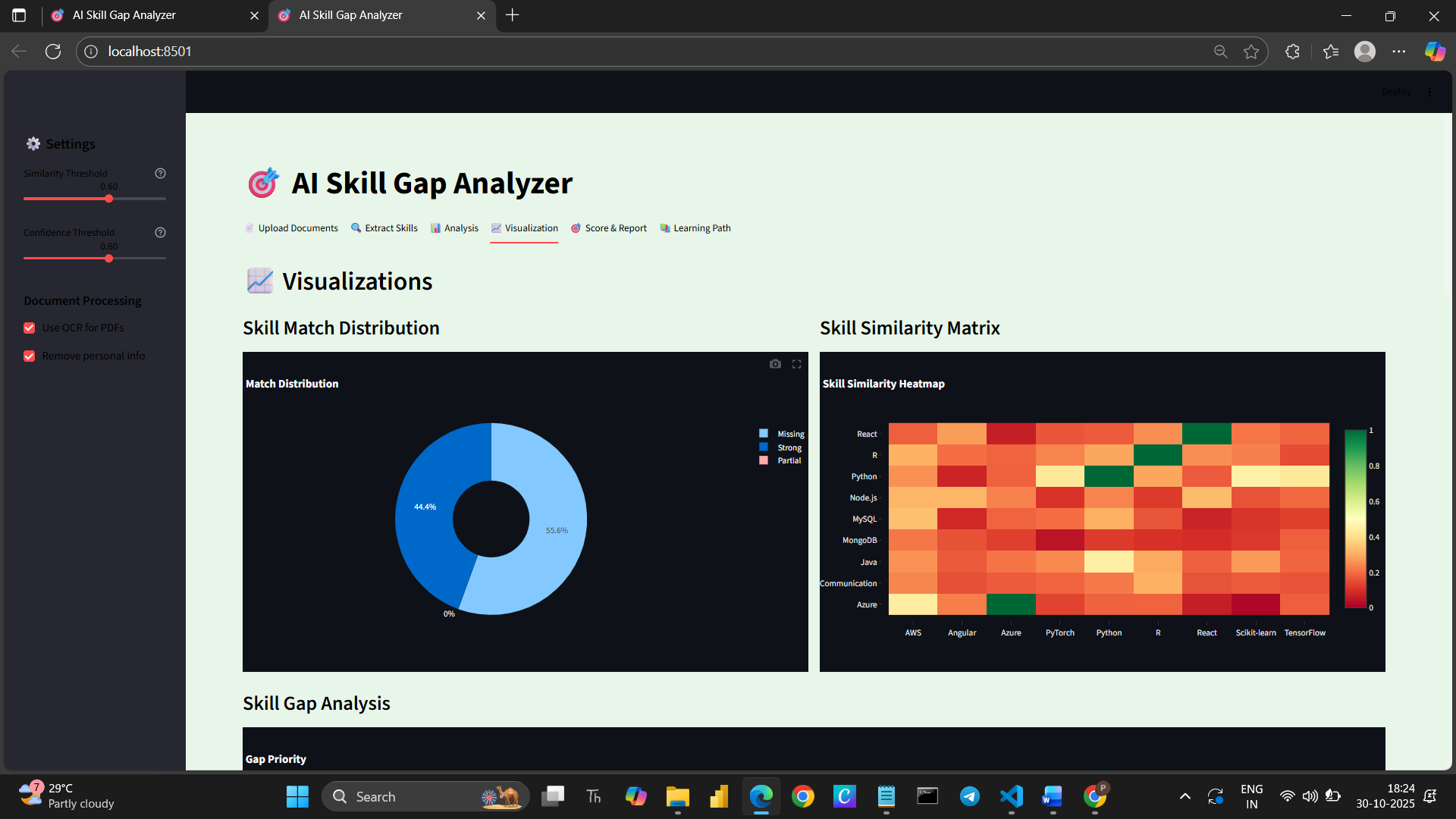
**8.screenshots**

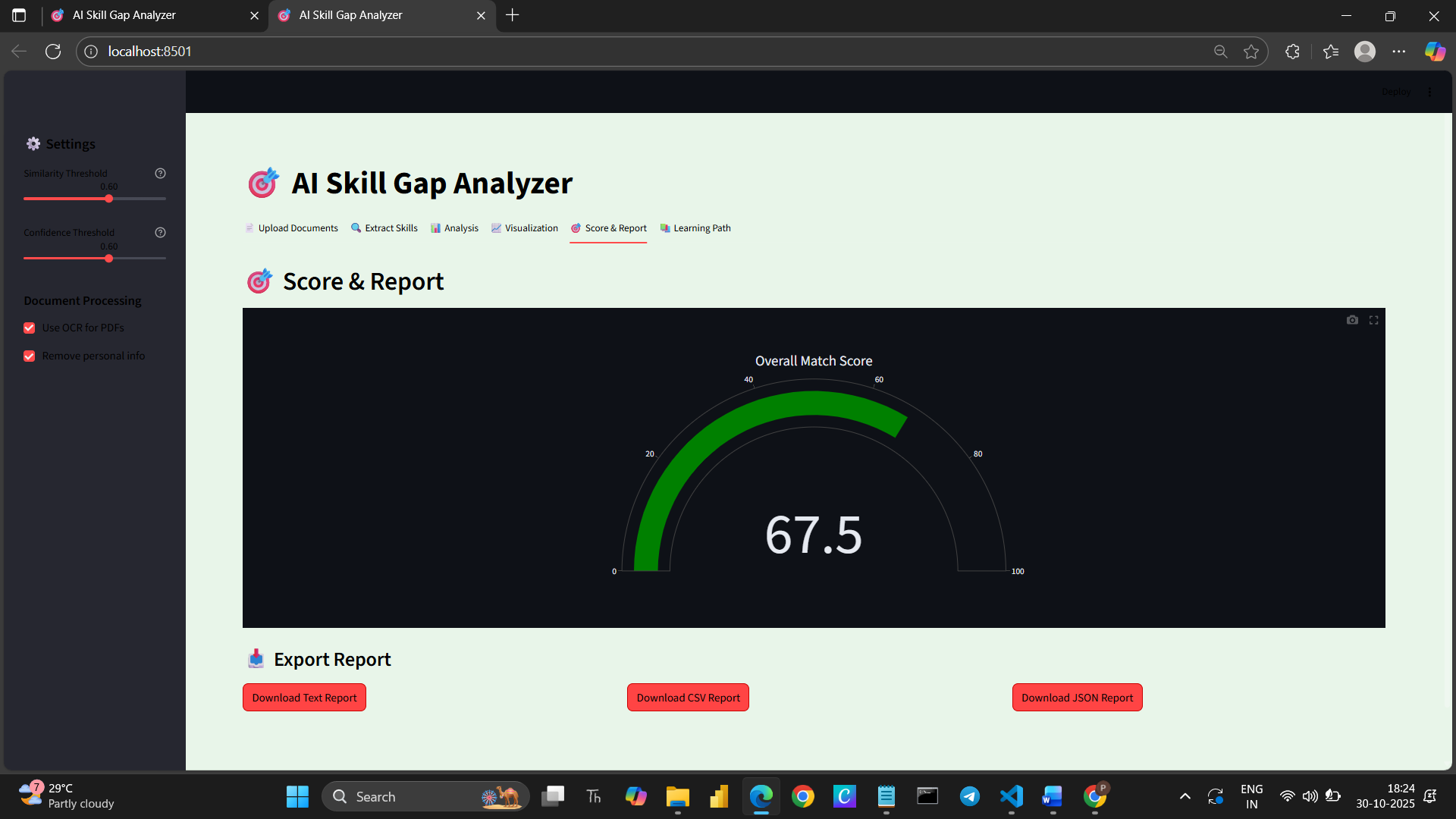


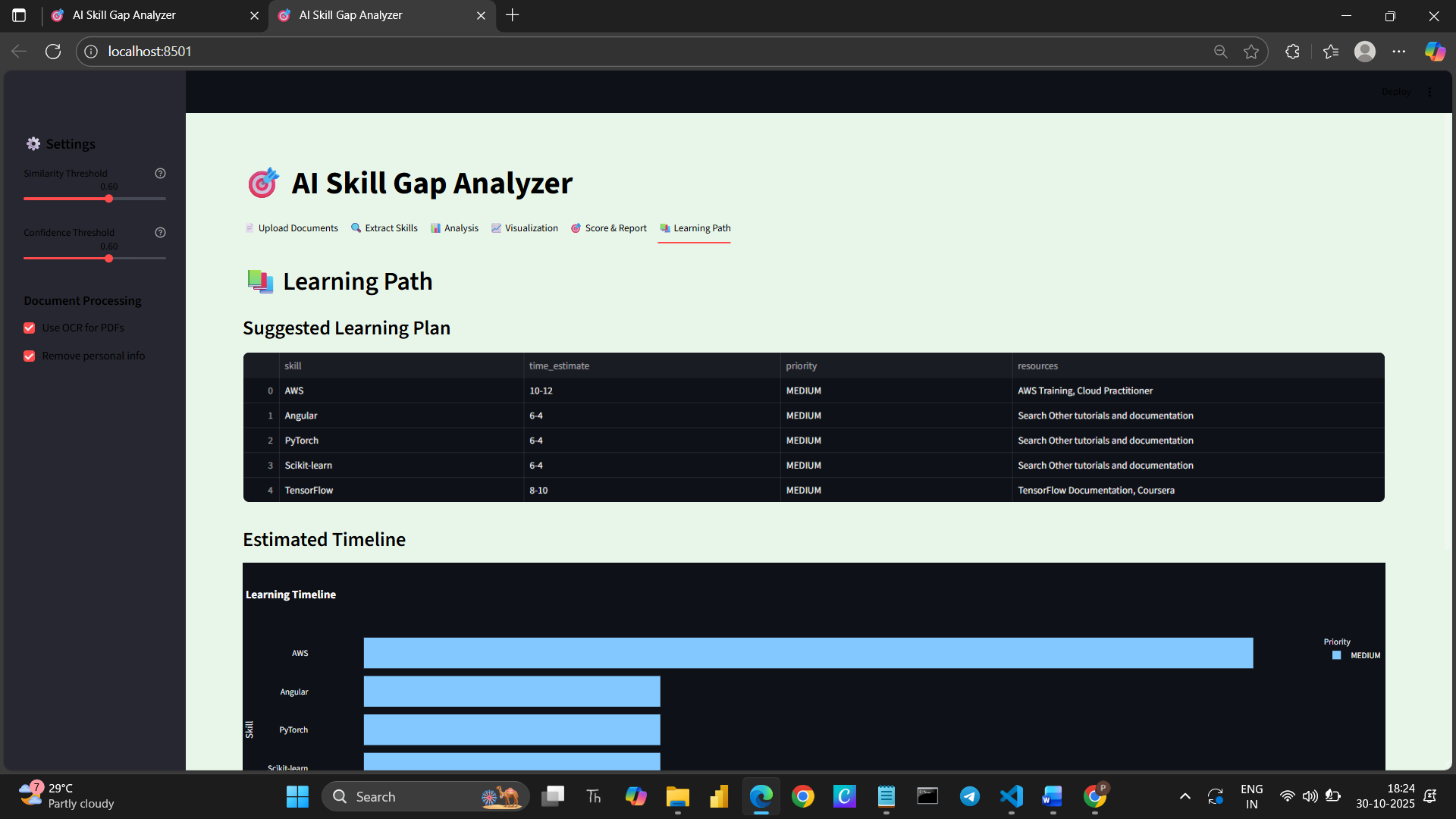












**9.Project Architecture**

The AI Skill Gap Analyzer follows a modular, layered architecture designed for scalability, maintainability, and clarity. The system integrates natural language processing, similarity computation, visualization, and report generation within a Streamlit-based web interface.

The architecture is divided into six primary layers:

1️⃣ **Input Layer – Document Processing**

Components:

* DocumentProcessor
* External libraries: PyPDF2, python-docx

Responsibilities:

* Handles file uploads for resumes and job descriptions (PDF, DOCX, TXT formats).
* Extracts raw text from documents.
* (Optionally) applies OCR for scanned PDFs and PII removal for data privacy.

Data Flow:  
User Upload → DocumentProcessor → Cleaned text output stored in st.session\_state.

2️⃣ **Skill Extraction Layer**

Components:

* SkillDatabase
* SkillExtractor

Responsibilities:

* Uses a curated keyword database to detect both technical and soft skills.
* Categorizes skills (e.g., programming\_languages, web\_frameworks, soft\_skills).
* Outputs confidence scores and category statistics.

Data Flow:  
Resume/JD Text → SkillExtractor.extract\_skills() → Skill sets & categorized results.

**3️⃣ Skill Matching & Gap Analysis Layer**

Components:

* SentenceBERTEncoder
* SimilarityCalculator
* SkillGapAnalyzer

Responsibilities:

* Converts skills into embeddings using Sentence-BERT (or TF-IDF fallback).
* Computes cosine similarity between resume and job description skill vectors.
* Classifies results as Matched, Partially Matched, or Missing based on thresholds.
* Calculates overall match score and category-level insights.

Data Flow:  
Extracted Skills → Embedding & Similarity → GapAnalysisResult object.

4️⃣ **Visualization Layer**

Components:

* GapVisualizer

Responsibilities:

* Generates interactive visualizations using Plotly:
  + Skill Similarity Heatmap
  + Match Distribution Pie Chart
  + Skill Gap Priority Bar Chart
  + Resume Skill Frequency and Category Charts
* Helps users visually interpret skill alignment and deficiencies.

Data Flow:  
GapAnalysisResult → Plotly Charts → Streamlit UI.

5️⃣ **Reporting Layer**

Components:

* ReportGenerator

Responsibilities:

* Generates exportable text, CSV, and JSON reports summarizing the analysis.
* Provides match statistics, missing skills, and overall similarity metrics for documentation or sharing.

Data Flow:  
GapAnalysisResult → ReportGenerator → Downloadable Reports.

**6️⃣ Learning Path Recommendation Layer**

Components:

* LearningPathGenerator

Responsibilities:

* Translates missing skills into personalized learning plans.
* Estimates time to learn, assigns priority levels, and suggests resources.
* Generates a timeline visualization to guide progressive upskilling.

Data Flow:  
Missing Skills → Learning Plan → Timeline Chart + CSV Export.

7️⃣ **User Interface Layer**

Framework:

* Streamlit Web App

Responsibilities:

* Manages page layout, navigation, and user interaction via six tabs:
  1. 📄 Upload Documents
  2. 🔍 Extract Skills
  3. 📊 Analysis
  4. 📈 Visualization
  5. 🎯 Score & Report
  6. 📚 Learning Path
* Provides real-time feedback, progress indicators, and interactive charts.
* Maintains persistent session data across tabs via st.session\_state.

**8️⃣ Data Flow Summary**

User Uploads Resume & JD

↓

DocumentProcessor (Text Extraction)

↓

SkillExtractor (Skill Detection)

↓

SkillGapAnalyzer (Embedding + Similarity + Gap Analysis)

↓

GapVisualizer & ReportGenerator (Charts + Reports)

↓

LearningPathGenerator (Personalized Learning Recommendations)

↓

Streamlit Interface (Visualization + Export)

**Technology Stack**

| Layer | Technology / Library |
| --- | --- |
| UI | Streamlit |
| Data Processing | pandas, numpy |
| Text Parsing | PyPDF2, python-docx |
| NLP & Similarity | SentenceTransformers, scikit-learn (TF-IDF, cosine\_similarity) |
| Visualization | Plotly, Streamlit |
| Reporting | JSON, CSV, Plain Text |
| Learning Recommendations | Custom heuristic logic |

**10.Project Completed**

The AI Skill Gap Analyzer provides a comprehensive and data-driven overview of your current skill alignment with the target job description. Through automated extraction, similarity analysis, and visualization, it identifies both strengths and areas for improvement.

Based on the analysis:

* **Strongly matched skills** indicate areas where your expertise aligns well with the job’s technical and soft skill requirements.
* **Partially matched skills** suggest domains where you have foundational knowledge that could be further developed to meet specific role expectations.
* **Missing skills** highlight critical learning opportunities to enhance employability and readiness for the targeted role.

The generated **learning path** offers a structured approach to bridge these gaps efficiently. By following the recommended resources and estimated timelines, you can progressively build proficiency in the most impactful areas.