AI Document Analyzer – Phase 1: Research Documentation

---

1. Project Title

AI Document Analyzer

---

2. Phase Title

Phase 1: Research Documentation

---

3. Objective

The objective of Phase 1 is to conduct an in-depth research and feasibility analysis for developing an **AI-powered Document Analyzer** that can process various document formats—**PDF, DOCX, TXT, and images**—and perform the following tasks:

- Extract text content
- Analyze sentiment
- Identify keywords and named entities
- Generate concise summaries
- Answer user-specific questions based on document content

The phase aims to define the **project scope**, identify **suitable tools and technologies**, and assess **technical feasibility** and potential risks.

---

4. Activities Performed

4.1 Problem Definition

- Identified the need for automating document analysis to reduce manual efforts in processing resumes, reports, and similar documents.
- Defined essential functionalities:
    - Text extraction
    - Sentiment analysis
    - Keyword and entity extraction

    - Summarization
    - Question answering
- Set success criteria:
    - 95% accuracy in text extraction
    - Reliable sentiment scoring
    - Relevant keyword/entity detection
    - Summaries that retain key points
    - Accurate and concise question responses

4.2 Market and Literature Review

- Reviewed tools like **IBM Watson NLU**, **Google Cloud NLP**, and open-source libraries such as **spaCy**, **NLTK**, and **transformers**.
- Analyzed academic papers and research on **Transformer-based models** like BERT for summarization and QA.
- Explored challenges in handling scanned PDFs, low-resolution images, and document noise.

4.3 Technology Stack Evaluation

- **Text Extraction:**
    - pdfplumber, PyMuPDF for PDFs
    - python-docx for DOCX
    - pytesseract for OCR from images
- **NLP Processing:**
    - IBM Watson NLU for cloud-based NLP
    - vaderSentiment for fallback sentiment analysis
    - Hugging Face transformers for summarization and QA
- **Web Interface:**
    - Flask selected for its simplicity and Python compatibility

- **Visualization:**
  - Considered matplotlib and Chart.js for representing keyword relevance and sentiment scores

---

4.4 Data Requirements

- Collected test documents (e.g., **Resume_Tejesh.pdf**)
- Defined test parameters:
  - Custom keywords: "project", "deadline", "duration"
  - Example questions: “What is the phone number?”, “What projects are mentioned?”
- Planned for preprocessing steps to clean and normalize extracted text

---

4.5 Feasibility and Risk Assessment

- **Risks Identified:**
  - API rate limits (for IBM Watson)
  - OCR errors for low-quality images
  - Latency in processing large documents
- **Mitigation Plans:**
  - Implement local NLP fallback
  - Optimize image preprocessing
  - Split and process large documents in chunks

---

5. Deliverables

- ✅ **Project Proposal** – Defines the objectives and proposed system capabilities
- ✅ **Technology Stack Report** – Comparison of NLP and extraction tools
- ✅ **Requirements Specification** – Functional and non-functional requirements
- ✅ **Sample Dataset** – Includes Resume_Musaib.pdf and others
- ✅ **Risk Analysis Document** – Lists risks and corresponding mitigations

---

6. Outcomes

- Project scope finalized with well-defined goals
- Tools and frameworks selected for the core features
- Identified core challenges and planned mitigation strategies
- Established success metrics such as:
    - 95%+ text extraction accuracy
    - <5 seconds system response time
    - Relevant QA and summarization output