Title: AI Document Analyzer and Keyword Extractor

**College Name: Nagarjuna College of Engineering and Technology**

**Team Members:**

1. Mohammed Musaib CAN\_32727772

The research phase aimed to establish a comprehensive understanding of the requirements, technologies, and methodologies necessary to develop an AI Document Analyzer capable of processing various document formats (PDF, DOCX, TXT, images) to extract text, perform sentiment analysis, identify keywords and entities, generate summaries, and answer user questions. The goal was to define the project scope, identify suitable tools, and assess feasibility.

**Activities**

1. **Problem Definition**:
   * Identified the need for an automated tool to analyze documents for insights, reducing manual effort in processing resumes, reports, and other text-based files.
   * Defined core functionalities: text extraction, sentiment analysis, keyword and entity extraction, text summarization, and question answering.
   * Established success criteria: accurate text extraction (>95% accuracy), reliable sentiment scoring, relevant keyword/entity identification, concise summaries, and precise question answering.
2. **Market and Literature Review**:
   * Reviewed existing tools like IBM Watson NLU, Google Cloud NLP, and open-source libraries (e.g., spaCy, NLTK) to benchmark capabilities.
   * Studied academic papers and industry reports on NLP techniques, focusing on transformer models (e.g., BERT) for summarization and question answering.
   * Analyzed challenges in multi-format document processing, such as handling scanned PDFs and noisy images.
3. **Technology Stack Evaluation**:
   * Evaluated libraries for text extraction: pdfplumber and PyMuPDF for PDFs, python-docx for DOCX, pytesseract for OCR on images.
   * Assessed NLP tools: IBM Watson NLU for sentiment, keywords, and entities; vaderSentiment for fallback sentiment analysis; transformers (Hugging Face) for summarization and question answering.
   * Selected Flask for the web interface due to its simplicity and Python compatibility.
   * Considered matplotlib and Chart.js for visualizing analysis results (e.g., keyword relevance charts).
4. **Data Requirements**:
   * Identified need for sample documents (e.g., resumes, reports) for testing, with Resume\_Musaib.pdf as a primary test case.
   * Defined custom keywords (e.g., “project,” “deadline”) and sample questions (e.g., “What is the phone number?”) for analysis.
   * Planned for data preprocessing to handle encoding issues and text noise.
5. **Feasibility and Risk Assessment**:
   * Assessed risks: API rate limits (IBM Watson), OCR inaccuracies for low-quality images, and model latency for large documents.
   * Confirmed feasibility based on availability of open-source libraries and cloud APIs, with fallback local processing for offline scenarios.

**Deliverables**

* **Project Proposal**: Document outlining the problem statement, objectives, and proposed functionalities.
* **Technology Stack Report**: Detailed comparison of tools and libraries, justifying selections.
* **Requirements Specification**: List of functional (e.g., multi-format support, question answering) and non-functional (e.g., response time <5 seconds) requirements.
* **Sample Dataset**: Collection of test documents, including Resume\_Musaib.pdf.
* **Risk Analysis**: Table of potential risks and mitigation strategies (e.g., local NLP fallback for API failures).

**Outcomes**

* Clear project scope defined, focusing on document analysis with NLP-driven insights.
* Technology stack finalized, balancing cloud APIs (IBM Watson) with local processing (vaderSentiment, transformers).
* Identified key challenges (e.g., PDF text extraction, sentiment accuracy) to address in design and development.
* Established baseline metrics for success, such as 95% text extraction accuracy and relevant keyword identification.

**Next Steps**

* Proceed to Phase 2: Design, to create system architecture and detailed workflows based on research findings.