**AI Document Analyzer: Phase 2 - Design Documentation**

**Objective**

The design phase focused on creating a detailed system architecture and workflow for the AI Document Analyzer, ensuring scalability, modularity, and alignment with research findings. The goal was to define how components (text extraction, NLP analysis, web interface) interact and address identified challenges (e.g., multi-format support, API reliability).

**Activities**

1. **System Architecture Design**:
   * Developed a modular architecture with three layers: presentation (Flask web interface), business logic (NLP processing), and data (file storage, logs).
   * Designed a client-server model where users upload documents via a web interface, and the server processes files and returns results.
   * Planned for local and cloud-based NLP processing, with IBM Watson NLU as the primary service and local fallbacks (vaderSentiment, regex-based extraction).
2. **Component Design**:
   * **Text Extraction Module**: Designed workflows for handling PDFs (pdfplumber, PyMuPDF), DOCX (python-docx), TXT (file I/O), and images (pytesseract with preprocessing).
   * **NLP Analysis Module**: Structured pipelines for sentiment analysis, keyword/entity extraction, summarization, and question answering, using IBM Watson, vaderSentiment, and transformers.
   * **Web Interface**: Planned a responsive UI with file upload, keyword input, result display (text, charts), and download options (TXT, JSON, CSV, PDF).
   * **Logging and Error Handling**: Designed logging system to capture processing details and errors in logs/analyzer.log and logs/app.log.
3. **Data Flow Design**:
   * Created data flow diagrams showing document upload, text extraction, NLP analysis, result storage, and user interaction (e.g., asking questions).
   * Planned temporary storage for uploaded files (static/uploads) and output files (static/outputs), with automatic cleanup post-processing.
4. **User Interface Mockups**:
   * Designed wireframes for the web interface, including a homepage with file upload and keyword input, a results page with analysis details, and a question-answering section.
   * Incorporated visualization elements, such as bar charts for keyword relevance using Chart.js.
5. **Error Handling and Fallbacks**:
   * Planned for API failures (e.g., IBM Watson rate limits) by implementing local NLP methods (regex for entities, vaderSentiment for sentiment).
   * Designed user feedback mechanisms for errors (e.g., “File too large,” “Invalid file format”) via web alerts.

**Deliverables**

* **System Architecture Diagram**: Visual representation of components and their interactions.
* **Data Flow Diagrams**: Detailed workflows for document processing and analysis.
* **UI Wireframes**: Mockups of the web interface, including upload, results, and question-answering sections.
* **Component Specifications**: Documentation for each module (text extraction, NLP, UI), detailing inputs, outputs, and dependencies.
* **Error Handling Plan**: Strategies for managing API failures, file errors, and user input issues.

**Outcomes**

* Comprehensive system design established, ensuring modularity and scalability.
* Clear workflows defined for multi-format document processing and NLP analysis.
* User-friendly interface planned to enhance accessibility and interaction.
* Mitigation strategies for technical challenges (e.g., API limits, OCR errors) integrated into the design.

**Next Steps**

* Move to Phase 3: Development, to implement the designed system using the selected technology stack.