AI Document Analyzer: Phase 3 – Development Documentation

Objective

The goal of the development phase was to implement all components defined in the design phase into a functional prototype. This included creating modules for multi-format document text extraction, NLP analysis (using both cloud APIs and local fallbacks), and a user-friendly web interface for interaction and visualization.

Activities

1. Environment Setup

- **Python Version:** 3.8+
- **Libraries Installed:**
    - pdfplumber, PyMuPDF for PDF handling
    - python-docx for DOCX processing
    - pytesseract, OpenCV for image OCR
    - transformers for BART and RoBERTa
    - vaderSentiment for local sentiment analysis
    - flask for the web interface
- **OCR Setup:** Installed and configured Tesseract-OCR
- **Project Directory Structure:**

2. Text Extraction Module

- **PDF Files:**
    - Used pdfplumber for high-quality text extraction.
    - Fallback to PyMuPDF for scanned PDFs.
- **DOCX Files:** Processed using python-docx.
- **TXT Files:** Read via standard Python file I/O.
- **Image Files (JPG, PNG):**

    - Preprocessed using OpenCV (grayscale, thresholding).
    - OCR performed using pytesseract.

---

3. NLP Analysis Module

- **Primary Processing: IBM Watson NLU**
    - Sentiment Analysis
    - Keyword Extraction
    - Entity Recognition
- **Fallbacks:**
    - vaderSentiment for local sentiment scoring
    - regex and spaCy for basic keyword/entity extraction
    - **Summarization:** BART transformer model
    - **Question Answering:** RoBERTa QA pipeline from Hugging Face
- **Custom Keyword Matching:**
    - User-defined terms (e.g., “deadline”) extracted using re and string matching.

---

4. Logging and Error Handling

- **Logging:**
    - Created logs/app.log and logs/analyzer.log for debugging and auditing
- **Errors Handled:**
    - Unsupported file types
    - Files > 10MB
    - IBM API timeout or failure
    - Missing or unreadable content
- **User Feedback:**
    - Implemented error banners/messages for invalid file uploads or missing answers

---

Deliverables

- ✅ **Working Prototype** (AI Document Analyzer Flask App)
- ✅ **Module Documentation** (docstrings + README.md)
- ✅ **Test Cases** using Resume_Musaib.pdf and others
- ✅ **Setup Guide** (for installing dependencies and running the app locally)

Outcomes

- Successfully developed a scalable, functional application with:
  - Multi-format document support (PDF, DOCX, TXT, images)
  - Robust NLP insights (sentiment, summary, keywords, Q&A)
  - Real-time feedback and downloadable results
- Identified minor issues:
  - API rate limits under free tier
  - PDF download format inconsistency (to be fixed in testing)