Title: AI Document Analyzer and Keyword Extractor

College Name: BMS Institute of Technology and Management

Team Members:

1. Priyanshu Sharma - CAN_35606461
2. Satish Mallappa B Patil – CAN_35717314
3. Kishor C – CAN_35990034
4. Albin Akkara – CAN_35608542

Overview:

This application allows users to upload documents in various formats (images, PDFs, and text files), extracts the textual content using OCR or direct reading, and applies Natural Language Processing (NLP) techniques to extract **keywords** and **entities** using IBM Watson Natural Language Understanding (NLU). The extracted results can be viewed on the web interface and downloaded as a report.

Functionality of the Application:

- Accept document uploads: .jpg, .jpeg, .png, .pdf, .txt.
- Extract text:
    - For images using **Tesseract OCR**.
    - For PDFs using **PyMuPDF**.
    - For text files using Python file reading.
- Perform NLP using **IBM Watson NLU**:
    - Extract **keywords** (most important phrases).
    - Identify **entities** (people, locations, organizations, etc.).
- Display the extracted text, keywords, and entities on the web interface.
- Allow users to **download a summary report**.

Services to Use:

- **IBM Watson Natural Language Understanding (NLU)**: For keyword and entity extraction.
- **IBM Cloud Object Storage**: To store uploaded files and logs (can be added).
- **IBM Watson Studio + Machine Learning** : For advanced document classification or retraining a custom NLU model.

Tools:

Tool/Library	Purpose
Flask	Backend framework to handle web app and routes
HTML + JavaScript	Frontend interface
pytesseract (Tesseract OCR)	Optical Character Recognition (OCR)
PyMuPDF (fitz)	PDF text extraction
IBM Watson NLU SDK	Access IBM's NLP services
Pillow	Image handling with Python

Steps for Project:

1. Set up the environment:

- Install Python libraries using pip:
    - flask, pytesseract, pillow, PyMuPDF, ibm-watson

2. Build the backend using Flask:

- Create app.py to handle:
    - Uploading files
    - Text extraction
    - Sending extracted text to IBM Watson NLU
    - Returning results to the frontend

3. Create frontend (HTML):

- Upload form to select files
- Display section for:
    - Extracted text
    - Keywords
    - Entities
- Button to download results

4. Connect to IBM Watson NLU:

- Use the IBM Cloud API Key and URL to send extracted text to Watson
- Receive JSON response with insights

5. Download report:

- Combine the results into a .txt file and provide a download link.

Tools and Services:

1. **IBM Watson Natural Language Understanding:**

- A cloud service that analyzes text to extract:
    - Keywords
    - Entities
    - Sentiment, emotion (optional)

2. **Tesseract OCR (pytesseract):**

- Converts printed or handwritten text in images into machine-readable text.

3. **Flask:**

- Lightweight Python web framework for building REST APIs and web apps.

4. **PyMuPDF (fitz):**

- Extracts text and images from PDF files efficiently.

5. **IBM Cloud Object Storage :**

- Secure cloud storage for storing uploaded documents, logs, or outputs.

6. **IBM Watson Studio :**

- Build, train, and deploy machine learning models. Used if extending with model training.

Reference Links:

- **IBM Watson NLU Docs**: https://cloud.ibm.com/apidocs/natural-language-understanding
- **IBM Watson SDK for Python**: https://github.com/watson-developer-cloud/python-sdk
- **Flask Documentation**: https://flask.palletsprojects.com/
- **Tesseract OCR GitHub**: https://github.com/tesseract-ocr/tesseract
- **PyMuPDF Docs**: https://pymupdf.readthedocs.io/