Project Documentation: Document-AI

### **Overview**

Document-AI is an Intelligent Document Processing (IDP) solution designed to extract essential information from invoice documents. By leveraging advanced data extraction techniques, the system identifies and retrieves key details such as invoice number, date, vendor name, total amount, and tax amount. The application is containerized using Docker, facilitating seamless deployment across various environments.

### **Workflow and Thought Process**

1. Problem Identification:
   * Recognized the need for an automated solution to process and extract data from invoices, reducing manual effort and minimizing errors.
2. Solution Design:
   * Data Extraction: Implemented algorithms to accurately detect and extract relevant fields from invoice images.
   * Containerization: Utilized Docker to package the application, ensuring consistency and ease of deployment.
3. API Development:
   * Developed a RESTful API endpoint to handle invoice image uploads and return extracted data in a structured format.
4. Testing and Validation:
   * Conducted thorough testing using various invoice samples to validate the accuracy and reliability of the extraction process.

### **Technical Documentation**

Architecture:

* The application follows a modular architecture comprising:
  + Data Extraction Module: Processes invoice images to extract key information.
  + API Layer: Handles client requests and responses.
  + Docker Container: Encapsulates the application for consistent deployment.

Dependencies:

* Python 3.9 or later: Primary programming language.
* OpenCV: For image processing tasks.
* Tesseract-OCR: Optical Character Recognition engine for text extraction.
* Flask: Web framework for building the RESTful API.
* Docker: Containerization tool to ensure consistent environments.

Setup Instructions:

1. Clone the Repository:

git clone https://github.com/kashinath12345/ASSIGNMENT.git

1. Navigate to the Project Directory:

cd ASSIGNMENT/

1. Set Up Virtual Environment:

python -m venv env

source env/bin/activate # On Unix/MacOS

env\Scripts\activate # On Windows

1. Install Dependencies:

pip install -r requirements.txt

1. Run the Application:

python app.py

1. Use Docker (Optional):
   * Build and start the Docker container:

docker build -t document-ai .

docker run -p 5000:5000 document-ai

Testing:

* Use tools like Postman to test the API endpoint for invoice processing.
* Verify the extracted fields (e.g., invoice number, date) against sample invoices.

### **Key Learnings from the Project**

1. Optical Character Recognition (OCR):
   * Gained a deeper understanding of text recognition techniques using Tesseract-OCR and OpenCV.
2. Containerization:
   * Learned how to package and deploy the application consistently across environments using Docker.
3. RESTful API Design:
   * Developed API endpoints to interact with the application, ensuring seamless communication between the client and server.
4. Testing and Validation:
   * Improved skills in validating application outputs against expected results using real-world data.

### **Time Spent on the Project**

* Research and Initial Setup: 3 hours
* OCR Development: 5 hours
* API Development: 3 hours
* Testing and Debugging: 4 hours
* Dockerization: 2 hours
* Documentation: 2 hours

### **Explanation of What Works Well**

1. Accurate Text Extraction:
   * The OCR engine accurately extracts key fields from invoice images, achieving high reliability.
2. API Functionality:
   * The RESTful API endpoints provide a user-friendly way to interact with the application.
3. Containerization:
   * Docker ensures the application runs consistently across different environments.

### **Known Limitations**

1. Limited Template Support:
   * The system may require additional configuration to handle non-standard invoice formats.
2. Resource Dependency:
   * The accuracy of text extraction is dependent on the quality of the invoice image.
3. Scalability:
   * While suitable for small-scale deployments, large-scale use may require optimization of the OCR process and database storage.