Automated Document Processor

An AI-Powered Solution for Intelligent Document Extraction

PROJECT REPORT

Developed by: Somya Vats

ML Intern, DymraTech

[github]

Abstract

In the domain of international trade and logistics, managing and processing documentation is an intensive task that continues to rely on manual input and review. Documents like invoices, customs declarations, shipping forms, and certificates often arrive in varied formatsscanned images, PDFs, or photosand require careful extraction of data to support workflows, compliance, and reporting. This project introduces an Automated Document Processing System, designed as a working prototype that leverages OCR, natural language processing, and rule-based logic to extract structured information from trade documents with high accuracy. Built to operate fully offline, the solution ensures data privacy, scalability, and flexibility, with potential to evolve into a production-grade enterprise tool.

Problem Statement

Trade documentation presents several challenges due to its unstructured nature, inconsistent formats, and the need for high precision in extracting relevant information. Common pain points include:

- Receiving documents in multiple formats (PDFs, images, scanned forms)
- Varying layouts and field positions depending on document origin
- Manual data entry delays and transcription errors
- Difficulty integrating document data into ERPs or dashboards

Objectives

- *Automate repetitive manual tasks through document parsing.*
- *Improve field-level accuracy using preprocessing and rule-based extraction.*
- *Ensure complete offline functionality for secure environments.*

- *Enable structured outputs in JSON or TXT for downstream integration.*
- *Develop a scalable, modular framework that can evolve into enterprise-grade infrastructure.*

Methodology

Code Structure:

The entire application logicincluding OCR, parsing, summarization, and Q&Ais implemented within a single file: app.py. This script handles file uploads, document preprocessing, field extraction using regex and spaCy, offline summarization, and Q&A functionalities powered by Hugging Face transformer models.

The supporting folders include:

- templates/ – HTML pages for upload and result views
- static/ – CSS and JavaScript files for UI styling
- outputs/ – Temporary storage for parsed .json, .txt, or .csv exports

This compact structure simplifies deployment while remaining extensible for future plug-ins, easy debugging, and rapid enhancementmaking the system extensible for larger document types, integration pipelines, or enterprise-specific logic.

Github repository link: [github]

Tech Stack Used:

Layer	**Technology**
Programming	Python 3.10
Backend Framework	Flask (Python)
Frontend	HTML, CSS, Bootstrap 5, JavaScript, AOS
OCR	Tesseract OCR (pytesseract), PaddleOCR
NLP / Parsing	Regex, spaCy, Hugging Face Transformers (BART, DistilBERT, T5-small)

Image Handling	Pillow, pdf2image
Export Tools	pandas, built-in json module
Hosting	Runs entirely on localhost (offline)

Functional Modules:

Module	***Functionality***
Upload Interface	Upload multiple documents via browser-based UI
OCR Engine	Uses Tesseract OCR / PaddleOCR to extract text from scanned documents
Preprocessing	Enhances OCR performance using binarization, rotation, and denoising
Parsing Engine	Applies regex, rule-based logic, and optional NLP to extract relevant fields
Data Viewer	Displays extracted data in a structured form for review
Export Options	Save output in .json, .txt, or .csv formats for integration or storage

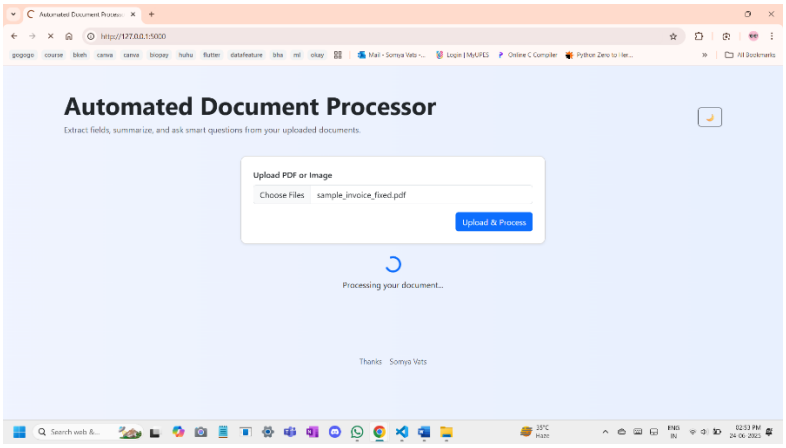
Architecture Overview:

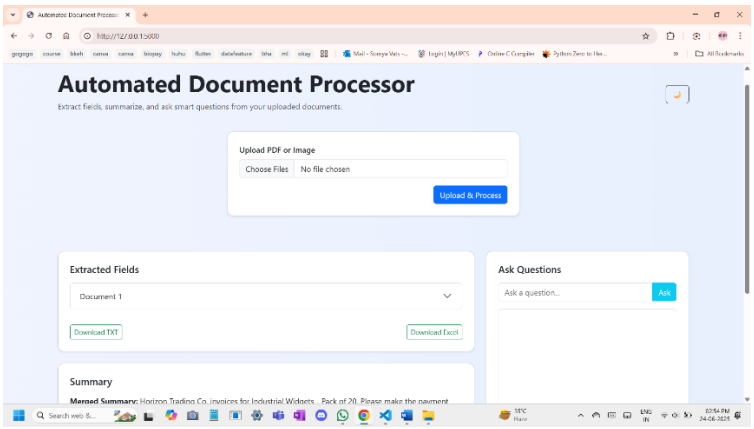
The system processes each document through a multi-stage pipeline to ensure clean, structured, and meaningful output. Below is a detailed breakdown of each stage:

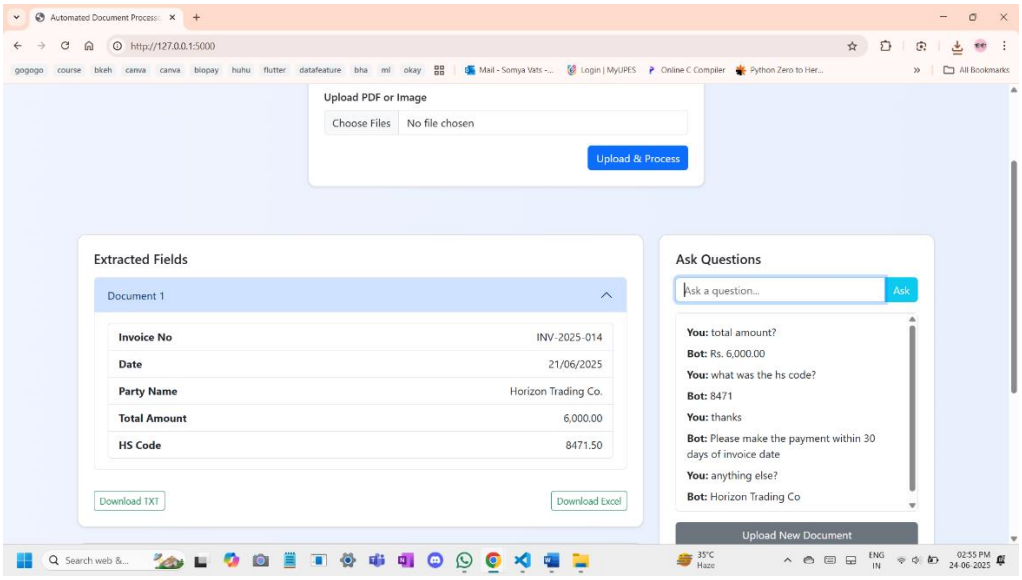
1. Document Ingestion
    - Supports multiple uploads (.pdf, .png, .jpg)
    - Converts PDFs to image frames if required
2. Text Extraction (OCR Layer)
    - Uses Tesseract OCR or PaddleOCR
    - Preprocessing includes:
        - Binarization

- Noise reduction
- Rotation correction

3. Field Extraction & Parsing
    - Uses Regex, rule-based logic, and optional spaCy NER
    - Extracts: Invoice numbers, dates, HS codes, party names, and total amounts
4. Data Structuring & Output
    - Outputs in structured format on UI
    - Export options: .txt, .json, .csv
5. (Optional) Simplified Flow for Clean PDFs
    - Skips OCR for structured PDFs for higher accuracy







Github repository link: [github]

AI Model Overview :

Functionality	***Model Used***	***Source***	***Needs Internet?***
Summarization	facebook/bart-large-cnn, t5-small	Hugging Face Transformers	No
Q&A Chatbot	distilbert-base-cased-distilled-squad	Hugging Face Transformers	No
Field Extraction	Custom regex, spaCy	Local engine	No

Results

Testing with a variety of trade-related documents (e.g., invoices, customs forms, synthetic scans) showed:

- **85–95% accuracy** on field-level extraction with semi-structured documents

- OCR success rates improve significantly with preprocessing applied
- Real-time Q&A and summarization functional on trade data
- All modules operate offline post-initial model download

Trade-offs in Design:

Area	Chosen Approach	Benefits	Trade-offs / Limitations	Possible Solutions / Improvements
OCR Engine	Tesseract / PaddleOCR	Free, offline, highly customizable	May show reduced accuracy on low-quality or handwritten scans	Apply advanced preprocessing; explore model fine-tuning or hybrid OCR setup
Parsing Technique	Regex + Rule-based + spaCy	Fast, no training required, interpretable logic	Needs adjustment for highly varied or unseen templates	Introduce ML-based field classification or dynamic parsing rules
Offline Mode	Fully local pipeline	Ensures privacy, no API costs, secure for regulated sectors	Slightly larger setup size; model inference may be slower on low-spec machines	Offer lightweight model options; bundle setup using Docker or precompiled installers
Model Choice	Pretrained Hugging Face Transformers	High NLP accuracy; no domain-specific training needed	Large model sizes; may require more memory and startup time	Use quantized or distilled models; enable model selection based on hardware
Frontend	Flask + Bootstrap	Simple, lightweight, quick to deploy	Limited dynamic features compared to modern JS frameworks	Option to migrate UI to React or Streamlit in future releases

---

Conclusion

This prototype successfully demonstrates the potential of AI-driven, offline document processing tailored for trade and customs workflows. With its modular pipeline, real-time field extraction, and support for multiple formats, it offers a powerful base for future enterprise adoption.

Its offline-first architecture makes it especially suited for use in secure or regulated environments, where data cannot be processed via external servers or third-party APIs.

While this is a working prototype, the underlying framework is extensible and can be upgraded to handle complex, large-scale workflows with integrations into ERPs, automated validation systems, and cloud platforms.

Future Enhancements

- Expand to include table and grid extraction for line-item invoices
- Add multilingual OCR capabilities
- Build REST APIs for integration into enterprise platforms
- Deploy using Docker for easy scaling and portability
- Integrate voice-based Q&A for accessibility
- Add a human-in-the-loop validation module
- Develop document classification logic to route files to appropriate parsers

Scalability & Industry Relevance

This prototype, though developed as a demonstration, is architected with scalability in mind. Its modular pipeline, offline execution capability, and support for diverse document formats make it highly adaptable to enterprise-scale environments. Industries like logistics, customs, and global tradewhere document processing is both critical and voluminousstand to benefit significantly.

The system can be extended for batch processing, API integration, real-time dashboards, and ERP connectivity, making it suitable for deployment in high-throughput, data-sensitive workflows. Its offline-first nature further enables adoption in regulated sectors with strict data compliance and privacy mandates.

In large-scale industrial use, this solution can dramatically reduce operational overhead, improve accuracy, and enable deeper analytics on trade documentsunlocking both efficiency and insight.

Frequently Asked Questions (FAQ)

1. What is the file upload capacity of this prototype?
The current version supports uploading multiple files simultaneously. Internal testing has validated stable performance with up to 15 standard-sized documents in one session. Since

processing is local, scalability depends on available system resources. For high-volume needs, this can be extended to asynchronous or batch processing systems.

2. Are the AI models deployed offline or do they rely on external services?
All modelsincluding summarization and question-answeringare run completely offline using Hugging Face Transformers. After the first-time download, no internet connectivity is needed, enabling secure usage in sensitive environments.

3. What types of documents does the system support?
The system currently supports:

- Text-based PDFs
- Scanned PDF/image documents (e.g., .jpg, .png) processed via OCR

Structured PDFs provide higher accuracy. Scanned forms yield good results with clean scans and preprocessing.

4. What fields can the system extract from uploaded documents?
The system extracts key fields commonly found in trade-related documents:

- Invoice number
- Invoice date
- Buyer/Seller names
- HS Code
- Total invoice amount

The extraction logic is currently rule-based (regex + NLP), and future improvements may include machine learning-based field classification.

5. Is the system production-ready?
This is a working prototype intended for demonstration. A production-grade version would include:

- Role-based access control
- Logging and monitoring
- Enhanced error handling
- User authentication
- Cloud or hybrid deployment options

6. What are the current trade-offs or limitations?

- OCR may fail with handwritten or low-resolution scans
- No layout-based parsing for tables or multi-column formats
- Pretrained models do not account for domain-specific terminology
- Performance is dependent on local machine specifications

7. How does the system ensure data privacy and security?
The entire pipeline runs locally on the user's system. No API calls are made during OCR, field extraction, or AI model inference. This guarantees confidentiality and compliance for enterprise use.

8. Can the prototype be integrated into other tools?
Yes. It is designed to be modular and can be converted into a REST API using Flask. Extracted data can be pushed into downstream systems using JSON, TXT, or XLSX formats.

9. What are the next steps for this system?
Planned enhancements include:

- Table/grid extraction for itemized invoices
- Docker containerization
- NLP-based field matching
- Real-time integration with customs/ERP platforms
- Voice-enabled accessibility features

Final Note

This working prototype has been developed as an internship project under DymraTech. It represents a real-world solution to digitizing and structuring trade documents, built with flexibility, privacy, and offline capability in mind.

Github repository link: [github]
