Document processing pipelines for AI

Vladimir Mijić

whoami

- Vladimir Mijić
- AI/ML Engineer @ LANACO
- BHFF alumnus and volunteer 

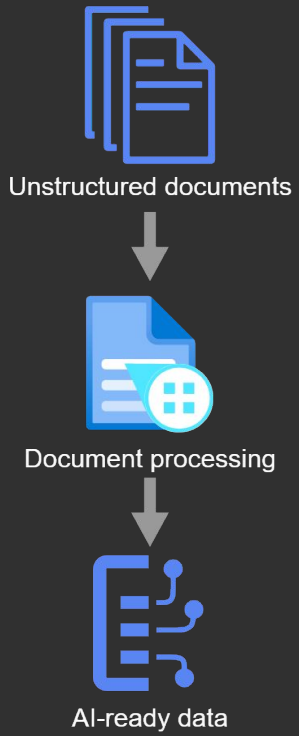
- Volunteer @ CODE4SEE 

Why document processing matters for AI

Organizations already have valuable knowledge.

The challenge is that much of it is locked inside documents.

Document processing makes that knowledge available as structured, searchable, and reliable data.



Why document processing matters for AI

Organizations already have valuable knowledge.

The challenge is that much of it is locked inside documents.

Document processing makes that knowledge available as structured, searchable, and reliable data.

What happens when the extracted data is incomplete or wrong?

Source: Ian Giles / LinkedIn

A new wave of Document AI

Document AI is improving quickly:

- stronger OCR and layout models
- Vision-Language Models for visual context
- LLMs for structured extraction
- managed services and open-source parsers
- growing demand from RAG, agents, and automation

A new wave of Document AI

Document AI is improving quickly:

- stronger OCR and layout models
- Vision-Language Models for visual context
- LLMs for structured extraction
- managed services and open-source parsers
- growing demand from RAG, agents, and automation

But better tools do not remove the need for a pipeline!

Why a single processing step is not enough

Documents can fail in different ways.

One tool is rarely enough to produce reliable AI-ready data.

A pipeline makes document processing more reliable, traceable, and easier to improve.

Documents are easy for humans, but hard for machines

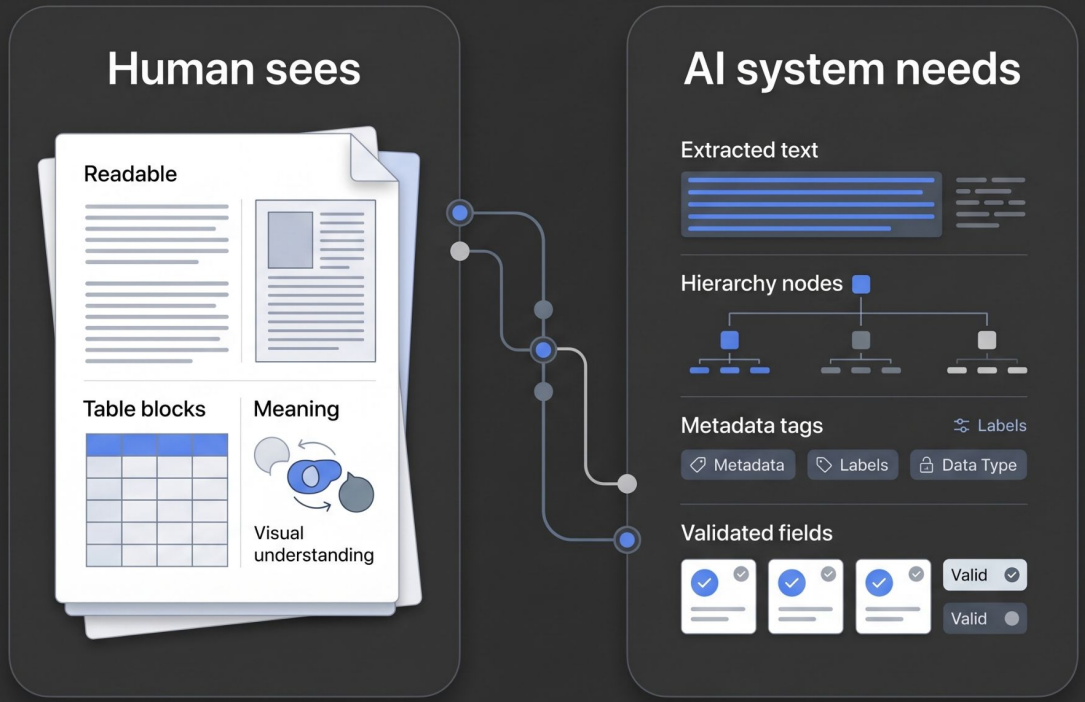


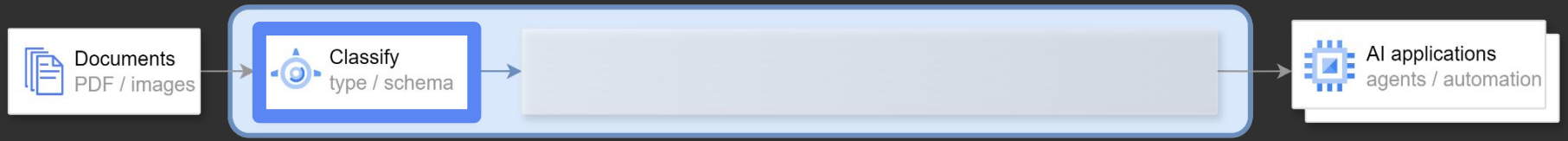
Image generated with Nano Banana

Different documents, different challenges

Type	Example	Main challenge
Digital PDF	Generated report	Text exists, but **structure** may be lost
Scanned PDF	Scanned contract or invoice	**OCR** is required
Image	Receipt, photo, form	**Quality, rotation, noise**
Form	Application or tax form	**Fields, checkboxes, labels**
Table-heavy document	Invoice or report	**Rows, columns, totals**
Mixed-layout document	Business report	**Reading order, sections, figures**


Documents
PDF / images
Document processing pipeline
AI applications
agents / automation

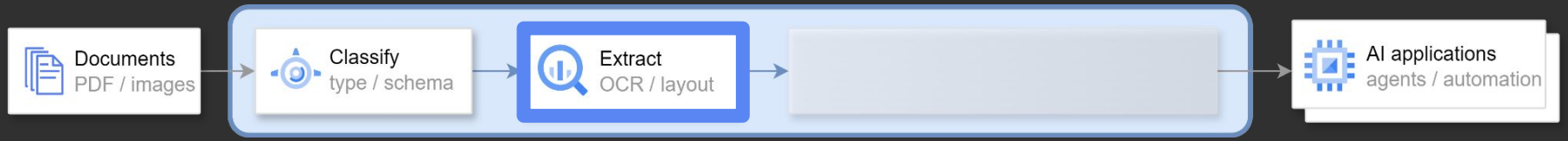
Document processing pipeline



Core processing steps:

1. Classify the document type

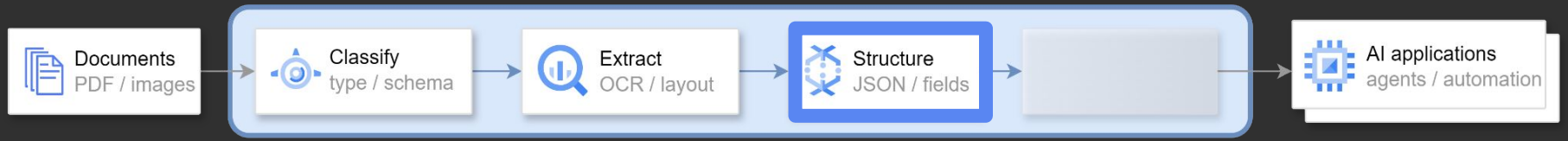
Document processing pipeline



Core processing steps:

1. Classify the document type
2. Extract the content

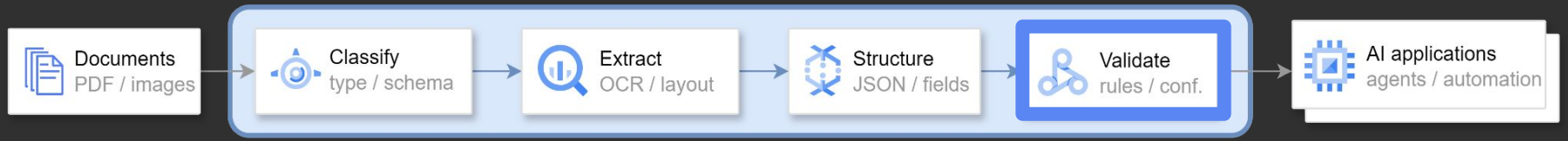
Document processing pipeline



Core processing steps:

1. Classify the document type
2. Extract the content
3. Structure the data

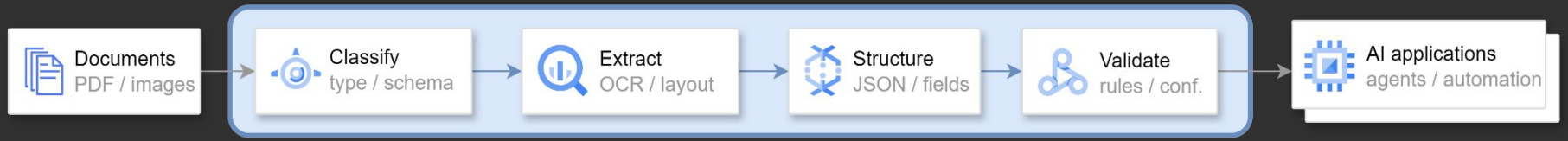
Document processing pipeline



Core processing steps:

1. Classify the document type
2. Extract the content
3. Structure the data
4. Validate the result

Document processing pipeline



Core processing steps:

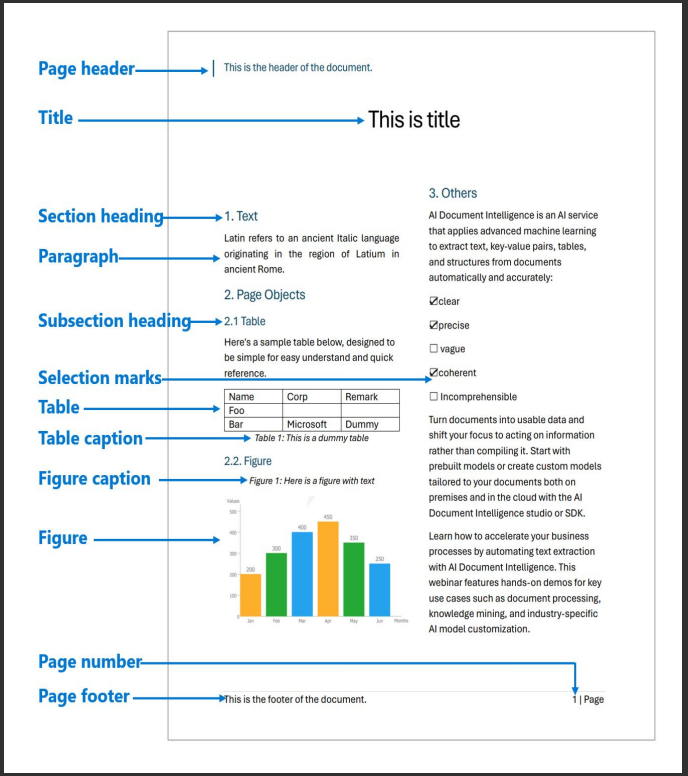
1. Classify the document type
2. Extract the content
3. Structure the data
4. Validate the result

Each step adds control before the data reaches the AI system.

Extract: what is on the page?

Extraction is about recovering the document content:

- text and OCR
- layout and reading order
- tables
- forms and key-value pairs
- images, figures, captions
- page metadata and confidence

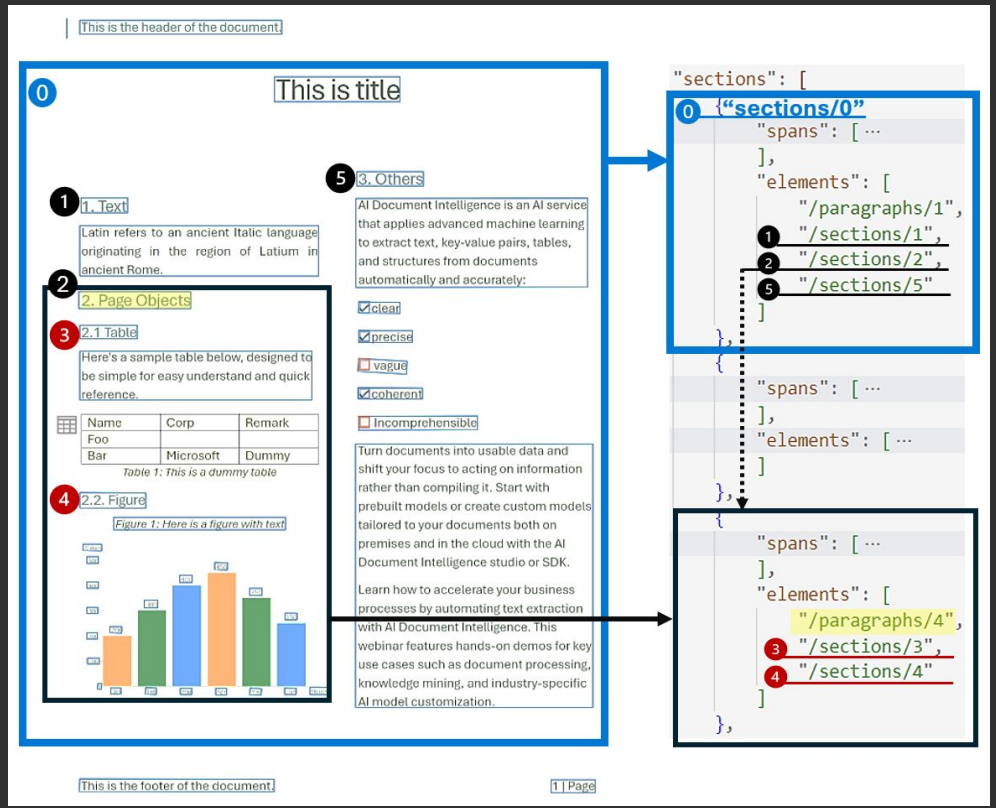


Source: Azure Document Intelligence docs

Structure: from content to data

Structuring turns extracted content into usable data:

- map content to a schema
- extract fields and entities
- convert tables into rows and columns
- create JSON output
- keep source references and confidence



Source: Azure Document Intelligence docs

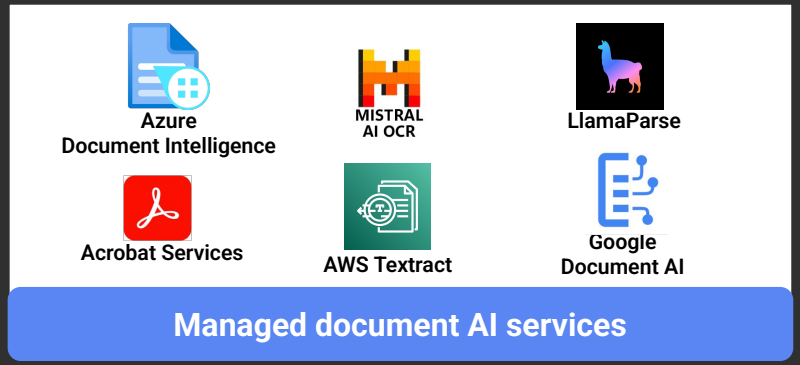
What the pipeline produces

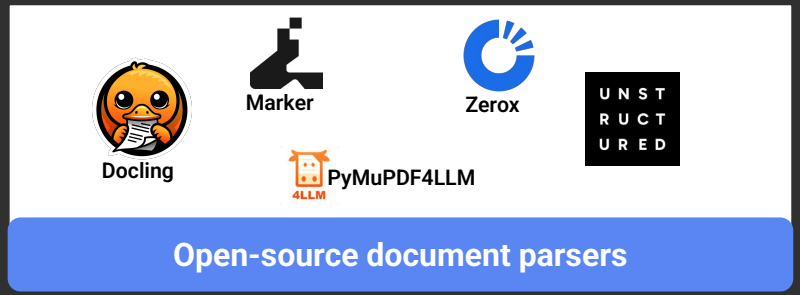
One document can produce multiple useful data representations.

Output	Purpose
Extracted text	Makes document content machine-readable
Layout metadata	Preserves pages, sections, positions, reading order
Tables	Keeps rows, columns, totals, and line items
Structured JSON	Provides fields and entities for applications
Validation results	Shows confidence, errors, and review status
Chunks + metadata	Supports agents, and semantic retrieval

Document AI tooling landscape

Selected examples - the ecosystem is moving fast!





Practical design decisions / trade-offs

There is no single best approach - it depends on the documents and the use case.

Cloud service or open-source stack?
Speed and managed infrastructure vs. control and customization.

OCR, layout model, LLM/VLM?
Choose based on document quality, layout complexity, and target output.

Text chunks, structured fields, or both?
RAG and agents need chunks; automation and analytics need structured data.
Preserve useful outputs for future use cases.

Fully automated or human-in-the-loop?
High-confidence cases can be automated; uncertain cases need review.

Generic or specialized pipeline?
Start simple, then specialize for high-value document types.

Document pipeline checklist

Input Assessment & Routing

- ❑ Identify document type and input quality
- ❑ Route by structure and complexity
- ❑ Select parser, OCR, VLM, or hybrid

Extraction & Structuring

- ❑ Extract text, layout, tables, and images
- ❑ Handle tables and forms separately
- ❑ Use predictable JSON schemas

Traceability & Context

- ❑ Keep raw text, structured output, metadata
- ❑ Preserve page references and coordinates
- ❑ Make fields traceable to the source

Quality & Operations

- ❑ Validate fields, formats, and business rules
- ❑ Track confidence, uncertainty, and errors
- ❑ Use human review when needed

Conclusion

Document processing is not just OCR.

Good pipelines transform raw documents into structured, searchable, and validated data.

A good design depends on the document type, document quality, and the target AI use case.

Use LLMs and VLMs as pipeline components - not the workflow owners.

Reliable document data leads to more reliable AI systems.

Start with a pipeline, then add intelligence where it matters. Keep automation controlled and reliable.

Useful resources and people to follow

Papers: LayoutLM, Donut, Nougat, DocLLM, ColPali, Dolphin, SmolDocling, PaddleOCR-VL

Conferences: ICDAR, CVPR

People: Lei Cui (Microsoft Research), Niels Rogge (Hugging Face), Luca Soldaini (Microsoft AI), Vik Paruchuri (Marker/Datalab)

Thank you!