Content Understanding

A look at its document processing capabilities

What is Content Understanding?

- Azure AI Content Understanding, built on the same foundational capabilities as Document Intelligence, extends **document** scenarios to **images** and embedded content, expanding to multimodal scenarios with **audio** and **video**.
- Content Understanding is built for content processing with Generative AI, improving your ability to generate the specific output you need with inferred fields, enrichments, validations, and reasoning.
  - **Inferred fields & enrichments**: Output required that aren't always directly present in the document, like the total tax on an invoice or the jurisdiction on a contract that can be inferred from the parties' addresses or clause wording.
  - **Multi-file input**: Process multiple input files in the same request and extract a unified schema across all the input files.
  - **Classification & Splitting**: Parse large files into individual documents for routing and schema extraction.
  - **Reasoning**: Intelligent document processing typically is a multi-step process with extraction, validation, aggregation, and reviews. Content Understanding is built for IDP, simplifying everything into a single step process.
  - **Post processing & validations**: Use the description to define any post processing rules like converting date formats, currency codes, and consistency checks.

What is Content Understanding for Document Processing?

- It is a way to enhance your OCR of documents by using an LLM for generating additional field values or metadata not actually in the document but is based on the fields that are in the document (or in supplemental content you provide in pro-mode).
- For generated fields it is like if you sent the document pages to the LLM yourself with a prompt to generate a value based on the fields in the document
- But without the need to:
  - Convert the document pages into images
  - Batch the documents to fit under image limits
  - Create and manage the prompt.
  - Managing the coordination of asynchronous API calls to multiple endpoints to accommodate your quota and response times
  - Merge the returned JSON results from batches into a single result

How a Content Understanding Classifier Works

A lower code approach to enhancing your OCR

The **Classifier Schema** describes the different document types using a prompt for each and defines which analyzer to use to do field extraction

This is transparent to you .
You make no calls to an LLM

LLM
All orchestration to and from LLM is handled automatically

Analyzers extracts fields from a document based on a Field Schema.
This is where the OCR is performed

Generated Fields

Extracted Fields

Classifier Schema

Classifier

What kind of document is this?

Routes to an analyzer based on classified doc type

Field Schema 1

Field Schema 2

Field Schema 3

Analyzer 1

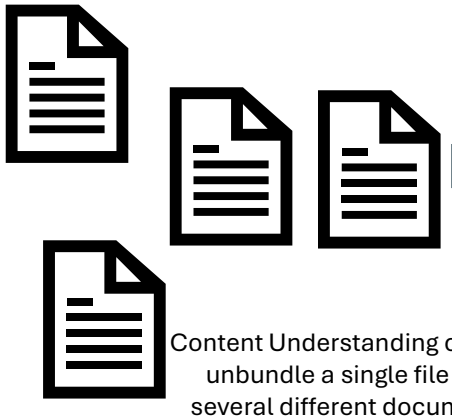
Analyzer 2

Analyzer 3

A **Field Schema** defines the fields to extract using a Description for each

If it is a generated field the Description becomes the **prompt**

The fields in the field schema may be schemas for different entity types or may all be the same where multiple document types each have some of the needed fields to extract to complete the entire schema for a single entity



Content Understanding can also logically unbundle a single file that contains several different documents into their individual documents automatically

{ Structured Output }

Many document types that look different may share the same analyzer because they are completing the same schema

Why Content Understanding for Document Processing?

- Schema based extraction using Templates
  - Multiple document types can share the same analyzer
- No dependency on field locations on a form
- Less fragile than parsing extracted text for fields
- Uses Document Intelligence like models to do the extraction
  - Specify “generate” or “extract” as the method in the field schema
  - “generate” will use an LLM to generate a field value
  - It doesn’t use the LLM to do the OCR
- Can extract top-level entity data and child entity data such as
  - an order and order items
  - or a bill and line-item charges
- Returns a structured output

When to choose Content Understanding over build your own model

Advantage	Azure AI Content Understanding	Build your own model
Unified, multimodal pipeline	✅ Supports docs, images, audio, video	❌ Requires orchestration
Enterprise reasoning workflows	✅ In-built reasoning capabilities	❌ Custom chaining
Prebuilt enrichments and schema normalization	✅ Prebuilt templates available	❌ Requires implementation
Simplified pricing	✅ Token based pricing	✅ Token based pricing
Enterprise governance & security	✅ Azure security compliance	❌ Custom implementation
Confidence and Grounding	✅ In-built scores	❌ Custom implementation
Chunking & normalization	✅ Built-in algorithms	❌ Custom implementation
Prompt tuning	✅ Optimized automatically	❌ Needs engineering
Context window	✅ Optimized for long files	❌ Manual handling

A use case example

Auditing of submitted medical charges for a claims processing company

Automatic unbundling of single file into constituent documents

Classifier

Schema (11 document types)

```
enhanced_classifier_with_document_metadata_and_fields_schema_9 = {
            "categories": {
                    "Completed_Claim_Form": {"description": "a Completed Claim Form", "analyzerId": analyzer_id_9C},
                    "HIPAA_Release": {"description": "a HIPAA Release", "analyzerId": analyzer_id_9A},
                    "Signed_Physician_Statement": {"description": "a Signed Physician Statement", "analyzerId": analyzer_id_9A},
                    "Pathology_Report": {"description": "a Pathology Report", "analyzerId": analyzer_id_9A},
                    "Doctor_Office_Visit_Report": {"description": "a Doctor Office Visit Report contains a narrative of the visit, in
                    "Scanner_Report": {"description": "a Scanner Report that list issue with the scan of the documents", "analyzerId"
                    "Other_Document_Type": {"description": "A document type other the other ones specified", "analyzerId": analyzer_i
                    "Itemized_Bill_for_Lab_Services": {"description": "an Itemized Bill from a laboratory for Lab test and services.
                    "Itemized_Bill_for_Radiology_Services": {"description": "an Itemized Bill from a radiology department for imaging
                    "Itemized_Bill_from_Other_Service_Providers_Type": {"description": "an Itemized Bill from a other than a laborato
                    "UB04_Bill": {"description": "A special type of itemized bill. It will have the notation on it UB04 or UB-04 or U
                    },
                "splitMode": "auto"  # IMPORTANT: Automatically detect document boundaries. Can change mode for your needs.
            }
# Make 2 columns in the output columns aligned
```

The **blue** text in the schema is the name of the analyzer to use for this type of document

Analyzer 1

Analyzer 2

Analyzer 3

Document Title

Itemized charges

Patient Information

Field Schema for non-bills, non-claim forms document types

```
"fieldSchema": {
    "fields": {
        "title_on_first_page_of_document": {
            "type": "string",
            "method": "generate",
            "description": "This is the title of the document. It will typically
        }
    }
```

Field Schema for bills

```
"Expenses": {
    "type": "array",
    "items": {
        "type": "object",
        "properties": {
            "Expense_Amount": {
                "type": "number",
                "method": "generate",
                "description": "A table of the expense items amounts bil
            },
            "ICD_Code": {
                "type": "string",
                "method": "generate",
                "description": "The ICD code associated with the expense
            },
            "Date": {
                "type": "date",
                "method": "generate",
                "description": "The date of the expense. The date is usu
            },
            "Expense_Description": {
                "type": "string",
                "method": "generate",
                "description": "The description of the expense. This may
            },
            "Surgeon_Name_or_Provider": {
                "type": "string",
```

Field Schema for claim forms

```
"fieldSchema": {
    "fields": {
        "title_on_first_page_of_document": {
            "type": "string",
            "method": "generate",
            "description": "This is the title of the document
        },
        "Patient_First_Name": {
            "type": "string",
            "method": "generate",
            "description": "The first name of the patient. Th
        },
        "Patient_Last_Name": {
            "type": "string",
            "method": "generate",
            "description": "The last name of the patient. Thi
        },
        "DOB": {
            "type": "string",
            "method": "generate",
            "description": "The DOB of the patient. This is
```

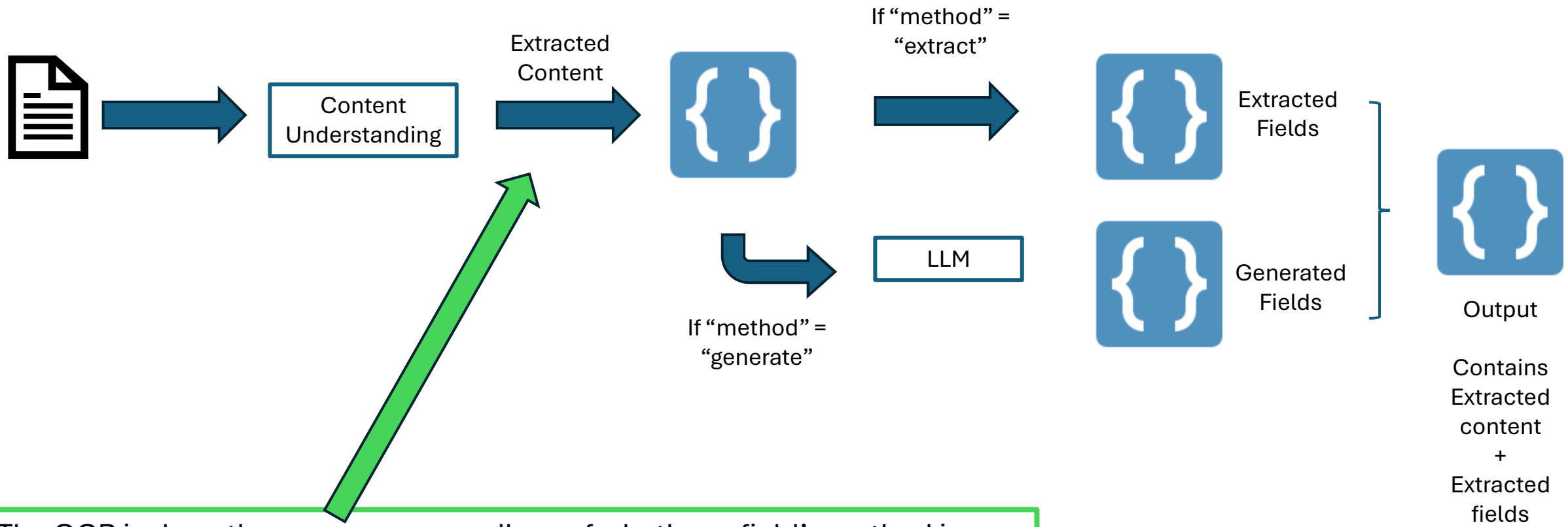
Generate vs extract

- Content Understanding's field extraction makes it easier to generate structured output from unstructured content.
- Define a schema to **extract**, **classify**, or **generate** field values with no complex prompt engineering
  - **Extract**: Directly extract values as they appear in the input content, such as dates from receipts or item details from invoices.
  - **Classify**: Classify content from a predefined set of categories, such as call sentiment or chart type.
  - **Generate**: Generate values freely from input data, such as summarizing an audio conversation or creating scene descriptions from videos.
- All extracted fields that are captured directly from the document use Document Intelligence like models.
- Only classify and generated model use an LLM. LLMs are not used for OCR.

Service capabilities

Capabilities	Document Intelligence	Content Understanding	Build Your Own with AOAI
OCR	**Industry leading OCR**	**Industry leading OCR**	Requires preprocessing
Complex document structure	Layout with tables, sections, selection marks, figures, and more	Layout with tables, sections, selection marks, figures, and more	Requires preprocessing
Extract fields	Yes	Yes	Yes
Confidence and Grounding	Yes	Yes	No, requires extra implementation
Inferred fields	No	**Yes, has support for generative and classify fields**	Yes
Generate metadata	No	**Yes**	Yes
Post-processing	Limited	Custom with limitations	User defined process
Process large files	Yes	Yes	Requires chunking and other strategies to get optimal performance
Ease of use	**Requires labeling and training** to build a custom model, can directly use layout and prebuilt models	**Simple schema definition without any labeling** required to get zero shot results	Optimize results with prompt engineering
Scale for use	Managed	Managed	Manually scale components as needed
Latency	Low	Medium	Depends on PTUs deployed
Multi-file inputs	No	Yes, support in multi file analysis or Pro Mode	No
Knowledge base	No	Yes	Complex and requires engineering
Reasoning	No	Yes, support in multi file analysis or Pro Mode	Complex and requires engineering

How it works

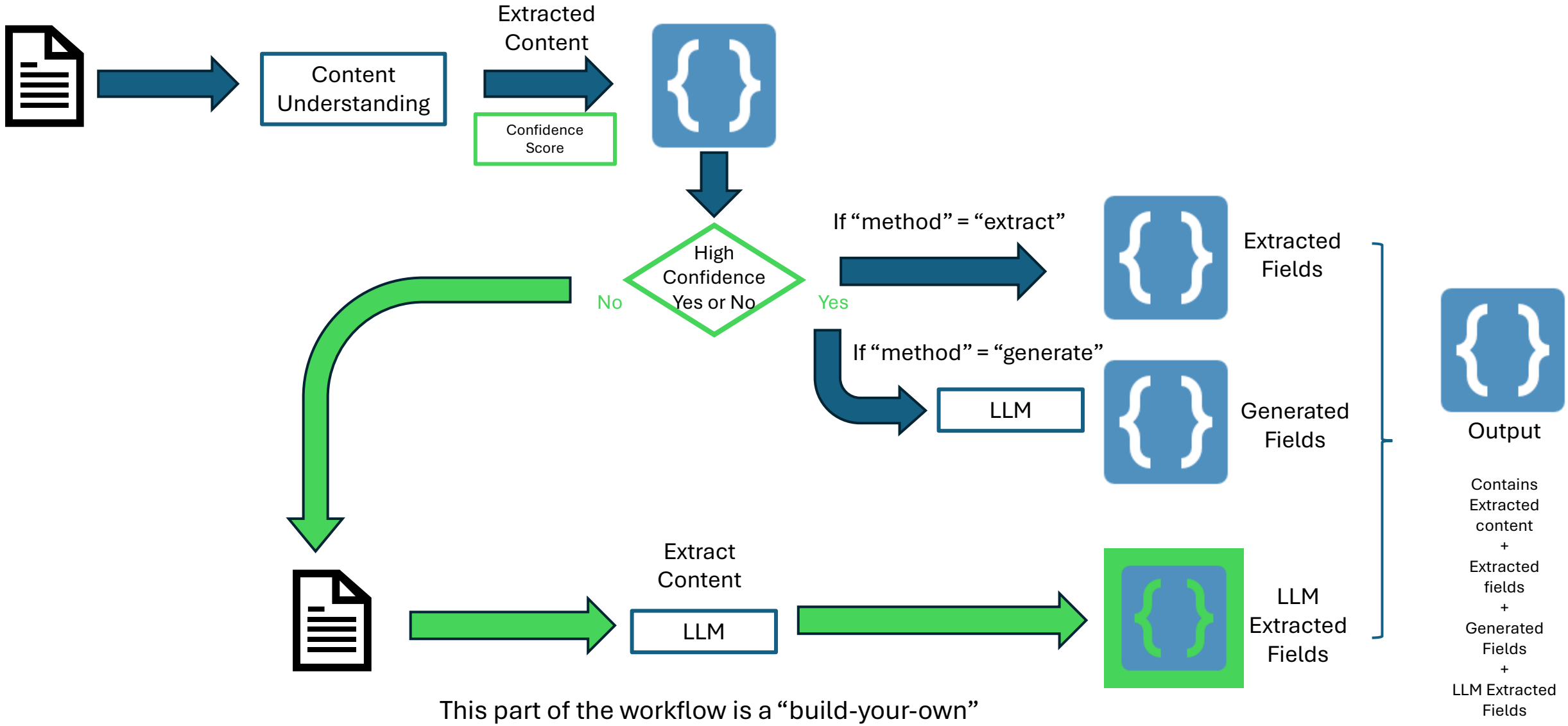


The OCR is done the same way regardless of whether a field’s method is marked “extract” or “generate”. For “generate” fields the OCR extract is then used as content and the description in the schema is used as a prompt to call a LLM to get a response providing the generated value for the field. (This is like what you get from Document Intelligence)

OCR performance

- Content Understanding provides the same quality of OCR as Document Intelligence.
- It provides confidence scores just like Document Intelligence does.
- It give you the capability to enhance the OCR with generated fields and classified fields.

How to handle pages with low confidence – a possible solution



This part of the workflow is a “build-your-own”

Resources

- What is Azure AI Content Understanding?
- Choose the right tool for document processing
- What's new
- Create an Azure AI Foundry resource
- Service quotas and limits
- Language and region support
- Pricing
- FAQ
- Glossary
- Document Modality
  - Overview
  - Elements 🆕
  - Markdown 🆕
  - Enrichments