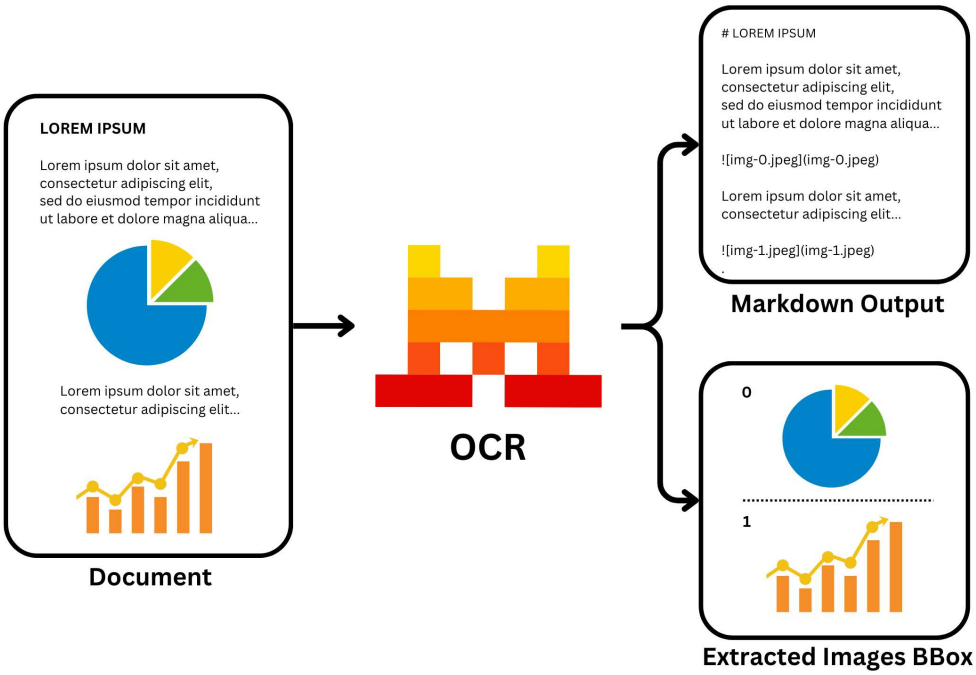
Basic OCR

Document AI OCR processor

Mistral Document AI API comes with a Document OCR (Optical Character Recognition) processor, powered by our latest OCR model `mistral-ocr-latest`, which enables you to extract text and structured content from PDF documents.



Key features:

- Extracts text content while maintaining document structure and hierarchy
- Preserves formatting like headers, paragraphs, lists and tables
- Returns results in markdown format for easy parsing and rendering
- Handles complex layouts including multi-column text and mixed content
- Processes documents at scale with high accuracy
- Supports multiple document formats including:
  - `image_url`: png, jpeg/jpg, avif and more...
  - `document_url`: pdf, pptx, docx and more...

The OCR processor returns the extracted **text content**, **images bboxes** and metadata about the document structure, making it easy to work with the recognized content programmatically.

OCR with PDF

python typescript curl

```python
import os
from mistralai import Mistral

api_key = os.environ["MISTRAL_API_KEY"]
client = Mistral(api_key=api_key)

ocr_response = client.ocr.process(
    model="mistral-ocr-latest",
    document={
        "type": "document_url",
        "document_url": "https://arxiv.org/pdf/2201.04234"
    },
    include_image_base64=True
)
```

Or passing a Base64 encoded pdf:

```python
import base64
import os
from mistralai import Mistral

def encode_pdf(pdf_path):
    """Encode the pdf to base64."""
    try:
        with open(pdf_path, "rb") as pdf_file:
            return base64.b64encode(pdf_file.read()).decode('utf-8')
    except FileNotFoundError:
        print(f"Error: The file {pdf_path} was not found.")
        return None
    except Exception as e:  # Added general exception handling
        print(f"Error: {e}")
        return None

# Path to your pdf
pdf_path = "path_to_your_pdf.pdf"

# Getting the base64 string
base64_pdf = encode_pdf(pdf_path)

api_key = os.environ["MISTRAL_API_KEY"]
client = Mistral(api_key=api_key)

ocr_response = client.ocr.process(
    model="mistral-ocr-latest",
    document={
        "type": "document_url",
        "document_url": f"data:application/pdf;base64,{base64_pdf}"
    },
```

```
    include_image_base64=True
)
```

▸ **Example output:**

OCR with uploaded PDF

You can also upload a PDF file and get the OCR results from the uploaded PDF.

Upload a file

python typescript curl

```python
from mistralai import Mistral
import os

api_key = os.environ["MISTRAL_API_KEY"]

client = Mistral(api_key=api_key)

uploaded_pdf = client.files.upload(
    file={
        "file_name": "uploaded_file.pdf",
        "content": open("uploaded_file.pdf", "rb"),
    },
    purpose="ocr"
)
```

Retrieve File

python typescript curl

```python
retrieved_file = client.files.retrieve(file_id=uploaded_pdf.id)
```

```
id='00edaf84-95b0-45db-8f83-f71138491f23' object='file' size_bytes=3749788
created_at=1741023462 filename='uploaded_file.pdf' purpose='ocr'
sample_type='ocr_input' source='upload' deleted=False num_lines=None
```

Get signed URL

python typescript curl

```
signed_url = client.files.get_signed_url(file_id=uploaded_pdf.id)
```

Get OCR results

python typescript curl

```
import os
from mistralai import Mistral

api_key = os.environ["MISTRAL_API_KEY"]
client = Mistral(api_key=api_key)

ocr_response = client.ocr.process(
    model="mistral-ocr-latest",
    document={
        "type": "document_url",
        "document_url": signed_url.url,
    },
    include_image_base64=True
)
```

OCR with image

python typescript curl

```
import os
from mistralai import Mistral

api_key = os.environ["MISTRAL_API_KEY"]
client = Mistral(api_key=api_key)

ocr_response = client.ocr.process(
    model="mistral-ocr-latest",
    document={
        "type": "image_url",
        "image_url":
"https://raw.githubusercontent.com/mistralai/cookbook/refs/heads/main/mistral/ocr/receipt.
    },
    include_image_base64=True
)
```

Or passing a Base64 encoded image:

```
import base64
import os
```

```python
from mistralai import Mistral

def encode_image(image_path):
    """Encode the image to base64."""
    try:
        with open(image_path, "rb") as image_file:
            return base64.b64encode(image_file.read()).decode('utf-8')
    except FileNotFoundError:
        print(f"Error: The file {image_path} was not found.")
        return None
    except Exception as e:  # Added general exception handling
        print(f"Error: {e}")
        return None

# Path to your image
image_path = "path_to_your_image.jpg"

# Getting the base64 string
base64_image = encode_image(image_path)

api_key = os.environ["MISTRAL_API_KEY"]
client = Mistral(api_key=api_key)

ocr_response = client.ocr.process(
    model="mistral-ocr-latest",
    document={
        "type": "image_url",
        "image_url": f"{base64_image}"
    },
    include_image_base64=True
)
```

Cookbooks

For more information and guides on how to make use of OCR, we have the following cookbooks:

- Tool Use
- Batch OCR

FAQ

Q: Are there any limits regarding the OCR API?
A: Yes, there are certain limitations for the OCR API. Uploaded document files must not exceed 50 MB in size and should be no longer than 1,000 pages.