

# Azure AI Document Intelligence documentation

Azure AI Document Intelligence is a cloud-based Azure AI service that uses machine-learning models to automate your data processing in applications and workflows. Document Intelligence is essential for enhancing data-driven strategies and enriching document search capabilities.

## About Azure AI Document Intelligence

### OVERVIEW

[What is Azure AI Document Intelligence?](#)

[Document Intelligence FAQ](#)

### WHAT'S NEW

[What's new in Azure AI Document Intelligence?](#)

## Document Intelligence models

### TRAINING

[Read](#)

[Layout](#)

[Financial Services and Legal](#)

[US Tax](#)

[Personal identification](#)

[Custom field extraction](#)

[Custom classification](#)

[Add-on capabilities](#)

[Query field extraction](#)

## Document Intelligence Studio

## OVERVIEW

[Studio overview](#)

[Studio concepts](#)

## Document Intelligence concepts

### CONCEPT

 [Retrieval-Augmented Generation \(RAG\)](#)

 [Batch document analysis](#)

[Accuracy and confidence scores](#)

[Use the analyzeDocument response](#)

[Custom model overview](#)

## Responsible AI

### REFERENCE

[Transparency notes](#)

[Characteristics and limitations](#)

[Guidance for integration and responsible use](#)

[Data privacy, compliance, and security](#)

## How-to-guides

### HOW-TO GUIDE

[Create a Document Intelligence resource](#)

[Create SAS tokens for storage containers](#)

[Create and use managed identities](#)

[Build a custom extraction model](#)

[Build a custom classification model](#)

[Compose custom models](#)

[Install and run Document Intelligence containers](#)

# What is Azure AI Document Intelligence?

Article • 02/06/2025

This content applies to: v4.0 (GA) | Previous versions: v3.1 (GA) v3.0 (GA) v2.1 (GA)

Azure AI Document Intelligence is a cloud-based [Azure AI service](#) that enables you to build intelligent document processing solutions. Massive amounts of data, spanning a wide variety of data types, are stored in forms and documents. Document Intelligence enables you to effectively manage the velocity at which data is collected and processed and is key to improved operations, informed data-driven decisions, and enlightened innovation.

For information on region access, see Azure AI Services [Product Availability by Region](#).

| [Document analysis models](#) | [Prebuilt models](#) | [Custom models](#) |

## Document analysis models

Document analysis (general extraction) models enable text extraction from forms and documents and return structured business-ready content ready for your organization's action, use, or development.

[Read](#) | Extract printed and handwritten text.

[Layout](#) | Extract text, tables, and document structure.

## Prebuilt models

Prebuilt models enable you to add intelligent document processing to your apps and flows without having to train and build your own models.

## Financial Services and Legal

[Bank Statement](#) | Extract account information and details from bank statements.

[Check](#) | Extract relevant information from checks.

[Contract](#) | Extract agreement and party details.

[Credit card](#) | Extract payment card information.

[Invoice](#) | Extract customer and vendor details.

[Pay Stub](#) | Extract pay stub details.

[Receipt](#) | Extract sales transaction details.

## US Tax

[Unified US tax](#) | Extract from any US tax forms supported.

[US Tax W-2](#) | Extract taxable compensation details.

[US Tax 1098](#) | Extract `1098` variation details.

[US Tax 1099](#) | Extract `1099` variation details.

[US Tax 1040](#) | Extract `1040` variation details.

## US Mortgage

[US mortgage 1003](#) | Extract loan application details.

[US mortgage 1004](#) | Extract information from appraisal.

[US mortgage 1005](#) | Extract information from validation of employment.

[US mortgage 1008](#) | Extract loan transmittal details.

[US mortgage disclosure](#) | Extract final closing loan terms.

## Personal Identification

[Health Insurance card](#) | Extract insurance coverage details.

[Identity](#) | Extract verification details.

[Marriage certificate](#) | Extract certified marriage information.

## Custom models

Custom models are trained using your labeled datasets to extract distinct data from forms and documents, specific to your use cases. Standalone custom models can be combined to create composed models.

## Document field extraction models

✓ Document field extraction models are trained to extract labeled fields from documents.

[Custom neural](#) | Extract data from mixed-type documents.

[Custom template](#) | Extract data from static layouts.

[Custom composed](#) | Extract data using a collection of models.

## Custom classification models

✓ Custom classifiers identify document types before invoking an extraction model.

[Custom classifier](#) | Identify designated document types (classes) before invoking an extraction model.

## Add-on capabilities

Document Intelligence supports optional features that can be enabled and disabled depending on the document extraction scenario:

- [ocr.highResolution](#)
- [ocr.formula](#)
- [ocr.font](#)
- [ocr.barcode](#)
- [Read model support for searchable PDF](#)
- [Searchable PDF](#)
- [queryFields](#)
- [keyValuePairs](#)

## Analysis features

 Expand table

Model ID	Content Extraction	Query fields	Paragraphs	Paragraph Roles	Selection Marks	Tables	Key-Value Pairs	Languages	Barcodes	Document Analysis	Formulas*
prebuilt-read	✓	✓					O	O			O
prebuilt-layout	✓	✓	✓	✓	✓	✓	O	O	O		O
prebuilt-contract	✓	✓	✓	✓	✓		O	O	✓	O	
prebuilt-healthInsuranceCard.us	✓	✓					O	O	✓	O	
prebuilt-idDocument	✓	✓					O	O	✓	O	
prebuilt-invoice	✓	✓			✓	✓	O	O	O	✓	O
prebuilt-receipt	✓	✓					O	O	✓	O	
prebuilt-marriageCertificate.us	✓	✓			✓		O	O	✓	O	
prebuilt-creditCard	✓	✓					O	O	✓	O	
prebuilt-check.us	✓	✓					O	O	✓	O	
prebuilt-payStub.us	✓	✓					O	O	✓	O	
prebuilt-bankStatement	✓	✓					O	O	✓	O	
prebuilt-mortgage.us.1003	✓	✓			✓		O	O	✓	O	
prebuilt-mortgage.us.1004	✓	✓			✓		O	O	✓	O	
prebuilt-mortgage.us.1005	✓	✓			✓		O	O	✓	O	
prebuilt-mortgage.us.1008	✓	✓			✓		O	O	✓	O	
prebuilt-mortgage.us.closingDisclosure	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.w2	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.w4	✓	✓					O	O	✓	O	
prebuilt-tax.us.1040 (various)	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.1095A	✓	✓					O	O	✓	O	
prebuilt-tax.us.1095C	✓	✓					O	O	✓	O	
prebuilt-tax.us.1098	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.1098E	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.1098T	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.1099 (various)	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.1099SSA	✓	✓					O	O	✓	O	
{ customModelName }	✓	✓	✓	✓	✓	✓	O	O	✓	O	

✓ - Enabled

O - Optional

\* - Premium features incur extra costs

## Models and development options

You can use Document Intelligence to automate document processing in applications and workflows, enhance data-driven strategies, and enrich document search capabilities. Use the links in the table to learn more about each model and browse development options.

### Read

Analyze | All pages | Range

Content Result Code

JSON

While healthcare is still in the early stages of its AI journey, we are seeing pharmaceutical and other life sciences organizations making major investments in AI and related technologies.

TOM LAWRY | National Director for AI Health and Life Sciences | Microsoft

As pharmaceutical and other life sciences organizations invest in and deploy advanced technologies, they are beginning to see broad and diverse areas across their organizations. Companies are looking at how AI can offer efficiencies and cost savings by investing in drug discovery, research and development, and manufacturing and supply chain management. Many life sciences organizations are also choosing to stay with more virtual approaches in the "new normal" – particularly in clinical trials and sales and marketing areas.

**Enhancing the patient and provider experience**

Clinical trial sponsors are continually seeking to make clinical trials faster and to improve the experience for patients and physicians. The COVID-19 pandemic has accelerated the adoption of decentralized clinical trials, with an increase in trial activities conducted remotely and in participants' homes. In McKinsey's survey<sup>1</sup> up to 91 percent of patients reported satisfaction with telemedicine. In the same report, 72 percent of physicians surveyed reported similar or better experiences with

Tandem was able to create and deploy this innovation by leveraging the AI and machine learning capabilities of the intelligent cloud. "We believe other technologies continue to advance, and our use cases will continue to 'Speed to value' going to continue to accelerate," said Lawry.

In addition, pharmaceutical and other life sciences companies can leverage advanced technologies to improve relationships with providers. For example, COVID-19 drove changes in the way companies interact with physicians. Prior to COVID-19, physicians were more likely to prefer in-person visits from medtech reps. Likewise, 77 percent of physicians preferred in-person sales visits from pharma reps.

Since the advent of COVID-19, however, physician preferences are moving toward virtual visits. Only 53 percent of physicians now express a preference for in-person visits from medtech reps and only 40 percent prefer in-person visits from pharma reps.<sup>2</sup> That puts the onus on pharmaceutical and life sciences organizations to deliver valuable and engaging virtual visits to providers.

While healthcare is still in the early stages of its AI journey, we are seeing pharmaceutical and other life sciences organizations making major investments in AI and related technologies." TOM LAWRY | National Director for AI Health and Life Sciences | Microsoft

257, 54, 826, 56, 826, 167, 257, 166

11, 12, 13, 14, 15, 16, 17, 18

{"pageNumber": 1, "angle": 0, "width": 915, "height": 1190, "unit": "pixel", "words": [ {

ed", "2023-02-21T19:27:23Z", "me": "2023-02-21T19:27:25Z", "022-08-31", "uilt-read", "": "utf16CodeUnit", "e healthcare is still in the early stages of its A

Expand table

Model ID	Description	Automation use cases	Development options
prebuilt-read	<ul style="list-style-type: none"> <li>Extract text from documents.</li> <li>Data extraction</li> </ul>	<ul style="list-style-type: none"> <li>Digitizing any document.</li> <li>Compliance and auditing.</li> <li>Processing handwritten notes before translation.</li> </ul>	<ul style="list-style-type: none"> <li>Document Intelligence Studio</li> <li>REST API</li> <li>C# SDK</li> <li>Python SDK</li> <li>Java SDK</li> <li>JavaScript</li> </ul>

Return to model types

## Layout

Analyze | All pages | Range

Content Result Code

NEWS TODAY

Latest news and bulletin updates

Role Content Polygon

title

NEWS TODAY Latest news and bulletin updates

139, 9, 608, 8, 608, 89, 139, 90

5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17

"analyzeResult": { "apiVersion": "2022-08-31", "modelId": "prebuilt-layout", "stringIndexType": "utf16CodeUnit", "content": "Tuesday, Sep 20, YYYY\nNEWS TODAY Latest new", "pages": [ { "pageNumber": 1, "angle": 0, "width": 756, "height": 1066, "unit": "pixel", "words": [ {

"succeeded", "ateTime": "2023-02-21T19:39:32Z", "tedDateTime": "2023-02-21T19:39:34Z",

Expand table

Model ID	Description	Automation use cases	Development options
prebuilt-layout	<ul style="list-style-type: none"> <li>Extract text and layout information from documents.</li> <li>Data extraction</li> </ul>	<ul style="list-style-type: none"> <li>Document indexing and retrieval by structure.</li> <li>Financial and medical report analysis.</li> </ul>	<ul style="list-style-type: none"> <li>Document Intelligence Studio</li> <li>REST API</li> <li>C# SDK</li> <li>Python SDK</li> <li>Java SDK</li> <li>JavaScript</li> </ul>

Return to model types

## Invoice

Analyze | All pages | Range

**CONTOSO LTD.**

**INVOICE**

Contoso Headquarters  
123 456th St  
New York, NY, 10001

Microsoft Corp  
123 Other St,  
Redmond WA, 98052

Customer Finance  
123 Bill St,  
Redmond WA, 98052

Shipped by  
123 Ship St,  
Redmond WA, 98052

Customer Services  
123 Service St,  
Redmond WA, 98052

Salesperson  
Requisitioner  
Shipped via  
F.O.B. point  
Terms

Date	Item Code	Description	Qty	U.M.	Price	Tax	Amount
3/4/2021	A123	Consulting Services	2	hours	\$30.00	\$6.00	\$60.00
3/5/2021	B456	Document Fee	3		\$10.00	\$3.00	\$30.00
3/6/2021	C789	Printing Fee	10	pages	\$1.00	\$1.00	\$10.00

THANK YOU FOR YOUR BUSINESS!

REMIT TO:  
Contoso Billing  
123 Remit St  
New York, NY, 10001

**Fields** Content Result Code

Prebuilt invoice Key-Value pairs

- INVOICE: INV-100 92.40%
- INVOICE DATE: 11/15/2019 90.80%
- DUE DATE: 12/15/2019 90.70%
- CUSTOMER NAME: MICROSOFT CORPORATION 88.90%
- SERVICE PERIOD: 10/14/2019 – 11/14/2019 87.40%
- CUSTOMER ID: CID-12345 90.70%

[Expand table](#)

Model ID	Description	Automation use cases	Development options
prebuilt-invoice	<ul style="list-style-type: none"> <li>Extract key information from invoices.</li> <li>Data and field extraction</li> </ul>	<ul style="list-style-type: none"> <li>Accounts payable processing.</li> <li>Automated tax recording and reporting.</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li>REST API</li> <li>C# SDK</li> <li>Python SDK</li> <li>Java SDK</li> <li>JavaScript</li> </ul>

[Return to model types](#)

## Receipt

Analyze | All pages | Range

**Contoso**

DocType: receipt.hotel

● ArrivalDate #1 99.40%  
2021-03-27

● Currency 99.50%  
USD

● DepartureDate #1 99.30%  
2021-03-28

● Items (6) #1 98.70%  
5600 148th Ave NE, Redmond, WA 98052

HouseNumber  
5600

Alex Morgan  
5600 148th Ave NE  
Redmond, WA 98052  
Contoso

Room: 515  
Room Type: S1Q1  
Number of Guests: 1  
Rate: \$127.00  
Clerk: MVB

Arrive: 2021-03-27 Time: 05:02PM Depart: 2021-03-28 Time: 02:52PM Folo Number: 12345

DATE	DESCRIPTION	CHARGES	CREDITS
ARRIVAL	ARRIVAL		
ROOM	ROOM		
MEALS	MEALS		
DRINKS	DRINKS		
WASH	WASH		
EXTRA	EXTRA		
REFUND	REFUND		

Total: 104.92  
Balance: 0.00 USD

**Fields** Result Code

[Expand table](#)

Model ID	Description	Automation use cases	Development options
prebuilt-receipt	<ul style="list-style-type: none"> <li>Extract key information from receipts.</li> <li>Data and field extraction</li> <li>Receipt model v3.0 supports processing of single-page hotel receipts.</li> </ul>	<ul style="list-style-type: none"> <li>Expense management.</li> <li>Consumer behavior data analysis.</li> <li>Customer loyalty program.</li> <li>Merchandise return processing.</li> <li>Automated tax recording and reporting.</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li>REST API</li> <li>C# SDK</li> <li>Python SDK</li> <li>Java SDK</li> <li>JavaScript</li> </ul>

[Return to model types](#)

## Identity (ID)

Analyze | All pages | Range
Fields Result Code



DocType: idDocument.passport

CountryRegion #1	99.00%
MYS	
DateOfBirth #1	99.00%
1975-11-02	
DateOfExpiration #1	99.00%
2022-03-10	
DateOfIssue #1	99.00%
2017-03-11	
DocumentNumber #1	99.00%
R00000000	
DocumentType #1	99.00%
P<MYSHAIRUDDIN<<AFIFAH<<<<<<<<<<<<	
R00000009MYS7511024F2203104751102076384<<04	

[Expand table](#)

Model ID	Description	Automation use cases	Development options
prebuilt-idDocument	<ul style="list-style-type: none"> <li>Extract key information from passports and ID cards.</li> <li><a href="#">Document types</a></li> <li>Extract endorsements, restrictions, and vehicle classifications from US driver's licenses.</li> </ul>	<ul style="list-style-type: none"> <li>Know your customer (KYC) financial services guidelines compliance.</li> <li>Medical account management.</li> <li>Identity checkpoints and gateways.</li> <li>Hotel registration.</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript</a></li> </ul>

[Return to model types](#)

## Check

Run analysis Analyze options
Fields Result Code

Drag & drop file here or Browse for files or Fetch from URL



check\_0731.pdf

Sample

check\_0731.pdf

Contoso Bank

Contoso Ltd. 123 Main St, Redmond, WA 98052

No. 370654 98 - 2  
Date: June 20, 2024 125

\$ \*\*\*123,456.00

Pay To The Order Of 22nd Century Insurance John Doe

One Hundred Twenty-Three Thousand Four Hundred Fifty-Six And 00/100 Dollars

Memo: Fees & Charges

Ammy Walker Authorized Signature

13 706 54 11 125 0000 24 0 84 9 50 00 55 4 3 2 1

PayerAddress #1	99.50%
123 Main St, Redmond, WA 98052	
HouseNumber	
123	
Road	
Main St	
PostalCode	
98052	
City	
NumberAmount	
Content	\$ 123,456.00
Value	123456
Confidence	99.50%
StreetAddress	
123 Main St	
PayerName #1	59.40%
Contoso Ltd.	
PayTo #1	99.50%
22nd Century Insurance John Doe	
WordAmount #1	99.50%
123456	

[Expand table](#)

Model ID	Description	Automation use cases	Development options
prebuilt-check	<ul style="list-style-type: none"> <li>Extract key information from checks.</li> <li>Data and field extraction</li> </ul>	<ul style="list-style-type: none"> <li>Credit management.</li> <li>Automated lender management.</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript</a></li> </ul>

[Return to model types](#)

## Pay stub

**CONTOSO LTD**

EARNINGS	HOURS	RATE	CURRENT	YTD
Regular Hours	56.00	24.00	\$ 1,344.00	\$ 5,000.00
Overtime	2.00	36.00	\$ 72.00	\$ 500.00
Sick Pay	8.00	24.00	\$ 192.00	\$ 500.00
Vacation Pay	8.00	24.00	\$ 192.00	\$ 500.00
Holiday Pay	8.00	24.00	\$ 192.00	\$ 500.00
	82.00		\$ 1,992.00	\$ 7,000.00

DEDUCTIONS	RATE	CURRENT	YTD
Aftertax Health Ins		\$ 120.00	\$ 500.00
Aftertax Dental Ins		\$ 35.00	\$ 400.00
Aftertax 401k		\$ 78.00	\$ 200.00
Child Support		\$ 155.00	\$ 200.00
Garnishment		\$ 45.00	\$ 500.00
		\$ 433.00	\$ 1,800.00

TAXES	RATE	CURRENT	YTD
Federal Income Tax		\$ 100.00	\$ 500.00
Social Security	0.0620	\$ 123.50	\$ 475.22
Medicare	0.0145	\$ 28.88	\$ 199.00
State Income Tax		\$ 50.00	\$ 200.00
Local Income Tax		\$ 10.00	\$ 40.00
		\$ 312.38	\$ 1,414.22

NET PAY	CURRENT	YTD
	\$ 1,246.61	\$ 3,785.78

Direct Deposited to Account of:  
Carl Anderson

Routing Number: 485066128      Account Number: 8300567112      Amount: \$ 1,246.61

---

**CONTOSO LTD**      **1/12/2024**      **1001**

123 Main Street  
Redmond, WA 98052

**EARNINGS STATEMENT**

Pay Date: **1/12/2024**  
Pay Start: **1/1/2024**  
Pay End: **1/12/2024**

Carl Anderson  
203 Denver Blvd  
Seattle, WA 98040

SSN: **997-60-1241**  
DOB: 01-09-1997

CurrentPeriodGrossPay #1      99.50%  
1992

CurrentPeriodNetPay #1      99.50%  
1246.61

CurrentPeriodTaxes #1      99.50%  
312.39

EmployeeAddress #1      99.50%  
203 Denver Blvd Seattle, WA 98040

HouseNumber  
203

Road  
Denver Blvd

PostalCode  
98040

City  
Seattle

State  
WA

StreetAddress  
203 Denver Blvd

[Expand table](#)

Model ID	Description	Automation use cases	Development options
prebuilt-paystub	<ul style="list-style-type: none"> <li>Extract key information from pay stubs.</li> <li>Data and field extraction</li> </ul>	<ul style="list-style-type: none"> <li>Employee payroll detail verification.</li> <li>Fraud detection for employment.</li> <li>Automated tax processing.</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript</a></li> </ul>

[Return to model types](#)

## Bank statement

**CONTOSO BANK**

Contoso Bank  
2096 Godfrey Road  
New York, NY 10021

Lela R. Duvall  
2096 Godfrey Road  
New York, NY 10021

**Account Number** 168552031585  
**Account Type** Simple Checking  
**Statement Period** Nov 1, 2023 – Nov 30, 2023

**Account Summary**

Beginning Balance on Nov 1, 2023	\$ 3,000.00
Deposits / Credits	+ 2,500.00
Withdrawals / Debits	- 1,200.00
Service Fees	- 10.00
<b>Ending Balance on Nov 30, 2023</b>	<b>\$ 4,290.00</b>

**Deposits / Credits**

Date	Description	Amount
11/01/2023	Deposit	\$ 2,000.00
11/20/2023	Deposit	\$ 500.00
<b>Total Deposits / Credits</b>	<b>\$ 2,500.00</b>	

**Withdrawals / Debits**

Date	Description	Amount
11/02/2023	Online Payment	-\$ 200.00
11/14/2023	Online Transfer To xxxxxxxxx1375	-\$ 1,000.00
<b>Total Withdrawals / Debits</b>	<b>-\$ 1,200.00</b>	

**Service Fees**

Date	Description	Amount
11/02/2023	International Transaction Fee	-\$ 10.00
<b>Total Service Fees</b>	<b>\$ 10.00</b>	

Page 1 of 1

Classified as Microsoft Confidential

DocType: bankStatement.us.layout

● AccountHolderAddress #1 99.50%  
2096 Godfrey Road New York, NY 10021

HouseNumber 2096

Road Godfrey Road

PostalCode 10021

City New York

State NY

StreetAddress 2096 Godfrey Road

● Transactions Content 11/02/2023 Online Payment -200.00 IderName #1 99.50%

Lela R. Duvall

● AccountNumber #1 99.50% 168552031585

● AccountType #1 99.50%

[ ] Expand table

Model ID	Description	Automation use cases	Development options
prebuilt-bankStatement	<ul style="list-style-type: none"> <li>Extract key information from bank statements.</li> <li>Data and field extraction</li> </ul>	<ul style="list-style-type: none"> <li>Tax Processing use cases.</li> <li>Automated accounting management.</li> <li>Credit-debit management.</li> <li>Loan documentation processing.</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript</a></li> </ul>

[Return to model types](#)

## Health insurance card

[ ] Run analysis | Analyze options

[ ] Fields Result Code

**PREMERA BLUE CROSS**

Member ANGEL BROWN  
Prefix Identification # Suffix ABC 123456789 01  
Group # 1000000 Rx Group # BCAAXY2 Rx BIN# 987654 BCBS 456

Medical Network HERITAGE  
Premera Dental YES  
Premera Vision YES

**HEALTH SAVINGS PLAN**  
Shared In and Out of Network  
Deductible \$1,500  
Coinsurance Max \$1,000

Note: Rx and Medical Cost-Shares are Shared

**PPO**

● Copays (2) #1 99.50%  
1 Amount \$1,500 Benefit Deductible

2 Amount \$1,000 Benefit Coinsurance Max

● GroupNumber #1 99.50% 1000000

● IdNumber #1 99.50% 123456789

Prefix ABC

● Insurer #1 99.50% PREMERA BLUE CROSS

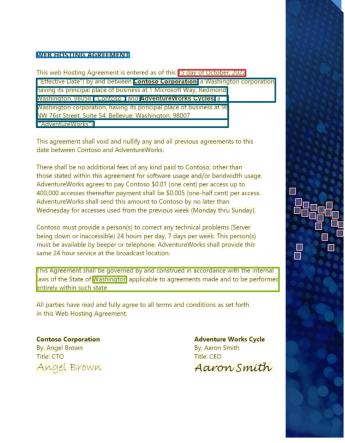
[ ] Expand table

Model ID	Description	Automation use cases	Development options
prebuilt-healthInsuranceCard.us	<ul style="list-style-type: none"> <li>Extract key information from US health insurance cards.</li> <li><a href="#">Data and field extraction</a></li> </ul>	<ul style="list-style-type: none"> <li>Coverage and eligibility verification.</li> <li>Predictive modeling.</li> <li>Value-based analytics.</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript</a></li> </ul>

[Return to model types](#)

## Contract model

Analyze | All pages | Range
Fields Content Result Code



DocType: contract
EffectiveDate #1 15 day of October, 2022
ExecutionDate #1 15 day of October, 2022
Jurisdictions #1 Clause This Agreement shall be governed by and construed in accordance with the internal laws of the State of Washington applicable to agreements made and to be performed entirely within such state.
Region Washington
Parties (2) #1 Title #1 WEB HOSTING AGREEMENT

[Expand table](#)

Model ID	Description	Development options
prebuilt-contract	Extract contract agreement and party details. <ul style="list-style-type: none"> <li><a href="#">Data and field extraction</a></li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript</a></li> </ul>

[Return to model types](#)

## Credit card model

Run analysis
Query fields
Analyze options

**Contoso Bank**

For customer services, call +1 200-345-6789 or +1 200-000-8888

NOT VALID UNLESS SIGNED

LOREM IPSUM DOLOR SIT AMET, CONSECTETUER ADIPISCING ELIT, SED DIAM NONUMY NIBH EUISMOD TINCIDUNT UT LAOREET DOLORE MAGNA ALIQUAM ERAT VOLUPAT.

Fields
Result
Code

DocType: creditCard

- CardHolderName #1 99.50%  
ADAM SMITH
- CardNumber #1 99.50%  
5412 1234 5656 8888
- CardVerificationValue #1 99.50%  
123
- CustomerServicePhoneNumbers (2) #1 ▾  
1 +1 200-345-6789  
2 +1 200-000-8888
- ExpirationDate #1 99.50%  
01/28
- IssuingBank #1 99.50%  
Contoso Bank
- PaymentNetwork #1 99.10%  
mastercard

[Expand table](#)

Model ID	Description	Development options
<a href="#">prebuilt-creditCard</a>	Extract contract agreement and party details. <ul style="list-style-type: none"> <li>● <a href="#">Data and field extraction</a></li> </ul>	<ul style="list-style-type: none"> <li>● <a href="#">Document Intelligence Studio</a></li> <li>● <a href="#">REST API</a></li> <li>● <a href="#">C# SDK</a></li> <li>● <a href="#">Python SDK</a></li> <li>● <a href="#">Java SDK</a></li> <li>● <a href="#">JavaScript</a></li> </ul>

[Return to model types](#)

## Marriage certificate model

Run analysis
Query fields
Analyze options

Fields
Result
Code

DocType: marriageCertificate.us

- IssueDate #1 99.50%  
March 22, 2017
- MarriageDate #1 98.60%  
19th March AD. 2017
- MarriagePlace #1 99.50%  
Detroit Wayne MICHIGAN,

[Expand table](#)

Model ID	Description	Development options
<a href="#">prebuilt-marriageCertificate.us</a>	Extract contract agreement and party details. <ul style="list-style-type: none"> <li>● <a href="#">Data and field extraction</a></li> </ul>	<ul style="list-style-type: none"> <li>● <a href="#">Document Intelligence Studio</a></li> <li>● <a href="#">REST API</a></li> <li>● <a href="#">C# SDK</a></li> </ul>

Model ID	Description	Development options
		<ul style="list-style-type: none"> <li>• Python SDK</li> <li>• Java SDK</li> <li>• JavaScript</li> </ul>

## US mortgage 1003 form

The screenshot shows the mortgage.us.1003 model interface. On the left is the Uniform Residential Loan Application form with various fields like Name, Social Security Number, Birth Date, Citizenship, and Marital Status. On the right is an analysis results table with columns for Fields, Result, and Code. The table shows the following data:

Fields	Result	Code
DocType	mortgage.us.1003	
AgencyCaseNumber	#1 115894	99.50%
Borrower	BirthDate 04 / 07 / 1989	99.50%
	CellPhoneNumber ( 831 ) 728 – 4766	90.20%
	NumberOfBorrowers 2	99.50%
	CurrentAddress 1634 W Glenoaks Blvd Glendale CA 91201 United States	99.50%
	HouseNumber 1634	
	Road W Glenoaks Blvd	

[Expand table](#)

Model ID	Description	Automation use cases	Development options
prebuilt-mortgage.us.1003	<ul style="list-style-type: none"> <li>• Extract key information from 1003 loan applications.</li> <li>• Data and field extraction</li> </ul>	<ul style="list-style-type: none"> <li>• Fannie Mae and Freddie Mac documentation requirements.</li> </ul>	<ul style="list-style-type: none"> <li>• Document Intelligence Studio</li> <li>• REST API</li> <li>• C# SDK</li> <li>• Python SDK</li> <li>• Java SDK</li> <li>• JavaScript</li> </ul>

[Return to model types](#)

## US mortgage 1004 form

Uniform Residential Appraisal Report

File #4L748221

The purpose of this summary appraisal report is to provide the lender/developer with an accurate and adequately supported opinion of the market value of the subject property.											
Property Address: 800 S. Crosby Way State: Colorado Zip Code: 80205 County: Denver State Tax ID: 00000000000000000000 Legal Description: 2500 sq. feet, 85.68% LTV Surveyor's Parcel #: 30-353-01 Tax Year: 2010 E.R. Taxes: \$1,000 Lot Reference: 477200 Cadastral: Section 1714 Special Assessments: \$ 0 PUD: No H.O.A.: No Condominium: No Assessor's Name: John Smith Assessor's Address: 123 Main Street, CA 80005 Assessor's Phone: (303) 555-1234											
Is the subject property currently offered for sale or has it been offered for sale in the twelve months prior to the effective date of this appraisal? <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No Report data source used: offing process, and/or <input checked="" type="checkbox"/> Data from public records											
Date of Contract: 01/01/2010 Did the agent did not analyze the contract for sale by the subject purchased transaction. Explain the reason of the analysis for contract for sale as why the analysis was not performed.											
The terms of the contract for sale indicates that the purchase price of lot 300 is 10% lower than current market value in the area.											
Contract Price: \$100,000.00 Date of Contract: 01/01/2010 Is the property owner of record of public record? <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No Data Sources: no record and Lien information Is there any financial assistance (tax changes, sale concessions, gift or government assistance, etc.) to be paid by party on behalf of the borrower? <input type="checkbox"/> Yes <input checked="" type="checkbox"/> No You repeat the total dollar amount and describe the item to be paid.											
<b>CONDOMINIUM</b>											
Note: Race and the racial composition of the neighborhood are not appraiser factors.											
Neighborhood Characteristics											
Neighborhood Trends											
One-Unit Housing Trends											
Neighborhood Boundaries											
Neighborhood Information: Resiliency of the neighborhood consists of income level individuals and families with active children.											
Market Conditions (including supply to the subject location)											
Recent sales activity: Notes activity in the neighborhood has been robust, with homes typical spending near days on the market compared to surrounding areas.											
Dimensions: W = 10' x L = 20'											
Area Amt: 200 Square Footage: 4000 Zoning: Residential View: N/A											
Specific Zoning Classification: Residential Zoning Description: Neighborhood, commercial											
Zoning Compliance: <input checked="" type="checkbox"/> Legal <input type="checkbox"/> Illegal Nonconforming (Grandfathered) <input type="checkbox"/> No Zoning <input type="checkbox"/> Legal (described)											
Is the highest and best use of the subject property as proposed for any planned and specifications the present use? <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No <input type="checkbox"/> If No, describe											
<b>Utilities</b> Public Other (describe) <b>Water</b> Public Other (describe) <b>Site Improvements</b> Type Public Private											
Electricity: <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No <input type="checkbox"/> Unknown <input type="checkbox"/> Metered <input type="checkbox"/> Unmetered <input type="checkbox"/> View: N/A											
Gas: <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No <input type="checkbox"/> Unknown <input type="checkbox"/> Metered <input type="checkbox"/> Unmetered <input type="checkbox"/> View: N/A											
Sewer: <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No <input type="checkbox"/> Unknown <input type="checkbox"/> Metered <input type="checkbox"/> Unmetered <input type="checkbox"/> View: N/A											
Water: <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No <input type="checkbox"/> Unknown <input type="checkbox"/> Metered <input type="checkbox"/> Unmetered <input type="checkbox"/> View: N/A											
FEMA Special Flood Hazard Area: <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No <input type="checkbox"/> Unknown <input type="checkbox"/> Map: N/A <input type="checkbox"/> FEMA Map: N/A											
Has the subject and site improvements typical for the market area? <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No <input type="checkbox"/> If No, describe											
Is the subject property located in an area with significant hazards such as flooding, earthquakes, environmental conditions, land uses, etc.? <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No <input type="checkbox"/> If Yes, describe											
Is the subject property encumbered on subject?											
<b>General Description</b> <b>Foundation</b> <b>Exterior Description</b> <b>Material/Condition</b> <b>Interior</b> <b>Material/Condition</b>											
Jobs: <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No <input type="checkbox"/> Unknown <input type="checkbox"/> Foundation: Stone <input type="checkbox"/> Floors: Carpet <input type="checkbox"/> Plaster: Drywall											
Lot: <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No <input type="checkbox"/> Unknown <input type="checkbox"/> Foundation: Stucco <input type="checkbox"/> Floors: Vinyl <input type="checkbox"/> Plaster: Gypsum											
Type: <input checked="" type="checkbox"/> Single Family <input type="checkbox"/> Multi-Family <input type="checkbox"/> Manufactured <input type="checkbox"/> Foundation: Galvanized <input type="checkbox"/> Bath: Ceramic <input type="checkbox"/> Plaster: Drywall											
Year Built: <input checked="" type="checkbox"/> 1980 <input type="checkbox"/> 1981-1990 <input type="checkbox"/> 1991-2000 <input type="checkbox"/> 2001-2010 <input type="checkbox"/> 2011-2012 <input type="checkbox"/> 2013-2014 <input type="checkbox"/> 2015-2016 <input type="checkbox"/> 2017-2018 <input type="checkbox"/> 2019-2020 <input type="checkbox"/> 2021-2022 <input type="checkbox"/> 2023-2024 <input type="checkbox"/> 2025-2026 <input type="checkbox"/> 2027-2028 <input type="checkbox"/> 2029-2030 <input type="checkbox"/> 2031-2032 <input type="checkbox"/> 2033-2034 <input type="checkbox"/> 2035-2036 <input type="checkbox"/> 2037-2038 <input type="checkbox"/> 2039-2040 <input type="checkbox"/> 2041-2042 <input type="checkbox"/> 2043-2044 <input type="checkbox"/> 2045-2046 <input type="checkbox"/> 2047-2048 <input type="checkbox"/> 2049-2050 <input type="checkbox"/> 2051-2052 <input type="checkbox"/> 2053-2054 <input type="checkbox"/> 2055-2056 <input type="checkbox"/> 2057-2058 <input type="checkbox"/> 2059-2060 <input type="checkbox"/> 2061-2062 <input type="checkbox"/> 2063-2064 <input type="checkbox"/> 2065-2066 <input type="checkbox"/> 2067-2068 <input type="checkbox"/> 2069-2070 <input type="checkbox"/> 2071-2072 <input type="checkbox"/> 2073-2074 <input type="checkbox"/> 2075-2076 <input type="checkbox"/> 2077-2078 <input type="checkbox"/> 2079-2080 <input type="checkbox"/> 2081-2082 <input type="checkbox"/> 2083-2084 <input type="checkbox"/> 2085-2086 <input type="checkbox"/> 2087-2088 <input type="checkbox"/> 2089-2090 <input type="checkbox"/> 2091-2092 <input type="checkbox"/> 2093-2094 <input type="checkbox"/> 2095-2096 <input type="checkbox"/> 2097-2098 <input type="checkbox"/> 2099-20100 <input type="checkbox"/> 20101-20102 <input type="checkbox"/> 20103-20104 <input type="checkbox"/> 20105-20106 <input type="checkbox"/> 20107-20108 <input type="checkbox"/> 20109-20110 <input type="checkbox"/> 20111-20112 <input type="checkbox"/> 20113-20114 <input type="checkbox"/> 20115-20116 <input type="checkbox"/> 20117-20118 <input type="checkbox"/> 20119-20120 <input type="checkbox"/> 20121-20122 <input type="checkbox"/> 20123-20124 <input type="checkbox"/> 20125-20126 <input type="checkbox"/> 20127-20128 <input type="checkbox"/> 20129-20130 <input type="checkbox"/> 20131-20132 <input type="checkbox"/> 20133-20134 <input type="checkbox"/> 20135-20136 <input type="checkbox"/> 20137-20138 <input type="checkbox"/> 20139-20140 <input type="checkbox"/> 20141-20142 <input type="checkbox"/> 20143-20144 <input type="checkbox"/> 20145-20146 <input type="checkbox"/> 20147-20148 <input type="checkbox"/> 20149-20150 <input type="checkbox"/> 20151-20152 <input type="checkbox"/> 20153-20154 <input type="checkbox"/> 20155-20156 <input type="checkbox"/> 20157-20158 <input type="checkbox"/> 20159-20160 <input type="checkbox"/> 20161-20162 <input type="checkbox"/> 20163-20164 <input type="checkbox"/> 20165-20166 <input type="checkbox"/> 20167-20168 <input type="checkbox"/> 20169-20170 <input type="checkbox"/> 20171-20172 <input type="checkbox"/> 20173-20174 <input type="checkbox"/> 20175-20176 <input type="checkbox"/> 20177-20178 <input type="checkbox"/> 20179-20180 <input type="checkbox"/> 20181-20182 <input type="checkbox"/> 20183-20184 <input type="checkbox"/> 20185-20186 <input type="checkbox"/> 20187-20188 <input type="checkbox"/> 20189-20190 <input type="checkbox"/> 20191-20192 <input type="checkbox"/> 20193-20194 <input type="checkbox"/> 20195-20196 <input type="checkbox"/> 20197-20198 <input type="checkbox"/> 20199-20200 <input type="checkbox"/> 20201-20202 <input type="checkbox"/> 20203-20204 <input type="checkbox"/> 20205-20206 <input type="checkbox"/> 20207-20208 <input type="checkbox"/> 20209-20210 <input type="checkbox"/> 20211-20212 <input type="checkbox"/> 20213-20214 <input type="checkbox"/> 20215-20216 <input type="checkbox"/> 20217-20218 <input type="checkbox"/> 20219-20220 <input type="checkbox"/> 20221-20222 <input type="checkbox"/> 20223-20224 <input type="checkbox"/> 20225-20226 <input type="checkbox"/> 20227-20228 <input type="checkbox"/> 20229-20230 <input type="checkbox"/> 20231-20232 <input type="checkbox"/> 20233-20234 <input type="checkbox"/> 20235-20236 <input type="checkbox"/> 20237-20238 <input type="checkbox"/> 20239-20240 <input type="checkbox"/> 20241-20242 <input type="checkbox"/> 20243-20244 <input type="checkbox"/> 20245-20246 <input type="checkbox"/> 20247-20248 <input type="checkbox"/> 20249-20250 <input type="checkbox"/> 20251-20252 <input type="checkbox"/> 20253-20254 <input type="checkbox"/> 20255-20256 <input type="checkbox"/> 20257-20258 <input type="checkbox"/> 20259-20260 <input type="checkbox"/> 20261-20262 <input type="checkbox"/> 20263-20264 <input type="checkbox"/> 20265-20266 <input type="checkbox"/> 20267-20268 <input type="checkbox"/> 20269-20270 <input type="checkbox"/> 20271-20272 <input type="checkbox"/> 20273-20274 <input type="checkbox"/> 20275-20276 <input type="checkbox"/> 20277-20278 <input type="checkbox"/> 20279-20280 <input type="checkbox"/> 20281-20282 <input type="checkbox"/> 20283-20284 <input type="checkbox"/> 20285-20286 <input type="checkbox"/> 20287-20288 <input type="checkbox"/> 20289-20290 <input type="checkbox"/> 20291-20292 <input type="checkbox"/> 20293-20294 <input type="checkbox"/> 20295-20296 <input type="checkbox"/> 20297-20298 <input type="checkbox"/> 20299-20300 <input type="checkbox"/> 20301-20302 <input type="checkbox"/> 20303-20304 <input type="checkbox"/> 20305-20306 <input type="checkbox"/> 20307-20308 <input type="checkbox"/> 20309-20310 <input type="checkbox"/> 20311-20312 <input type="checkbox"/> 20313-20314 <input type="checkbox"/> 20315-20316 <input type="checkbox"/> 20317-20318 <input type="checkbox"/> 20319-20320 <input type="checkbox"/> 20321-20322 <input type="checkbox"/> 20323-20324 <input type="checkbox"/> 20325-20326 <input type="checkbox"/> 20327-20328 <input type="checkbox"/> 20329-20330 <input type="checkbox"/> 20331-20332 <input type="checkbox"/> 20333-20334 <input type="checkbox"/> 20335-20336 <input type="checkbox"/> 20337-20338 <input type="checkbox"/> 20339-20340 <input type="checkbox"/> 20341-20342 <input type="checkbox"/> 20343-20344 <input type="checkbox"/> 20345-20346 <input type="checkbox"/> 20347-20348 <input type="checkbox"/> 20349-20350 <input type="checkbox"/> 20351-20352 <input type="checkbox"/> 20353-20354 <input type="checkbox"/> 20355-20356 <input type="checkbox"/> 20357-20358 <input type="checkbox"/> 20359-20360 <input type="checkbox"/> 20361-20362 <input type="checkbox"/> 20363-20364 <input type="checkbox"/> 20365-20366 <input type="checkbox"/> 20367-20368 <input type="checkbox"/> 20369-20370 <input type="checkbox"/> 20371-20372 <input type="checkbox"/> 20373-20374 <input type="checkbox"/> 20375-20376 <input type="checkbox"/> 20377-20378 <input type="checkbox"/> 20379-20380 <input type="checkbox"/> 20381-20382 <input type="checkbox"/> 20383-20384 <input type="checkbox"/> 20385-20386 <input type="checkbox"/> 20387-20388 <input type="checkbox"/> 20389-20390 <input type="checkbox"/> 20391-20392 <input type="checkbox"/> 20393-20394 <input type="checkbox"/> 20395-20396 <input type="checkbox"/> 20397-20398 <input type="checkbox"/> 20399-20400 <input type="checkbox"/> 20401-20402 <input type="checkbox"/> 20403-20404 <input type="checkbox"/> 20405-20406 <input type="checkbox"/> 20407-20408 <input type="checkbox"/> 20409-20410 <input type="checkbox"/> 20411-20412 <input type="checkbox"/> 20413-20414 <input type="checkbox"/> 20415-20416 <input type="checkbox"/> 20417-20418 <input type="checkbox"/> 20419-20420 <input type="checkbox"/> 20421-20422 <input type="checkbox"/> 20423-20424 <input type="checkbox"/> 20425-20426 <input type="checkbox"/> 20427-20428 <input type="checkbox"/> 20429-20430 <input type="checkbox"/> 20431-20432 <input type="checkbox"/> 20433-20434 <input type="checkbox"/> 20435-20436 <input type="checkbox"/> 20437-20438 <input type="checkbox"/> 20439-20440 <input type="checkbox"/> 20441-20442 <input type="checkbox"/> 20443-20444 <input type="checkbox"/> 20445-20446 <input type="checkbox"/> 20447-20448 <input type="checkbox"/> 20449-20450 <input type="checkbox"/> 20451-20452 <input type="checkbox"/> 20453-20454 <input type="checkbox"/> 20455-20456 <input type="checkbox"/> 20457-20458 <input type="checkbox"/> 20459-20460 <input type="checkbox"/> 20461-20462 <input type="checkbox"/> 20463-20464 <input type="checkbox"/> 20465-20466 <input type="checkbox"/> 20467-20468 <input type="checkbox"/> 20469-20470 <input type="checkbox"/> 20471-20472 <input type="checkbox"/> 20473-20474 <input type="checkbox"/> 20475-20476 <input type="checkbox"/> 20477-20478 <input type="checkbox"/> 20479-20480 <input type="checkbox"/> 20481-20482 <input type="checkbox"/> 20483-20484 <input type="checkbox"/> 20485-20486 <input type="checkbox"/> 20487-20488 <input type="checkbox"/> 20489-20490 <input type="checkbox"/> 20491-20492 <input type="checkbox"/> 20493-20494 <input type="checkbox"/> 20495-20496 <input type="checkbox"/> 20497-20498 <input type="checkbox"/> 20499-20500 <input type="checkbox"/> 20501-20502 <input type="checkbox"/> 20503-20504 <input type="checkbox"/> 20505-20506 <input type="checkbox"/> 20507-20508 <input type="checkbox"/> 20509-20510 <input type="checkbox"/> 20511-20512 <input type="checkbox"/> 20513-20514 <input type="checkbox"/> 20515-20516 <input type="checkbox"/> 20517-20518 <input type="checkbox"/> 20519-20520 <input type="checkbox"/> 20521-20522 <input type="checkbox"/> 20523-20524 <input type="checkbox"/> 20525-20526 <input type="checkbox"/> 20527-20528 <input type="checkbox"/> 20529-20530 <input type="checkbox"/> 20531-20532 <input type="checkbox"/> 20533-20534 <input type="checkbox"/> 20535-20536 <input type="checkbox"/> 20537-20538 <input type="checkbox"/> 20539-20540 <input type="checkbox"/> 20541-20542 <input type="checkbox"/> 20543-20544 <input type="checkbox"/> 20545-20546 <input type="checkbox"/> 20547-20548 <input type="checkbox"/> 20549-20550 <input type="checkbox"/> 20551-20552 <input type="checkbox"/> 20553-20554 <input type="checkbox"/> 20555-20556 <input type="checkbox"/> 20557-20558 <input type="checkbox"/> 20559-20560 <input type="checkbox"/> 20561-20562 <input type="checkbox"/> 20563-20564 <input type="checkbox"/> 20565-20566 <input type="checkbox"/> 20567-20568 <input type="checkbox"/> 20569-20570 <input type="checkbox"/> 20571-20572 <input type="checkbox"/> 20573-20574 <input type="checkbox"/> 20575-20576 <input type="checkbox"/> 20577-20578 <input type="checkbox"/> 20579-20580 <input type="checkbox"/> 20581-20582 <input type="checkbox"/> 20583-20584 <input type="checkbox"/> 20585-20586 <input type="checkbox"/> 20587-20588 <input type="checkbox"/> 20589-20590 <input type="checkbox"/> 20591-20592 <input type="checkbox"/> 20593-20594 <input type="checkbox"/> 20595-20596 <input type="checkbox"/> 20597-20598 <input type="checkbox"/> 20599-20600 <input type="checkbox"/> 20601-20602 <input type="checkbox"/> 20603-20604 <input type="checkbox"/> 20605-20606 <input type="checkbox"/> 20607-20608 <input type="checkbox"/> 20609-20610 <input type="checkbox"/> 20611-20612 <input type="checkbox"/> 20613-20614 <input type="checkbox"/> 20615-20616 <input type="checkbox"/> 20617-20618 <input type="checkbox"/> 20619-20620 <input type="checkbox"/> 20621-20622 <input type="checkbox"/> 20623-20624 <input type="checkbox"/> 20625-20626 <input type="checkbox"/> 20627-20628 <input type="checkbox"/> 20629-20630 <input type="checkbox"/> 20631-20632 <input type="checkbox"/> 20633-20634 <input type="checkbox"/> 20635-20636 <input type="checkbox"/> 20637-20638 <input type="checkbox"/> 20639-20640 <input type="checkbox"/> 20641-20642 <input type="checkbox"/> 20643-20644 <input type="checkbox"/> 20645-20646 <input type="checkbox"/> 20647-20648 <input type="checkbox"/> 20649-20650 <input type="checkbox"/> 20651-20652 <input type="checkbox"/> 20653-20654 <input type="checkbox"/> 20655-20656 <input type="checkbox"/> 20657-20658 <input type="checkbox"/> 20659-20660 <input type="checkbox"/> 20661-20662 <input type="checkbox"/> 20663-20664 <input type="checkbox"/> 20665-20666 <input type="checkbox"/> 20667-20668 <input type="checkbox"/> 20669-20670 <input type="checkbox"/> 20671-20672 <input type="checkbox"/> 20673-20674 <input type="checkbox"/> 20675-20676 <input type="checkbox"/> 20677-20678 <input type="checkbox"/> 20679-20680 <input type="checkbox"/> 20681-20682 <input type="checkbox"/> 20683-20684 <input type="checkbox"/> 20685-20686 <input type="checkbox"/> 20687-20688 <input type="checkbox"/> 20689-20690 <input type="checkbox"/> 20691-20692 <input type="checkbox"/> 20693-20694 <input type="checkbox"/> 20695-20696 <input type="checkbox"/> 20697-20698 <input type="checkbox"/> 20699-20700 <input type="checkbox"/> 20701-20702 <input type="checkbox"/> 20703-20704 <input type="checkbox"/> 20705-20706 <input type="checkbox"/> 20707-20708 <input type="checkbox"/> 20709-20710 <input type="checkbox"/> 20711-20712 <input type="checkbox"/> 20713-20714 <input type="checkbox"/> 20715-20716 <input type="checkbox"/> 20717-20718 <input type="checkbox"/> 20719-20720 <input type="checkbox"/> 20721-20722 <input type="checkbox"/> 20723-20724 <input type="checkbox"/> 20725-20726 <input type="checkbox"/> 20727-20728 <input type="checkbox"/> 20729-20730 <input type="checkbox"/> 20731-20732 <input type="checkbox"/> 20733-20734 <input type="checkbox"/> 20735-20736 <input type="checkbox"/> 20737-20738 <input type="checkbox"/> 20739-20740 <input type="checkbox"/> 20741-20742 <input type="checkbox"/> 20743-20744 <input type="checkbox"/> 20745-20746 <input type="checkbox"/> 20747-20748 <input type="checkbox"/> 20749-20750 <input type="checkbox"/> 20751-20752 <input type="checkbox"/> 20753-20754 <input type="checkbox"/> 20755-20756 <input type="checkbox"/> 20757-20758 <input type="checkbox"/> 20759-20760 <input type="checkbox"/> 20761-20762 <input type="checkbox"/> 20763-20764 <input type="checkbox"/> 20765-20766 <input type="checkbox"/> 20767-20768 <input type="checkbox"/> 20769-20770 <input type="checkbox"/> 20771-20772 <input type="checkbox"/> 20773-20774 <input type="checkbox"/> 20775-20776 <input type="checkbox"/> 20777-20778 <input type="checkbox"/> 20779-20780 <input type="checkbox"/> 20781-20782 <input type="checkbox"/> 20783-20784 <input type="checkbox"/> 20785-20786 <input type="checkbox"/> 20787-20788 <input type="checkbox"/> 20789-20790 <input type="checkbox"/> 20791-20792 <input type="checkbox"/> 20793-20794 <input type="checkbox"/> 20795-20796 <input type="checkbox"/> 20797-20798 <input type="checkbox"/> 20799-20800 <input type="checkbox"/> 20801-20802 <input type="checkbox"/> 20803-20804 <input type="checkbox"/> 20805-20806 <input type="checkbox"/> 20807-20808 <input type="checkbox"/> 20809-20810 <input type="checkbox"/> 20811-20812 <input type="checkbox"/> 20813-20814 <input type="checkbox"/> 20815-20816 <input type="checkbox"/> 20817-20818 <input type="checkbox"/> 20819-20820 <input type="checkbox"/> 20821-20822 <input type="checkbox"/> 20823-20824 <input type="checkbox"/> 20825-20826 <input type="checkbox"/> 20827-20828 <input type="checkbox"/> 20829-20830 <input type="checkbox"/> 20831-20832 <input type="checkbox"/> 20833-20834 <input type="checkbox"/> 20835-20836 <input type="checkbox"/> 20837-20838 <input type="checkbox"/> 20839-20840 <input type="checkbox"/> 20841-20842 <input type="checkbox"/> 20843-20844 <input type="checkbox"/> 20845-20846 <input type="checkbox"/> 20847-20848 <input type="checkbox"/> 20849-20850 <input type="checkbox"/> 20851-20852 <input type="checkbox"/> 20853-20854 <input type="checkbox"/> 20855-20856 <input type="checkbox"/> 20857-20858 <input type="checkbox"/> 20859-20860 <input type="checkbox"/> 20861-20862 <input type="checkbox"/> 20863-20864 <input type="checkbox"/> 20865-20866 <input type="checkbox"/> 20867-20868 <input type="checkbox"/> 20869-20870 <input type="checkbox"/> 20871-20872 <input type="checkbox"/> 20873-20874 <input type="checkbox"/> 20875-20876 <input type="checkbox"/> 20877-20878 <input type="checkbox"/> 20879-20880 <input type="checkbox"/> 20881-20882 <input type="checkbox"/> 20883-20884 <input type="checkbox"/> 20885-20886 <input type="checkbox"/> 20887-20888 <input type="checkbox"/> 20889-20890 <input type="checkbox"/> 20891-20892 <input type="checkbox"/> 20893-20894 <input type="checkbox"/> 20895-20896 <input type="checkbox"/> 20897-20898 <input type="checkbox"/> 20899-20900 <input type="checkbox"/> 20901-20902 <input type="checkbox"/> 20903-20904 <input type="checkbox"/> 20905-20906 <input type="checkbox"/> 20907-20908 <input type="checkbox"/> 20909-20910 <input type="checkbox"/> 20911-20912 <input type="checkbox"/> 20913-20914 <input type="checkbox"/> 20915-20916 <input type="checkbox"/> 20917-20918 <input type="checkbox"/> 20919-20920 <input type="checkbox"/> 20921-20922 <input type="checkbox"/> 20923-20924 <input type="checkbox"/> 20925-20926 <input type="checkbox"/> 20927-20928 <input type="checkbox"/> 20929-20930 <input type="checkbox"/> 20931-20932 <input type="checkbox"/> 20933-20934 <input type="checkbox"/> 20935-20936 <input type="checkbox"/> 20937-20938 <input type="checkbox"/> 20939-20940 <input type="checkbox"/> 20941-20942 <input type="checkbox"/> 20943-20944 <input type="checkbox"/> 20945-20946 <input type="checkbox"/> 20947-20948 <input type="checkbox"/> 20949-20950 <input type="checkbox"/> 20951-20952 <input type="checkbox"/> 20953-20954 <input type="checkbox"/> 20955-20956 <input type="checkbox"/> 20957-20958 <input type="checkbox"/> 20959-20960 <input type="checkbox"/> 20961-20962 <input type="checkbox"/> 20963-20964 <input type="checkbox"/> 20965-20966 <input type="checkbox"/> 20967-20968 <input type="checkbox"/> 20969-20970 <input type="checkbox"/> 20971-20972 <input type="checkbox"/> 20973-20974 <input type="checkbox"/> 20975-20976 <input type="checkbox"/> 20977-20978 <input type="checkbox"/> 20979-20980 <input type="checkbox"/> 20981-20982 <input type="checkbox"/> 20983-20984 <input type="checkbox"/> 20985-20986 <input type="checkbox"/> 20987-20988 <input type="checkbox"/> 20989-20989 <input type="checkbox"/> 20991-20992 <input type="checkbox"/> 20993-20994 <input type="checkbox"/> 20995-20996 <input type="checkbox"/> 20997-20998 <input type="checkbox"/> 20999-20999 <input type="checkbox"/> 21001-20999 <input type="checkbox"/> 21003-20999 <input type="checkbox"/> 21005-20999 <input type="checkbox"/> 21007-20999 <input type="checkbox"/> 21009-20999 <input type="checkbox"/> 21011-20999 <input type="checkbox"/> 21013-20999 <input type="checkbox"/> 21015-20999 <input type="checkbox"/> 21017-20999 <input type="checkbox"/> 21019-20999 <input type="checkbox"/> 21021-20999 <input type="checkbox"/> 21023-20999 <input type="checkbox"/> 21025-20999 <input type="checkbox"/> 21027-20999 <input type="checkbox"/> 21029-20999 <input type="checkbox"/> 21031-20999 <input type="checkbox"/> 21033-20999 <input type="checkbox"/> 21035-20999 <input type="checkbox"/> 21037-20999 <input type="checkbox"/> 21039-20999 <input type="checkbox"/> 21041-20999 <input type="checkbox"/> 21043-20999 <input type="checkbox"/> 21045-20999 <input type="checkbox"/> 21047-20999 <input type="checkbox"/> 21049-20999 <input type="checkbox"/> 21051-20999 <input type="checkbox"/> 21053-20999 <input type="checkbox"/> 21055-20999 <input type="checkbox"/> 21057-20999 <input type="checkbox"/> 21059-20999 <input type="checkbox"/> 21061-20999 <input type="checkbox"/> 21063-20999 <input type="checkbox"/> 21065-20999 <input type="checkbox"/> 21067-20999 <input type="checkbox"/> 21069-20999 <input type="checkbox"/> 21071-20999 <input type="checkbox"/> 21073-20999 <input type="checkbox"/> 21075-20999 <input type="checkbox"/> 21077-20999 <input type="checkbox"/> 21079-20999 <input type="checkbox"/> 21081-20999 <input type="checkbox"/> 21083-20999 <input type="checkbox"/> 21085-20999 <input type="checkbox"/> 21087-20999 <input type="checkbox"/> 21089-20999 <input type="checkbox"/> 21091-20999 <input type="checkbox"/> 21093-20999 <input type="checkbox"/> 21095-20999 <input type="checkbox"/> 21097-20999 <input type="checkbox"/> 21099-20999 <input type="checkbox"/> 21101-20999 <input type="checkbox"/> 21103-20999 <input type="checkbox"/> 21105-20999 <input type="checkbox"/> 21107-20999 <input type="checkbox"/> 21109-20999 <input type="checkbox"/> 21111-20999 <input type="checkbox"/> 21113-20999 <input type="checkbox"/> 21115-20999 <input type="checkbox"/> 21117-20999 <input type="checkbox"/> 21119-20999 <input type="checkbox"/> 21121-20999 <input type="checkbox"/> 21123-20999 <input type="checkbox"/> 21125-20999 <input type="checkbox"/> 21127-20999 <input type="checkbox"/> 21129-20999 <input type="checkbox"/> 21131-20999 <input type="checkbox"/> 21133-20999 <input type="checkbox"/> 21135-20999 <input type="checkbox"/> 21137-20999 <input type="checkbox"/> 21139-20999 <input type="checkbox"/> 21141-20999 <input type="checkbox"/> 21143-20999 <input type="checkbox"/> 21145-20999 <input type="checkbox"/> 21147-20999 <input type="checkbox"/> 21149-20999 <input type="checkbox"/> 21151-20999 <input type="checkbox"/> 21153-20999 <input type="checkbox"/> 21155-20999 <input type="checkbox"/> 21157-20999 <input type="checkbox"/> 2115											

DocType: mortgage.us.1004

- Appraiser #6

- Contract #1

### ContractDate

2023-06-17

### ContractPrice

498605

IsPropertySellerOwnerOfPublicRecord

99.50%

8

99.50%

- Improvements #1

- Neighborhood #1

- PudInfo #3

- Reconciliation #2

- ## SalesComparisonA

- Site #1

 Expand table

Model ID	Description	Automation use cases	Development options
<a href="#">prebuilt-mortgage.us.1004</a>	<ul style="list-style-type: none"><li>Extract key information from 1004 appraisals.</li><li><a href="#">Data and field extraction</a></li></ul>	<ul style="list-style-type: none"><li>Fannie Mae and Freddie Mac documentation requirements.</li><li>Uniform Residential Appraisal report to help lender/client with the market value of the subject property.</li></ul>	<ul style="list-style-type: none"><li><a href="#">Document Intelligence Studio</a></li><li><a href="#">REST API</a></li><li><a href="#">C# SDK</a></li><li><a href="#">Python SDK</a></li><li><a href="#">Java SDK</a></li><li><a href="#">JavaScript</a></li></ul>

[Return to model types](#)

## US mortgage 1005 form

 Fannie Mae

### Request for Verification of Employment

**Privacy Act Notice:** This information is to be used by the agency collecting it or its assignees in determining whether you qualify as a prospective mortgagor under its program and will not be released to anyone else without your consent. Any information you provide on this form is voluntary. Your failure to provide this information may result in your application being denied or delayed.

**Instructions:** Lender – Complete items 1 through 7. Have applicant complete items 8–Forward directly to employer named in item 1. The form is to be transmitted directly to the lender and is not to be transmitted through the applicant or any other party.

**Part I – Request**

1. To (Name and address of employer)	2. From (Name and address of lender)
ABC Software Company 123 Main Street, Seattle, WA 98111 Attn: Jane Doe	John Doe

I certify that this verification has been sent directly.  
3. Signature of Lender  


I have applied for a mortgage loan and stated that  
Name and address of Applicant (include employe)  
ABC Software Company  
123 Main Street, Seattle, WA 98111  
Attn: Jane Doe

**Part II – Verification of Present Employment**

8. Applicant's Date of Employment	10. 
12A. Current Gross Base Pay (Enter Amount and	13. Confidence
\$ 154,895.00	99.50%

**12B. Gross Earnings**

Type	Year To Date	Past Year 2019	Past Year 2018	Rations	\$	16. If paid hourly – average hours per week
Base Pay	\$ 150402	\$ 132607	\$ 100639	Fight or Hazard	\$	16. Date of applicant's next pay increase
Overtime	\$ 2093	\$ 1698	\$ 3007	Comming	\$	03/05/2021
Commission	\$ 764	\$ 1254	\$ 521	Quarters	\$	17. Projected amount of next pay increase
Bonus	\$ 1636	\$ 2233	\$ 1261	Pro Pay	\$	5,200
Total	\$ 154,895.00	\$ 137,892.00	\$ 105,428.00	Overseas or Commuter	\$	18. Date of applicant's last pay increase
				Standard Housing Allowance	\$	12/14/2019
					\$	19. Amount of last pay increase
						2,800

20. Remarks (If employee was off work for any length of time, please indicate period and reason)

**Part III – Verification of Previous Employment**

21. Date Hired	22. Date Terminated	23. Salary/age at Termination per (Year) (Month) (Week)
Base _____	Base _____	Overtime _____
Commissions _____		Bonus _____
24. Reason for Leaving	25. Position Held	

**Part IV – Authorized Signature** Federal statutes provide severe penalties for any fraud, intentional misrepresentation, or criminal conspiracy or conspiracy purposed to influence the issuance of any guarantee or insurance by the VA Secretary, the U.S.D.A., FHFA/FHA Commissioner, or the HUD/CIO Assistant Secretary.

26. Signature of Employer 	27. Title (Please print or type) Supervisor	28. Date 11/11/2020
29. Print or type name signed in Item 26 Jane Doe	30. Phone No. (900) 333-4444	Fannie Mae Form 1009 July 96

 Expand table

Model ID	Description	Automation use cases	Development options
<a href="#">prebuilt-mortgage.us.1005</a>	<ul style="list-style-type: none"> <li>Extract key information from <a href="#">1005</a> validation of employment.</li> <li><a href="#">Data and field extraction</a></li> </ul>	<ul style="list-style-type: none"> <li>Fannie Mae and Freddie Mac documentation requirements.</li> <li>Verification of employment document to determine the qualification as a prospective mortgagor.</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript</a></li> </ul>

[Return to model types](#)

## US mortgage 1008 form

**Uniform Underwriting and Transmittal Summary**

**I. Borrower and Property Information**

Borrower Name	Stan Hettinger	Occupancy Status	Primary Residence
Total # of Borrowers	2	Sales Price \$	1,000,000.00
Property Address	225 Bustleton Pike, Feasterville-Trevose, PA 19053	Appraised Value \$	800,000.00

Property Type:  Single Family  Condominium  Manufactured Housing  Multi-Family  Land  Commercial  Other

Project Classification:  Freddie Mac  Fannie Mae  Limited Revolving CDO Project  Limited Revolving Non-Condo Project  Limited Revolving Non-Revolving Project  Fannie Mae Review through PERS  FHA-Approved CDO Project  Construction Project Review Waived  Fannie Mae Review through PERS Co-Op Project  Fannie Mae Review through PERS Co-Construct  Fannie Mae Review through PERS Co-Permanent  Fannie Mae CDO Project Manager™ Project ID# (if any) \_\_\_\_\_

Occupancy Status:  Primary Residence  Second Home  Investment Property  Vacant Home  Household

Property Rights:  Freehold  Leasehold

Broker/Correspondent Name and Company Name: \_\_\_\_\_

**II. Mortgage Information**

Loan Type	Amortization Type	Loan Purpose	Lien Position
<input type="checkbox"/> Conventional <input type="checkbox"/> Second-Home <input type="checkbox"/> SIV <input type="checkbox"/> USDA/R	<input type="checkbox"/> Fixed-Rate–Monthly Payments <input type="checkbox"/> Fixed-Rate–Weekly Payments <input type="checkbox"/> Balloon <input type="checkbox"/> ARM (type specify) <input type="checkbox"/> Other (specify)	<input type="checkbox"/> Construction <input type="checkbox"/> Cash-Out Refinance <input type="checkbox"/> Limited-Cash-Out Refinance (Fannie) <input type="checkbox"/> Non-Cash-Out Refinance (Freddie) <input type="checkbox"/> Home Improvement <input type="checkbox"/> Construction Conversion/Constructor to Permanent	<input type="checkbox"/> First Mortgage <input type="checkbox"/> Second Mortgage <input type="checkbox"/> Subordinate Financing <input type="checkbox"/> Homeowner's Insurance <input type="checkbox"/> Supplemental Homeowner's Insurance
Note Information	Mortgage Originator	Temporary Buydown	Term _____
Loan Amount \$ 400,000.00 Note Rate 6.5000 % Loan Term (in months) 240	<input type="checkbox"/> Agent <input type="checkbox"/> Correspondent <input type="checkbox"/> Correspondent	<input type="checkbox"/> Agent <input type="checkbox"/> Correspondent <input type="checkbox"/> Correspondent	Term _____

Broker/Correspondent Name and Company Name: \_\_\_\_\_

**III. Underwriting Information**

Underwriter's Name	Appraiser's Name/License #	Appraisal Company Name
Adam Keeling	Kristina Deesek 710280	Padova Appraisal Service
Stable Monthly Income		Proposed Monthly Payment for the Property
Borrower 1 \$ 12,956.70		First Mortgage P&I \$ 2,982.29
Borrower 2 \$ 1,000.00		Subordinate Lien (s) P&I \$ 166.67
Borrower 3 \$ 1,000.00		Homeowner's Insurance \$ 0.00
Borrower 4 \$ 1,000.00		Supplemental Homeowner's Insurance \$ 0.00

**IV. Loan-to-Value Ratios**

17% LTV 80% CLTV

**DocType: mortgage.us.1008**

**Borrower** #1

Name	Stan Hettinger
NumberOfBorrowers	1

**Mortgage** #1

BrokerOrCorrespondentNameAndCompany Name	Mesh Nicki
LoanAmount	400000
LoanTermInMonths	240
NoteRatePercentage	6.5

 Expand table

Model ID	Description	Automation use cases	Development options
prebuilt-mortgage.us.1008	<ul style="list-style-type: none"> <li>Extract key information from Uniform Underwriting and Transmittal Summary.</li> <li>Data and field extraction</li> </ul>	<ul style="list-style-type: none"> <li>Loan underwriting processing using summary data.</li> </ul>	<ul style="list-style-type: none"> <li>Document Intelligence Studio</li> <li>REST API</li> <li>C# SDK</li> <li>Python SDK</li> <li>Java SDK</li> <li>JavaScript</li> </ul>

[Return to model types](#)

## US mortgage disclosure form

**Closing Disclosure**

This form is a statement of final loan terms and closing costs. Compare this document with your Loan Estimate.

Closing Information	Transaction Information	Loan Information
Document ID: 64-5689	Seller: Bonita Andersson	Loan Term: Fixed
Closing Date: 5/17/2022	Address: 1634 W Glenoaks Blvd, Glendale, CA 91201	Purchase Product: Fixed Rate
Disbursement Date: 5/21/2022	Seller: Bonita Andersson	Loan Type: Conventional
Settlement Agent: Bright Title Co.	Property Address: 1720 University Blvd, Rexburg, ID 83459	Mortgage Type: FHA
File # 64-5689	Property Address: 1634 W Glenoaks Blvd, Glendale, CA 91201	Loan ID #: 672047873
Property 1634 W Glenoaks Blvd, Glendale, CA 91201	Lender: Skye AU Bank	MIC #: 147188500
Sale Price: \$1,800,000.00		

<b>Loan Terms</b>	
Loan Amount: \$1,440,000	Can this amount increase after closing?: NO
Interest Rate: 7.033%	NO
Monthly Principal & Interest: \$9,609.39	NO
See Projected Payments below for your Estimated Total Monthly Payment	
Does the loan have these features?	
Prepayment Penalty: NO	
Balloon Payment: NO	

<b>Projected Payments</b>		
Payment Calculation:	Years 1-7	Years 8-30
Principal & Interest	\$9,609.39	\$9,609.39
Mortgage Insurance	+ \$100.00	+ —
Estimated Escrow	+ \$1,775.00	+ \$1,775.00
Estimated Total Monthly Payment	\$11,484.39	\$11,384.39
Estimated Taxes, Insurance & Assessments	\$2,525.00	This estimate includes In escrow?
Amount can increase over time See page 4 for details	a month	<input checked="" type="checkbox"/> Property Taxes YES <input checked="" type="checkbox"/> Homeowner's Insurance YES <input checked="" type="checkbox"/> Other Homeowner's Association Dues NO
See Escrow Account on page 4 for details. You must pay for other property costs separately.		

Run analysis    Query fields    Analyze options

Fields    Result    Code

DocType: mortgage.us.closingDisclosure

Closing #1

ClosingDate: 99.50%  
4/17/2022

DisbursementDate: 99.50%  
4/21/2022

FileNumber: 99.50%  
64-5681

IssueDate: 99.50%  
4/12/2022

PropertyAddress: 99.50%  
1634 W Glenoaks Blvd, Glendale, CA 91201

HouseNumber: 1634

Road: W Glenoaks Blvd

PostalCode: 91201

City: Glendale

[Expand table](#)

Model ID	Description	Automation use cases	Development options
prebuilt-mortgage.us.closingDisclosure	<ul style="list-style-type: none"> <li>Extract key information from Uniform Underwriting and Transmittal Summary.</li> <li>Data and field extraction</li> </ul>	<ul style="list-style-type: none"> <li>Mortgage loan final details requirements.</li> </ul>	<ul style="list-style-type: none"> <li>Document Intelligence Studio</li> <li>REST API</li> <li>C# SDK</li> <li>Python SDK</li> <li>Java SDK</li> <li>JavaScript</li> </ul>

[Return to model types](#)

## US Tax W-2 model

Analyze | All pages | Range

Copy 2 -- To Be Filed with Employee's State City, or Local Income Tax Return.		OMB NO. 1545-0008	
a. Employee's Soc Sec No 123-45-6785	1 Wages, tips, other comp. 37160.56	2 Federal income tax withheld 3894.54	
b. Employer ID number (EIN) 98-7654321	3 Social security wages 37160.56	4 Social security tax withheld 2303.95	
c. Employer's name, address and ZIP code CONTOSO LTD 123 MICROSOFT WAY REDMOND, WA 98768	5 Medicare wages and tips 37160.56	6 Medicare tax withheld 538.83	
d Control Number 000086242			
e Employee's name, address, and ZIP code ANGEL BROWN 4567 MAIN STREET BUFFALO, WA 12345			
7 Social security tips 802.80	8 Allocated tips 874.20	9	
10 Dependent care benefits 9873.20	11 Nonqualified plans 653.21	12a Code See Inst. for box 12 DD	12b Code 5432.00
13 Statutory employee X	14 Other DISINS 170.83	12c Code B	12d Code 876.30
Retirement plan X			
Third-party sick pay X			
P4 87654321 WA 12345678	37160.56	1135.65	
15 State Employer's state ID number 12345678	16 State wages, tips, etc. 9631.20	17 State income tax 1032.30	
18 Local wages, tips, etc. 37160.56	19 Local income tax 51.00	20 Locality name Cumberland Vly/Mdd	
		E.Pennsboro/E.Pnns	

[Expand table](#)

Model ID	Description	Automation use cases	Development options
prebuilt-tax.us.w2	<ul style="list-style-type: none"> <li>Extract key information from IRS US W2 tax forms (year 2018-2021).</li> <li>•</li> </ul>	<ul style="list-style-type: none"> <li>Automated tax document management.</li> <li>Mortgage loan application processing.</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript</a></li> </ul>

[Return to model types](#)

## US tax 1098 (and variations) forms

Analyze | All pages | Range

8181 <input type="checkbox"/> VOID <input type="checkbox"/> CORRECTED		Mortgage Interest Statement	
RECIPIENT'S/LENDER'S name, street address, city or town, state or province, country, ZIP or foreign postal code, and telephone no.		OMB No. 1545-1380	Form 1098 (Rev. January 2022) For calendar year 20
RECIPIENT'S/LENDER'S TIN	PAYER'S/BORROWER'S TIN	1 Mortgage interest received from payer(s)/borrower(s)	<b>Copy A</b> For Internal Revenue Service Center  File with Form 1096.  For Privacy Act and Paperwork Reduction Act Notice, see the current General Instructions for Certain Information Returns.
		2 Outstanding mortgage principal	
		3 Mortgage origination date	
		4 Refund of overpaid interest	
		5 Mortgage insurance premiums	
		6 Points paid on purchase of principal residence	
		7 <input type="checkbox"/> If address of property securing mortgage is the same as PAYER'S/BORROWER'S address, check the box, or enter the address or description in box 8.	
		8 Address or description of property securing mortgage (see instructions)	
9 Number of properties securing the mortgage	10 Other		
Account number (see instructions)		11 Mortgage acquisition date	
Form 1098 (Rev. 1-2022) Cat. No. 14402K www.irs.gov/Form1098 Department of the Treasury - Internal Revenue Service Do Not Cut or Separate Forms on This Page — Do Not Cut or Separate Forms on This Page			

[Expand table](#)

Model ID	Description	Development options
prebuilt-tax.us.1098{variation}	<ul style="list-style-type: none"> <li>Extract key information from 1098-form variations.</li> <li>•</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> </ul>

Model ID	Description	Development options
		<ul style="list-style-type: none"> <li>• Java SDK</li> <li>• JavaScript</li> </ul>

[Return to model types](#)

## US tax 1099 (and variations) forms

7171    VOID    CORRECTED

PAYER'S name, street address, city or town, state or province, country, ZIP or foreign postal code, and telephone no.	OMB No. 1545-0116 Form 1099-NEC (Rev. January 2022) For calendar year <span style="color: green;">20</span>
PAYER'S TIN	RECIPIENT'S TIN
RECIPIENT'S name	1 Nonemployee compensation  \$
Street address (including apt. no.)	2 Payer made direct sales totaling \$5,000 or more of consumer products to recipient for resale  \$
City or town, state or province, country, and ZIP or foreign postal code	3  4 Federal income tax withheld  \$
Account number (see instructions)	5 State tax withheld   6 State/Payer's state no.  \$      \$
	7 State income  \$

Form 1099-NEC (Rev. 1-2022) Cat. No. 72590N www.irs.gov/Form1099NEC Department of the Treasury - Internal Revenue Service  
Do Not Cut or Separate Forms on This Page — Do Not Cut or Separate Forms on This Page

Fields   Result   Code

Document 1 ▾

DocType: tax.us.1099NEC

● Payer #2   #1   ^  
 PhoneNumber   85.50%
   
● Box2 #2   99.90%  
 false
   
● TaxYear #2   99.90%
   
 20

🔍

[ ] Expand table

Model ID	Description	Development options
<a href="#">prebuilt-tax.us.1099{variation}</a>	<ul style="list-style-type: none"> <li>• Extract information from 1099-form variations.</li> <li>• </li> </ul>	<ul style="list-style-type: none"> <li>• Document Intelligence Studio <span style="color: blue;">↗</span></li> <li>• REST API</li> <li>• C# SDK</li> <li>• Python SDK</li> <li>• Java SDK</li> <li>• JavaScript</li> </ul>

[Return to model types](#)

## US tax 1040 (and variations) forms

1040   2023   OMB No. 1545-0074   IRS Use Only—Do not write or staple in this space

Your first name and middle initial	Last name	
Pascal	Weydert	
If joint return, spouse's first name and middle initial	Last name	
Weydert		
Home address, number and street. If you have a P.O. box, see instructions.	Apt. no.	
123 Fremont ave	88	
City, town, or post office. If you have a foreign address, also complete spaces below.	State	Zip code
Seattle	WA	98103
Foreign country name	Foreign province/state/country	Foreign postal code
Filing Status		
<input type="checkbox"/> Single <input type="checkbox"/> Head of household (HOH) <input type="checkbox"/> Married filing jointly (even if only one had income) <input type="checkbox"/> Qualifying surviving spouse (QSS) <input type="checkbox"/> Married filing separately (MFS)		
If you checked the MFS box, enter the name of your spouse. If you checked the HOH or QSS box, enter the child's name if the qualifying person is a child but not your dependent:  At any time during 2023, did you: (a) receive (as a reward, award, or payment for property or services); or (b) sell, exchange, or otherwise dispose of a digital asset (or a financial interest in a digital asset)? (See instructions.) <input type="checkbox"/> Yes <input type="checkbox"/> No		
Standard Deduction <input type="checkbox"/> Someone can claim: <input type="checkbox"/> You as a dependent <input type="checkbox"/> Your spouse as a dependent <input type="checkbox"/> Spouse itemizes on a separate return or you were a dual-status alien		
Age/Blindness: You: <input type="checkbox"/> Were born before January 2, 1959 <input type="checkbox"/> Are blind   Spouse: <input type="checkbox"/> Was born before January 2, 1959 <input type="checkbox"/> Is blind Dependents (see instructions): If more than four dependents, see instructions and check here: <input type="checkbox"/>		
(1) First name   Last name   (2) Social security number   (3) Relationship to you   (4) Check the box if qualifies for two (instructions): Credit for child tax credit   Credit for other dependents		
Income 1a Total amount from Form(s) W-2, box 1 (see instructions)   1b <span style="color: green;">83,650</span>		

Fields   Result   Code

DocType: tax.us.1040.2022

● TaxYear #1   99.80%
   
 2023
   
● Taxpayer #1   #2   ^  
 SSN   81.40%
   
 LastName   99.90%
   
 Weydert
   
 FirstNameAndInitials   99.90%
   
 Pascal

[ ] Expand table

Model ID	Description	Development options
<a href="#">prebuilt-tax.us.1040{variation}</a>	<ul style="list-style-type: none"> <li>• Extract information from 1040-form variations.</li> <li>• </li> </ul>	<ul style="list-style-type: none"> <li>• Document Intelligence Studio <span style="color: blue;">↗</span></li> <li>• REST API</li> <li>• C# SDK</li> </ul>

Model ID	Description	Development options
		<ul style="list-style-type: none"> <li>• Python SDK</li> <li>• Java SDK</li> <li>• JavaScript</li> </ul>

## Unified US tax forms

[Expand table](#)

Model ID	Description	Development options
prebuilt-tax.us	Extract information from any of the supported US tax forms.	<ul style="list-style-type: none"> <li>• Document Intelligence Studio</li> <li>• REST API</li> <li>• C# SDK</li> <li>• Python SDK</li> <li>• Java SDK</li> <li>• JavaScript</li> </ul>

## Custom model overview

Label data

[Train](#)

☰	Receipt No	1	⋮
☰	Sold To	2	⋮
☰	ID #	3	⋮
☰	Live Delivery?	4	⋮
☰	Online Delivery?	5	⋮
☰	Video Delivery?	6	⋮

[Expand table](#)

About	Description	Automation use cases	Development options
Custom model	Extracts information from forms and documents into structured data based on a model created from a set of representative training document sets.	Extract distinct data from forms and documents specific to your business and use cases.	<ul style="list-style-type: none"> <li>• Document Intelligence Studio</li> <li>• REST API</li> <li>• C# SDK</li> <li>• Java SDK</li> <li>• JavaScript SDK</li> <li>• Python SDK</li> </ul>

[Return to custom model types](#)

## Custom neural

## HOUSE RENTAL AGREEMENT

This House Rental Agreement ("Agreement," "rental agreement," or "lease") is entered into between **Opay LLC** (Landlord) and **SAM CAMPBELL AND SALLY BETTY BROWN** (Tenants). If more than one person is named as Tenant they shall be jointly and severally liable and responsible under the terms of this Agreement. This lease Agreement involves a residential house, yard, and related facilities located at **5 Bellows Dr., Salt Lake City, Utah 84138** (the "premises"). The date of this Agreement is **January 5, 2024**.

1. Landlord rents to Tenant, unfurnished, the premises on a month to month basis, terminable by either party at the end of any calendar month on at least 30 days notice to the other party. Tenant shall be entitled to possession of the premises and rent shall commence on **April 5, 2024**. Tenant shall not assign, sublease, or allow anyone other than persons permitted under this lease to at any time be in possession of any portion of the premises. Landlord will provide five (5) keys to Tenant; each key fits all outside door locks. Landlord will also provide two keys to the garage, and one remote garage door opener. All keys and the remote door opener will be returned to Landlord at the end of the tenancy.

2. Tenant agrees to pay the stipulated monthly rent in advance on the first day of each calendar month without offset or reduction. All rent is payable to Landlord at the address of Landlord: **5 Bellows Dr., Salt Lake City, Utah 84138** or at such other place as Landlord

### ⓘ Note

To train a custom neural model, set the `buildMode` property to `neural`. For more information, see [Training a neural model](#)

[Expand table](#)

About	Description	Automation use cases	Development options
<b>Custom Neural model</b>	The custom neural model is used to extract labeled data from structured (surveys, questionnaires), semi-structured (invoices, purchase orders), and unstructured documents (contracts, letters).	Extract text data, checkboxes, and tabular fields from structured and unstructured documents.	<a href="#">Document Intelligence Studio</a> <ul style="list-style-type: none"> <li>• <a href="#">REST API</a></li> <li>• <a href="#">C# SDK</a></li> <li>• <a href="#">Java SDK</a></li> <li>• <a href="#">JavaScript SDK</a></li> <li>• <a href="#">Python SDK</a></li> </ul>

[Return to custom model types](#)

## Custom template

**W-9**  
(Rev. October 2018)  
Department of the Treasury  
Internal Revenue Service

**Request for Taxpayer Identification Number and Certification**

► Go to [www.irs.gov/FormW9](http://www.irs.gov/FormW9) for instructions and the latest information.

**Give Form to the requester. Do not send to the IRS.**

**Part or type:** See Specific Instructions on page 3.

**1. Name** (as shown on your income tax return). Name is required on this line; do not leave this line blank.  
 **2. Business name/dissociated entity name, if different from above:**  
**Arctech Inc.**

**3. Check the appropriate box for federal tax classification of the person whose name is entered on line 1. Check only one of the following seven boxes:**

Individual/single-member LLC    C corporation    corporation    Partnership    Trust/estate

Limited liability company. Enter the tax classification (C/LC corporation, full/LC corporation, Pl/LC partnership)      
 Note: If you are a partnership, enter the classification of the single-member owner. Do not check C/LC if the LLC is classified as a single-member LLC that is disregarded from the owner unless the owner of the LLC is another LLC that is not disregarded from the owner for U.S. federal tax purposes. Otherwise, a single-member LLC that is disregarded from the owner should check the appropriate box for the tax classification of its owner.

Other (see instructions) ►

**4. Exemptions (boxes apply only to certain entities, not individuals; see instructions on page 3):**

Exempt payee code (if any)    
**5. Exemption from FATCA reporting code (if any)**

Applies to accounts maintained outside the U.S.

**6. Address number, street, and apt. or suite no.) (see instructions):**  
**100 Test Address**  
 City, state, and ZIP code  
 Telephone (WYA 93005)  
**7. List account numbers here (optional):** 123456789123, 123456789987

**Part I Taxpayer Identification Number (TIN)**  
 Enter your TIN in the appropriate box. The TIN provided must match the name given on line 1 to avoid back-and-forth correspondence. For example, if your name is John Doe, enter John Doe for a resident alien, sole proprietor, or disregarded entity; see the instructions for Part I, later. For other entities, it is your employer identification number (EIN). If you do not have a number, see How to get a TIN, later.  
Note: If the account is in more than one name, see the instructions for line 1. Also see What Name and Social security number

**Social security number**   -   -     
 or **Employer identification number**

### ⓘ Note

To train a custom template model, set the `buildMode` property to `template`. For more information, see [Training a template model](#)

[Expand table](#)

About	Description	Automation use cases	Development options
<b>Custom Template model</b>	The custom template model extracts labeled values and fields from structured and semi-structured documents.	Extract key data from highly structured documents with defined visual templates or common visual layouts, forms.	<ul style="list-style-type: none"> <li>• <a href="#">Document Intelligence Studio</a></li> <li>• <a href="#">REST API</a></li> <li>• <a href="#">C# SDK</a></li> </ul>

About	Description	Automation use cases	Development options
			<ul style="list-style-type: none"> <li>• Python SDK</li> <li>• Java SDK</li> <li>• JavaScript SDK</li> </ul>

[Return to custom model types](#)

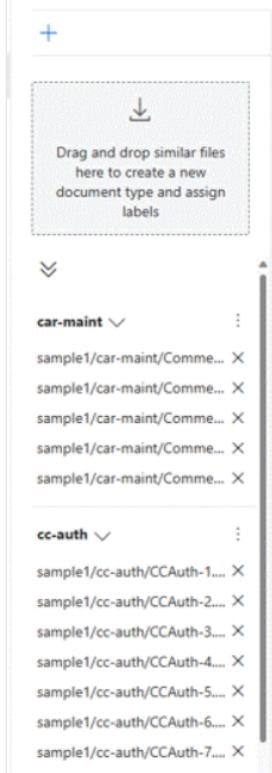
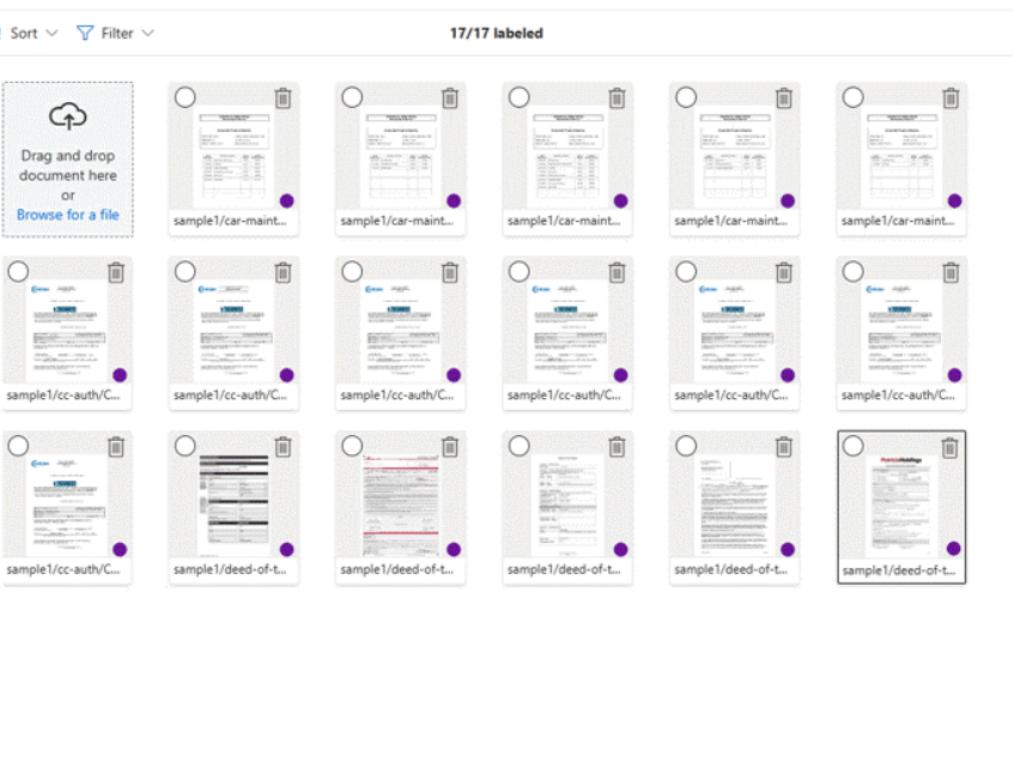
## Custom composed

[Expand table](#)

About	Description	Automation use cases	Development options
Composed custom models	A composed model is created by taking a collection of custom models and assigning them to a single model built from your form types.	Useful when you train several models and want to group them to analyze similar form types like purchase orders.	<ul style="list-style-type: none"> <li>• Document Intelligence Studio ↗</li> <li>• REST API</li> <li>• C# SDK</li> <li>• Java SDK</li> <li>• JavaScript SDK</li> <li>• Python SDK</li> </ul>

[Return to custom model types](#)

## Custom classification model

Label data	
 <p>Drag and drop similar files here to create a new document type and assign labels</p> <p>car-maint</p> <ul style="list-style-type: none"> <li>sample1/car-maint/Comme...</li> <li>sample1/car-maint/Comme...</li> <li>sample1/car-maint/Comme...</li> <li>sample1/car-maint/Comme...</li> <li>sample1/car-maint/Comme...</li> </ul> <p>cc-auth</p> <ul style="list-style-type: none"> <li>sample1/cc-auth/CCAuth-1...</li> <li>sample1/cc-auth/CCAuth-2...</li> <li>sample1/cc-auth/CCAuth-3...</li> <li>sample1/cc-auth/CCAuth-4...</li> <li>sample1/cc-auth/CCAuth-5...</li> <li>sample1/cc-auth/CCAuth-6...</li> <li>sample1/cc-auth/CCAuth-7...</li> </ul>	<p>Sort Filter</p> <p>17/17 labeled</p> 

[Expand table](#)

About	Description	Automation use cases	Development options
Composed classification model	Custom classification models combine layout and language features to detect, identify, and classify documents within an input file.	<ul style="list-style-type: none"> <li>• A loan application packaged containing application form, payslip, and, bank statement.</li> <li>• A collection of scanned invoices.</li> </ul>	<ul style="list-style-type: none"> <li>• Document Intelligence Studio ↗</li> <li>• REST API</li> </ul>

[Return to custom model types](#)

# Data privacy and security

As with all AI services, developers using the Document Intelligence service should be aware of Microsoft policies on customer data. See our [Data, privacy, and security for Document Intelligence](#) page.

## Next steps

- Choose a Document Intelligence model.
- Try processing your own forms and documents with the [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

---

## Feedback

Was this page helpful?



[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# What's new in Azure AI Document Intelligence

Article • 03/10/2025

This content applies to: v4.0 (GA) v3.1 (GA) v3.0 (GA) v2.1 (GA)

Document Intelligence service is updated on an ongoing basis. Bookmark this page to stay up to date with release notes, feature enhancements, and our newest documentation.

## Important

Preview API versions are retired once the GA API is released. The 2023-02-28-preview API version is retiring. If you're still using the preview API or the associated SDK versions, update your code to target the latest API version [2024-11-30 \(GA\)](#).

## December 2024

Document Intelligence v4.0 programming language SDKs are now generally available (GA)!

The latest client libraries default to the [2024-11-30 REST API \(GA\)](#) version of the service.

For more information, see client libraries for the following supported programming languages:

- [.NET \(C#\)](#)
- [Java](#)
- [JavaScript](#)
- [Python](#)

## November 2024

Document Intelligence REST API v4.0: [2024-11-30 REST API \(GA\)](#) is now generally available (GA)! The v4.0 REST API includes the following changes:

- [Batch API](#)

- Batch API now supports all models, including all read, layout, prebuilt verticals, and custom models.
  - Batch API supports LIST function to allow users to list batch jobs within past seven days.
  - Batch API supports DELETE function to explicitly delete batch job for GDPR and privacy compliance.
  - GetAnalyzeBatchResult supports resultId in response to LIST all resultIds.
-  Searchable PDF. The [prebuilt read](#) model now supports images formats (JPEG/JPG, PNG, BMP, TIFF, HEIF) and language expansion to include Chinese, Japanese, and Korean for [PDF output](#).
- [Custom classification model](#)
  - Custom classification model supports incremental training. You can add new samples to existing classes or add new classes by referencing an existing classifier.
  - With v4.0, custom classification model doesn't split documents by default during analysis. You need to explicitly set 'splitMode' property to auto to preserve the older behavior.
  - Custom classification model now supports 25,000 pages as new training page limit.
- [Custom Neural Model](#)
  - Custom Neural model now supports signature detection.
  - Custom neural models support paid training for longer duration when you need to train model with a larger labeled dataset. The first 20 training runs in a calendar month continue to be free. Any training operations over 20 is on the paid tier. Learn more details on [billing](#).
- [US Bank statement model](#)
  - US Bank Statement Model now supports check table extraction.
- [Check model](#)
  - Supports Payer's Signature extraction
- [Mortgage documents model](#)
  - Mortgage model now supports signature detection for forms 1003, 1004, 1005 and closing disclosure.
- [Receipt Model](#)
  - Receipt Model now supports more fields including ReceiptType, Tax rate, CountryRegion, net amount and description.
-  [US Tax model](#)

- New prebuilt tax models added for 1095A, 1095C, 1099SSA, and W4.
- [Delete analyze response](#)
  - Analyze response is stored for 24 hours from when the operation completes for retrieval. For scenarios where you want to delete the response sooner, use the delete analyze response API to delete the response.
- The v4.0 API includes cumulative updates from preview releases as listed:
  - [August 2024](#)
  - [May 2024](#)
  - [Feb 2024](#)

## August 2024

The Document Intelligence [2024-07-31-preview](#) REST API is now available. This preview API introduces new and updated capabilities:

- Public preview version [2024-07-31-preview](#) is currently available only in the following Azure regions. The new document field extraction model in [Azure AI Foundry portal](#) is only available in North Central US region:
  - **East US**
  - **West US2**
  - **West Europe**
  - **North Central US**
- **NEW Model compose with custom classifiers**
  - Document Intelligence now adds support for composing model with an explicit custom classification model. [Learn more about the benefits](#) of using the new compose capability.
- **Custom classification model**
  - Custom classification model now supports updating the model in-place as well.
  - Custom classification model adds support for model copy operation to enable backup and disaster recovery.
  - Custom classification model now supports explicitly specifying pages to be classified from an input document.
- **NEW Mortgage documents model**
  - Extract information from Appraisal (Form 1004).
  - Extract information from Validation of Employment (Form 1005).

-  [Check model](#)
  - Extract payee, amount, date, and other relevant information from checks.
-  [Pay Stub model](#)
  - New prebuilt to process pay stubs to extract wages, hours, deductions, net pay and more.
-  [Bank statement model](#)
  - New prebuilt to extract account information including beginning and ending balances, transaction details from bank statements.
-  [US Tax model](#)
  - New unified US tax model that can extract from forms such as W-2, 1098, 1099, and 1040.
-  Searchable PDF. The [prebuilt read model](#) now supports [PDF output](#) to download PDFs with embedded text from extraction results, allowing for PDF to be utilized in scenarios such as search copy of contents.
- [Layout model](#) now supports improved [figure detection](#) where figures from documents can now be downloaded as an image file to be used for further figure understanding. The layout model also features improvements to the OCR model for scanned text targeting improvements for single characters, boxed text, and dense text documents.
-  [Batch API](#)
  - Document Intelligence now adds support for batch analysis operation to support analyzing a set of documents to simplify developer experience and improve efficiency.
- [Add-on capabilities](#)
  - [Query fields](#) AI quality of extraction is improved with the latest model.

## May 2024

The Document Intelligence Studio adds support for Microsoft Entra (formerly Azure Active Directory) authentication. For more information, see [Authentication in Document Intelligence Studio](#).

## February 2024

The Document Intelligence [2024-07-31-preview](#) REST API is now available. This preview API introduces new and updated capabilities:

- Public preview version [2024-07-31-preview](#) is currently available only in the following Azure regions:
  - [East US](#)
  - [West US2](#)
  - [West Europe](#)
- Layout model now supports [figure detection](#) and [hierarchical document structure analysis \(sections and subsections\)](#). The AI quality of reading order and logical roles detection is also improved.
- [Custom extraction models](#)
  - Custom extraction models now support cell, row, and table level confidence scores. Learn more about [table, row, and cell confidence](#).
  - Custom extraction models have AI quality improvements for field extraction.
  - Custom template extraction model now supports extracting overlapping fields. Learn more about [overlapping fields and how you use them](#).
- [Custom classification model](#)
  - Custom classification model now supported incremental training for scenarios where you need to update the classifier model with added samples or classes. Learn more about [incremental training](#).
  - Custom classification model adds support for Office document types (.docx, .pptx, and .xls). Learn more about [expanded document type support](#).
- [Invoice model](#)
  - Support for new locales:

[] Expand table

Locale	Code
Arabic	( <a href="#">ar</a> )
Bulgarian	( <a href="#">bg</a> )
Greek	( <a href="#">el</a> )
Hebrew	( <a href="#">he</a> )
Macedonian	( <a href="#">mk</a> )
Russian ( <a href="#">ru</a> )	Serbian Cyrillic ( <a href="#">sr-cyr1</a> )

Locale	Code
Ukrainian	(uk)
Thai	(th)
Turkish	(tr)
Vietnamese	(vi)

- Support for new currency codes:

[\[+\] Expand table](#)

Currency	Locale	Code
BAM	Bosnian Convertible Mark	(ba)
BGN	Bulgarian Lev	(bg)
ILS	Israeli New Shekel	(il)
MKD	Macedonian Denar	(mk)
RUB	Russian Ruble	(ru)
THB	Thai Baht	(th)
TRY	Turkish Lira	(tr)
UAH	Ukrainian Hryvnia	(ua)
VND	Vietnamese Dong	(vn)

- Tax items support expansion for Germany (de), Spain (es), Portugal (pt), English Canada en-CA.

- ID model
  - Expanded field support for European Union IDs and driver license.
- **NEW Mortgage documents**
  - Extract information from Uniform Residential Loan Application (Form 1003).
  - Extract information from Uniform Underwriting and Transmittal Summary or Form 1008.
  - Extract information from mortgage closing disclosure.
- **NEW Credit/Debit card model**
  - Extract information from bank cards.
- **NEW Marriage certificate**

- New prebuilt to extract information from marriage certificates.

## December 2023

The [Document Intelligence client libraries](#) targeting REST API [2023-10-31-preview](#) are now available for use!

## November 2023

The Document Intelligence [2023-10-31-preview](#) REST API is now available. This preview API introduces new and updated capabilities:

- Public preview version [2023-10-31-preview](#) is currently only available in the following Azure regions:
  - **East US**
  - **West US2**
  - **West Europe**
- [Read model](#)
  - Language Expansion for Handwriting: Russian(`ru`), Arabic(`ar`), Thai(`th`).
  - Cyber Executive Order (EO) compliance.
- [Layout model](#)
  - Support office and HTML files.
  - Markdown output support.
  - Table extraction, reading order, and section heading detection improvements.
  - With the Document Intelligence 2023-10-31-preview, the general document model (`prebuilt-document`) is deprecated. Going forward, to extract key-value pairs from documents, use the `prebuilt-layout` model with the optional query string parameter `features=keyValuePairs` enabled.
- [Receipt model](#)
  - Now extracts currency for all price-related fields.
- [Health Insurance Card model](#)
  - New field support for Medicare and Medicaid information.
- [US Tax Document models](#)
  - New 1099 tax model. Supports base 1099 form and the following variations: A, B, C, CAP, DIV, G, H, INT, K, LS, LTC, MISC, NEC, OID, PATR, Q, QA, R, S, SA, SB.
- [Invoice model](#)

- Support for `KVK` field.
  - Support for `BPAY` field.
  - Numerous field refinements.
- **Custom Classification**
    - Support for multi-language documents.
    - New page splitting options: autosplit, always split by page, no split.
  - **Add-on capabilities**
    - [Query fields](#) are available with the `2023-10-31-preview` release.
    - Add-on capabilities are available within all models excluding the [Read model](#).

 **Note**

With the `2022-08-31` API general availability (GA) release, the associated preview APIs are being deprecated. If you're using the `2021-09-30-preview`, `2022-01-30-preview`, or `2022-06-30-preview` API versions, update your applications to target the `2022-08-31` API version. There are a few minor changes involved, for more information, see the [migration guide](#).

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Which model should I choose?

Article • 12/11/2024

Azure AI Document Intelligence supports a wide variety of models that enable you to add intelligent document processing to your applications and optimize your workflows. Selecting the right model is essential to ensure the success of your enterprise. In this article, we explore the available Document Intelligence models and provide guidance for how to choose the best solution for your projects.

<https://www.microsoft.com/en-us/videoplayer/embed/RE5fX1b?postJsIMsg=true>

The following decision charts highlight the features of each supported model to help you choose the model that best meets the needs and requirements of your application.

## ⓘ Important

Be sure to check the [language support](#) page for supported language text and field extraction by feature.

## Pretrained document-analysis models

expand Expand table

Document type	Example	Data to extract	Your best solution
A generic document.	A contract or letter.	You want to primarily extract written or printed text lines, words, locations, and detected languages.	<a href="#">Read OCR model</a>
A document that includes structural information.	A report or study.	In addition to written or printed text, you need to extract structural information like tables, selection marks, paragraphs, titles, headings, and subheadings.	<a href="#">Layout analysis model</a>
A structured or semi-structured document that includes content	A form or document that is a standardized format commonly used in	You want to extract fields and values including ones not covered by the	<a href="#">**Layout analysis model with the optional query string parameter</a>

Document type	Example	Data to extract	Your best solution
formatted as fields (keys) and values.	your business or industry like a credit application or survey.	scenario-specific prebuilt models without having to train a custom model.	<a href="#">features=keyValuePairs enabled **</a>

## Pretrained scenario-specific models

[+] [Expand table](#)

Document type	Data to extract	Your best solution
US Unified Tax	You want to extract key information across all tax forms of W2, 1040, 1090, 1098 from a single file without running any custom classification of your own.	<a href="#">US Unified tax model</a>
US Tax W-2 tax	You want to extract key information such as salary, wages, and taxes withheld.	<a href="#">US tax W-2 model</a>
US Tax W-4 tax	You want to extract key information such as claim adjustments, personal information.	<a href="#">US tax W-4 model</a>
US Tax 1095(A,C)	You want to extract premium tax credit, advance credit payment details.	<a href="#">US tax 1095 model</a>
US Tax 1098	You want to extract mortgage interest details such as principal, points, and tax.	<a href="#">US tax 1098 model</a>
US Tax 1098-E	You want to extract student loan interest details such as lender and interest amount.	<a href="#">US tax 1098-E model</a>
US Tax 1098T	You want to extract qualified tuition details such as scholarship adjustments, student status, and lender information.	<a href="#">US tax 1098-T model</a>
US Tax 1099(Variations)	You want to extract information from 1099 forms and its variations (A, B, C, CAP, DIV, G, H, INT, K, LS, LTC, MISC, NEC, OID, PATR, Q, QA, R, S, SA, SB).	<a href="#">US tax 1099 model</a>
US Tax 1040(Variations)	You want to extract information from 1040 forms and its variations (Schedule 1, Schedule 2, Schedule 3, Schedule 8812, Schedule A, Schedule B, Schedule C, Schedule D, Schedule E, Schedule EIC, Schedule F, Schedule H, Schedule J, Schedule R, Schedule SE, Schedule Senior).	<a href="#">US tax 1040 model</a>

Document type	Data to extract	Your best solution
Bank Statement	You want to extract key information from US bank statement	\Bank Statement
Bank check	You want to extract key information from check document.	Bank Check
Contract (legal agreement between parties).	You want to extract contract agreement details such as parties, dates, and intervals.	Contract model
Health insurance card or health insurance ID.	You want to extract key information such as insurer, member ID, prescription coverage, and group number.	Health insurance card model
Credit/Debit card	You want to extract key information bank cards such as card number and bank name.	Credit/Debit card model
Marriage Certificate	You want to extract key information from marriage certificates.	Marriage certificate model
Invoice or billing statement	You want to extract key information such as customer name, billing address, and amount due.	Invoice model
Receipt, voucher, or single-page hotel receipt.	You want to extract key information such as merchant name, transaction date, and transaction total.	Receipt model
Identity document (ID) like a U.S. driver's license or international passport	You want to extract key information such as first name, surname, date of birth, address, and signature.	Identity document (ID) model
Pay stub	You want to extract key information from the pay stub document.	Pay stub Model
US Mortgage 1003	You want to extract key information from the Uniform Residential loan application.	1003 form model
US Mortgage 1004	You want to extract key information from the Uniform Residential Appraisal Report (URAR).	1004 form model
US Mortgage 1005	You want to extract key information from the Verification of employment form	1005 form model
US Mortgage 1008	You want to extract key information from the Uniform Underwriting and Transmittal summary.	1008 form model
US Mortgage Closing Disclosure	You want to extract key information from a mortgage closing disclosure form.	Mortgage closing

Document type	Data to extract	Your best solution
		disclosure form model
Mixed-type document(s) with structured, semi-structured, and/or unstructured elements	You want to extract key-value pairs, selection marks, tables, signature fields, and selected regions not extracted by prebuilt or general document models.	Custom model

### Tip

- If you're still unsure which pretrained model to use, try the **layout model** with the optional query string parameter `features=keyValuePairs` enabled.
- The layout model is powered by the Read OCR engine to detect pages, tables, styles, text, lines, words, locations, and languages.

## Custom extraction models

[\[+\] Expand table](#)

Training set	Example documents	Your best solution
Structured, consistent, documents with a static layout.	Structured forms such as questionnaires or applications.	Custom template model
Structured and semi-structured.	<ul style="list-style-type: none"> <li>• Structured → surveys</li> <li>• Semi-structured → invoices</li> </ul>	Custom neural model
A collection of several models each trained on similar-type documents.	<ul style="list-style-type: none"> <li>• Supply purchase orders</li> <li>• Equipment purchase orders</li> <li>• Furniture purchase orders</li> </ul> <p>All composed into a single model.</p>	Composed custom model

## Custom classification model

[\[+\] Expand table](#)

Training set	Example documents	Your best solution
At least two different types of documents.	Forms, letters, or documents	<a href="#">Custom classification model</a>

## Next steps

- Learn how to process your own forms and documents with the [Document Intelligence Studio](#)
- 

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | Get help at Microsoft Q&A

# Create a Document Intelligence resource

Article • 11/19/2024

This content applies to: ✓ v4.0 (GA) ✓ v3.1 (GA) ✓ v3.0 (GA) ✓ v2.1 (GA)

Azure AI Document Intelligence is a cloud-based [Azure AI service](#) that uses machine-learning models to extract key-value pairs, text, and tables from your documents. In this article, learn how to create a Document Intelligence resource in the Azure portal.

## Visit the Azure portal

The Azure portal is a single platform you can use to create and manage Azure services.

Let's get started:

1. Sign in to the [Azure portal](#).
2. Select **Create a resource** from the Azure home page.
3. Search for and choose **Document Intelligence** from the search bar.
4. Select the **Create** button.

## Create a resource

1. Next, you're going to fill out the **Create Document Intelligence** fields with the following values:
  - **Subscription.** Select your current subscription.
  - **Resource group.** The [Azure resource group](#) that contains your resource. You can create a new group or add it to an existing group.
  - **Region.** Select your local region.
  - **Name.** Enter a name for your resource. We recommend using a descriptive name, for example *YourNameFormRecognizer*.
  - **Pricing tier.** The cost of your resource depends on the pricing tier you choose and your usage. For more information, see [pricing details](#). You can use the free pricing tier (F0) to try the service, and upgrade later to a paid tier for production.
2. Select **Review + Create**.

Home > Create a resource > Form Recognizer >

## Create Form Recognizer

[...](#)

[Basics](#) [Network](#) [Identity](#) [Tags](#) [Review + create](#)

Accelerate your business processes by automating information extraction. Form Recognizer applies advanced machine learning to accurately extract text, key/value pairs, and tables from documents. With just a few samples, Form Recognizer tailors its understanding to your documents, both on-premises and in the cloud. Turn forms into usable data at a fraction of the time and cost, so you can focus more time acting on the information rather than compiling it. [Learn more](#).

**Project Details**

Subscription \* ⓘ  [▼](#)

Resource group \* ⓘ  [▼](#)  
[Create new](#)

**Instance Details**

Region ⓘ  [▼](#)

Name \* ⓘ  [▼](#)

Pricing tier \* ⓘ  [▼](#)

[View full pricing details](#)

[Review + create](#) [< Previous](#) [Next : Network >](#)

3. Azure will run a quick validation check, after a few seconds you should see a green banner that says **Validation Passed**.
4. Once the validation banner appears, select the **Create** button from the bottom-left corner.
5. After you select create, you'll be redirected to a new page that says **Deployment in progress**. After a few seconds, you'll see a message that says, **Your deployment is complete**.

## Get Endpoint URL and keys

1. Once you receive the *deployment is complete* message, select the **Go to resource** button.
2. Copy the key and endpoint values from your Document Intelligence resource paste them in a convenient location, such as *Microsoft Notepad*. You need the key and endpoint values to connect your application to the Document Intelligence API.
3. If your overview page doesn't have the keys and endpoint visible, you can select the **Keys and Endpoint** button, on the left navigation bar, and retrieve them there.

The screenshot shows the 'Contoso-DI | Keys and Endpoint' page in the Azure portal. On the left, a sidebar lists various service management options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, and Keys and Endpoint (which is selected and highlighted with a red box). The main content area displays two key fields: 'KEY 1' and 'KEY 2', both represented by redacted text boxes with copy icons. Below these is a 'Location/Region' field set to 'westus2' and an 'Endpoint' field containing the URL 'https://contoso-di.cognitiveservices.azure.com/'. A callout box in the top right corner provides a note about securely storing keys.

That's it! You're now ready to start automating data extraction using Azure AI Document Intelligence.

## Next steps

- Try the [Document Intelligence Studio](#), an online tool for visually exploring, understanding, and integrating features from the Document Intelligence service into your applications.
- Complete a Document Intelligence quickstart and get started creating a document processing app in the development language of your choice:
  - [C#](#)
  - [Python](#)
  - [Java](#)
  - [JavaScript](#)

## Feedback

Was this page helpful?

Yes

No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Studio experience for Document Intelligence

Article • 03/19/2025

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)

The studio is an online tool to visually explore, understand, train, and integrate features from the Document Intelligence service into your applications. The studio provides a platform for you to experiment with the different Document Intelligence models and sample returned data in an interactive manner without the need to write code. You can use the studio experience to:

- Learn more about the different capabilities in Document Intelligence.
- Use your Document Intelligence resource to test models on sample documents or upload your own documents.
- Experiment with different add-on and preview features to adapt the output to your needs.
- Train custom classification models to classify documents.
- Train custom extraction models to extract fields from documents.
- Get sample code for the language specific [SDKs](#) to integrate into your applications.

Currently, we're undergoing the migration of features from the [Document Intelligence Studio](#) to the new [Azure AI Foundry portal](#). There are some differences in the offerings for the two studios, which determine the correct studio for your use case.

## Choosing the correct studio experience

There are currently two studios, the [Azure AI Foundry portal](#) and the [Document Intelligence Studio](#) for building and validating Document Intelligence models. As the experiences migrate to the new Azure AI Foundry portal, some experiences are available in both studios, while other experiences/models are only available in only one of the studios. To follow are a few guidelines for choosing the Studio experience for your needs. All of our [prebuilt models](#) and [general extraction models](#) are available on both studios.

## When to use [Document Intelligence Studio](#)

Document Intelligence Studio contains all features released on or before November 2024. For any of the v2.1, v3.0, v3.1 features, continue to use the Document Intelligence

Studio. Studios provide a visual experience for labeling, training, and validating custom models. For custom document field extraction models, use the Document Intelligence Studio for template and neural models. Custom classification models can only be trained and used on Document Intelligence Studio. Use Document Intelligence Studio if you want to try out GA versions of the models from version v3.0, v3.1, and v4.0.

## When to use Azure AI Foundry portal ↗

Start with the new Azure AI Foundry and try any of the prebuilt document models from **2024-11-30** version including general extraction models like Read or Layout.

## Learn more about Document Intelligence Studio

Select the studio experience from the following tabs to learn more about each studio and how you can get started.

Document Intelligence Studio

### ⓘ Important

- Document Intelligence Studio has distinct URLs for sovereign cloud regions.
- Azure for US Government: [Document Intelligence Studio \(Azure Fairfax\) ↗](#)
- Microsoft Azure operated by 21Vianet: [Document Intelligence Studio \(Azure China\) ↗](#)

The studio supports Document Intelligence v3.0 and later API versions for model analysis and custom model training. Previously trained v2.1 models with labeled data are supported, but not v2.1 model training. Refer to the [REST API migration guide](#) for detailed information about migrating from v2.1 to v3.0.

Use the [Document Intelligence Studio quickstart](#) to get started analyzing documents with document analysis or prebuilt models. Build custom models and reference the models in your applications using one of the [language specific SDKs](#).

## Document Intelligence model support

Use the help wizard, labeling interface, training step, and interactive visualizations to understand how each feature works.

- **Read:** Try out Document Intelligence's [Studio Read feature](#) with sample documents or your own documents and extract text lines, words, detected languages, and handwritten style if detected. To learn more, see [Read overview](#).
- **Layout:** Try out Document Intelligence's [Studio Layout feature](#) with sample documents or your own documents and extract text, tables, selection marks, and structure information. To learn more, see [Layout overview](#).
- **Prebuilt models:** Document Intelligence's prebuilt models enable you to add intelligent document processing to your apps and flows without having to train and build your own models. As an example, start with the [Studio Invoice feature](#). To learn more, see [Models overview](#).
- **Custom extraction models:** Document Intelligence's [Studio Custom models feature](#) enables you to extract fields and values from models trained with your data, tailored to your forms and documents. To extract data from multiple form types, create standalone custom models or combine two, or more, custom models and create a composed model. Test the custom model with your sample documents and iterate to improve the model. To learn more, see the [Custom models overview](#).
- **Custom classification models:** Document classification is a new scenario supported by Document Intelligence. The document classifier API supports classification and splitting scenarios. Train a classification model to identify the different types of documents your application supports. The input file for the classification model can contain multiple documents and classifies each document within an associated page range. To learn more, see [custom classification models](#).
- **Add-on Capabilities:** Document Intelligence supports more sophisticated analysis capabilities. These optional capabilities can be enabled and disabled in the studio using the `Analyze Options` button in each model page. There are four add-on capabilities available: `highResolution`, `formula`, `font`, and `barcode extraction` capabilities. To learn more, see [Add-on capabilities](#).

## Next steps

- Visit [Document Intelligence Studio](#).
  - Visit [Azure AI Foundry portal](#).
  - Get started with [Document Intelligence Studio quickstart](#).
- 

## Feedback

Was this page helpful?



Yes



No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Language support: document analysis

Article • 11/19/2024

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)   
[v2.1 \(GA\)](#)

Azure AI Document Intelligence models provide multilingual document processing support. Our language support capabilities enable your users to communicate with your applications in natural ways and empower global outreach. Document analysis models enable text extraction from forms and documents and return structured business-ready content ready for your organization's action, use, or progress. The following tables list the available language and locale support by model and feature:

- **Read:** The read model enables extraction and analysis of printed and handwritten text. This model is the underlying OCR engine for other Document Intelligence prebuilt models like layout, general document, invoice, receipt, identity (ID) document, health insurance card, tax documents and custom models. For more information, see [Read model overview](#)
- **Layout:** The layout model enables extraction and analysis of text, tables, document structure, and selection marks (like radio buttons and checkboxes) from forms and documents.

## Note

### Language code optional

- Document Intelligence's deep learning based universal models extract all multi-lingual text in your documents, including text lines with mixed languages, and don't require specifying a language code.
- Don't provide the language code as the parameter unless you are sure of the language and want to force the service to apply only the relevant model. Otherwise, the service may return incomplete and incorrect text.
- Also, it's not necessary to specify a locale. This is an optional parameter. The Document Intelligence deep-learning technology will auto-detect the text language in your image.

# Read model

## Model ID: prebuilt-read

Read: printed text

The following table lists read model language support for extracting and analyzing printed text.

 Expand table

Language	Code (optional)
Abaza	abq
Abkhazian	ab
Achinese	ace
Acoli	ach
Adangme	ada
Adyghe	ady
Afar	aa
Afrikaans	af
Akan	ak
Albanian	sq
Algonquin	alq
Angika (Devanagari)	anp
Arabic	ar
Asturian	ast
Asu (Tanzania)	asa
Avaric	av
Awadhi-Hindi (Devanagari)	awa

Language	Code (optional)
Aymara	ay
Azerbaijani (Latin)	az
Bafia	ksf
Bagheli	bfy
Bambara	bm
Bashkir	ba
Basque	eu
Belarusian (Cyrillic)	be, be-cyr1
Belarusian (Latin)	be, be-latn
Bemba (Zambia)	bem
Bena (Tanzania)	bez
Bhojpuri-Hindi (Devanagari)	bho
Bikol	bik
Bini	bin
Bislama	bi
Bodo (Devanagari)	brx
Bosnian (Latin)	bs
Brajbha	bra
Breton	br
Bulgarian	bg
Bundeli	bns
Buryat (Cyrillic)	bua
Catalan	ca
Cebuano	ceb
Chamling	rab
Chamorro	ch

Language	Code (optional)
Chechen	ce
Chhattisgarhi (Devanagari)	hne
Chiga	cgg
Chinese Simplified	zh-Hans
Chinese Traditional	zh-Hant
Choctaw	cho
Chukot	ckt
Chuvash	cv
Cornish	kw
Corsican	co
Cree	cr
Creek	mus
Crimean Tatar (Latin)	crh
Croatian	hr
Crow	cro
Czech	cs
Danish	da
Dargwa	dar
Dari	prs
Dhimal (Devanagari)	dhi
Dogri (Devanagari)	doi
Duala	dua
Dungan	dng
Dutch	nl
Efik	efi
English	en

Language	Code (optional)
Erzya (Cyrillic)	myv
Estonian	et
Faroese	fo
Fijian	fj
Filipino	fil
Finnish	fi

[\[+\] Expand table](#)

Language	Code (optional)
Fon	fon
French	fr
Friulian	fur
Ga	gaa
Gagauz (Latin)	gag
Galician	gl
Ganda	lg
Gayo	gay
German	de
Gilbertese	gil
Gondi (Devanagari)	gon
Greek	el
Greenlandic	kl
Guarani	gn
Gurung (Devanagari)	gvr
Gusii	guz
Haitian Creole	ht

Language	Code (optional)
Halbi (Devanagari)	hbl
Hani	hni
Haryanvi	bgc
Hawaiian	haw
Hebrew	he
Herero	hz
Hiligaynon	hil
Hindi	hi
Hmong Daw (Latin)	mww
Ho(Devanagiri)	hoc
Hungarian	hu
Iban	iba
Icelandic	is
Igbo	ig
Iloko	ilo
Inari Sami	smn
Indonesian	id
Ingush	inh
Interlingua	ia
Inuktitut (Latin)	iu
Irish	ga
Italian	it
Japanese	ja
Jaunsari (Devanagari)	jns
Javanese	jv
Jola-Fonyi	dyo

Language	Code (optional)
Kabardian	kbd
Kabuverdianu	kea
Kachin (Latin)	kac
Kalenjin	kln
Kalmyk	xal
Kangri (Devanagari)	xnr
Kanuri	kr
Karachay-Balkar	krc
Kara-Kalpak (Cyrillic)	kaa-cyrl
Kara-Kalpak (Latin)	kaa
Kashubian	csb
Kazakh (Cyrillic)	kk-cyrl
Kazakh (Latin)	kk-latn
Khakas	kjh
Khaling	klr
Khasi	kha
K'iche'	quc
Kikuyu	ki
Kildin Sami	sjd
Kinyarwanda	rw
Komi	kv
Kongo	kg
Korean	ko
Korku	kfq
Koryak	kpy
Kosraean	kos

Language	Code (optional)
Kpelle	kpe
Kuanyama	kj
Kumyk (Cyrillic)	kum
Kurdish (Arabic)	ku-arab
Kurdish (Latin)	ku-latn
Kurukh (Devanagari)	kru
Kyrgyz (Cyrillic)	ky
Lak	lbe
Lakota	lkt

[\[+\] Expand table](#)

Language	Code (optional)
Latin	la
Latvian	lv
Lezghian	lex
Lingala	ln
Lithuanian	lt
Lower Sorbian	dsb
Lozi	loz
Lule Sami	smj
Luo (Kenya and Tanzania)	luo
Luxembourgish	lb
Luyia	luy
Macedonian	mk
Machame	jmc
Madurese	mad

Language	Code (optional)
Mahasu Pahari (Devanagari)	bfz
Makhuwa-Meetto	mgh
Makonde	kde
Malagasy	mg
Malay (Latin)	ms
Maltese	mt
Malto (Devanagari)	kmj
Mandinka	mnk
Manx	gv
Maori	mi
Mapudungun	arn
Marathi	mr
Mari (Russia)	chm
Masai	mas
Mende (Sierra Leone)	men
Meru	mer
Meta'	mgo
Minangkabau	min
Mohawk	moh
Mongolian (Cyrillic)	mn
Mongondow	mog
Montenegrin (Cyrillic)	cnr-cyrl
Montenegrin (Latin)	cnr-latn
Morisyen	mfe
Mundang	mua
Nahuatl	nah

Language	Code (optional)
Navajo	nv
Ndonga	ng
Neapolitan	nap
Nepali	ne
Ngomba	jgo
Niuean	niu
Nogay	nog
North Ndebele	nd
Northern Sami (Latin)	sme
Norwegian	no
Nyanja	ny
Nyankole	yn
Nzima	nzi
Occitan	oc
Ojibwa	oj
Oromo	om
Ossetic	os
Pampanga	pam
Pangasinan	pag
Papiamento	pap
Pashto	ps
Pedi	nso
Persian	fa
Polish	pl
Portuguese	pt
Punjabi (Arabic)	pa

Language	Code (optional)
Quechua	qu
Ripuarian	ksh
Romanian	ro
Romansh	rm
Rundi	rn
Russian	ru
Rwa	rwk
Sadri (Devanagari)	sck
Sakha	sah
Samburu	saq
Samoan (Latin)	sm
Sango	sg

[Expand table](#)

Language	Code (optional)
Sangu (Gabon)	snq
Sanskrit (Devanagari)	sa
Santali(Devanagiri)	sat
Scots	sco
Scottish Gaelic	gd
Sena	seh
Serbian (Cyrillic)	sr-cyrl
Serbian (Latin)	sr, sr-latn
Shambala	ksb
Shona	sn
Siksika	bla

Language	Code (optional)
Sirmauri (Devanagari)	srx
Skolt Sami	sms
Slovak	sk
Slovenian	sl
Soga	xog
Somali (Arabic)	so
Somali (Latin)	so-latn
Songhai	son
South Ndebele	nr
Southern Altai	alt
Southern Sami	sma
Southern Sotho	st
Spanish	es
Sundanese	su
Swahili (Latin)	sw
Swati	ss
Swedish	sv
Tabassaran	tab
Tachelhit	shi
Tahitian	ty
Taita	dav
Tajik (Cyrillic)	tg
Tamil	ta
Tatar (Cyrillic)	tt-cyrl
Tatar (Latin)	tt
Teso	teo

Language	Code (optional)
Tetum	tet
Thai	th
Thangmi	thf
Tok Pisin	tpi
Tongan	to
Tsonga	ts
Tswana	tn
Turkish	tr
Turkmen (Latin)	tk
Tuvan	tyv
Udmurt	udm
Uighur (Cyrillic)	ug-cyrl
Ukrainian	uk
Upper Sorbian	hsb
Urdu	ur
Uyghur (Arabic)	ug
Uzbek (Arabic)	uz-arab
Uzbek (Cyrillic)	uz-cyrl
Uzbek (Latin)	uz
Vietnamese	vi
Volapük	vo
Vunjo	vun
Walser	wae
Welsh	cy
Western Frisian	fy
Wolof	wo

Language	Code (optional)
Xhosa	xh
Yucatec Maya	yua
Zapotec	zap
Zarma	dje
Zhuang	za
Zulu	zu

## Layout

Model ID: prebuilt-layout

Layout: printed text

The following table lists the supported languages for printed text:

Expand table

Language	Code (optional)
Abaza	abq
Abkhazian	ab
Achinese	ace
Acoli	ach
Adangme	ada
Adyge	ady
Afar	aa
Afrikaans	af
Akan	ak
Albanian	sq

Language	Code (optional)
Algonquin	alq
Angika (Devanagari)	anp
Arabic	ar
Asturian	ast
Asu (Tanzania)	asa
Avaric	av
Awadhi-Hindi (Devanagari)	awa
Aymara	ay
Azerbaijani (Latin)	az
Bafia	ksf
Bagheli	bfy
Bambara	bm
Bashkir	ba
Basque	eu
Belarusian (Cyrillic)	be, be-cyr1
Belarusian (Latin)	be, be-latn
Bemba (Zambia)	bem
Bena (Tanzania)	bez
Bhojpuri-Hindi (Devanagari)	bho
Bikol	bik
Bini	bin
Bislama	bi
Bodo (Devanagari)	brx
Bosnian (Latin)	bs
Brajbha	bra
Breton	br

Language	Code (optional)
Bulgarian	bg
Bundeli	bns
Buryat (Cyrillic)	bua
Catalan	ca
Cebuano	ceb
Chamling	rab
Chamorro	ch
Chechen	ce
Chhattisgarhi (Devanagari)	hne
Chiga	cgg
Chinese Simplified	zh-Hans
Chinese Traditional	zh-Hant
Choctaw	cho
Chukot	ckt
Chuvash	cv
Cornish	kw
Corsican	co
Cree	cr
Creek	mus
Crimean Tatar (Latin)	crh
Croatian	hr
Crow	cro
Czech	cs
Danish	da
Dargwa	dar
Dari	prs

Language	Code (optional)
Dhimal (Devanagari)	dhi
Dogri (Devanagari)	doi
Duala	dua
Dungan	dng
Dutch	nl
Efik	efi
English	en
Erzya (Cyrillic)	myv
Estonian	et
Faroese	fo
Fijian	fj
Filipino	fil
Finnish	fi

[Expand table](#)

Language	Code (optional)
Fon	fon
French	fr
Friulian	fur
Ga	gaa
Gagauz (Latin)	gag
Galician	gl
Ganda	lg
Gayo	gay
German	de
Gilbertese	gil

Language	Code (optional)
Gondi (Devanagari)	gon
Greek	el
Greenlandic	kl
Guarani	gn
Gurung (Devanagari)	gvr
Gusii	guz
Haitian Creole	ht
Halbi (Devanagari)	hlb
Hani	hni
Haryanvi	bgc
Hawaiian	haw
Hebrew	he
Herero	hz
Hiligaynon	hil
Hindi	hi
Hmong Daw (Latin)	mww
Ho(Devanagiri)	hoc
Hungarian	hu
Iban	iba
Icelandic	is
Igbo	ig
Iloko	ilo
Inari Sami	snn
Indonesian	id
Ingush	inh
Interlingua	ia

Language	Code (optional)
Inuktitut (Latin)	iu
Irish	ga
Italian	it
Japanese	ja
Jaunsari (Devanagari)	Jns
Javanese	jav
Jola-Fonyi	dyo
Kabardian	kbd
Kabuverdianu	kea
Kachin (Latin)	kac
Kalenjin	kln
Kalmyk	xal
Kangri (Devanagari)	xnr
Kanuri	kr
Karachay-Balkar	krc
Kara-Kalpak (Cyrillic)	caa-cyrl
Kara-Kalpak (Latin)	caa
Kashubian	csb
Kazakh (Cyrillic)	kk-cyrl
Kazakh (Latin)	kk-latn
Khakas	kjh
Khaling	klr
Khasi	kha
K'iche'	quc
Kikuyu	ki
Kildin Sami	sjd

Language	Code (optional)
Kinyarwanda	rw
Komi	kv
Kongo	kg
Korean	ko
Korku	kfq
Koryak	kpy
Kosraean	kos
Kpelle	kpe
Kuanyama	kj
Kumyk (Cyrillic)	kum
Kurdish (Arabic)	ku-arab
Kurdish (Latin)	ku-latn

[Expand table](#)

Language	Code (optional)
Kurukh (Devanagari)	kru
Kyrgyz (Cyrillic)	ky
Lak	lbe
Lakota	lkt
Latin	la
Latvian	lv
Lezghian	lex
Lingala	ln
Lithuanian	lt
Lower Sorbian	dsb
Lozi	loz

Language	Code (optional)
Lule Sami	smj
Luo (Kenya and Tanzania)	luo
Luxembourgish	lb
Luyia	luy
Macedonian	mk
Machame	jmc
Madurese	mad
Mahasu Pahari (Devanagari)	bfz
Makhuwa-Meetto	mgh
Makonde	kde
Malagasy	mg
Malay (Latin)	ms
Maltese	mt
Malto (Devanagari)	kmj
Mandinka	mnk
Manx	gv
Maori	mi
Mapudungun	arn
Marathi	mr
Mari (Russia)	chm
Masai	mas
Mende (Sierra Leone)	men
Meru	mer
Meta'	mgo
Minangkabau	min
Mohawk	moh

Language	Code (optional)
Mongolian (Cyrillic)	mn
Mongondow	mog
Montenegrin (Cyrillic)	cnr-cyr1
Montenegrin (Latin)	cnr-latn
Morisyen	mfe
Mundang	mua
Nahuatl	nah
Navajo	nv
Ndonga	ng
Neapolitan	nap
Nepali	ne
Ngomba	jgo
Niuean	niu
Nogay	nog
North Ndebele	nd
Northern Sami (Latin)	sme
Norwegian	no
Nyanja	ny
Nyankole	yn
Nzima	nzi
Occitan	oc
Ojibwa	oj
Oromo	om
Ossetic	os
Pampanga	pam
Pangasinan	pag

Language	Code (optional)
Papiamento	pap
Pashto	ps
Pedi	nso
Persian	fa
Polish	pl
Portuguese	pt
Punjabi (Arabic)	pa
Quechua	qu
Ripuarian	ksh
Romanian	ro
Romansh	rm
Rundi	rn
Russian	ru

[Expand table](#)

Language	Code (optional)
Rwa	rwk
Sadri (Devanagari)	sck
Sakha	sah
Samburu	saq
Samoan (Latin)	sm
Sango	sg
Sangu (Gabon)	snq
Sanskrit (Devanagari)	sa
Santali(Devanagiri)	sat
Scots	sco

Language	Code (optional)
Scottish Gaelic	gd
Sena	seh
Serbian (Cyrillic)	sr-cyrl
Serbian (Latin)	sr, sr-latn
Shambala	ksb
Shona	sn
Siksika	bla
Sirmauri (Devanagari)	srx
Skolt Sami	sms
Slovak	sk
Slovenian	sl
Soga	xog
Somali (Arabic)	so
Somali (Latin)	so-latn
Songhai	son
South Ndebele	nr
Southern Altai	alt
Southern Sami	sma
Southern Sotho	st
Spanish	es
Sundanese	su
Swahili (Latin)	sw
Swati	ss
Swedish	sv
Tabassaran	tab
Tachelhit	shi

Language	Code (optional)
Tahitian	ty
Taita	dav
Tajik (Cyrillic)	tg
Tamil	ta
Tatar (Cyrillic)	tt-cyrl
Tatar (Latin)	tt
Teso	teo
Tetum	tet
Thai	th
Thangmi	thf
Tok Pisin	tpi
Tongan	to
Tsonga	ts
Tswana	tn
Turkish	tr
Turkmen (Latin)	tk
Tuvan	tyv
Udmurt	udm
Uighur (Cyrillic)	ug-cyrl
Ukrainian	uk
Upper Sorbian	hsb
Urdu	ur
Uyghur (Arabic)	ug
Uzbek (Arabic)	uz-arab
Uzbek (Cyrillic)	uz-cyrl
Uzbek (Latin)	uz

Language	Code (optional)
Vietnamese	vi
Volapük	vo
Vunjo	vun
Walser	wae
Welsh	cy
Western Frisian	fy
Wolof	wo
Xhosa	xh
Yucatec Maya	yua
Zapotec	zap
Zarma	dje
Zhuang	za
Zulu	zu

## General document

### ⓘ Important

With Document Intelligence v4.0:2024-11-30 (GA), the general document model (prebuilt-document) is being added to layout (prebuilt-layout). To extract key-value pairs, selection marks, text, tables, and structure from documents, use the following models:

[\[+\] Expand table](#)

Key value pairs	version	Model ID
Layout model with query string <code>features=keyValuePairs</code> specified.	<ul style="list-style-type: none"> <li>• v4:2024-11-30 (GA)</li> <li>• v3.1:2023-07-31 (GA)</li> </ul>	<code>prebuilt-layout</code>

Key value pairs	version	Model ID
General document model	<ul style="list-style-type: none"><li>• v3.1:2023-07-31 (GA)</li><li>• v3.0:2022-08-31 (GA)</li></ul>	<b>prebuilt-document</b>

---

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | Get help at Microsoft Q&A

# Language support: prebuilt models

Article • 12/11/2024

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)   
[v2.1 \(GA\)](#)

Azure AI Document Intelligence models provide multilingual document processing support. Our language support capabilities enable your users to communicate with your applications in natural ways and empower global outreach. Prebuilt models enable you to add intelligent domain-specific document processing to your apps and flows without having to train and build your own models. The following tables list the available language and locale support by model and feature:

## Business card

### Important

Starting with Document Intelligence **v4.0 preview versions**, and going forward, the business card model (prebuilt-businessCard) is deprecated. To extract data from business cards, use earlier models.

 Expand table

Feature	version	Model ID
Business card model	<ul style="list-style-type: none"><li>• v3.1:2023-07-31 (GA)</li><li>• v3.0:2022-08-31 (GA)</li><li>• v2.1 (GA)</li></ul>	<a href="#">prebuilt-businessCard</a>

## Bank Statement

*Model ID: prebuilt-bankStatement*

 Expand table

Language Locale code	Default
English (United States) <a href="#">en-US</a>	English (United States) <a href="#">en-US</a>

# Contract

*Model ID: prebuilt-contract*

[\[+\] Expand table](#)

Language Locale code	Default
English (United States) en-US	English (United States) en-US

# Check

*Model ID: prebuilt-check*

[\[+\] Expand table](#)

Language Locale code	Default
English (United States) en-US	English (United States) en-US

# Health insurance card

*Model ID: prebuilt-healthInsuranceCard.us*

[\[+\] Expand table](#)

Language Locale code	Default
English (United States)	English (United States) en-US

# ID document

*Model ID: prebuilt-idDocument*

## Supported document types

[\[+\] Expand table](#)

Region	Document Types
Worldwide	Passport Book, Passport Card

<b>Region</b>	<b>Document Types</b>
United States	Driver License, Identification Card, Residency Permit (Green card), Social Security Card, Military ID
North America	Driver License, Identification Card, Residency Permit
South America	Driver License, Identification Card, Residency Permit
Europe	Driver License, Identification Card, Residency Permit
Southeast Asia	Driver License, Identification Card, Residency Permit
India	Driver License, PAN Card, Aadhaar Card
Australia	Driver License, Photo Card, Key-pass ID (including digital version)
New Zealand	Driver License, Identification Card, Residency Permit

*Model ID: prebuilt-idDocument*

## Supported document types

[\[+\] Expand table](#)

<b>Region</b>	<b>Document types</b>
Worldwide	Passport Book, Passport Card
United States	Driver License, Identification Card, Residency Permit (Green card), Social Security Card, Military ID
Europe	Driver License, Identification Card, Residency Permit
India	Driver License, PAN Card, Aadhaar Card
Canada	Driver License, Identification Card, Residency Permit (Maple Card)
Australia	Driver License, Photo Card, Key-pass ID (including digital version)

## Invoice

*Model ID: prebuilt-invoice*

## Supported languages

[Expand table](#)

Languages	Details
• Albanian (sq)	Albania (al)
• Arabic (ar)	Arabic (ar)
• Bulgarian (bg)	Bulgaria (bg)
• Chinese (simplified (zh-hans))	China (zh-hans-cn)
• Chinese (traditional (zh-hant))	Hong Kong SAR (zh-hant-hk), Taiwan (zh-hant-tw)
• Croatian (hr)	Bosnia and Herzegovina (ba), Croatia (hr), Serbia (rs)
• Czech (cs)	Czech Republic (cz)
• Danish (da)	Denmark (dk)
• Dutch (nl)	Netherlands (nl)
• English (en)	United States (us), Australia (au), Canada (ca), United Kingdom (-uk), India (-in)
• Estonian (et)	Estonia (ee)
• Finnish (fi)	Finland (f1)
• French (fr)	France (fr)
• German (de)	Germany (de)
• Greek (el)	Greece (el)
• Hebrew (he)	Hebrew (he)
• Hungarian (hu)	Hungary (hu)
• Icelandic (is)	Iceland (is)
• Italian (it)	Italy (it)
• Japanese (ja)	Japan (ja)
• Korean (ko)	Korea (kr)

Languages	Details
• Latvian (lv)	Latvia (lv)
• Lithuanian (lt)	Lithuania (lt)
• Macedonian (mk)	North Macedonia (mk)
• Malay (ms)	Malaysia (ms)
• Norwegian (nb)	Norway (no)
• Polish (pl)	Poland (pl)
• Portuguese (pt)	Portugal (pt), Brazil (br)
• Romanian (ro)	Romania (ro)
• Russian (ru)	Russia (ru)
• Serbian (Cyrillic) (sr-cyr1)	Serbia (sr)
• Serbian (sr-Latn)	Serbia (latn-rs)
• Slovak (sk)	Slovakia (sv)
• Slovenian (sl)	Slovenia (sl)
• Spanish (es)	Spain (es)
• Swedish (sv)	Sweden (se)
• Thai (th)	Thailand (th)
• Turkish (tr)	Turkey (tr)
• Ukrainian (uk)	Ukraine (uk)
• Vietnamese (vi)	Vietnam (vi)

# Mortgage

*Model ID: prebuilt-mortgage*

[\[\] Expand table](#)

Model ID	Language Locale code	Default
prebuilt-mortgage-1003	English (United States)	English (United States) en-US
prebuilt-mortgage-1004	English (United States)	English (United States) en-US
prebuilt-mortgage-1005	English (United States)	English (United States) en-US
prebuilt-mortgage-1008	English (United States)	English (United States) en-US
prebuilt-mortgage-.closingDisclosure	English (United States)	English (United States) en-US

## Pay stub

*Model ID: prebuilt-paystub*

[\[+\] Expand table](#)

Language Locale code	Default
English (United States) en-US	English (United States) en-US

## Receipt

*Model ID: prebuilt-receipt*

Thermal receipts

Language name	Language code	Language name	Language code
English	en	Lithuanian	lt
Afrikaans	af	Luxembourghish	lb
Akan	ak	Macedonian	mk
Albanian	sq	Malagasy	mg
Arabic	ar	Malay	ms
Azerbaijani	az	Maltese	mt
Bamanankan	bm	Maori	mi

Language name	Language code	Language name	Language code
Basque	eu	Marathi	mr
Belarusian	be	Maya, Yucatán	yua
Bhojpuri	bho	Mongolian	mn
Bosnian	bs	Nepali	ne
Bulgarian	bg	Norwegian	no
Catalan	ca	Nyanja	ny
Cebuano	ceb	Oromo	om
Corsican	co	Pashto	ps
Croatian	hr	Persian	fa
Czech	cs	Persian (Dari)	prs
Danish	da	Polish	pl
Dutch	nl	Portuguese	pt
Estonian	et	Punjabi	pa
Faroese	fo	Quechua	qu
Fijian	fj	Romanian	ro
Filipino	fil	Russian	ru
Finnish	fi	Samoan	sm
French	fr	Sanskrit	sa
Galician	gl	Scottish Gaelic	gd
Ganda	lg	Serbian (Cyrillic)	sr-cyr1
German	de	Serbian (Latin)	sr-latn
Greek	el	Sesotho	st
Guarani	gn	Sesotho sa Leboa	nso
Haitian Creole	ht	Shona	sn
Hawaiian	haw	Slovak	sk
Hebrew	he	Slovenian	sl

Language name	Language code	Language name	Language code
Hindi	hi	Somali (Latin)	so-latn
Hmong Daw	mww	Spanish	es
Hungarian	hu	Sundanese	su
Icelandic	is	Swedish	sv
Igbo	ig	Tahitian	ty
Iloko	ilo	Tajik	tg
Indonesian	id	Tamil	ta
Irish	ga	Tatar	tt
isiXhosa	xh	Tatar (Latin)	tt-latn
isiZulu	zu	Thai	th
Italian	it	Tongan	to
Japanese	ja	Turkish	tr
Javanese	jv	Turkmen	tk
Kazakh	kk	Ukrainian	uk
Kazakh (Latin)	kk-latn	Upper Sorbian	hsb
Kinyarwanda	rw	Uyghur	ug
Kiswahili	sw	Uyghur (Arabic)	ug-arab
Korean	ko	Uzbek	uz
Kurdish	ku	Uzbek (Latin)	uz-latn
Kurdish (Latin)	ku-latn	Vietnamese	vi
Kyrgyz	ky	Welsh	cy
Latin	la	Western Frisian	fy
Latvian	lv	Xitsonga	ts
Lingala	ln		

# Tax documents

[Expand table](#)

Model ID	Language Locale code	Default
prebuilt-tax.us.w2	English (United States)	English (United States) en-US
prebuilt-tax.us.w4	English (United States)	English (United States) en-US
prebuilt-tax.us	English (United States)	English (United States) en-US
prebuilt-tax.us.1099Combo	English (United States)	English (United States) en-US
prebuilt-tax.us.1098	English (United States)	English (United States) en-US
prebuilt-tax.us.1095	English (United States)	English (United States) en-US
prebuilt-tax.us.1098E	English (United States)	English (United States) en-US
prebuilt-tax.us.1098T	English (United States)	English (United States) en-US
prebuilt-tax.us.1099	English (United States)	English (United States) en-US

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Language support: custom models

Article • 11/19/2024

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)   
[v2.1 \(GA\)](#)

Azure AI Document Intelligence models provide multilingual document processing support. Our language support capabilities enable your users to communicate with your applications in natural ways and empower global outreach. Custom models are trained using your labeled datasets to extract distinct data from structured, semi-structured, and unstructured documents specific to your use cases. Standalone custom models can be combined to create composed models. The following tables list the available language and locale support by model and feature:

## Custom classifier

 Expand table

Language	Code (optional)
Afrikaans	af
Albanian	sq
Arabic	ar
Bulgarian	bg
Chinese (Han (Simplified variant))	zh-Hans
Chinese (Han (Traditional variant))	zh-Hant
Croatian	hr
Czech	cs
Danish	da
Dutch	nl
Estonian	et
Finnish	fi
French	fr

Language	Code (optional)
German	de
Hebrew	he
Hindi	hi
Hungarian	hu
Indonesian	id
Italian	it
Japanese	ja
Korean	ko
Latvian	lv
Lithuanian	lt
Macedonian	mk
Marathi	mr
Modern Greek (1453-)	el
Nepali (macrolanguage)	ne
Norwegian	no
Punjabi	pa
Persian	fa
Polish	pl
Portuguese	pt
Romanian	rm
Russian	ru
Slovak	sk
Slovenian	sl
Somali (Arabic)	so
Somali (Latin)	so-latn
Spanish	es

Language	Code (optional)
Swahili (macrolanguage)	sw
Swedish	sv
Tamil	ta
Thai	th
Turkish	tr
Ukrainian	uk
Urdu	ur
Vietnamese	vi

## Custom neural

### Printed text

The following table lists the supported languages for printed text.

[Expand table](#)

Language	Code (optional)
Afrikaans	af
Albanian	sq
Arabic	ar
Bulgarian	bg
Chinese Simplified	zh-Hans
Chinese Traditional	zh-Hant
Croatian	hr
Czech	cs
Danish	da
Dutch	nl
English	en

<b>Language</b>	<b>Code (optional)</b>
Estonian	et
Finnish	fi
French	fr
German	de
Hebrew	he
Hindi	hi
Hungarian	hu
Indonesian	id
Italian	it
Japanese	ja
Korean	ko
Latvian	lv
Lithuanian	lt
Macedonian	mk
Marathi	mr
Modern Greek (1453-)	el
Nepali (macrolanguage)	ne
Norwegian	no
Punjabi	pa
Persian	fa
Polish	pl
Portuguese	pt
Romanian	rm
Russian	ru
Slovak	sk
Slovenian	sl

Language	Code (optional)
Somali (Arabic)	so
Somali (Latin)	so-latn
Spanish	es
Swahili (macrolanguage)	sw
Swedish	sv
Tamil	ta
Thai	th
Turkish	tr
Ukrainian	uk
Urdu	ur
Vietnamese	vi

## Custom template

### Printed

The following table lists the supported languages for **printed** text.

[\[+\] Expand table](#)

Language	Code (optional)
Abaza	abq
Abkhazian	ab
Achinese	ace
Acoli	ach
Adangme	ada
Adyghe	ady
Afar	aa

<b>Language</b>	<b>Code (optional)</b>
Afrikaans	af
Akan	ak
Albanian	sq
Algonquin	alq
Angika (Devanagari)	anp
Arabic	ar
Asturian	ast
Asu (Tanzania)	asa
Avaric	av
Awadhi-Hindi (Devanagari)	awa
Aymara	ay
Azerbaijani (Latin)	az
Bafia	ksf
Bagheli	bfy
Bambara	bm
Bashkir	ba
Basque	eu
Belarusian (Cyrillic)	be, be-cyrl
Belarusian (Latin)	be, be-latn
Bemba (Zambia)	bem
Bena (Tanzania)	bez
Bhojpuri-Hindi (Devanagari)	bho
Bikol	bik
Bini	bin
Bislama	bi
Bodo (Devanagari)	brx

<b>Language</b>	<b>Code (optional)</b>
Bosnian (Latin)	bs
Brajbha	bra
Breton	br
Bulgarian	bg
Bundeli	bns
Buryat (Cyrillic)	bua
Catalan	ca
Cebuano	ceb
Chamling	rab
Chamorro	ch
Chechen	ce
Chhattisgarhi (Devanagari)	hne
Chiga	cgg
Chinese Simplified	zh-Hans
Chinese Traditional	zh-Hant
Choctaw	cho
Chukot	ckt
Chuvash	cv
Cornish	kw
Corsican	co
Cree	cr
Creek	mus
Crimean Tatar (Latin)	crh
Croatian	hr
Crow	cro
Czech	cs

<b>Language</b>	<b>Code (optional)</b>
Danish	da
Dargwa	dar
Dari	prs
Dhimal (Devanagari)	dhi
Dogri (Devanagari)	doi
Duala	dua
Dungan	dng
Dutch	nl
Efik	efi
English	en
Erzya (Cyrillic)	myv
Estonian	et
Faroese	fo
Fijian	fj
Filipino	fil
Finnish	fi

[\[+\] Expand table](#)

<b>Language</b>	<b>Code (optional)</b>
Fon	fon
French	fr
Friulian	fur
Ga	gaa
Gagauz (Latin)	gag
Galician	gl
Ganda	lg

<b>Language</b>	<b>Code (optional)</b>
Gayo	gay
German	de
Gilbertese	gil
Gondi (Devanagari)	gon
Greek	el
Greenlandic	kl
Guarani	gn
Gurung (Devanagari)	gvr
Gusii	guz
Haitian Creole	ht
Halbi (Devanagari)	hlb
Hani	hni
Haryanvi	bgc
Hawaiian	haw
Hebrew	he
Herero	hz
Hiligaynon	hil
Hindi	hi
Hmong Daw (Latin)	mww
Ho(Devanagiri)	hoc
Hungarian	hu
Iban	iba
Icelandic	is
Igbo	ig
Iloko	ilo
Inari Sami	smn

<b>Language</b>	<b>Code (optional)</b>
Indonesian	id
Ingush	inh
Interlingua	ia
Inuktitut (Latin)	iu
Irish	ga
Italian	it
Japanese	ja
Jaunsari (Devanagari)	Jns
Javanese	jv
Jola-Fonyi	dyo
Kabardian	kbd
Kabuverdianu	kea
Kachin (Latin)	kac
Kalenjin	kln
Kalmyk	xal
Kangri (Devanagari)	xnr
Kanuri	kr
Karachay-Balkar	krc
Kara-Kalpak (Cyrillic)	kaa-cyrl
Kara-Kalpak (Latin)	kaa
Kashubian	csb
Kazakh (Cyrillic)	kk-cyrl
Kazakh (Latin)	kk-latn
Khakas	kjh
Khaling	klr
Khasi	kha

Language	Code (optional)
K'iche'	quc
Kikuyu	ki
Kildin Sami	sjd
Kinyarwanda	rw
Komi	kv
Kongo	kg
Korean	ko
Korku	kfq
Koryak	kpy
Kosraean	kos
Kpelle	kpe
Kuanyama	kj
Kumyk (Cyrillic)	kum
Kurdish (Arabic)	ku-arab
Kurdish (Latin)	ku-latn
Kurukh (Devanagari)	kru
Kyrgyz (Cyrillic)	ky
Lak	lbe
Lakota	lkt

[\[+\]](#) Expand table

Language	Code (optional)
Latin	la
Latvian	lv
Lezghian	lex
Lingala	ln

<b>Language</b>	<b>Code (optional)</b>
Lithuanian	lt
Lower Sorbian	dsb
Lozi	loz
Lule Sami	smj
Luo (Kenya and Tanzania)	luo
Luxembourgish	lb
Luyia	luy
Macedonian	mk
Machame	jmc
Madurese	mad
Mahasu Pahari (Devanagari)	bfz
Makhuwa-Meetto	mgh
Makonde	kde
Malagasy	mg
Malay (Latin)	ms
Maltese	mt
Malto (Devanagari)	kmj
Mandinka	mnk
Manx	gv
Maori	mi
Mapudungun	arn
Marathi	mr
Mari (Russia)	chm
Masai	mas
Mende (Sierra Leone)	men
Meru	mer

<b>Language</b>	<b>Code (optional)</b>
Meta'	mgo
Minangkabau	min
Mohawk	moh
Mongolian (Cyrillic)	mn
Mongondow	mog
Montenegrin (Cyrillic)	cnr-cyrl
Montenegrin (Latin)	cnr-latn
Morisyen	mfe
Mundang	mua
Nahuatl	nah
Navajo	nv
Ndonga	ng
Neapolitan	nap
Nepali	ne
Ngomba	jgo
Niuean	niu
Nogay	nog
North Ndebele	nd
Northern Sami (Latin)	sme
Norwegian	no
Nyanja	ny
Nyankole	yn
Nzima	nzi
Occitan	oc
Ojibwa	oj
Oromo	om

<b>Language</b>	<b>Code (optional)</b>
Ossetic	os
Pampanga	pam
Pangasinan	pag
Papiamento	pap
Pashto	ps
Pedi	nso
Persian	fa
Polish	pl
Portuguese	pt
Punjabi (Arabic)	pa
Quechua	qu
Ripuarian	ksh
Romanian	ro
Romansh	rm
Rundi	rn
Russian	ru
Rwa	rwk
Sadri (Devanagari)	sck
Sakha	sah
Samburu	saq
Samoan (Latin)	sm
Sango	sg

[Expand table](#)

<b>Language</b>	<b>Code (optional)</b>
Sangu (Gabon)	snq

<b>Language</b>	<b>Code (optional)</b>
Sanskrit (Devanagari)	sa
Santali(Devanagiri)	sat
Scots	sco
Scottish Gaelic	gd
Sena	seh
Serbian (Cyrillic)	sr-cyrl
Serbian (Latin)	sr, sr-latn
Shambala	ksb
Shona	sn
Siksika	bla
Sirmauri (Devanagari)	srx
Skolt Sami	sms
Slovak	sk
Slovenian	sl
Soga	xog
Somali (Arabic)	so
Somali (Latin)	so-latn
Songhai	son
South Ndebele	nr
Southern Altai	alt
Southern Sami	sma
Southern Sotho	st
Spanish	es
Sundanese	su
Swahili (Latin)	sw
Swati	ss

<b>Language</b>	<b>Code (optional)</b>
Swedish	sv
Tabassaran	tab
Tachelhit	shi
Tahitian	ty
Taita	dav
Tajik (Cyrillic)	tg
Tamil	ta
Tatar (Cyrillic)	tt-cyr1
Tatar (Latin)	tt
Teso	teo
Tetum	tet
Thai	th
Thangmi	thf
Tok Pisin	tpi
Tongan	to
Tsonga	ts
Tswana	tn
Turkish	tr
Turkmen (Latin)	tk
Tuvan	tyv
Udmurt	udm
Uighur (Cyrillic)	ug-cyr1
Ukrainian	uk
Upper Sorbian	hsb
Urdu	ur
Uyghur (Arabic)	ug

Language	Code (optional)
Uzbek (Arabic)	uz-arab
Uzbek (Cyrillic)	uz-cyr
Uzbek (Latin)	uz
Vietnamese	vi
Volapük	vo
Vunjo	vun
Walser	wae
Welsh	cy
Western Frisian	fy
Wolof	wo
Xhosa	xh
Yucatec Maya	yua
Zapotec	zap
Zarma	dje
Zhuang	za
Zulu	zu

## Feedback

Was this page helpful?



Yes



No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Service quotas and limits

Article • 04/04/2025

This content applies to: v4.0 (GA) | Previous versions: v3.1 (GA) v3.0 (GA)

This article contains both a quick reference and detailed description of Azure AI Document Intelligence service Quotas and Limits for all [pricing tiers](#). It also contains some best practices to avoid request throttling.

## Model usage

Expand table

Document types supported	Read	Layout	Prebuilt models	Custom models	Add-on capabilities
PDF					
Images: <a href="#">JPEG/JPG</a> , <a href="#">PNG</a> , <a href="#">BMP</a> , <a href="#">TIFF</a> , <a href="#">HEIF</a>					
Microsoft Office: <a href="#">DOCX</a> , <a href="#">PPTX</a> , <a href="#">XLS</a>					

= supported = Not supported

For Document Intelligence v4.0 [2024-11-30](#) (GA) supports page and line features with the following restrictions:

- Angle, width/height, and unit aren't supported.
- For each object detected, bounding polygon or bounding regions aren't supported.
- The `lines` object isn't supported.

- [Document Intelligence SDKs](#)
- [Document Intelligence REST API](#)
- [Document Intelligence Studio v3.0](#)

Expand table

Quota	Free (F0) <sup>1</sup>	Standard (S0)
Analyze transactions Per Second limit	1	15 (default value)
Adjustable	No	Yes <sup>2</sup>

<b>Quota</b>	<b>Free (F0)<sup>1</sup></b>	<b>Standard (S0)</b>
<b>Get operations Per Second limit</b>	1	50 (default value)
Adjustable	No	Yes <sup>2</sup>
<b>Model management operations Per Second limit</b>	1	5 (default value)
Adjustable	No	Yes <sup>2</sup>
<b>List operations Per Second limit</b>	1	10 (default value)
Adjustable	No	Yes <sup>2</sup>
<b>Max document size</b>	4 MB	500 MB
Adjustable	No	No
<b>Max number of pages (Analysis)</b>	2	2000
Adjustable	No	No
<b>Max size of labels file</b>	10 MB	10 MB
Adjustable	No	No
<b>Max size of OCR json response</b>	500 MB	500 MB
Adjustable	No	No
<b>Max number of Template models</b>	500	5000
Adjustable	No	No
<b>Max number of Neural models</b>	100	500
Adjustable	No	No

## Custom model usage

- ✓ [Custom template model](#)
- ✓ [Custom neural model](#)
- ✓ [Composed classification models](#)
- ✓ [Composed custom models](#)

[\[ \]](#) [Expand table](#)

<b>Quota</b>	<b>Free (F0)<sup>1</sup></b>	<b>Standard (S0)</b>
<b>Compose Model limit</b>	5	500 (default value)
Adjustable	No	No
<b>Training dataset size * Neural and Generative</b>	1 GB <sup>3</sup>	1 GB (default value)
Adjustable	No	No
<b>Training dataset size * Template</b>	50 MB <sup>4</sup>	50 MB (default value)
Adjustable	No	No
<b>Max number of pages (Training) * Template</b>	500	500 (default value)
Adjustable	No	No
<b>Max number of pages (Training) * Neural and Generative</b>	50,000	50,000 (default value)
Adjustable	No	No
<b>Custom neural model train</b>	10 hours per month <sup>5</sup>	no limit (pay by the hour), start with 10 free hours each month
Adjustable	No	Yes <sup>3</sup>
<b>Max number of pages (Training) * Classifier</b>	10,000	10,000 (default value)
Adjustable	No	No
<b>Max number of document types (classes) * Classifier</b>	500	500 (default value)
Adjustable	No	No
<b>Training dataset size * Classifier</b>	1GB	2GB (default value)
Adjustable	No	No
<b>Min number of samples per class * Classifier</b>	5	5 (default value)
Adjustable	No	No

<sup>1</sup> For Free (F0) pricing tier see also monthly allowances at the [pricing page](#)<sup>2</sup>.

<sup>2</sup> See [best practices](#), and [adjustment instructions](#).

<sup>3</sup> Neural models training count is reset every calendar month. Open a support request to

increase the monthly training limit. Starting with the v4.0 API, training requests over 20 requests in a calendar month are billed on the training tier. See [pricing](#) for details.

<sup>4</sup> This limit applies to all documents found in your training dataset folder prior to any labeling-related updates.

<sup>5</sup> This limit applies for `v 4.0 (2024-11-30 GA)` custom neural models only. Starting from `v 4.0`, we support training larger documents for longer durations (up to 10 hours for free, and incurring charges after). For more information, please refer to [custom neural model page](#).

## Detailed description, Quota adjustment, and best practices

The default limits can be extended by requesting an increase via a support ticket. Before requesting a quota increase (where applicable), ensure that it's necessary. Document Intelligence service uses autoscaling to bring the required computational resources `on-demand`, keep the customer costs low, and deprovision unused resources by not maintaining an excessive amount of hardware capacity.

If your application returns Response Code 429 (*Too many requests*) you are over the threshold for one or more of the transactions per second limits (TPS):

- **Analyze transactions Per Second limit** The TPS for submitting analyze requests (POST)
- **Get operations Per Second limit** The TPS for polling for results on analyze operations (GET)
- **Model management operations Per Second limit** Operations related to model management like build/train and copy.
- **List operations Per Second limit** Operations related to listing models, operations.

## General best practices to mitigate throttling during autoscaling

To minimize issues related to throttling (Response Code 429), we recommend using the following techniques:

- Implement retry logic in your application
- Avoid sharp changes in the workload. Increase the workload gradually  
*Example.* Your application is using Document Intelligence and your current workload is 10 TPS (transactions per second). The next second you increase the load to 40 TPS. The result

is a 429 response code for some requests as you are over the 15 TPS limit for submitting analyze operations. You could either back off the processing to stay under the 15 TPS or request an increase on the TPS to support your higher volumes.

The next sections describe specific cases of adjusting quotas. Jump to [Document Intelligence: increasing concurrent request limit](#)

## Increasing transactions per second request limit

By default the number of transactions per second is limited to 15 transactions per second for a Document Intelligence resource. For the Standard pricing tier, this amount can be increased. Before submitting the request, ensure you're familiar with the material in [this section](#) and aware of these [best practices](#).

The first step would be to enable auto scaling. Follow this document to enable auto scaling on your resource \* [enable auto scaling](#). With auto scaling enabled your resource can continue to accept requests over the TPS limits configured if there's capacity on the service. It can still result in request throttled.

Increasing the Concurrent Request limit does **not** directly affect your costs. Document Intelligence service uses "Pay only for what you use" model. The limit defines how high the Service can scale before it starts throttle your requests.

The existing value of different request limit categories is available via Azure portal, under the monitoring tab on the resource overview blade.

## Create and submit support request for TPS increase

Initiate the increase of transactions per second(TPS) limit for your resource by submitting the Support Request:

- Sign in to the [Azure portal](#)
- Select the Document Intelligence Resource for which you would like to increase the TPS limit
- Select -New support request- (-Support + troubleshooting- group). A new window appears with autopopulated information about your Azure Subscription and Azure Resource
- Enter -Summary- (like "Increase Document Intelligence TPS limit")
- Select "Quota or usage validation" for problem type field.
- Select -Next: Solutions-
- Proceed further with the request creation
- Enter the following information in the -Description- field, under the Details tab:

- a note, that the request is about Document Intelligence quota.
- Provide a TPS expectation you would like to scale to meet. While TPS increases are free, you should only request a TPS that is reasonable for your workload.
- Azure resource information
- Complete entering the required information and select -Create- button in -Review + create- tab
- Note the support request number in Azure portal notifications. Look for Support to contact you shortly for further processing.

## Example of a workload pattern best practice

This example presents the approach we recommend following to mitigate possible request throttling due to [Autoscaling being in progress](#). It isn't an *exact recipe*, but merely a template we invite to follow and adjust as necessary.

Let us suppose that a Document Intelligence resource has the default limit set. Start the workload to submit your analyze requests. If you find that you're seeing frequent throttling with response code 429 when checking for completion, start by implementing an exponential backoff on the GET analyze response request. By using a progressively longer wait time between retries for consecutive error responses, for example a 2-5-13-34 pattern of delays between requests. In general, we recommended not calling the get analyze response more than once every 2 seconds for a corresponding POST request. The `analyze` response also contains a **retry-after** header that indicates how long you should wait in seconds before checking for completion of that request.

If you find that you're being throttled on the number of POST requests for documents being submitted, consider adding a delay between the requests. If your workload requires a higher degree of concurrent processing, you then need to create a support request to increase your service limits on transactions per second.

Generally, we recommended testing the workload and the workload patterns before going to production.

## Next steps

[Learn about error codes and troubleshooting](#)

# SDK target: REST API v4.0 (GA)

Article • 02/11/2025

## ✓ REST API version 2024-11-30 GA

Azure AI Document Intelligence is a cloud service that uses machine learning to analyze text and structured data from documents. The Document Intelligence software development kit (SDK) is a set of libraries and tools that enable you to easily integrate Document Intelligence models and capabilities into your applications. Document Intelligence SDK is available across platforms in C#/.NET, Java, JavaScript, and Python programming languages.

## Supported programming languages

Document Intelligence SDK supports the following languages and platforms:

  Expand table

Language → Document Intelligence SDK version	Package	Supported API version	Platform support
<a href="#">.NET/C# → 1.0.0 (GA)</a>	<a href="#">NuGet ↗</a>	<a href="#">2024-11-30 (GA)</a>	Windows, macOS, Linux, <a href="#">Docker ↗</a>
<a href="#">Java → 1.0.0 (GA)</a>	<a href="#">Maven repository ↗</a>	<a href="#">2024-11-30 (GA)</a>	Windows, macOS, Linux
<a href="#">JavaScript → 1.0.0 (GA)</a>	<a href="#">npm ↗</a>	<a href="#">2024-11-30 (GA)</a>	Browser, Windows, macOS, Linux ↗
<a href="#">Python → 1.0.0 (GA)</a>	<a href="#">PyPI ↗</a>	<a href="#">2024-11-30 (GA)</a>	Windows, macOS, Linux

For more information on other SDK versions, see:

- [2023-07-31 v3.1 \(GA\)](#)
- [2022-08-31 v3.0 \(GA\)](#)
- [v2.1 \(GA\)](#)

## Supported Clients

The following tables present the correlation between each SDK version the supported API versions of the Document Intelligence service.

[+] Expand table

Language	SDK alias	API version (default)	Supported clients
.NET/C# 1.0.0 (GA)	v4.0 (GA)	2024-11-30 GA	DocumentIntelligenceClient DocumentIntelligenceAdministrationClient
.NET/C# 4.1.0	v3.1 latest (GA)	2023-07-31	DocumentAnalysisClient DocumentModelAdministrationClient
.NET/C# 4.0.0	v3.0 (GA)	2022-08-31	DocumentAnalysisClient DocumentModelAdministrationClient
.NET/C# 3.1.x	v2.1	v2.1	FormRecognizerClient FormTrainingClient
.NET/C# 3.0.x	v2.0	v2.0	FormRecognizerClient FormTrainingClient

## Use Document Intelligence SDK in your applications

The Document Intelligence SDK enables the use and management of the Document Intelligence service in your application. The SDK builds on the underlying Document Intelligence REST API allowing you to easily use those APIs within your programming language paradigm. Here's how you use the Document Intelligence SDK for your preferred language:

### 1. Install the SDK client library

.NET CLI

```
dotnet add package Azure.AI.DocumentIntelligence -Version 1.0.0
```

PowerShell

```
Install-Package Azure.AI.DocumentIntelligence -Version 1.0.0
```

## 2. Import the SDK client library into your application

C#/.NET

C#

```
using Azure;
using Azure.AI.DocumentIntelligence;
```

## 3. Set up authentication

There are two supported methods for authentication:

- Use a [Document Intelligence API key](#) with AzureKeyCredential from azure.core.credentials.
- Use a [token credential from azure-identity](#) to authenticate with [Microsoft Entra ID](#).

### Use your API key

Here's where to find your Document Intelligence API key in the Azure portal:

The screenshot shows the Azure portal interface for a resource named 'Contoso-DI'. The left sidebar has a 'Resource Management' section expanded, with 'Keys and Endpoint' selected and highlighted by a red box. Other items in the sidebar include Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Encryption, Pricing tier, Networking, Identity, Cost analysis, Properties, Locks, Monitoring, Automation, and Help. At the top right, there are 'Regenerate Key1' and 'Regenerate Key2' buttons. A callout message in the center says: 'These keys are used to access your Azure AI service API. Do not share your keys. Store them securely—for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.' Below the message are buttons for 'Show Keys', 'KEY 1' (with a red box around it), 'KEY 2', 'Location/Region' set to 'westus2', and 'Endpoint' set to 'https://contoso-di.cognitiveservices.azure.com/' (also with a red box around it).

## Important

We recommend Microsoft Entra ID authentication with [managed identities for Azure resources](#) to avoid storing credentials with your applications that run in the cloud.

If you use an API key, store it securely somewhere else, such as in [Azure Key Vault](#). Don't include the API key directly in your code, and never post it publicly.

For more information about AI services security, see [Authenticate requests to Azure AI services](#).

C#/.NET

C#

```
//set `<your-endpoint>` and `<your-key>` variables with the values from the
//Azure portal to create your `AzureKeyCredential` and
//`DocumentIntelligenceClient` instance
string key = "<your-key>";
string endpoint = "<your-endpoint>";
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentIntelligenceClient client = new DocumentIntelligenceClient(new
Uri(endpoint), new AzureKeyCredential(key));
```

# Use a Microsoft Entra token credential

## (!) Note

Regional endpoints don't support Microsoft Entra authentication. Create a [custom subdomain](#) for your resource in order to use this type of authentication.

Authorization is easiest using the `DefaultAzureCredential`. It provides a default token credential, based upon the running environment, capable of handling most Azure authentication scenarios.

### C#/.NET

Here's how to acquire and use the `DefaultAzureCredential` for .NET applications:

1. Install the [Azure Identity library for .NET](#):

Console

```
dotnet add package Azure.Identity
```

PowerShell

```
Install-Package Azure.Identity
```

2. Register a Microsoft Entra application and create a new service principal.

3. Grant access to Document Intelligence by assigning the `Cognitive Services User` role to your service principal.

4. Set the values of the client ID, tenant ID, and client secret in the Microsoft Entra application as environment variables: `AZURE_CLIENT_ID`, `AZURE_TENANT_ID`, and `AZURE_CLIENT_SECRET`, respectively.

5. Create your `DocumentIntelligenceClient` instance including the `DefaultAzureCredential`:

C#

```
string endpoint = "<your-endpoint>";
var client = new DocumentIntelligenceClient(new Uri(endpoint), new
DefaultAzureCredential());
```

For more information, see [Authenticate the client](#).

## 4. Build your application

Create a client object to interact with the Document Intelligence SDK, and then call methods on that client object to interact with the service. The SDKs provide both synchronous and asynchronous methods. For more insight, try a [quickstart](#) in a language of your choice.

## Help options

The [Microsoft Q&A](#) and [Stack Overflow](#) forums are available for the developer community to ask and answer questions about Azure AI Document Intelligence and other services. Microsoft monitors the forums and replies to questions that the community has yet to answer. To make sure, use the following tags so that we see your question.

- Microsoft Q&A: `Azure AI Document Intelligence`.
- Stack Overflow: `azure-ai-document-intelligence`.

## Next steps

Explore [Document Intelligence REST API 2023-10-31-rest](#) operations.

# SDK target: REST API v3.1 (GA)

Article • 02/10/2025

## REST API version 2023-07-31 (GA)

Azure AI Document Intelligence is a cloud service that uses machine learning to analyze text and structured data from documents. The Document Intelligence software development kit (SDK) is a set of libraries and tools that enable you to easily integrate Document Intelligence models and capabilities into your applications. Document Intelligence SDK is available across platforms in C#/.NET, Java, JavaScript, and Python programming languages.

## Supported programming languages

Document Intelligence SDK supports the following languages and platforms:

 Expand table

Language → Document Intelligence SDK version	Package	Supported API version	Platform support
.NET/C# → latest (GA)	NuGet ↗	2023-07-31 (GA)	
Java → latest (GA)	Maven repository ↗	2023-07-31 (GA)	Windows, macOS, Linux
JavaScript → latest (GA)	npm ↗	2023-07-31 (GA)	Browser, Windows, macOS, Linux ↗
Python → latest (GA)	PyPI ↗	2023-07-31 (GA)	Windows, macOS, Linux

For more information on other SDK versions, see:

- [2024-02-29 v4.0 \(preview\)](#)
- [2022-08-31 v3.0 \(GA\)](#)
- [v2.1 \(GA\)](#)

## Supported Clients

The following tables present the correlation between each SDK version the supported API versions of the Document Intelligence service.

[+] Expand table

Language	SDK version	API version (default)	Supported clients
.NET/C# 4.1.0	v3.1 latest (GA)	2023-07-31	DocumentAnalysisClient DocumentModelAdministrationClient
.NET/C# 4.0.0	v3.0 (GA)	2022-08-31	DocumentAnalysisClient DocumentModelAdministrationClient
.NET/C# 3.1.x	v2.1	v2.1	FormRecognizerClient FormTrainingClient
.NET/C# 3.0.x	v2.0	v2.0	FormRecognizerClient FormTrainingClient

## Use Document Intelligence SDK in your applications

The Document Intelligence SDK enables the use and management of the Document Intelligence service in your application. The SDK builds on the underlying Document Intelligence REST API allowing you to easily use those APIs within your programming language paradigm. Here's how you use the Document Intelligence SDK for your preferred language:

### 1. Install the SDK client library

.NET CLI

```
dotnet add package Azure.AI.FormRecognizer --version 4.1.0
```

PowerShell

```
Install-Package Azure.AI.FormRecognizer -Version 4.1.0
```

## 2. Import the SDK client library into your application

```
C#/.NET

C#

using Azure;
using Azure.AI.FormRecognizer.DocumentAnalysis;
```

## 3. Set up authentication

There are two supported methods for authentication:

- Use a [Document Intelligence API key](#) with `AzureKeyCredential` from `azure.core.credentials`.
- Use a [token credential from azure-identity](#) to authenticate with [Microsoft Entra ID](#).

### Use your API key

Here's where to find your Document Intelligence API key in the Azure portal:

The screenshot shows the Azure portal interface for a resource named "Contoso-DI". The left sidebar contains navigation links like Home, Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, Keys and Endpoint (which is selected and highlighted with a red box), Encryption, Pricing tier, Networking, Identity, Cost analysis, Properties, Locks, Monitoring, Automation, and Help.

The main content area displays the "Contoso-DI | Keys and Endpoint" page. It includes a search bar, two "Regenerate Key" buttons, and a note about securely storing keys. Below this is a "Show Keys" button. The "KEY 1" field and the "Endpoint" field at the bottom are both highlighted with red boxes. The "Location/Region" field shows "westus2".

### Important

We recommend Microsoft Entra ID authentication with [managed identities for Azure resources](#) to avoid storing credentials with your applications that run in the cloud.

If you use an API key, store it securely somewhere else, such as in [Azure Key Vault](#). Don't include the API key directly in your code, and never post it publicly.

For more information about AI services security, see [Authenticate requests to Azure AI services](#).

C#/.NET

C#

```
//set `<your-endpoint>` and `<your-key>` variables with the values from the
//Azure portal to create your `AzureKeyCredential` and `DocumentAnalysisClient`
//instance
string key = "<your-key>";
string endpoint = "<your-endpoint>";
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentAnalysisClient client = new DocumentAnalysisClient(new Uri(endpoint),
credential);
```

## Use a Microsoft Entra token credential

### Note

Regional endpoints don't support Microsoft Entra authentication. Create a [custom subdomain](#) for your resource in order to use this type of authentication.

Authorization is easiest using the `DefaultAzureCredential`. It provides a default token credential, based upon the running environment, capable of handling most Azure authentication scenarios.

C#/.NET

Here's how to acquire and use the `DefaultAzureCredential` for .NET applications:

1. Install the [Azure Identity library for .NET](#):

Console

```
dotnet add package Azure.Identity
```

PowerShell

```
Install-Package Azure.Identity
```

2. Register a Microsoft Entra application and create a new service principal.
3. Grant access to Document Intelligence by assigning the **Cognitive Services User** role to your service principal.
4. Set the values of the client ID, tenant ID, and client secret in the Microsoft Entra application as environment variables: **AZURE\_CLIENT\_ID**, **AZURE\_TENANT\_ID**, and **AZURE\_CLIENT\_SECRET**, respectively.
5. Create your **DocumentAnalysisClient** instance including the **DefaultAzureCredential**:

C#

```
string endpoint = "<your-endpoint>";  
var client = new DocumentAnalysisClient(new Uri(endpoint), new  
DefaultAzureCredential());
```

For more information, see [Authenticate the client](#).

## 4. Build your application

Create a client object to interact with the Document Intelligence SDK, and then call methods on that client object to interact with the service. The SDKs provide both synchronous and asynchronous methods. For more insight, try a [quickstart](#) in a language of your choice.

## Help options

The [Microsoft Q & A](#) and [Stack Overflow](#) forums are available for the developer community to ask and answer questions about Azure AI Document Intelligence and other services. Microsoft monitors the forums and replies to questions that the community has yet to answer. To make sure that we see your question, tag it with **azure-form-recognizer**.

## Next steps

Explore Document Intelligence REST API 2023-07-31 operations.

# SDK target: REST API 2022-08-31 (GA)

Article • 11/19/2024

![Document Intelligence checkmark]../media/yes-icon.png) REST API version 2022-08-31 (GA)

Azure AI Document Intelligence is a cloud service that uses machine learning to analyze text and structured data from documents. The Document Intelligence software development kit (SDK) is a set of libraries and tools that enable you to easily integrate Document Intelligence models and capabilities into your applications. Document Intelligence SDK is available across platforms in C#/.NET, Java, JavaScript, and Python programming languages.

## Supported programming languages

Document Intelligence SDK supports the following languages and platforms:

[ ] Expand table

Language → Document Intelligence SDK version	Package	Supported API version	Platform support
.NET/C# → 4.0.0 (GA)	NuGet ↗	v3.0	Windows, macOS, Linux, Docker ↗
Java → 4.0.6 (GA)	Maven repository ↗	v3.0	Windows, macOS, Linux
JavaScript → 4.0.0 (GA)	npm ↗	v3.0	Browser, Windows, macOS, Linux ↗
Python → 3.2.0 (GA)	PyPI ↗	v3.0	Windows, macOS, Linux

For more information on other SDK versions, see:

- [2023-07-31 v3.1 \(GA\)](#)
- [v2.1 \(GA\)](#)

## Supported Clients

[ ] Expand table

<b>Language</b>	<b>SDK version</b>	<b>API version</b>	<b>Supported clients</b>
.NET/C#	4.0.0 (GA)	v3.0:2022-08-31 (default)	<a href="#">DocumentAnalysisClient</a>
Java			<a href="#">DocumentModelAdministrationClient</a>
JavaScript			
.NET/C#	3.1.x	v2.1 (default) v2.0	<a href="#">FormRecognizerClient</a> <a href="#">FormTrainingClient</a>
Java			
JavaScript			
.NET/C#	3.0.x	v2.0	<a href="#">FormRecognizerClient</a> <a href="#">FormTrainingClient</a>
Java			
JavaScript			
Python	3.2.x (GA)	v3.0:2022-08-31 (default)	<a href="#">DocumentAnalysisClient</a> <a href="#">DocumentModelAdministrationClient</a>
Python	3.1.x	v2.1 (default) v2.0	<a href="#">FormRecognizerClient</a> <a href="#">FormTrainingClient</a>
Python	3.0.0	v2.0	<a href="#">FormRecognizerClient</a> <a href="#">FormTrainingClient</a>

## Use Document Intelligence SDK in your applications

The Document Intelligence SDK enables the use and management of the Document Intelligence service in your application. The SDK builds on the underlying Document Intelligence REST API allowing you to easily use those APIs within your programming language paradigm. Here's how you use the Document Intelligence SDK for your preferred language:

### 1. Install the SDK client library

```
C#/.NET
```

```
.NET CLI
```

```
dotnet add package Azure.AI.FormRecognizer --version 4.0.0
```

```
PowerShell
```

```
Install-Package Azure.AI.FormRecognizer -Version 4.0.0
```

## 2. Import the SDK client library into your application

```
C#/.NET

C#

using Azure;
using Azure.AI.FormRecognizer.DocumentAnalysis;
```

## 3. Set up authentication

There are two supported methods for authentication:

- Use a [Document Intelligence API key](#) with `AzureKeyCredential` from `azure.core.credentials`.
- Use a [token credential from azure-identity](#) to authenticate with [Microsoft Entra ID](#).

### Use your API key

Here's where to find your Document Intelligence API key in the Azure portal:

The screenshot shows the Azure portal interface for a resource named "Contoso-DI". The left sidebar contains navigation links like Home, Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, Keys and Endpoint (which is selected and highlighted with a red box), Encryption, Pricing tier, Networking, Identity, Cost analysis, Properties, Locks, Monitoring, Automation, and Help.

The main content area displays the "Contoso-DI | Keys and Endpoint" page. It includes a search bar, two regenerate key buttons, and a note about securely storing keys. Below this, there are sections for "Show Keys", "KEY 1" (with a red box around it), "KEY 2", "Location/Region" set to "westus2", and "Endpoint" (https://contoso-di.cognitiveservices.azure.com/), which is also highlighted with a red box. A cursor arrow is visible at the bottom right of the page.

### Important

We recommend Microsoft Entra ID authentication with [managed identities for Azure resources](#) to avoid storing credentials with your applications that run in the cloud.

If you use an API key, store it securely somewhere else, such as in [Azure Key Vault](#). Don't include the API key directly in your code, and never post it publicly.

For more information about AI services security, see [Authenticate requests to Azure AI services](#).

C#/.NET

C#

```
//set `<your-endpoint>` and `<your-key>` variables with the values from the
//Azure portal to create your `AzureKeyCredential` and `DocumentAnalysisClient`
//instance
string key = "<your-key>";
string endpoint = "<your-endpoint>";
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentAnalysisClient client = new DocumentAnalysisClient(new Uri(endpoint),
credential);
```

## Use a Microsoft Entra token credential

### ⓘ Note

Regional endpoints do not support Microsoft Entra authentication. Create a [custom subdomain](#) for your resource in order to use this type of authentication.

Authorization is easiest using the `DefaultAzureCredential`. It provides a default token credential, based upon the running environment, capable of handling most Azure authentication scenarios.

C#/.NET

Here's how to acquire and use the `DefaultAzureCredential` for .NET applications:

1. Install the [Azure Identity library for .NET](#):

Console

```
dotnet add package Azure.Identity
```

PowerShell

```
Install-Package Azure.Identity
```

2. Register a Microsoft Entra application and create a new service principal.
3. Grant access to Document Intelligence by assigning the **Cognitive Services User** role to your service principal.
4. Set the values of the client ID, tenant ID, and client secret in the Microsoft Entra application as environment variables: **AZURE\_CLIENT\_ID**, **AZURE\_TENANT\_ID**, and **AZURE\_CLIENT\_SECRET**, respectively.
5. Create your **DocumentAnalysisClient** instance including the **DefaultAzureCredential**:

C#

```
string endpoint = "<your-endpoint>";  
var client = new DocumentAnalysisClient(new Uri(endpoint), new  
DefaultAzureCredential());
```

For more information, see [Authenticate the client](#).

## 4. Build your application

Create a client object to interact with the Document Intelligence SDK, and then call methods on that client object to interact with the service. The SDKs provide both synchronous and asynchronous methods. For more insight, try a [quickstart](#) in a language of your choice.

## Help options

The [Microsoft Q & A](#) and [Stack Overflow](#) forums are available for the developer community to ask and answer questions about Azure AI Document Intelligence and other services. Microsoft monitors the forums and replies to questions that the community has yet to answer. To make sure that we see your question, tag it with **azure-form-recognizer**.

## Next steps

[Explore Document Intelligence REST API v3.0](#)

[Try a Document Intelligence quickstart](#)

# SDK changelog, release history, and migration guide

Article • 12/18/2024

This reference article provides a version-based description of Document Intelligence feature and capability releases, changes, updates, and enhancements.

## December 2024 (GA) release

### .NET (C#)

- Document Intelligence 1.0.0
- Targets REST API 2024-11-30 (GA) by default

[Changelog/Release History ↗](#)

[Package \(NuGet\) ↗](#)

[Azure SDK for .NET](#)

[ReadMe ↗](#)

[Samples ↗](#)

[Migration guide ↗](#)

## August 2024 (preview) release

### .NET (C#)

- Document Intelligence 1.0.0-beta.3
- Targets REST API 2024-07-31-preview by default

[Changelog/Release History ↗](#)

[Package \(NuGet\) ↗](#)

[ReadMe ↗](#)

[Samples ↗](#)

[Migration guide ↗](#)

## March 2024 (preview) release

### .NET (C#)

- Document Intelligence 1.0.0-beta.2
- Targets REST API 2024-02-29-preview by default

[Changelog/Release History ↗](#)

[Package \(NuGet\) ↗](#)

[ReadMe ↗](#)

[Samples ↗](#)

[Migration guide ↗](#)

## November 2023 (preview) release

### .NET (C#)

- Document Intelligence 1.0.0-beta.1
- Targets REST API 2023-10-31-preview by default

[Package \(NuGet\) ↗](#)

[ReadMe ↗](#)

[Samples ↗](#)

[Migration guide ↗](#)

## August 2023 (GA) release

### C#

- Form Recognizer 4.1.0 (2023-08-10)
- Targets REST API 2023-07-31 by default
- REST API target 2023-02-28-preview is no longer supported
- [Breaking changes ↗](#)

[Changelog/Release History ↗](#)

[Package \(NuGet\)](#)

[ReadMe](#)

[Samples](#)

## April 2023 (preview) release

This release includes the following updates:

C#

- Form Recognizer 4.1.0-beta.1 (2023-04-13)
- Targets 2023-02-28-preview by default
- No breaking changes

[Changelog/Release History](#)

[Package \(NuGet\)](#)

[ReadMe](#)

[Samples](#)

## September 2022 (GA) release

This release includes the following updates:

### Important

The `DocumentAnalysisClient` and `DocumentModelAdministrationClient` now target API v3.0 GA, released 2022-08-31. Document Intelligence no longer supports clients from 2020-06-30-preview APIs or earlier.

C#

- Form Recognizer 4.0.0 GA (2022-09-08)
- Supports REST API v3.0 and v2.0 clients

[Changelog/Release History](#)

[Package \(NuGet\)](#)

[Migration guide ↗](#)

[ReadMe ↗](#)

[Samples ↗](#)

## August 2022 (preview) release

This release includes the following updates:

C#

- Form Recognizer 4.0.0-beta.5 (2022-08-09)
- Supports REST API 2022-06-30-preview clients

[Changelog/Release History ↗](#)

[Package \(NuGet\) ↗](#)

[SDK reference documentation](#)

## June 2022 (preview) release

This release includes the following updates:

C#

- Form Recognizer 4.0.0-beta.4 (2022-06-08)

[Changelog/Release History ↗](#)

[Package \(NuGet\) ↗](#)

[SDK reference documentation](#)

# Frequently asked questions

FAQ

This content applies to:  v4.0 (GA)  v3.1 (GA)  v3.0 (GA)  v2.1 (GA)

Azure AI Document Intelligence is a cloud-based service that uses machine-learning models to extract key/value pairs, text, and tables from your documents. The returned result is a structured JSON output. Document Intelligence use cases include automated data processing, enhanced data-driven strategies, and enriched document search capabilities.

## Overview

### Are Azure AI Document Intelligence and Azure AI Form Recognizer the same service?

Yes.

Azure AI Document Intelligence and Azure AI Form Recognizer are the same service. The service was renamed from Azure AI Form Recognizer to Azure AI Document Intelligence in July 2023. The service provides the same capabilities and features as before the renaming.

- **Pricing changes:** There are no changes to pricing. The names Cognitive Services and Applied AI Services continue to be used in Azure billing, cost analysis, price lists, and price APIs.
- **Breaking changes:** There are no breaking changes to APIs or client libraries.

### Does Document Intelligence integrate with other Microsoft services?

Yes.

Document Intelligence integrates with the following services:

- [AI Builder workflows](#)
- [Azure AI Search](#)
- [Azure Functions](#)
- [Azure Logic Apps](#)

# AI capabilities

## Can I use Document Intelligence with generative AI for document processing?

Yes.

You can also use a document generative AI solution to chat with your documents (RAG), generate captivating content from those documents, and access Azure OpenAI Service models on your data.

- With Azure AI Document Intelligence and Azure OpenAI combined, you can build an enterprise application to seamlessly interact with your documents using natural language. You can easily find answers, gain valuable insights, and generate new and engaging content from existing documents.
- You can find more details on the [retrieval-augmented generation pattern here](#).

## Can Document Intelligence help with semantic chunking within documents for retrieval-augmented generation?

Yes.

Document Intelligence can provide the building blocks to enable semantic chunking. Semantic chunking is a key step in retrieval-augmented generation (RAG) to ensure context dense chunks and relevance improvement.

- Document Intelligence provides a layout model that provides a visual decomposition of the document into lines, paragraphs, sections, headers, and footers.
- You can then choose to retrieve the results in markdown format, to further chunk the document on section or paragraph boundaries.

For more information, see [overview of RAG in Document Intelligence](#)

## Document Intelligence Studio

## Do I need specific permissions to access Document Intelligence Studio?

Yes.

You need an active [Azure account](#) and subscription with at least a Reader role to access Document Intelligence Studio.

For document analysis and prebuilt models, here are the role requirements for user scenarios:

- Basic
  - **Cognitive Services User:** You need this role for a [Document Intelligence](#) or [Azure Cognitive Services multiple-service](#) resource to use Document Intelligence Studio.
- Advanced
  - **Contributor:** You need this role to create a resource group or a Document Intelligence resource.

For custom model projects, here are the role requirements for user scenarios:

- Basic
  - **Cognitive Services User:** You need this role for a [Document Intelligence](#) or [Cognitive Services multiple-service](#) resource to train a custom model or analyze with trained models.
  - **Storage Blob Data Contributor:** You need this role for a storage account to create project and label data.
- Advanced
  - **Storage Account Contributor:** You need this role for the storage account to set up cross-origin resource sharing (CORS) settings. It's a one-time effort if you reuse the same storage account.
  - **Contributor:** You need this role to create a resource group and resources. **Contributor** or **Storage Account Contributor** role doesn't give you access to use your Document Intelligence resource or storage account if local (key-based) authentication is disabled. You still need the basic roles (**Cognitive Services User** and **Storage Data Blob Contributor**) to use the functions on Document Intelligence Studio.

For more information, see [Microsoft Entra built-in roles](#) and the sections about Azure role assignments in the [Document Intelligence Studio quickstart](#).

## Can I process documents with more than two pages in Document Intelligence Studio?

Yes, for paid-tier resources.

**No**, for free-tier resources.

- For free-tier (F0) resources, **only the first two pages** are analyzed whether you're using Document Intelligence Studio, the REST API, or client libraries.
- If you want to analyze **all pages** in a document, change to a paid (S0) resource. In Document Intelligence Studio, select the **Settings** (gear) button, select the **Resources** tab, and check the price tier to use for analyzing your documents.

## Can I change directories or subscriptions in Document Intelligence Studio?

**Yes.**

- To change a directory in Document Intelligence Studio, select the **Settings** (gear) button. Under **Directory**, select the directory from the list, and then select **Switch Directory**. Sign in again after you switch the directory.
- To change a subscription or resource, go to the **Resource** tab under **Settings**.

## Can I use Document Intelligence Studio with a resource that is configured with a firewall or virtual network?

**Yes.**

For v4.0 11-30-2024 (GA), auto labeling is hosted natively with the rest of the service, so there's no need for IP allowlisting. For any previous version, if your Document Intelligence resource is configured with a firewall or virtual network, you need to add the dedicated IP address 20.3.165.95 to the firewall allowlist for your Document Intelligence resource. Some functions in custom projects (for example, autolabel, project management and human in the loop) don't work if the public network access is disabled.

## When I upload a file in Document Intelligence Studio by "Fetch from URL" function, can I use a URL from my blob storage?

**Yes.**

If your Azure blob storage URL includes a SAS token, and is accessible from public networks. You can't use the **Fetch** function for storage accounts where the key access is disabled or

behind a firewall/VNet.

## Can I reuse or customize the labeling experience from Document Intelligence Studio and build it into my own application?

Yes.

The labeling experience from Document Intelligence Studio is open sourced in the [Toolkit repo ↗](#).

## Are there separate URL endpoints for Document Intelligence sovereign cloud regions?

Yes.

Document Intelligence Studio has separate URL endpoints for sovereign cloud regions:

- URL for the Azure US Government cloud (Azure Fairfax): [Document Intelligence Studio US Government ↗](#).
- URL Microsoft Azure operated by 21Vianet (Azure China): [Document Intelligence Studio China ↗](#).

## App development

## Can I develop applications using Azure AI Document Intelligence using the latest development options?

Yes.

Document Intelligence offers the latest development options within the following platforms:

- [REST API](#)
- [Document Intelligence Studio ↗](#)
- [C#/.NET ↗](#)
- [Java ↗](#)
- [JavaScript/TypeScript ↗](#)

- [Python ↗](#)

## Can I migrate my application to the latest version of Document Intelligence?

Yes.

The following table provides links to detailed instructions for migrating to the newest version of Document Intelligence:

 [Expand table](#)

Language/API	Migration guide
REST API	<a href="#">v3</a>
C#/.NET	<a href="#">4.0.0 ↗</a>
Java	<a href="#">4.0.0 ↗</a>
JavaScript	<a href="#">4.0.0 ↗</a>
Python	<a href="#">3.2.0 ↗</a>

## Can I specify a range of pages to be analyzed in a document?

Yes.

Use the `pages` parameter (supported in v2.1, v3.0, and later versions of the REST API) and specify pages for multiple-page PDF and TIFF documents. Accepted input includes the following ranges:

- Single pages. For example, if you specify `1, 2`, pages 1 and 2 are processed.
- Finite ranges. For example, if you specify `2-5`, pages 2 to 5 are processed.
- Open-ended ranges. For example, if you specify `5-`, all the pages from page 5 are processed. If you specify `-10`, pages 1 to 10 are processed.

You can mix these parameters together, and ranges can overlap. For example, if you specify `-5, 1, 3, 5-10`, pages 1 to 10 are processed.

The service accepts the request if it can process at least one page of the document. For example, using `5-100` on a five-page document is a valid input that means page 5 is processed.

If you don't provide a page range, the entire document is processed.

## Do you recommend using Document Intelligence Studio rather than the FOTT Sample Labeling tool for my project?

Yes.

We recommend [Document Intelligence Studio](#) most of the time because it can reduce your time for configuring Document Intelligence resources and storage services.

Only consider using the Form Testing Tool (FOTT) for the following scenarios:

- Your data must remain within a single machine. Use the [FOTT Sample Labeling tool](#) and a [Document Intelligence container](#).
- Your project is highly dependent on Document Intelligence V2.1 and you want to keep using the v2.1 APIs.

## Are there best practices to mitigate throttling?

Yes.

Document Intelligence uses autoscaling to provide the required computational resources on demand, while keeping customer costs low. To mitigate throttling during autoscaling, we recommend the following approach:

- Implement retry logic in your application.
- If you find that you're being throttled on the number of `POST` requests, consider adding a delay between the requests.
- Increase the workload gradually. Avoid sharp changes.
- [Create a support request](#) to increase transactions per second (TPS) limit.

Learn more about Document Intelligence [service quotas and limits](#).

# Custom models

## Can I improve an estimated accuracy score for a custom model?

Yes.

Variances in the visual structure of your documents can influence the accuracy of a model. Here are some tips:

- Include all variations of a document in the training dataset. Variations include different formats; for example, digital versus scanned PDFs.
- Separate visually distinct document types and train different models.
- Make sure that you don't have extraneous labels.
- For signature and region labeling, don't include the surrounding text.

For more information, see [Accuracy and confidence scores](#).

## Can I retrain a custom model?

No.

- Document Intelligence doesn't have an explicit retrain operation. Each train operation generates a new model.
- If you find that your model needs to retrain, you can add more samples to your training dataset and train a new model.
- You can also create a new model to compose with your original model as follows:
  1. Create a dataset for your new template.
  2. Label and train a new model.
  3. Validate that the new model performs well for your specific document types.
  4. Compose your new model with the existing model into a single endpoint. Document Intelligence can then determine the best model for each document to be analyzed.

For more information, see [composed models](#).

# Can I move my trained models from one environment (like beta) to another (like production)?

Yes.

You can use the [Copy API](#) to copy custom models from one Document Intelligence account into others that exist in any supported geographical region. For detailed instructions, see [Disaster recovery](#).

The copy operation is limited to copying models within the specific cloud environment where you trained the model. For instance, copying models from the public cloud to the Azure Government cloud isn't supported.

## Am I charged when using auto labeling?

Yes. Auto label incurs a cost which is equivalent to an analyze request for the corresponding model for a document.

## Am I charged when training a custom models?

Yes.

For [v4.0 11-30-2024 \(GA\)](#) custom neural models can be trained for free for a **maximum of 10 hours**. Whether you're training a single model for the 10 hours, or training multiple models for the total of 10 hours, you aren't charged for the first 10 hours. After using up the free 10 hours, you're **automatically charged by the extra training hour**. For details on prices, refer to the [pricing page](#). This new paid training feature enables training models for an extended duration to process larger documents. For more information on this paid training feature, check [custom neural model billing section](#).

For [v3.0 2022-08-31](#) or [v3.1 2023-07-31](#), custom neural models can be trained for free for a maximum of 20 training sessions, with each session capped at 30 minutes of training duration. Once you use up all of the 20 training sessions, you can submit Azure support ticket to increase the training session limit. To increase the limit, two training sessions are considered as one training hour, and you're charged per two sessions / one training hour. For details on the prices, refer to the [pricing page](#). For more information on ways to increase the limit, check [custom neural model billing section](#). For [v3.0](#) and [v3.1](#), paid training feature is unavailable. Paid training feature for custom neural model is only available on [v4.0](#).

## Storage account

# Is there an expiry time for the shared access signature (SAS) token that I for my storage account authentication?

Yes.

When you create a shared access signature (SAS), the default duration is 48 hours. After 48 hours, you need to create a new token.

Consider setting a longer duration period for the time that you're using your storage account with Document Intelligence.

# Can Document Intelligence access data in my storage account if it is behind a virtual network or firewall?

No, not directly.

Document Intelligence can't access your storage account if it's protected by a virtual network or firewall.

However, private Azure storage account access and authentication support [managed identities for Azure resources](#). When you use a managed identity, the Document Intelligence service can access your storage account by using an assigned credential.

If you intend to analyze your private storage account data by using FOTT, you must deploy the tool behind the virtual network or firewall.

Learn how to [create and use a managed identity for your Document Intelligence resource](#).

## Containers

### Is there a difference between disconnected and connected containers?

Yes.

Although the model capabilities are the same for connected and disconnected containers, the billing and connectivity methods differ:

- Connected containers send billing information to Azure by using a Document Intelligence resource on your Azure account. With connected containers, internet connectivity is required to send [billing information](#) to Azure. Document Intelligence connected containers send billing information to Azure by using a Document Intelligence resource on your Azure account. Connected containers don't send customer data, such as the image or text that's being analyzed, to Microsoft. For an example of the information that connected containers send to Microsoft for billing, see the [Azure AI container FAQ](#).
- [Disconnected containers](#) enable you to use APIs that are disconnected from the internet. [Billing information](#) isn't sent via the internet. Instead, Charges are based on a purchased commitment tier. Currently, disconnected container usage is available for Document Intelligence custom and invoice models.

## Can I use local storage for the Document Intelligence Sample Labeling Tool (FOTT) container?

Yes.

FOTT has a version that uses local storage. The version needs to be installed on a Windows machine. You can install it from [this location](#).

On the project page, specify the label folder URI as `/shared` or `/shared/sub-dir` if your labeling files are in a subdirectory. All other Document Intelligence Sample Labeling Tool behavior is the same as the hosted service.

## Is there a best practice for scaling up?

Yes.

For asynchronous calls, you can run multiple containers with shared storage. The container that's processing the `POST` `analyze` call stores the output in the storage. Then, any other container can fetch the results from the storage and serve the `GET` calls. The request ID isn't tied to a container.

For synchronous calls, you can run multiple containers, but only one container serves a request. Because it's a blocking call, any container from the pool can serve the request and send the response. Here, only one container is tied to a request at a time, and no polling is required.

## Can I set up containers with shared storage?

Yes.

The containers use the `Mounts: Shared` property while starting up for specifying the shared storage to store the processing files. To see the use of this property, refer to the [containers documentation](#).

## Security and privacy

### Does Document Intelligence store my data?

Yes, briefly.

For all features, Document Intelligence temporarily stores data and results in Azure Storage in the same region as the request. Your data is then deleted 24 hours from the time that you submit an analyze request. If you would like the data deleted sooner, you can call the [delete analyze response](#). This API marks the results for deletion and is available in the v4.0 API.

Learn more about [data, privacy, and security for Document Intelligence](#).

For trained custom models, the interim outputs after analysis and labeling are stored in the same Azure Storage location where you store your training data. The trained custom models are stored in Azure Storage in the same region, and are logically isolated with your Azure subscription and API credentials.

## More help and support

### Are there other resources available to provide solutions to Azure AI Document Intelligence questions?

Yes.

[Microsoft Q & A](#) is the home for technical questions and answers at Microsoft. You can filter queries that are specific to Document Intelligence.

### Can I provide direct feedback if the service doesn't recognize specific text, or recognizes it incorrectly, when I'm labeling documents?

Yes.

We continually update and improve the Document Intelligence models. You can [email the Document Intelligence team](#). If possible, share a sample document with the issue highlighted.

# Document processing models

Article • 03/14/2025

This content applies to: ✓ v4.0 (GA) | Previous versions: ✓ v3.1 (GA) ✓ v3.0 (GA) ✓ v2.1 (GA)

Azure AI Document Intelligence supports a wide variety of models that enable you to add intelligent document processing to your apps and flows. You can use a prebuilt domain-specific model or train a custom model tailored to your specific business needs and use cases. Document Intelligence can be used with the REST API or Python, C#, Java, and JavaScript client libraries.

## Note

- Document processing projects that involve financial data, protected health data, personal data, or highly sensitive data require careful attention.
- Be sure to comply with all [national/regional and industry-specific requirements](#).

## Model overview

The following table shows the available models for each stable API:

[ ] Expand table

Model Type	Model	2024-11-30 (GA)	2023-07-31 (GA)	2022-08-31 (GA)	v2.1 (GA)
Document analysis models	Read	✓	✓	✓	n/a
Document analysis models	Layout	✓	✓	✓	✓
Document analysis models	** General document	supported in layout model	✓	✓	n/a
Prebuilt models	Bank Check	✓	n/a	n/a	n/a
Prebuilt models	Bank Statement	✓	n/a	n/a	n/a
Prebuilt models	Paystub	✓	n/a	n/a	n/a
Prebuilt models	Contract	✓	✓	n/a	n/a
Prebuilt models	Health insurance card	✓	✓	✓	n/a
Prebuilt models	ID document	✓	✓	✓	✓
Prebuilt models	Invoice	✓	✓	✓	✓
Prebuilt models	Receipt	✓	✓	✓	✓
Prebuilt models	US Unified Tax*	✓	n/a	n/a	n/a
Prebuilt models	US 1040 Tax*	✓	✓	n/a	n/a
Prebuilt models	US 1095 Tax*	✓	n/a	n/a	n/a
Prebuilt models	US 1098 Tax*	✓	n/a	n/a	n/a
Prebuilt models	US 1099 Tax*	✓	n/a	n/a	n/a
Prebuilt models	US W2 Tax	✓	✓	✓	n/a
Prebuilt models	US W4 Tax	✓	n/a	n/a	n/a
Prebuilt models	US Mortgage 1003 URLA	✓	n/a	n/a	n/a
Prebuilt models	US Mortgage 1004 URAR	✓	n/a	n/a	n/a
Prebuilt models	US Mortgage 1005	✓	n/a	n/a	n/a
Prebuilt models	US Mortgage 1008 Summary	✓	n/a	n/a	n/a
Prebuilt models	US Mortgage closing disclosure	✓	n/a	n/a	n/a
Prebuilt models	Marriage certificate	✓	n/a	n/a	n/a

Model Type	Model	2024-11-30 (GA)	2023-07-31 (GA)	2022-08-31 (GA)	v2.1 (GA)
Prebuilt models	Credit card	✓	n/a	n/a	n/a
Prebuilt models	Business card	deprecated	✓	✓	✓
Custom classification model	Custom classifier	✓	✓	n/a	n/a
Custom extraction model	Custom neural	✓	✓	✓	n/a
Custom extraction model	Custom template	✓	✓	✓	✓
Custom extraction model	Custom composed	✓	✓	✓	✓
All models	Add-on capabilities	✓	✓	n/a	n/a

\* Contains submodels. See the model specific information for supported variations and subtypes.

\*\* All the General Document model capabilities are available in layout model. General model is no longer supported.

## Latency

Latency is the amount of time it takes for an API server to handle and process an incoming request and deliver the outgoing response to the client. The time to analyze a document depends on the size (for example, number of pages) and associated content on each page. Document Intelligence is a multitenant service where latency for similar documents is comparable but not always identical. Occasional variability in latency and performance is inherent in any microservice-based, stateless, asynchronous service that processes images and large documents at scale. Although we're continuously scaling up the hardware and capacity and scaling capabilities, you might still have latency issues at runtime.

## Add-on Capability

Following are the add-on capability available in document intelligence. For all models, except Business card model, Document Intelligence now supports add-on capabilities to allow for more sophisticated analysis. These optional capabilities can be enabled and disabled depending on the scenario of the document extraction. There are seven add-on capabilities available for the 2023-07-31 (GA) and later API version:

- [ocrHighResolution](#)
- [formulas](#)
- [styleFont](#)
- [barcodes](#)
- [languages](#)
- [keyValuePairs](#)
- [queryFields](#) Not available with the US.Tax models
- [searchablePDF](#) Only available for Read Model

[+] Expand table

Add-on Capability	Add-On/Free	2024-11-30 (GA)	2023-07-31 (GA)	2022-08-31 (GA)	v2.1 (GA)
Font property extraction	Add-On	✓	✓	n/a	n/a
Formula extraction	Add-On	✓	✓	n/a	n/a
High resolution extraction	Add-On	✓	✓	n/a	n/a
Barcode extraction	Free	✓	✓	n/a	n/a
Language detection	Free	✓	✓	n/a	n/a
Key value pairs	Free	✓	n/a	n/a	n/a
Query fields	Add-On*	✓	n/a	n/a	n/a
Searchable pdf	Add-On*	✓	n/a	n/a	n/a

## Model analysis features

[+] Expand table

Model ID	Content Extraction	Query fields	Paragraphs	Paragraph Roles	Selection Marks	Tables	Key-Value Pairs	Languages	Barcodes	Document Analysis	Formulas*
prebuilt-read	✓	✓					O	O			O
prebuilt-layout	✓	✓	✓	✓	✓	✓	O	O	O		O
prebuilt-contract	✓	✓	✓	✓	✓		O	O	✓	O	
prebuilt-healthInsuranceCard.us	✓	✓					O	O	✓	O	
prebuilt-idDocument	✓	✓					O	O	✓	O	
prebuilt-invoice	✓	✓			✓	✓	O	O	O	✓	O
prebuilt-receipt	✓	✓					O	O	✓	O	
prebuilt-marriageCertificate.us	✓	✓			✓		O	O	✓	O	
prebuilt-creditCard	✓	✓					O	O	✓	O	
prebuilt-check.us	✓	✓					O	O	✓	O	
prebuilt-payStub.us	✓	✓					O	O	✓	O	
prebuilt-bankStatement	✓	✓					O	O	✓	O	
prebuilt-mortgage.us.1003	✓	✓			✓		O	O	✓	O	
prebuilt-mortgage.us.1004	✓	✓			✓		O	O	✓	O	
prebuilt-mortgage.us.1005	✓	✓			✓		O	O	✓	O	
prebuilt-mortgage.us.1008	✓	✓			✓		O	O	✓	O	
prebuilt-mortgage.us.closingDisclosure	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.w2	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.w4	✓	✓					O	O	✓	O	
prebuilt-tax.us.1040 (various)	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.1095A	✓	✓					O	O	✓	O	
prebuilt-tax.us.1095C	✓	✓					O	O	✓	O	
prebuilt-tax.us.1098	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.1098E	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.1098T	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.1099 (various)	✓	✓			✓		O	O	✓	O	
prebuilt-tax.us.1099SSA	✓	✓					O	O	✓	O	
{ customModelName }	✓	✓	✓	✓	✓	✓	O	O	✓	O	

✓ - Enabled

O - Optional

\* - Premium features incur extra costs

Add-On\* - Query fields are priced differently than the other add-on features. See [pricing](#) for details.

## Bounding box and polygon coordinates

A bounding box (`polygon` in v3.0 and later versions) is an abstract rectangle that surrounds text elements in a document used as a reference point for object detection.

- The bounding box specifies position by using an x and y coordinate plane presented in an array of four numerical pairs. Each pair represents a corner of the box in the following order: upper left, upper right, lower right, lower left.
- Image coordinates are presented in pixels. For a PDF, coordinates are presented in inches.

## Language support

The deep-learning-based universal models in Document Intelligence support many languages that can extract multilingual text from your images and documents, including text lines with mixed languages. Language support varies by Document Intelligence service functionality. For a complete list, see the following articles:

- [Language support: document analysis models](#)
- [Language support: prebuilt models](#)
- [Language support: custom models](#)

## Regional availability

Document Intelligence is generally available in many of the [60+ Azure global infrastructure regions](#).

For more information, see our [Azure geographies](#) page to help choose the region that's best for you and your customers.

## Model details

This section describes the output you can expect from each model. You can extend the output of most models with add-on features.

### Read OCR



The Read API analyzes and extracts lines, words, their locations, detected languages, and handwritten style if detected.

*Sample document processed using the [Document Intelligence Studio](#):*

Analyze API version: 2022-01-30-preview

# IRS-Unterstützung in Katastrophenfällen

Von der US-Bundesregierung erklärtes Katastrophengebiet

Sie können den vollständigen oder teilweisen Verlust Ihres Wohngegenstums, Ihres Hausrats und Ihrer Kraftfahrzeuge infolge einer Beschädigung im Katastrophenfall auf Ihrer persönlichen Bundesinkommenssteuererklärung absetzen. Wenn Sie im Steuerjahr unmittelbar vor dem Steuerjahr, in dem der Katastrophenfall eintrat, Steuern abgeführt haben, können Sie Ihren Verlust auf einem Formular 1040X (Amended U.S. Individual Income Tax Return – Abgeänderte US-Einkommenssteuererklärung) für das Vorjahr absetzen, anstatt zu warten, bis Sie Ihre Steuererklärung für das laufende Jahr einreichen. Auf diese Weise können Sie sich Ihre Steuern, die Sie in Verbindung mit Ihrer Steuererklärung des Vorjahres bezahlt haben, teilweise oder ganz erstatten lassen.

**Was das für Sie bedeutet**

- Wenn Sie im vorausgegangenen Steuerjahr eine Bundesinkommenssteuererklärung abgegeben und Bundessteuern bezahlt haben ...
- können Sie jetzt u. U. eine abgeänderte Steuererklärung einreichen (oder damit bis nächstes Jahr warten), um Ihren Verlust geltend zu machen und sich die bezahlten Steuern erstatten zu lassen.
- Sie müssen Ihre Abzüge auf Formular 1040, Aufstellung A, einzeln aufzulösen.

**So machen Sie Ihren Verlust geltend**

- Fertigen Sie eine Liste Ihres gesamten verloren gegangenen Besitzes an.
- Stellen Sie die Anschaffungskosten (bzw. die angepasste Steuerbemessungsgrundlage) fest.
- Stellen Sie den Verkehrswert jedes Besitzguts fest.
  - Dabei handelt es sich um den Betrag, zu dem das Besitzgut unmittelbar vor Eintreten des Katastrophenfalls hätte verkauft werden können.
- Stellen Sie den derzeitigen Wert – also nach Eintreten des Katastrophenfalls – fest.
- Stellen Sie Versicherungs- oder andere Erstattungsleistungen fest, die Sie bereits erhalten haben oder voraussichtlich erhalten werden.

**So können Sie sich bei der Geltendmachung von Sachverlusten helfen lassen**

- Besorgen Sie sich die IRS-Publikation 2194, *Disaster Resource Guide* (Hilfreiche Informationen und Materialien für Katastrophenfälle), für Einzelpersonen und Unternehmen.
- Besorgen Sie sich rechnergestellte Kopien Ihrer letzjährigen Steuererklärung vom IRS.
- Der IRS kann Ihnen beim Erstellen Ihrer abgeänderten Steuererklärungen behilflich sein.

**So erhalten Sie weitere Informationen und Unterstützung**

- IRS-Hotline für Hilfe im Katastrophenfall – 1-866-562-5227 (Montag bis Freitag von 7:00 bis 19:00 Uhr Ortszeit).  
*Bitte treffen Sie vor dem Anruf bei der Hotline-Nummer bei Bedarf selbst Vorkehrungen für einen Dolmetscher.*
- Besuchen Sie die IRS-Website unter [www.irs.gov](http://www.irs.gov) oder
- Wenden Sie sich an Ihren Steuerberater.

Publication 3067 EN-DE (Rev. 10-2017) Catalog Number 53671Z Department of the Treasury Internal Revenue Service www.irs.gov

Content Result Code

IRS-Unterstützung in IRS Katastrophenfällen VAV

Von der US-Bundesregierung erklärt Sie können den vollständigen oder Kraftfahrzeuge infolge einer Besch Bundesinkommenssteuererklärung ab dem der Katastrophenfall eintrat, 1040X (Amended U.S. Individual Inc Vorjahr absetzen, anstatt zu warte Weise können Sie sich Ihre Steuern haben, teilweise oder ganz erstatt Was das für Sie bedeutet

- Wenn Sie im vorausgegangenen Ste Bundessteuern bezahlt haben ...
- können Sie jetzt u. U. eine abge warten), um Ihren Verlust geltend
- Sie müssen Ihre Abzüge auf Formu So machen Sie Ihren Verlust gelten
- Fertigen Sie eine Liste Ihres ge
- Stellen Sie die Anschaffungskosten Stellen Sie den Verkehrswert jedes
- Dabei handelt es sich um den Bet Katastrophenfalls hätte verkauft w
- Stellen Sie den derzeitigen Wert - Stellen Sie Versicherungs- oder an voraussichtlich erhalten werden.
- So können Sie sich bei der Geltend . Besorgen Sie sich die IRS-Publik Materialien für Katastrophenfälle)

Learn more: [read model](#)

## Layout analysis



The Layout analysis model analyzes and extracts text, tables, selection marks, and other structure elements like titles, section headings, page headers, page footers, and more.

Sample document processed using the [Document Intelligence Studio](#):

**NEWS TODAY**

Latest news and bulletin updates

**Title**  
Mirjam Nilsson

**The scoop of the day**  
The latest updates

**Content**  
Video provides a powerful way to help you prove your point. When you click Online Video, you can paste in the embed code for the video you want to add. You can also type a keyword to search online for the video that best fits your document.

**Polygon**  
To make your document look professionally produced, Word provides header, footer, cover page, and text box designs that complement each other. For example, you can add a matching cover page, header, and sidebar.

**Mirjam Nilsson**  
**The scoop of the day**  
The latest updates to get you through the day

Issue #10

Text Tables Selection marks

Title  
NEWS TODAY

Paragraph  
Issue #10

Title  
Latest news and bulletin updates

Paragraph  
Mirjam Nilsson

SectionHeading  
The scoop of the day The latest updates

Paragraph  
Video provides a powerful way to help you prove

[Learn more: layout model](#)

## Health insurance card



The health insurance card model combines powerful Optical Character Recognition (OCR) capabilities with deep learning models to analyze and extract key information from US health insurance cards.

*Sample US health insurance card processed using [Document Intelligence Studio](#):*

Analyze | All pages | Range

Fields Result Code

DocType: healthInsuranceCard.us

● Copays (2) #1

- 1 Amount \$1,500 Benefit Deductible
- 2 Amount \$1,000 Benefit Coinsurance Max

● GroupNumber #1 1000000 99.50%

● IdNumber #1 123456789 99.50%

● Prefix ABC 99.50%

● Insurer #1 PREMERA BLUE CROSS 99.50%

● Member #1 Employer Microsoft 99.50%

IdNumberSuffix 99.50%

Member ANGEL BROWN

Prefix Identification # Suffix ABC 123456789 01

Group # 1000000 Rx Group # BCAAXYZ Rx BIN# 987654 BCBS 456

PREMERA BLUE CROSS Microsoft

Medical Network HERITAGE Premera Dental YES Premera Vision YES

HEALTH SAVINGS PLAN Shared In and Out of Network Deductible \$1,500 Coinsurance Max \$1,000

Rx PPO

Note: Rx and Medical Cost-Shares are Shared

[Learn more: Health insurance card model](#)

## US tax documents



The US tax document models analyze and extract key fields and line items from a select group of tax documents. The API supports the analysis of English-language US tax documents of various formats and quality including phone-captured images, scanned documents, and digital PDFs. The following models are currently supported:

[Expand table](#)

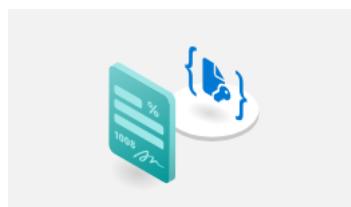
Model	Description	ModelID
US Tax W-2	Extract taxable compensation details.	prebuilt-tax.us.w2
US Tax W-4	Extract taxable compensation details.	prebuilt-tax.us.w4
US Tax 1040	Extract mortgage interest details.	prebuilt-tax.us.1040(variations)
US Tax 1095	Extract health insurance details.	prebuilt-tax.us.1095(variations)
US Tax 1098	Extract mortgage interest details.	prebuilt-tax.us.1098(variations)
US Tax 1099	Extract income received from sources other than employer.	prebuilt-tax.us.1099(variations)

Sample W-2 document processed using [Document Intelligence Studio](#):

Fields	Result	Code
DocType	tax.us.w2	
AdditionalInfo	#1	100.00%
AllocatedTips	#1	100.00%
874.2		
ControlNumber	#1	100.00%
000086242		
DependentCareBenefits	#1	100.00%
9873.2		
Employee	#1	
Employer	#1	
FederalIncomeTaxWithheld	#1	100.00%
3894.54		
LocalTaxInfos	(2) #1	
MedicareTaxWithheld	#1	100.00%
538.83		
MedicareWagesAndTips	#1	100.00%

[Learn more: Tax document models](#)

## US mortgage documents



The US mortgage document models analyze and extract key fields including borrower, loan, and property information from a select group of mortgage documents. The API supports the analysis of English-language US mortgage documents of various formats and quality including phone-captured images, scanned documents, and digital PDFs. The following models are currently supported:

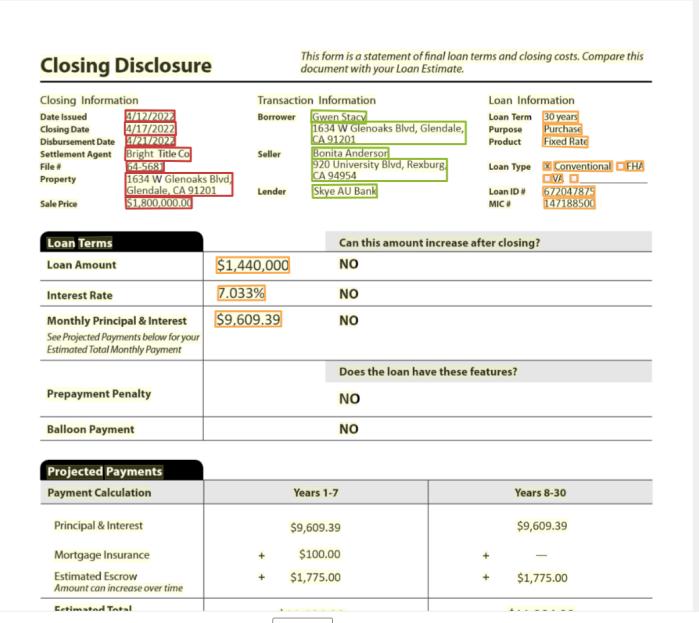
[Expand table](#)

Model	Description	ModelID
1003 End-User License Agreement (EULA)	Extract loan, borrower, property details.	prebuilt-mortgage.us.1003

Model	Description	ModelID
1004 Uniform Residential Appraisal Report (URAR)	Extract loan, borrower, property details.	prebuilt-mortgage.us.1004
1005 Verification of Employment	Extract loan, borrower, property details.	prebuilt-mortgage.us.1005
1008 Summary document	Extract borrower, seller, property, mortgage, and underwriting details.	prebuilt-mortgage.us.1008
Closing disclosure	Extract closing, transaction costs, and loan details.	prebuilt-mortgage.us.closingDisclosure

Sample Closing disclosure document processed using [Document Intelligence Studio](#):

Run analysis
Query fields
Analyze options



Fields
Result
Code

DocType: mortgage.us.closingDisclosure

●	C	#1	^
Closing	Date	99.50%	
	ClosingDate	4/17/2022	
	DisbursementDate	99.50%	
	4/21/2022		
	FileNumber	99.50%	
	64-5681		
	IssueDate	99.50%	
	4/12/2022		
	PropertyAddress	99.50%	
	1634 W Glenoaks Blvd, Glendale, CA 91201		
	HouseNumber		
	1634		
	Road		
	W Glenoaks Blvd		
	PostalCode		
	91201		
	City		
	Glendale		
	State		

Learn more: [Mortgage document models](#)

## Contract



The contract model analyzes and extracts key fields and line items from contractual agreements including parties, jurisdictions, contract ID, and title. The model currently supports English-language contract documents.

Sample contract processed using [Document Intelligence Studio](#):

[Run analysis](#) [Analyze options](#)

Fields Content Result Code

DocType: contract

Jurisdictions #1

Clause

This Agreement shall be governed by and construed in accordance with the internal laws of the State of Washington applicable to agreements made and to be performed entirely within such state.

Region

Washington

Parties (2) #1

EffectiveDate #1 99.90% 15 day of October, 2022

ExecutionDate #1 99.90% 15 day of October, 2022

Title #1 99.90% WEB HOSTING AGREEMENT

WEB HOSTING AGREEMENT

This web Hosting Agreement is entered as of this [5 day of October, 2022] "Effective Date" by and between Contoso Corporation, a Washington corporation, having its principal place of business at 1 Microsoft Way, Redmond, Washington, 98054; Contoso, and AdventureWorks Cycles, a Washington corporation, having its principal place of business at 98 NW 76th Street, Suite 54, Bellevue, Washington, 98007; AdventureWorks.

This agreement shall void and nullify any and all previous agreements to this date between Contoso and AdventureWorks.

There shall be no additional fees of any kind paid to Contoso, other than those stated within this agreement for software usage and/or bandwidth usage. AdventureWorks agrees to pay Contoso \$0.01 (one cent) per access up to 400,000 accesses thereafter payment shall be \$0.005 (one-half cent) per access. AdventureWorks shall send this amount to Contoso by no later than Wednesday for accesses used from the previous week (Monday thru Sunday).

Contoso must provide a person(s) to correct any technical problems (Server being down or inaccessible) 24 hours per day, 7 days per week. This person(s) must be available by beeper or telephone. AdventureWorks shall provide this same 24 hour service at the broadcast location.

This Agreement shall be governed by and construed in accordance with the internal laws of the State of Washington applicable to agreements made and to be performed entirely within such state.

All parties have read and fully agree to all terms and conditions as set forth in this Web Hosting Agreement.

Contoso Corporation  
By: Angel Brown  
Title: CTO  
*Angel Brown*

Adventure Works Cycle  
By: Aaron Smith  
Title: CEO  
*Aaron Smith*



[Learn more: contract model](#)

## US Bank Check



The contract model analyzes and extracts key fields from check including check details, account details, amount, memo, is extracted from US bank checks.

A bank check sample processed using [Document Intelligence Studio](#):

Contoso Bank

Contoso Ltd. No. 370654 98 - 2  
123 Main St, Redmond, WA 98052 125

Date: June 20, 2024

Pay To The Order Of 22<sup>nd</sup> Century Insurance John Doe

\$ \*\*\*123,456.00

One Hundred Twenty-Three Thousand Four Hundred Fifty-Six And 00/100 Dollars

Memo: Fees & Charges

WordAmount  
Content: One Hundred Twenty-Three Thousand Four Hundred Fifty-Six And 00/100 Dollars  
Value: 123456  
Confidence: 99.50%

Authorized Signature: *Ukeer*

137065411 1250000241 8495011554321

[Learn more: contract model](#)

## US Bank Statement



The bank statement model analyzes and extracts key fields and line items from US bank statements account number, bank details, statement details, and transaction details.

*Sample bank statement processed using Document Intelligence Studio* :

**CONTOSO BANK**

Contoso Bank  
1234 Grove Street  
New York, NY 10001

Account Number  
168552031585

Account Type  
Simple Checking

Statement Period  
Nov 1, 2023 – Nov 30, 2023

**Account Summary**

Beginning Balance on Nov 1, 2023	\$ 3,000.00	
Deposits / Credits	+ 2,500.00	
Withdrawals / Debits	- 1,200.00	
Service Fees	- 10.00	
<b>Ending Balance on Nov 30, 2023</b>	<b>\$ 4,290.00</b>	

**Deposits / Credits**

Date	Description	Amount
11/01/2023	Deposit	\$ 2,500.00
11/29/2023	Deposit	\$ 0.00
<b>Total Deposits / Credits</b>		<b>\$ 2,500.00</b>

**Withdrawals / Debits**

Date	Description	Amount
11/02/2023	Online Payment	-\$ 200.00
11/18/2023	Online Transfer To xxxxxxxx1375	-\$ 1,000.00
<b>Total Withdrawals / Debits</b>		<b>-\$ 1,200.00</b>

**Service Fees**

Date	Description	Amount
11/02/2023	International Transaction Fee	-\$ 10.00
<b>Total Service Fees</b>		<b>-\$ 10.00</b>

Page 1 of 1

Classified as Microsoft Confidential

< 1 of 1 >

[Learn more: contract model](#)

## PayStub



The paystub model analyzes and extracts key fields and line items from documents and files with payroll related information.

*Sample paystub processed using Document Intelligence Studio* :

**CONTOSO LTD**

**EARNINGS STATEMENT**

EARNINGS	HOURS	RATE	CURRENT	YTD
Regular Hours	56.00	\$ 24.00	\$ 1,344.00	\$ 5,000.00
Overtime	2.00	\$ 36.00	\$ 72.00	\$ 500.00
Sick Pay	8.00	\$ 24.00	\$ 192.00	\$ 500.00
Vacation Pay	8.00	\$ 24.00	\$ 192.00	\$ 500.00
Holiday Pay	8.00	\$ 24.00	\$ 192.00	\$ 500.00
	82.00		\$ 1,992.00	\$ 7,000.00

DEDUCTIONS	RATE	CURRENT	YTD
Aftertax Health Ins	\$ 120.00	\$ 500.00	
Aftertax Dental Ins	\$ 35.00	\$ 400.00	
Aftertax 401k	\$ 78.00	\$ 200.00	
Child Support	\$ 155.00	\$ 200.00	
Garnishment	\$ 45.00	\$ 500.00	
	\$ 433.00		\$ 1,800.00

TAXES	RATE	CURRENT	YTD
Federal Income Tax	\$ 100.00	\$ 500.00	
Social Security	0.0620	\$ 123.50	\$ 475.22
Medicare	0.0145	\$ 28.85	\$ 199.00
State Income Tax	\$ 50.00	\$ 200.00	
Local Income Tax	\$ 10.00	\$ 40.00	
	\$ 312.35		\$ 1,414.22

NET PAY	CURRENT	YTD
	\$ 1,246.61	\$ 3,785.78

Direct Deposited To Account of: Carl Anderson

Routing Number: 485066128

Account Number: 8300567112

Amount: \$ 1,246.61

**CONTOSO LTD**

123 Main Street  
Redmond, WA 98052

1/12/2024 1001

DocType: payStub.us

- CurrentPeriodDeductions #1  
433
- CurrentPeriodGrossPay #1  
1992
- CurrentPeriodNetPay #1  
1246.61
- CurrentPeriodTaxes #1  
312.39
- EmployeeAddress #1  
203 Denver Blvd Seattle, WA 98040
- HouseNumber  
203
- Road  
Denver Blvd
- PostalCode  
98040
- City  
Seattle
- State  
WA
- StreetAddress  
203 Denver Blvd
- EmployeeName #1  
Carl Anderson

Learn more: contract model

## Invoice



The invoice model automates processing of invoices to extracts customer name, billing address, due date, and amount due, line items, and other key data.

Sample invoice processed using Document Intelligence Studio [↗](#):

Analyze API version: 2021-09-30-preview

**CONTOSO LTD**

**INVOICE**

INVOICE: #1  
INVOICE DATE: #1  
DUE DATE: #1  
CUSTOMER NAME: #1  
SERVICE PERIOD: #1 - #1  
CUSTOMER ID: #1

BILL TO: Microsoft Corp  
123 Other St.  
Redmond WA 98052

SHIP TO: Microsoft Finance  
123 Ship St.  
Redmond WA 98052

SERVICE ADDRESS:  
#1 Service St.  
Redmond WA 98052

SALESPERSON P.O. NUMBER REQUISITIONER SHIPPED VIA F.O.B. POINT TERMS

DATE ITEM CODE DESCRIPTION QTY UM PRICE TAX AMOUNT

REDACTED 123 Consulting Services 1 EA \$100.00 \$0.00 \$100.00

REDACTED 123 Consulting Services 1 EA \$100.00 \$0.00 \$100.00

REDACTED 123 Consulting Services 1 EA \$100.00 \$0.00 \$100.00

REDACTED

THANK YOU FOR YOUR BUSINESS!

REMIT TO: Contoso Corp  
123 Remit St.  
New York, NY, 10001

Subtotal: \$100.00  
Sales Tax: \$0.00  
Total: \$100.00  
Previous Unpaid Balance: \$0.00  
Amount Due: \$100.00

Values Result Code

● AmountDue #1 97.30%

610

● BillingAddress #1 94.70%

123 Bill St, Redmond WA, 98052

● BillingAddressRecipient #1 95.70%

Microsoft Finance

● CustomerAddress #1 94.70%

123 Other St, Redmond WA, 98052

● CustomerAddressRecipient #1 95.60%

Microsoft Corp

● CustomerId #1 96.40%

CID-12345

Learn more: invoice model

## Receipt



Use the receipt model to scan sales receipts for merchant name, dates, line items, quantities, and totals from printed and handwritten receipts. The version v3.0 also supports single-page hotel receipt processing.

*Sample receipt processed using Document Intelligence Studio ↗:*

Analyze API version: 2021-09-30-preview ▾

Values	Result	Code
● Items (2) #1		
1 Name	Surface Pro 6	98.40%
Quantity		96.90%
1		
TotalPrice	999	98.50%
2 Name	SurfacePen	96.90%
Quantity		97.00%
1		
TotalPrice	99.99	98.50%
● Locale		98.70%

[Learn more: receipt model](#)

## Identity document (ID)



Use the Identity document (ID) model to process U.S. Driver's Licenses (all 50 states and District of Columbia) and biographical pages from international passports (excluding visa and other travel documents) to extract key fields.

*Sample U.S. Driver's License processed using Document Intelligence Studio ↗:*

Analyze

Values	Result	Code
Address #1	123 STREET ADDRESS YOUR CITY WA 99999-1234	87.20%
CountryRegion	USA	99.50%
DateOfBirth #1	1958-01-06	98.90%
DateOfExpiration #1	2020-08-12	98.60%
DocumentNumber	WDLABCD456DG	97.50%
Endorsements #1	L	98.40%
FirstName #1	LIAM R.	84.60%
LastName #1	TALBOT	93.10%
Locale	en-US	100.00%

[Learn more: identity document model](#)

## Marriage certificate



Use the marriage certificate model to process U.S. marriage certificates to extract key fields including the individuals, date, and location.

*Sample U.S. marriage certificate processed using Document Intelligence Studio :*

Run analysis

Query fields

Analyze options

Fields

DocType: marriageCertificate.us

Value	Result	Code
IssueDate #1	March 22, 2017	99.50%
MarriageDate #1	19th March AD, 2017	98.60%
MarriagePlace #1	Detroit Wayne MICHIGAN,	99.50%
Spouse1Address #1	Detroit, Michigan	99.50%
Spouse1Age #1	30	99.50%
Spouse1BirthPlace #1	Detroit, Michigan	99.50%

[Learn more: identity document model](#)

## Credit card



Use the credit card model to process credit and debit cards to extract key fields.

*Sample credit card processed using [Document Intelligence Studio](#):*

The screenshot shows the Document Intelligence Studio interface with a credit card image and its analysis results. The card is from Contoso Bank, with the number 5412 1234 5656 8888, valid until 01/28, and issued to ADAM SMITH. The analysis results table on the right lists the following fields:

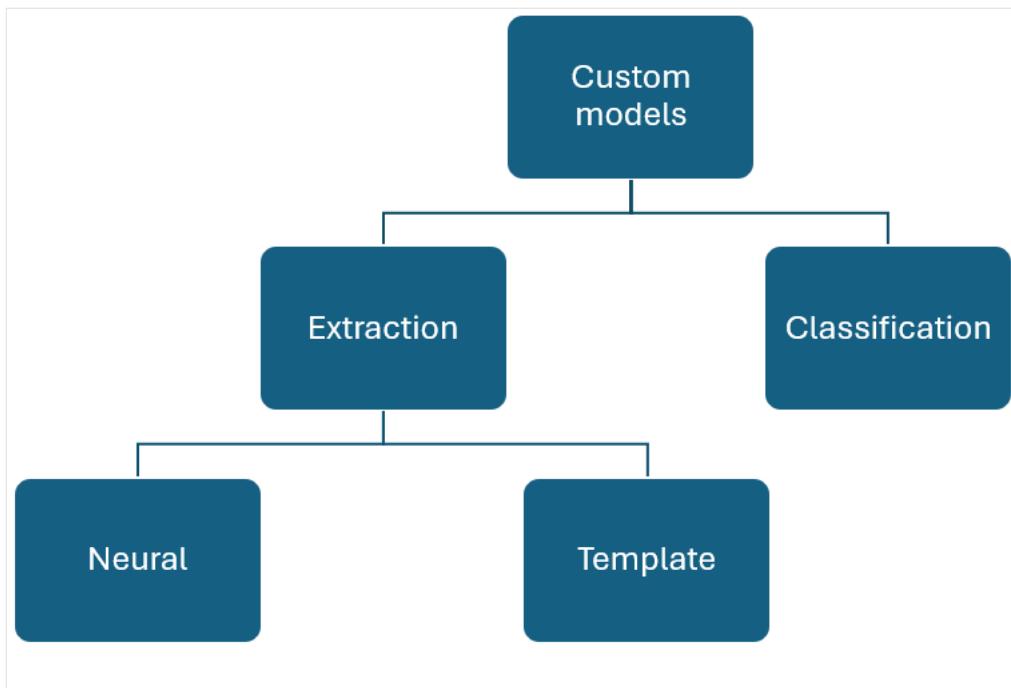
Field	Value	Confidence
CardHolderName	ADAM SMITH	99.50%
CardNumber	5412 1234 5656 8888	99.50%
CardVerificationValue	123	99.50%
CustomerServicePhoneNumbers	+1 200-345-6789, +1 200-000-8888	99.50%
ExpirationDate	01/28	99.50%
IssuingBank	Contoso Bank	99.50%
PaymentNetwork	mastercard	99.10%

[Learn more: identity document model](#)

## Custom models



Custom models can be broadly classified into two types. Custom classification models that support classification of a "document type" and custom extraction models that can extract a defined schema from a specific document type.



Custom document models analyze and extract data from forms and documents specific to your business. They recognize form fields within your distinct content and extract key-value pairs and table data. You only need one example of the form type to get started.

Version v3.0 and later custom models support signature detection in custom template (form) and cross-page tables in both template and neural models. [Signature detection](#) looks for the presence of a signature, not the identity of the person who signs the document. If the model returns **unsigned** for signature detection, the model didn't find a signature in the defined field.

*Sample custom template processed using [Document Intelligence Studio](#):*

### Label data

The screenshot shows the 'Label data' section of the Document Intelligence Studio. On the left, there are four thumbnail previews of invoices: 'Invoice\_1.pdf', 'Invoice\_2.pdf', 'Invoice\_3.pdf', and 'Invoice\_4.pdf'. In the center, the first invoice is displayed with its content: 'Contoso' address (1 Redmond way Suite 6000 Redmond, WA 98043), 'Invoice For: Microsoft' (1020 Enterprise Way Sunnyvale, CA 97659), and a table with columns 'Invoice Number', 'Invoice Date', 'Invoice Due Date', 'Charges', and 'VAT ID'. The 'Charges' column shows '\$56,051.49' and 'PT'. To the right, a list of labeled fields is shown:

Label	Value	Action
Receipt No	2468	⋮
Sold To	Fabrikam Residences	⋮
ID #	1197531	⋮
Live Delivery?	<input checked="" type="checkbox"/> unselected	⋮
Online Delivery?	<input checked="" type="checkbox"/> selected	⋮
Video Delivery?	<input checked="" type="checkbox"/> selected	⋮

A large red arrow points to the 'ID #' row in the list.

[Learn more: custom model](#)

## Custom extraction



Custom extraction model can be one of two types, **custom template**, **custom neural**. To create a custom extraction model, label a dataset of documents with the values you want extracted and train the model on the labeled dataset. You only need five examples of the same form or document type to get started.

*Sample custom extraction processed using [Document Intelligence Studio](#):*

**Custom extraction models**

Extract information from forms and documents with custom extraction models. Train a model by labeling as few as 5 example documents. (The same labeled dataset can train all types of custom extraction models.) [Learn more about custom extraction models](#).

**Template (Custom form) models**

Template models work well when the target documents share a common visual layout. Training only takes a few minutes, and more than 100 languages are supported.



**Neural (Custom document) models**

Neural models can flexibly handle both structured and unstructured documents. Training takes up to half an hour, and currently only English language documents are supported. The current version can extract inline field data and checkboxes. Neural models are available only in select regions. [Click here for details](#).

**HOUSE RENTAL AGREEMENT**

This House Rental Agreement ("Agreement," "rental agreement," or "lease") is entered into between **Opay LLC** (Landlord) and **Sarah Johnson and Jamie Savins and Sarah Williams** (Tenants). If more than one person is named as Tenant, they shall be jointly and severally liable and responsible under the terms of this Agreement. This lease Agreement involves a residential house, yard, and related facilities located at **12 Bellevue Dr, Ball Lake, City, MI, 49303** (the "premises"). The date of this Agreement is **January 15, 2024**.

1. Landlord rents to Tenant, unfurnished, the premises on a month to month basis, terminable by either party at the end of any calendar month on at least 30 days notice to the other party. Tenant shall be entitled to possession of the premises and rent shall commence on **April 15, 2024**. Tenant shall not assign, sublease, or allow anyone other than persons permitted under this lease to at any time be in possession of any portion of the premises. Landlord will provide five (5)

[Learn more: custom template model](#)

[Learn more: custom neural model](#)

## Custom classifier



The custom classification model enables you to identify the document type before invoking the extraction model. The classification model is available starting with the **2023-07-31 (GA)** API. Training a custom classification model requires at least two distinct classes and a minimum of five samples per class.

[Learn more: custom classification model](#)

## Composed models

A composed model is created by taking a collection of custom models and assigning them to a single model built from your form types. You can assign multiple custom models to a composed model called with a single model ID. You can assign up to 200 trained custom models to a single composed model.

*Composed model dialog window in [Document Intelligence Studio](#):*

The screenshot shows the Microsoft Form Recognizer Studio - Preview interface. On the left, there's a sidebar with navigation items: 'Custom Form' (highlighted with a red box), 'composed', 'Label data', 'Models' (highlighted with a red box), 'Test', and 'Settings'. The main area is titled 'Models' and contains a table with three rows. The first row has columns 'Model ID' (8aa16866-16fe-44ca-b13a-8bfc6ad1d) and 'Model Description'. The second row has columns 'Model ID' (773fb140-f173-47a2-8aa9-a5fce1ceb) and 'Model Description'. The third row has columns 'Model ID' (4c493f98-87c3-4f6d-b0d8-3a1aab49e) and 'Model Description'. At the top of the main area, there are buttons for 'Compose' (highlighted with a red box), 'Test', 'Download', and 'Delete'.

[Learn more: custom model](#)

## Input requirements

Supported file formats:

[Expand table](#)

Model	PDF	Image: JPEG/JPG, PNG, BMP, TIFF, HEIF	Microsoft Office: Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

**Note**

The [Sample Labeling tool](#) doesn't support the BMP file format. The limitation is derived from the tool not the Document Intelligence Service.

## Version migration

Learn how to use Document Intelligence v3.0 in your applications by following our [Document Intelligence v3.1 migration guide](#)

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence bank check model

Article • 12/11/2024

The Document Intelligence bank check model combines powerful Optical Character Recognition (OCR) capabilities with deep learning models to analyze and extract data from US bank checks. The API analyzes printed checks; extracts key information, and returns a structured JSON data representation. The latest version 4.0 for bank check supports signature detection on bank checks.

[+] Expand table

Feature	version	Model ID
Check model	v4.0: <a href="#">2024-11-30 (GA)</a>	<a href="#">prebuilt-check.us</a>

## Check data extraction

A check is a secure way to transfer amount from payee's account to receiver's account. Businesses use check to pay their vendors as a signed document to instruct the bank for payment. See how data, including check details, account details, amount, memo, is extracted from bank check US. You need the following resources:

- An Azure subscription—you can [create one for free](#)
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (**F0**) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

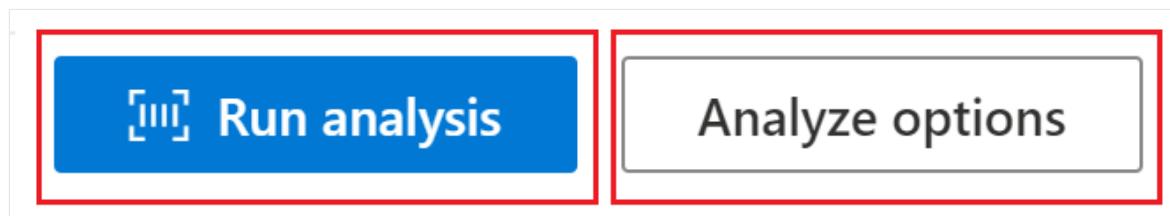
The screenshot shows the Azure portal interface for the 'Contoso-DI' resource. On the left, a navigation menu lists various service management options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, Keys and Endpoint (which is selected and highlighted with a red box), Encryption, Pricing tier, Networking, Identity, Cost analysis, Properties, Locks, Monitoring, Automation, and Help. The main content area is titled 'Contoso-DI | Keys and Endpoint'. It contains a note: 'These keys are used to access your Azure AI service API. Do not share your keys. Store them securely—for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.' Below the note, there are four fields: 'KEY 1' (redacted), 'KEY 2' (redacted), 'Location/Region' set to 'westus2', and 'Endpoint' set to 'https://contoso-di.cognitiveservices.azure.com/'. Each field has a copy icon (a blue square with a white 'C') to its right.

## Document Intelligence Studio

### ⓘ Note

Document Intelligence Studio is available with v3.1 and v3.0 APIs.

1. On the [Document Intelligence Studio home page](#), select **check**.
2. You can analyze the sample check or upload your own files.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options**:



[Try Document Intelligence Studio](#)

## Input requirements

- Supported file formats:

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

# Supported languages and locales

For a complete list of supported languages, see our [prebuilt model language support](#) page.

## Field extractions

For supported document extraction fields, see the [bank check model schema](#) page in our GitHub sample repository.

## Supported locales

The [prebuilt-check.us](#) version 2024-11-30 (GA) supports the `en-us` locale.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#)
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

---

## Feedback

Was this page helpful?



[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence bank statement model

Article • 12/11/2024

The Document Intelligence bank statement model combines powerful Optical Character Recognition (OCR) capabilities with deep learning models to analyze and extract data from US bank statements. The API analyzes printed bank statements; extracts key information such as account number, bank details, statement details, transaction details, and fees; and returns a structured JSON data representation. With V4.0 GA, you can now extract check tables in the US bank statements.

[ Expand table

Feature	version	Model ID
Bank statement model	v4.0: <a href="#">2024-11-30 (GA)</a>	<a href="#">prebuilt-bankStatement.us</a>

## Bank statement data extraction

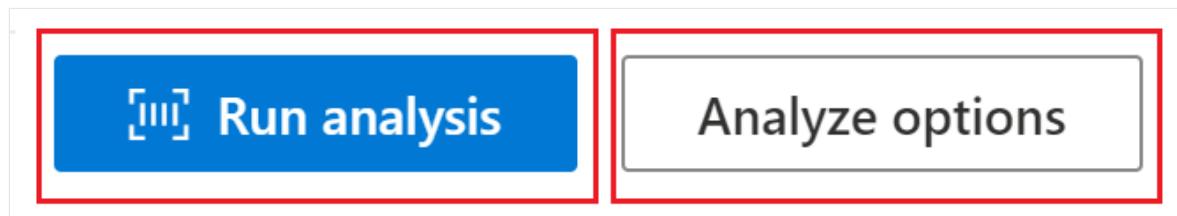
A bank statement helps review account's activities during a specified period. It's an official statement that helps in detecting fraud, tracking expenses, accounting errors and record the period's activities. See how data is extracted using the [prebuilt-bankStatement.us](#) model. You need the following resources:

- An Azure subscription—you can [create one for free](#)
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier ([F0](#)) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

The screenshot shows the Azure portal interface for a resource named 'Contoso-DI'. The left sidebar lists various management options under 'Resource Management', with 'Keys and Endpoint' selected and highlighted with a red box. The main content area displays two key fields ('KEY 1' and 'KEY 2') and an 'Endpoint' field. The 'Endpoint' field contains the URL 'https://contoso-di.cognitiveservices.azure.com/' and is also highlighted with a red box.

## Document Intelligence Studio

1. On the [Document Intelligence Studio home page](#), select **bank statements**.
2. You can analyze the sample bank statement or upload your own files.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options** :



## Input requirements

- Supported file formats:

[Expand table](#)

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

## Supported languages and locales

For a complete list of supported languages, see our [prebuilt model language support](#) page.

# Field extractions

For supported document extraction fields, see the [bank statement model schema](#) page in our GitHub sample repository.

## Supported locales

The `prebuilt-bankStatement.us` version 2027-11-30 supports the `en-us` locale.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#)
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

---

## Feedback

Was this page helpful?



[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence contract model

Article • 12/11/2024

This content applies to:  v4.0 (GA) | Previous version:  v3.1 (GA) ::moniker-end

The Document Intelligence contract model uses powerful Optical Character Recognition (OCR) capabilities to analyze and extract key fields and line items from a select group of important contract entities. Contracts can be of various formats and quality including phone-captured images, scanned documents, and digital PDFs. The API analyzes document text; extracts key information such as Parties, Jurisdictions, Contract ID, and Title; and returns a structured JSON data representation. The model currently supports English-language document formats.

## Automated contract processing

Automated contract processing is the process of extracting key contract fields from documents. Historically, the contract analysis process is achieved manually and, hence, very time consuming. Accurate extraction of key data from contracts is typically the first and one of the most critical steps in the contract automation process.

## Development options

Document Intelligence v4.0: 2024-11-30 (GA) supports the following tools, applications, and libraries:

 Expand table

Feature	Resources	Model ID
Contract model	<ul style="list-style-type: none"><li><a href="#">Document Intelligence Studio</a></li><li><a href="#">REST API</a></li><li><a href="#">C# SDK</a></li><li><a href="#">Python SDK</a></li><li><a href="#">Java SDK</a></li><li><a href="#">JavaScript SDK</a></li></ul>	prebuilt-contract

## Input requirements

- Supported file formats:

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

# Try contract document data extraction

See how data, including customer information, vendor details, and line items, is extracted from contracts. You need the following resources:

- An Azure subscription—you can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (`F0`) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

The screenshot shows the Azure portal interface for a resource named 'Contoso-DI'. The left sidebar lists various service categories like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, Keys and Endpoint (which is selected and highlighted with a red box), Encryption, Pricing tier, Networking, Identity, Cost analysis, Properties, Locks, Monitoring, Automation, and Help. The main content area displays the 'Keys and Endpoint' settings. It includes a note about securely storing keys, two text input fields for 'KEY 1' and 'KEY 2' (both highlighted with red boxes), a dropdown for 'Location/Region' set to 'westus2', and a text input field for 'Endpoint' containing the URL 'https://contoso-di.cognitiveservices.azure.com/'. There are also 'Regenerate Key1' and 'Regenerate Key2' buttons at the top.

## Document Intelligence Studio

1. On the [Document Intelligence Studio home page](#), select **Tax Documents**.
2. You can analyze the sample tax documents or upload your own files.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options**:



[Try Document Intelligence Studio](#)

# Supported languages and locales

For a complete list of supported languages, see our [Language Support—prebuilt models](#) page.

## Field extraction

- For supported document extraction fields, see the [contract model schema](#) page in our GitHub sample repository.
- The contract key-value pairs and line items extracted are in the `documentResults` section of the JSON output.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | Get help at Microsoft Q&A

# Document Intelligence credit card model

Article • 12/11/2024

This content applies to: v4.0

The Document Intelligence credit/debit card model uses powerful Optical Character Recognition (OCR) capabilities to analyze and extract key fields from credit and debit cards. Credit cards and debit cards can be of various formats and quality including phone-captured images, scanned documents, and digital PDFs. The API analyzes document text; extracts key information such as card number, issuing bank, and expiration date; and returns a structured JSON data representation. The model currently supports English-language document formats.

## Automated card processing

Automated Credit/Debit card processing is the process of extracting key fields from bank cards. Historically, bank card analysis process is achieved manually and, hence, very time consuming. Accurate extraction of key data from bank cards is typically the first and one of the most critical steps in the contract automation process.

## Development options

Document Intelligence v4.0: 2024-11-30 (GA) supports the following tools, applications, and libraries:

Expand table

Feature	Resources	Model ID
Contract model	<ul style="list-style-type: none"><li><a href="#">Document Intelligence Studio</a></li><li><a href="#">REST API</a></li><li><a href="#">C# SDK</a></li><li><a href="#">Python SDK</a></li><li><a href="#">Java SDK</a></li><li><a href="#">JavaScript SDK</a></li></ul>	prebuilt-creditCard

## Input requirements

- Supported file formats:

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

# Try credit card data extraction

To see how data extraction works for the credit/debit card service, you need the following resources:

- An Azure subscription—you can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (`F0`) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

Home > Contoso-DI

**Contoso-DI | Keys and Endpoint**

Search | Regenerate Key1 | Regenerate Key2

Overview | Activity log | Access control (IAM) | Tags | Diagnose and solve problems | Resource Management | Keys and Endpoint | Encryption | Pricing tier | Networking | Identity | Cost analysis | Properties | Locks | Monitoring | Automation | Help

**Show Keys**

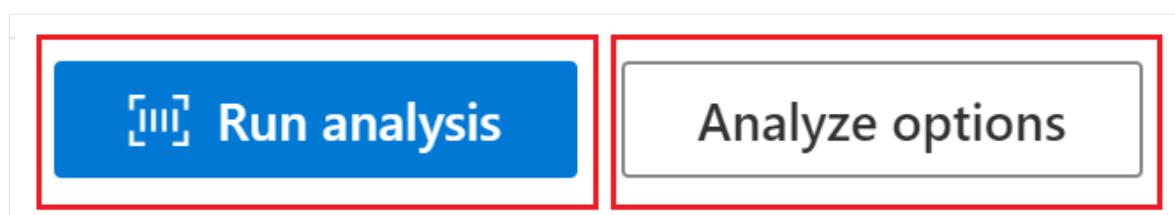
KEY 1  
.....  
KEY 2  
.....

Location/Region westus2

Endpoint https://contoso-di.cognitiveservices.azure.com/

## Document Intelligence Studio

1. On the [Document Intelligence Studio home page](#), select **Credit/Debit Card**.
2. You can analyze the sample credit/debit documents or upload your own files.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options**:



[Try Document Intelligence Studio](#)

# Supported languages and locales

For a complete list of supported languages, see our [prebuilt model language support](#) page.

## Field extraction

- For supported document extraction fields, see the [credit card model schema](#) page in our GitHub sample repository.
- The bank cards key-value pairs and line items extracted are in the `documentResults` section of the JSON output.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | Get help at Microsoft Q&A

# Document Intelligence health insurance card model

Article • 12/11/2024

This content applies to: ✓ v4.0 | Previous versions: ✓ v3.1 (GA) ✓ v3.0 (GA) :: moniker-end

The Document Intelligence health insurance card model combines powerful Optical Character Recognition (OCR) capabilities with deep learning models to analyze and extract key information from US health insurance cards. A health insurance card is a key document for care processing. It can be digitally analyzed for patient onboarding, financial coverage information, cashless payments, and insurance claim processing. The health insurance card model analyzes health card images; extracts key information such as insurer, member, prescription, and group number; and returns a structured JSON representation. Health insurance cards can be presented in various formats and quality including phone-captured images, scanned documents, and digital PDFs.

*Sample health insurance card processed using Document Intelligence Studio*

The screenshot shows the Microsoft Document Intelligence Studio interface. On the left, a sample health insurance card from PREMERA BLUE CROSS is displayed. The card includes fields like Member (ANGEL BROWN), Prefix, Identification #, Suffix, Group # (1000000), Rx Group # (BCAAXY), Rx BIN#, and BCBS 456. On the right, the extracted data is shown in a table:

Fields	Result	Code
GroupNumber	#1 1000000	80.00%
IdNumber	#1 Number 123456789	87.00%
Insurer	#1 PREMERA	87.00%
Member	#1 Name ANGEL BROWN	100.00%
PrescriptionInfo	#1 RxBIN	80.00%

## Development options

Document Intelligence v4.0: 2024-11-30 (GA) supports the following tools, applications, and libraries:

[] Expand table

Feature	Resources	Model ID
Health insurance card model	<ul style="list-style-type: none"> <li>• <a href="#">Document Intelligence Studio</a></li> <li>• <a href="#">REST API</a></li> <li>• <a href="#">C# SDK</a></li> <li>• <a href="#">Python SDK</a></li> <li>• <a href="#">Java SDK</a></li> <li>• <a href="#">JavaScript SDK</a></li> </ul>	prebuilt-healthInsuranceCard.us

## Input requirements

- Supported file formats:

[\[+\] Expand table](#)

Model	PDF	Image: JPEG/JPG, PNG, BMP, TIFF, HEIF	Microsoft Office: Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.

- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

## Try Document Intelligence Studio

See how data is extracted from health insurance cards using the Document Intelligence Studio. You need the following resources:

- An Azure subscription—you can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (F0) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

The screenshot shows the Azure portal interface for managing a Document Intelligence instance named "Contoso-DI". The left sidebar lists several service management categories: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, Keys and Endpoint (which is selected and highlighted with a red box), Encryption, Pricing tier, Networking, Identity, Cost analysis, Properties, Locks, Monitoring, Automation, and Help. The main content area is titled "Contoso-DI | Keys and Endpoint". It contains a note about securely storing keys, a "Show Keys" button, and fields for "KEY 1" and "KEY 2" (both redacted with dots), "Location/Region" set to "westus2", and the "Endpoint" URL "https://contoso-di.cognitiveservices.azure.com/". Buttons for "Regenerate Key1" and "Regenerate Key2" are located above the key fields.

## ⓘ Note

Document Intelligence Studio is available with API version v3.0.

1. On the [Document Intelligence Studio home page](#), select **Health insurance cards**.
2. You can analyze the sample insurance card document or select the  **Add** button to upload your own sample.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options**:



## Supported languages and locales

For a complete list of supported languages, see our [prebuilt model language support](#) page.

## Field extraction

For supported document extraction fields, see the [health insurance card model schema](#) page in our GitHub sample repository.

## Migration guide and REST API v3.1

- Follow our [Document Intelligence v3.1 migration guide](#) to learn how to use the v3.1 version in your applications and workflows.
- Explore our [REST API](#) to learn more about the v3.1 version and new capabilities.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).

- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.
- 

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence ID document model

Article • 02/10/2025

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)  v2.1 (GA)

::: moniker-end

Document Intelligence Identity document (ID) model combines Optical Character Recognition (OCR) with deep learning models to analyze and extract key information from identity documents. The API analyzes identity documents (including the following) and returns a structured JSON data representation.

 Expand table

Region	Document types
Worldwide	Passport Book, Passport Card
United States	Driver License, Identification Card, Residency Permit (Green card), Social Security Card, Military ID
Europe	Driver License, Identification Card, Residency Permit
India	Driver License, PAN Card, Aadhaar Card
Canada	Driver License, Identification Card, Residency Permit (Maple Card)
Australia	Driver License, Photo Card, Key-pass ID (including digital version)

## Identity document processing

Identity document processing involves extracting data from identity documents either manually or by using OCR-based technology. ID document processing is an important step in any business operation that requires proof of identity. Examples include customer verification in banks and other financial institutions, mortgage applications, medical visits, claim processing, hospitality industry, and more. Individuals provide some proof of their identity via driver licenses, passports, and other similar documents so that the business can efficiently verify them before providing services and benefits.

*Sample U.S. Driver's License processed with Document Intelligence Studio* 

Analyze

The image shows a Washington state driver's license. The card is light blue with a photo of a man with dark hair and a purple shirt. The text on the card includes:

- Address: 123 STREET ADDRESS YOUR CITY WA 99999-1234
- CountryRegion: USA
- DateOfBirth: 1958-01-06
- DateOfExpiration: 2020-08-12
- DocumentNumber: WDLABCD456DG
- Endorsements: L
- FirstName: LIAM R.
- LastName: TALBOT
- Locale: en-US

A red arrow points to the "4b EXP 08/12/2020" field.

## Development options

Document Intelligence v4.0: 2024-11-30 (GA) supports the following tools, applications, and libraries:

[Expand table](#)

Feature	Resources	Model ID
ID document model	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript SDK</a></li> </ul>	prebuilt-idDocument

## Input requirements

Supported file formats:

[Expand table](#)

Model	PDF	Image: JPEG/JPG, PNG, BMP, TIFF, HEIF	Microsoft Office: Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML	
Read	✓	✓		✓
Layout	✓	✓		✓
General Document	✓	✓		
Prebuilt	✓	✓		
Custom extraction	✓	✓		
Custom classification	✓	✓		✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

## ID document model data extraction

Extract data, including name, birth date, and expiration date, from ID documents. You need the following resources:

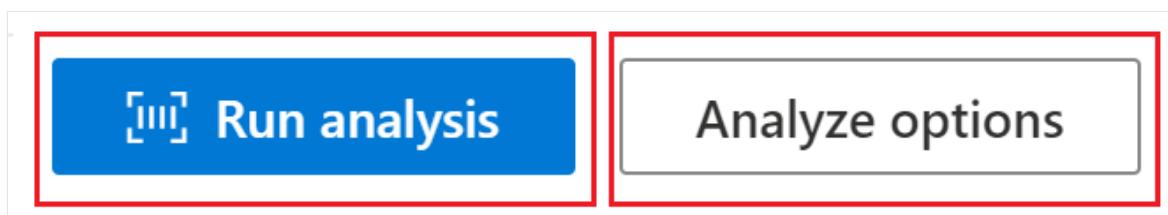
- An Azure subscription—you can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier ( $F0$ ) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

The screenshot shows the 'Keys and Endpoint' page for a Document Intelligence instance named 'Contoso-DI'. The left sidebar shows various management options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, Keys and Endpoint (which is selected and highlighted with a red box), Encryption, Pricing tier, Networking, Identity, Cost analysis, Properties, Locks, Monitoring, Automation, and Help. The main content area displays two key fields ('KEY 1' and 'KEY 2'), a location dropdown set to 'westus2', and an endpoint URL 'https://contoso-di.cognitiveservices.azure.com/'. A note at the top of the main area says: 'These keys are used to access your Azure AI service API. Do not share your keys. Store them securely—for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.'

### ⓘ Note

Document Intelligence Studio is available with v3.1 and v3.0 APIs and later versions.

1. On the [Document Intelligence Studio home page](#), select **Identity documents**.
2. You can analyze the sample invoice or upload your own files.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options**:



[Try Document Intelligence Studio](#)

# Field extractions

For supported document extraction fields, see the [ID document model schema](#) page in our GitHub sample repository.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).
  - Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.
  - Find more samples on [GitHub](#).
- 

## Feedback

Was this page helpful?



[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence invoice model

Article • 12/11/2024

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)   
[v2.1 \(GA\)](#)

::: moniker-end

The Document Intelligence invoice model uses powerful Optical Character Recognition (OCR) capabilities to analyze and extract key fields and line items from sales invoices, utility bills, and purchase orders. Invoices can be of various formats and quality including phone-captured images, scanned documents, and digital PDFs. The API analyzes invoice text; extracts key information such as customer name, billing address, due date, and amount due; and returns a structured JSON data representation. The model currently supports invoices in 27 languages.

**Supported document types:**

- Invoices
- Utility bills
- Sales orders
- Purchase orders

## Automated invoice processing

Automated invoice processing is the process of extracting key `accounts payable` fields from billing account documents. Extracted data includes line items from invoices integrated with your accounts payable (AP) workflows for reviews and payments.

Historically, the accounts payable process is performed manually and, hence, very time consuming. Accurate extraction of key data from invoices is typically the first and one of the most critical steps in the invoice automation process.

Sample invoice processed with [Document Intelligence Studio](#):

Values	Result	Code
AmountDue #1	610	97.30%
BillingAddress #1	123 Bill St. Redmond WA, 98052	94.70%
BillingAddressRecipient #1	Microsoft Finance	95.70%
CustomerAddress #1	123 Other St, Redmond WA, 98052	94.70%
CustomerAddressRecipient #1	Microsoft Corp	95.60%
CustomerId #1	CID-12345	96.40%
CustomerName #1	MICROSOFT CORPORATION	94.90%
DueDate #1	2019-12-15	97.30%
InvoiceDate #1	2019-11-15	97.20%

## Development options

Document Intelligence v4.0: 2024-11-30 (GA) supports the following tools, applications, and libraries:

[Expand table](#)

Feature	Resources	Model ID
Invoice model	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript SDK</a></li> </ul>	prebuilt-invoice

## Input requirements

- Supported file formats:

[Expand table](#)

Model	PDF	Image:	Microsoft Office:
	JPEG/JPG, PNG, BMP, TIFF, HEIF		Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

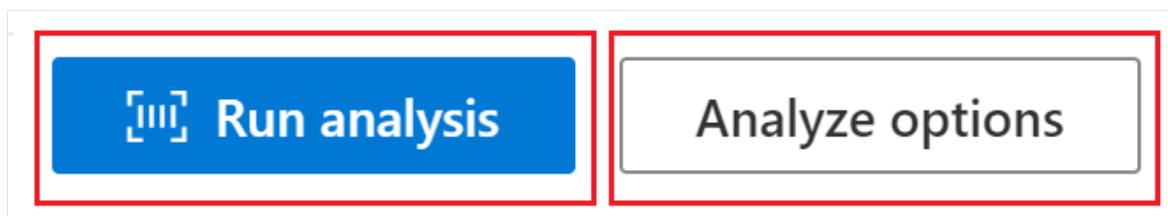
## Invoice model data extraction

See how data, including customer information, vendor details, and line items, is extracted from invoices. You need the following resources:

- An Azure subscription—you can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (F0) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

The screenshot shows the 'Keys and Endpoint' page for the 'Contoso-DI' resource in the Azure portal. The left sidebar lists various management options, and the main area shows the configuration for the 'Keys and Endpoint' section. Key fields shown include 'KEY 1', 'KEY 2', 'Location/Region' set to 'westus2', and the 'Endpoint' URL. The 'Endpoint' URL is specifically highlighted with a red box.

1. On the [Document Intelligence Studio home page](#), select **Invoices**.
2. You can analyze the sample invoice or upload your own files.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options**:



[Try Document Intelligence Studio](#)

## Supported languages and locales

For a complete list of supported languages, see our [prebuilt model language support](#) page.

# Field extraction

- For supported document extraction fields, see the [invoice model schema](#) page in our GitHub sample repository.
- The invoice key-value pairs and line items extracted are in the `documentResults` section of the JSON output.

## Key-value pairs

The prebuilt invoice model supports the optional return of key-value pairs. By default, the return of key-value pairs is disabled. Key-value pairs are specific spans within the invoice that identify a label or key and its associated response or value. In an invoice, these pairs could be the label and the value the user entered for that field or telephone number. The AI model is trained to extract identifiable keys and values based on a wide variety of document types, formats, and structures.

Keys can also exist in isolation when the model detects that a key exists, with no associated value or when processing optional fields. For example, a middle name field can be left blank on a form in some instances. Key-value pairs are always spans of text contained in the document. For documents where the same value is described in different ways, for example, customer/user, the associated key is either customer or user (based on context).

## JSON output

The JSON output has three parts:

- `"readResults"` node contains all of the recognized text and selection marks. Text is organized via page, then by line, then by individual words.
- `"pageResults"` node contains the tables and cells extracted with their bounding boxes, confidence, and a reference to the lines and words in `readResults`.
- `"documentResults"` node contains the invoice-specific values and line items that the model discovered. It's where to find all the fields from the invoice such as invoice ID, ship to, bill to, customer, total, line items and lots more.

## Migration guide

- Follow our [Document Intelligence v3.1 migration guide](#) to learn how to use the v3.0 version in your applications and workflows.

::: moniker-end

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).
  - Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.
  - Find more samples on [GitHub](#).
- 

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# What is Document Intelligence layout model?

Article • 05/05/2025

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)  v2.1 (GA)

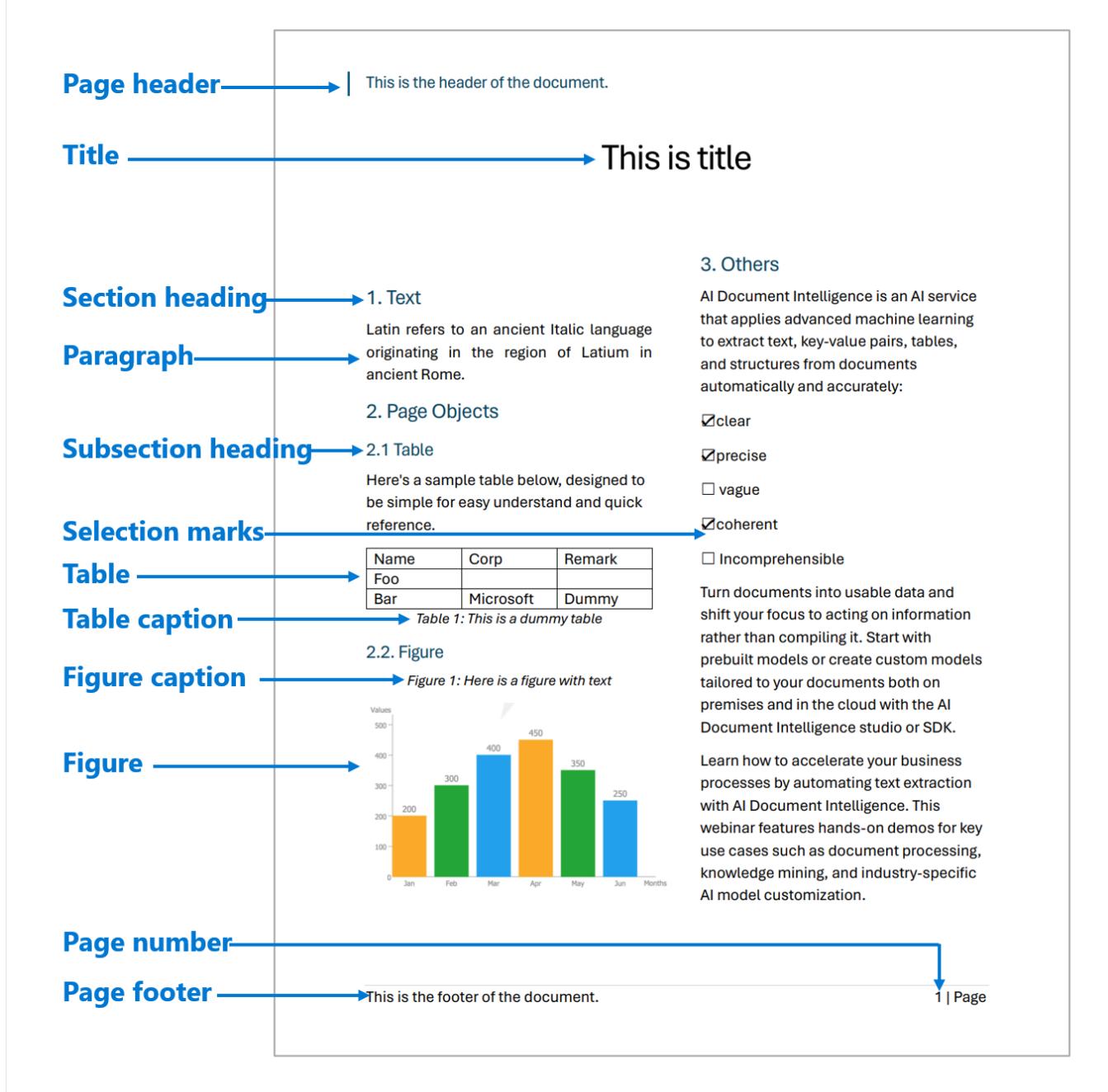
Document Intelligence layout model is an advanced machine-learning based document analysis API available in the Document Intelligence cloud. It enables you to take documents in various formats and return structured data representations of the documents. It combines an enhanced version of our powerful [Optical Character Recognition \(OCR\)](#) capabilities with deep learning models to extract text, tables, selection marks, and document structure.

## Document structure layout analysis

Document structure layout analysis is the process of analyzing a document to extract regions of interest and their inter-relationships. The goal is to extract text and structural elements from the page to build better semantic understanding models. There are two types of roles in a document layout:

- **Geometric roles:** Text, tables, figures, and selection marks are examples of geometric roles.
- **Logical roles:** Titles, headings, and footers are examples of logical roles of texts.

The following illustration shows the typical components in an image of a sample page.



## Development options

Document Intelligence v4.0: 2024-11-30 (GA) supports the following tools, applications, and libraries:

[\[ \] Expand table](#)

Feature	Resources	Model ID
Layout model	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> </ul>	prebuilt-layout

Feature	Resources	Model ID
	<ul style="list-style-type: none"><li>Java SDK</li><li>JavaScript SDK</li></ul>	

## Supported languages

See [Language Support—document analysis models](#) for a complete list of supported languages.

## Supported file types

Document Intelligence v4.0: 2024-11-30 (GA) layout model supports the following file formats:

[\[+\] Expand table](#)

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Layout	✓	✓	✓

## Input requirements

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- If your PDFs are password-locked, you must remove the lock before submission.
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.

- For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
- For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

For more information on model usage, quotas, and service limits, see [service limits](#).

## Get started with Layout model

See how data, including text, tables, table headers, selection marks, and structure information is extracted from documents using Document Intelligence. You need the following resources:

- An Azure subscription—you can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (F0) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

The screenshot shows the Azure portal interface for a Document Intelligence instance named "Contoso-DI". The left sidebar has a tree view with "Resource Management" expanded, showing "Keys and Endpoint" selected. The main content area is titled "Contoso-DI | Keys and Endpoint". It contains a "Show Keys" button and two sections for "KEY 1" and "KEY 2". Each key section has a text input field and a copy icon. Below the keys is a "Location/Region" input field set to "westus2". At the bottom is an "Endpoint" input field containing the URL "https://contoso-di.cognitiveservices.azure.com/". A large red box highlights the entire "Show Keys" section, which includes both key sections and the location input.

After you retrieve your key and endpoint, use the following development options to build and deploy your Document Intelligence applications:

[REST API](#)

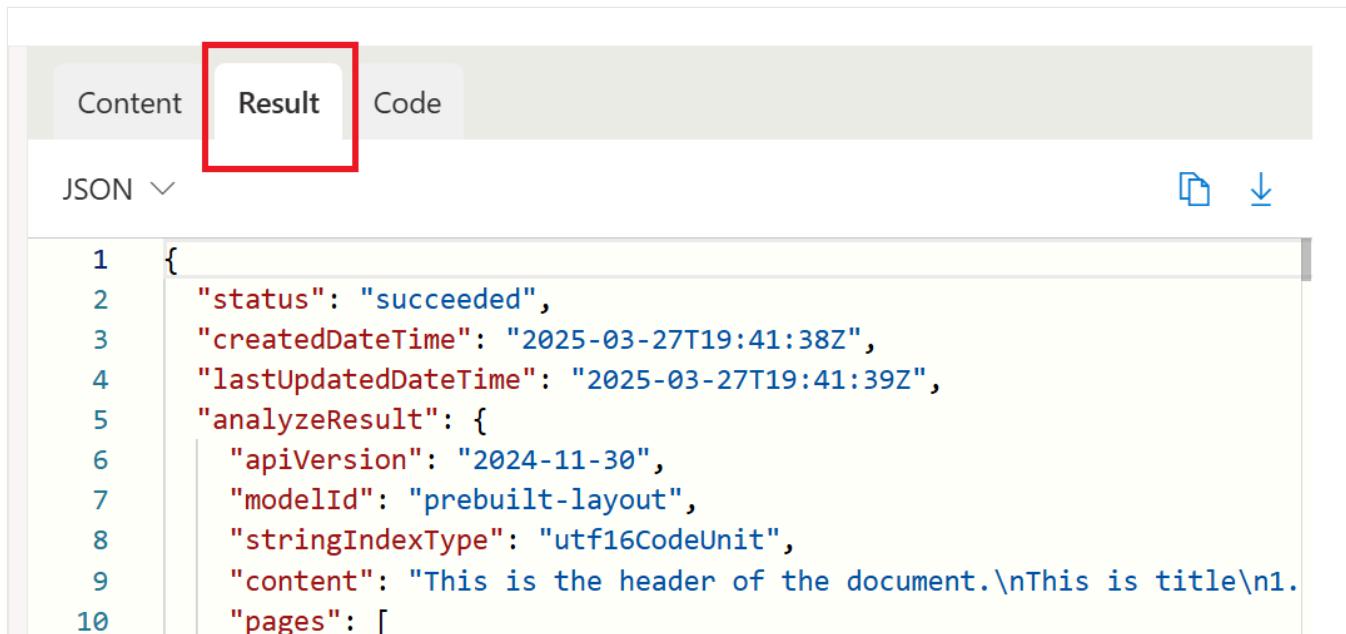
- Document Intelligence REST API
- How to guide

## Data extraction

The layout model extracts structural elements from your documents. To follow are descriptions of these structural elements with guidance on how to extract them from your document input:

- [Pages](#)
- [Paragraphs](#)
- [Text, lines, and words](#)
- [Selection marks](#)
- [Tables](#)
- [Output response to markdown](#)
- [Figures](#)
- [Sections](#)

Run the sample layout document analysis within [Document Intelligence Studio](#), then navigate to the results tab and access the full JSON output.



The screenshot shows the Document Intelligence Studio interface. At the top, there are three tabs: 'Content', 'Result' (which is highlighted with a red box), and 'Code'. Below the tabs, there is a dropdown menu labeled 'JSON' with a downward arrow, and two icons: a clipboard and a download arrow. The main area contains a code editor with numbered lines (1-10) displaying a JSON object. The JSON structure includes fields like 'status', 'createdDateTime', 'lastUpdatedDateTime', 'analyzeResult', 'apiVersion', 'modelId', 'stringIndexType', 'content', and 'pages'.

```
1 {  
2   "status": "succeeded",  
3   "createdDateTime": "2025-03-27T19:41:38Z",  
4   "lastUpdatedDateTime": "2025-03-27T19:41:39Z",  
5   "analyzeResult": {  
6     "apiVersion": "2024-11-30",  
7     "modelId": "prebuilt-layout",  
8     "stringIndexType": "utf16CodeUnit",  
9     "content": "This is the header of the document.\nThis is title\n1.",  
10    "pages": [  
11      {"id": 1, "order": 1, "type": "Text", "text": "This is the header of the document.", "x": 100, "y": 100, "width": 300, "height": 50, "angle": 0},  
12      {"id": 2, "order": 2, "type": "Text", "text": "This is title", "x": 100, "y": 150, "width": 200, "height": 50, "angle": 0},  
13      {"id": 3, "order": 3, "type": "Text", "text": "1.", "x": 100, "y": 200, "width": 50, "height": 50, "angle": 0}  
14    ]  
15  }  
16}
```

## Pages

The pages collection is a list of pages within the document. Each page is represented sequentially within the document and includes the orientation angle indicating if the page is rotated and the width and height (dimensions in pixels). The page units in the model output are computed as shown:

File format	Computed page unit	Total pages
Images (JPEG/JPG, PNG, BMP, HEIF)	Each image = 1 page unit	Total images
PDF	Each page in the PDF = 1 page unit	Total pages in the PDF
TIFF	Each image in the TIFF = 1 page unit	Total images in the TIFF
Word (DOCX)	Up to 3,000 characters = 1 page unit, embedded or linked images not supported	Total pages of up to 3,000 characters each
Excel (XLSX)	Each worksheet = 1 page unit, embedded or linked images not supported	Total worksheets
PowerPoint (PPTX)	Each slide = 1 page unit, embedded or linked images not supported	Total slides
HTML	Up to 3,000 characters = 1 page unit, embedded or linked images not supported	Total pages of up to 3,000 characters each

### Sample code

#### Python

```
# Analyze pages.
for page in result.pages:
    print(f"----Analyzing layout from page #{page.page_number}----")
    print(f"Page has width: {page.width} and height: {page.height}, measured with
unit: {page.unit}")
```

[View samples on GitHub.](#)

## Extract selected pages

For large multi-page documents, use the `pages` query parameter to indicate specific page numbers or page ranges for text extraction.

## Paragraphs

The Layout model extracts all identified blocks of text in the `paragraphs` collection as a top level object under `analyzeResults`. Each entry in this collection represents a text block and

../includes the extracted text as `content` and the bounding `polygon` coordinates. The `span` information points to the text fragment within the top level `content` property that contains the full text from the document.

JSON

```
"paragraphs": [
  {
    "spans": [],
    "boundingRegions": [],
    "content": "While healthcare is still in the early stages of its AI
journey, we are seeing pharmaceutical and other life sciences organizations making
major investments in AI and related technologies.\\" TOM LAWRY | National Director
for AI, Health and Life Sciences | Microsoft"
  }
]
```

## Paragraph roles

The new machine-learning based page object detection extracts logical roles like titles, section headings, page headers, page footers, and more. The Document Intelligence Layout model assigns certain text blocks in the `paragraphs` collection with their specialized role or type predicted by the model. It's best to use paragraph roles with unstructured documents to help understand the layout of the extracted content for a richer semantic analysis. The following paragraph roles are supported:

[ ] Expand table

Predicted role	Description	Supported file types
<code>title</code>	The main headings in the page	pdf, image, docx, pptx, xlsx, html
<code>sectionHeading</code>	One or more subheadings on the page	pdf, image, docx, xlsx, html
<code>footnote</code>	Text near the bottom of the page	pdf, image
<code>pageHeader</code>	Text near the top edge of the page	pdf, image, docx
<code>pageFooter</code>	Text near the bottom edge of the page	pdf, image, docx, pptx, html
<code>pageNumber</code>	Page number	pdf, image

JSON

```
{  
  "paragraphs": [
```

```
        },
        "spans": [],
        "boundingRegions": [],
        "role": "title",
        "content": "NEWS TODAY"
    },
    {
        "spans": [],
        "boundingRegions": [],
        "role": "sectionHeading",
        "content": "Mirjam Nilsson"
    }
]
```

## Text, lines, and words

The document layout model in Document Intelligence extracts print and handwritten style text as `lines` and `words`. The `styles` collection includes any handwritten style for lines if detected along with the spans pointing to the associated text. This feature applies to [supported handwritten languages](#).

For Microsoft Word, Excel, PowerPoint, and HTML, Document Intelligence v4.0 [2024-11-30](#) (GA) Layout model extract all embedded text as is. Texts are extracted as words and paragraphs. Embedded images aren't supported.

Sample code

Python

```
# Analyze lines.
if page.lines:
    for line_idx, line in enumerate(page.lines):
        words = get_words(page, line)
        print(
            f"...Line # {line_idx} has word count {len(words)} and text
'{line.content}'"
            f"within bounding polygon '{line.polygon}'"
        )

# Analyze words.
for word in words:
    print(f".....Word '{word.content}' has a confidence of
{word.confidence}")
```

[View samples on GitHub.](#)

## Handwritten style for text lines

The response `./includes` classifying whether each text line is of handwriting style or not, along with a confidence score. For more information. See [Handwritten language support](#). The following example shows an example JSON snippet.

JSON

```
"styles": [
{
  "confidence": 0.95,
  "spans": [
    {
      "offset": 509,
      "length": 24
    }
  "isHandwritten": true
  ]
}
```

If you enable the [font/style addon capability](#), you also get the font/style result as part of the `styles` object.

## Selection marks

The Layout model also extracts selection marks from documents. Extracted selection marks appear within the `pages` collection for each page. They include the bounding `polygon`, `confidence`, and selection `state` (`selected/unselected`). The text representation (that is, `:selected:` and `:unselected`) is also included as the starting index (`offset`) and `length` that references the top level `content` property that contains the full text from the document.

Sample code

Python

```
# Analyze selection marks.
if page.selection_marks:
    for selection_mark in page.selection_marks:
        print(
            f"Selection mark is '{selection_mark.state}' within bounding
            polygon "
            f"'{selection_mark.polygon}' and has a confidence of
```

```
{selection_mark.confidence}"  
    )
```

[View samples on GitHub.](#)

## Tables

Extracting tables is a key requirement for processing documents containing large volumes of data typically formatted as tables. The Layout model extracts tables in the `pageResults` section of the JSON output. Extracted table information .. /includes the number of columns and rows, row span, and column span. Each cell with its bounding polygon is output along with information whether the area is recognized as a `columnHeader` or not. The model supports extracting tables that are rotated. Each table cell contains the row and column index and bounding polygon coordinates. For the cell text, the model outputs the `span` information containing the starting index (`offset`). The model also outputs the `length` within the top-level content that contains the full text from the document.

Here are a few factors to consider when using the Document Intelligence table extraction capability:

- Is the data that you want to extract presented as a table, and is the table structure meaningful?
- Can the data fit in a two-dimensional grid if the data isn't in a table format?
- Do your tables span multiple pages? If so, to avoid having to label all the pages, split the PDF into pages before sending it to Document Intelligence. After the analysis, post-process the pages to a single table.
- Refer to [Tabular fields](#) if you're creating custom models. Dynamic tables have a variable number of rows for each column. Fixed tables have a constant number of rows for each column.

### Note

- Table analysis isn't supported if the input file is XLSX.
- For [2024-11-30 \(GA\)](#), the bounding regions for figures and tables cover only the core content and exclude associated caption and footnotes.

Sample code

Python

```
if result.tables:
    for table_idx, table in enumerate(result.tables):
        print(f"Table # {table_idx} has {table.row_count} rows and " f"
{table.column_count} columns")
        if table.bounding_regions:
            for region in table.bounding_regions:
                print(f"Table # {table_idx} location on page:
{region.page_number} is {region.polygon}")
                # Analyze cells.
                for cell in table.cells:
                    print(f"...Cell[{cell.row_index}][{cell.column_index}] has text
'{cell.content}'")
                    if cell.bounding_regions:
                        for region in cell.bounding_regions:
                            print(f"...content on page {region.page_number} is within
bounding polygon '{region.polygon}'")
```

[View samples on GitHub.](#)

## Output response to markdown format

The Layout API can output the extracted text in markdown format. Use the `outputContentFormat=markdown` to specify the output format in markdown. The markdown content is output as part of the `content` section.

### ⓘ Note

For v4.0 2024-11-30 (GA), the representation of tables is changed to HTML tables to enable rendering of merged cells, multi-row headers, etc. Another related change is to use Unicode checkbox characters ✕ and ☐ for selection marks instead of `:selected:` and `:unselected:`. This update means that the content of selection mark fields contains `:selected:` even though their spans refer to Unicode characters in the top-level span. Refer to the [Markdown Output Format](#) for full definition of Markdown elements.

Sample code

Python

```
document_intelligence_client = DocumentIntelligenceClient(endpoint=endpoint,
credential=AzureKeyCredential(key))
poller = document_intelligence_client.begin_analyze_document(
    "prebuilt-layout",
    AnalyzeDocumentRequest(url_source=url),
    output_content_format=ContentFormat.MARKDOWN,
)
```

[View samples on GitHub.](#)

## Figures

Figures (charts, images) in documents play a crucial role in complementing and enhancing the textual content, providing visual representations that aid in the understanding of complex information. The figures object detected by the Layout model has key properties like

`boundingRegions` (the spatial locations of the figure on the document pages, including the page number and the polygon coordinates that outline the figure's boundary), `spans` (details the text spans related to the figure, specifying their offsets and lengths within the document's text. This connection helps in associating the figure with its relevant textual context), `elements` (the identifiers for text elements or paragraphs within the document that are related to or describe the figure) and `caption` if there's any.

When `output=figures` is specified during the initial analyze operation, the service generates cropped images for all detected figures that can be accessed via `/analyzeResults/{resultId}/figures/{figureId}`. `FigureId` is included in each figure object, following an undocumented convention of `{pageNumber}.{figureIndex}` where `figureIndex` resets to one per page.

### ⓘ Note

For v4.0 2024-11-30 (GA), the bounding regions for figures and tables cover only the core content and exclude associated caption and footnotes.

Sample code

Python

```
# Analyze figures.
if result.figures:
    for figures_idx,figures in enumerate(result.figures):
```

```
print(f"Figure # {figures_idx} has the following spans:  
{figures.spans}")  
    for region in figures.bounding_regions:  
        print(f"Figure # {figures_idx} location on page:  
{region.page_number} is within bounding polygon '{region.polygon}'")
```

[View samples on GitHub.](#)

## Sections

Hierarchical document structure analysis is pivotal in organizing, comprehending, and processing extensive documents. This approach is vital for semantically segmenting long documents to boost comprehension, facilitate navigation, and improve information retrieval. The advent of [retrieval-augmented generation \(RAG\)](#) in document generative AI underscores the significance of hierarchical document structure analysis. The Layout model supports sections and subsections in the output, which identifies the relationship of sections and object within each section. The hierarchical structure is maintained in `elements` of each section. You can use [output response to markdown format](#) to easily get the sections and subsections in markdown.

Sample code

Python

```
document_intelligence_client = DocumentIntelligenceClient(endpoint=endpoint,  
credential=AzureKeyCredential(key))  
poller = document_intelligence_client.begin_analyze_document(  
    "prebuilt-layout",  
    AnalyzeDocumentRequest(url_source=url),  
    output_content_format=ContentFormat.MARKDOWN,  
)
```

[View samples on GitHub.](#)

## Next steps

- Learn how to process your own forms and documents with the [Document Intelligence Studio](#).

- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.
- [Find more samples on GitHub.](#) ↗

# Document Intelligence marriage certificate model

Article • 12/11/2024

This content applies to:  v4.0 (GA)

The Document Intelligence Marriage Certificate model uses powerful Optical Character Recognition (OCR) capabilities to analyze and extract key fields from Marriage Certificates. Marriage certificates can be of various formats and quality including phone-captured images, scanned documents, and digital PDFs. The API analyzes document text; extracts key information such as Spouse names, Issue date, and marriage place; and returns a structured JSON data representation. The model currently supports English-language document formats.

## Automated marriage certificate processing

Automated marriage certificate processing is the process of extracting key fields from Marriage certificates. Historically, the marriage certificate analysis process is achieved manually and, hence, very time consuming. Accurate extraction of key data from marriage certificates is typically the first and one of the most critical steps in the marriage certificate automation process.

## Development options

Document Intelligence v4.0: 2024-11-30 (GA) supports the following tools, applications, and libraries:

 Expand table

Feature	Resources	Model ID
prebuilt-marriageCertificate.us	<ul style="list-style-type: none"><li><a href="#">Document Intelligence Studio</a></li><li><a href="#">REST API</a></li><li><a href="#">C# SDK</a></li><li><a href="#">Python SDK</a></li><li><a href="#">Java SDK</a></li><li><a href="#">JavaScript SDK</a></li></ul>	prebuilt-marriageCertificate.us

# Input requirements

- Supported file formats:

[Expand table](#)

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

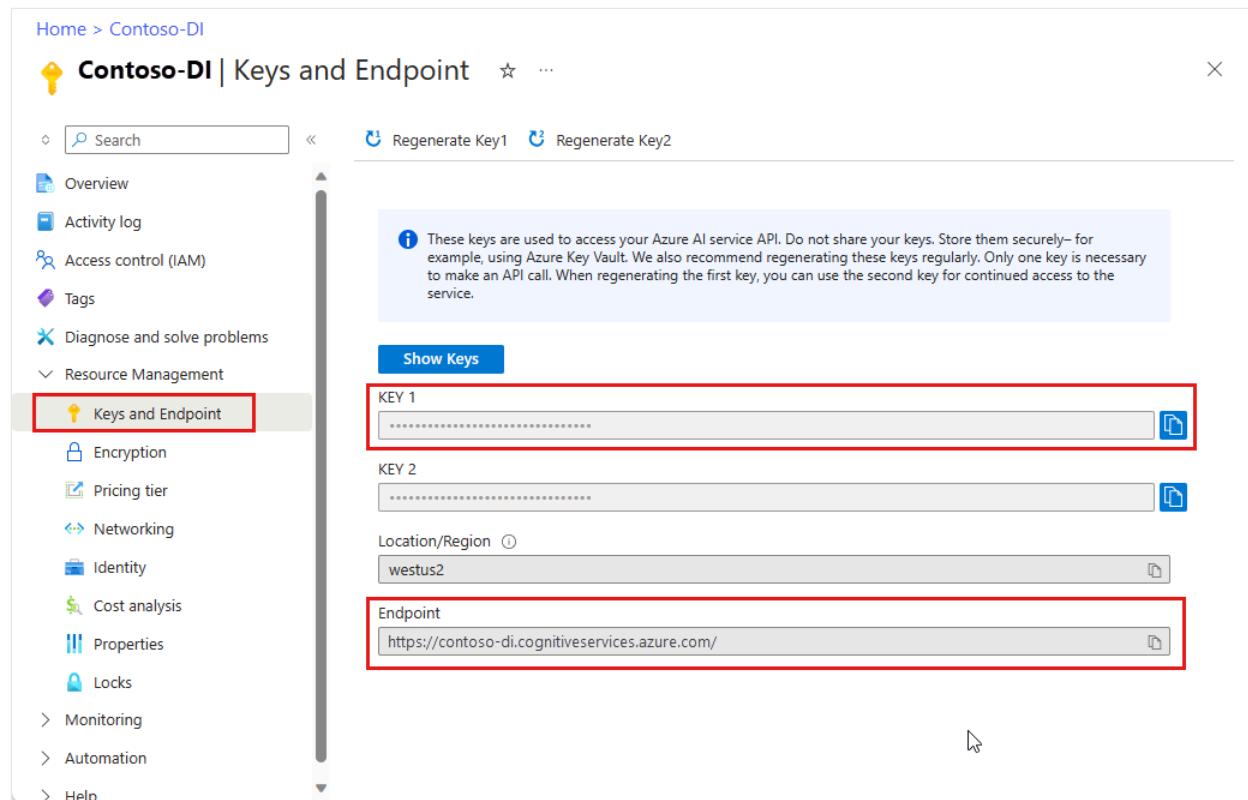
- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.

- For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

## Try marriage certificate document data extraction

To see how data extraction works for the marriage certificate card service, you need the following resources:

- An Azure subscription—you can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (F0) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.



## Document Intelligence Studio

- On the [Document Intelligence Studio home page](#), select **Marriage Certificate**.
- You can analyze the sample Marriage certificates or upload your own files.
- Select the **Run analysis** button and, if necessary, configure the **Analyze options**:

 Run analysis

Analyze options

[Try Document Intelligence Studio](#)

## Supported languages and locales

For a complete list of supported languages, see our [prebuilt model language support](#) page.

## Field extraction

- For supported document extraction fields, see the [marriage certificate model schema](#) page in our GitHub sample repository.
- The marriage certificate key-value pairs and line items extracted are in the `documentResults` section of the JSON output.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence mortgage document models

Article • 12/11/2024

This content applies to:  v4.0 (GA)

The Document Intelligence Mortgage models use powerful Optical Character Recognition (OCR) capabilities and deep learning models to analyze and extract key fields from mortgage documents. Mortgage documents can be of various formats and quality. The API analyzes mortgage documents and returns a structured JSON data representation. The models currently support English-language documents only. With the latest V4.0, you can now extract signatures from mortgage applications and forms.

**Supported document types:**

- Uniform Residential Loan Application (Form 1003)
- Uniform Residential Appraisal Report (Form 1004)
- Verification of employment form (Form 1005)
- Uniform Underwriting and Transmittal Summary (Form 1008)
- Closing Disclosure form

## Development options

Document Intelligence v4.0 (2024-11-30-GA) supports the following tools, applications, and libraries:

 Expand table

Feature	Resources	Model ID
Mortgage model	<ul style="list-style-type: none"><li>• <a href="#">Document Intelligence Studio</a></li><li>• <a href="#">REST API</a></li><li>• <a href="#">C# SDK</a></li><li>• <a href="#">Python SDK</a></li><li>• <a href="#">Java SDK</a></li><li>• <a href="#">JavaScript SDK</a></li></ul>	<ul style="list-style-type: none"><li>• prebuilt-mortgage.us.1003</li><li>• prebuilt-mortgage.us.1004</li><li>• prebuilt-mortgage.us.1005</li><li>• prebuilt-mortgage.us.1008</li><li>• prebuilt-mortgage.us.closingDisclosure</li></ul>

## Input requirements

- Supported file formats:

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

# Try mortgage documents data extraction

To see how data extraction works for the mortgage documents service, you need the following resources:

- An Azure subscription—you can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (`F0`) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

The screenshot shows the Azure portal interface for a resource named 'Contoso-DI'. The left sidebar shows various service categories like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, and Keys and Endpoint (which is selected and highlighted with a red box). The main content area displays the 'Keys and Endpoint' settings. It includes a note about securely storing keys, a 'Show Keys' button, and fields for KEY 1, KEY 2, Location/Region (set to westus2), and Endpoint (set to https://contoso-di.cognitiveservices.azure.com/). The 'Endpoint' field and the 'Show Keys' button are also highlighted with red boxes.

## Document Intelligence Studio

1. On the [Document Intelligence Studio home page](#), select **Mortgage**.
2. You can analyze the sample mortgage documents or upload your own files.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options**:



[Try Document Intelligence Studio](#)

# Supported languages and locales

See our [Language Support—prebuilt models](#) page for a complete list of supported languages.

## Field extraction

For supported document extraction fields, see the [mortgage document model schema](#) pages in our GitHub sample repository.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

---

## Feedback

Was this page helpful?



[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence payStub model

Article • 12/11/2024

The Document Intelligence payStub model combines powerful Optical Character Recognition (OCR) capabilities with deep learning models to analyze and extract compensation and earnings data from pay slips. The API analyzes documents and files with payroll related information; extracts key information and returns a structured JSON data representation.

[ ] Expand table

Feature	version	Model ID
payStub model	v4.0: <a href="#">2024-11-30</a> (GA)	<a href="#">prebuilt-payStub.us</a>

## Try payStub data extraction

Pay stubs are essential documents issued by employers to employees, providing earnings, deductions, and net pay information for a specific pay period. See how data is extracted using [prebuilt-payStub.us](#) model. You need the following resources:

- An Azure subscription—you can [create one for free](#)
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (`F0`) to try the service. After your resource deploys, select [Go to resource](#) to get your key and endpoint.

The screenshot shows the Azure portal interface for a resource named 'Contoso-DI'. The left sidebar lists various management options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, Keys and Endpoint (which is selected and highlighted with a red box), Encryption, Pricing tier, Networking, Identity, Cost analysis, Properties, Locks, Monitoring, Automation, and Help. The main content area displays information about keys and endpoints. A callout box provides instructions on handling keys securely. Below it, there are fields for 'KEY 1' (containing a redacted key value) and 'Endpoint' (containing the URL 'https://contoso-di.cognitiveservices.azure.com/'), both of which are also highlighted with red boxes.

# Document Intelligence Studio

1. On the [Document Intelligence Studio home page](#), select **payStub**.
2. You can analyze the sample pay stub or upload your own files.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options**:

## Input requirements

- Supported file formats:

[\[ \]](#) Expand table

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

## Supported languages and locales

For a complete list of supported languages, see our [prebuilt model language support](#) page.

## Field extractions

For supported document extraction fields, see the [payStub model schema](#) page in our GitHub sample repository.

# Supported locales

The prebuilt-payStub.us version supports the `en-us` locale.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#)
  - Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.
- 

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence read model

Article • 02/19/2025

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)

## Note

To extract text from external images like labels, street signs, and posters, use the [Azure AI Image Analysis v4.0 Read](#) feature optimized for general (not document) images with a performance-enhanced synchronous API. This capability makes it easier to embed OCR in real-time user experience scenarios.

Document Intelligence Read Optical Character Recognition (OCR) model runs at a higher resolution than Azure AI Vision Read and extracts print and handwritten text from PDF documents and scanned images. It also includes support for extracting text from Microsoft Word, Excel, PowerPoint, and HTML documents. It detects paragraphs, text lines, words, locations, and languages. The Read model is the underlying OCR engine for other Document Intelligence prebuilt models like Layout, General Document, Invoice, Receipt, Identity (ID) document, Health insurance card, W2 in addition to custom models.

## What is Optical Character Recognition?

Optical Character Recognition (OCR) for documents is optimized for large text-heavy documents in multiple file formats and global languages. It includes features like higher-resolution scanning of document images for better handling of smaller and dense text; paragraph detection; and fillable form management. OCR capabilities also include advanced scenarios like single character boxes and accurate extraction of key fields commonly found in invoices, receipts, and other prebuilt scenarios.

## Development options (v4)

Document Intelligence v4.0: 2024-11-30 (GA) supports the following tools, applications, and libraries:

 Expand table

Feature	Resources	Model ID
Read OCR model	<ul style="list-style-type: none"> <li>• <a href="#">Document Intelligence Studio</a></li> <li>• <a href="#">REST API</a></li> <li>• <a href="#">C# SDK</a></li> <li>• <a href="#">Python SDK</a></li> <li>• <a href="#">Java SDK</a></li> <li>• <a href="#">JavaScript SDK</a></li> </ul>	prebuilt-read

## Input requirements (v4)

Supported file formats:

[\[ \] Expand table](#)

Model	PDF	Image: JPEG/JPG, PNG, BMP, TIFF, HEIF	Microsoft Office: Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch

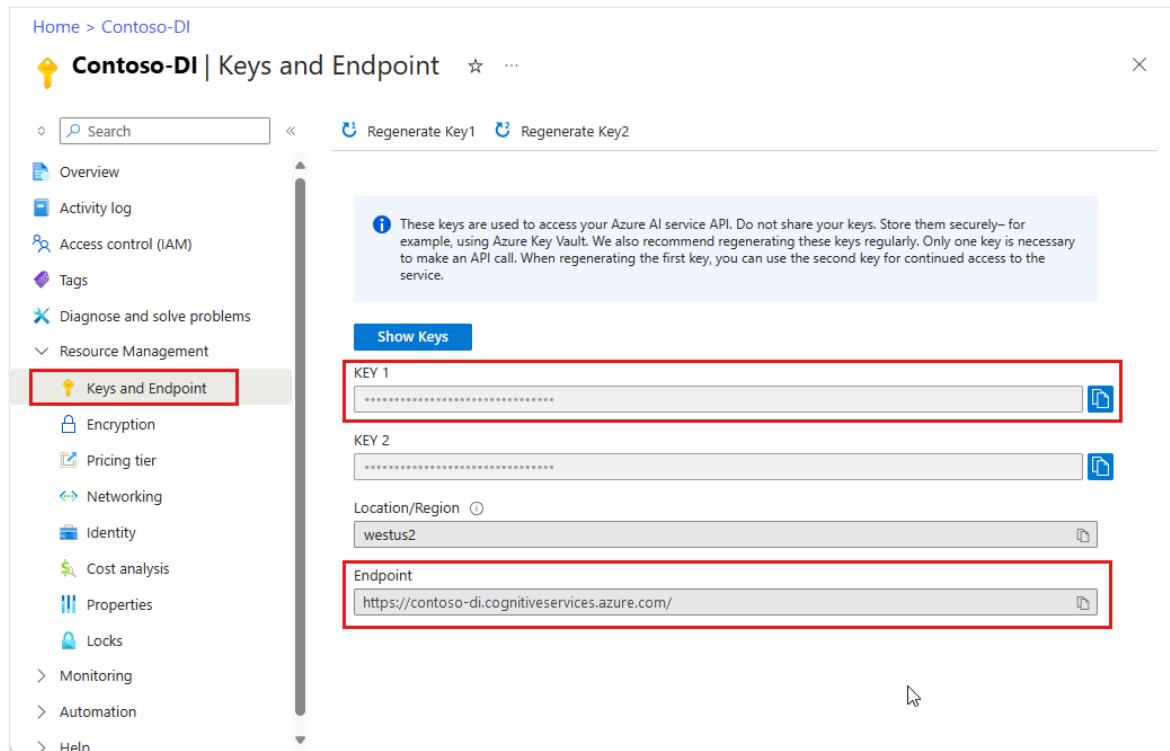
(DPI).

- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

## Get started with Read model (v4)

Try extracting text from forms and documents using the Document Intelligence Studio. You need the following assets:

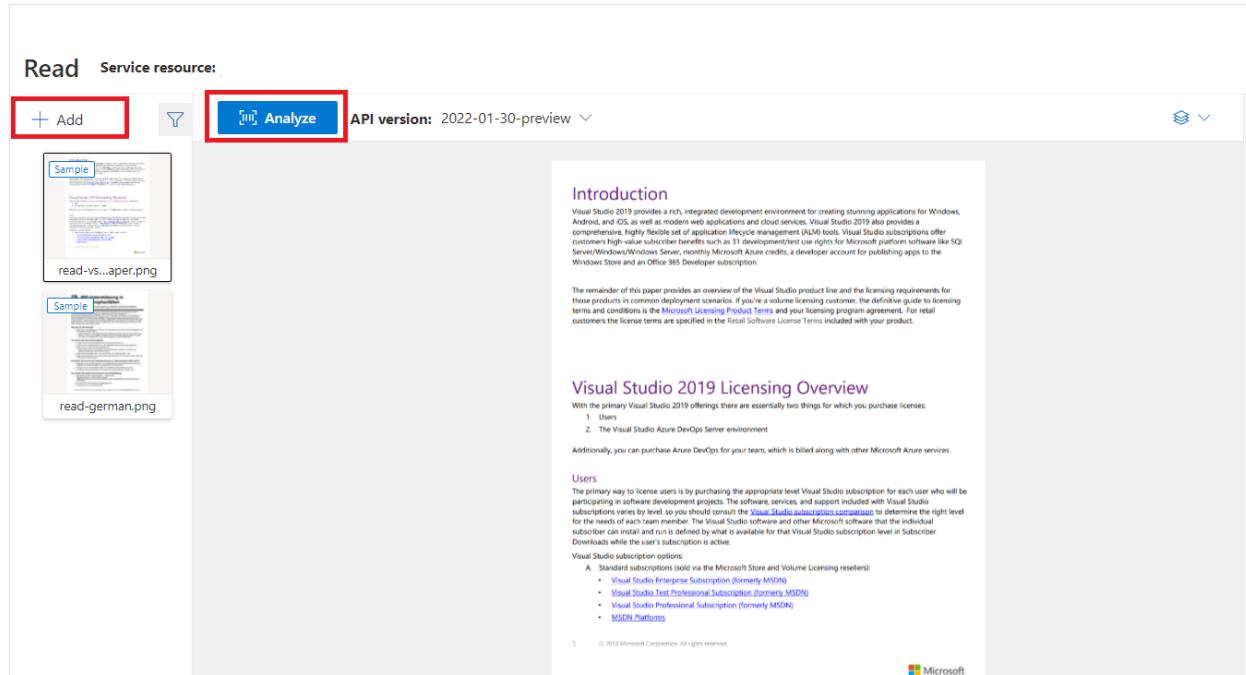
- An Azure subscription—you can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (F0) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.



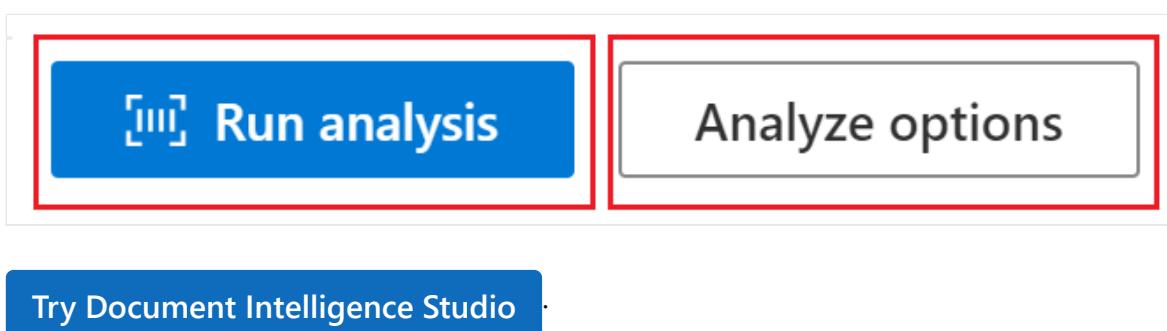
### ① Note

Currently, Document Intelligence Studio doesn't support Microsoft Word, Excel, PowerPoint, and HTML file formats.

### Sample document processed with [Document Intelligence Studio](#)



1. On the [Document Intelligence Studio home page](#), select Read.
2. You can analyze the sample document or upload your own files.
3. Select the Run analysis button and, if necessary, configure the Analyze options:



[Try Document Intelligence Studio](#)

## Supported languages and locales (v4)

See our [Language Support—document analysis models](#) page for a complete list of supported languages.

## Data extraction (v4)

### Note

Microsoft Word and HTML file are supported in v4.0. The following capabilities are currently not supported:

- No angle, width/height, and unit returned with each page object.
- No bounding polygon or bounding region for each object detected.
- No page range (`pages`) as a parameter returned.
- No `lines` object.

## Searchable PDFs

The searchable PDF capability enables you to convert an analog PDF, such as scanned-image PDF files, to a PDF with embedded text. The embedded text enables deep text search within the PDF's extracted content by overlaying the detected text entities on top of the image files.

### i Important

- Currently, only the Read OCR model `prebuilt-read` supports the searchable PDF capability. When using this feature, specify the `modelId` as `prebuilt-read`. Other model types return an error for this preview version.
- Searchable PDF is included with the `2024-11-30` GA `prebuilt-read` model with no added cost for generating a searchable PDF output.

## Use searchable PDFs

To use searchable PDF, make a `POST` request using the `Analyze` operation and specify the output format as `pdf`:

Bash

```
POST {endpoint}/documentintelligence/documentModels/prebuilt-
read:analyze?_overload=analyzeDocument&api-version=2024-11-30&output=pdf
{...}
202
```

Poll for completion of the `Analyze` operation. Once the operation is complete, issue a `GET` request to retrieve the PDF format of the `Analyze` operation results.

Upon successful completion, the PDF can be retrieved and downloaded as `application/pdf`. This operation allows direct downloading of the embedded text form of PDF instead of Base64-encoded JSON.

Bash

```
// Monitor the operation until completion.  
GET /documentModels/prebuilt-read/analyzeResults/{resultId}  
200  
{...}
```

```
// Upon successful completion, retrieve the PDF as application/pdf.  
GET {endpoint}/documentIntelligence/documentModels/prebuilt-  
read/analyzeResults/{resultId}/pdf?api-version=2024-11-30
```

URI Parameters

Name	In	Required	Type	Description
endpoint	path	True	string	

uri

The Document Intelligence service endpoint.

modelId	path	True
	string	

Unique document model name.

Regex pattern: `^[a-zA-Z0-9][a-zA-Z0-9._~-]{1,63}$`

resultId	path	True
	string	

uuid

Analyze operation result ID.

api-version	query	True
	string	

The API version to use [for](#) this operation.

Responses

Name	Type	Description
200	OK	
	file	

The request has succeeded.

Media Types: `"application/pdf"`, `"application/json"`

Other Status Codes

`DocumentIntelligenceErrorResponse`

An unexpected error response.

Media Types: "application/pdf", "application/json"

Security

Ocp-Apim-Subscription-Key

Type: apiKey

In: header

OAuth2Auth

Type: oauth2

Flow: accessCode

Authorization URL: <https://login.microsoftonline.com/common/oauth2/authorize>

Token URL: <https://login.microsoftonline.com/common/oauth2/token>

Scopes

Name Description

<https://cognitiveservices.azure.com/.default>

Examples

Get Analyze Document Result PDF

Sample request

HTTP

HTTP

Copy

GET

<https://myendpoint.cognitiveservices.azure.com/documentintelligence/documentModels/prebuilt-invoice/analyzeResults/3b31320d-8bab-4f88-b19c-2322a7f11034/pdf?api-version=2024-11-30>

Sample response

Status code:

200

JSON

Copy

"{pdfBinary}"

Definitions

Name Description

DocumentIntelligenceError

The error object.

DocumentIntelligenceErrorResponse

Error response object.

DocumentIntelligenceInnerError

An object containing more specific information about the error.

DocumentIntelligenceError

The error object.

Name Type Description

code

string

One of a server-defined [set](#) of error codes.

**details**  
DocumentIntelligenceError[]

An array of details about specific errors that led to this reported error.

**innererror**  
DocumentIntelligenceInnerError

An object containing more specific information than the current object about the error.

**message**  
string

A human-readable representation of the error.

**target**  
string

The target of the error.

DocumentIntelligenceErrorResponse  
Error response object.

Name	Type	Description
error	DocumentIntelligenceError	

Error info.

DocumentIntelligenceInnerError  
An object containing more specific information about the error.

Name	Type	Description
code	string	

One of a server-defined [set](#) of error codes.

**innererror**  
DocumentIntelligenceInnerError

Inner error.

**message**  
string

A human-readable representation of the error.

In this article  
URI Parameters  
Responses  
Security  
Examples

200 OK

Content-Type: application/pdf

## Pages parameter

The pages collection is a list of pages within the document. Each page is represented sequentially within the document and includes the orientation angle indicating if the page is rotated and the width and height (dimensions in pixels). The page units in the model output are computed as shown:

[\[+\] Expand table](#)

File format	Computed page unit	Total pages
Images (JPEG/JPG, PNG, BMP, HEIF)	Each image = 1 page unit	Total images
PDF	Each page in the PDF = 1 page unit	Total pages in the PDF
TIFF	Each image in the TIFF = 1 page unit	Total images in the TIFF
Word (DOCX)	Up to 3,000 characters = 1 page unit, embedded or linked images not supported	Total pages of up to 3,000 characters each
Excel (XLSX)	Each worksheet = 1 page unit, embedded or linked images not supported	Total worksheets
PowerPoint (PPTX)	Each slide = 1 page unit, embedded or linked images not supported	Total slides
HTML	Up to 3,000 characters = 1 page unit, embedded or linked images not supported	Total pages of up to 3,000 characters each

### Sample code

#### Python

```
# Analyze pages.
for page in result.pages:
    print(f"----Analyzing document from page #{page.page_number}----")
    print(f"Page has width: {page.width} and height: {page.height}, measured with unit: {page.unit}")
```

[View samples on GitHub.](#)

## Use pages for text extraction

For large multi-page PDF documents, use the `pages` query parameter to indicate specific page numbers or page ranges for text extraction.

## Paragraph extraction

The Read OCR model in Document Intelligence extracts all identified blocks of text in the `paragraphs` collection as a top level object under `analyzeResults`. Each entry in this collection represents a text block and includes the extracted text as `content` and the bounding `polygon` coordinates. The `span` information points to the text fragment within the top-level `content` property that contains the full text from the document.

JSON

```
"paragraphs": [
  {
    "spans": [],
    "boundingRegions": [],
    "content": "While healthcare is still in the early stages of its AI journey, we are seeing pharmaceutical and other life sciences organizations making major investments in AI and related technologies.\\" TOM LAWRY | National Director for AI, Health and Life Sciences | Microsoft"
  }
]
```

## Text, lines, and words extraction

The Read OCR model extracts print and handwritten style text as `lines` and `words`. The model outputs bounding `polygon` coordinates and `confidence` for the extracted words. The `styles` collection includes any handwritten style for lines if detected along with the spans pointing to the associated text. This feature applies to [supported handwritten languages](#).

For Microsoft Word, Excel, PowerPoint, and HTML, Document Intelligence Read model v3.1 and later versions extracts all embedded text as is. Texts are extracted as words and paragraphs. Embedded images aren't supported.

Sample code

Python

```
# Analyze lines.  
if page.lines:  
    for line_idx, line in enumerate(page.lines):  
        words = get_words(page, line)  
        print(  
            f"...Line # {line_idx} has {len(words)} words and text  
'{line.content}' within bounding polygon '{line.polygon}'"  
        )  
  
        # Analyze words.  
        for word in words:  
            print(f".....Word '{word.content}' has a confidence of  
{word.confidence}")
```

[View samples on GitHub.](#)

## Handwritten style extraction

The response includes classifying whether each text line is of handwriting style or not, along with a confidence score. For more information, see [handwritten language support](#). The following example shows an example JSON snippet.

JSON

```
"styles": [  
  {  
    "confidence": 0.95,  
    "spans": [  
      {  
        "offset": 509,  
        "length": 24  
      }  
      "isHandwritten": true  
    ]  
  }]
```

If you enabled the [font/style addon capability](#), you also get the font/style result as part of the `styles` object.

## Next steps v4.0

Complete a Document Intelligence quickstart:

- ✓ REST API
- ✓ C# SDK
- ✓ Python SDK
- ✓ Java SDK
- ✓ JavaScript

Explore our REST API:

[Document Intelligence API v4.0](#)

Find more samples on GitHub:

[Read model.](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence receipt model

Article • 12/11/2024

This content applies to: ✓ v4.0 (GA) | Previous versions: ✓ v3.1 (GA) ✓ v3.0 (GA) ✓ v2.1 (GA)

::: moniker-end

The Document Intelligence receipt model combines powerful Optical Character Recognition (OCR) capabilities with deep learning models to analyze and extract key information from sales receipts. Receipts can be of various formats and quality including printed and handwritten receipts. The API extracts key information such as merchant name, merchant phone number, transaction date, tax, and transaction total and returns structured JSON data. Receipt model v4.0 (GA) also supports other fields including

`ReceiptType`, `TaxDetails.NetAmount`, `TaxDetails.Description`, `TaxDetails.Rate` and

`CountryRegion`.

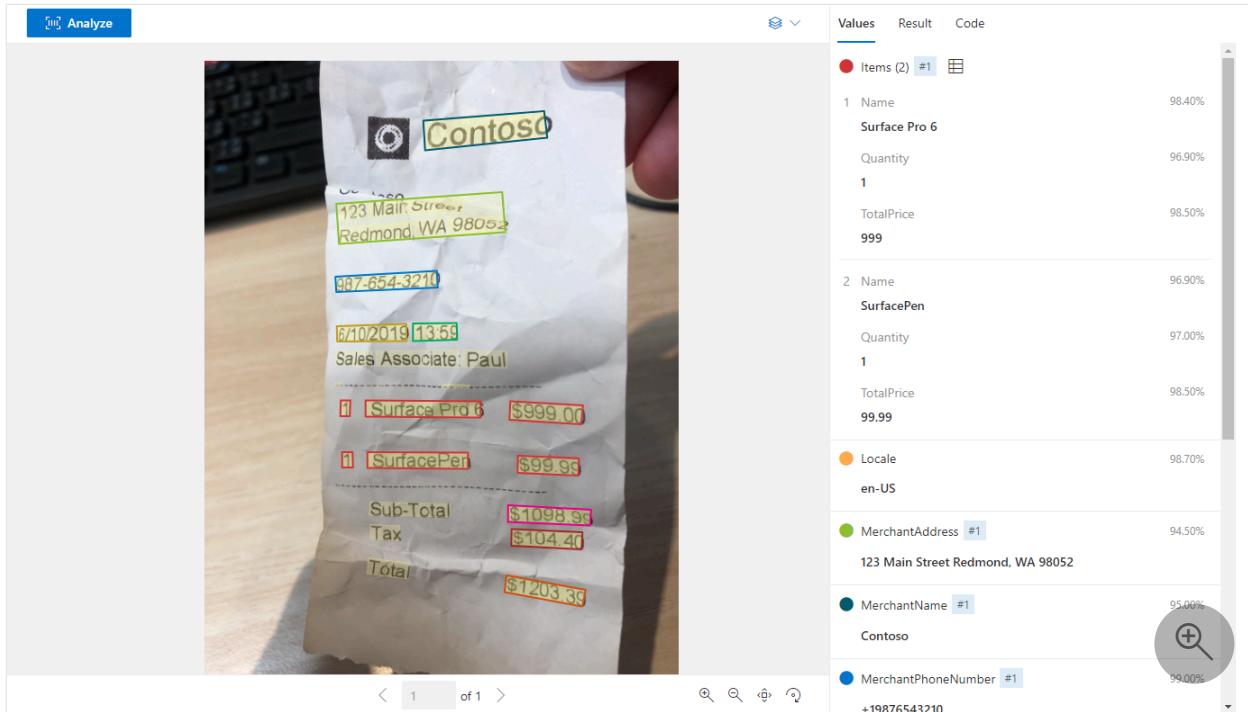
## Supported receipt types:

- Meal
- Supplies
- Hotel
- Fuel&Energy
- Transportation
- Communication
- Subscriptions
- Entertainment
- Training
- Healthcare

## Receipt data extraction

Receipt digitization encompasses the transformation of various types of receipts, including scanned, photographed, and printed copies, into a digital format for streamlined downstream processing. Examples include expense management, consumer behavior analysis, tax automation, etc. Using Document Intelligence with OCR (Optical Character Recognition) technology can extract and interpret data from these diverse receipt formats. Document Intelligence processing simplifies the conversion process but also significantly reduces the time and effort required, thus facilitating efficient data management, and retrieval.

Sample receipt processed with [Document Intelligence Studio](#):



The screenshot shows the Document Intelligence Studio interface. On the left is a photograph of a receipt from Contoso, dated 6/10/2019 at 13:53, with items for a Surface Pro 6 and a Surface Pen. On the right is a table of analysis results:

	Values	Result	Code
1	Items (2) #1	98.40%	
1	Name Surface Pro 6	96.90%	
1	Quantity 1		
1	TotalPrice \$999	98.50%	
2	Name SurfacePen	96.90%	
1	Quantity 1	97.00%	
1	TotalPrice \$99.99	98.50%	
1	Locale en-US	98.70%	
1	MerchantAddress #1 123 Main Street Redmond, WA 98052	94.50%	
1	MerchantName #1 Contoso	95.00%	
1	MerchantPhoneNumber #1 +19876543210	99.00%	

## Development options

Document Intelligence v4.0: 2024-11-30 (GA) supports the following tools, applications, and libraries:

[\[+\] Expand table](#)

Feature	Resources	Model ID
Receipt model	<ul style="list-style-type: none"><li><a href="#">Document Intelligence Studio</a></li><li><a href="#">REST API</a></li><li><a href="#">C# SDK</a></li><li><a href="#">Python SDK</a></li><li><a href="#">Java SDK</a></li><li><a href="#">JavaScript SDK</a></li></ul>	prebuilt-receipt

## Input requirements

- Supported file formats:

[\[+\] Expand table](#)

Model	PDF	Image:		Microsoft Office: Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
		JPEG/JPG	PNG, BMP, TIFF, HEIF	
Read	✓	✓		✓
Layout	✓	✓		✓
General Document	✓	✓		
Prebuilt	✓	✓		
Custom extraction	✓	✓		
Custom classification	✓	✓		✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

## Receipt model data extraction

See how Document Intelligence extracts data, including time and date of transactions, merchant information, and amount totals from receipts. You need the following resources:

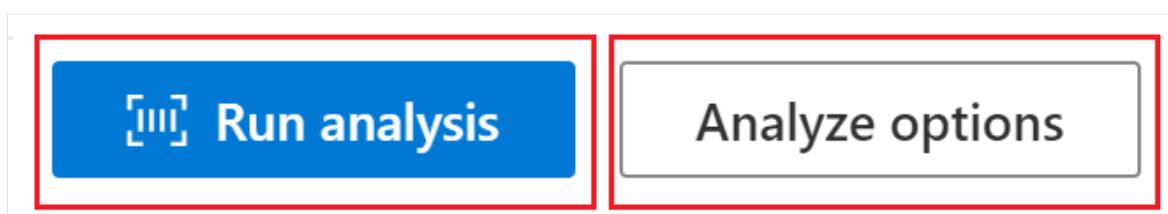
- An Azure subscription—you can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (`F0`) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

A screenshot of the Azure portal showing the 'Contoso-DI | Keys and Endpoint' page. The left sidebar shows various service management options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, and Keys and Endpoint (which is selected and highlighted with a red box). The main content area displays two key fields (KEY 1 and KEY 2) and their locations. A note about key security is present, and the endpoint URL is listed at the bottom.

### ⓘ Note

Document Intelligence Studio is available with v3.1 and v3.0 APIs and later versions.

1. On the [Document Intelligence Studio home page](#), select **Receipts**.
2. You can analyze the sample receipt or upload your own files.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options**:



[Try Document Intelligence Studio](#)

# Supported languages and locales

For a complete list of supported languages, see our [prebuilt models language support](#) page.

## Field extraction

For supported document extraction fields, refer to the [receipt model schema](#) page in our GitHub sample repository

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.
- Find more samples on [GitHub](#).

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence US tax document models

Article • 12/11/2024

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA) ::moniker-end

The Document Intelligence tax model uses powerful Optical Character Recognition (OCR) capabilities to analyze and extract key fields and line items from a select group of tax documents. Tax documents can be of various formats like 1099, 1098, W2, 1040, 1095A, 1095C, W-4, 1099-SSA. Input format can include phone-captured images, scanned documents, and digital PDFs. The API analyzes document text; extracts key information and returns a structured JSON data representation. The model currently supports certain English tax document formats.

## Supported tax form types:

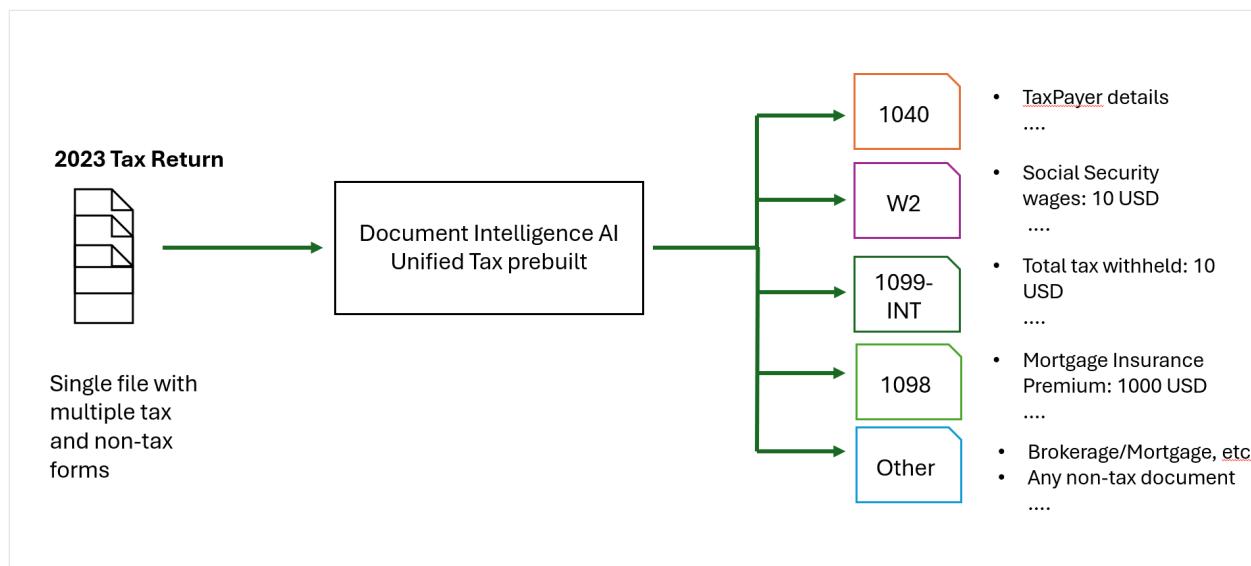
- Unified tax US
- W-2
- 1098
- 1098-E
- 1098-T
- 1099 and variations (added 1099-SSA)
- 1040 and variations
- 1095A, 1095C
- W-4

## Automated tax document processing

Automated tax document processing is the process of extracting key fields from tax documents. Historically, tax documents were processed manually. This model allows for the easy automation of tax scenarios.

## Unified Tax US

The `Unified US Tax` prebuilt model automatically detects and extracts data from `W2`, `1098`, `1040`, and `1099` tax forms in submitted documents. These documents can be composed of many tax or non-tax-related documents. The model only processes the forms it supports.



## Development options

Document Intelligence v4.0: 2024-11-30 (GA) supports the following tools, applications, and libraries:

  Expand table

Feature	Resources	Model ID
US tax form models	<ul style="list-style-type: none"> <li><a href="#">Document Intelligence Studio</a></li> <li><a href="#">REST API</a></li> <li><a href="#">C# SDK</a></li> <li><a href="#">Python SDK</a></li> <li><a href="#">Java SDK</a></li> <li><a href="#">JavaScript SDK</a></li> </ul>	<ul style="list-style-type: none"> <li>prebuilt-tax.us</li> <li>prebuilt-tax.us.W-2</li> <li>prebuilt-tax.us.W-4</li> <li>prebuilt-tax.us.1095A</li> <li>prebuilt-tax.us.1095C</li> <li>prebuilt-tax.us.1098</li> <li>prebuilt-tax.us.1098E</li> <li>prebuilt-tax.us.1098T</li> <li>prebuilt-tax.us.1099A</li> <li>prebuilt-tax.us.1099B</li> <li>prebuilt-tax.us.1099C</li> <li>prebuilt-tax.us.1099CAP</li> <li>prebuilt-tax.us.1099Combo</li> <li>prebuilt-tax.us.1099DIV</li> <li>prebuilt-tax.us.1099G</li> <li>prebuilt-tax.us.1099H</li> <li>prebuilt-tax.us.1099INT</li> <li>prebuilt-tax.us.1099K</li> <li>prebuilt-tax.us.1099LS</li> <li>prebuilt-tax.us.1099LTC</li> <li>prebuilt-tax.us.1099MISC</li> <li>prebuilt-tax.us.1099NEC</li> <li>prebuilt-tax.us.1099OID</li> <li>prebuilt-tax.us.1099PATR</li> </ul>

Feature	Resources	Model ID
		<ul style="list-style-type: none"> <li>• prebuilt-tax.us.1099Q</li> <li>• prebuilt-tax.us.1099QA</li> <li>• prebuilt-tax.us.1099R</li> <li>• prebuilt-tax.us.1099S</li> <li>• prebuilt-tax.us.1099SA</li> <li>• prebuilt-tax.us.1099SB</li> <li>• prebuilt-tax.us.1099SSA</li> <li>• prebuilt-tax.us.1040</li> <li>• prebuilt-tax.us.1040Schedule1</li> <li>• prebuilt-tax.us.1040Schedule2</li> <li>• prebuilt-tax.us.1040Schedule3</li> <li>• prebuilt-tax.us.1040Schedule8812</li> <li>• prebuilt-tax.us.1040ScheduleA</li> <li>• prebuilt-tax.us.1040ScheduleB</li> <li>• prebuilt-tax.us.1040ScheduleC</li> <li>• prebuilt-tax.us.1040ScheduleD</li> <li>• prebuilt-tax.us.1040ScheduleE</li> <li>• prebuilt-tax.us.1040ScheduleEIC</li> <li>• prebuilt-tax.us.1040ScheduleF</li> <li>• prebuilt-tax.us.1040ScheduleH</li> <li>• prebuilt-tax.us.1040ScheduleJ</li> <li>• prebuilt-tax.us.1040ScheduleR</li> <li>• prebuilt-tax.us.1040ScheduleSE</li> <li>• prebuilt-tax.us.1040Senior</li> </ul>

## Input requirements

- Supported file formats:

[Expand table](#)

Model	PDF	Image: JPEG/JPG, PNG, BMP, TIFF, HEIF	Microsoft Office: Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom	✓	✓	✓

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
classification			

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

## Try tax document data extraction

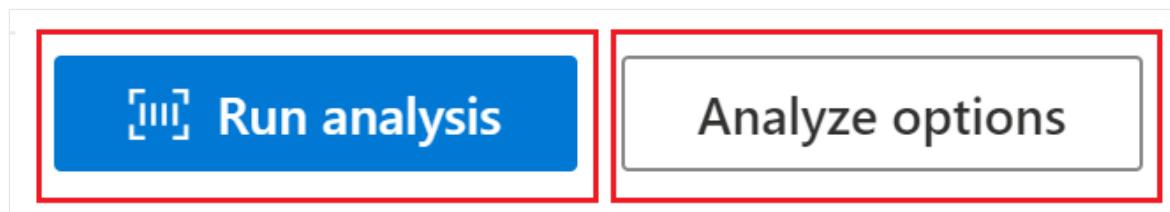
See how data, including customer information, vendor details, and line items, is extracted from invoices. You need the following resources:

- An Azure subscription—you can [create one for free ↗](#).
- A [Document Intelligence instance ↗](#) in the Azure portal. You can use the free pricing tier (F0) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.

The screenshot shows the Azure portal interface for managing a Cognitive Services resource named 'Contoso-DI'. The left sidebar lists various management options, and the main content area shows the 'Keys and Endpoint' configuration. A callout box highlights the 'Show Keys' section, which contains the two API keys ('KEY 1' and 'KEY 2'), their location ('westus2'), and the endpoint URL ('https://contoso-di.cognitiveservices.azure.com/'). A note at the top of the page cautions against sharing the keys securely.

## Document Intelligence Studio

1. On the [Document Intelligence Studio home page](#), select the supported tax document model.
2. You can analyze a sample tax document or upload your own files.
3. Select the **Run analysis** button and, if necessary, configure the **Analyze options**:



[Try Document Intelligence Studio](#)

## Supported languages and locales

See our [Language Support—prebuilt models](#) page for a complete list of supported languages.

## Field extraction

For supported document extraction fields, see the [tax document model schema](#) pages in our GitHub sample repository.

The tax documents key-value pairs and line items extracted are in the `documentResults` section of the JSON output.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

[Find more samples on GitHub.](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence custom models

Article • 12/11/2024

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)   
[v2.1 \(GA\)](#)

::: moniker-end

Document Intelligence uses advanced machine learning technology to identify documents, detect and extract information from forms and documents, and return the extracted data in a structured JSON output. With Document Intelligence, you can use document analysis models, pre-built/pre-trained, or your trained standalone custom models.

Custom models now include [custom classification models](#) for scenarios where you need to identify the document type before invoking the extraction model. Classifier models are available starting with the [2023-07-31 \(GA\)](#) API. A classification model can be paired with a custom extraction model to analyze and extract fields from forms and documents specific to your business. Standalone custom extraction models can be combined to create [composed models](#).

## Custom document model types

Custom document models can be one of two types, [custom template](#) or custom form and [custom neural](#) or custom document models. The labeling and training process for both models is identical, but the models differ as follows:

### Custom extraction models

To create a custom extraction model, label a dataset of documents with the values you want extracted and train the model on the labeled dataset. You only need five examples of the same form or document type to get started.

### Custom neural model

#### Important

Document Intelligence [v4.0 2024-11-30 \(GA\)](#) API supports custom neural model overlapping fields, signature detection and table, row and cell level confidence.

The custom neural (custom document) model uses deep learning models and base model trained on a large collection of documents. This model is then fine-tuned or adapted to your data when you train the model with a labeled dataset. Custom neural models support extracting key data fields from structured, semi-structured, and unstructured documents. When you're choosing between the two model types, start with a neural model to determine if it meets your functional needs. See [neural models](#) to learn more about custom document models.

## Custom template model

The custom template or custom form model relies on a consistent visual template to extract the labeled data. Variances in the visual structure of your documents affect the accuracy of your model. Structured forms such as questionnaires or applications are examples of consistent visual templates.

Your training set consists of structured documents where the formatting and layout are static and constant from one document instance to the next. Custom template models support key-value pairs, selection marks, tables, signature fields, and regions. Template models can be trained on documents in any of the [supported languages](#). For more information, see [custom template models](#).

If the language of your documents and extraction scenarios supports custom neural models, we recommend that you use custom neural models over template models for higher accuracy.

### Tip

To confirm that your training documents present a consistent visual template, remove all the user-entered data from each form in the set. If the blank forms are identical in appearance, they represent a consistent visual template.

For more information, see [Interpret and improve accuracy and confidence for custom models](#).

## Input requirements

- For best results, provide one clear photo or high-quality scan per document.
- Supported file formats:

 Expand table

Model	PDF	Image: jpeg/jpg, png, bmp, tiff, heif	Microsoft Office: Word (docx), Excel (xlsx), PowerPoint (pptx)
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

\* Microsoft Office files are currently not supported for other models or versions.

- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 x 50 pixels and 10,000 px x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8-point text at 150 dots per inch.
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
- For custom extraction model training, the total size of training data is 50 MB for template model and 1G-MB for the neural model.
- For custom classification model training, the total size of training data is 1GB with a maximum of 10,000 pages.

## Optimal training data

Training input data is the foundation of any machine learning model. It determines the quality, accuracy, and performance of the model. Therefore, it's crucial to create the best training input data possible for your Document Intelligence project. When you use the

Document Intelligence custom model, you provide your own training data. Here are a few tips to help train your models effectively:

- Use text-based instead of image-based PDFs when possible. One way to identify an image\*based PDF is to try selecting specific text in the document. If you can select only the entire image of the text, the document is image based, not text based.
- Organize your training documents by using a subfolder for each format (JPEG/JPG, PNG, BMP, PDF, or TIFF).
- Use forms that have all of the available fields completed.
- Use forms with differing values in each field.
- Use a larger dataset (more than five training documents) if your images are low quality.
- Determine if you need to use a single model or multiple models composed into a single model.
- Consider segmenting your dataset into folders, where each folder is a unique template. Train one model per folder, and compose the resulting models into a single endpoint. Model accuracy can decrease when you have different formats analyzed with a single model.
- Consider segmenting your dataset to train multiple models if your form has variations with formats and page breaks. Custom forms rely on a consistent visual template.
- Ensure that you have a balanced dataset by accounting for formats, document types, and structure.

## Build mode

The `build custom model` operation adds support for the *template* and *neural* custom models. Previous versions of the REST API and client libraries only supported a single build mode that is now known as the *template* mode.

- Template models only accept documents that have the same basic page structure—a uniform visual appearance—or the same relative positioning of elements within the document.

- Neural models support documents that have the same information, but different page structures. Examples of these documents include United States W2 forms, which share the same information, but vary in appearance across companies.

This table provides links to the build mode programming language SDK references and code samples on GitHub:

[\[+\] Expand table](#)

Programming language	SDK reference	Code sample
C#/.NET	<a href="#">DocumentBuildMode Struct</a>	<a href="#">Sample_BuildCustomModelAsync.cs ↗</a>
Java	<a href="#">DocumentBuildMode Class</a>	<a href="#">BuildModel.java ↗</a>
JavaScript	<a href="#">DocumentBuildMode type</a>	<a href="#">buildModel.js ↗</a>
Python	<a href="#">DocumentBuildMode Enum</a>	<a href="#">sample_build_model.py ↗</a>

## Compare model features

The following table compares custom template and custom neural features:

[\[+\] Expand table](#)

Feature	Custom template (form)	Custom neural (document)
Document structure	Template, form, and structured	Structured, semi-structured, and unstructured
Training time	1 to 5 minutes	20 minutes to 1 hour
Data extraction	Key-value pairs, tables, selection marks, coordinates, and signatures	Key-value pairs, selection marks, and tables
Overlapping fields	Not supported	Supported
Document variations	Requires a model per each variation	Uses a single model for all variations
Language support	<a href="#">Language support custom template</a>	<a href="#">Language support custom neural</a>

## Custom classification model

Document classification is a new scenario supported by Document Intelligence with the [2023-07-31](#) (v3.1 GA) API. The document classifier API supports classification and splitting scenarios. Train a classification model to identify the different types of documents your application supports. The input file for the classification model can contain multiple documents and classifies each document within an associated page range. To learn more, see [custom classification](#) models.

 **Note**

The [v4.0 2024-11-30 \(GA\)](#) document classification model supports Office document types for classification. This API version also introduces [incremental training](#) for the classification model.

## Custom model tools

Document Intelligence v3.1 and later models support the following tools, applications, and libraries, programs, and libraries:

[Expand table](#)

Feature	Resources	Model ID
Custom model	<ul style="list-style-type: none"><li><a href="#">Document Intelligence Studio</a></li><li><a href="#">REST API</a></li><li><a href="#">C# SDK</a></li><li><a href="#">Python SDK</a></li></ul>	<i>custom-model-id</i>

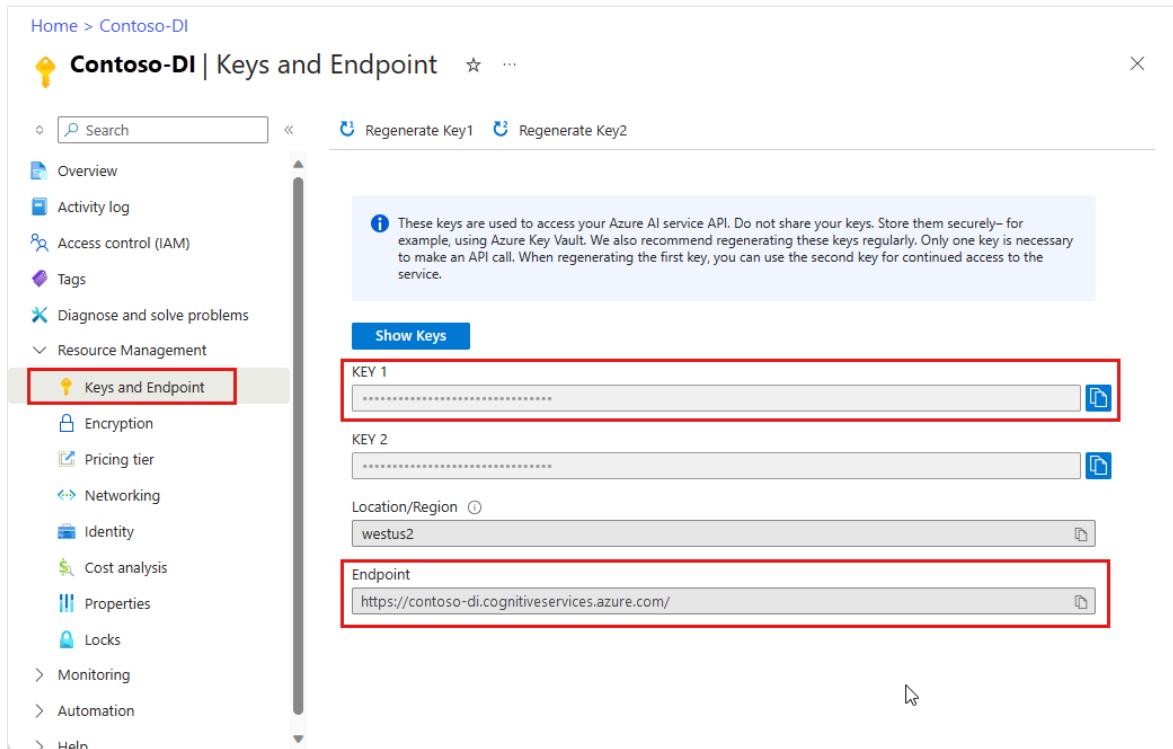
## Custom model life cycle

The life cycle of a custom model depends on the API version that is used to train it. If the API version is a general availability (GA) version, the custom model has the same life cycle as that version. The custom model isn't available for inference when the API version is deprecated. If the API version is a preview version, the custom model has the same life cycle as the preview version of the API.

## Build a custom model

Extract data from your specific or unique documents using custom models. You need the following resources:

- An Azure subscription. You can [create one for free](#).
- A [Document Intelligence instance](#) in the Azure portal. You can use the free pricing tier (F0) to try the service. After your resource deploys, select **Go to resource** to get your key and endpoint.



## Document Intelligence Studio

### ! Note

Document Intelligence Studio is available with v3.1 and v3.0 APIs.

1. On the **Document Intelligence Studio** home page, select **Custom extraction models**.
2. Under **My Projects**, select **Create a project**.
3. Complete the project details fields.
4. Configure the service resource by adding your **Storage account** and **Blob container** to **Connect your training data source**.
5. Review and create your project.
6. Add your sample documents to label, build, and test your custom model.

[Try Document Intelligence Studio](#)

For a detailed walkthrough to create your first custom extraction model, see [How to create a custom extraction model](#).

## Custom model extraction summary

This table compares the supported data extraction areas:

[\[+\] Expand table](#)

Model	Form fields	Selection marks	Structured fields (Tables)	Signature	Region labeling	Overlapping fields
Custom template	✓	✓	✓	✓	✓	n/a
Custom neural	✓	✓	✓	✓	*	✓

**Table symbols:**

✓—Supported

\*\*n/a—Currently unavailable;

\*—Behaves differently depending upon model. With template models, synthetic data is generated at training time. With neural models, exiting text recognized in the region is selected.

### 💡 Tip

When choosing between the two model types, start with a custom neural model if it meets your functional needs. See [custom neural](#) to learn more about custom neural models.

## Custom model development options

The following table describes the features available with the associated tools and client libraries. As a best practice, ensure that you use the compatible tools listed here.

[\[+\] Expand table](#)

Document type	REST API	SDK	Label and Test Models
Custom template v 4.0 v3.1 v3.0	<a href="#">Document Intelligence 3.1</a>	<a href="#">Document Intelligence SDK</a>	<a href="#">Document Intelligence Studio ↗</a>
Custom neural v4.0 v3.1 v3.0	<a href="#">Document Intelligence 3.1</a>	<a href="#">Document Intelligence SDK</a>	<a href="#">Document Intelligence Studio ↗</a>
Custom form v2.1	<a href="#">Document Intelligence 2.1 GA API</a>	<a href="#">Document Intelligence SDK</a>	<a href="#">Sample labeling tool ↗</a>

### ⓘ Note

Custom template models trained with the 3.0 API will have a few improvements over the 2.1 API stemming from improvements to the OCR engine. Datasets used to train a custom template model using the 2.1 API can still be used to train a new model using the 3.0 API.

- For best results, provide one clear photo or high-quality scan per document.
- Supported file formats are JPEG/JPG, PNG, BMP, TIFF, and PDF (text-embedded or scanned). Text-embedded PDFs are best to eliminate the possibility of error in character extraction and location.
- For PDF and TIFF files, up to 2,000 pages can be processed. With a free tier subscription, only the first two pages are processed.
- The file size must be less than 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 x 50 pixels and 10,000 x 10,000 pixels.
- PDF dimensions are up to 17 x 17 inches, corresponding to Legal or A3 paper size, or smaller.
- The total size of the training data is 500 pages or less.
- If your PDFs are password-locked, you must remove the lock before submission.

### 💡 Tip

Training data:

- If possible, use text-based PDF documents instead of image-based documents. Scanned PDFs are handled as images.

- Please supply only a single instance of the form per document.
- For filled-in forms, use examples that have all their fields filled in.
- Use forms with different values in each field.
- If your form images are of lower quality, use a larger dataset. For example, use 10 to 15 images.

## Supported languages and locales

See our [Language Support—custom models](#) page for a complete list of supported languages.

## Next steps

- Try processing your own forms and documents with the [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence composed custom models

Article • 02/27/2025

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)   
[v2.1 \(GA\)](#)

::: moniker-end

## Important

The v4.0 `2024-11-30` (GA) [model compose](#) operation adds an explicitly trained classifier instead of an implicit classifier for analysis. For the previous composed model version, see Composed custom models v3.1. If you're currently using composed models, consider upgrading to the latest implementation.

## What is a composed model?

With composed models, you can group multiple custom models into a composed model called with a single model ID. For example, your composed model might include custom models trained to analyze your supply, equipment, and furniture purchase orders. Instead of manually trying to select the appropriate model, you can use a composed model to determine the appropriate custom model for each analysis and extraction.

Some scenarios require classifying the document first and then analyzing the document with the model best suited to extract the fields from the model. Such scenarios can include ones where a user uploads a document but the document type isn't explicitly known. Another scenario can be when multiple documents are scanned together into a single file and the file is submitted for processing. Your application then needs to identify the component documents and select the best model for each document.

In previous versions, the `model compose` operation performed an implicit classification to decide which custom model best represents the submitted document. The `2024-11-30 (GA)` implementation of the `model compose` operation replaces the implicit classification from the earlier versions with an explicit classification step and adds conditional routing.

## Benefits of the new model compose operation

The new `model compose` operation requires you to train an explicit classifier and provides several benefits.

- **Continual incremental improvement.** You can consistently improve the quality of the classifier by adding more samples and [incrementally improving classification](#). This fine tuning ensures your documents are always routed to the right model for extraction.
- **Complete control over routing.** By adding confidence-based routing, you provide a confidence threshold for the document type and the classification response.
- **Ignore document specific document types during the operation.** Earlier implementations of the `model compose` operation selected the best analysis model for extraction based on the confidence score even if the highest confidence scores were relatively low. By providing a confidence threshold or explicitly not mapping a known document type from classification to an extraction model, you can ignore specific document types.
- **Analyze multiple instances of the same document type.** When paired with the `splitMode` option of the classifier, the `model compose` operation can detect multiple instances of the same document in a file and split the file to process each document independently. Using `splitMode` enables the processing of multiple instances of a document in a single request.
- **Support for add on features.** [Add on features](#) like query fields or barcodes can also be specified as a part of the analysis model parameters.
- **Assigned custom model maximum expanded to 500.** The new implementation of the `model compose` operation allows you to assign up to 500 trained custom models to a single composed model.

## How to use model compose

- Start by collecting samples of all your needed documents including samples with information that should be extracted or ignored.
- Train a classifier by organizing the documents in folders where the folder names are the document type you intend to use in your composed model definition.
- Finally, train an extraction model for each of the document types you intend to use.

- Once your classification and extraction models are trained, use the Document Intelligence Studio, client libraries, or the [REST API](#) to compose the classification and extraction models into a composed model.

Use the `splitMode` parameter to control the file splitting behavior:

- None.** The entire file is treated as a single document.
- perPage.** Each page in the file is treated as a separate document.
- auto.** The file is automatically split into documents.

## Billing and pricing

Composed models are billed the same as individual custom models. The pricing is based on the number of pages analyzed by the downstream analysis model. Billing is based on the extraction price for the pages routed to an extraction model. With the addition of the explicit classification charges are incurred for the classification of all pages in the input file. For more information, see the [Document Intelligence pricing page](#).

## Development options

Document Intelligence v4.0:2024-11-30 (GA) supports the following tools, applications, and libraries:

[+] [Expand table](#)

Feature	Resources
<i>Custom model</i>	<ul style="list-style-type: none"><li><a href="#">Document Intelligence Studio</a></li><li><a href="#">REST API</a></li><li><a href="#">C# SDK</a></li><li><a href="#">Java SDK</a></li><li><a href="#">JavaScript SDK</a></li><li><a href="#">Python SDK</a></li></ul>
<i>Composed model</i>	<ul style="list-style-type: none"><li><a href="#">Document Intelligence Studio</a></li><li><a href="#">REST API</a></li><li><a href="#">C# SDK</a></li><li><a href="#">Java SDK</a></li><li><a href="#">JavaScript SDK</a></li><li><a href="#">Python SDK</a></li></ul>

## Next steps

Learn to create and compose custom models:

[Build a custom model](#)

[Compose custom models](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Use Document Intelligence incremental classifiers

Article • 02/27/2025

This content applies to:  v4.0 (GA) 

Azure AI Document Intelligence is a cloud-based Azure AI service that enables you to build intelligent document processing solutions. Document Intelligence APIs analyze images, PDFs, and other document files to extract and detect various content, layout, style, and semantic elements.

[Document Intelligence custom classification models](#) are deep-learning-model types that combine layout and language features to accurately detect and identify documents you process within your applications. Custom classification models perform classification of input files one page at a time to identify the documents within and can also identify multiple documents or multiple instances of a single document within an input file.

Document Intelligence document classifiers identify known document types in files. When processing an input file with multiple document types or when you don't know the document type, use a classifier to identify the document. Classifiers should be periodically updated whenever the following changes occur:

- You add new templates for an existing class.
- You add new document types for recognition.
- Classifier confidence is low.

In some scenarios, you can no longer have the original set of documents used to train the classifier. With incremental training, you can update the classifier with just the new labeled samples.

## Note

Incremental training only applies to document classifier models and not custom models.

Incremental training is useful when you want to improve the quality of a custom classifier. Adding new training samples for existing classes improves the confidence of the model for existing document types. For instance, if a new version of an existing form is added or there's a new document type. An example can be when your application starts supporting a new document type as a valid input.

# Getting started with incremental training

- Incremental training doesn't introduce any new API endpoints.
- The `documentClassifiers:build` request payload is modified to support incremental training.
- Incremental training results in a new classifier model being created with the existing classifier left untouched.
- The new classifier has all the document samples and types of the old classifier along with the newly provided samples. You need to ensure your application is updated to work with the newly trained classifier.

## ⓘ Note

Copy operation for classifiers is currently unavailable.

## Create an incremental classifier build request

The incremental classifier build request is similar to the [classify document build request](#) but includes the new `baseClassifierId` property. The `baseClassifierId` is set to the existing classifier that you want to extend. You also need to provide the `docTypes` for the different document types in the sample set. By providing a `docType` that exists in the `baseClassifier`, the samples provided in the request are added to the samples provided when the base classifier was trained. New `docType` values added in the incremental training are only added to the new classifier. The process to specify the samples remains unchanged. For more information, see [training a classifier model](#).

## Sample POST request

*Sample POST request to build an incremental document classifier*

`POST {your-endpoint}/documentintelligence/documentClassifiers:build?api-version=2024-11-30`

JSON

```
{  
  "classifierId": "myAdaptedClassifier",  
  "description": "Classifier description",  
  "baseClassifierId": "myOriginalClassifier",  
  "sampleDocuments": [  
    {"id": "1", "text": "This is a sample document.", "type": "Text"},  
    {"id": "2", "text": "This is another sample document.", "type": "Text"}]  
}
```

```
"docTypes": {
  "formA": {
    "azureBlobSource": {
      "containerUrl": "https://myStorageAccount.blob.core.windows.net/myContainer?mySasToken",
      "prefix": "formADocs/"
    }
  },
  "formB": {
    "azureBlobFileListSource": {
      "containerUrl": "https://myStorageAccount.blob.core.windows.net/myContainer?mySasToken",
      "fileList": "formB.jsonl"
    }
  }
}
```

## POST response

All Document Intelligence APIs are asynchronous, polling the returned operation location provides a status on the build operation. Classifiers are fast to train and your classifier can be ready to use in a minute or two.

Upon successful completion:

- The successful `POST` method returns a `202 OK` response code indicating that the service created the request.
- The translated documents are located in your target container.
- The `POST` request also returns response headers including `Operation-Location`. The value of this header contains a `resultId` that can be queried to get the status of the asynchronous operation and retrieve the results using a `GET` request with your same resource subscription key.

## Sample GET request

*Sample `GET` request to retrieve the result of an incremental document classifier*

```
GET {your-
endpoint}/documentintelligence/documentClassifiers/{classifierId}/analyzeResults/{re-
sultId}?api-version=2024-11-30
```

JSON

```
{
```

```
"classifierId": "myAdaptedClassifier",
"description": "Classifier description",
"createdDateTime": "2022-07-30T00:00:00Z",
"expirationDateTime": "2023-01-01T00:00:00Z",
"apiVersion": "2024-11-30",

"baseClassifierId": "myOriginalClassifier",

"docTypes": {
    "formA": {
        "azureBlobSource": {
            "containerUrl": "https://myStorageAccount.blob.core.windows.net/myContainer",
            "prefix": "formADocs/"
        }
    },
    "formB": {
        "azureBlobFileListSource": {
            "containerUrl": "https://myStorageAccount.blob.core.windows.net/myContainer",
            "fileList": "formB.jsonl"
        }
    }
}
```

## GET response

The `GET` response from an incrementally trained classifier differs from the standard classifier `GET` response. The incrementally trained classifier doesn't return all the document types supported. It returns the document types added or updated in the incremental training step and the extended base classifier. To get a complete list of document types, the base classifier must be listed. Deleting a base classifier doesn't impact the use of an incrementally trained classifier.

## Limits

- Incremental training only works when the base classifier and the incrementally trained classifier are both trained on the same API version. As a result, the incrementally trained classifier has the same [model lifecycle](#) as the base classifier.
- Training dataset size limits for the incremental classifier are the same as for other classifier model. See [service limits](#) for a complete list of applicable limits.

## Next steps

- Learn more about [document classification](#)
- 

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence custom template model

Article • 12/11/2024

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)  v2.1 (GA)

Custom template (formerly custom form) is an easy-to-train document model that accurately extracts labeled key-value pairs, selection marks, tables, regions, and signatures from documents. Template models use layout cues to extract values from documents and are suitable to extract fields from highly structured documents with defined visual templates.

Custom template models share the same labeling format and strategy as custom neural models, with support for more field types and languages.

## Model capabilities

Custom template models support key-value pairs, selection marks, tables, signature fields, and selected regions.

 Expand table

Form fields	Selection marks	Tabular fields (Tables)	Signature	Selected regions	Overlapping fields
Supported	Supported	Supported	Supported	Supported	Not supported

## Tabular fields

With the release of API versions v3.0 and later, custom template models add support for cross page tabular fields (tables):

- To label a table that spans multiple pages, label each row of the table across the different pages in a single table.
- As a best practice, ensure that your dataset contains a few samples of the expected variations. For example, include samples where the entire table is on a single page and where tables span two or more pages if you expect to see those variations in documents.

Tabular fields are also useful when extracting repeating information within a document that isn't recognized as a table. For example, a repeating section of work experiences in a resume can be labeled and extracted as a tabular field.

## Dealing with variations

Template models rely on a defined visual template, changes to the template results in lower accuracy. In those instances, split your training dataset to include at least five samples of each template and train a model for each of the variations. You can then [compose](#) the models into a single endpoint. For subtle variations, like digital PDF documents and images, it's best to include at least five examples of each type in the same training dataset.

## Input requirements

- For best results, provide one clear photo or high-quality scan per document.
- Supported file formats:

[+] Expand table

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), and HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom	✓	✓	

\* Microsoft Office files are currently not supported for other models or versions.

- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 x 50 pixels and 10,000 px x 10,000 pixels.

- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8-point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
- For custom extraction model training, the total size of training data is 50 MB for template model and 1G-MB for the neural model.
- For custom classification model training, the total size of training data is 1GB with a maximum of 10,000 pages.

## Training a model

Custom template models are generally available starting with v2.0 API and later versions. If you're starting with a new project or have an existing labeled dataset, use the v3.1 or v3.0 API with Document Intelligence Studio to train a custom template model.

[\[+\] Expand table](#)

Model	REST API	SDK	Label and Test Models
Custom template	<a href="#">v3.1 API</a>	<a href="#">Document Intelligence SDK</a>	<a href="#">Document Intelligence Studio</a> ↗

With the v3.0 and later APIs, the build operation to train model supports a new `buildMode` property, to train a custom template model, set the `buildMode` to `template`.

REST

```
https://{{endpoint}}/documentintelligence/documentModels:build?api-version=2024-11-30
```

```
{
  "modelId": "string",
  "description": "string",
  "buildMode": "template",
  "azureBlobSource": {
    "containerUrl": "string",
    "prefix": "string"
  }
}
```

# Supported languages and locales

See our [Language Support—custom models](#) page for a complete list of supported languages.

## Next steps

Learn to create and compose custom models:

[Build a custom model](#)

[Compose custom models](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence custom neural model

Article • 01/15/2025

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)

Custom neural document models or neural models are a deep learned model type that combines layout and language features to accurately extract labeled fields from documents. The base custom neural model is trained on various document types that makes it suitable to be trained for extracting fields from structured and semi-structured documents. Custom neural models are available in the [v3.0 and later models](#). With V4.0, custom neural model now supports signature detection. The table below lists common document types for each category:

 Expand table

Documents	Examples
Structured	surveys, questionnaires
Semi-structured	invoices, purchase orders

Custom neural models share the same labeling format and strategy as [custom template](#) models. Currently custom neural models only support a subset of the field types supported by custom template models.

## Model capabilities

### Important

Custom neural v4.0 2024-11-30 (GA) model supports signature detection, table cell confidence, and overlapping fields.

Custom neural models currently support key-value pairs and selection marks and structured fields (tables).

 Expand table

Form fields	Selection marks	Tabular fields	Signature	Region labeling	Overlapping fields
Supported	Supported	Supported	Supported	Supported <sup>1</sup>	Supported <sup>2</sup>

<sup>1</sup> Region labels in custom neural models use the results from the Layout API for specified region. This feature is different from template models where, if no value is present, text is generated at training time.

<sup>2</sup> Overlapping fields are supported with REST API version **2024-11-30 (GA)**. Overlapping fields have some limits. For more information, see [overlapping fields](#).

## Build mode

The `Build` operation supports *template* and *neural* custom models. Previous versions of the REST API and client libraries only supported a single build mode that is now known as the *template* mode.

Neural models support documents that have the same information, but different page structures. Examples of these documents include United States W2 forms, which share the same information, but can vary in appearance across companies. For more information, see [Custom model build mode](#).

## Signature detection

Custom neural v4.0 2024-11-30 (GA) model supports signature detection. To label a signature, use field type as Signature and draw the regions for signature. Signature field only supports one draw region per field. To train a custom neural model with signature detection, you need to use at least **five samples** with signature labeled along with variations to get the most accurate results.

## Tabular fields

Custom neural v4.0 **2024-11-30 (GA)** model supports tabular fields (tables) to analyze table, row, and cell data with added confidence:

- Models trained with API version 2022-06-30-preview, or later will accept tabular field labels.
- Documents analyzed with custom neural models using API version 2022-06-30-preview or later will produce tabular fields aggregated across the tables.
- The results can be found in the `analyzeResult` object's `documents` array that is returned following an analysis operation.

Tabular fields support **cross page tables** by default:

- To label a table that spans multiple pages, label each row of the table across the different pages in a single table.
- As a best practice, ensure that your dataset contains a few samples of the expected variations. For example, include samples where the entire table is on a single page and where tables span two or more pages.

Tabular fields are also useful when extracting repeating information within a document that isn't recognized as a table. For example, a repeating section of work experiences in a resume can be labeled and extracted as a tabular field.

Tabular fields provide **table**, **row** and **cell confidence** with the [2024-11-30 \(GA\)](#) API:

- Fixed or dynamic tables add confidence support for the following elements:
  - Table confidence, a measure of how accurately the entire table is recognized.
  - Row confidence, a measure of recognition of an individual row.
  - Cell confidence, a measure of recognition of an individual cell.
- The recommended approach is to review the accuracy in a top-down manner starting with the table first, followed by the row and then the cell. See [confidence and accuracy scores](#) to learn more about table, row, and cell confidence.

## Overlapping fields

Custom neural v4.0 2024-11-30 (GA) model supports overlapping fields:

To use the overlapping fields, your dataset needs to contain at least one sample with the expected overlap. To label an overlap, use **region labeling** to designate each of the spans of content (with the overlap) for each field. Labeling an overlap with field selection (highlighting a value) fails in the Studio as region labeling is the only supported labeling tool for indicating field overlaps. Overlap support includes:

- Complete overlap. The same set of tokens are labeled for two different fields.
- Partial overlap. Some tokens belong to both fields, but there are tokens that are only part of one field or the other.

Overlapping fields have some limits:

- Any token or word can only be labeled as two fields.
- overlapping fields in a table can't span table rows.
- Overlapping fields can only be recognized if at least one sample in the dataset contains overlapping labels for those fields.

To use overlapping fields, label your dataset with the overlaps and train the model with the API version **\*\*2024-11-30 (GA)\*\***.

## Supported languages and locales

See our [Language Support—custom models](#) for a complete list of supported languages.

## Supported regions

As of October 18, 2022, Document Intelligence custom neural model training will only be available in the following Azure regions until further notice:

- Australia East
- Brazil South
- Canada Central
- Central India
- Central US
- East Asia
- East US
- East US2
- France Central
- Japan East
- South Central US
- Southeast Asia
- UK South
- West Europe
- West US2
- US Gov Arizona
- US Gov Virginia

### Tip

You can [copy a model](#) trained in one of the select regions listed to **any other region** and use it accordingly.

Use the [REST API](#) or [Document Intelligence Studio](#) ↗ to copy a model to another region.

## Input requirements

- For best results, provide one clear photo or high-quality scan per document.
- Supported file formats:

[\[+\] Expand table](#)

Model	PDF	Image: jpeg/jpg, png, bmp, tiff, heif	Microsoft Office: Word (docx), Excel (xlsx), PowerPoint (pptx), and HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom neural	✓	✓	

\* Microsoft Office files are currently not supported for other models or versions.

- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 x 50 pixels and 10,000 px x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8-point text at 150 dots per inch.
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
- For custom extraction model training, the total size of training data is 50 MB for template model and 1G-MB for the neural model.
- For custom classification model training, the total size of training data is 1GB with a maximum of 10,000 pages.

## Best practices

Custom neural models differ from custom template models in a few different ways. The custom template or model relies on a consistent visual template to extract the labeled data. Custom neural models support structured, and semi-structured to extract fields. When you're choosing between the model types, start with a neural model, and test to determine if it supports your functional needs.

- **Dealing with variations** - Custom neural models can generalize across different formats of a single document type. As a best practice, create a single model for all variations of a document type. Add at least five labeled samples for each of the different variations to the training dataset.
- **Field naming** - When you label the data, labeling the field relevant to the value improves the accuracy of the key-value pairs extracted. For example, for a field value containing the supplier ID, consider naming the field *supplier\_id*. Field names should be in the language of the document.
- **Labeling contiguous values** - Value tokens/words of one field must be either:
  - In a consecutive sequence in natural reading order, without interleaving with other fields
  - In a region that don't cover any other fields
- **Representative data** - Values in training cases should be diverse and representative. For example, if a field is named *date*, values for this field should be a date. Synthetic value like a random string can affect model performance.

## Current Limitations

- Custom neural model doesn't recognize values split across page boundaries.
- Custom neural unsupported field types are ignored if a dataset labeled for custom template models is used to train a custom neural model.
- Custom neural models are limited to 20 build operations per month for versions 3.x. Open a support request if you need the limit increased. For more information, see [Document Intelligence service quotas and limits](#).

## Training a model

Custom neural models are available in the [v3.0 and later models](#).

 Expand table

Document Type	REST API	SDK	Label and Test Models
Custom document	Document Intelligence 3.1	Document Intelligence SDK	Document Intelligence Studio ↗

The `Build` operation to train model supports a new `buildMode` property, to train a custom neural model, set the `buildMode` to `neural`.

Bash

```
https://{{endpoint}}/documentintelligence/documentModels:build?api-version=2024-11-30

{
  "modelId": "string",
  "description": "string",
  "buildMode": "neural",
  "azureBlobSource": {
    "containerUrl": "string",
    "prefix": "string"
  }
}
```

## Billing

With version `v4.0 2024-11-30 (GA)`, you can train your custom neural model for longer durations than the standard 30 minutes. Previous versions are limited to 30 minutes per training instance, with a total of 20 free training instances per month. With version `v4.0 2024-11-30 (GA)`, you can receive **10 hours of free model training**, and train a model for as long as 10 hours.

You can choose to spend all of 10 free hours on a single model build with a large set of data, or utilize it across multiple builds by adjusting the maximum duration value for the `build` operation by specifying `maxTrainingHours`:

Bash

```
POST https://{{endpoint}}/documentintelligence/documentModels:build?api-version=2024-11-30

{
  "modelId": "string",
  "description": "string",
  "buildMode": "neural",
  "maxTrainingHours": 10
}
```

```
...,  
    "maxTrainingHours": 10  
}
```

## ⓘ Important

- If you would like to train more neural models or train models for a longer time period that **exceed 10 hours**, billing charges apply. For details on the billing charges, refer to the [pricing page](#).
- You can opt in for this paid training service by setting the `maxTrainingHours` to the desired maximum number of hours. API calls with no budget but with the `maxTrainingHours` set to over 10 hours fail.
- Each build takes a different amount of time depending on the type and size of the training dataset. Billing is calculated for the actual time spent training the neural model with a minimum of 30 minutes per training job.
- This paid training feature enables you to train larger data sets for longer durations with flexibility in the training hours.

## Bash

```
GET /documentModels/{myCustomModel}  
{  
    "modelId": "myCustomModel",  
    "trainingHours": 0.23,  
    "docTypes": { ... },  
    ...  
}
```

## ⓘ Note

For Document Intelligence versions v3.1 (2023-07-31) and v3.0 (2022-08-31), custom neural model's paid training isn't enabled. For the two older versions, there's a maximum of 30-minutes training duration per model. If you would like to train more than 20 model instances, you can create an [Azure support ticket](#) to increase in the training limit.

# Next steps

Learn to create and compose custom models:

[Build a custom model](#)

[Compose custom models](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence custom classification model

Article • 02/27/2025

This content applies to:  v4.0 (GA) | Previous version:  v3.1 (GA)

## Important

- The `v4.0 2024-11-30 (GA)` API, custom classification model doesn't split documents by default during the analyzing process.
- You need to explicitly set the `splitMode` property to `auto` to preserve the behavior from previous releases. The default for `splitMode` is `none`.
- If your input file contains multiple documents, you need to enable splitting by setting the `splitMode` to `auto`.

Azure AI Document Intelligence is a cloud-based Azure AI service that enables you to build intelligent document processing solutions. Document Intelligence APIs analyze images, PDFs, and other document files to extract and detect various content, layout, style, and semantic elements.

Custom classification models are deep-learning-model types that combine layout and language features to accurately detect and identify documents you process within your application. Custom classification models perform classification of an input file one page at a time to identify the documents within and can also identify multiple documents or multiple instances of a single document within an input file.

## Model capabilities

### Note

- Custom classification v4.0 `2024-11-30 (GA)` models support incremental training. You can add new samples to existing classes or add new classes by referencing an existing classifier.
- Custom classification v3.1 `2023-07-31 (GA)` model doesn't support model copy. To use the model copy feature, train the model using the latest v4.0 (GA) model.

Custom classification models can analyze a single- or multi-file documents to identify if any of the trained document types are contained within an input file. Here are the currently supported scenarios:

- A single file containing one document type, such as a loan application form.
  - A single file containing multiple document types. For instance, a loan application package that contains a loan application form, payslip, and bank statement.
  - A single file containing multiple instances of the same document. For instance, a collection of scanned invoices.
- ✓ Training a custom classifier requires at least `two` distinct classes and a minimum of `five` document samples per class. The model response contains the page ranges for each of the classes of documents identified.
- ✓ The maximum allowed number of classes is `1,000`. The maximum allowed number of document samples per class is `100`.

The model classifies each page of the input document, unless specified, to one of the classes in the labeled dataset. You can specify the page numbers to analyze in the input document as well. To set the threshold for your application, use the confidence score from the response.

## Incremental training

With custom models, you need to maintain access to the training dataset to update your classifier with new samples for an existing class, or add new classes. Classifier models now support incremental training where you can reference an existing classifier and append new samples for an existing class or add new classes with samples. Incremental training enables scenarios where data retention is a challenge and the classifier needs to be updated to align with changing business needs. Incremental training is supported with models trained with API version `v4.0 2024-11-30 (GA)`.

### ⓘ Important

Incremental training is only supported with models trained with the same API version. If you're trying to extend a model, use the API version the original model was trained with to extend the model. Incremental training is only supported with API version `v4.0 2024-11-30 (GA)` or later.

Incremental training requires that you provide the original model ID as the `baseClassifierId`. See [incremental training](#) to learn more about how to use incremental training.

## Office document type support

You can now train classifiers to recognize document types in various formats including PDF, images, Word, PowerPoint, and Excel. When assembling your training dataset, you can add documents of any of the supported types. The classifier doesn't require you to explicitly label specific types. As a best practice, ensure your training dataset has at least one sample of each format to improve the overall accuracy of the model.

## Compare custom classification and composed models

A custom classification model can replace [a composed model](#) in some scenarios but there are a few differences to be aware of:

[ ] Expand table

Capability	Custom classifier process	Composed model process
Analyze a single document of unknown type belonging to one of the types trained for extraction model processing.	<ul style="list-style-type: none"><li>• Requires multiple calls.</li><li>• Call the classification model based on the document class. This step allows for a confidence-based check before invoking the extraction model analysis.</li><li>• Invoke the extraction model.</li></ul>	<ul style="list-style-type: none"><li>• Requires a single call to a composed model containing the model corresponding to the input document type.</li></ul>
Analyze a single document of unknown type belonging to several types trained for extraction model processing.	<ul style="list-style-type: none"><li>• Requires multiple calls.</li><li>• Make a call to the classifier that ignores documents not matching a designated type for extraction.</li><li>• Invoke the extraction model.</li></ul>	<ul style="list-style-type: none"><li>• Requires a single call to a composed model. The service selects a custom model within the composed model with the highest match.</li><li>• A composed model can't ignore documents.</li></ul>
Analyze a file containing multiple documents of known or unknown type belonging to one of the types trained for extraction model processing.	<ul style="list-style-type: none"><li>• Requires multiple calls.</li><li>• Call the extraction model for each identified document in the input file.</li><li>• Invoke the extraction model.</li></ul>	<ul style="list-style-type: none"><li>• Requires a single call to a composed model.</li><li>• The composed model invokes the component model once on the first instance of the document.</li></ul>

Capability	Custom classifier process	Composed model process
		<ul style="list-style-type: none"> <li>The remaining documents are ignored.</li> </ul>

## Language support

Classification models can now be trained on documents of different languages. See [supported languages](#) for a complete list.

## Input requirements

Supported file formats:

[\[+\] Expand table](#)

Model	PDF	Image: jpeg/jpg, png, bmp, tiff, heif	Microsoft Office: Word (docx), Excel (xlsx), PowerPoint (pptx)
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓ (not supported in the studio)

- For best results, provide five clear photos or high-quality scans per document type.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 x 50 pixels and 10,000 px x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.

- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8-point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
- For custom extraction model training, the total size of training data is 50 MB for template model and 1G-MB for the neural model.
- For custom classification model training, the total size of training data is 2 GB with a maximum of 25,000 pages.

## Document splitting

When you have more than one document in a file, the classifier can identify the different document types contained within the input file. The classifier response contains the page ranges for each of the identified document types contained within a file. This response can include multiple instances of the same document type.

The `analyze` operation now includes a `splitMode` property that gives you granular control over the splitting behavior.

- To treat the entire input file as a single document for classification set the `splitMode` to `none`. When you do so, the service returns just one class for the entire input file.
- To classify each page of the input file, set the `splitMode` to `perPage`. The service attempts to classify each page as an individual document.
- Set the `splitMode` to `auto` and the service identifies the documents and associated page ranges.

## Best practices

Custom classification models require a minimum of five samples per class to train. If the classes are similar, adding extra training samples improves model accuracy.

The classifier attempts to assign each document to one of the classes, if you expect the model to see document types not in the classes that are part of the training dataset, you should plan to set a threshold on the classification score or add a few representative samples of the document types to an "other" class. Adding an "other" class ensures that unneeded documents don't affect your classifier quality.

# Training a model

Custom classification models are supported by the v4.0 2024-11-30 (GA) API. [Document Intelligence Studio](#) provides a no-code user interface to interactively train a custom classifier. Follow the [how to guide](#) to get started.

When using the REST API, if you organize your documents by folders, you can use the `azureBlobSource` property of the request to train a classification model.

JSON

```
https://{{endpoint}}/documentintelligence/documentClassifiers:build?api-version=2024-11-30

{
  "classifierId": "demo2.1",
  "description": "",
  "docTypes": {
    "car-maint": {
      "azureBlobSource": {
        "containerUrl": "SAS URL to container",
        "prefix": "sample1/car-maint/"
      }
    },
    "cc-auth": {
      "azureBlobSource": {
        "containerUrl": "SAS URL to container",
        "prefix": "sample1/cc-auth/"
      }
    },
    "deed-of-trust": {
      "azureBlobSource": {
        "containerUrl": "SAS URL to container",
        "prefix": "sample1/deed-of-trust/"
      }
    }
  }
}
```

Alternatively, if you have a flat list of files or only plan to use a few select files within each folder to train the model, you can use the `azureBlobFileListSource` property to train the model. This step requires a `file list` in [JSON Lines](#) format. For each class, add a new file with a list of files to be submitted for training.

JSON

```
{  
  "classifierId": "demo2",  
  "description": "",  
  "docTypes": {  
    "car-maint": {  
      "azureBlobFileListSource": {  
        "containerUrl": "SAS URL to container",  
        "fileList": "{path to dataset root}/car-maint.jsonl"  
      }  
    },  
    "cc-auth": {  
      "azureBlobFileListSource": {  
        "containerUrl": "SAS URL to container",  
        "fileList": "{path to dataset root}/cc-auth.jsonl"  
      }  
    },  
    "deed-of-trust": {  
      "azureBlobFileListSource": {  
        "containerUrl": "SAS URL to container",  
        "fileList": "{path to dataset root}/deed-of-trust.jsonl"  
      }  
    }  
  }  
}
```

As an example, the file list `car-maint.jsonl` contains the following files.

JSON

```
{"file":"classifier/car-maint/Commercial Motor Vehicle - Adatum.pdf"}  
{"file":"classifier/car-maint/Commercial Motor Vehicle - Fincher.pdf"}  
{"file":"classifier/car-maint/Commercial Motor Vehicle - Lamna.pdf"}  
{"file":"classifier/car-maint/Commercial Motor Vehicle - Liberty.pdf"}  
{"file":"classifier/car-maint/Commercial Motor Vehicle - Trey.pdf"}
```

## Overwriting a model

### ⓘ Note

The v4.0 2024-11-30 (GA) custom classification model supports overwriting a model in-place.

You can now update the custom classification in-place. Directly overwriting the model would lose you the ability to compare model quality before deciding to replace the existing model. Model overwriting is allowed when the `allowOverwrite` property is

explicitly specified in the request body. It's impossible to recover the overwritten, original model once this action is performed.

JSON

```
{  
  "classifierId": "existingClassifierName",  
  "allowOverwrite": true, // Default=false  
  ...  
}
```

## Copy a model

### ⓘ Note

The custom classification v4.0 2024-11-30 (GA) model supports copying a model to and from any of the following regions:

- East US
- West US2
- West Europe

Use the [REST API](#) or [Document Intelligence Studio](#) ↗ to copy a model to another region.

## Generate Copy authorization request

The following HTTP request gets copy authorization from your target resource. You need to enter the endpoint and key of your target resource as headers.

HTTP

```
POST  
https://myendpoint.cognitiveservices.azure.com/documentintelligence/document  
Classifiers:authorizeCopy?api-version=2024-11-30  
Ocp-Apim-Subscription-Key: {<your-key>}
```

Request body

JSON

```
{  
  "classifierId": "targetClassifier",  
  "description": "Target classifier description"  
}
```

You receive a `200` response code with response body that contains the JSON payload required to initiate the copy.

JSON

```
{  
  "targetResourceId":  
  "/subscriptions/targetSub/resourceGroups/targetRG/providers/Microsoft.CognitiveServices/accounts/targetService",  
  "targetResourceRegion": "targetResourceRegion",  
  "targetClassifierId": "targetClassifier",  
  "targetClassifierLocation":  
  "https://targetEndpoint.cognitiveservices.azure.com/documentintelligence/documentClassifiers/targetClassifier",  
  "accessToken": "accessToken",  
  "expirationDateTime": "timestamp"  
}
```

## Start Copy operation

The following HTTP request starts the copy operation on the source resource. You need to enter the endpoint and key of your source resource as the url and header. Notice that the request URL contains the classifier ID of the source classifier you want to copy.

HTTP

```
POST  
{endpoint}/documentintelligence/documentClassifiers/{classifierId}:copyTo?  
api-version=2024-11-30  
Ocp-Apim-Subscription-Key: {<your-key>}
```

The body of your request is the response from the previous step.

JSON

```
{  
  "targetResourceId":  
  "/subscriptions/targetSub/resourceGroups/targetRG/providers/Microsoft.CognitiveServices/accounts/targetService",  
  "targetResourceRegion": "targetResourceRegion",  
  "targetClassifierId": "targetClassifier",  
  "targetClassifierLocation":  
  "https://targetEndpoint.cognitiveservices.azure.com/documentintelligence/documentClassifiers/targetClassifier",  
  "accessToken": "accessToken",  
  "expirationDateTime": "timestamp"  
}
```

```
"https://targetEndpoint.cognitiveservices.azure.com/documentintelligence/documentClassifiers/targetClassifier",
  "accessToken": "accessToken",
  "expirationDateTime": "timestamp"
}
```

## Model response

Analyze an input file with the document classification model.

JSON

```
https://{{endpoint}}/documentintelligence/documentClassifiers/{{classifier}}:analyze?api-version=2024-11-30
```

The `v4.0 2024-11-30 (GA)` API enables you to specify pages to analyze from the input document using the `pages` query parameter in the request.

The response contains the identified documents with the associated page ranges in the `documents` section of the response.

JSON

```
{
  ...
  "documents": [
    {
      "docType": "formA",
      "boundingRegions": [
        { "pageNumber": 1, "polygon": [...] },
        { "pageNumber": 2, "polygon": [...] }
      ],
      "confidence": 0.97,
      "spans": []
    },
    {
      "docType": "formB",
      "boundingRegions": [
        { "pageNumber": 3, "polygon": [...] }
      ],
      "confidence": 0.97,
      "spans": []
    },
    ...
  ]
}
```

# Next steps

Learn to create custom classification models:

[Build a custom classification model](#)

[Custom models overview](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Best practices: generating labeled datasets

Article • 12/11/2024

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)

Custom models (template and neural) require a labeled dataset of at least five documents to train a model. The quality of the labeled dataset affects the accuracy of the trained model. This guide helps you learn more about generating a model with high accuracy by assembling a diverse dataset and provides best practices for labeling your documents.

## Understand the components of a labeled dataset

A labeled dataset consists of several files:

- You provide a set of sample documents (typically PDFs or images). A minimum of five documents is needed to train a model.
- Additionally, the labeling process generates the following files:
  - A `fields.json` file is created when the first field is added. There's one `fields.json` file for the entire training dataset, the field list contains the field name and associated sub fields and types.
  - The Studio runs each of the documents through the [Layout API](#). The layout response for each of the sample files in the dataset is added as `{file}.ocr.json`. The layout response is used to generate the field labels when a specific span of text is labeled.
  - A `{file}.labels.json` file is created or updated when a field is labeled in a document. The label file contains the spans of text and associated polygons from the layout output for each span of text the user adds as a value for a specific field.

## Video: Custom label tips and pointers

- The following video is the first of two presentations intended to help you build custom models with higher accuracy (The second presentation examines [Best practices for labeling documents](#)).

- We explore how to create a balanced data set and select the right documents to label. This process sets you on the path to higher quality models.

<https://www.microsoft.com/en-us/videoplayer/embed/RWWHru?postJslIMsg=true>

## Create a balanced dataset

Before you start labeling, it's a good idea to look at a few different samples of the document to identify which samples you want to use in your labeled dataset. A balanced dataset represents all the typical variations you would expect to see for the document. Creating a balanced dataset results in a model with the highest possible accuracy. A few examples to consider are:

- **Document formats:** If you expect to analyze both digital and scanned documents, add a few examples of each type to the training dataset.
- **Variations (template model):** Consider splitting the dataset into folders and train a model for each of variation. Any variations that include either structure or layout should be split into different models. You can then compose the individual models into a single [composed model](#).
- **Variations (Neural models):** When your dataset has a manageable set of variations, about 15 or fewer, create a single dataset with a few samples of each of the different variations to train a single model. If the number of template variations is larger than 15, you train multiple models and [compose](#) them together.
- **Tables:** For documents containing tables with a variable number of rows, ensure that the training dataset also represents documents with different numbers of rows.
- **Multi page tables:** When tables span multiple pages, label a single table. Add documents to the training dataset with the expected variations represented—documents with the table on a single page only and documents with the table spanning two or more pages with all the rows labeled.
- **Optional fields:** If your dataset contains documents with optional fields, validate that the training dataset has a few documents with the options represented.

## Start by identifying the fields

Take the time to identify each of the fields you plan to label in the dataset. Pay attention to optional fields. Define the fields with the labels that best match the supported types.

Use the following guidelines to define the fields:

- For custom neural models, use semantically relevant names for fields. For example, if the value being extracted is `Effective Date`, name it `effective_date` or `EffectiveDate` not a generic name like `date1`.
- Ideally, name your fields with Pascal or camel case.
- If a value is part of a visually repeating structure and you only need a single value, label it as a table and extract the required value during post-processing.
- For tabular fields spanning multiple pages, define and label the fields as a single table.

 **Note**

Custom neural models share the same labeling format and strategy as custom template models. Currently custom neural models only support a subset of the field types supported by custom template models.

## Model capabilities

Custom neural models currently only support key-value pairs, structured fields (tables), and selection marks.

 Expand table

Model type	Form fields	Selection marks	Tabular fields	Signature	Region	Overlapping fields
Custom neural	✓ Supported	✓ Supported	✓ Supported	Unsupported	✓ Supported <sup>1</sup>	✓ Supported <sup>2</sup>
Custom template	✓ Supported	✓ Supported	✓ Supported	✓ Supported	✓ Supported	Unsupported

<sup>1</sup> Region labeling implementation differs between template and neural models. For template models, the training process injects synthetic data at training time if no text is found in the region labeled. With neural models, no synthetic text is injected and the recognized text is used as is.

<sup>2</sup> Overlapping fields are supported starting with the API version `v4.0 2024-11-30 (GA)`. Overlapping fields have some limits. For more information, see [overlapping fields](#).

## Tabular fields

Tabular fields (tables) are supported with custom neural models with API version v4.0 2024-11-30 (GA). Models trained with API version 2022-06-30-preview or later will accept tabular field labels and documents analyzed with the model with API version 2022-06-30-preview or later will produce tabular fields in the output within the `documents` section of the result in the `analyzeResult` object.

Tabular fields support **cross page tables** by default. To label a table that spans multiple pages, label each row of the table across the different pages in the single table. As a best practice, ensure that your dataset contains a few samples of the expected variations. For example, include both samples where an entire table is on a single page and samples of a table spanning two or more pages.

Tabular fields are also useful when extracting repeating information within a document that isn't recognized as a table. For example, a repeating section of work experiences in a resume can be labeled and extracted as a tabular field.

#### Note

Table field when labeled are extracted as part of the `documents` section of the response. The response also contains a `tables` section which contains the tables extracted from the document by the layout model. If you have labeled a field as a table, look for the field in the `documents` section of the response.

## Labeling guidelines

- **Labeling values is required.** Don't include the surrounding text. For example when labeling a checkbox, name the field to indicate the check box selection for example `selectionYes` and `selectionNo` rather than labeling the yes or no text in the document.
- **Don't provide interleaving field values.** The value of words and/or regions of one field must be a consecutive sequence in natural reading order.
- **Consistent labeling.** If a value appears in multiple contexts within the document, consistently pick the same context across documents to label the value.
- **Visually repeating data.** Tables support visually repeating groups of information not just explicit tables. Explicit tables are identified in `tables` section of the analyzed documents as part of the layout output and don't need to be labeled as tables. Only label a table field if the information is visually repeating and not identified as a table as part of the layout response. An example would be the repeating work experience section of a resume.

- **Region labeling (custom template).** Labeling specific regions allows you to define a value when none exists. If the value is optional, ensure that you leave a few sample documents with the region not labeled. When labeling regions, don't include the surrounding text with the label.
- **Overlapping fields (custom neural).** Label the field overlaps using region labeling. Ensure that you have at least one sample that describes how the fields can overlap in your training dataset.

## Next steps

- Train a custom model:

[How to train a model](#)

- View the REST APIs:

[Document Intelligence API v4.0:2024-11-30 \(GA\)](#)

[Document Intelligence API v3.1:2023-07-31 \(GA\)](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Tips for building labeled datasets

Article • 12/11/2024

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)

This article highlights the best methods for labeling custom model datasets in the Document Intelligence Studio. Labeling documents can be time consuming when you have a large number of labels, long documents, or documents with varying structure. These tips should help you label documents more efficiently.

## Video: Custom labels best practices

- The following video is the second of two presentations intended to help you build custom models with higher accuracy (the first presentation explores [How to create a balanced data set](#)).
- We examine best practices for labeling your selected documents. With semantically relevant and consistent labeling, you should see an improvement in model performance.

<https://www.microsoft.com/en-us/videoplayer/embed/RE5fZKB?postJs||Msg=true> ↗

## Search

The Studio now includes a search box for instances when you know you need to find specific words to label, but just don't know where to locate them in the document. Simply search for the word or phrase and navigate to the specific section in the document to label the occurrence.

## Auto label tables

Tables can be challenging to label, when they have many rows or dense text. If the layout table extracts the result you need, you should just use that result and skip the labeling process. In instances where the layout table isn't exactly what you need, you can start with generating the table field from the values layout extracts. Start by selecting the table icon on the page and select on the auto label button. You can then edit the values as needed. Auto label currently only supports single page tables.

## Shift select

When labeling a large span of text, rather than mark each word in the span, hold down the shift key as you're selecting the words to speed up labeling and ensure you don't miss any words in the span of text.

## Region labeling

A second option for labeling larger spans of text is to use region labeling. When region labeling is used, the `OCR` results are populated in the value at training time. The difference between the shift select and region labeling is only in the visual feedback the shift labeling approach provides.

## Label overlapping fields

Overlapping fields are supported for fields and table cells. If you expect your analyze results to contain overlapping fields, you should add at least one sample to the training dataset with the specific field overlaps labeled. To label an overlapping field, use the region labeling feature to select the regions for each field. Both complete and partial overlaps are supported. Any single word in the document can only be labeled for two fields.

## Field subtypes

When creating a field, select the right subtype to minimize post processing, for instance select the `dmy` option for dates to extract the values in a `dd-mm-yyyy` format.

## Next steps

- Learn more about custom labeling:

[Custom labels](#)

- Learn more about custom template models:

[Custom models](#)

---

## Feedback

Was this page helpful?

Yes

No



# Document Intelligence custom model lifecycle

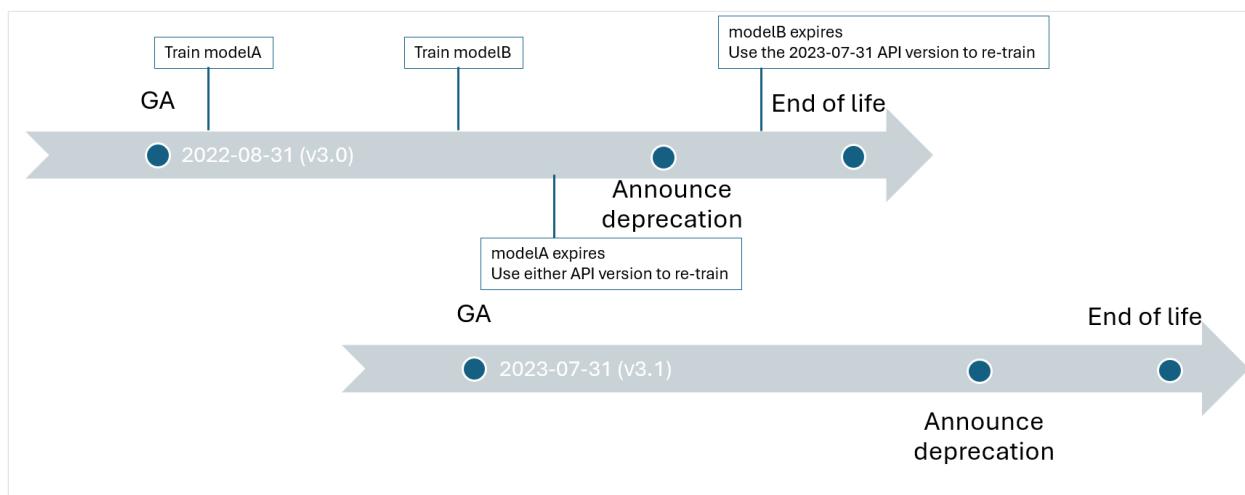
Article • 12/11/2024

This content applies to: ✓ v4.0 (GA) ✓ v3.1 (GA)

With the v3.1 (GA) and later APIs, custom models introduce a `expirationDateTime` property that is set for each model trained with the 3.1 API or later. Custom models are dependent on the API version of the Layout API version and the API version of the model build operation. For best results, continue to use the API version the model was trained with for all analyze requests. The guidance applies to all Document Intelligence custom models including extraction and classification models.

## Models trained with GA API version

With the v3.1 API, custom models introduce a new model expiration property. The model expiration is set to two years from the date the model is built for all requests that use a GA API to build a model. To continue to use the model past the expiration date, you need to train the model with a current GA API version. The API version can be the one that the model was originally trained with or a later API version. The following figure illustrates the options when you need to retrain an expiring or expired model.



## Models trained with preview API version

For build requests, using a preview API version, the expiration date is set to two years from the date the model is built. Models trained with a preview API shouldn't be used in production and should be retrained once the corresponding GA API version is available. Compatibility between preview API versions and GA API versions isn't always

maintained. Models trained with a preview API version are no longer usable the corresponding GA API is available.

## Viewing model expiration date

The GET model API returns the model details including the `expirationDateTime` property.

rest

```
GET /documentModels/{customModelId}?api-version={apiVersion}
{
    "modelId": "{customModelId}",
    "description": "{customModelDescription}",
    "createdDateTime": "2021-09-24T12:54:35Z",
    "expirationDateTime": "2023-01-01T00:00:00Z",
    "apiVersion": "2023-07-31",
    "docTypes": { ... }
}
```

## Retrain a model

To retrain a model with a more recent API version, ensure that the layout results for the documents in your training dataset correspond to the API version of the build model request. For instance, if you plan to build the model with the `v3.1:2023-07-31` API version, the corresponding `*.ocr.json` files in your training dataset should also be generated with the `v3.1:2023-07-31` API version. The `ocr.json` files are generated by running layout on your training dataset. To validate the version of the layout results, check the `apiVersion` property in the `analyzeResult` of the `ocr.json` documents.

## Next steps

Learn to create and compose custom models:

[Build a custom model](#)

[Compose custom models](#)

## Feedback

Was this page helpful?

 Yes

 No



# Document Intelligence add-on capabilities

Article • 02/27/2025

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA) ::moniker-end

## Capabilities

Document Intelligence supports more sophisticated and modular analysis capabilities. Use the add-on features to extend the results to include more features extracted from your documents. Some add-on features incur an extra cost. These optional features can be enabled and disabled depending on the scenario of the document extraction. To enable a feature, add the associated feature name to the `features` query string property. You can enable more than one add-on feature on a request by providing a comma-separated list of features. The following add-on capabilities are available for `2023-07-31 (GA)` and later releases.

- [ocrHighResolution](#)
- [formulas](#)
- [styleFont](#)
- [barcodes](#)
- [languages](#)
- [Searchable PDF support](#)
- [queryFields](#)
- [keyValuePairs](#)

 **Note**

Not all models or Microsoft Office file types support add-on capabilities. For more information, see [model data extraction](#).

## Version availability

 Expand table

Add-on Capability	Add-On/Free	2024-11-30 (GA)	2023-07-31 (GA)	2022-08-31 (GA)	v2.1 (GA)
Barcode extraction	Free	✓	✓	n/a	n/a
Language detection	Free	✓	✓	n/a	n/a
Key value pairs	Free	✓	n/a	n/a	n/a
Searchable PDF	Free	✓	n/a	n/a	n/a
Font property extraction	Add-On	✓	✓	n/a	n/a
Formula extraction	Add-On	✓	✓	n/a	n/a
High resolution extraction	Add-On	✓	✓	n/a	n/a
Query fields	Add-On	✓	n/a	n/a	n/a

\* Add-On - Query fields are priced differently than the other add-on features. See [pricing ↗](#) for details.

\*\* Add-On - Searchable pdf is available only with Read model as an add-on feature.

## Supported file formats

- `PDF`
- Images: `JPEG / JPG`, `PNG`, `BMP`, `TIFF`, `HEIF`

\* Microsoft Office files are currently not supported.

## High resolution extraction

The task of recognizing small text from large-size documents, like engineering drawings, is a challenge. Often the text is mixed with other graphical elements and has varying fonts, sizes, and orientations. Moreover, the text can be broken into separate parts or connected with other symbols. Document Intelligence now supports extracting content from these types of documents with the `ocr.highResolution` capability. You get improved quality of content extraction from A1/A2/A3 documents by enabling this add-on capability.

REST API

Bash

```
{your-resource-  
endpoint}.cognitiveservices.azure.com/documentintelligence/documentModels/preb  
uilt-layout:analyze?api-version=2024-11-30&features=ocrHighResolution
```

## Formula extraction

The `ocr.formula` capability extracts all identified formulas, such as mathematical equations, in the `formulas` collection as a top level object under `content`. Inside `content`, detected formulas are represented as `:formula:`. Each entry in this collection represents a formula that includes the formula type as `inline` or `display`, and its LaTeX representation as `value` along with its `polygon` coordinates. Initially, formulas appear at the end of each page.

 **Note**

The `confidence` score is hard-coded.

REST API

Bash

```
{your-resource-  
endpoint}.cognitiveservices.azure.com/documentintelligence/documentModels/preb  
uilt-layout:analyze?api-version=2024-11-30&features=formulas
```

## Font property extraction

The `ocr.font` capability extracts all font properties of text extracted in the `styles` collection as a top-level object under `content`. Each style object specifies a single font property, the text span it applies to, and its corresponding confidence score. The existing `style` property is extended with more font properties such as `similarFontFamily` for the font of the text, `fontStyle` for styles such as italic and normal, `fontWeight` for bold or normal, `color` for color of the text, and `backgroundColor` for color of the text bounding box.

REST API

Bash

```
{your-resource-  
endpoint}.cognitiveservices.azure.com/documentintelligence/documentModels/preb  
uilt-layout:analyze?api-version=2024-11-30&features=styleFont
```

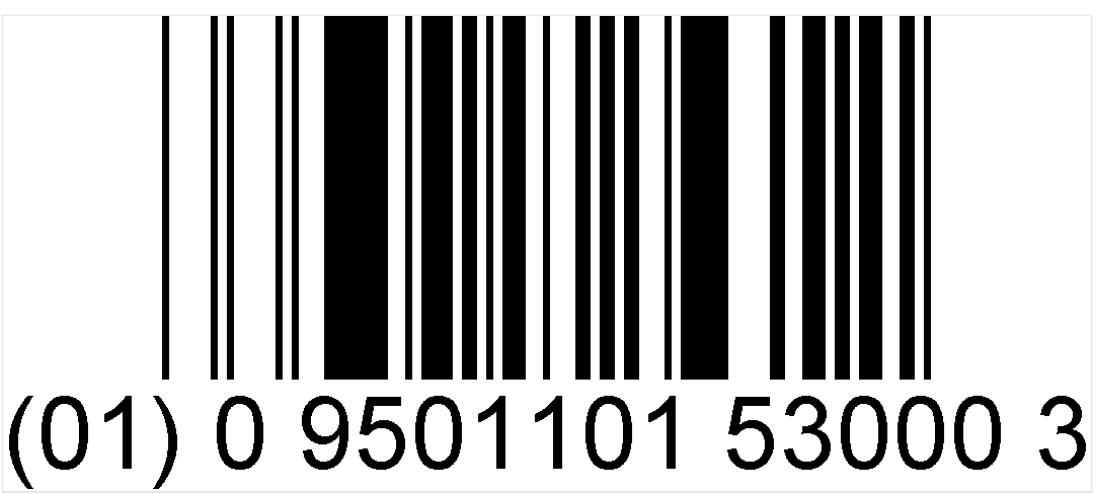
## Barcode property extraction

The `ocr.barcode` capability extracts all identified barcodes in the `barcodes` collection as a top level object under `content`. Inside the `content`, detected barcodes are represented as `:barcode:`. Each entry in this collection represents a barcode and includes the barcode type as `kind` and the embedded barcode content as `value` along with its `polygon` coordinates. Initially, barcodes appear at the end of each page. The `confidence` is hard-coded for as 1.

## Supported barcode types

 Expand table

Barcode Type	Example
QR Code	
Code 39	 0123456789ABC
Code 93	 ABC-1234-/+
Code 128	 0123456789abcdefg

Barcode Type	Example
UPC (UPC-A & UPC-E)	 <p>A standard UPC-A barcode with vertical bars of varying widths. Below it, the numbers 0 12345 67890 5 are displayed.</p>
PDF417	 <p>A dense, multi-level barcode used for data storage.</p>
EAN-8	 <p>A barcode with vertical bars. Below it, the numbers 9031 1017 are displayed.</p>
EAN-13	 <p>A barcode with vertical bars. Below it, the numbers 9 780201 379624 are displayed.</p>
Codabar	 <p>A barcode with vertical bars. Below it, the numbers 0123456789 are displayed.</p>
DataBar	 <p>A barcode with vertical bars. Below it, the numbers (01) 0 9501101 53000 3 are displayed.</p>

Barcode Type	Example
Databar Expanded	 (01)1234567890123-ABCabc
ITF	 0 5 0 1 2 3 4 5 6 7 8 9 0 0
Data Matrix	

REST API
Bash
{your-resource-endpoint}.cognitiveservices.azure.com/documentintelligence/documentModels/prebuilt-layout:analyze?api-version=2024-11-30&features=barcodes

## Language detection

Adding the `languages` feature to the `analyzeResult` request predicts the detected primary language for each text line along with the `confidence` in the `languages` collection under `analyzeResult`.

REST API
Bash

```
{your-resource-  
endpoint}.cognitiveservices.azure.com/documentintelligence/documentModels/preb  
uilt-layout:analyze?api-version=2024-11-30&features=languages
```

## Searchable PDF

The searchable PDF capability enables you to convert an analog PDF, such as scanned-image PDF files, to a PDF with embedded text. The embedded text enables deep text search within the PDF's extracted content by overlaying the detected text entities on top of the image files.

### *i* Important

- Currently, only the Read model `prebuilt-read` supports the searchable PDF capability. When using this feature, specify the `modelId` as `prebuilt-read`.
- Searchable PDF is included with the `2024-11-30 (GA)` `prebuilt-read` model with no usage cost for general PDF consumption.

## Use searchable PDF

To use searchable PDF, make a `POST` request using the `Analyze` operation and specify the output format as `pdf`:

Bash

```
POST /documentModels/prebuilt-read:analyze?output=pdf  
{...}  
202
```

Once the `Analyze` operation is complete, make a `GET` request to retrieve the `Analyze` operation results.

Upon successful completion, the PDF can be retrieved and downloaded as `application/pdf`. This operation allows direct downloading of the embedded text form of PDF instead of Base64-encoded JSON.

Bash

```
// Monitor the operation until completion.
```

```
GET /documentModels/prebuilt-read/analyzeResults/{resultId}  
200  
{...}  
  
// Upon successful completion, retrieve the PDF as application/pdf.  
GET /documentModels/prebuilt-read/analyzeResults/{resultId}/pdf  
200 OK  
Content-Type: application/pdf
```

## Key-value Pairs

In earlier API versions, the `prebuilt-document` model extracted key-value pairs from forms and documents. With the addition of the `keyValuePairs` feature to prebuilt-layout, the layout model now produces the same results.

Key-value pairs are specific spans within the document that identify a label or key and its associated response or value. In a structured form, these pairs could be the label and the value the user entered for that field. In an unstructured document, they could be the date a contract was executed on based on the text in a paragraph. The AI model is trained to extract identifiable keys and values based on a wide variety of document types, formats, and structures.

Keys can also exist in isolation when the model detects that a key exists, with no associated value or when processing optional fields. For example, a middle name field can be left blank on a form in some instances. Key-value pairs are spans of text contained in the document. For documents where the same value is described in different ways, for example, customer/user, the associated key is either customer or user (based on context).

## REST API

Bash

```
{your-resource-  
endpoint}.cognitiveservices.azure.com/documentintelligence/documentModels/prebuilt  
-layout:analyze?api-version=2024-11-30&features=keyValuePairs
```

## Query Fields

Query fields are an add-on capability to extend the schema extracted from any prebuilt model or define a specific key name when the key name is variable. To use query fields, set the features to `queryFields` and provide a comma-separated list of field names in the `queryFields` property.

- Document Intelligence now supports query field extractions. With query field extraction, you can add fields to the extraction process using a query request without the need for added training.
- Use query fields when you need to extend the schema of a prebuilt or custom model or need to extract a few fields with the output of layout.
- Query fields are a premium add-on capability. For best results, define the fields you want to extract using camel case or Pascal case field names for multi-word field names.
- Query fields support a maximum of 20 fields per request. If the document contains a value for the field, the field and value are returned.
- This release has a new implementation of the query fields capability that is priced lower than the earlier implementation and should be validated.

**! Note**

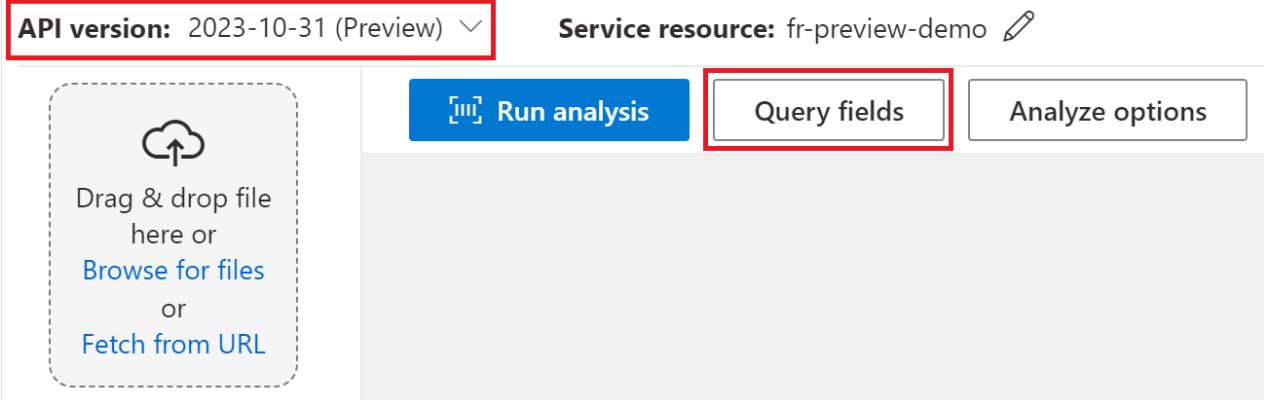
Document Intelligence Studio query field extraction is currently available with the Layout and Prebuilt models 2024-11-30 (GA) API except for US tax models W2, 1098, and 1099.

## Query field extraction

For query field extraction, specify the fields you want to extract and Document Intelligence analyzes the document accordingly. Here's an example:

- If you're processing a contract in the [Document Intelligence Studio](#), use the 2024-11-30 (GA) version:

### Layout



- You can pass a list of field labels like `Party1`, `Party2`, `TermsOfUse`, `PaymentTerms`, `PaymentDate`, and `TermEndDate` as part of the `analyze document` request.

The screenshot shows the Azure AI Document Intelligence Studio interface. In the background, there's a file upload area with options to "Drag & drop file here or Browse for files or Fetch from URL". Below it are three sample document thumbnails labeled "layout-news.png", "layout-report.png", and "layout-form.png". At the top, the API version is set to "2023-10-31 (Preview)" and the service resource is "fr-preview-dem". A "Run analysis" button is visible. In the foreground, a modal window titled "Query fields" is open. It contains a list of fields with checkboxes: "Party1" (checked), "Party2" (checked), "TermsOfUse" (checked), "PaymentTerms" (checked), "PaymentDate" (checked), and "TermEndDate" (checked). Each field has edit and delete icons to its right. A note at the bottom says "Maximum of 6/20 fields". A checkbox for acknowledging usage is present, along with "Delete all", "Save", and "Cancel" buttons.

- Document Intelligence is able to analyze and extract the field data and return the values in a structured JSON output.
- In addition to the query fields, the response includes text, tables, selection marks, and other relevant data.

The screenshot shows the REST API documentation for Document Intelligence. On the left, a sidebar lists "REST API", "Bash", and "Python". The main content area is titled "Bash" and contains a code snippet:

```
{your-resource-endpoint}.cognitiveservices.azure.com/documentintelligence/documentModels/prebuilt-layout:analyze?api-version=2024-11-30&features=queryFields&queryFields=TERMS
```

## Next steps

Learn more: [Read model](#) [Layout model](#)

SDK samples: [python](#)

Find more samples: [Add-on capabilities](#)



# Document Intelligence query field extraction

Article • 02/13/2025

Document Intelligence now supports query field to extend the schema of any prebuilt model to extract the specific fields you need. Query fields can also be added to layout to extract fields in addition to structure from forms or documents.

## ! Note

Document Intelligence Studio query field extraction is currently available with layout and prebuilt models, excluding the UX.Tax prebuilt models.

## Query fields or key value pairs

Query fields and key value pairs perform similar functions, there are a few distinctions to be aware of when deciding which feature to choose.

- Key value pairs are only available with layout and invoice models. If you're looking to extend the schema for a prebuilt model, use query fields.
- You don't know the specific fields to be extracted, or the number of fields is large (greater than 20), key value pairs might be a better solution.
- Key-value pairs extract the keys and values as they exist in the form or document, you need to plan for any key variations. For example, keys `First Name` or `Given Name`. With query fields, you define the key and the model only extracts the corresponding value.
- Use query fields when the value you require can't be described as a key value pair in the document. For example, the agreement date of a contract.

For query field extraction, specify the fields you want to extract and Document Intelligence analyzes the document accordingly. Here's an example:

- If you're processing a contract in the [Document Intelligence Studio](#), use the **2024-11-30 (GA)**, API version:

# Layout

API version: 2023-10-31 (Preview) Service resource: fr-preview-demo

Run analysis

Query fields

Analyze options

Drag & drop file here or  
Browse for files or  
Fetch from URL

- You can pass a list of field labels like `Party1`, `Party2`, `TermsOfUse`, `PaymentTerms`, `PaymentDate`, and `TermEndDate` as part of the `AnalyzeDocument` request.

Query fields

Add fields you want to extract to the list and select the ones to be used in Analyze operations. Press Save to update all edits in the list. [Learn more about query fields](#).

Party1

Party2

TermsOfUse

PaymentTerms

PaymentDate

TermEndDate

Maximum of 6/20 fields

I acknowledge that using Query Fields will incur usage to my account. See [pricing](#).

Delete all

Save

Cancel

- In addition to the query fields, the response includes the model output. For a list of features or schema extracted by each model, see [model analysis features](#).

## Query fields REST API request

Use the query fields feature with the [prebuilt layout](#) model, and add fields to the extraction process without having to train a custom model:

HTTP

```
POST https://{{endpoint}}/documentintelligence/documentModels/prebuilt-
layout:analyze?api-version=2024-11-
30&features=queryFields&queryFields=OurReference,BookingDate HTTP/1.1
Host: *.cognitiveservices.azure.com
Content-Type: application/json
Ocp-Apim-Subscription-Key:

{
  "urlSource": "https://raw.githubusercontent.com/Azure-Samples/cognitive-
  services-REST-api-samples/master/curl/form-recognizer/rest-api/layout.png"
}
```

## Next steps

[Try the Document Intelligence Studio quickstart](#)

[Learn about other add-on capabilities](#)

# Interpret and improve accuracy and confidence scores

Article • 03/03/2025

A confidence score indicates probability by measuring the degree of statistical certainty that the extracted result is detected correctly. The estimated accuracy is calculated by running a few different combinations of the training data to predict the labeled values. In this article, learn to interpret accuracy and confidence scores and best practices for using those scores to improve accuracy and confidence results.

## Confidence scores

### Note

- Field level confidence includes word confidence scores with **2024-11-30 (GA)** API version for **custom models**.
- Confidence scores for tables, table rows, and table cells are available starting with the **2024-11-30 (GA)** API version for **custom models**.

Document Intelligence analysis results return an estimated confidence for predicted words, key-value pairs, selection marks, regions, and signatures. Currently, not all document fields return a confidence score.

Field confidence indicates an estimated probability between 0 and 1 that the prediction is correct. For example, a confidence value of 0.95 (95%) indicates that the prediction is likely correct 19 out of 20 times. For scenarios where accuracy is critical, confidence can be used to determine whether to automatically accept the prediction or flag it for human review.

Document Intelligence Studio  
Analyzed invoice prebuilt-invoice model

----- Analyzing invoice -----

Analyzed document has doc type prebuilt:invoice with confidence : 1.00

Vendor Name: CONTOSO LTD., confidence: 0.96

Vendor address: 123 456th St New York, NY, 10001, confidence: 0.95

Customer Name: MICROSOFT CORPORATION, confidence: 0.95

Customer Address Recipient: Microsoft Corp, confidence: 0.96

Invoice ID: INV-100, confidence: 0.98

Invoice Date: 2019-11-15, confidence: 0.98

Invoice Total: 110.00, confidence: 0.97

Invoice Items:

Unit Price: 1.000000, confidence: 0.68

Description: Test for 23 fields, confidence: 0.90s

Quantity: 1.000000, confidence: 0.88

## Improve confidence scores

After an analysis operation, review the JSON output. Examine the `confidence` values for each key/value result under the `pageResults` node. You should also look at the confidence score in the `readResults` node, which corresponds to the text-read operation. The confidence of the read results doesn't affect the confidence of the key/value extraction results, so you should check both. Here are some tips:

- If the confidence score for the `readResults` object is low, improve the quality of your input documents.
- If the confidence score for the `pageResults` object is low, ensure that the documents you're analyzing are of the same type.
- Consider incorporating human review into your workflows.
- Use forms that have different values in each field.
- For custom models, use a larger set of training documents. A larger training set teaches your model to recognize fields with greater accuracy.

## Accuracy scores for custom models

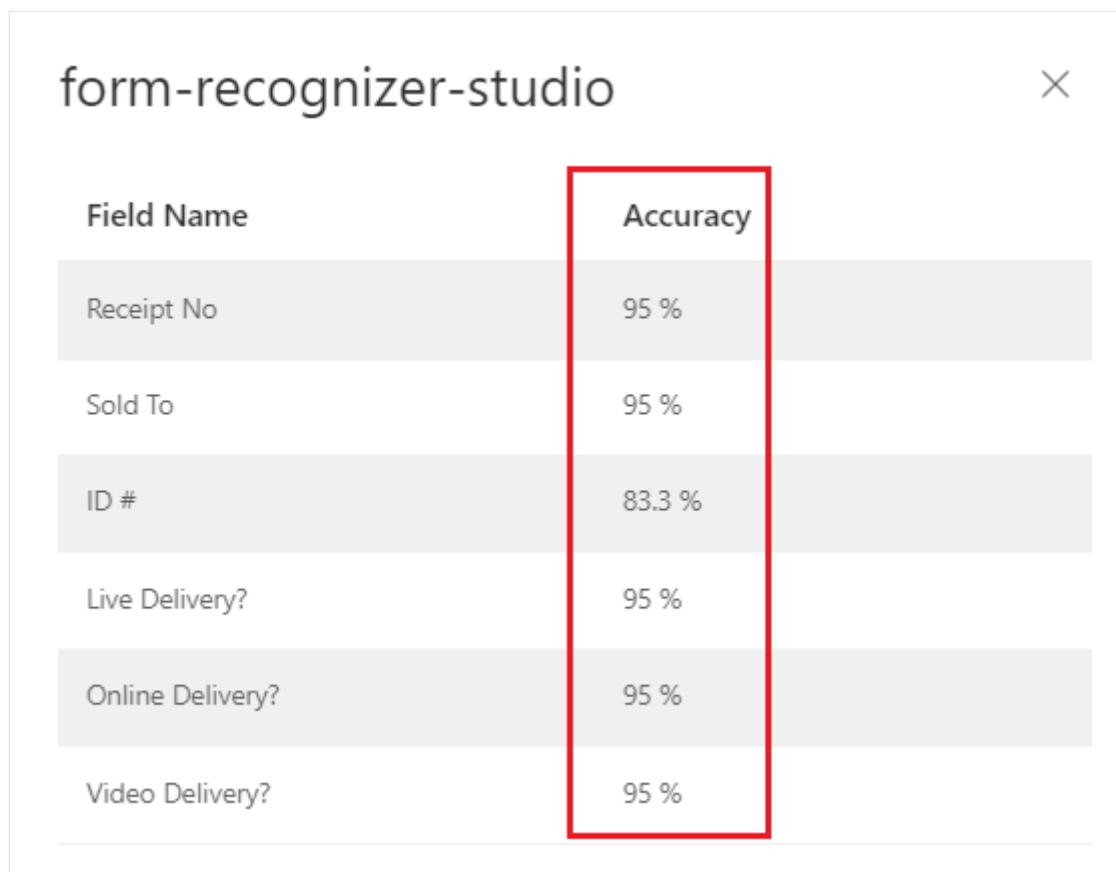
## (!) Note

- Custom neural and generative models don't provide accuracy scores during training.

The output of a `build` (v3.0 and onward) or `train` (v2.1) custom model operation includes the estimated accuracy score. This score represents the model's ability to accurately predict the labeled value on a visually similar document. Accuracy is measured within a percentage value range from 0% (low) to 100% (high). It's best to target a score of 80% or higher. For more sensitive cases, like financial or medical records, we recommend a score of close to 100%. You can also add a human review stage to validate for more critical automation workflows.

### Document Intelligence Studio

#### Trained custom model (invoice)



## Interpret accuracy and confidence scores for custom models

Custom template models generate an estimated accuracy score when trained. Documents analyzed with a custom model produce a confidence score for extracted

fields. When interpreting the confidence score from a custom model, you should consider all the confidence scores returned from the model. Let's start with a list of all the confidence scores.

- **Document type confidence score:** The document type confidence is an indicator of closely the analyzed document resembles documents in the training dataset. When the document type confidence is low, it's indicative of template or structural variations in the analyzed document. To improve the document type confidence, label a document with that specific variation and add it to your training dataset. Once the model is retrained, it should be better equipped to handle that class of variations.
- **Field level confidence:** Each labeled field extracted has an associated confidence score. This score reflects the model's confidence on the position of the value extracted. While evaluating confidence scores, you should also look at the underlying extraction confidence to generate a comprehensive confidence for the extracted result. Evaluate the `OCR` results for text extraction or selection marks depending on the field type to generate a composite confidence score for the field.
- **Word confidence score** Each word extracted within the document has an associated confidence score. The score represents the confidence of the transcription. The pages array contains an array of words and each word has an associated span and confidence score. Spans from the custom field extracted values match the spans of the extracted words.
- **Selection mark confidence score:** The pages array also contains an array of selection marks. Each selection mark has a confidence score representing the confidence of the selection mark and selection state detection. When a labeled field has a selection mark, the custom field selection combined with the selection mark confidence is an accurate representation of overall confidence accuracy.

The following table demonstrates how to interpret both the accuracy and confidence scores to measure your custom model's performance.

[\[+\] Expand table](#)

Accuracy	Confidence	Result
High	High	<ul style="list-style-type: none"><li>• The model is performing well with the labeled keys and document formats.</li><li>• You have a balanced training dataset.</li></ul>
High	Low	<ul style="list-style-type: none"><li>• The analyzed document appears different from the training dataset.</li><li>• The model would benefit from retraining with at least five more labeled</li></ul>

Accuracy	Confidence	Result
		documents. • These results could also indicate a format variation between the training dataset and the analyzed document. Consider adding a new model.
Low	High	• This result is most unlikely. • For low accuracy scores, add more labeled data or split visually distinct documents into multiple models.
Low	Low	• Add more labeled data. • Split visually distinct documents into multiple models.

## Ensure high model accuracy for custom models

Variances in the visual structure of your documents affect the accuracy of your model. Reported accuracy scores can be inconsistent when the analyzed documents differ from documents used in training. Keep in mind that a document set can look similar when viewed by humans but appear dissimilar to an AI model. To follow, is a list of the best practices for training models with the highest accuracy. Following these guidelines should produce a model with higher accuracy and confidence scores during analysis and reduce the number of documents flagged for human review.

- Ensure that all variations of a document are included in the training dataset.  
Variations include different formats, for example, digital versus scanned PDFs.
- Add at least five samples of each type to the training dataset if you expect the model to analyze both types of PDF documents.
- Separate visually distinct document types to train different models for custom template and neural models.
  - As a general rule, if you remove all user entered values and the documents look similar, you need to add more training data to the existing model.
  - If the documents are dissimilar, split your training data into different folders and train a model for each variation. You can then [compose](#) the different variations into a single model.
- Ensure that you don't have any extraneous labels.
- Ensure that signature and region labeling doesn't include the surrounding text.

## Table, row, and cell confidence

Here are some common questions that should help with interpreting the table, row, and cell scores:

## **Can cells have high confidence scores while the row has a low confidence score?**

The different levels of table confidence (cell, row, and table) are meant to capture the correctness of a prediction at that specific level. A correctly predicted cell that belongs to a row with other possible misses would have high cell confidence, but the row's confidence should be low. Similarly, a correct row in a table with challenges with other rows would have high row confidence whereas the table's overall confidence would be low.

## **How does merging cells affect confidence scores, given the change in the number of identified columns?**

Regardless of the type of table, the expectation for merged cells is that they should have lower confidence values. Furthermore, the cell that is missing (because it was merged with an adjacent cell) should have `NULL` value with lower confidence as well. How much lower these values might be depends on the training dataset, the general trend of both merged and missing cell having lower scores should hold.

## **What is the confidence score for optional values? Should you expect a cell with a "NULL" value to have a high confidence score since the value is absent?**

If your training dataset is representative of the optionality of cells, it helps the model know how often a value tends to appear in the training set, and thus what to expect during inference. This feature is used when computing the confidence of either a prediction or of making no prediction at all (`NULL`). You should expect an empty field with high confidence for missing values that are mostly empty in the training set too.

## **Can confidence scores alter if an optional field is absent? Do the confidence scores reflect this change?**

When a value is missing from a row, the cell has a `NULL` value and confidence assigned. A high confidence score here should mean that the model prediction (of there not being a value) is more likely to be correct. In contrast, a low score should signal more uncertainty from the model (and thus the possibility of an error, like the value being missed).

## **What are the expectations for cell and row confidence when extracting a multi-page table with a row split across pages?**

Expect the cell confidence to be high and row confidence to be potentially lower than rows that aren't split. The proportion of split rows in the training data set can affect the confidence score. In general, a split row looks different than the other rows in the table (thus, the model is less certain that it's correct).

## For tables spanning multiple pages, can we assume confidence scores remain consistent if rows end and start cleanly at page boundaries?

Since rows look similar in shape and contents, regardless of where they are in the document (or in which page), their respective confidence scores should be consistent.

## What is the best way to utilize the new confidence scores?

- Look at all levels of table confidence starting in a top-to-bottom approach: begin by checking a table's confidence as a whole, then drill down to the row level and look at individual rows, finally look at cell-level confidences. Depending on the type of table, there are a couple of things of note:
- For **fixed tables**, cell-level confidence already captures quite a bit of information on the correctness of things. This means that simply going over each cell and looking at its confidence can be enough to help determine the quality of the prediction. For **dynamic tables**, the levels are meant to build on top of each other, so the top-to-bottom approach is more important.

## Next step

[Learn more about custom models](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Analyze document API response

Article • 04/23/2025

In this article, let's examine the different objects returned as part of the `AnalyzeDocument` response and how to use the document analysis API response in your applications.

## Analyze document request

The Document Intelligence APIs analyze images, PDFs, and other document files to extract and detect various content, layout, style, and semantic elements. The `Analyze` operation is an async API. Submitting a document returns an **Operation-Location** header that contains the URL to poll for completion. When an analysis request completes successfully, the response contains the elements described in the [model data extraction](#).

## Response elements

- Content elements are the basic text elements extracted from the document.
- Layout elements group content elements into structural units.
- Style elements describe the font and language of content elements.
- Semantic elements assign meaning to the specified content elements.

All content elements are grouped according to pages, specified by page number (1-indexed). They're also sorted by reading order that arranges semantically contiguous elements together, even if they cross line or column boundaries. When the reading order among paragraphs and other layout elements is ambiguous, the service generally returns the content in a left-to-right, top-to-bottom order.

### Note

Currently, Document Intelligence doesn't support reading order across page boundaries. Selection marks aren't positioned within the surrounding words.

The top-level content property contains a concatenation of all content elements in reading order. All elements specify their position in the reader order via spans within this content string. The content of some elements isn't always contiguous.

## Analyze response

The `Analyze` response for each API returns different objects. API responses contain elements from component models where applicable.

 [Expand table](#)

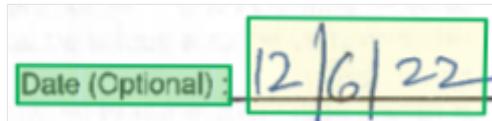
Response content	Description	API
<code>pages</code>	Words, lines and spans recognized from each page of the input document.	Read, Layout, General Document, Prebuilt, and Custom models
<code>paragraphs</code>	Content recognized as paragraphs.	Read, Layout, General Document, Prebuilt, and Custom models
<code>styles</code>	Identified text element properties.	Read, Layout, General Document, Prebuilt, and Custom models
<code>languages</code>	Identified language associated with each span of the text extracted	Read
<code>tables</code>	Tabular content identified and extracted from the document. Tables relate to tables identified by the pretrained layout model. Content labeled as tables is extracted as structured fields in the documents object.	Layout, General Document, Invoice, and Custom models
<code>figures</code>	Figures (charts, images) identified and extracted from the document, providing visual representations that aid in the understanding of complex information.	The Layout model
<code>sections</code>	Hierarchical document structure identified and extracted from the document. Section or subsection with the corresponding elements (paragraph, table, figure) attached to it.	The Layout model
<code>keyValuePairs</code>	Key-value pairs recognized by a pretrained model. The key is a span of text from the document with the associated value.	General document and Invoice models
<code>documents</code>	Fields recognized are returned in the <code>fields</code> dictionary within the list of documents	Prebuilt models, Custom models.

For more information on the objects returned by each API, see [model data extraction](#).

## Element properties

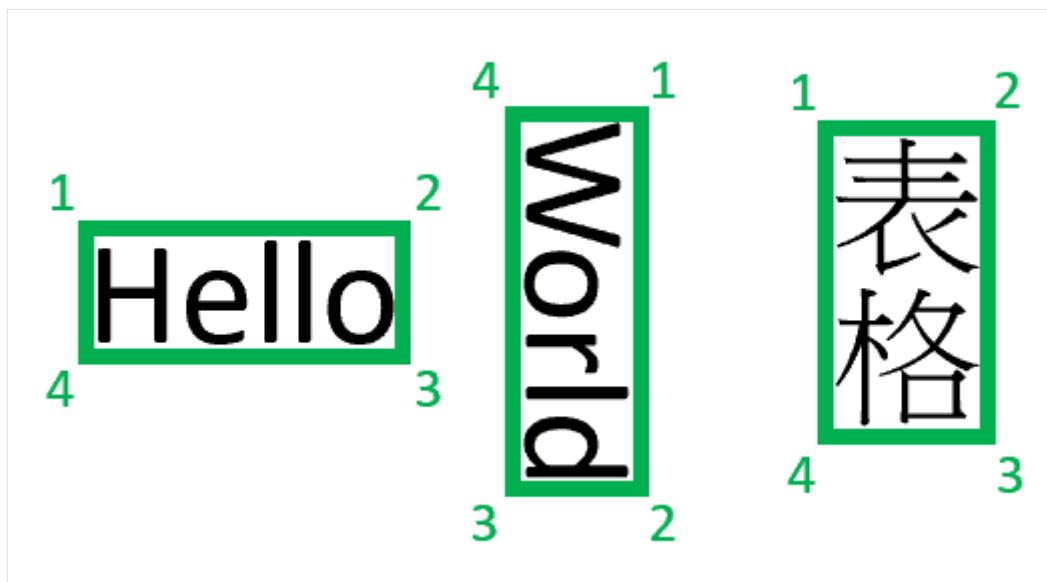
### Spans

Spans specify the logical position of each element in the overall reading order, with each span specifying a character offset and length into the top-level content string property. By default, character offsets and lengths are returned in units of user-perceived characters (also known as [grapheme clusters](#) or text elements). To accommodate different development environments that use different character units, user can specify the `stringIndexIndex` query parameter to return span offsets and lengths in Unicode code points (Python 3) or UTF16 code units (Java, JavaScript, .NET) as well. For more information, see [multilingual/emoji support](#).



## Bounding Region

Bounding regions describe the visual position of each element in the file. When elements aren't visually contiguous or cross pages (tables), the positions of most elements are described via an array of bounding regions. Each region specifies the page number (1-indexed) and bounding polygon. The bounding polygon is described as a sequence of points, clockwise from the left relative to the natural orientation of the element. For quadrilaterals, plot points are top-left, top-right, bottom-right, and bottom-left corners. Each point represents its x, y coordinate in the page unit specified by the unit property. In general, unit of measure for images is pixels while PDFs use inches.



### ⓘ Note

Currently, Document Intelligence only returns 4-vertex quadrilaterals as bounding polygons. Future versions may return different number of points to describe more complex shapes, such as curved lines or nonrectangular images. Bounding regions are

applied only to rendered files, if the file isn't rendered, bounding regions aren't returned.

Currently files of docx/xlsx/pptx/html format aren't rendered.

## Content elements

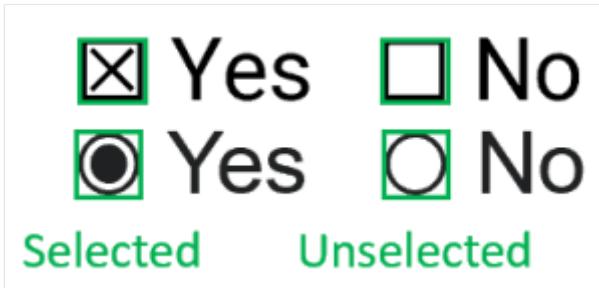
### Word

A word is a content element composed of a sequence of characters. With Document Intelligence, a word is defined as a sequence of adjacent characters, with whitespace separating words from one another. For languages that don't use space separators between words each character is returned as a separate word, even if it doesn't represent a semantic word unit.



### Selection marks

A selection mark is a content element that represents a visual glyph indicating the state of a selection. Checkbox is a common form of selection marks. However, they're also represented via radio buttons or a boxed cell in a visual form. The state of a selection mark can be selected or unselected, with different visual representation to indicate the state.



## Layout elements

### Line

A line is an ordered sequence of consecutive content elements separated by a visual space, or ones that are immediately adjacent for languages without space delimiters between words. Content elements in the same horizontal plane (row) but separated by more than a single visual space are most often split into multiple lines. While this feature sometimes splits semantically contiguous content into separate lines, it enables the representation of textual

content split into multiple columns or cells. Lines in vertical writing are detected in the vertical direction.

To make your document look professionally produced, Word provides header, footer, cover page, and text box designs that complement each other. For example, you can add a matching cover page, header, and sidebar.

## Paragraph

A paragraph is an ordered sequence of lines that form a logical unit. Typically, the lines share common alignment and spacing between lines. Paragraphs are often delimited via indentation, added spacing, or bullets/numbering. Content can only be assigned to a single paragraph. Select paragraphs can also be associated with a functional role in the document. Currently supported roles include page header, page footer, page number, title, section heading, and footnote.

Video provides a powerful way to help you prove your point. When you click Online Video, you can paste in the embed code for the video you want to add. You can also type a keyword to search online for the video that best fits your document.

To make your document look professionally produced, Word provides header, footer, cover page, and text box designs that

Themes and styles also help keep your document coordinated. When you click Design and choose a new Theme, the pictures, charts, and SmartArt graphics change to match your new theme. When you apply styles, your headings change to match the new theme.

Save time in Word with new buttons that show up where you need them. To change

## Page

A page is a grouping of content that typically corresponds to one side of a sheet of paper. A rendered page is characterized via width and height in the specified unit. In general, images use pixel while PDFs use inch. The angle property describes the overall text angle in degrees for pages that can be rotated.

### ① Note

For spreadsheets like Excel, each sheet is mapped to a page. For presentations, like PowerPoint, each slide is mapped to a page. For file formats like HTML or Word documents, which lack a native page concept without rendering, the entire main content is treated as a single page.

## Table

A table organizes content into a group of cells in a grid layout. The rows and columns can be visually separated by grid lines, color banding, or greater spacing. The position of a table cell is specified via its row and column indices. A cell can span across multiple rows and columns.

Based on its position and styling, a cell can be classified as general content, row header, column header, stub head, or description:

- A row header cell is typically the first cell in a row that describes the other cells in the row.
- A column header cell is typically the first cell in a column that describes the other cells in a column.
- A row or column can contain multiple header cells to describe hierarchical content.
- A stub head cell is typically the cell in the first row and first column position. It can be empty or describe the values in the header cells in the same row/column.
- A description cell generally appears at the topmost or bottom area of a table, describing the overall table content. However, it can sometimes appear in the middle of a table to break the table into sections. Typically, description cells span across multiple cells in a single row.
- A table caption specifies content that explains the table. A table can further have an associated caption and a set of footnotes. Unlike a description cell, a caption typically lies outside the grid layout. A table footnote annotates content inside the table, often marked with a footnote symbol often found below the table grid.

**Layout tables differ from document fields extracted from tabular data.** Layout tables are extracted from tabular visual content in the document without considering the semantics of the content. In fact, some layout tables are designed purely for visual layout and don't always contain structured data. The method to extract structured data from documents with diverse visual layout, like itemized details of a receipt, generally requires significant post processing. It's essential to map the row or column headers to structured fields with normalized field names. Depending on the document type, use prebuilt models or train a custom model to extract such structured content. The resulting information is exposed as document fields. Such trained models can also handle tabular data without headers and structured data in nontabular forms, for example the work experience section of a resume.

#### Note

The bounding regions for figures and tables cover only the core content and exclude associated caption and footnotes.

(In millions, except earnings per share)	2021	2020	2019
Year Ended June 30,			
Net income available for common shareholders (A)	\$ 61,271	\$ 44,281	\$ 39,240
Weighted average outstanding shares of common stock (B)	7,547	7,610	7,673
Dilutive effect of stock-based awards	61	73	80
Common stock and common stock equivalents (C)	7,508	7,683	7,753
Earnings Per Share			
Basic (A/B)	\$ 8.12	\$ 5.82	\$ 5.11
Diluted (A/C)	\$ 8.05	\$ 5.76	\$ 5.06

## Figures

Figures (charts, images) in documents play a crucial role in complementing and enhancing the textual content, providing visual representations that aid in the understanding of complex information. The figures object detected by the Layout model has key properties like

`boundingRegions` (the spatial locations of the figure on the document pages, including the page number and the polygon coordinates that outline the figure's boundary), `spans` (details the text spans related to the figure, specifying their offsets and lengths within the document's text. This connection helps in associating the figure with its relevant textual context), `elements` (the identifiers for text elements or paragraphs within the document that are related to or describe the figure) and `caption`, if any.

When `output=figures` is specified during the initial `Analyze` operation, the service generates cropped images for all detected figures that can be accessed via

`/analyzeResults/{resultId}/figures/{figureId}`. `FigureId` is included in each figure object, following an undocumented convention of `{pageNumber}.{figureIndex}` where `figureIndex` resets to one per page.

JSON

```
{
  "figures": [
    {
      "id": "{figureId}",
      "boundingRegions": [],
      "spans": [],
      "elements": [
        "/paragraphs/15",
        ...
      ],
      "caption": {
        "content": "Here is a figure with some text",
        "boundingRegions": [],
        "spans": [],
        "elements": [
          "/paragraphs/15"
        ]
      }
    }
  ]
}
```

```
    ]  
}
```

## Sections

Hierarchical document structure analysis is pivotal in organizing, comprehending, and processing extensive documents. This approach is vital for semantically segmenting long documents to boost comprehension, facilitate navigation, and improve information retrieval. The advent of [retrieval-augmented generation \(RAG\)](#) in document generative AI underscores the significance of hierarchical document structure analysis. The Layout model supports sections and subsections in the output, which identifies the relationship of sections and object within each section. The hierarchical structure is maintained in `elements` of each section.

JSON

```
{  
  "sections": [  
    {  
      "spans": [],  
      "elements": [  
        "/paragraphs/0",  
        "/sections/1",  
        "/sections/2",  
        "/sections/5"  
      ]  
    },  
    ...  
  ]  
}
```

## Form field (key value pair)

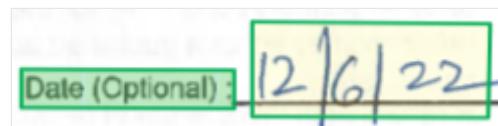
A form field consists of a field label (key) and value. The field label is generally a descriptive text string describing the meaning of the field. It often appears to the left of the value, though it can also appear over or under the value. The field value contains the content value of a specific field instance. The value can consist of words, selection marks, and other content elements. It can also be empty for unfilled form fields. A special type of form field has a selection mark value with the field label to its right. Document field is a similar but distinct concept from general form fields. The field label (key) in a general form field must appear in the document. Thus, it can't generally capture information like the merchant name in a receipt. Document fields are labeled and don't extract a key. Document fields only map an extracted value to a labeled key. For more information, see [document fields](#).

7. PROPOSED START DATE
2/1/2022

## Style elements

### Style

A style element describes the font style to apply to text content. The content is specified via spans into the global content property. Currently, the only detected font style is whether the text is handwritten. As other styles are added, text can be described via multiple nonconflicting style objects. For compactness, all text sharing the particular font style (with the same confidence) are described via a single style object.



#### JSON

```
{  
  "confidence": 1,  
  "spans": [  
    {  
      "offset": 2402,  
      "length": 7  
    }  
  ],  
  "isHandwritten": true  
}
```

## Language

A language element describes the detected language for content specified via spans into the global content property. The detected language is specified via a [BCP-47 language tag](#) to indicate the primary language and optional script and region information. For example, English and traditional Chinese are recognized as "en" and *zh-Hant*, respectively. Regional spelling differences for UK English can lead to text being detected as *en-GB*. Language elements don't cover text without a dominant language (ex. numbers).

## Semantic elements

### Note

The mentioned semantic elements apply to Document Intelligence prebuilt models. Your custom models may return different data representations. For example, date and time returned by a custom model may be represented in a pattern that differs from standard ISO 8601 formatting.

## Document

A document is a semantically complete unit. A file can contain multiple documents, such as multiple tax forms within a PDF file, or multiple receipts within a single page. However, the ordering of documents within the file doesn't fundamentally affect the information it conveys.

### Note

Currently, Document Intelligence doesn't support multiple documents on a single page.

The document type describes documents sharing a common set of semantic fields, represented by a structured schema, independent of its visual template or layout. For example, all documents of type "receipt" can contain the merchant name, transaction date, and transaction total, although restaurant and hotel receipts often differ in appearance.

A document element includes the list of recognized fields from among the fields specified by the semantic schema of the detected document type:

- A document field can be extracted or inferred. Extracted fields are represented via the extracted content and optionally its normalized value, if interpretable.
- An inferred field doesn't have content property and is represented only via its value.
- An array field doesn't include a content property. The content can be concatenated from the content of the array elements.
- An object field does contain a content property that specifies the full content representing the object that can be a superset of the extracted subfields.

The semantic schema of a document type is described via the fields it contains. Each field schema is specified via its canonical name and value type. Field value types include basic (ex. string), compound (ex. address), and structured (ex. array, object) types. The field value type also specifies the semantic normalization performed to convert detected content into a normalization representation. Normalization can be locale dependent.

## Basic types

[\[+\] Expand table](#)

Field value type	Description	Normalized representation	Example (Field content -> Value)
string	Plain text	Same as content	MerchantName: "Contoso" → "Contoso"
date	Date	ISO 8601 - YYYY-MM-DD	InvoiceDate: "5/7/2022" → "2022-05-07"
time	Time	ISO 8601 - hh:mm:ss	TransactionTime: "9:45 PM" → "21:45:00"
phoneNumber	Phone number	E.164 - +{CountryCode}{SubscriberNumber}	WorkPhone: "(800) 555-7676" → "+18005557676"
countryRegion	Country/Region	ISO 3166-1 alpha-3	CountryRegion: "United States" → "USA"
selectionMark	Is selected	"signed" or "unsigned"	AcceptEula: <input checked="" type="checkbox"/> → "selected"
signature	Is signed	Same as content	LendeeSignature: {signature} → "signed"
number	Floating point number	Floating point number	Quantity: "1.20" → 1.2
integer	Integer number	64-bit signed number	Count: "123" → 123
boolean	Boolean value	true/false	IsStatutoryEmployee: <input checked="" type="checkbox"/> → true

## Compound types

- Currency: Currency amount with optional currency unit. A value, for example:

InvoiceTotal: \$123.45

JSON

```
{  
    "amount": 123.45,  
    "currencySymbol": "$"  
}
```

- Address: Parsed address. For example: ShipToAddress: 123 Main St., Redmond, WA 98052

JSON

```
{  
  "poBox": "PO Box 12",  
  "houseNumber": "123",  
  "streetName": "Main St.",  
  "city": "Redmond",  
  "state": "WA",  
  "postalCode": "98052",  
  "countryRegion": "USA",  
  "streetAddress": "123 Main St."  
}
```

## Structured types

- Array: List of fields of the same type

JSON

```
"Items": {  
  "type": "array",  
  "valueArray": [  
    ]  
}
```

- Object: Named list of subfields of potentially different types

JSON

```
"InvoiceTotal": {  
  "type": "currency",  
  "valueCurrency": {  
    "currencySymbol": "$",  
    "amount": 110  
  },  
  "content": "$110.00",  
  "boundingRegions": [  
    {  
      "pageNumber": 1,  
      "polygon": [  
        7.3842,  
        7.465,  
        7.9181,  
        7.465,  
        7.9181,  
        7.6089,  
        7.3842,  
        7.6089  
      ]  
    }  
  ]
```

```
        }
    ],
    "confidence": 0.945,
    "spans": [
        {
            "offset": 806,
            "length": 7
        }
    ]
}
```

## Next steps

- Try processing your own forms and documents with [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

# Understanding Document Intelligence Layout API Markdown Output Format

Article • 05/05/2025

Azure AI Document Intelligence Layout API can transform your documents into rich Markdown, preserving their original structure and formatting. Just specify `outputContentFormat=markdown` in your request to receive semantically structured content that maintains paragraphs, headings, tables, and other document elements in their proper hierarchy.

This Markdown output elegantly captures the document's original organization while providing standardized, easily consumable content for downstream applications. The preserved semantic structure enables more sophisticated document processing workflows without losing the context and relationships between document elements.

## Markdown elements supported in Layout Analysis

The following Markdown elements are included in Layout API responses:

- Paragraph
- Heading
- Table
- Figure
- Selection Mark
- Formula
- Barcode
- PageNumber/PageHeader/PageFooter
- PageBreak
- KeyValuePairs/Language/Style
- Spans and Content

## Paragraph

Paragraphs represent cohesive blocks of text that belong together semantically. The Layout API maintains paragraph integrity by:

- Preserving paragraph boundaries with empty lines between separate paragraphs
- Using line breaks within paragraphs to maintain the visual structure of the original document
- Maintaining proper text flow that respects the original document's reading order

Here's an example:

## Markdown

This is paragraph 1.

This is still paragraph 1, even if in another Markdown line.

This is paragraph 2. There is a blank line between paragraph 1 and paragraph 2.

## Heading

Headings organize document content into a hierarchical structure to make navigation and understanding easier. The Layout API has the following capabilities:

- Uses standard Markdown heading syntax with 1-6 hash symbols (#) corresponding to heading levels.
- Maintains proper spacing with two blank lines before each heading for improved readability.

Here's an example:

## Markdown

```
# This is a title  
## This is heading 1  
### This is heading 2  
#### This is heading 3
```

## Table

Tables preserve complex structured data in a visually organized format. The Layout API uses HTML table syntax for maximum fidelity and compatibility:

- Implements full HTML table markup (`<table>`, `<tr>`, `<th>`, `<td>`) rather than standard Markdown tables
- Preserves merged cell with HTML rowspan and colspan attributes.
- Preserves table captions with the `<caption>` tag to maintain document context
- Handles complex table structures including headers, cells, and footers
- Maintains proper spacing with two blank lines before each table for improved readability
- Preserves table footnotes as separate paragraph following the table

Here's an example:

Markdown

```
<table>
<caption>Table 1. This is a demo table</caption>
<tr><th>Header</th><th>Header</th></tr>
<tr><td>Cell</td><td>Cell</td></tr>
<tr><td>Cell</td><td>Cell</td></tr>
<tr><td>Cell</td><td>Cell</td></tr>
<tr><td>Footer</td><td>Footer</td></tr>
</table>
This is the footnote of the table.
```

## Figure

The Layout API preserves figure elements:

- Encapsulates figure content in `<figure>` tags to maintain semantic distinction from surrounding text
- Preserves figure captions with the `<figcaption>` tag to provide important context
- Preserves figure footnotes as separate paragraphs following the figure container

Here's an example:

Markdown

```
<figure>
<figcaption>Figure 2 This is a figure</figcaption>
```

Values

300  
200  
100  
0

Jan Feb Mar Apr May Jun Months

```
</figure>
```

This is footnote if the figure have.

## Selection Mark

Selection marks represent checkbox-like elements in forms and documents. The Layout API:

- Uses Unicode characters for visual clarity:  (checked) and  (unchecked)
- Filters out low-confidence checkbox detections (below 0.1 confidence) to improve reliability
- Maintains the semantic relationship between selection marks and their associated text

## Formula

Mathematical formulas are preserved with LaTeX-compatible syntax that allows for rendering of complex mathematical expressions:

- Inline formulas are enclosed in single dollar signs ( $\$...$$ ) to maintain text flow
- Block formulas use double dollar signs ( $\$\$\dots\$\$$ ) for standalone display
- Multi-line formulas are represented as consecutive block formulas, preserving mathematical relationships
- Original spacing and formatting are maintained to ensure accurate representation

Here's an example of inline formula, single-line formula block and multiple-lines formula block:

### Markdown

```
The mass-energy equivalence formula  $E = m c^2$  is an example of an inline formula
```

```

$$\frac{n!}{k!} \left( n - k \right)! = \binom{n}{k}$$

```

```

$$\frac{p_j p_{j+1}}{\Delta E_{k+1}} = \prod_{k=1}^{j-1} e^{-\beta_{k+1}}$$


$$= \exp \left[ - \sum_{k=1}^{j-1} \beta_{k+1} \Delta E_{k+1} \right]$$

```

## Barcode

Barcodes and QR codes are represented using Markdown image syntax with added semantic information:

- Uses standard image Markdown syntax with descriptive attributes
- Captures both the barcode type (QR code, barcode, etc.) and its encoded value
- Preserves the semantic relationship between barcodes and surrounding content

Here's an example:

```
![QRCode](barcodes/1.1 "https://www.microsoft.com")  
![UPCA](barcodes/1.2 "012345678905")  
![barcode type](barcodes/pagenumber.barcodenumbers "barcode value/content")
```

## PageNumber/PageHeader/PageFooter

Page metadata elements provide context about document pagination but aren't meant to be displayed inline with the main content:

- Enclosed in HTML comments to preserve the information while keeping it hidden from standard Markdown rendering
- Maintains original page structure information that might be valuable for document reconstruction
- Enables applications to understand document pagination without disrupting the content flow

Here's an example:

Markdown

```
<!-- PageHeader="This is page header" -->  
<!-- PageFooter="This is page footer" -->  
<!-- PageNumber="1" -->
```

## PageBreak

To easily figure out which parts belong to which page base on the pure Markdown content, we introduced PageBreak as the delimiter of the pages

Here's an example:

Markdown

```
<!-- PageBreak -->
```

## KeyValuePairs/Language/Style

For KeyValuePairs/Language/Style, we map them to Analytics JSON body and not in the Markdown content.

 Note

For more information on Markdown that is currently supported for user content on GitHub.com, see [GitHub Flavored Markdown Spec](#).

## Conclusion

Document Intelligence's Markdown elements provide a powerful way to represent the structure and content of analyzed documents. By understanding and properly utilizing these Markdown elements, you can enhance your document processing workflows and build more sophisticated content extraction applications.

## Next steps

- Try processing your documents with [Document Intelligence Studio](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

# Retrieval-Augmented Generation with Azure AI Document Intelligence

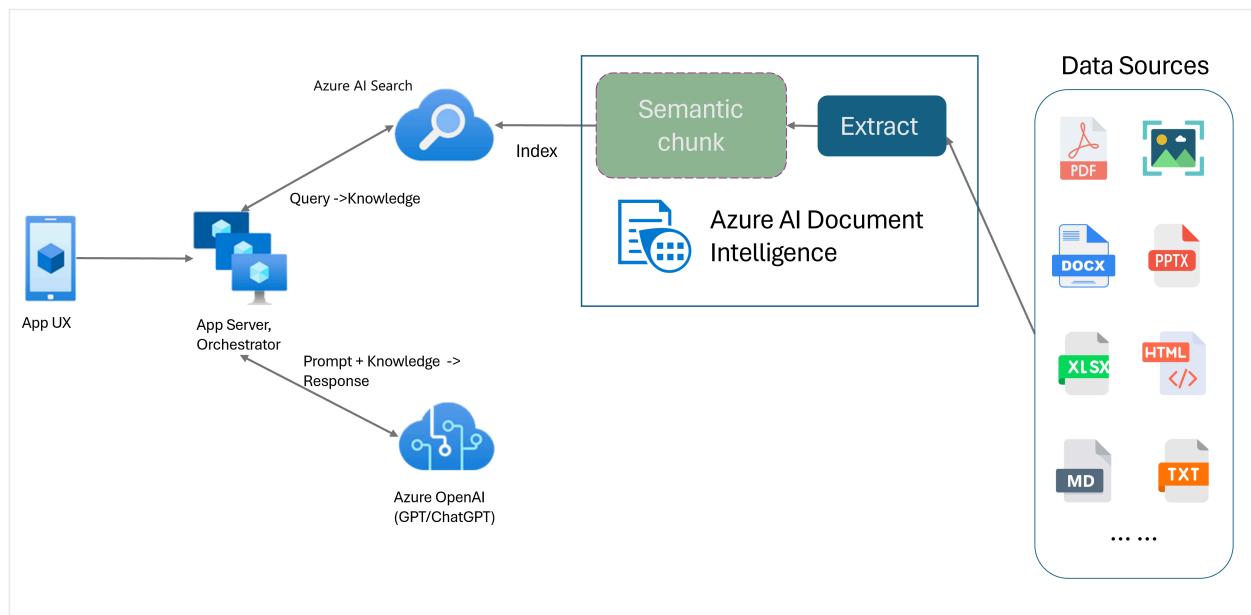
Article • 02/27/2025

This content applies to:  v4.0 (GA)

## Introduction

Retrieval-Augmented Generation (RAG) is a design pattern that combines a pretrained Large Language Model (LLM) like ChatGPT with an external data retrieval system to generate an enhanced response incorporating new data outside of the original training data. Adding an information retrieval system to your applications enables you to chat with your documents, generate captivating content, and access the power of Azure OpenAI models for your data. You also have more control over the data used by the LLM as it formulates a response.

The Document Intelligence [Layout model](#) is an advanced machine-learning based document analysis API. The Layout model offers a comprehensive solution for advanced content extraction and document structure analysis capabilities. With the Layout model, you can easily extract text and structural elements to divide large bodies of text into smaller, meaningful chunks based on semantic content rather than arbitrary splits. The extracted information can be conveniently outputted to Markdown format, enabling you to define your semantic chunking strategy based on provided building blocks.



## Semantic chunking

Long sentences are challenging for natural language processing (NLP) applications. Especially when they're composed of multiple clauses, complex noun or verb phrases, relative clauses, and parenthetical groupings. Just like the human beholder, an NLP system also needs to successfully keep track of all the presented dependencies. The goal of semantic chunking is to find semantically coherent fragments of a sentence representation. These fragments can then be processed independently and recombined as semantic representations without loss of information, interpretation, or semantic relevance. The inherent meaning of the text is used as a guide for the chunking process.

Text data chunking strategies play a key role in optimizing the RAG response and performance. Fixed-sized and semantic are two distinct chunking methods:

- **Fixed-sized chunking.** Most chunking strategies used in RAG today are based on fix-sized text segments known as chunks. Fixed-sized chunking is quick, easy, and effective with text that doesn't have a strong semantic structure such as logs and data. However it isn't recommended for text that requires semantic understanding and precise context. The fixed-size nature of the window can result in severing words, sentences, or paragraphs impeding comprehension and disrupting the flow of information and understanding.
- **Semantic chunking.** This method divides the text into chunks based on semantic understanding. Division boundaries are focused on sentence subject and use significant computational algorithmically complex resources. However, it has the distinct advantage of maintaining semantic consistency within each chunk. It's useful for text summarization, sentiment analysis, and document classification tasks.

## Semantic chunking with Document Intelligence Layout model

Markdown is a structured and formatted markup language and a popular input for enabling semantic chunking in RAG (Retrieval-Augmented Generation). You can use the Markdown content from the [Layout model](#) to split documents based on paragraph boundaries, create specific chunks for tables, and fine-tune your chunking strategy to improve the quality of the generated responses.

## Benefits of using the Layout model

- **Simplified processing.** You can parse different document types, such as digital and scanned PDFs, images, office files (docx, xlsx, pptx), and HTML, with just a single API call.

- **Scalability and AI quality.** The Layout model is highly scalable in Optical Character Recognition (OCR), table extraction, and [document structure analysis](#). It supports [309 printed and 12 handwritten languages](#), further ensuring high-quality results driven by AI capabilities.
- **Large language model (LLM) compatibility.** The Layout model Markdown formatted output is LLM friendly and facilitates seamless integration into your workflows. You can turn any table in a document into Markdown format and avoid extensive effort parsing the documents for greater LLM understanding.

**Text image processed with Document Intelligence Studio and output to MarkDown using Layout model**

The screenshot shows a complex document layout being processed by Document Intelligence Studio. On the left, the original document contains various elements labeled with arrows:

- Title:** "NEWS TODAY" (with a blue arrow pointing to it)
- Page header:** "Tuesday, Sep 20, YYYY" and "Latest news and bulletin updates" (with a blue arrow pointing to it)
- Section heading:** "The scoop of the day" (with a blue arrow pointing to it)
- Paragraph:** A detailed paragraph about video embeds (with a blue arrow pointing to it)
- Picture caption:** "Power Center: To make your document look professionally produced, Word provides header, footer, cover page, and text box designs that complement each other. For example, you can add a matching cover page, header, and sidebar." (with a blue arrow pointing to it)
- Page number:** "Page XX" (with a blue arrow pointing to it)

On the right, the processed output is shown in a "Content" tab of the studio interface:

```

<!-- PageHeader="Tuesday, Sep 20, YYYY" -->
NEWS TODAY

<!-- PageHeader="Latest news and bulletin updates" --> <!--
PageHeader="Issue \#10" -->
Mirjam Nilsson

The scoop of the day The latest updates

Video provides a powerful way to help you prove your point. When you click Online Video, you can paste in the embed code for the video you want to add. You can also type a keyword to search online for the video that best fits your document.

Themes and styles also help keep your document coordinated. When you click Design and choose a new Theme, the pictures, charts, and SmartArt graphics change to match your new theme. When you apply styles, your headings change to match the new theme.

Save time in Word with new buttons that show up where you need them. To change the look of your document, click a button for layout options, and then click the plus sign. When you work on a table, click where you want to add a row or a column, and then click the plus sign.

Reading is easier, too, in the new Reading view. You can collapse parts of the document and focus on the text you want. Page XX

To make your document look professionally produced, Word provides header, footer, cover page, and text box designs that complement each other. For example, you can add a matching cover page, header, and sidebar.

Click Insert and then choose the elements you want from the different galleries.

Themes and styles also help keep your document coordinated. When you click Design and choose a new Theme, the pictures, charts, and SmartArt graphics change to match your new theme. When you apply styles, your headings change to match the new theme.

```

**Table image processed with Document Intelligence Studio using Layout model**

## NOTE 2 — EARNINGS PER SHARE

Basic earnings per share ("EPS") is computed based on the weighted average number of shares of common stock outstanding during the period. Diluted EPS is computed based on the weighted average number of shares of common stock plus the effect of dilutive potential common shares outstanding during the period using the treasury stock method. Dilutive potential common shares include outstanding stock options and stock awards.

The components of basic and diluted EPS were as follows:

(In millions, except earnings per share)	2021	2020	2019
Year Ended June 30,			
Net income available for common shareholders (A)	\$ 61,271	\$ 44,281	\$ 39,240
Weighted average outstanding shares of common stock (B)	7,547	7,610	7,673
Dilutive effect of stock-based awards	61	73	80
Common stock and common stock equivalents (C)	7,608	7,683	7,753
Earnings Per Share			
Basic (A/B)	\$ 8.12	\$ 5.82	\$ 5.11
Diluted (A/C)	\$ 8.05	\$ 5.76	\$ 5.06

Table



(In millions, except earnings per share)	2021	2020	2019
Year Ended June 30,			
Net income available for common shareholders (A)	\$ 61,271	\$ 44,281	\$ 39,240
Weighted average outstanding shares of common stock (B)	7,547	7,610	7,673
Dilutive effect of stock-based awards	61	73	80
Common stock and common stock equivalents (C)	7,608	7,683	7,753
Earnings Per Share			
Basic (A/B)	\$ 8.12	\$ 5.82	\$ 5.11
Diluted (A/C)	\$ 8.05	\$ 5.76	\$ 5.06

## Get started

The Document Intelligence Layout model 2024-11-30 (GA) supports the following development options:

- [Document Intelligence Studio](#).
- [REST API](#).
- [.NET • Java • JavaScript • Python](#) programming language client libraries (SDKs).

Ready to begin?

## Document Intelligence Studio

You can follow the [Document Intelligence Studio quickstart](#) to get started. Next, you can integrate Document Intelligence features with your own application using the sample

code provided.

- Start with the [Layout model](#). You need to select the following **Analyze options** to use RAG in the studio:

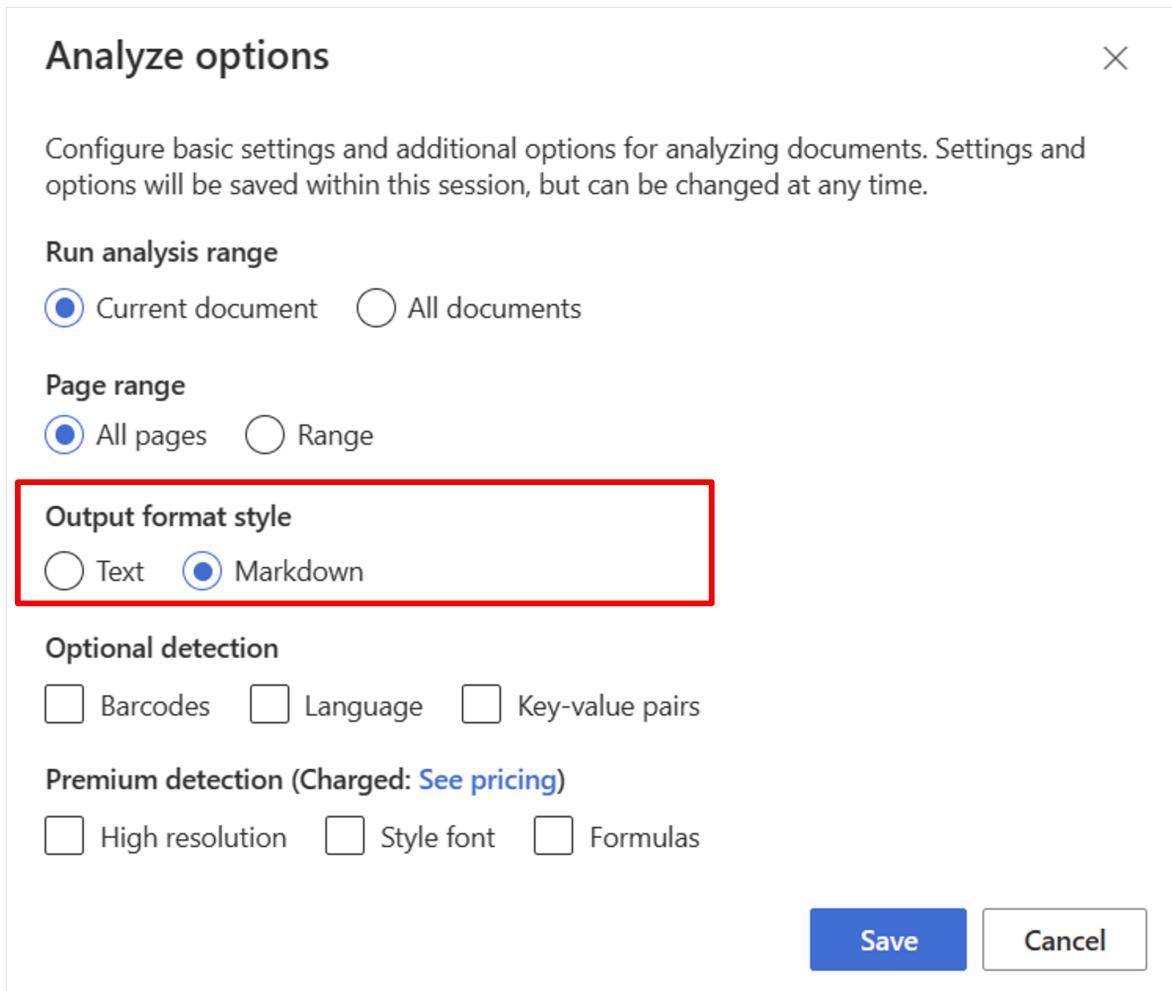
**\*\*Required\*\***

- Run analysis range → **Current document**.
- Page range → **All pages**.
- Output format style → **Markdown**.

**\*\*Optional\*\***

- You can also select relevant optional detection parameters.

- Select **Save**.



- Select the **Run analysis** button to view the output.



**SDK or REST API**

- You can follow the [Document Intelligence quickstart](#) for your preferred programming language SDK or REST API. Use the Layout model to extract content and structure from your documents.
- You can also check out GitHub repos for code samples and tips for analyzing a document in markdown output format.
  - [Python ↗](#)
  - [JavaScript ↗](#)
  - [Java ↗](#)
  - [.NET ↗](#)

## Build document chat with semantic chunking

- [Azure OpenAI on your data](#) enables you to run supported chat on your documents. Azure OpenAI on your data applies the Document Intelligence Layout model to extract and parse document data by chunking long text based on tables and paragraphs. You can also customize your chunking strategy using [Azure OpenAI sample scripts ↗](#) located in our GitHub repo.
- Azure AI Document Intelligence is now integrated with [LangChain ↗](#) as one of its document loaders. You can use it to easily load the data and output to Markdown format. For more information, see our [sample code ↗](#) that shows a simple demo for RAG pattern with Azure AI Document Intelligence as document loader and Azure Search as retriever in LangChain.
- The chat with your data solution accelerator [code sample ↗](#) demonstrates an end-to-end baseline RAG pattern sample. It uses Azure AI Search as a retriever and Azure AI Document Intelligence for document loading and semantic chunking.

## Use case

If you're looking for a specific section in a document, you can use semantic chunking to divide the document into smaller chunks based on the section headers helping you to find the section you're looking for quickly and easily:

Python

```
# pip install azure-ai-documentintelligence==1.0.0b1
# pip install langchain langchain-community azure-ai-documentintelligence
```

```
from azure.ai.documentintelligence import DocumentIntelligenceClient

endpoint = "https://<my-custom-subdomain>.cognitiveservices.azure.com/"
key = "<api_key>"

from langchain_community.document_loaders import
AzureAIDocumentIntelligenceLoader
from langchain.text_splitter import MarkdownHeaderTextSplitter

# Initiate Azure AI Document Intelligence to load the document. You can
either specify file_path or url_path to load the document.
loader = AzureAIDocumentIntelligenceLoader(file_path=<path to your file>,
api_key = key, api_endpoint = endpoint, api_model="prebuilt-layout")
docs = loader.load()

# Split the document into chunks base on markdown headers.
headers_to_split_on = [
    ("#", "Header 1"),
    ("##", "Header 2"),
    ("###", "Header 3"),
]
text_splitter =
MarkdownHeaderTextSplitter(headers_to_split_on=headers_to_split_on)

docs_string = docs[0].page_content
splits = text_splitter.split_text(docs_string)
splits
```

[View samples on GitHub.](#)

## Next steps

- Learn more about [Azure AI Document Intelligence](#).
- Learn how to process your own forms and documents with the [Document Intelligence Studio ↗](#).
- Complete a [Document Intelligence quickstart](#) and get started creating a document processing app in the development language of your choice.

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | Get help at Microsoft Q&A

# Document Intelligence batch analysis

Article • 02/26/2025

The batch analysis API allows you to bulk process up to 10,000 documents using one request. Instead of analyzing documents one by one and keeping track of their respective request IDs, you can simultaneously analyze a collection of documents like invoices, loan papers, or custom documents. The input documents must be stored in an Azure blob storage container. Once the documents are processed, the API writes the results to a specified storage container.

## Batch analysis limits

- The maximum number of document files that can be in a single batch request is 10,000.
- Batch operation results are retained for 24 hours after completion. The batch operation status is no longer available 24 hours after batch processing is completed. The input documents and respective result files remain in the storage containers provided.

## Prerequisites

- An active Azure subscription. If you don't have an Azure subscription, you can [create one for free](#).
- A [Document Intelligence Azure Resource](#): once you have your Azure subscription, create a Document Intelligence resource in the Azure portal. You can use the free pricing tier (F0) to try the service. After your resource is deployed, select "**Go to resource**" to retrieve your **key** and **endpoint**. You need the resource key and endpoint to connect your application to the Document Intelligence service. You can also find these values on the **Keys and Endpoint** page in Azure portal.
- An [Azure Blob Storage account](#). [Create two containers](#) in your Azure Blob Storage account for your source and result files:
  - **Source container:** This container is where you upload document files for analysis.
  - **Result container:** This container is where results from the batch analysis API are stored.

## Storage container authorization

To allow the API to process documents and write results in your Azure storage containers, you must authorize using one of the following two options:

✓ **Managed Identity.** A managed identity is a service principal that creates a Microsoft Entra identity and specific permissions for an Azure managed resource. Managed identities enable you to run your Document Intelligence application without having to embed credentials in

your code, a safer way to grant access to storage data without including access signature tokens (SAS) in your code.

Review [Managed identities for Document Intelligence](#) to learn how to enable a managed identity for your resource and grant it access to your storage container.

 **Important**

When using managed identities, don't include a SAS token URL with your HTTP requests. Using managed identities replaces the requirement for you to include shared access signature tokens (SAS).

 **Shared Access Signature (SAS)**. A shared access signature is a URL that grants restricted access to your storage container. To use this method, create Shared Access Signature (SAS) tokens for your source and result containers. Go to the storage container in Azure portal and select "Shared access tokens" to generate SAS token and URL.

- Your **source** container or blob must designate **read**, **write**, **list**, and **delete** permissions.
- Your **result** container or blob must designate **write**, **list**, **delete** permissions.

## Generate SAS

X

A shared access signature (SAS) is a URI that grants restricted access to an Azure Storage container. Use it when you want to grant access to storage account resources for a specific time range without sharing your storage account key. [Learn more](#)

Signing method

Account key

User delegation key

Permissions \* ⓘ

4 selected

Read

Add

Create

Write

Delete

List

2:19:14 PM

US & Canada)

10:19:14 PM

(UTC-08:00) Pacific Time (US & Canada)

Allowed IP addresses ⓘ

for example, 168.1.5.65 or 168.1.5.65-168.1...

Allowed protocols ⓘ

HTTPS only  HTTPS and HTTP

**Generate SAS token and URL**

Review [Create SAS tokens](#) to learn more about generating SAS tokens and how they work.

## Calling the batch analysis API

### 1. Specify the input files

The batch API supports two options for specifying the files to be processed.

- If you want to process all the files in a container or a folder, and the number of files is less than the 10000 limit, use the `azureBlobSource` object in your request.

Bash

```
POST /documentModels/{modelId}:analyzeBatch
```

```
{
```

```
    "azureBlobSource": {  
        "containerUrl":
```

```
        "https://myStorageAccount.blob.core.windows.net/myContainer?mySasToken",
```

```
    ...  
},  
...  
}
```

- If you don't want to process all the files in a container or folder, but rather specific files in that container or folder, use the `azureBlobFileListSource` object. This operation requires a File List JSONL file which lists the files to be processed. Store the JSONL file in the root folder of the container. Here's an example JSONL file with two files listed:

JSON

```
{"file": "Adatum Corporation.pdf"}  
{"file": "Best For You Organics Company.pdf"}
```

Use a file list `JSONL` file with the following conditions:

- When you need to process specific files instead of all files in a container;
- When the total number of files in the input container or folder exceeds the 10,000 file batch processing limit;
- When you want more control over which files get processed in each batch request;

Bash

```
POST /documentModels/{modelId}:analyzeBatch  
  
{  
  "azureBlobFileListSource": {  
    "containerUrl": "https://myStorageAccount.blob.core.windows.net/myContainer?  
mySasToken",  
    "fileList": "myFileList.jsonl"  
    ...  
  },  
  ...  
}
```

A container URL or a container SAS URL is required in both options. Use container URL if using managed Identity to access your storage container. If you're using Shared Access Signature (SAS), use a SAS URL.

## 2. Specify the results location

- Specify the Azure Blob Storage container URL (or container SAS URL) for where you want your results to be stored using `resultContainerURL` parameter. We recommend using separate containers for source and results to prevent accidental overwriting.
- Set the `overwriteExisting` Boolean property to `False` and prevent overwriting any existing results for the same document. If you'd like to overwrite any existing results, set the Boolean to `True`. You're still billed for processing the document even if any existing results aren't overwritten.
- Use `resultPrefix` to group and store results in a specific container folder.

### 3. Build and run the POST request

Remember to replace the following sample container URL values with real values from your Azure Blob storage containers.

This example shows a POST request with `azureBlobSource` input

Bash

```
POST /documentModels/{modelId}:analyzeBatch

{
  "azureBlobSource": {
    "containerUrl": "https://myStorageAccount.blob.core.windows.net/myContainer?mySasToken",
    "prefix": "inputDocs/"
  },
  "resultContainerUrl": "https://myStorageAccount.blob.core.windows.net/myOutputContainer?mySasToken",
  "resultPrefix": "batchResults/",
  "overwriteExisting": true
}
```

This example shows a POST request with `azureBlobFileListSource` and a file list input

Bash

```
POST /documentModels/{modelId}:analyzeBatch

{
  "azureBlobFileListSource": {
    "containerUrl": "https://myStorageAccount.blob.core.windows.net/myContainer?mySasToken",
    "fileList": "myFileList.jsonl"
  },
  "resultContainerUrl": "
```

```
"https://myStorageAccount.blob.core.windows.net/myOutputContainer?mySasToken",  
    "resultPrefix": "batchResults/",  
    "overwriteExisting": true  
}
```

Here's an example **successful** response

Bash

```
202 Accepted  
Operation-Location: /documentModels/{modelId}/analyzeBatchResults/{resultId}
```

## 4. Retrieve API results

Use the `GET` operation to retrieve batch analysis results after the POST operation is executed. The GET operation fetches status information, batch completion percentage, and operation creation and update date/time. This information is **only retained for 24 hours** after the batch analysis is completed.

Bash

```
GET /documentModels/{modelId}/analyzeBatchResults/{resultId}  
200 OK  
  
{  
    "status": "running",      // notStarted, running, completed, failed  
    "percentCompleted": 67,   // Estimated based on the number of processed  
    documents  
    "createdDateTime": "2021-09-24T13:00:46Z",  
    "lastUpdatedDateTime": "2021-09-24T13:00:49Z"  
    ...  
}
```

## 5. Interpret status messages

For each document processed, a status is assigned, either `succeeded`, `failed`, `running`, `notStarted`, or `skipped`. A source URL, which is the source blob storage container for the input document, is provided.

- Status `notStarted` or `running`. The batch analysis operation isn't initiated or isn't completed. Wait until the operation is completed for all documents.
- Status `completed`. The batch analysis operation is finished.

- Status `succeeded`. The batch operation was successful, and input document was processed. The results are available at `resultUrl`, which is created by combining `resultContainerUrl`, `resultPrefix`, `input filename`, and `.ocr.json` extension. **Only files that have succeeded have the property `resultUrl`.**

Example of a `succeeded` status response:

Bash

```
{
  "resultId": "myresultId-",
  "status": "succeeded",
  "percentCompleted": 100,
  "createdDateTime": "2025-01-01T00:00:00",
  "lastUpdatedDateTime": "2025-01-01T00:00:00",
  "result": {
    "succeededCount": 10,000,
    "failedCount": 0,
    "skippedCount": 0,
    "details": [
      {
        "sourceUrl": "https://{{your-source-
container}}/inputFolder/document1.pdf",
        "resultUrl": "https://{{your-result-
container}}/resultsFolder/document1.pdf.ocr.json",
        "status": "succeeded"
      },
      ...
      {
        "sourceUrl": "https://{{your-source-
container}}/inputFolder/document10000.pdf",
        "resultUrl": "https://{{your-result-
container}}/resultsFolder/document10000.pdf.ocr.json",
        "status": "succeeded"
      }
    ]
  }
}
```

- Status `failed`. This error is only returned if there are errors in the overall batch request. Once the batch analysis operation starts, the individual document operation status doesn't affect the status of the overall batch job, even if all the files have the status `failed`.

Example of a `failed` status response:

Bash

```
[
  "result": {
    "succeededCount": 0,
    "failedCount": 2,
    "skippedCount": 0,
    "details": [
      {
        "sourceUrl": "https://{{your-source-
        container}}/inputFolder/document1.jpg",
        "status": "failed",
        "error": {
          "code": "InvalidArgumentException",
          "message": "Invalid argument.",
          "innererror": {
            "code": "InvalidSasToken",
            "message": "The shared access signature (SAS) is invalid."
          }
        }
      }
    ]
  }
]
...

```

- Status `skipped`: Typically, this status happens when output for the document is already present in the specified output folder and the `overwriteExisting` Boolean property is set to `false`.

Example of `skipped` status response:

Bash

```
[
  "result": {
    "succeededCount": 3,
    "failedCount": 0,
    "skippedCount": 2,
    "details": [
      ...
      {
        "sourceUrl": "https://{{your-source-
        container}}/inputFolder/document1.pdf",
        "status": "skipped",
        "error": {
          "code": "OutputExists",
          "message": "Analysis skipped because result file https://{{your-
          result-container}}/resultsFolder/document1.pdf.ocr.json already exists."
        }
      }
    ]
  }
]
...

```

 **Note**

Analysis results aren't returned for individual files until analysis for the entire batch is completed. To track detailed progress beyond `percentCompleted`, you can monitor `*.ocr.json` files as they're written into the `resultContainerUrl`.

## Next steps

[View code samples on GitHub.](#) ↗

# Troubleshooting latency issues in Azure AI Document Intelligence

Article • 02/06/2025

This article presents troubleshooting tips, remedial solutions, and best practices to address Document Intelligence latency issues. Latency refers to the duration an API server takes to handle and process an incoming request before delivering the response to the client. The time required to analyze a document varies based on its size (such as the number of pages) and the content on each page.

Document Intelligence operates as a multitenant service, ensuring that latency for similar documents is comparable, though not always identical. Variability in latency and performance is an inherent characteristic of any microservice-based, stateless, asynchronous service, especially when processing images and large documents on a large scale. Despite continuous efforts to increase hardware capacity and enhance scalability, some latency issues can still arise during runtime.

## Note

- Azure AI services don't provide a Service Level Agreement (SLA) for latency.
- The Document Intelligence API offers asynchronous functionality, allowing you to access results up to 24 hours after sending your request to our backend.
- Use the request ID provided by the POST operation to retrieve these results. If you encounter issues during your standard polling sequence, save the request ID and try again later before considering a retry. For further assistance, refer to our [service page](#).

## Set your latency baseline

To evaluate latency, you should first establish baseline metrics for your specific scenario. These metrics give you the expected end-to-end and server latency within the context of your application environment. Once you have these baseline metrics, it becomes easier to distinguish between expected and unexpected conditions.

## Check Azure region status

When you're experiencing latency issues, the first step is to check [Azure status](#) for any current outages or issues that might impact your services.

- All active events are listed under the **Current Impact** tab.
- You can also check your resource in the host region. Go to Geography → Products And Services → AI + Machine Learning → Azure AI Document Intelligence and check the status for your region:

	Current Impact	Americas	Europe	Asia Pacific	Middle East and Africa	Azure Government	Azure China	Jio <sup>4</sup>							
Products And Services	*Non-Regional	East US	East US 2	Central US	North Central US	South Central US	West Central US	West US	West US 2	^West US 3	Canada East	Canada Central	Brazil South	Brazil Southeast	Mexico Central
AI + MACHINE LEARNING	Azure AI Document Intelligence	--	✓	✓	✓	✓	✓	✓	✓	✓	--	✓	✓	--	--

## Check file size

Monitor the size of files you send via the request API. Processing larger files in parallel can result in increased processing times. Normalize your metric by measuring latency per page. If you observe sustained periods (exceeding one hour) where latency per page consistently surpasses 15 seconds, consider addressing the issue.

## Check Azure Blob storage latency

The size of a request affects latency in Azure Storage operations. Larger operations take more time to complete due to the increased volume of data transferred over the network and processed by Azure Storage.

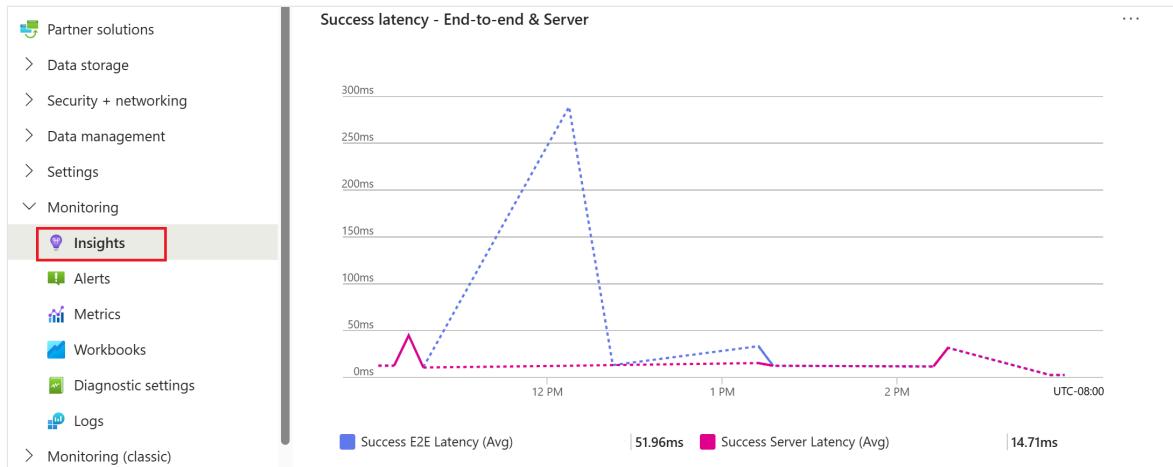
Azure Storage provides two latency metrics for block blobs in the Azure portal:

- End-to-end (E2E) latency measures the interval from when Azure Storage receives the first packet of the request until Azure Storage receives a client acknowledgment on the last packet of the response.
- Server latency measures the interval from when Azure Storage receives the last packet of the request until the first packet of the response is returned from Azure Storage.

To view latency metrics, navigate to your storage resource in the Azure portal:

- On the left navigation window, select **Insights** from the **Monitoring** drop-down menu.

- The insights tab opens a window that includes a chart showing both E2E and Server latency metrics:

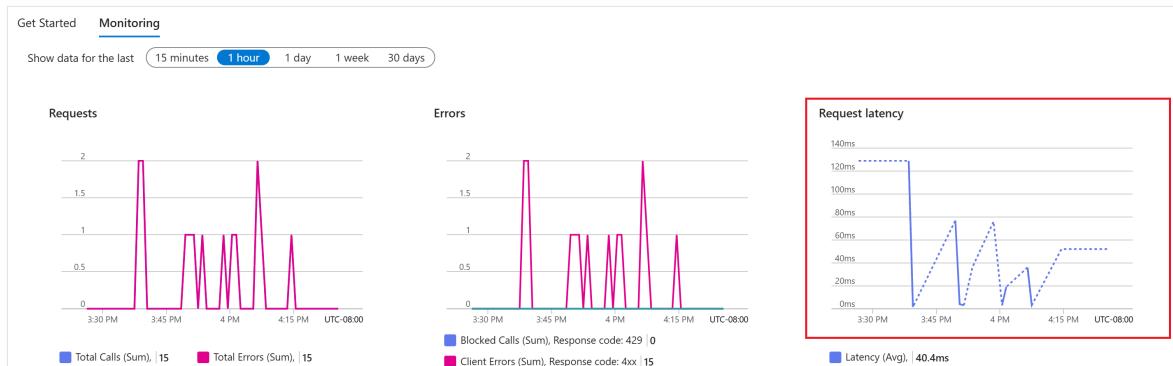


For more information, see [Latency in Blob storage](#).

## Check monitoring metrics for your resource

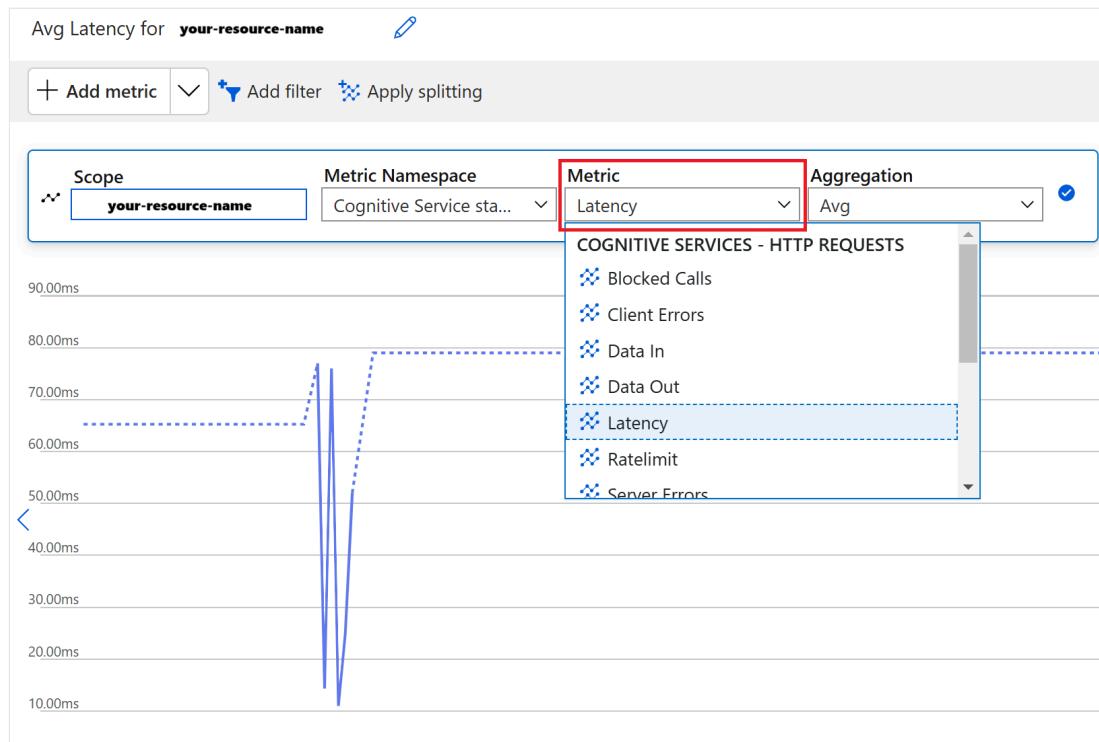
Azure portal monitors offer insights into your applications to enhance their performance and availability. There are several tools that you can use to monitor your app's performance in the Azure portal:

- On the **Overview** page, select **Monitoring**, select the time period, and review the **Request latency** metrics on page.



- On the left navigation window, select **Metrics** from the **Monitoring** drop-down menu.

- In the main window, select **+Add metric**.
- Keep the **Scope** and **Metric Namespace** fields unchanged. Add the **Latency** parameter to the **Metric** field and adjust the **Aggregation** field as needed.



## Set a latency alert in the Azure portal

Alerts assist you in identifying and resolving issues by providing proactive notifications when Azure Monitor data suggests a potential issue. An alert rule keeps an eye on your data and notifies you when set criteria are met on your specified resource. You can set up an alert in the Azure portal as follows:

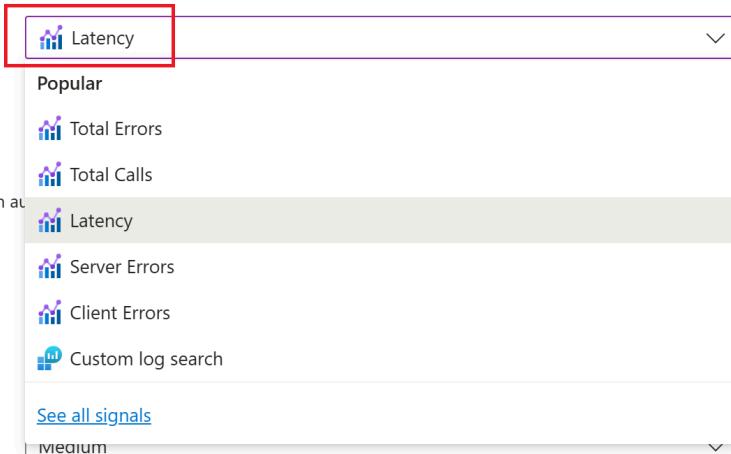
1. On the left navigation window, select **Alerts** from the **Monitoring** drop-down menu.
2. Select the **Create alert rule** button.
3. In the new window that opens, select **Latency** from the **Select a signal** drop-down menu.

## Create an alert rule

Scope   Condition   Actions   Details   Tags   Review + create

Configure when the alert rule should trigger by selecting a signal and defining its logic.

Signal name \* ⓘ



4. Configure the alert by completing the fields on the page.

5. After you complete the configuration, select **Review + create**

## Contact us

If you're unable to resolve long latency issue, [email us](#) with the following information:

- Model Name
- Version
- Subscription ID
- Resource ID
- Timestamp and issue description
- Request IDs of the concerning operations (if possible)
- Logs
- Sample files
- JSON file (output/analyze results)
- Training set (if it's a training issue related to custom neural models)

For more assistance, you can also use the feedback widget at the bottom of any Microsoft Learn page.

---

# Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence Studio custom projects

Article • 03/14/2025

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)

[Document Intelligence Studio](#) is an online tool for visually exploring, understanding, and integrating features from the Document Intelligence service in your applications. This quickstart aims to give you a guide of setting up a custom project in Document Intelligence Studio.

## Prerequisites for new users

For details on subscription, resource, and authentication setup, see [Get started with Document Intelligence Studio](#).

## Additional prerequisites for custom projects

In addition to the Azure account and a Document Intelligence or Azure AI services resource, you need:

### Azure Blob Storage container

A standard performance [Azure Blob Storage account](#). You create containers to store and organize your training documents within your storage account. If you don't know how to create an Azure storage account with a container, following these quickstarts:

- [Create a storage account](#). When creating your storage account, make sure to select Standard performance in the **Instance details** → **Performance** field.
- [Create a container](#). When creating your container, set the **Public access level** field to **Container** (anonymous read access for containers and blobs) in the **New Container** window.

### Azure role assignments

For custom projects, the following role assignments are required for different scenarios.

- Basic

- **Cognitive Services User:** You need this role for Document Intelligence or Azure AI services resource to train the custom model or do analysis with trained models.
- **Storage Blob Data Contributor:** You need this role for the Storage Account to create a project and label data.
- Advanced
  - **Storage Account Contributor:** You need this role for the Storage Account to set up CORS settings (this action is a one-time effort if the same storage account is reused).
  - **Contributor:** You need this role to create a resource group and resources.

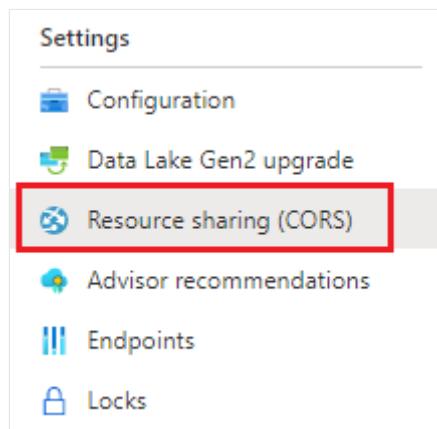
① Note

If local (key-based) authentication is disabled for your Document Intelligence service resource and storage account, be sure to obtain **Cognitive Services User** and **Storage Blob Data Contributor** roles respectively, so you have enough permissions to use Document Intelligence Studio. The **Storage Account Contributor** and **Contributor** roles only allow you to list keys but don't give you permission to use the resources when key-access is disabled.

## Configure CORS

[CORS \(Cross Origin Resource Sharing\)](#) needs to be configured on your Azure storage account for it to be accessible from the Document Intelligence Studio. To configure CORS in the Azure portal, you need access to the CORS tab of your storage account.

1. Select the CORS tab for the storage account.



2. Start by creating a new CORS entry in the Blob service.

3. Set the **Allowed origins** to <https://documentintelligence.ai.azure.com>.

The screenshot shows the 'Blob service' tab selected in the top navigation bar. Under the 'Cors' section, there is a table with five columns: 'Allowed origins', 'Allowed methods', 'Allowed headers', 'Exposed headers', and 'Max age'. The 'Allowed origins' column contains the URL 'https://documentintelligence.ai.azure...'. The 'Allowed methods' column has a dropdown menu showing '8 selected'. The 'Allowed headers' and 'Exposed headers' columns both have an asterisk (\*) in each input field. The 'Max age' column contains the value '120'. A red box highlights the 'Allowed origins' input field.

### Tip

You can use the wildcard character '\*' rather than a specified domain to allow all origin domains to make requests via CORS.

4. Select all the available 8 options for **Allowed methods**.
5. Approve all **Allowed headers** and **Exposed headers** by entering an \* in each field.
6. Set the **Max Age** to 120 seconds or any acceptable value.
7. To save the changes, select the save button at the top of the page.

CORS should now be configured to use the storage account from Document Intelligence Studio.

## Sample documents set

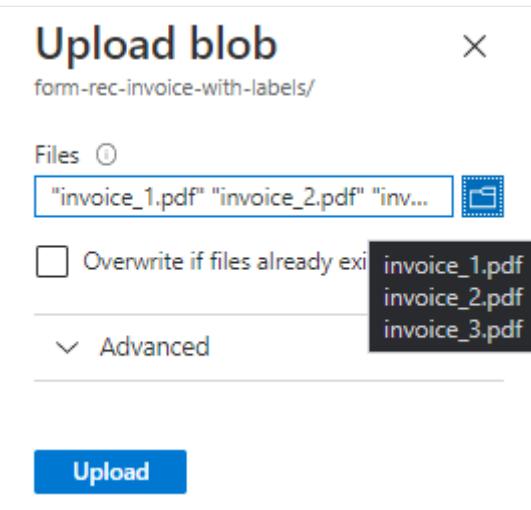
1. Sign in to the [Azure portal](#) and navigate to **Your storage account > Data storage > Containers**.

The screenshot shows the 'Data storage' section of the Azure portal. It lists four items: 'Containers' (highlighted with a red box), 'File shares', 'Queues', and 'Tables'. Each item has a corresponding icon next to its name.

2. Select a **container** from the list.
3. Select **Upload** from the menu at the top of the page.

The screenshot shows the top navigation menu for a storage container. The 'Upload' option is highlighted with a red box. Other menu items include 'Change access level', 'Refresh', 'Delete', 'Change tier', 'Acquire lease', 'Break lease', 'View snapshots', and 'Create snapshot'.

4. The **Upload blob** window appears.
5. Select your files to upload.



### ⓘ Note

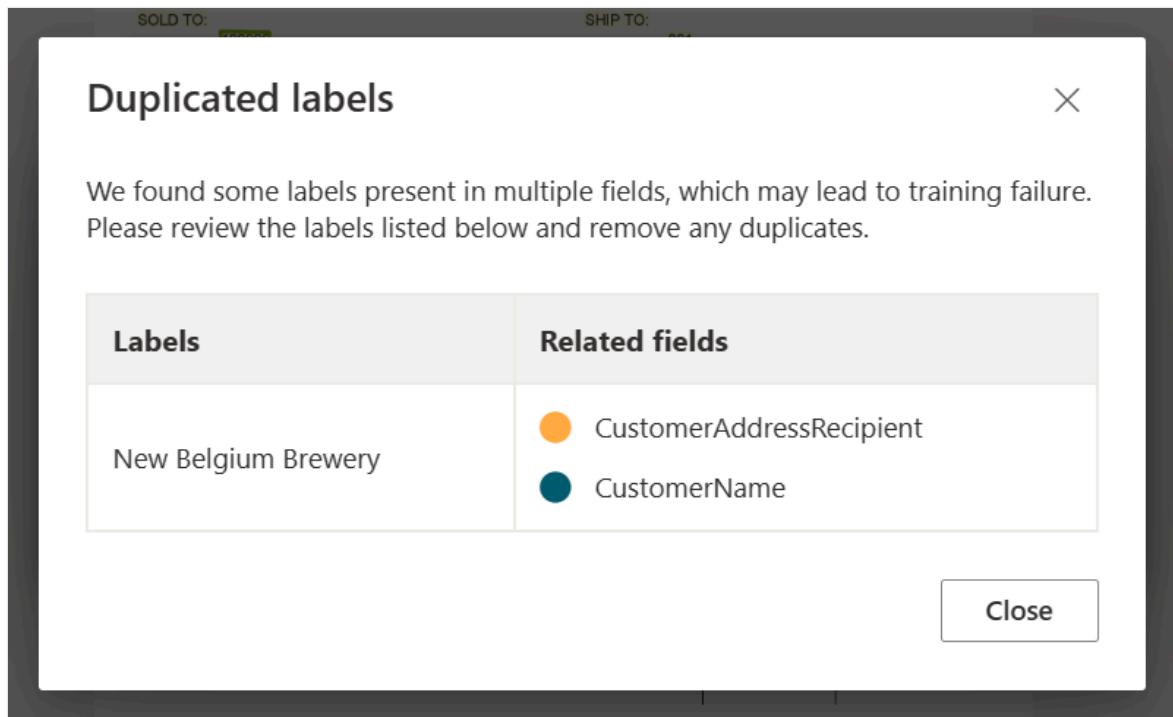
By default, the Studio uses documents that are located at the root of your container. However, you can use data organized in folders by specifying the folder path in the Custom form project creation steps. See [Organize your data in subfolders](#)

## Use Document Intelligence Studio features

### Auto label documents with prebuilt models or one of your own models

- In custom extraction model labeling page, you can now auto label your documents using one of Document Intelligent Service prebuilt models or your trained models.

- For some documents, duplicate labels after running autolabel are possible. Make sure to modify the labels so that there are no duplicate labels in the labeling page afterwards.



## Auto label tables

- In custom extraction model labeling page, you can now auto label the tables in the document without having to label the tables manually.

Azure AI | Document Intelligence Studio

Custom extraction model

Hello-Custom

Label data

Run layout Auto label Draw region

Received from: Liberty's Delightful Sinfu Bakery & Café  
765 Halifax St, Clearwater, FL 33756  
Our reference: 3456623 Your reference: 2334566

Received from: 8555 Indian Summer Ave.  
New Haven, CT 06511  
Name: Summer River  
Tel: 3456623  
E-mail: [arun@libertydelightfulsinfu.com](mailto:arun@libertydelightfulsinfu.com)

Booking Confirmation – ORIGINAL

Our reference: 3456623  
Your reference: 2334566  
BLN/NO: EURH234  
Summary: #5x72

Booking date: 05-Dec-2018 Contract No: 334566

Export Common

Booking Confirmation – ORIGINAL

Our reference: 3456623  
Your reference: 2334566  
BLN/NO: EURH234  
Summary: #5x72

Booking date: 05-Dec-2018 Contract No: 334566

Export Common

Export empty pick up depots(s)

Rows: 3, Columns: 5

18 Queen Street Hoboken, NJ 07030	52 West Trenton St. Harleysville, PA	cause science slow	ETD 09-Dec-2018	ETA 09-Dec-2020
9 Ketch Harbour Ave., 12000town, NJ	75 Fawn Street Peabody, MA 01960	tone late spoken	12-Dec-2018	19-Dec-2020
			10:00	

Deadline	Table	Location	Date/Time (local)	Required action
Flight	Harleysville (PA)	10-Dec-2019	08-Dec-2019	Nobody likes a pig wearing clothes.
Round	Harleysville (PA)	10-Dec-2019	09-Dec-2019	Two more days and all his problems would be solved.
Accent	Harleysville (PA)	11-Dec-2019	10-Dec-2019	
Monkey	Harleysville (PA)	11-Dec-2019	11-Dec-2019	
Route	Harleysville (PA)	11-Dec-2019	11-Dec-2019	Peanuts don't grow on trees.

< 1 of 1 >

Privacy & Cookies © Microsoft 2022

## Add test files directly to your training dataset

- Once you train a custom extraction model, make use of the test page to improve your model quality by uploading test documents to training dataset if needed.
- If a low confidence score is returned for some labels, make sure to correctly label your content. If not, add them to the training dataset and relabel to improve the model quality.

Azure AI | Document Intelligence Studio

Custom extraction model

Hello-Custom

Test model test-model

Run analysis Analyze options

Fields Result Code

receipt.png

DocType: test-model

address #1 24.80%  
123 Main Street Redmond, WA 98052

name #1 58.10%  
Contoso

123 Main Street Redmond, WA 98052

987-654-3210

6/10/2019 13:59

Sales Associate: Paul

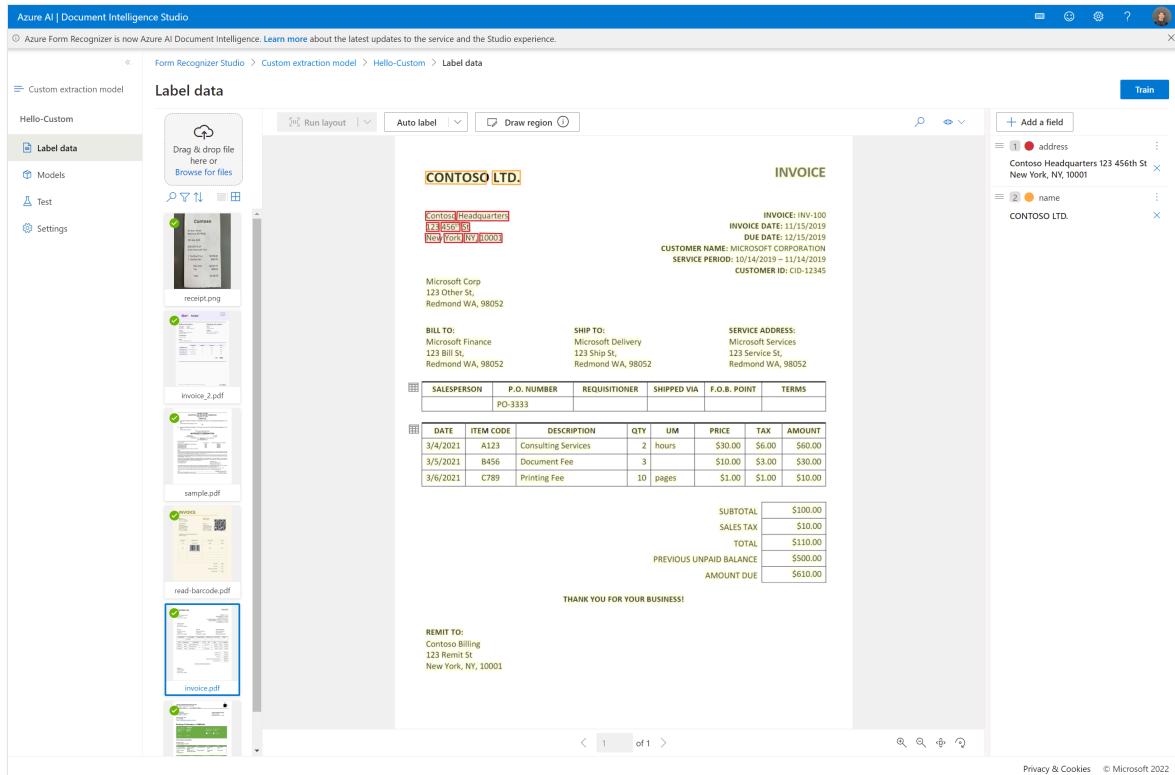
2 Surface Pro 6	\$1,998.00
3 Surface Pen	\$299.97
Sub-Total	\$2,297.97
Tax	\$218.31
Total	\$2,516.28

< 1 of 1 >

Privacy & Cookies © Microsoft 2022

# Make use of the document list options and filters in custom projects

- Use the custom extraction model labeling page to navigate through your training documents with ease by making use of the search, filter, and sort by feature.
- Utilize the grid view to preview documents or use the list view to scroll through the documents more easily.



## Project sharing

Share custom extraction projects with ease. For more information, see [Project sharing with custom models](#).

## Next steps

- Follow our [Document Intelligence v3.1 migration guide](#) to learn the differences from the previous version of the REST API.
- Explore our [v4.0 SDK quickstarts](#) to try the v3.0 features in your applications using the new client libraries.
- Refer to our [v4.0 REST API quickstarts](#) to try the v3.0 features using the new REST API.

Get started with the Document Intelligence Studio [↗](#).

---

# Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Get started: Document Intelligence Studio

Article • 03/19/2025

Document Intelligence Studio [↗](#) is an online tool for visually exploring, understanding, and integrating features from the Document Intelligence service in your applications. You can get started by exploring the pretrained models with sample or your own documents. You can also create projects to build custom template models and reference the models in your applications.

<https://www.microsoft.com/en-us/videoplayer/embed/RE56n49?postJslIMsg=true> ↗

## Prerequisites for new users

To use Document Intelligence Studio, you need to acquire the following assets from the Azure portal:

- Azure subscription - [Create one for free](#) ↗.
- An Azure AI services or Document Intelligence resource. Once you have your Azure subscription, create a [single-service](#) ↗ or [Azure AI multi-service](#) ↗ resource, in the Azure portal, to get your key and endpoint.
- You can use the free pricing tier (`F0`) to try the service, and upgrade later to a paid tier for production.

### 💡 Tip

Create an Azure AI services resource if you plan to access multiple Azure AI services under a single endpoint/key. For Document Intelligence access only, create a Document Intelligence resource. You need a single-service resource if you intend to use [Microsoft Entra authentication](#).

Document Intelligence now supports Azure Active Directory (Azure AD) token authentication in addition to local (key-based) authentication when accessing the Document Intelligence resources and storage accounts. Be sure to follow below instructions to set up correct access roles, especially if your resources are applied with `DisableLocalAuth` policy.

There are added prerequisites for using custom models in Document Intelligence Studio. Refer to the [documentation](#) for step by step guidance.

# Authorization policies

Your organization can opt to disable local authentication and enforce Microsoft Entra (formerly Azure Active Directory) authentication for Azure AI Document Intelligence resources and Azure blob storage.

- Microsoft Entra authentication requires that key based authorization is disabled. After key access is disabled, Microsoft Entra ID is the only available authorization method.
- Microsoft Entra allows granting minimum privileges and granular control for Azure resources.

For more information, see the following guidance:

- [Disable local authentication for Azure AI Services](#).
- [Prevent Shared Key authorization for an Azure Storage account](#)

## ⓘ Note

If local (key-based) authentication is disabled for your Document Intelligence service resource, be sure to obtain **Cognitive Services User** role and your Azure AD token to authenticate requests on Document Intelligence Studio. The **Contributor** role only allows you to list keys but doesn't give you permission to use the resource when key-access is disabled.

- **Designating role assignments.** Document Intelligence Studio basic access requires the [Cognitive Services User](#) role. For more information, see [Document Intelligence role assignments](#).

## ⓘ Important

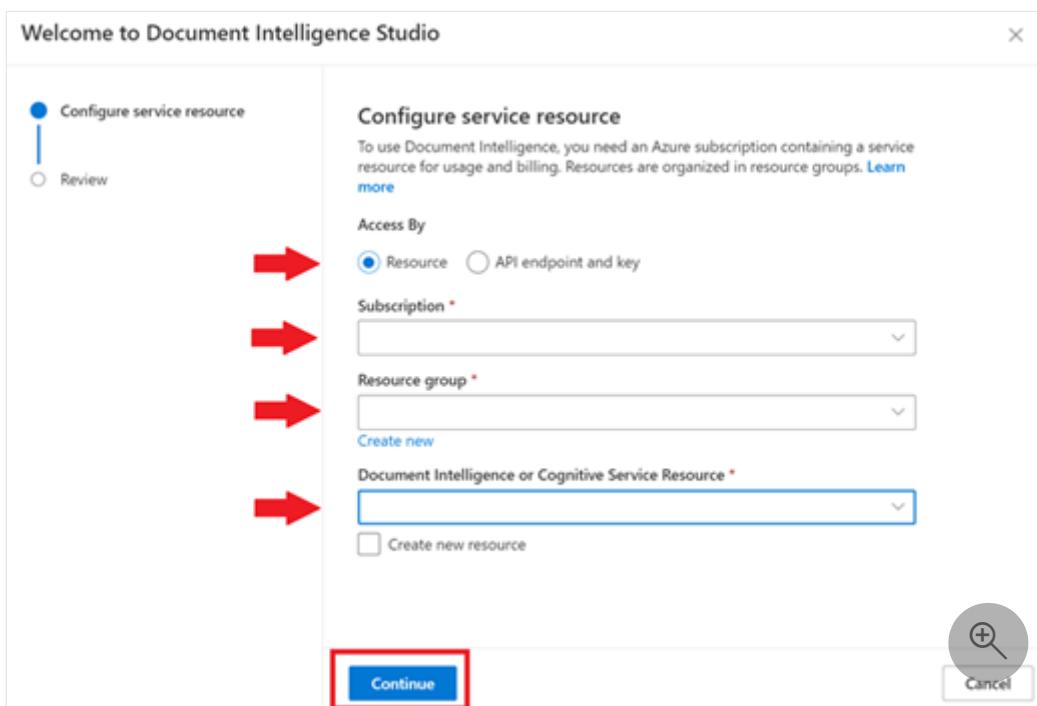
- Make sure you have the **Cognitive Services User role**, and not the Cognitive Services Contributor role when setting up Microsoft Entra ID authentication.
- **✓ Cognitive Services User:** you need this role to Document Intelligence or Azure AI services resource to enter the analyze page.
- **✓ Contributor:** you need this role to create resource group, Document Intelligence service, or Azure AI services resource.
- In Azure context, Contributor role can only perform actions to control and manage the resource itself, including listing the access keys.

- User accounts with a Contributor role are only able to access the Document Intelligence service by calling with access keys. However, when setting up access with Microsoft Entra ID, key-access is disabled and **Cognitive Services User** role is required for an account to use the resources.

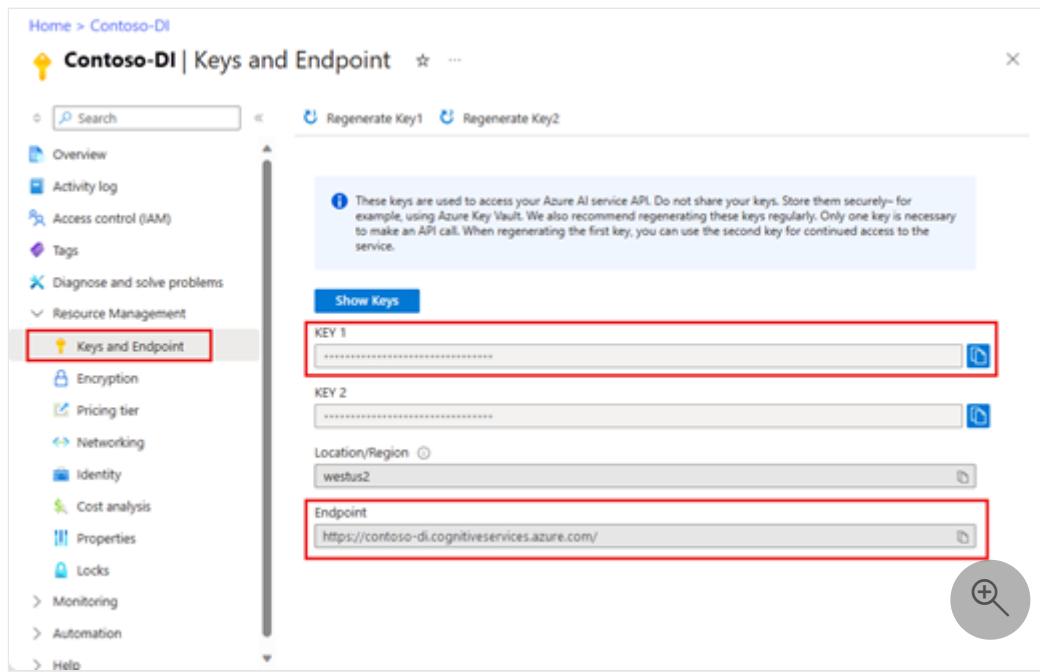
## Authentication in Studio

Navigate to the [Document Intelligence Studio](#). If it's your first time logging in, a popup window appears prompting you to configure your service resource. In accordance with your organization's policy, you have one or two options:

- Microsoft Entra authentication: access by Resource (recommended).**
  - Choose your existing subscription.
  - Select an existing resource group within your subscription or create a new one.
  - Select your existing Document Intelligence or Azure AI services resource.



- Local authentication: access by API endpoint and key.**
  - Retrieve your endpoint and key from the Azure portal.
  - Go to the overview page for your resource and select **Keys and Endpoint** from the left navigation bar.
  - Enter the values in the appropriate fields.



- After validating the scenario in the Document Intelligence Studio, use the [C#](#), [Java](#), [JavaScript](#), or [Python](#) client libraries or the [REST API](#) to get started incorporating Document Intelligence models into your own applications.

## Try a Document Intelligence model

To learn more about the available Document Intelligence models, see our [model support](#) page.

- Once your resource is configured, you can try the different models offered by Document Intelligence Studio. From the front page, select any Document Intelligence model to try using with a no-code approach.
- To test any of the document analysis or prebuilt models, select the model and use one of the sample documents or upload your own document to analyze. The analysis result is displayed at the right in the content-result-code window.
- Custom models need to be trained on your documents. See [custom models overview](#) for an overview of custom models.
- After validating the scenario in the Document Intelligence Studio, use the [C#](#), [Java](#), [JavaScript](#), or [Python](#) client libraries or the [REST API](#) to get started incorporating Document Intelligence models into your own applications.

## View resource details

To view resource details such as name and pricing tier, select the **Settings** icon in the top-right corner of the Document Intelligence Studio home page and select the

Resource tab. If you have access to other resources, you can switch resources as well.

The screenshot shows the 'Settings' page in the 'Form Recognizer Studio'. At the top left, there is a breadcrumb navigation: 'Form Recognizer Studio > Settings'. Below the breadcrumb, there are two tabs: 'Directory' and 'Resource'. The 'Resource' tab is highlighted with a red box around it. On the right side of the page, there is a circular icon with a magnifying glass and a plus sign (+). Below the tabs, a note reads: 'Resources are your unique aliases for the service and allow usage and billing. Choose your default resource. [Learn more about Azure resources](#).'. The entire screenshot is enclosed in a light gray border.

With Document Intelligence, you can quickly automate your data processing in applications and workflows, easily enhance data-driven strategies, and skillfully enrich document search capabilities.

## Manage third-party settings for Studio access

Edge:

- Go to **Settings** for Microsoft Edge
- Search for "**third-party**"
- Go to **Manage and delete cookies and site data**
- Turn off the setting of **Block third-party cookies**

Chrome:

- Go to **Settings** for Chrome
- Search for "**Third-party**"
- Under **Default behavior**, select **Allow third-party cookies**

Firefox:

- Go to **Settings** for Firefox
- Search for "**cookies**"
- Under **Enhanced Tracking Protection**, select **Manage Exceptions**
- Add exception for `https://documentintelligence.ai.azure.com` or the Document Intelligence Studio URL of your environment

Safari:

- Choose **Safari > Preferences**
- Select **Privacy**
- Deselect **Block all cookies**

## Troubleshooting

Scenario	Cause	Resolution
<p>You receive the error message <code>Form Recognizer Not Found</code> when opening a custom project.</p>	<p>Your Document Intelligence resource, bound to the custom project was deleted or moved to another resource group.</p>	<p>There are two ways to resolve this problem:</p> <ul style="list-style-type: none"> <li>• Re-create the Document Intelligence resource under the same subscription and resource group with the same name.</li> <li>• Re-create a custom project with the migrated Document Intelligence resource and specify the same storage account.</li> </ul>
<p>You receive the error message <code>PermissionDenied</code> when using prebuilt apps or opening a custom project.</p>	<p>The principal doesn't have access to API/Operation when analyzing against prebuilt models or opening a custom project. It's likely the local (key-based) authentication is disabled for your Document Intelligence resource don't have enough permission to</p>	<p>Reference <a href="#">Azure role assignments</a> to configure your access roles.</p>

Scenario	Cause	Resolution
	access the resource.	
You receive the error message <code>AuthorizationPermissionMismatch</code> when opening a custom project.	The request isn't authorized to perform the operation using the designated permission. It's likely the local (key-based) authentication is disabled for your storage account and you don't have the granted permission to access the blob data.	Reference <a href="#">Azure role assignments</a> to configure your access roles.

## Next steps

- Learn how to create custom projects in Document Intelligence Studio
- Get started with Document Intelligence client libraries

---

## Feedback

Was this page helpful?

 Yes

 No

Provide product feedback  | Get help at Microsoft Q&A

# Get started with Document Intelligence

Article • 04/11/2025

## ⓘ Important

- Azure Cognitive Services Form Recognizer is now Azure AI Document Intelligence.
- Some platforms are still awaiting the renaming update.
- All mention of Form Recognizer or Document Intelligence in our documentation refers to the same Azure service.

This content applies to:  v4.0 (GA) Earlier versions:  v3.1 (GA)  v3.0 (GA)

- Get started with Azure AI Document Intelligence latest stable version v4.0 2024-11-30 (GA).
- Azure AI Document Intelligence / Form Recognizer is a cloud-based Azure AI service that uses machine learning to extract key-value pairs, text, tables, and key data from your documents.
- You can easily integrate document processing models into your workflows and applications by using a programming language SDK or calling the REST API.
- We recommend that you use the free service while you're learning the technology for this quickstart. Remember that the number of free pages is limited to 500 per month.

To learn more about the API features and development options, visit our [Overview](#) page.

[Client library](#) | [REST API reference](#) | [Package ↗](#) | [Samples ↗](#) | [Supported REST API version](#)

In this quickstart, use the following features to analyze and extract data and values from forms and documents:

- **Layout model**—Analyze and extract tables, lines, words, and selection marks like radio buttons and check boxes in documents, without the need to train a model.
- **Prebuilt model**—Analyze and extract common fields from specific document types using a prebuilt model.

## Prerequisites

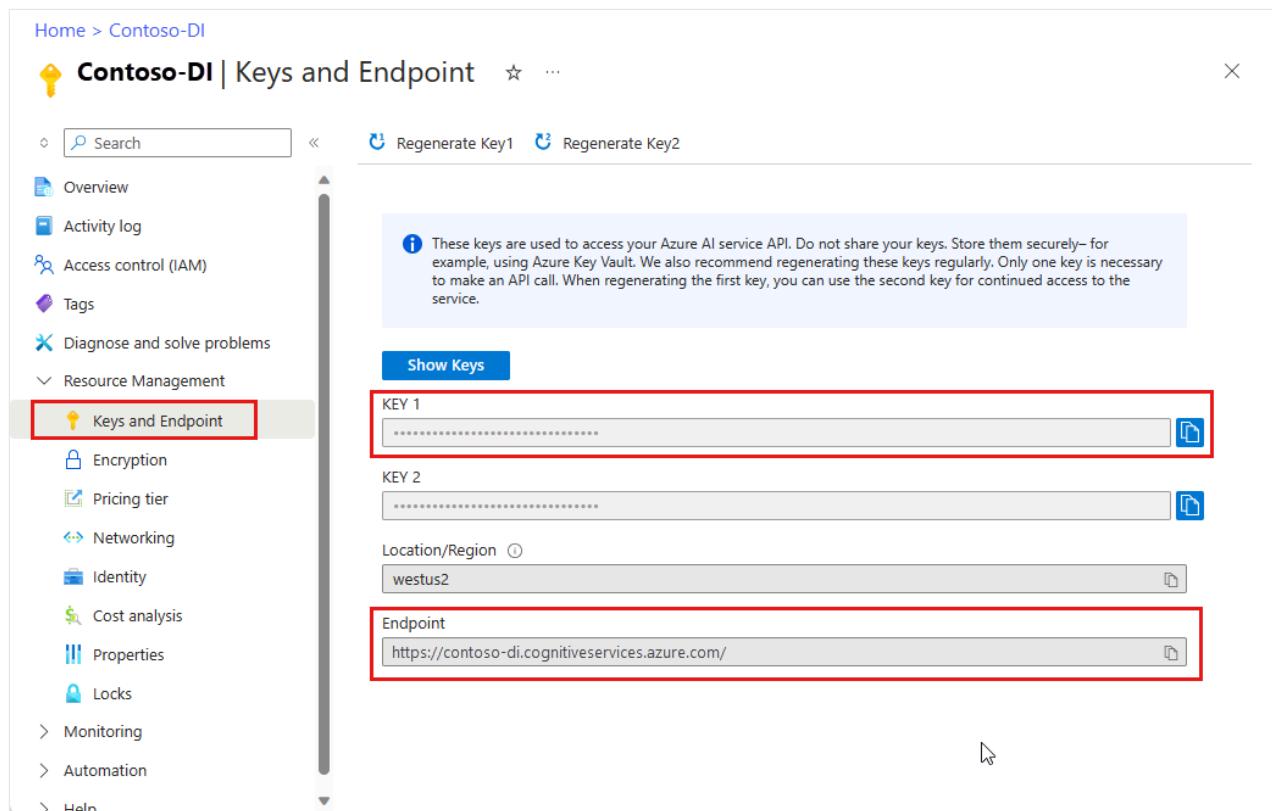
- Azure subscription - [Create one for free ↗](#).
- The current version of [Visual Studio IDE ↗](#).

- An Azure AI services or Document Intelligence resource. Once you have your Azure subscription, create a [single-service](#) or [Azure AI multi-service](#) resource, in the Azure portal, to get your key and endpoint.
- You can use the free pricing tier (F0) to try the service, and upgrade later to a paid tier for production.

### Tip

Create an Azure AI services resource if you plan to access multiple Azure AI services under a single endpoint/key. For Document Intelligence access only, create a Document Intelligence resource. You need a single-service resource if you intend to use [Microsoft Entra authentication](#).

- After your resource deploys, select **Go to resource**. You need the key and endpoint from the resource you create to connect your application to the Document Intelligence API. You paste your key and endpoint into the code later in the quickstart:



## Set up

1. Start Visual Studio.
2. On the start page, choose Create a new project.

# Visual Studio 2022

## Open recent

As you use Visual Studio, any projects, folders, or files that you open will show up here for quick access.

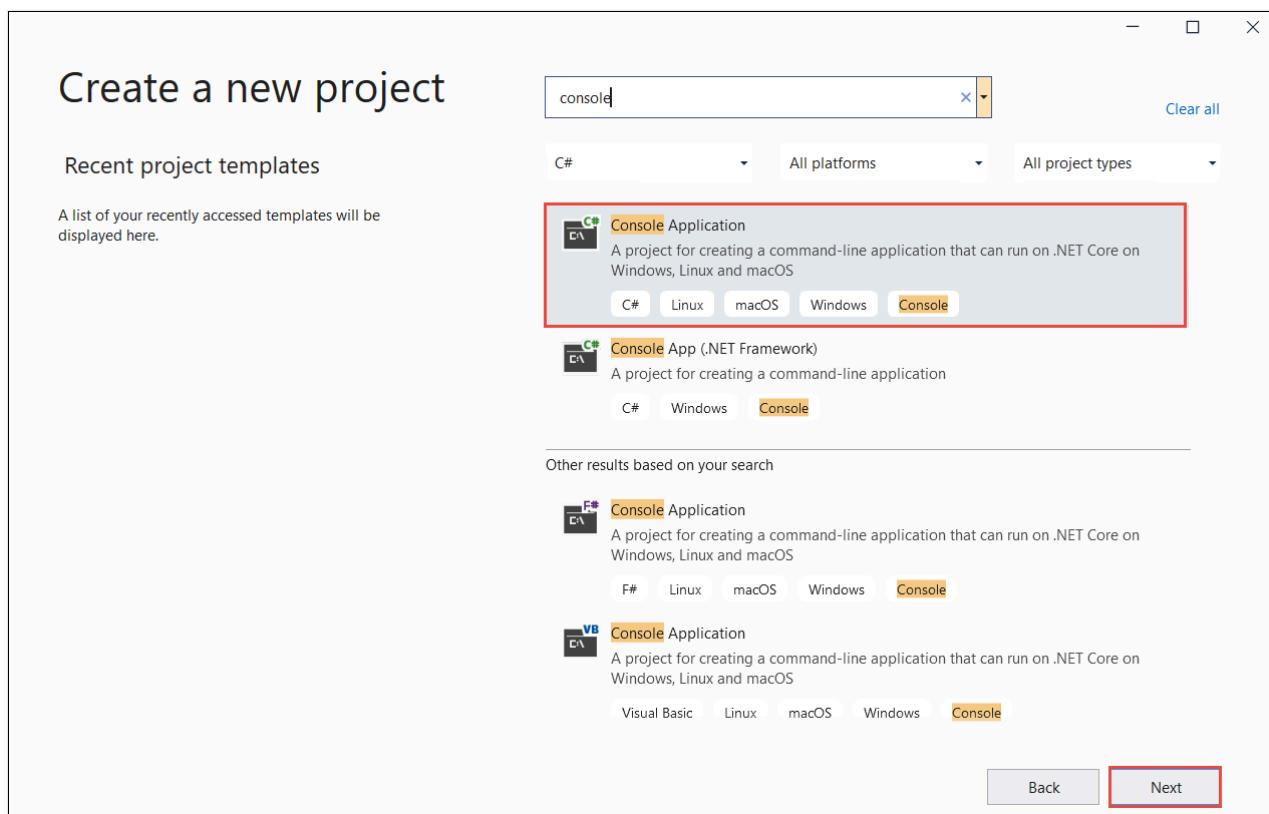
You can pin anything that you open frequently so that it's always at the top of the list.

## Get started

-  **Clone a repository**  
Get code from an online repository like GitHub or Azure DevOps
-  **Open a project or solution**  
Open a local Visual Studio project or .sln file
-  **Open a local folder**  
Navigate and edit code within any folder
-  **Create a new project**  
Choose a project template with code scaffolding to get started

[Continue without code →](#)

3. On the **Create a new project** page, enter **console** in the search box. Choose the **Console Application** template, then choose **Next**.



4. In the **Configure your new project** dialog window, enter `doc_intel_quickstart` in the Project name box. Then choose **Next**.

5. In the **Additional information** dialog window, select **.NET 8.0 (Long-term support)**, and then select **Create**.

## Additional information

### Console App

C#

Linux

macOS

Windows

Console

#### Framework i

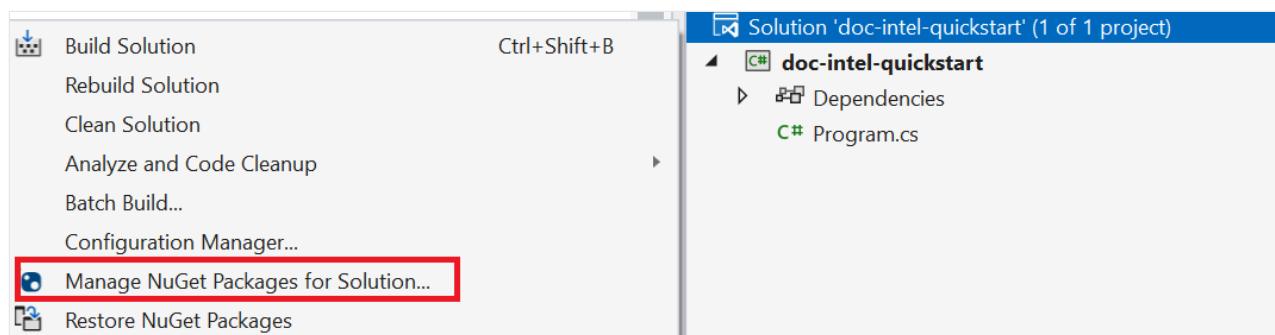
.NET 8.0 (Long Term Support)

Do not use top-level statements i

Enable native AOT publish i

## Install the client library with NuGet

1. Right-click on your `doc_intel_quickstart` project and select **Manage NuGet Packages...** .



2. Select the **Browse** tab and type `Azure.AI.DocumentIntelligence`.

3. Select the **Include prerelease** checkbox.



4. Choose a version from the dropdown menu and install the package in your project.

## Build your application

To interact with the Document Intelligence service, you need to create an instance of the `DocumentIntelligenceClient` class. To do so, you create an `AzureKeyCredential` with your `key` from the Azure portal and a `DocumentIntelligenceClient` instance with the `AzureKeyCredential` and your Document Intelligence `endpoint`.

### Note

- Starting with .NET 6, new projects using the `console` template generate a new program style that differs from previous versions.
- The new output uses recent C# features that simplify the code you need to write.
- When you use the newer version, you only need to write the body of the `Main` method. You don't need to include top-level statements, global using directives, or implicit using directives.
- For more information, see [New C# templates generate top-level statements](#).

1. Open the `Program.cs` file.

2. Delete the existing code, including the line `Console.WriteLine("Hello World!")`, and select one of the following code samples to copy and paste into your application's `Program.cs` file:

- [Layout model](#)
- [Prebuilt model](#)

### Important

We recommend Microsoft Entra ID authentication with [managed identities for Azure resources](#) to avoid storing credentials with your applications that run in the cloud.

Use API keys with caution. Don't include the API key directly in your code, and never post it publicly. If using API keys, store them securely in Azure Key Vault, rotate the keys regularly, and restrict access to Azure Key Vault using role based access control and network access restrictions. For more information about using API keys securely in your apps, see [API keys with Azure Key Vault](#).

For more information about AI services security, see [Authenticate requests to Azure AI services](#).

## Layout model

Extract text, selection marks, text styles, table structures, and bounding region coordinates from documents.

- ✓ For this example, you'll need a **document file from a URI**. You can use our [sample document ↗](#) for this quickstart.
- ✓ We've added the file URI value to the `Uri fileUri` variable at the top of the script.

- ✓ To extract the layout from a given file at a URI, use the `StartAnalyzeDocumentFromUri` method and pass `prebuilt-layout` as the model ID. The returned value is an `AnalyzeResult` object containing data from the submitted document.

Add the following code sample to the `Program.cs` file. Make sure you update the key and endpoint variables with values from your Azure portal Document Intelligence instance:

C#

```
using Azure;
using Azure.AI.DocumentIntelligence;

//set `<your-endpoint>` and `<your-key>` variables with the values from the Azure
portal to create your `AzureKeyCredential` and `DocumentIntelligenceClient`
instance
string endpoint = "<your-endpoint>";
string key = "<your-key>";
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentIntelligenceClient client = new DocumentIntelligenceClient(new
Uri(endpoint), credential);

//sample document
Uri fileUri = new Uri ("https://raw.githubusercontent.com/Azure-Samples/cognitive-
services-REST-api-samples/master/curl/form-recognizer/sample-layout.pdf");

AnalyzeDocumentContent content = new AnalyzeDocumentContent()
{
    UrlSource= fileUri
};

Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-layout", content);

AnalyzeResult result = operation.Value;

foreach (DocumentPage page in result.Pages)
{
    Console.WriteLine($"Document Page {page.PageNumber} has {page.Lines.Count}
line(s), {page.Words.Count} word(s), "
    $" and {page.SelectionMarks.Count} selection mark(s).");

    for (int i = 0; i < page.Lines.Count; i++)
    {
        DocumentLine line = page.Lines[i];

        Console.WriteLine($"    Line {i}:");
        Console.WriteLine($"        Content: '{line.Content}'");

        Console.Write("        Bounding polygon, with points ordered clockwise:");
        for (int j = 0; j < line.Polygon.Count; j += 2)
        {
```

```

        Console.WriteLine($" ({line.Polygon[j]}, {line.Polygon[j + 1]})");
    }

    Console.WriteLine();
}

for (int i = 0; i < page.SelectionMarks.Count; i++)
{
    DocumentSelectionMark selectionMark = page.SelectionMarks[i];

    Console.WriteLine($"  Selection Mark {i} is {selectionMark.State}.");
    Console.WriteLine($"    State: {selectionMark.State}");

    Console.Write("    Bounding polygon, with points ordered clockwise:");
    for (int j = 0; j < selectionMark.Polygon.Count; j++)
    {
        Console.WriteLine($" ({selectionMark.Polygon[j]}, {selectionMark.Polygon[j + 1]}"));
    }

    Console.WriteLine();
}
}

for (int i = 0; i < result.Paragraphs.Count; i++)
{
    DocumentParagraph paragraph = result.Paragraphs[i];

    Console.WriteLine($"Paragraph {i}:");
    Console.WriteLine($"  Content: {paragraph.Content}");

    if (paragraph.Role != null)
    {
        Console.WriteLine($"    Role: {paragraph.Role}");
    }
}

foreach (DocumentStyle style in result.Styles)
{
    // Check the style and style confidence to see if text is handwritten.
    // Note that value '0.8' is used as an example.

    bool isHandwritten = style.IsHandwritten.HasValue && style.IsHandwritten == true;

    if (isHandwritten && style.Confidence > 0.8)
    {
        Console.WriteLine($"Handwritten content found:");

        foreach (DocumentSpan span in style.Spans)
        {
            var handwrittenContent = result.Content.Substring(span.Offset, span.Length);
            Console.WriteLine($"  {handwrittenContent}");
        }
    }
}
```

```
        }

    for (int i = 0; i < result.Tables.Count; i++)
    {
        DocumentTable table = result.Tables[i];

        Console.WriteLine($"Table {i} has {table.RowCount} rows and
{table.ColumnCount} columns.");

        foreach (DocumentTableCell cell in table.Cells)
        {
            Console.WriteLine($" Cell ({cell.RowIndex}, {cell.ColumnIndex}) is a
'{cell.Kind}' with content: {cell.Content}");
        }
    }
}
```

## Run your application

Once you add a code sample to your application, choose the green **Start** button next to `formRecognizer_quickstart` to build and run your program, or press **F5**.



## Prebuilt model

Analyze and extract common fields from specific document types using a prebuilt model. In this example, we analyze an invoice using the `prebuilt-invoice` model.

### Tip

You aren't limited to invoices—there are several prebuilt models to choose from, each of which has its own set of supported fields. The model to use for the `analyze` operation depends on the type of document to be analyzed. See [model data extraction](#).

- ✓ Analyze an invoice using the prebuilt-invoice model. You can use our [sample invoice document](#) for this quickstart.
- ✓ We've added the file URI value to the `Uri invoiceUri` variable at the top of the `Program.cs` file.
- ✓ To analyze a given file at a URI, use the `StartAnalyzeDocumentFromUri` method and pass `prebuilt-invoice` as the model ID. The returned value is an `AnalyzeResult` object containing data from the submitted document.

- ✓ For simplicity, all the key-value pairs that the service returns are not shown here. To see the list of all supported fields and corresponding types, see our [Invoice](#) concept page.

Add the following code sample to your Program.cs file. Make sure you update the key and endpoint variables with values from your Azure portal Document Intelligence instance:

C#

```
using Azure;
using Azure.AI.DocumentIntelligence;

//set `<your-endpoint>` and `<your-key>` variables with the values from the Azure
portal to create your `AzureKeyCredential` and `DocumentIntelligenceClient`
instance
string endpoint = "<your-endpoint>";
string key = "<your-key>";
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentIntelligenceClient client = new DocumentIntelligenceClient(new
Uri(endpoint), credential);

//sample invoice document
Uri uriSource = new Uri("https://raw.githubusercontent.com/Azure-
Samples/cognitive-services-REST-api-samples/master/curl/form-recognizer/sample-
invoice.pdf");

Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-invoice", uriSource);

AnalyzeResult result = operation.Value;

for (int i = 0; i < result.Documents.Count; i++)
{
    Console.WriteLine($"Document {i}:");

    AnalyzedDocument document = result.Documents[i];

    if (document.Fields.TryGetValue("VendorName", out DocumentField
vendorNameField)
        && vendorNameField.FieldType == DocumentFieldType.String)
    {
        string vendorName = vendorNameField.ValueString;
        Console.WriteLine($"Vendor Name: '{vendorName}', with confidence
{vendorNameField.Confidence}");
    }

    if (document.Fields.TryGetValue("CustomerName", out DocumentField
customerNameField)
        && customerNameField.FieldType == DocumentFieldType.String)
    {
        string customerName = customerNameField.ValueString;
        Console.WriteLine($"Customer Name: '{customerName}', with confidence
{customerNameField.Confidence}");
    }
}
```

```

}

if (document.Fields.TryGetValue("Items", out DocumentField itemsField)
    && itemsField.FieldType == DocumentFieldType.List)
{
    foreach (DocumentField itemField in itemsField.ValueList)
    {
        Console.WriteLine("Item:");

        if (itemField.FieldType == DocumentFieldType.Dictionary)
        {
            IReadOnlyDictionary<string, DocumentField> itemFields =
itemField.ValueDictionary;

            if (itemFields.TryGetValue("Description", out DocumentField itemDescriptionField)
                && itemDescriptionField.FieldType == DocumentFieldType.String)
            {
                string itemDescription = itemDescriptionField.ValueString;
                Console.WriteLine($" Description: '{itemDescription}', with
confidence {itemDescriptionField.Confidence}");
            }

            if (itemFields.TryGetValue("Amount", out DocumentField itemAmountField)
                && itemAmountField.FieldType == DocumentFieldType.Currency)
            {
                CurrencyValue itemAmount = itemAmountField.ValueCurrency;
                Console.WriteLine($" Amount: '{itemAmount.CurrencySymbol}
{itemAmount.Amount}', with confidence {itemAmountField.Confidence}");
            }
        }
    }
}

if (document.Fields.TryGetValue("SubTotal", out DocumentField subTotalField)
    && subTotalField.FieldType == DocumentFieldType.Currency)
{
    CurrencyValue subTotal = subTotalField.ValueCurrency;
    Console.WriteLine($"Sub Total: '{subTotal.CurrencySymbol}
{subTotal.Amount}', with confidence {subTotalField.Confidence}");
}

if (document.Fields.TryGetValue("TotalTax", out DocumentField totalTaxField)
    && totalTaxField.FieldType == DocumentFieldType.Currency)
{
    CurrencyValue totalTax = totalTaxField.ValueCurrency;
    Console.WriteLine($"Total Tax: '{totalTax.CurrencySymbol}
{totalTax.Amount}', with confidence {totalTaxField.Confidence}");
}

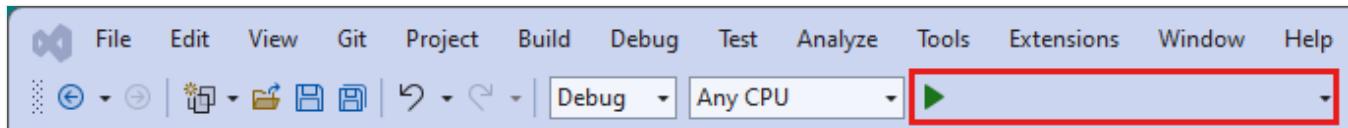
if (document.Fields.TryGetValue("InvoiceTotal", out DocumentField invoiceTotalField)
    && invoiceTotalField.FieldType == DocumentFieldType.Currency)
{

```

```
CurrencyValue invoiceTotal = invoiceTotalField.ValueCurrency;
Console.WriteLine($"Invoice Total: '{invoiceTotal.CurrencySymbol}
{invoiceTotal.Amount}', with confidence {invoiceTotalField.Confidence}");
}
}
```

## Run your application

Once you add a code sample to your application, choose the green **Start** button next to formRecognizer\_quickstart to build and run your program, or press **F5**.



That's it, congratulations!

In this quickstart, you used a document Intelligence model to analyze various forms and documents. Next, explore the Document Intelligence Studio and reference documentation to learn about Document Intelligence API in depth.

## Next steps

- For an enhanced experience and advanced model quality, try [Document Intelligence Studio](#).
- For v3.1 to v4.0 migration, see [Changelog Migration guides](#).
- [Find more samples on GitHub](#).

# Use Document Intelligence models

Article • 02/07/2025

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)   
[v2.1 \(GA\)](#)

In this guide, learn how to add Document Intelligence models to your applications and workflows. Use a programming language SDK of your choice or the REST API.

Azure AI Document Intelligence is a cloud-based Azure AI service that uses machine learning to extract key text and structure elements from documents. We recommend that you use the free service while you learn the technology. Remember that the number of free pages is limited to 500 per month.

Choose from the following Document Intelligence models and analyze and extract data and values from forms and documents:

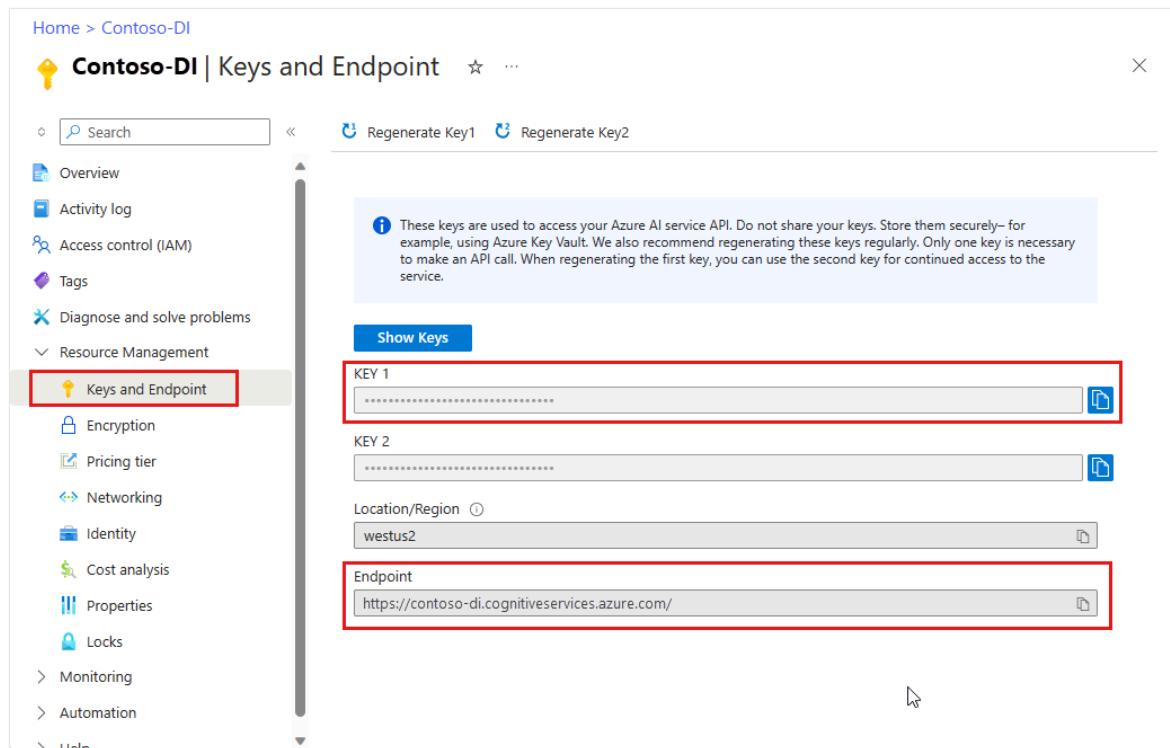
- ✓ The [prebuilt-read](#) model is at the core of all Document Intelligence models and can detect lines, words, locations, and languages. Layout, general document, prebuilt, and custom models all use the `read` model as a foundation for extracting texts from documents.
- ✓ The [prebuilt-layout](#) model extracts text and text locations, tables, selection marks, and structure information from documents and images. You can extract key/value pairs using the layout model with the optional query string parameter `features=keyValuePairs` enabled.
- ✓ The [prebuilt-contract](#) model extracts key information from contractual agreements.
- ✓ The [prebuilt-healthInsuranceCard.us](#) model extracts key information from US health insurance cards.
- ✓ The [prebuilt tax document models](#) model extracts information reported on US tax forms.
- ✓ The [prebuilt-invoice](#) model extracts key fields and line items from sales invoices in various formats and quality. Fields include phone-captured images, scanned documents, and digital PDFs.
- ✓ The [prebuilt-receipt](#) model extracts key information from printed and handwritten sales receipts.
- ✓ The [prebuilt-idDocument](#) model extracts key information from US drivers licenses, international passport biographical pages, US state IDs, social security cards, and

permanent resident cards.

[Client library](#) | [SDK reference ↗](#) | [REST API reference](#) | [Package ↗](#) | [Samples ↗](#) | [Supported REST API version](#)

## Prerequisites

- An Azure subscription - [Create one for free ↗](#).
- The [Visual Studio IDE ↗](#).
- An Azure AI services or Document Intelligence resource. Create a [single-service ↗](#) or [multi-service ↗](#). You can use the free pricing tier ([F0](#)) to try the service, and upgrade later to a paid tier for production.
- The key and endpoint from the resource you create to connect your application to the Azure Document Intelligence service.
  1. After your resource deploys, select **Go to resource**.
  2. In the left navigation menu, select **Keys and Endpoint**.
  3. Copy one of the keys and the **Endpoint** for use later in this article.



- A document file at a URL location. For this project, you can use the sample forms provided in the following table for each feature:

[+] [Expand table](#)

Feature	modelID	document-url
Read model	prebuilt-read	<a href="#">Sample brochure ↗</a>
Layout model	prebuilt-layout	<a href="#">Sample booking confirmation ↗</a>
W-2 form model	prebuilt-tax.us.w2	<a href="#">Sample W-2 form ↗</a>
Invoice model	prebuilt-invoice	<a href="#">Sample invoice ↗</a>
Receipt model	prebuilt-receipt	<a href="#">Sample receipt ↗</a>
ID document model	prebuilt-idDocument	<a href="#">Sample ID document ↗</a>

## Set your environment variables

To interact with the Document Intelligence service, you need to create an instance of the `DocumentAnalysisClient` class. To do so, instantiate the client with your `key` and `endpoint` from the Azure portal. For this project, use environment variables to store and access credentials.

### ⓘ Important

Use API keys with caution. Don't include the API key directly in your code, and never post it publicly. If you use an API key, store it securely in Azure Key Vault. For more information about using API keys securely in your apps, see [API keys with Azure Key Vault](#).

For more information about AI services security, see [Authenticate requests to Azure AI services](#).

To set the environment variable for your Document Intelligence resource key, open a console window, and follow the instructions for your operating system and development environment. Replace `<yourKey>` and `<yourEndpoint>` with the values from your resource in the Azure portal.

#### Windows

Environment variables in Windows aren't case-sensitive. They're typically declared in uppercase, with words joined by an underscore. At a command prompt, run the following commands:

1. Set your key variable:

Console

```
setx DI_KEY <yourKey>
```

## 2. Set your endpoint variable

Console

```
setx DI_ENDPOINT <yourEndpoint>
```

## 3. Close the Command Prompt window after you set your environment variables.

The values remain until you change them again.

## 4. Restart any running programs that read the environment variable. For example, if you're using Visual Studio or Visual Studio Code as your editor, restart before running the sample code.

Here are a few more helpful commands to use with environment variables:

 Expand table

Command	Action	Example
<code>setx VARIABLE_NAME=</code>	Delete the environment variable by setting the value to an empty string.	<code>setx DI_KEY=</code>
<code>setx VARIABLE_NAME=value</code>	Set or change the value of an environment variable.	<code>setx DI_KEY=&lt;yourKey&gt;</code>
<code>set VARIABLE_NAME</code>	Display the value of a specific environment variable.	<code>set DI_KEY</code>
<code>set</code>	Display all environment variables.	<code>set</code>

# Set up your programming environment

1. Start Visual Studio.
2. On the start page, choose **Create a new project**.

# Visual Studio 2022

## Open recent

As you use Visual Studio, any projects, folders, or files that you open will show up here for quick access.

You can pin anything that you open frequently so that it's always at the top of the list.

## Get started



### Clone a repository

Get code from an online repository like GitHub or Azure DevOps



### Open a project or solution

Open a local Visual Studio project or .sln file



### Open a local folder

Navigate and edit code within any folder

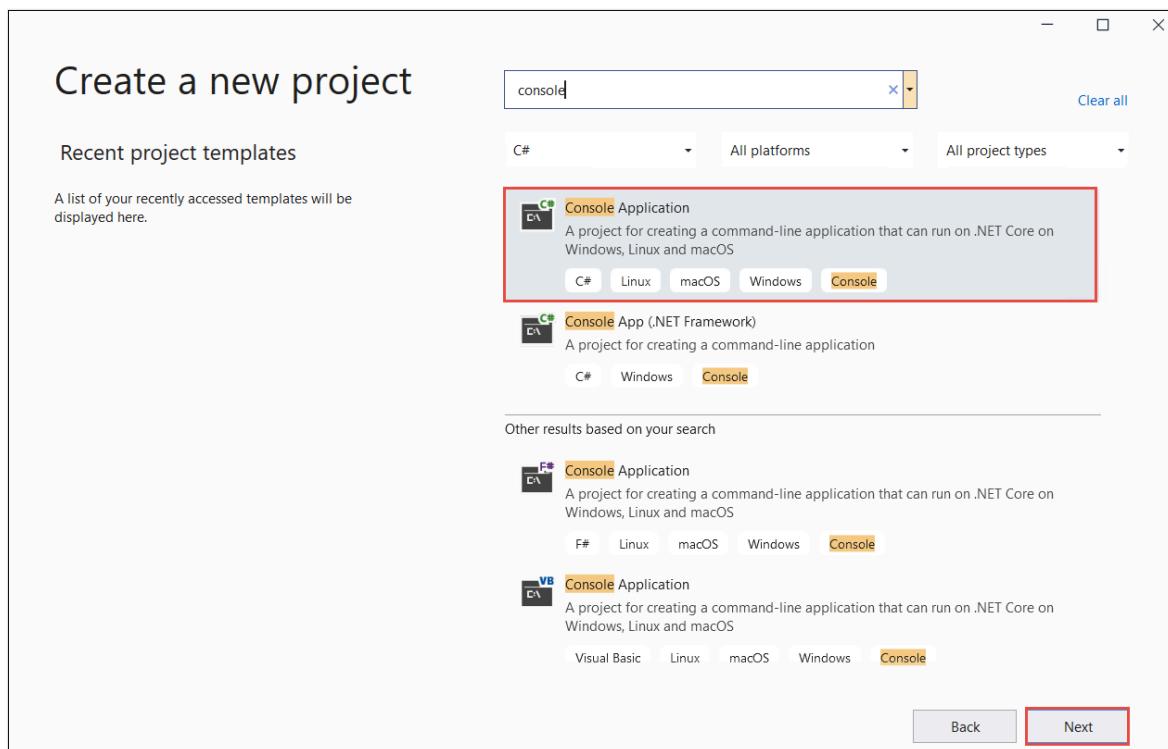


### Create a new project

Choose a project template with code scaffolding to get started

[Continue without code →](#)

3. On the **Create a new project** page, enter *console* in the search box. Select the **Console Application** template, then choose **Next**.



4. In the **Configure your new project** page, under **Project name** enter **docIntelligence\_app**. Then select **Next**.

## Configure your new project

Console App C# Linux macOS Windows Console

Project name

Location  C:\Users\...

Solution name

Place solution and project in the same directory

5. In the Additional information page, select .NET 8.0 (Long-term support), and then select **Create**.

## Additional information

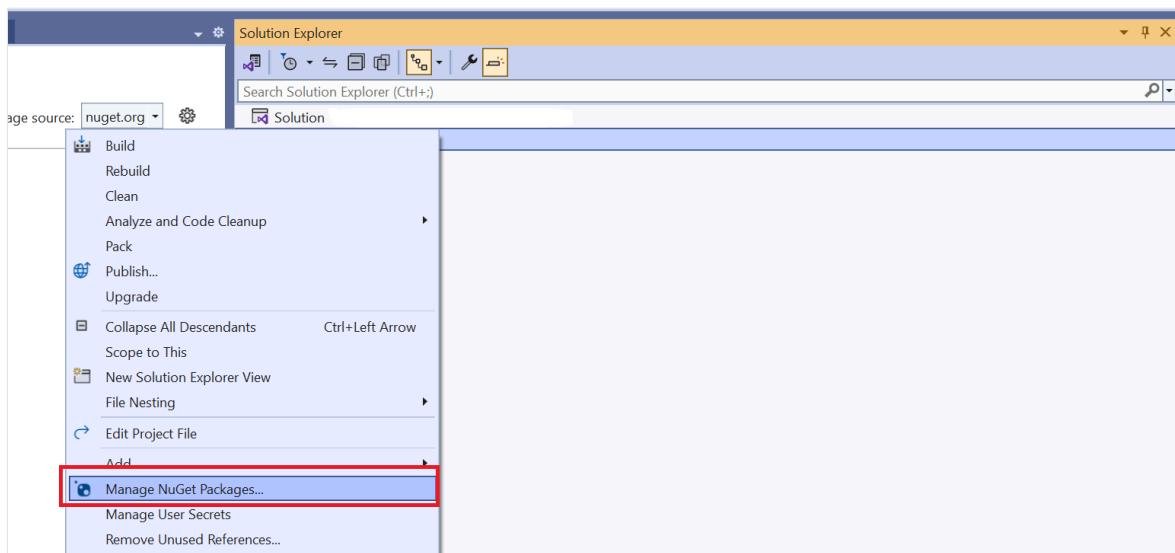
Console App C# Linux macOS Windows Console

Framework

Do not use top-level statements   
 Enable native AOT publish

## Install the client library with NuGet

1. Right-click on your **docIntelligence\_app** project and select **Manage NuGet Packages...** .



2. Select the **Browse** tab and type *Azure.AI.DocumentIntelligence*.
3. Choose a version from the dropdown menu and install the package in your project.

## Build your application

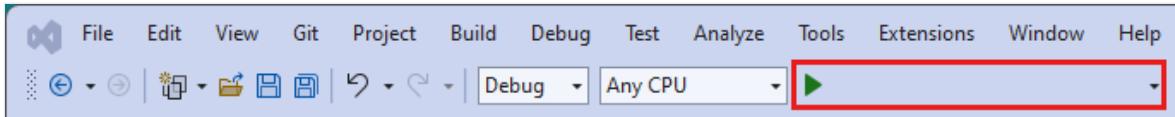
### Note

Starting with .NET 6, new projects using the `console` template generate a new program style that differs from previous versions. The new output uses recent C# features that simplify the code you need to write.

When you use the newer version, you only need to write the body of the `Main` method. You don't need to include top-level statements, global using directives, or implicit using directives. For more information, see [C# console app template generates top-level statements](#).

1. Open the `Program.cs` file.
2. Delete the existing code, including the line `Console.WriteLine("Hello World!")`.
3. Select one of the following code samples and copy/paste into your application's `Program.cs` file:
  - [prebuilt-read](#)
  - [prebuilt-layout](#)
  - [prebuilt-tax.us.w2](#)
  - [prebuilt-invoice](#)
  - [prebuilt-receipt](#)
  - [prebuilt-idDocument](#)

4. After you add a code sample to your application, choose the green **Start** button next to the project name to build and run your program, or press **F5**.



## Use the Read model

C#

```
using Azure;
using Azure.AI.DocumentIntelligence;

//use your `key` and `endpoint` environment variables to create your
//`AzureKeyCredential` and `DocumentIntelligenceClient` instances
string key = Environment.GetEnvironmentVariable("DI_KEY");
string endpoint = Environment.GetEnvironmentVariable("DI_ENDPOINT");
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentIntelligenceClient client = new DocumentIntelligenceClient(new
Uri(endpoint), credential);

//sample document
Uri fileUri = new Uri("https://raw.githubusercontent.com/Azure-
Samples/cognitive-services-REST-api-samples/master/curl/form-
recognizer/rest-api/read.png");

Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-read", fileUri);
AnalyzeResult result = operation.Value;

foreach (DocumentPage page in result.Pages)
{
    Console.WriteLine($"Document Page {page.PageNumber} has
{page.Lines.Count} line(s), {page.Words.Count} word(s),");
    Console.WriteLine($"and {page.SelectionMarks.Count} selection
mark(s).");

    for (int i = 0; i < page.Lines.Count; i++)
    {
        DocumentLine line = page.Lines[i];
        Console.WriteLine($" Line {i} has content: '{line.Content}'.");

        Console.WriteLine($" Its bounding polygon (points ordered
clockwise):");

        for (int j = 0; j < line.Polygon.Count; j++)
        {
            Console.WriteLine($" Point {j} => X: {line.Polygon[j].X},
Y: {line.Polygon[j].Y}");
        }
    }
}
```

```

}

foreach (DocumentStyle style in result.Styles)
{
    // Check the style and style confidence to see if text is handwritten.
    // Note that value '0.8' is used as an example.

    bool isHandwritten = style.IsHandwritten.HasValue && style.IsHandwritten
== true;

    if (isHandwritten && style.Confidence > 0.8)
    {
        Console.WriteLine($"Handwritten content found:");

        foreach (DocumentSpan span in style.Spans)
        {
            Console.WriteLine($"  Content:
{result.Content.Substring(span.Index, span.Length)}");
        }
    }
}

Console.WriteLine("Detected languages:");

foreach (DocumentLanguage language in result.Languages)
{
    Console.WriteLine($"  Found language with locale'{language.Locale}' with
confidence {language.Confidence}.");
}

```

Visit the Azure samples repository on GitHub and view the [read model output](#).

## Use the Layout model

C#

```

using Azure;
using Azure.AI.DocumentIntelligence;

//use your `key` and `endpoint` environment variables to create your
`AzureKeyCredential` and `DocumentIntelligenceClient` instances
string key = Environment.GetEnvironmentVariable("DI_KEY");
string endpoint = Environment.GetEnvironmentVariable("DI_ENDPOINT");
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentIntelligenceClient client = new DocumentIntelligenceClient(new
Uri(endpoint), credential);

// sample document document
Uri fileUri = new Uri ("https://raw.githubusercontent.com/Azure-
Samples/cognitive-services-REST-api-samples/master/curl/form-
recognizer/rest-api/layout.png");

```

```

Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-layout",
fileUri);

AnalyzeResult result = operation.Value;

foreach (DocumentPage page in result.Pages)
{
    Console.WriteLine($"Document Page {page.PageNumber} has
{page.Lines.Count} line(s), {page.Words.Count} word(s),");
    Console.WriteLine($"and {page.SelectionMarks.Count} selection
mark(s).");

    for (int i = 0; i < page.Lines.Count; i++)
    {
        DocumentLine line = page.Lines[i];
        Console.WriteLine($"  Line {i} has content: '{line.Content}'.");

        Console.WriteLine($"    Its bounding polygon (points ordered
clockwise):");

        for (int j = 0; j < line.Polygon.Count; j++)
        {
            Console.WriteLine($"      Point {j} => X: {line.Polygon[j].X},
Y: {line.Polygon[j].Y}");
        }
    }

    for (int i = 0; i < page.SelectionMarks.Count; i++)
    {
        DocumentSelectionMark selectionMark = page.SelectionMarks[i];

        Console.WriteLine($"  Selection Mark {i} is
{selectionMark.State}.");
        Console.WriteLine($"  Its bounding polygon (points ordered
clockwise):");

        for (int j = 0; j < selectionMark.Polygon.Count; j++)
        {
            Console.WriteLine($"    Point {j} => X:
{selectionMark.Polygon[j].X}, Y: {selectionMark.Polygon[j].Y}");
        }
    }
}

Console.WriteLine("Paragraphs:");

foreach (DocumentParagraph paragraph in result.Paragraphs)
{
    Console.WriteLine($"  Paragraph content: {paragraph.Content}");

    if (paragraph.Role != null)
    {
        Console.WriteLine($"    Role: {paragraph.Role}");
    }
}

```

```

        }

    }

    foreach (DocumentStyle style in result.Styles)
    {
        // Check the style and style confidence to see if text is handwritten.
        // Note that value '0.8' is used as an example.

        bool isHandwritten = style.IsHandwritten.HasValue && style.IsHandwritten
== true;

        if (isHandwritten && style.Confidence > 0.8)
        {
            Console.WriteLine($"Handwritten content found:");

            foreach (DocumentSpan span in style.Spans)
            {
                Console.WriteLine($" Content:
{result.Content.Substring(span.Index, span.Length)}");
            }
        }
    }

    Console.WriteLine("The following tables were extracted:");

    for (int i = 0; i < result.Tables.Count; i++)
    {
        DocumentTable table = result.Tables[i];
        Console.WriteLine($" Table {i} has {table.RowCount} rows and
{table.ColumnCount} columns.");

        foreach (DocumentTableCell cell in table.Cells)
        {
            Console.WriteLine($" Cell ({cell.RowIndex}, {cell.ColumnIndex})
has kind '{cell.Kind}' and content: '{cell.Content}'.");
        }
    }
}

```

Visit the Azure samples repository on GitHub and view the [layout model output](#).

## Use the General document model

C#

```

using Azure;
using Azure.AI.DocumentIntelligence;

//use your `key` and `endpoint` environment variables to create your
//`AzureKeyCredential` and `DocumentIntelligenceClient` instances
string key = Environment.GetEnvironmentVariable("DI_KEY");
string endpoint = Environment.GetEnvironmentVariable("DI_ENDPOINT");

```

```
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentIntelligenceClient client = new DocumentIntelligenceClient(new Uri(endpoint), credential);

// sample document document
Uri fileUri = new Uri("https://raw.githubusercontent.com/Azure-
Samples/cognitive-services-REST-api-samples/master/curl/form-
recognizer/sample-layout.pdf");

Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-document",
fileUri);

AnalyzeResult result = operation.Value;

Console.WriteLine("Detected key-value pairs:");

foreach (DocumentKeyValuePair kvp in result.KeyValuePairs)
{
    if (kvp.Value == null)
    {
        Console.WriteLine($"  Found key with no value:
'{kvp.Key.Content}'");
    }
    else
    {
        Console.WriteLine($"  Found key-value pair: '{kvp.Key.Content}' and
'{kvp.Value.Content}'");
    }
}

foreach (DocumentPage page in result.Pages)
{
    Console.WriteLine($"Document Page {page.PageNumber} has
{page.Lines.Count} line(s), {page.Words.Count} word(s),");
    Console.WriteLine($"and {page.SelectionMarks.Count} selection
mark(s).");

    for (int i = 0; i < page.Lines.Count; i++)
    {
        DocumentLine line = page.Lines[i];
        Console.WriteLine($"    Line {i} has content: '{line.Content}'.",

        Console.WriteLine($"      Its bounding polygon (points ordered
clockwise):");

        for (int j = 0; j < line.Polygon.Count; j++)
        {
            Console.WriteLine($"          Point {j} => X: {line.Polygon[j].X},
Y: {line.Polygon[j].Y}");
        }
    }

    for (int i = 0; i < page.SelectionMarks.Count; i++)
    {
```

```

        DocumentSelectionMark selectionMark = page.SelectionMarks[i];

        Console.WriteLine($"  Selection Mark {i} is
{selectionMark.State}.");
        Console.WriteLine($"    Its bounding polygon (points ordered
clockwise):");

        for (int j = 0; j < selectionMark.Polygon.Count; j++)
        {
            Console.WriteLine($"      Point {j} => X:
{selectionMark.Polygon[j].X}, Y: {selectionMark.Polygon[j].Y}");
        }
    }

foreach (DocumentStyle style in result.Styles)
{
    // Check the style and style confidence to see if text is handwritten.
    // Note that value '0.8' is used as an example.

    bool isHandwritten = style.IsHandwritten.HasValue && style.IsHandwritten
== true;

    if (isHandwritten && style.Confidence > 0.8)
    {
        Console.WriteLine($"Handwritten content found:");

        foreach (DocumentSpan span in style.Spans)
        {
            Console.WriteLine($"  Content:
{result.Content.Substring(span.Index, span.Length)}");
        }
    }
}

Console.WriteLine("The following tables were extracted:");

for (int i = 0; i < result.Tables.Count; i++)
{
    DocumentTable table = result.Tables[i];
    Console.WriteLine($"  Table {i} has {table.RowCount} rows and
{table.ColumnCount} columns.");

    foreach (DocumentTableCell cell in table.Cells)
    {
        Console.WriteLine($"    Cell ({cell.RowIndex}, {cell.ColumnIndex})
has kind '{cell.Kind}' and content: '{cell.Content}'.");
    }
}

```

Visit the Azure samples repository on GitHub and view the [general document model output](#).

# Use the W-2 tax model

C#

```
using Azure;
using Azure.AI.DocumentIntelligence;

//use your `key` and `endpoint` environment variables to create your
`AzureKeyCredential` and `DocumentIntelligenceClient` instances
string key = Environment.GetEnvironmentVariable("DI_KEY");
string endpoint = Environment.GetEnvironmentVariable("DI_ENDPOINT");
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentIntelligenceClient client = new DocumentIntelligenceClient(new
Uri(endpoint), credential);

// sample document document
Uri w2Uri = new Uri("https://raw.githubusercontent.com/Azure-
Samples/cognitive-services-REST-api-samples/master/curl/form-
recognizer/rest-api/w2.png");

Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-tax.us.w2",
w2Uri);

AnalyzeResult result = operation.Value;

for (int i = 0; i < result.Documents.Count; i++)
{
    Console.WriteLine($"Document {i}:");

    AnalyzedDocument document = result.Documents[i];

    if (document.Fields.TryGetValue("AdditionalInfo", out DocumentField?
additionalInfoField))
    {
        if (additionalInfoField.FieldType == DocumentFieldType.List)
        {
            foreach (DocumentField infoField in
additionalInfoField.Value.AsList())
            {
                Console.WriteLine("AdditionalInfo:");

                if (infoField.FieldType == DocumentFieldType.Dictionary)
                {
                    IReadOnlyDictionary<string, DocumentField> infoFields =
infoField.Value.AsDictionary();

                    if (infoFields.TryGetValue("Amount", out DocumentField?
amountField))
                    {
                        if (amountField.FieldType ==
DocumentFieldType.Double)
```

```

        {
            double amount = amountField.Value.AsDouble();

            Console.WriteLine($" Amount: '{amount}', with
confidence {amountField.Confidence}");
        }
    }

    if (infoFields.TryGetValue("LetterCode", out
DocumentField? letterCodeField))
    {
        if (letterCodeField.FieldType ==
DocumentFieldType.String)
        {
            string letterCode =
letterCodeField.ValueAsString();

            Console.WriteLine($" LetterCode:
'{letterCode}', with confidence {letterCodeField.Confidence}");
        }
    }
}

if (document.Fields.TryGetValue("AllocatedTips", out DocumentField?
allocatedTipsField))
{
    if (allocatedTipsField.FieldType == DocumentFieldType.Double)
    {
        double allocatedTips = allocatedTipsField.Value.AsDouble();
        Console.WriteLine($"Allocated Tips: '{allocatedTips}', with
confidence {allocatedTipsField.Confidence}");
    }
}

if (document.Fields.TryGetValue("Employer", out DocumentField?
employerField))
{
    if (employerField.FieldType == DocumentFieldType.Dictionary)
    {
        IReadOnlyDictionary<string, DocumentField> employerFields =
employerField.Value.AsDictionary();

        if (employerFields.TryGetValue("Name", out DocumentField?
employerNameField))
        {
            if (employerNameField.FieldType == DocumentFieldType.String)
            {
                string name = employerNameField.ValueAsString();

                Console.WriteLine($"Employer Name: '{name}', with
confidence {employerNameField.Confidence}");
            }
        }
    }
}

```

```

        }

        if (employerFields.TryGetValue("IdNumber", out DocumentField?
idNumberField))
        {
            if (idNumberField.FieldType == DocumentFieldType.String)
            {
                string id = idNumberField.ValueAsString();

                Console.WriteLine($"Employer ID Number: '{id}', with
confidence {idNumberField.Confidence}");
            }
        }

        if (employerFields.TryGetValue("Address", out DocumentField?
addressField))
        {
            if (addressField.FieldType == DocumentFieldType.Address)
            {
                Console.WriteLine($"Employer Address:
'{addressField.Content}', with confidence {addressField.Confidence}");
            }
        }
    }
}

```

Visit the Azure samples repository on GitHub and view the [W-2 tax model output](#).

## Use the Invoice model

C#

```

using Azure;
using Azure.AI.DocumentIntelligence;

//use your `key` and `endpoint` environment variables to create your
`AzureKeyCredential` and `DocumentIntelligenceClient` instances
string key = Environment.GetEnvironmentVariable("DI_KEY");
string endpoint = Environment.GetEnvironmentVariable("DI_ENDPOINT");
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentIntelligenceClient client = new DocumentIntelligenceClient(new
Uri(endpoint), credential);

// sample document document
Uri invoiceUri = new Uri("https://github.com/Azure-Samples/cognitive-
services-REST-api-samples/raw/master/curl/form-recognizer/rest-
api/invoice.pdf");

Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-invoice",

```

```

invoiceUri);

AnalyzeResult result = operation.Value;

for (int i = 0; i < result.Documents.Count; i++)
{
    Console.WriteLine($"Document {i}:");

    AnalyzedDocument document = result.Documents[i];

    if (document.Fields.TryGetValue("VendorName", out DocumentField
vendorNameField))
    {
        if (vendorNameField.FieldType == DocumentFieldType.String)
        {
            string vendorName = vendorNameField.ValueAsString();
            Console.WriteLine($"Vendor Name: '{vendorName}', with confidence
{vendorNameField.Confidence}");
        }
    }

    if (document.Fields.TryGetValue("CustomerName", out DocumentField
customerNameField))
    {
        if (customerNameField.FieldType == DocumentFieldType.String)
        {
            string customerName = customerNameField.ValueAsString();
            Console.WriteLine($"Customer Name: '{customerName}', with
confidence {customerNameField.Confidence}");
        }
    }

    if (document.Fields.TryGetValue("Items", out DocumentField itemsField))
    {
        if (itemsField.FieldType == DocumentFieldType.List)
        {
            foreach (DocumentField itemField in itemsField.Value.AsList())
            {
                Console.WriteLine("Item:");

                if (itemField.FieldType == DocumentFieldType.Dictionary)
                {
                    IReadOnlyDictionary<string, DocumentField> itemFields =
itemField.Value.AsDictionary();

                    if (itemFields.TryGetValue("Description", out
DocumentField itemDescriptionField))
                    {
                        if (itemDescriptionField.FieldType ==
DocumentFieldType.String)
                        {
                            string itemDescription =
itemDescriptionField.ValueAsString();

                            Console.WriteLine($" Description:
");
                        }
                    }
                }
            }
        }
    }
}

```

```

'{itemDescription}', with confidence {itemDescriptionField.Confidence});
        }
    }

        if (itemFields.TryGetValue("Amount", out DocumentField
itemAmountField))
    {
        if (itemAmountField.FieldType ==
DocumentFieldType.Currency)
    {
        CurrencyValue itemAmount =
itemAmountField.Value.AsCurrency();

        Console.WriteLine($" Amount:
'{itemAmount.Symbol}{itemAmount.Amount}', with confidence
{itemAmountField.Confidence}");
    }
}
}

if (document.Fields.TryGetValue("SubTotal", out DocumentField
subTotalField))
{
    if (subTotalField.FieldType == DocumentFieldType.Currency)
    {
        CurrencyValue subTotal = subTotalField.Value.AsCurrency();
        Console.WriteLine($"Sub Total: '{subTotal.Symbol}
{subTotal.Amount}', with confidence {subTotalField.Confidence}");
    }
}

if (document.Fields.TryGetValue("TotalTax", out DocumentField
totalTaxField))
{
    if (totalTaxField.FieldType == DocumentFieldType.Currency)
    {
        CurrencyValue totalTax = totalTaxField.Value.AsCurrency();
        Console.WriteLine($"Total Tax: '{totalTax.Symbol}
{totalTax.Amount}', with confidence {totalTaxField.Confidence}");
    }
}

if (document.Fields.TryGetValue("InvoiceTotal", out DocumentField
invoiceTotalField))
{
    if (invoiceTotalField.FieldType == DocumentFieldType.Currency)
    {
        CurrencyValue invoiceTotal =
invoiceTotalField.Value.AsCurrency();
        Console.WriteLine($"Invoice Total: '{invoiceTotal.Symbol}
{invoiceTotal.Amount}', with confidence {invoiceTotalField.Confidence}");
    }
}

```

```
    }  
}
```

Visit the Azure samples repository on GitHub and view the [invoice model output](#).

## Use the Receipt model

C#

```
using Azure;  
using Azure.AI.DocumentIntelligence;  
  
//use your `key` and `endpoint` environment variables to create your  
`AzureKeyCredential` and `DocumentIntelligenceClient` instances  
string key = Environment.GetEnvironmentVariable("DI_KEY");  
string endpoint = Environment.GetEnvironmentVariable("DI_ENDPOINT");  
AzureKeyCredential credential = new AzureKeyCredential(key);  
DocumentIntelligenceClient client = new DocumentIntelligenceClient(new  
Uri(endpoint), credential);  
  
// sample document document  
Uri receiptUri = new Uri("https://raw.githubusercontent.com/Azure-  
Samples/cognitive-services-REST-api-samples/master/curl/form-  
recognizer/rest-api/receipt.png");  
  
Operation<AnalyzeResult> operation = await  
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-receipt",  
receiptUri);  
  
AnalyzeResult receipts = operation.Value;  
  
foreach (AnalyzedDocument receipt in receipts.Documents)  
{  
    if (receipt.Fields.TryGetValue("MerchantName", out DocumentField  
merchantNameField))  
    {  
        if (merchantNameField.FieldType == DocumentFieldType.String)  
        {  
            string merchantName = merchantNameField.ValueAsString();  
  
            Console.WriteLine($"Merchant Name: '{merchantName}', with  
confidence {merchantNameField.Confidence}");  
        }  
    }  
  
    if (receipt.Fields.TryGetValue("TransactionDate", out DocumentField  
transactionDateField))  
    {  
        if (transactionDateField.FieldType == DocumentFieldType.Date)  
        {  
            DateTimeOffset transactionDate =
```

```

transactionDateField.Value.AsDate();

        Console.WriteLine($"Transaction Date: '{transactionDate}', with
confidence {transactionDateField.Confidence}");
    }
}

if (receipt.Fields.TryGetValue("Items", out DocumentField itemsField))
{
    if (itemsField.FieldType == DocumentFieldType.List)
    {
        foreach (DocumentField itemField in itemsField.Value.AsList())
        {
            Console.WriteLine("Item:");

            if (itemField.FieldType == DocumentFieldType.Dictionary)
            {
                IReadOnlyDictionary<string, DocumentField> itemFields =
itemField.Value.AsDictionary();

                    if (itemFields.TryGetValue("Description", out
DocumentField itemDescriptionField))
                    {
                        if (itemDescriptionField.FieldType ==
DocumentFieldType.String)
                        {
                            string itemDescription =
itemDescriptionField.ValueAsString();

                                Console.WriteLine($" Description:
'{itemDescription}', with confidence {itemDescriptionField.Confidence}");
                        }
                    }

                    if (itemFields.TryGetValue("TotalPrice", out
DocumentField itemTotalPriceField))
                    {
                        if (itemTotalPriceField.FieldType ==
DocumentFieldType.Double)
                        {
                            double itemTotalPrice =
itemTotalPriceField.Value.ToDouble();

                                Console.WriteLine($" Total Price:
'{itemTotalPrice}', with confidence {itemTotalPriceField.Confidence}");
                        }
                    }
                }
            }
        }
    }
}

if (receipt.Fields.TryGetValue("Total", out DocumentField totalField))
{
    if (totalField.FieldType == DocumentFieldType.Double)

```

```

        {
            double total = totalField.Value.AsDouble();

            Console.WriteLine($"Total: '{total}', with confidence
'{totalField.Confidence}');
        }
    }
}

```

Visit the Azure samples repository on GitHub and view the [receipt model output](#).

## Use the ID document model

C#

```

using Azure;
using Azure.AI.DocumentIntelligence;

//use your `key` and `endpoint` environment variables to create your
`AzureKeyCredential` and `DocumentIntelligenceClient` instances
string key = Environment.GetEnvironmentVariable("DI_KEY");
string endpoint = Environment.GetEnvironmentVariable("DI_ENDPOINT");
AzureKeyCredential credential = new AzureKeyCredential(key);
DocumentIntelligenceClient client = new DocumentIntelligenceClient(new
Uri(endpoint), credential);

// sample document document

Uri idDocumentUri = new Uri("https://raw.githubusercontent.com/Azure-
Samples/cognitive-services-REST-api-samples/master/curl/form-
recognizer/rest-api/identity_documents.png");

Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-idDocument",
idDocumentUri);

AnalyzeResult identityDocuments = operation.Value;

AnalyzedDocument identityDocument = identityDocuments.Documents.Single();

if (identityDocument.Fields.TryGetValue("Address", out DocumentField
addressField))
{
    if (addressField.FieldType == DocumentFieldType.String)
    {
        string address = addressField.Value.ToString();
        Console.WriteLine($"Address: '{address}', with confidence
{addressField.Confidence}");
    }
}

```

```
if (identityDocument.Fields.TryGetValue("CountryRegion", out DocumentField countryRegionField))
{
    if (countryRegionField.FieldType == DocumentFieldType.CountryRegion)
    {
        string countryRegion = countryRegionField.Value.AsCountryRegion();
        Console.WriteLine($"CountryRegion: '{countryRegion}', with confidence {countryRegionField.Confidence}");
    }
}

if (identityDocument.Fields.TryGetValue("DateOfBirth", out DocumentField dateOfBirthField))
{
    if (dateOfBirthField.FieldType == DocumentFieldType.Date)
    {
        DateTimeOffset dateOfBirth = dateOfBirthField.Value.AsDate();
        Console.WriteLine($"Date Of Birth: '{dateOfBirth}', with confidence {dateOfBirthField.Confidence}");
    }
}

if (identityDocument.Fields.TryGetValue("DateOfExpiration", out DocumentField dateOfExpirationField))
{
    if (dateOfExpirationField.FieldType == DocumentFieldType.Date)
    {
        DateTimeOffset dateOfExpiration =
dateOfExpirationField.Value.AsDate();
        Console.WriteLine($"Date Of Expiration: '{dateOfExpiration}', with confidence {dateOfExpirationField.Confidence}");
    }
}

if (identityDocument.Fields.TryGetValue("DocumentNumber", out DocumentField documentNumberField))
{
    if (documentNumberField.FieldType == DocumentFieldType.String)
    {
        string documentNumber = documentNumberField.ValueAsString();
        Console.WriteLine($"Document Number: '{documentNumber}', with confidence {documentNumberField.Confidence}");
    }
}

if (identityDocument.Fields.TryGetValue("FirstName", out DocumentField firstNameField))
{
    if (firstNameField.FieldType == DocumentFieldType.String)
    {
        string firstName = firstNameField.ValueAsString();
        Console.WriteLine($"First Name: '{firstName}', with confidence {firstNameField.Confidence}");
    }
}
```

```
}

if (identityDocument.Fields.TryGetValue("LastName", out DocumentField lastNameField))
{
    if (lastNameField.FieldType == DocumentFieldType.String)
    {
        string lastName = lastNameField.ValueAsString();
        Console.WriteLine($"Last Name: '{lastName}', with confidence {lastNameField.Confidence}");
    }
}

if (identityDocument.Fields.TryGetValue("Region", out DocumentField regionfield))
{
    if (regionfield.FieldType == DocumentFieldType.String)
    {
        string region = regionfield.ValueAsString();
        Console.WriteLine($"Region: '{region}', with confidence {regionfield.Confidence}");
    }
}

if (identityDocument.Fields.TryGetValue("Sex", out DocumentField sexfield))
{
    if (sexfield.FieldType == DocumentFieldType.String)
    {
        string sex = sexfield.ValueAsString();
        Console.WriteLine($"Sex: '{sex}', with confidence {sexfield.Confidence}");
    }
}
```

Visit the Azure samples repository on GitHub and view the [ID document model output](#).

## Next steps

Congratulations! You learned to use Document Intelligence models to analyze various documents in different ways. Next, explore the Document Intelligence Studio and reference documentation.

[Try the Document Intelligence Studio](#)

[Explore the Document Intelligence REST API](#)

# Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Check usage and estimate cost

Article • 04/07/2025

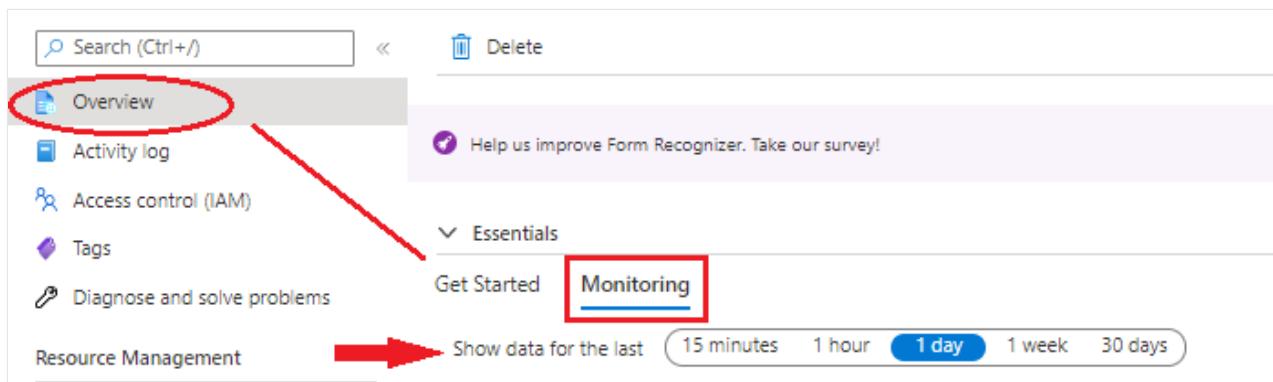
This content applies to: ✓ v4.0 (GA) ✓ v3.1 (GA) ✓ v3.0 (GA) ✓ v2.1 (GA)

In this guide, learn how to use the metrics dashboard in the Azure portal to view how many pages are processed. You also learn how to estimate the cost of processing those pages using the Azure pricing calculator.

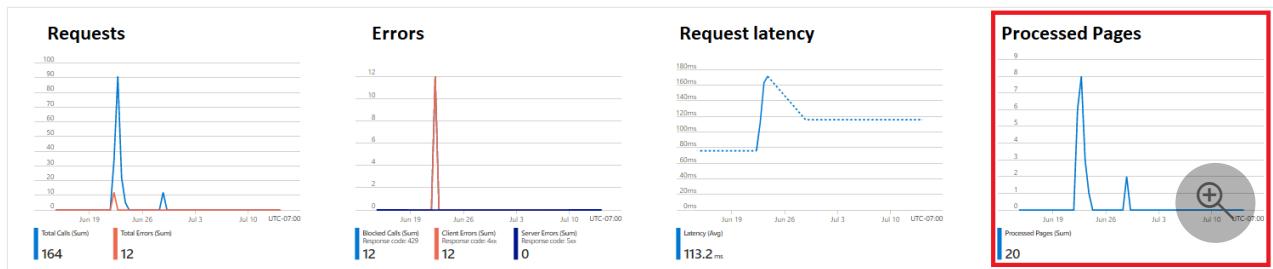
## Check how many pages were processed

We start by looking at the page processing data for a given time period:

1. Sign in to the [Azure portal](#).
2. Navigate to your Document Intelligence resource.
3. From the **Overview** page, select the **Monitoring** tab located near the middle of the page.



4. Select a time range and you see the **Processed Pages** chart displayed.



## Examine analyzed pages

We can now take a deeper dive to see each model's analyzed pages:

1. Under the **Monitoring** section, select **Metrics** from the left pane.

## Monitoring

Alerts

**Metrics**

Diagnostic settings

Logs

2. On the **Metrics** page, select **Add metric**.

3. Select the Metric dropdown menu and, under **USAGE**, choose **Processed Pages**.

The screenshot shows the Azure Metrics blade. At the top left is the 'Add metric' button, which is highlighted with a red box. Below it are sections for 'Scope' (set to 'form-recognizer') and 'Metric Namespace' (set to 'Cognitive Service stand...'). To the right is a large dropdown menu for 'Metric'. The 'Select metric' dropdown is open, showing several options like 'Data Out', 'Latency', 'Server Errors', etc., with 'Processed Pages' highlighted and also enclosed in a red box. The 'Aggregation' dropdown is also visible. A tooltip 'or learn more below:' points to a section with a chart icon.

4. From the upper right corner, configure the time range and select the **Apply** button.

The screenshot shows the Metrics blade with a chart for 'Sum Processed Pages for form-recognizer'. In the top right, a time range dialog is open, titled 'Local Time: Last 4 hours (Automatic - 1 minute)'. It includes a 'Time range' section with radio buttons for various time intervals (Last 30 minutes, Last 4 hours, etc.) and a 'Time granularity' section set to 'Automatic'. At the bottom are 'Apply' and 'Cancel' buttons, with 'Apply' highlighted with a red box.

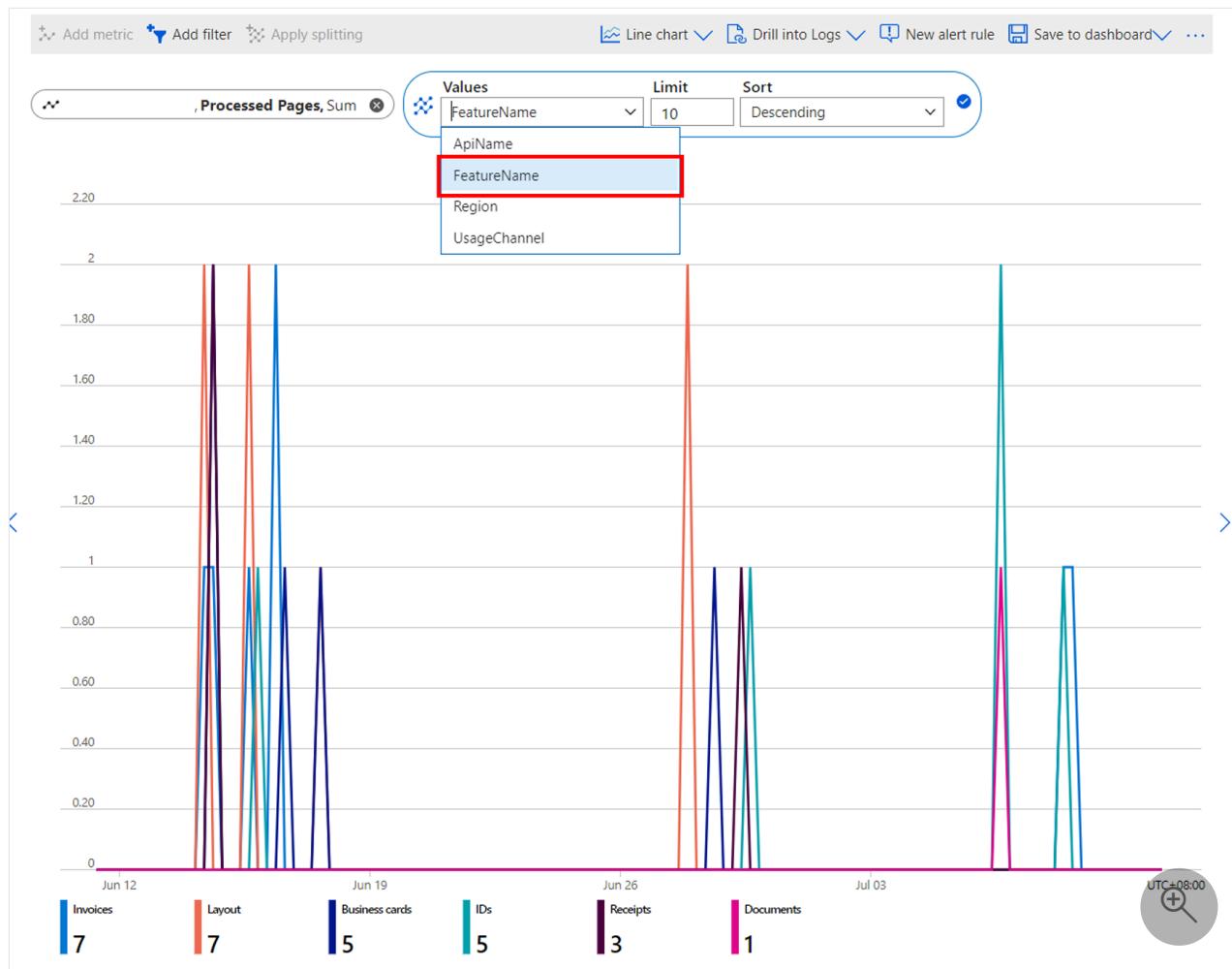
5. Select **Apply splitting**.

The screenshot shows the Metrics blade with a chart for 'Sum Processed Pages for form-recognizer'. At the top right is the 'Apply splitting' button, which is highlighted with a red box. Below the chart is a status bar showing 'form-recognizer, Processed Pages, Sum'.

6. Choose **FeatureName** from the **Values** dropdown menu.

The screenshot shows the Azure Metrics Explorer interface. At the top, there are buttons for 'Add metric', 'Add filter', and 'Apply splitting'. Below that, a search bar contains the text 'form-recognizer, Processed Pages, Sum'. To the right of the search bar is a dropdown menu labeled 'Values' with a blue icon. The 'Values' dropdown is open, showing a list of options: 'Select value(s)', 'ApiName', 'FeatureName' (which is highlighted with a red box), 'Region', and 'UsageChannel'. To the right of the 'Values' dropdown is a 'Limit' field set to '10'. Below the search bar, there is a chart area with a y-axis ranging from 80 to 100 and an x-axis showing dates from June 12 to July 03. The chart displays several vertical bars representing different feature names. At the bottom of the chart area, there are labels for categories: Invoices (7), Layout (7), Business cards (5), IDs (5), Receipts (3), and Documents (1). On the far right, there is a circular button with a magnifying glass icon and the text 'UTC+08:00'.

7. You see a breakdown of the pages analyzed by each model.



## Estimate price

Now that we have the page processed data from the portal, we can use the Azure pricing calculator to estimate the cost:

1. Sign in to [Azure pricing calculator](#) with the same credentials you use for the Azure portal.

Press Ctrl + right-click to open in a new tab!

2. Search for Azure AI Document Intelligence in the **Search products** search box.
3. Select **Azure AI Document Intelligence** and you see it was added to the page.
4. Under **Your Estimate**, select the relevant **Region**, **Payment Option**, and **Instance** for your Document Intelligence resource. For more information, see [Azure AI Document Intelligence pricing options](#).
5. Enter the number of pages processed from the Azure portal metrics dashboard. That data can be found using the steps in sections [Check how many pages are processed](#) or [Examine analyzed pages](#).
6. The estimated price is on the right page section, after the equal (=) sign.

The screenshot shows the Azure AI Document Intelligence pricing calculator. At the top, there are dropdown menus for REGION (West US), PAYMENT OPTION (Pay as you go), and INSTANCE (S0). Below this, there are two sections: 'Custom' and 'Pre-built (S1)'. The 'Custom' section contains a row with a red box around the input field <number of pages> (labeled 'Pages'), a multiplier 'x', a price per page '\$ <price>' (labeled 'Per 1,000 pages'), and a red box around the total calculation '= \$ <subtotal>'. The 'Pre-built (S1)' section contains a similar row with a red box around the input field <number of pages> (labeled 'Pages'), a multiplier 'x', a price per page '\$ <price>' (labeled 'Per 1,000 pages'), and a red box around the total calculation '= \$ <subtotal>'. At the bottom, there are buttons for 'Upfront cost' and 'Monthly cost', and a red box around the total value '\$ <total>'. Navigation links include 'Azure Form Recognizer', 'Azure Form Recognizer, Pay as you go, Free: Up to ...', 'Upfront: \$0.00', and 'Monthly: \$'.

That's it. You now know where to find how many pages you process using Document Intelligence and how to estimate the cost.

## Next steps

[Learn more about Document Intelligence service quotas and limits](#)

# Create SAS tokens for storage containers

Article • 12/11/2024

This content applies to: ✓ v4.0 (GA) ✓ v3.1 (GA) ✓ v3.0 (GA) ✓ v2.1 (GA)

In this article, learn how to create user delegation, shared access signature (SAS) tokens, using either the Azure portal or Azure Storage Explorer. User delegation SAS tokens are secured with Microsoft Entra credentials. SAS tokens provide secure, delegated access to resources in your Azure storage account.

Storage resource URI	Delimiter character	SAS token
<a href="https://storagesample.blob.core.windows.net/sample-container/sampleBlob.pdf">https://storagesample.blob.core.windows.net/sample-container/sampleBlob.pdf</a>	?sv=2023-11-31&sr=b&sig=39Up9jzHkxhUIhFEjEh9594Dxe6cIRCgOv6ICGS0%3A377&sp=rcw	

At a high level, here's how SAS tokens work:

- First, your application submits the SAS token to Azure Storage as part of a REST API request.
- Next, if the storage service verifies that the SAS is valid, the request is authorized. If, the SAS token is deemed invalid, the request is declined and the error code 403 (Forbidden) is returned.

Azure Blob Storage offers three resource types:

- **Storage accounts** provide a unique namespace in Azure for your data.
- **Data storage containers** are located in storage accounts and organize sets of blobs.
- **Blobs** are located in containers and store text and binary data such as files, text, and images.

## When to use a SAS token

- **Training custom models.** Your assembled set of training documents *must* be uploaded to an Azure Blob Storage container. You can opt to use a SAS token to grant access to your training documents.
- **Using storage containers with public access.** You can opt to use a SAS token to grant limited access to your storage resources that have public read access.

 **Important**

- If your Azure storage account is protected by a virtual network or firewall, you can't grant access with a SAS token. You'll have to use a [managed identity](#) to grant access to your storage resource.
- [Managed identity](#) supports both privately and publicly accessible Azure Blob Storage accounts.
- SAS tokens grant permissions to storage resources, and should be protected in the same manner as an account key.
- Operations that use SAS tokens should be performed only over an HTTPS connection, and SAS URIs should only be distributed on a secure connection such as HTTPS.

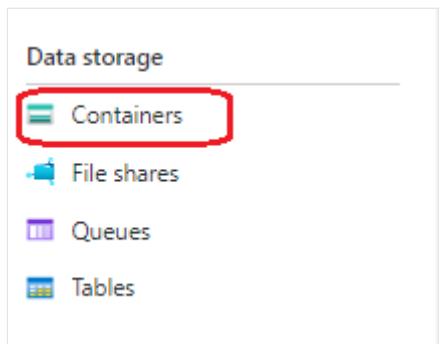
## Prerequisites

To get started, you need:

- An active [Azure account](#). If you don't have one, you can [create a free account](#).
- A [Document Intelligence](#) or [multi-service](#) resource.
- A **standard performance** [Azure Blob Storage account](#). You need to create containers to store and organize your blob data within your storage account. If you don't know how to create an Azure storage account with a storage container, follow these quickstarts:
  - [Create a storage account](#). When you create your storage account, select **Standard performance** in the **Instance details > Performance** field.
  - [Create a container](#). When you create your container, set **Public access level** to **Container** (anonymous read access for containers and blobs) in the **New Container** window.

## Upload your documents

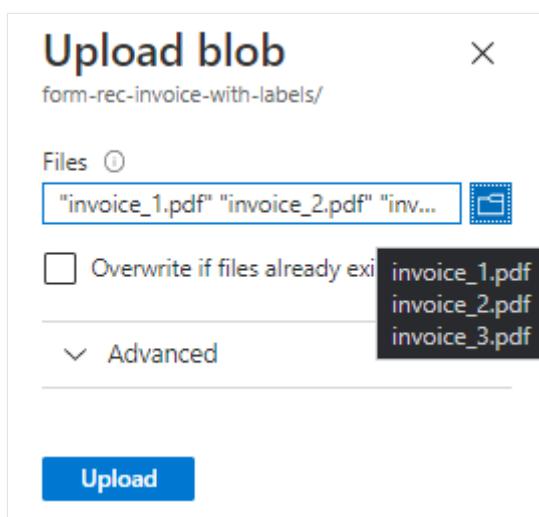
1. Sign in to the [Azure portal](#).
  - Select **Your storage account** → **Data storage** → **Containers**.



2. Select a container from the list.
3. Select **Upload** from the menu at the top of the page.



4. The **Upload blob** window appears. Select your files to upload.



#### ⚠ Note

By default, the REST API uses documents located at the root of your container. You can also use data organized in subfolders if specified in the API call. For more information, see [Organize your data in subfolders](#).

## Generating SAS tokens

Once the prerequisites are met and you upload your documents, you can now generate SAS tokens. There are two paths you can take from here; one using the Azure portal and the other using the Azure storage explorer. Select between the two following tabs for more information.

The Azure portal is a web-based console that enables you to manage your Azure subscription and resources using a graphical user interface (GUI).

1. Sign in to the [Azure portal](#).
2. Navigate to Your storage account > containers > your container.
3. Select **Generate SAS** from the menu near the top of the page.
4. Select **Signing method → User delegation key**.
5. Define **Permissions** by selecting or clearing the appropriate checkbox.
  - Make sure the **Read, Write, Delete, and List** permissions are selected.

**Generate SAS** X

A shared access signature (SAS) is a URI that grants restricted access to an Azure Storage container. Use it when you want to grant access to storage account resources for a specific time range without sharing your storage account key. [Learn more](#)

**Signing method**

Account key  User delegation key

**Permissions \*** ⓘ

4 selected ⓘ

<input checked="" type="checkbox"/> Read	2:19:14 PM
<input type="checkbox"/> Add	
<input type="checkbox"/> Create	
<input checked="" type="checkbox"/> Write	US & Canada)
<input checked="" type="checkbox"/> Delete	
<input checked="" type="checkbox"/> List	10:19:14 PM

(UTC-08:00) Pacific Time (US & Canada)

**Allowed IP addresses** ⓘ

for example, 168.1.5.65 or 168.1.5.65-168.1...

**Allowed protocols** ⓘ

HTTPS only  HTTPS and HTTP

**Generate SAS token and URL**

**ⓘ Important**

- If you receive a message similar to the following one, you'll also need to assign access to the blob data in your storage account:

**⚠️** You don't have permissions to grant read, write, delete access. You can still create a shared access signature, but you'll need an RBAC role with additional permissions before you can grant that level of access to your signature recipient.  
[Learn more about Azure roles for access to blob data](#)

- [\*\*Azure role-based access control\*\*](#) (Azure RBAC) is the authorization system used to manage access to Azure resources. Azure RBAC helps you manage access and permissions for your Azure resources.
- [\*\*Assign an Azure role for access to blob data\*\*](#) to assign a role that allows for read, write, and delete permissions for your Azure storage container. See [\*\*Storage Blob Data Contributor\*\*](#).

6. Specify the signed key **Start** and **Expiry** times.

- When you create a SAS token, the default duration is 48 hours. After 48 hours, you'll need to create a new token.
- Consider setting a longer duration period for the time you're using your storage account for Document Intelligence Service operations.
- The value of the expiry time is determined by whether you're using an **Account key** or **User delegation key Signing method**:
  - **Account key:** No imposed maximum time limit; however, best practices recommended that you configure an expiration policy to limit the interval and minimize compromise. [Configure an expiration policy for shared access signatures](#).
  - **User delegation key:** The value for the expiry time is a maximum of seven days from the creation of the SAS token. The SAS is invalid after the user delegation key expires, so a SAS with an expiry time of greater than seven days will still only be valid for seven days. For more information, see [Use Microsoft Entra credentials to secure a SAS](#).

7. The **Allowed IP addresses** field is optional and specifies an IP address or a range of IP addresses from which to accept requests. If the request IP address doesn't match the IP address or address range specified on the SAS token, authorization fails. The IP address or a range of IP addresses must be public IPs, not private. For more information, see, [Specify an IP address or IP range](#).

8. The **Allowed protocols** field is optional and specifies the protocol permitted for a request made with the SAS token. The default value is HTTPS.

9. Select **Generate SAS token and URL**.

10. The **Blob SAS token** query string and **Blob SAS URL** appear in the lower area of the window. To use the Blob SAS token, append it to a storage service URI.
11. Copy and paste the **Blob SAS token** and **Blob SAS URL** values in a secure location. The values are displayed only once and can't be retrieved after the window is closed.
12. To [construct a SAS URL](#), append the SAS token (URI) to the URL for a storage service.

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence v4.0 migration

Article • 02/09/2025

## ⓘ Important

Document Intelligence REST API v4.0 introduces breaking changes in the REST API request and analyze response JSON.

## Migrating from v3.1 to v4.0

Preview APIs are periodically deprecated. If you're using a preview API version, update your application to target the GA API version. To migrate from a preview API version to the [2024-11-30 \(GA\)](#) API version using the SDK, update to the [current version of the language specific SDK](#).

## Analysis features

[Expand table](#)

Model ID	Text Extraction	Paragraphs	Paragraph Roles	Selection Marks	Tables	Key-Value Pairs	Languages	Barcodes	Document Analysis	Formulas*	StyleFont*	OCR Resol
prebuilt-read	✓	✓					O	O		O	O	
prebuilt-layout	✓	✓	✓	✓	✓		O	O		O	O	
prebuilt-document	✓	✓	✓	✓	✓	✓	O	O		O	O	
prebuilt-businessCard	✓									✓		
prebuilt-idDocument	✓						O	O	✓	O	O	
prebuilt-invoice	✓			✓	✓	O	O	O	✓	O	O	
prebuilt-receipt	✓						O	O	✓	O	O	
prebuilt-healthInsuranceCard.us	✓						O	O	✓	O	O	
prebuilt-tax.us.w2	✓			✓			O	O	✓	O	O	
prebuilt-tax.us.1098	✓			✓			O	O	✓	O	O	
prebuilt-tax.us.1098E	✓			✓			O	O	✓	O	O	
prebuilt-tax.us.1098T	✓			✓			O	O	✓	O	O	
prebuilt-contract	✓	✓	✓	✓			O	O	✓	O	O	
{ customModelName }	✓	✓	✓	✓	✓		O	O	✓	O	O	

✓ - Enabled O - Optional Formulas/StyleFont/OCR High Resolution\* - Premium features incur added costs

## Migrating from v3.0

Compared with v3.0, Document Intelligence v3.1 introduces several new features and capabilities:

- [Barcode](#) extraction.
- [Add-on capabilities](#) including high resolution, formula, and font properties extraction.
- [Custom classification model](#) for document splitting and classification.
- Language expansion and new fields support in [Invoice](#) and [Receipt](#) model.
- New document type support in [ID document](#) model.
- New prebuilt [Health insurance card](#) model.
- Office/HTML files are supported in prebuilt-read model, extracting words and paragraphs without bounding boxes. Embedded images are no longer supported. If add-on features are requested for Office/HTML files, an empty array is returned without errors.
- Model expiration for custom extraction and classification models - Our new custom models build upon a large base model that we update periodically for quality improvement. An expiration date is introduced to all custom models to enable the retirement of the corresponding base models. Once a custom model expires, you need to retrain the model using the latest API version (base model).

## HTTP

```
GET /documentModels/{customModelId}?api-version={apiVersion}
{
  "modelId": "{customModelId}",
  "description": "{customModelDescription}",
  "createdDateTime": "2023-09-24T12:54:35Z",
  "expirationDateTime": "2025-01-01T00:00:00Z",
  "apiVersion": "2023-07-31",
  "docTypes": { ... }
}
```

- Custom neural model build quota - The number of neural models each subscription can build per region every month is limited. We expand the result JSON to include the quota and used information to help you understand the current usage as part of the resource information returned by GET /info.

## HTTP

```
{
  "customDocumentModels": { ... },
  "customNeuralDocumentModelBuilds": {
    "used": 1,
    "quota": 10,
    "quotaResetDateTime": "2023-03-01T00:00:00Z"
  }
}
```

- An optional `features` query parameter to Analyze operations can optionally enable specific features. Some premium features can incur added billing. Refer to [Analyze feature list](#) for details.
- Extend extracted currency field objects to output a normalized currency code field when possible. Currently, current fields can return amount (ex. 123.45) and currencySymbol (ex. \$). This feature maps the currency symbol to a canonical ISO 4217 code (ex. USD). The model can optionally utilize the global document content to disambiguate or infer the currency code.

## HTTP

```
{
  "fields": {
    "Total": {
      "type": "currency",
      "content": "$123.45",
      "valueCurrency": {
        "amount": 123.45,
        "currencySymbol": "$",
        "currencyCode": "USD"
      },
      ...
    }
  }
}
```

Besides model quality improvement, you're highly recommended to update your application to use v3.1 to benefit from these new capabilities.

## Migrating from v2.1 or v2.0

Document Intelligence v3.1 is the latest GA version with the richest features, most languages and document types coverage, and improved model quality. Refer to [model overview](#) for the features and capabilities available in v3.1.

Starting from v3.0, [Document Intelligence REST API](#) is redesigned for better usability. In this section, learn the differences between Document Intelligence v2.0, v2.1 and v3.1 and how to move to the newer version of the API.

### ⊗ Caution

- REST API 2023-07-31 release includes a breaking change in the REST API analyze response JSON.
- The `boundingBox` property is renamed to `polygon` in each instance.

## Changes to the REST API endpoints

The v3.1 REST API combines the analysis operations for layout analysis, prebuilt models, and custom models into a single pair of operations by assigning `documentModels` and `modelId` to the [layout analysis](#) and prebuilt models.

## POST request

HTTP

<https://{{your-form-recognizer-endpoint}}/formrecognizer/documentModels/{{modelId}}?api-version=2023-07-31>

## GET request

HTTP

<https://{{your-form-recognizer-endpoint}}/formrecognizer/documentModels/{{modelId}}/AnalyzeResult/{{resultId}}?api-version=2023-07-31>

## Analyze operation

- The request payload and call pattern remain unchanged.
- The `Analyze` operation specifies the input document and content-specific configurations, it returns the analyzed result URL via the Operation-Location header in the response.
- Poll the `Analyze Result` URL, via a GET request to check the status of the `Analyze` operation (minimum recommended interval between requests is 1 second).
- Upon success, status is set to succeeded and `analyzeResult` is returned in the response body. If errors are encountered, status sets to `failed`, and an error is returned.

[+] Expand table

Model	v2.0	v2.1	v3.1
Request URL prefix	<a href="https://{{your-form-recognizer-endpoint}}/formrecognizer/v2.0">https://{{your-form-recognizer-endpoint}}/formrecognizer/v2.0</a>	<a href="https://{{your-form-recognizer-endpoint}}/formrecognizer/v2.1">https://{{your-form-recognizer-endpoint}}/formrecognizer/v2.1</a>	<a href="https://{{your-form-recognizer-endpoint}}/formrecognizer">https://{{your-form-recognizer-endpoint}}/formrecognizer</a>
General document	N/A	N/A	<code>/documentModels/prebuilt-document:analyze</code>
Layout	<code>/layout/analyze</code>	<code>/layout/analyze</code>	<code>/documentModels/prebuilt-layout:analyze</code>
Custom	<code>/custom/models/{{modelId}}/analyze</code>	<code>/custom/{{modelId}}/analyze</code>	<code>/documentModels/{{modelId}}:analyze</code>
Invoice	N/A	<code>/prebuilt/invoice/analyze</code>	<code>/documentModels/prebuilt-invoice:analyze</code>
Receipt	<code>/prebuilt/receipt/analyze</code>	<code>/prebuilt/receipt/analyze</code>	<code>/documentModels/prebuilt-receipt:analyze</code>
ID document	N/A	<code>/prebuilt/idDocument/analyze</code>	<code>/documentModels/prebuilt-idDocument:analyze</code>
Business card	N/A	<code>/prebuilt/businessCard/analyze</code>	<code>/documentModels/prebuilt-businessCard:analyze</code>
W-2	N/A	N/A	<code>/documentModels/prebuilt-tax.us.w2:analyze</code>
Health insurance card	N/A	N/A	<code>/documentModels/prebuilt-healthInsuranceCard.us:analyze</code>
Contract	N/A	N/A	<code>/documentModels/prebuilt-contract:analyze</code>

## Analyze request body

The content to be analyzed is provided via the request body. Either the URL or base64 encoded data can be user to construct the request.

To specify a publicly accessible web URL, set Content-Type to `application/json` and send the following JSON body:

JSON

```
{  
    "urlSource": "{urlPath}"
```

```
}
```

Base 64 encoding is also supported in Document Intelligence v3.0:

JSON

```
{  
  "base64Source": "{base64EncodedContent}"  
}
```

## Additionally supported parameters

Parameters that continue to be supported:

- `pages` : Analyze only a specific subset of pages in the document. List of page numbers indexed from the number `1` to analyze. Ex. "1-3,5,7-9"
- `locale` : Locale hint for text recognition and document analysis. Value can contain only the language code (ex. `en`, `fr`) or BCP 47 language tag (ex. "en-US").

Parameters no longer supported:

- `includeTextDetails`

The new response format is more compact and the full output is always returned.

## Changes to analyze result

Analyze response is refactored to the following top-level results and supports multi-page elements.

- `pages`
- `tables`
- `keyValuePairs`
- `entities`
- `styles`
- `documents`

### ① Note

The `analyzeResult` response changes include changes such as moving up from a property of `pages` to a top level property within `analyzeResult`.

JSON

```
{  
  // Basic analyze result metadata  
  "apiVersion": "2022-08-31", // REST API version used  
  "modelId": "prebuilt-invoice", // ModelId used  
  "stringIndexType": "textElements", // Character unit used for string offsets and lengths:  
  // textElements, unicodeCodePoint, utf16CodeUnit // Concatenated content in global reading order across pages.  
  // Words are generally delimited by space, except CJK (Chinese, Japanese, Korean) characters.  
  // Lines and selection marks are generally delimited by newline character.  
  // Selection marks are represented in Markdown emoji syntax (:selected:, :unselected:).  
  "content": "CONTOSO LTD.\nINVOICE\nContoso Headquarters...", "pages": [ // List of pages analyzed  
    {  
      // Basic page metadata  
      "pageNumber": 1, // 1-indexed page number  
      "angle": 0, // Orientation of content in clockwise direction (degree)  
      "width": 0, // Page width  
      "height": 0, // Page height  
      "unit": "pixel", // Unit for width, height, and polygon coordinates  
      "spans": [ // Parts of top-level content covered by page  
        {  
          "offset": 0, // Offset in content  
          "length": 7 // Length in content  
        }  
      ], // List of words in page  
      "words": [  
        {  
          "text": "CONTOSO", // Equivalent to $.content.Substring(span.offset, span.length)  
        }  
      ]  
    }  
  ]  
}
```

```
"boundingBox": [ ... ], // Position in page
"confidence": 0.99, // Extraction confidence
"span": { ... } // Part of top-level content covered by word
}, ...
], // List of selectionMarks in page
"selectionMarks": [
{
"state": "selected", // Selection state: selected, unselected
"boundingBox": [ ... ], // Position in page
"confidence": 0.95, // Extraction confidence
"span": { ... } // Part of top-level content covered by selection mark
}, ...
], // List of lines in page
"lines": [
{
"content": "CONTOSO LTD.", // Concatenated content of line (may contain both words and selectionMarks)
"boundingBox": [ ... ], // Position in page
"spans": [ ... ], // Parts of top-level content covered by line
}, ...
]
]
},
], ...
], // List of extracted tables
"tables": [
{
"rowCount": 1, // Number of rows in table
"columnCount": 1, // Number of columns in table
"boundingRegions": [ // Polygons or Bounding boxes potentially across pages covered by table
{
"pageNumber": 1, // 1-indexed page number
"polygon": [ ... ], // Previously Bounding box, renamed to polygon in the 2022-08-31 API
}
],
"spans": [ ... ], // Parts of top-level content covered by table // List of cells in table
"cells": [
{
"kind": "stub", // Cell kind: content (default), rowHeader, columnHeader, stub, description
"rowIndex": 0, // 0-indexed row position of cell
"columnIndex": 0, // 0-indexed column position of cell
"rowSpan": 1, // Number of rows spanned by cell (default=1)
"columnSpan": 1, // Number of columns spanned by cell (default=1)
"content": "SALESPERSON", // Concatenated content of cell
"boundingRegions": [ ... ], // Bounding regions covered by cell
"spans": [ ... ] // Parts of top-level content covered by cell
}, ...
]
]
},
], ...
], // List of extracted key-value pairs
"keyValuePairs": [
{
"key": { // Extracted key
"content": "INVOICE:", // Key content
"boundingRegions": [ ... ], // Key bounding regions
"spans": [ ... ] // Key spans
},
"value": { // Extracted value corresponding to key, if any
"content": "INV-100", // Value content
"boundingRegions": [ ... ], // Value bounding regions
"spans": [ ... ] // Value spans
},
"confidence": 0.95 // Extraction confidence
}, ...
],
"styles": [
{
"isHandwritten": true, // Is content in this style handwritten?
"spans": [ ... ], // Spans covered by this style
"confidence": 0.95 // Detection confidence
}, ...
], // List of extracted documents
"documents": [
{
"docType": "prebuilt-invoice", // Classified document type (model dependent)
"boundingRegions": [ ... ], // Document bounding regions
"spans": [ ... ], // Document spans
"confidence": 0.99, // Document splitting/classification confidence // List of extracted fields
"fields": [
{
"VendorName": { // Field name (docType dependent)
"type": "string", // Field value type: string, number, array, object, ...
"valueString": "CONTOSO LTD.", // Normalized field value
"content": "CONTOSO LTD.", // Raw extracted field content
"boundingRegions": [ ... ], // Field bounding regions
"spans": [ ... ], // Field spans
}
]
```

```
"confidence": 0.99 // Extraction confidence
},
}
},
]
}
```

## Build or train model

The model object has three updates in the new API

- `modelId` is now a property that can be set on a model for a human readable name.
- `modelName` is renamed to `description`
- `buildMode` is a new property with values of `template` for custom form models or `neural` for custom neural models.

The `build` operation is invoked to train a model. The request payload and call pattern remain unchanged. The build operation specifies the model and training dataset, it returns the result via the Operation-Location header in the response. Poll this model operation URL, via a GET request to check the status of the build operation (minimum recommended interval between requests is 1 second). Unlike v2.1, this URL isn't the resource location of the model. Instead, the model URL can be constructed from the given modelId, also retrieved from the `resourceLocation` property in the response. Upon success, status is set to `succeeded` and result contains the custom model info. If errors are encountered, status is set to `failed`, and the error is returned.

The following code is a sample build request using a SAS token. Note the trailing slash when setting the prefix or folder path.

JSON

```
POST https://{{your-form-recognizer-endpoint}}/formrecognizer/documentModels:build?api-version=2022-08-31

{
  "modelId": {{modelId}},
  "description": "Sample model",
  "buildMode": "template",
  "azureBlobSource": {
    "containerUrl": "https://{{storageAccount}}.blob.core.windows.net/{{containerName}}?{{sasToken}}",
    "prefix": "{{folderName}}/"
  }
}
```

## Changes to compose model

Model compose is now limited to single level of nesting. Composed models are now consistent with custom models with the addition of `modelId` and `description` properties.

JSON

```
POST https://{{your-form-recognizer-endpoint}}/formrecognizer/documentModels:compose?api-version=2022-08-31
{
  "modelId": "{{composedModelId}}",
  "description": "{{composedModelDescription}}",
  "componentModels": [
    { "modelId": "{{modelId1}}" },
    { "modelId": "{{modelId2}}" },
  ]
}
```

## Changes to copy model

The call pattern for copy model remains unchanged:

- Authorize the copy operation with the target resource calling `authorizeCopy`. Now a POST request.
- Submit the authorization to the source resource and copy the model calling `copyTo`
- Poll the returned operation to validate the operation completed successfully

The only changes to the copy model function are:

- HTTP action on the `authorizeCopy` is now a POST request.
- The authorization payload contains all the information needed to submit the copy request.

#### *Authorize the copy*

JSON

```
POST https://{{targetHost}}/formrecognizer/documentModels:authorizeCopy?api-version=2022-08-31
{
  "modelId": "{targetModelId}",
  "description": "{targetModelDescription}",
}
```

Use the response body from the authorize action to construct the request for the copy.

JSON

```
POST https://{{sourceHost}}/formrecognizer/documentModels/{{sourceModelId}}:copyTo?api-version=2022-08-31
{
  "targetResourceId": "{targetResourceId}",
  "targetResourceRegion": "{targetResourceRegion}",
  "targetModelId": "{targetModelId}",
  "targetModelLocation": "https://{{targetHost}}/formrecognizer/documentModels/{{targetModelId}}",
  "accessToken": "{accessToken}",
  "expirationDateTime": "2021-08-02T03:56:11Z"
}
```

## Changes to list models

List models are extended to now return prebuilt and custom models. All prebuilt model names start with `prebuilt-`. Only models with a status of succeeded are returned. To list models that either failed or are in progress, see [List Operations](#).

#### *Sample list models request*

JSON

```
GET https://{{your-form-recognizer-endpoint}}/formrecognizer/documentModels?api-version=2022-08-31
```

## Change to get model operation

As `Get Model` now includes prebuilt models, the `Get` operation returns a `docTypes` dictionary. Each document type description includes name, optional description, field schema, and optional field confidence. The field schema describes the list of fields potentially returned with the document type.

JSON

```
GET https://{{your-form-recognizer-endpoint}}/formrecognizer/documentModels/{{modelId}}?api-version=2022-08-31
```

## New get info operation

The `info` operation on the service returns the custom model count and custom model limit.

JSON

```
GET https://{{your-form-recognizer-endpoint}}/formrecognizer/info? api-version=2022-08-31
```

#### *Sample response*

JSON

```
{
  "customDocumentModels": {
    "count": 5,
    "limit": 100
}
```

```
    }
```

## Next steps

- [Review the new REST API](#)
- [What is Document Intelligence?](#)
- [Document Intelligence quickstart 0](#)

# Build and train a custom extraction model

Article • 12/11/2024

This content applies to: ✓ v4.0 (GA) | Previous versions: ✓ v3.1 (GA) ✓ v3.0 (GA) ✓ v2.1

Document Intelligence custom models require a handful of training documents to get started. If you have at least five documents, you can get started training a custom model. You can train either a [custom template model \(custom form\)](#) or a [custom neural model \(custom document\)](#). This document walks you through the process of training the custom models.

## Custom model input requirements

First, make sure your training data set follows the input requirements for Document Intelligence.

- Supported file formats:

[+] Expand table

Model	PDF	Image:	Microsoft Office:
		JPEG/JPG, PNG, BMP, TIFF, HEIF	Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.
- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).

- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

## Training data tips

Follow these tips to further optimize your data set for training:

- Use text-based PDF documents instead of image-based documents. Scanned PDFs are handled as images.
- Use examples that have all of the fields completed for forms with input fields.
- Use forms with different values in each field.
- Use a larger data set (10-15 images) if your form images are of lower quality.

## Upload your training data

Once you gather a set of forms or documents for training, you need to upload it to an Azure blob storage container. If you don't know how to create an Azure storage account with a container, following the [Azure Storage quickstart for Azure portal](#). You can use the free pricing tier (F0) to try the service, and upgrade later to a paid tier for production.

## Video: Train your custom model

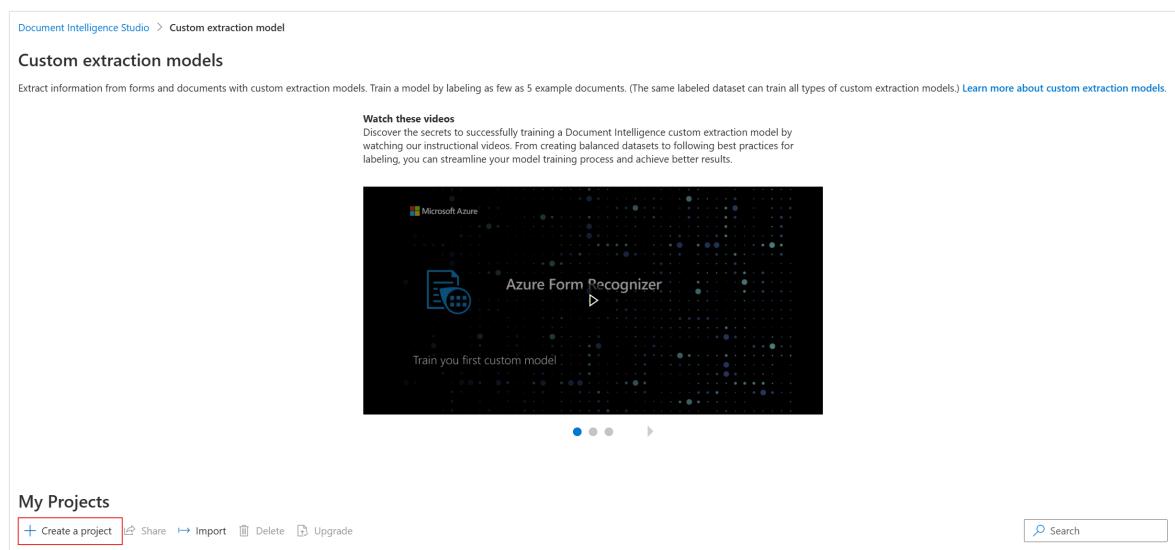
- Once you gather and upload your training dataset, you're ready to train your custom model. In the following video, we create a project and explore some of the fundamentals for successfully labeling and training a model.

[https://www.microsoft.com/en-us/videoplayer/embed/RE5fX1c?postJs||Msg=true ↗](https://www.microsoft.com/en-us/videoplayer/embed/RE5fX1c?postJs||Msg=true)

## Create a project in the Document Intelligence Studio

The Document Intelligence Studio provides and orchestrates all the API calls required to complete your dataset and train your model.

- Start by navigating to the [Document Intelligence Studio](#). The first time you use the Studio, you need to [initialize your subscription, resource group, and resource](#). Then, follow the [prerequisites for custom projects](#) to configure the Studio to access your training dataset.
- In the Studio, select the **Custom extraction model** tile and select the **Create a project** button.



- On the `create project` dialog, provide a name for your project, optionally a description, and select continue.
- On the next step in the workflow, choose or create a Document Intelligence resource before you select continue.

### Important

Custom neural models are only available in a few regions. If you plan on training a neural model, please select or create a resource in one of [these](#)

## supported regions.

Custom extraction model

<ul style="list-style-type: none"><li><input checked="" type="checkbox"/> Enter project details</li><li><input checked="" type="checkbox"/> Configure service resource</li><li><input type="radio"/> Connect training data source</li><li><input type="radio"/> Review and create</li></ul>	<p><b>Configure service resource</b></p> <p>To create a project in Document Intelligence Studio, you'll need an Azure subscription containing a service resource for usage and billing.</p> <p><b>Subscription *</b></p> <p>Select existing</p> <p><b>Resource group *</b></p> <p>Create new or select existing</p> <p>Create new</p> <p><b>Document Intelligence or Cognitive Service Resource *</b></p> <p>Select existing</p> <p><input type="checkbox"/> Create new resource    <input type="checkbox"/> Set as default</p> <p><b>API version *</b></p> <p>2023-10-31 (Preview)</p> <p>API version can only be changed by upgrading after the project is created.</p>
---	---

3. Next select the storage account you used to upload your custom model training dataset. The **Folder path** should be empty if your training documents are in the root of the container. If your documents are in a subfolder, enter the relative path from the container root in the **Folder path** field. Once your storage account is configured, select continue.

Custom extraction model

<ul style="list-style-type: none"><li><input checked="" type="checkbox"/> Enter project details</li><li><input checked="" type="checkbox"/> Configure service resource</li><li><input checked="" type="checkbox"/> Connect training data source</li><li><input type="radio"/> Review and create</li></ul>	<p><b>Connect training data source</b></p> <p>Link the Azure Blob Storage account and the folder that contains your training data. <a href="#">Learn more</a></p> <p><b>Subscription *</b></p> <p>Select existing</p> <p><b>Resource group *</b></p> <p>Create new or select existing</p> <p><b>Storage account *</b></p> <p>Select a storage account</p> <p><input type="checkbox"/> Create new storage account    <input type="checkbox"/> Set as default</p> <p><b>Blob container *</b></p> <p>Select a blob container</p> <p>Create new</p> <p><b>Folder path</b></p> <p>Enter folder path</p> <p><b>Back</b>    <b>Continue</b>    <b>Cancel</b></p>
---	---

4. Finally, review your project settings and select **Create Project** to create a new project. You should now be in the labeling window and see the files in your dataset

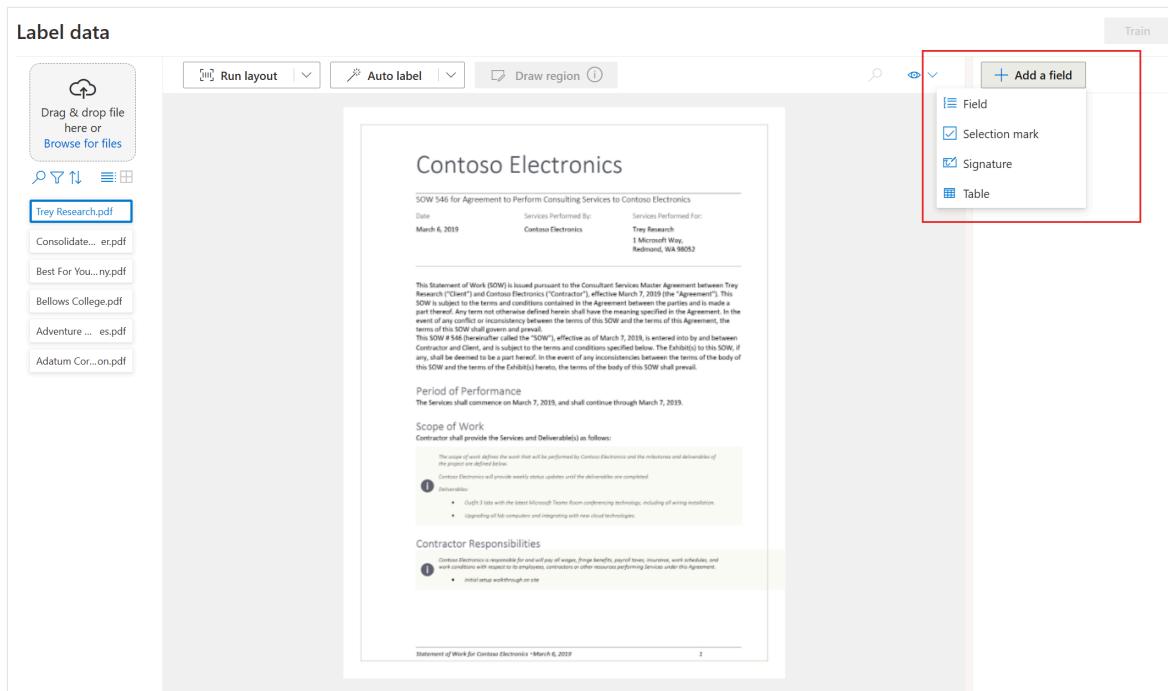
listed.

## Label your data

In your project, your first task is to label your dataset with the fields you wish to extract.

The files you uploaded to storage are listed on the left of your screen, with the first file ready to be labeled.

1. Start labeling your dataset and creating your first field by selecting the plus (+) button on the top-right of the screen.



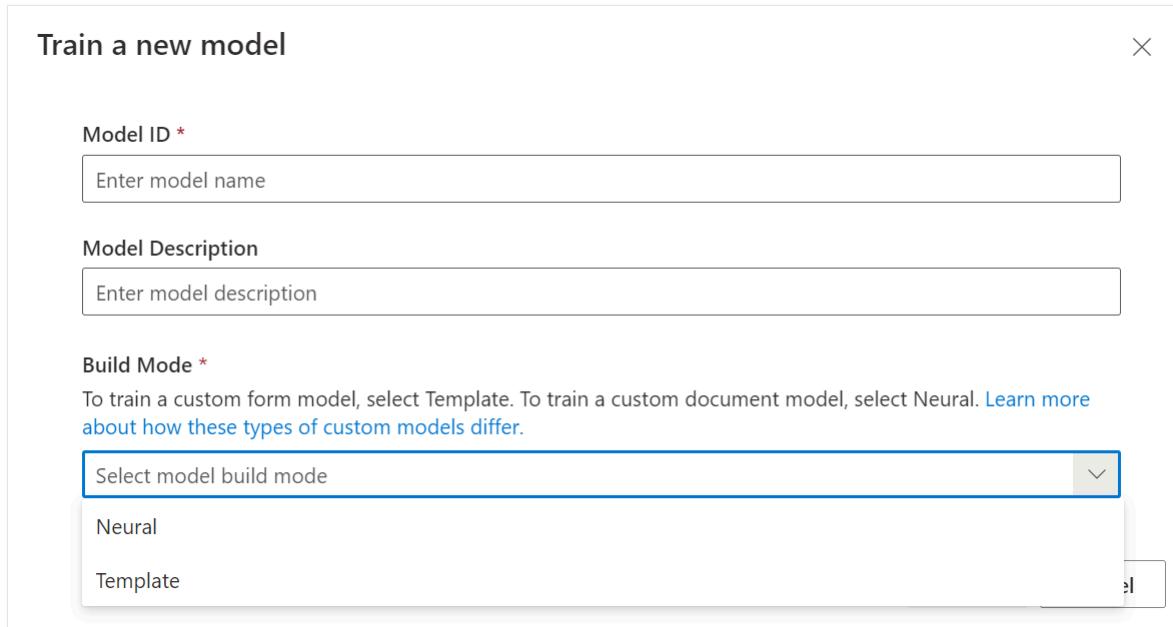
2. Enter a name for the field.
3. Assign a value to the field by choosing a word or words in the document. Select the field in either the dropdown or the field list on the right navigation bar. The labeled value is below the field name in the list of fields.
4. Repeat the process for all the fields you wish to label for your dataset.
5. Label the remaining documents in your dataset by selecting each document and selecting the text to be labeled.

You now have all the documents in your dataset labeled. The `.labels.json` and `.ocr.json` files correspond to each document in your training dataset and a new `fields.json` file. This training dataset is submitted to train the model.

## Train your model

With your dataset labeled, you're now ready to train your model. Select the train button in the upper-right corner.

1. On the train model dialog, provide a unique model ID and, optionally, a description. The model ID accepts a string data type.
2. For the build mode, select the type of model you want to train. Learn more about the [model types and capabilities](#).



3. Select **Train** to initiate the training process.
4. Template models train in a few minutes. Neural models can take up to 30 minutes to train.
5. Navigate to the *Models* menu to view the status of the train operation.

## Test the model

Once the model training is complete, you can test your model by selecting the model on the models list page.

1. Select the model and select on the **Test** button.
2. Select the **+ Add** button to select a file to test the model.
3. With a file selected, choose the **Analyze** button to test the model.
4. The model results are displayed in the main window and the fields extracted are listed in the right navigation bar.
5. Validate your model by evaluating the results for each field.

6. The right navigation bar also has the sample code to invoke your model and the JSON results from the API.

Congratulations you learned to train a custom model in the Document Intelligence Studio! Your model is ready for use with the REST API or the SDK to analyze documents.

## Next steps

Now that you learned how to build a training data set, follow a quickstart to train a custom Document Intelligence model and start using it on your forms.

[Learn about custom model types](#)

[Learn about accuracy and confidence with custom models](#)

## See also

- [Train a model and extract document data using the client library or REST API](#)
- [What is Document Intelligence?](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Build and train a custom classification model

Article • 12/11/2024

This content applies to: ✓ v4.0 (GA) | Previous versions: ✓ v3.1 (GA) ✓ v3.0 (GA)

Custom classification models can classify each page in an input file to identify one or more documents within. Classifier models can also identify multiple documents or multiple instances of a single document in the input file. Document Intelligence custom models require as few as five training documents per document class to get started. To get started training a custom classification model, you need at least **five documents** for each class and **two classes** of documents.

## Custom classification model input requirements

Make sure your training data set follows the input requirements for Document Intelligence.

- Supported file formats:

Expand table

Model	PDF	Image: JPEG/JPG, PNG, BMP, TIFF, HEIF	Microsoft Office: Word (DOCX), Excel (XLSX), PowerPoint (PPTX), HTML
Read	✓	✓	✓
Layout	✓	✓	✓
General Document	✓	✓	
Prebuilt	✓	✓	
Custom extraction	✓	✓	
Custom classification	✓	✓	✓

- For best results, provide one clear photo or high-quality scan per document.

- For PDF and TIFF, up to 2,000 pages can be processed (with a free tier subscription, only the first two pages are processed).
- The file size for analyzing documents is 500 MB for paid (S0) tier and 4 MB for free (F0) tier.
- Image dimensions must be between 50 pixels x 50 pixels and 10,000 pixels x 10,000 pixels.
- If your PDFs are password-locked, you must remove the lock before submission.
- The minimum height of the text to be extracted is 12 pixels for a 1024 x 768 pixel image. This dimension corresponds to about 8 point text at 150 dots per inch (DPI).
- For custom model training, the maximum number of pages for training data is 500 for the custom template model and 50,000 for the custom neural model.
  - For custom extraction model training, the total size of training data is 50 MB for template model and 1 GB for the neural model.
  - For custom classification model training, the total size of training data is 1 GB with a maximum of 10,000 pages. For 2024-11-30 (GA), the total size of training data is 2 GB with a maximum of 10,000 pages.

## Training data tips

Follow these tips to further optimize your data set for training:

- If possible, use text-based PDF documents instead of image-based documents. Scanned PDFs are handled as images.
- If your form images are of lower quality, use a larger data set (10-15 images, for example).

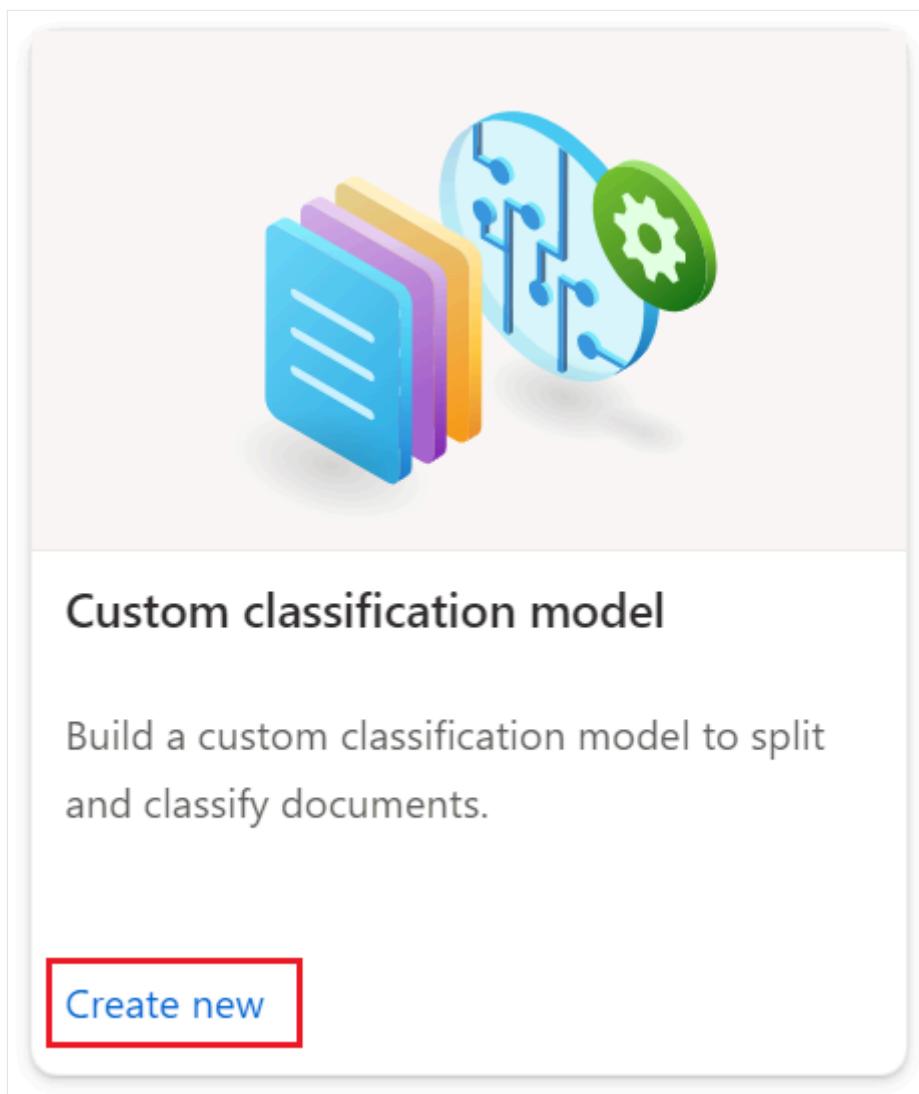
## Upload your training data

Once you put together the set of forms or documents for training, you need to upload it to an Azure blob storage container. If you don't know how to create an Azure storage account with a container, follow the [Azure Storage quickstart for Azure portal](#). You can use the free pricing tier (F0) to try the service, and upgrade later to a paid tier for production. If your dataset is organized as folders, preserve that structure as the Studio can use your folder names for labels to simplify the labeling process.

# Create a classification project in the Document Intelligence Studio

The Document Intelligence Studio provides and orchestrates all the API calls required to complete your dataset and train your model.

1. Start by navigating to the [Document Intelligence Studio](#). The first time you use the Studio, you need to [initialize your subscription, resource group, and resource](#). Then, follow the [prerequisites for custom projects](#) to configure the Studio to access your training dataset.
2. In the Studio, select the **Custom classification model** tile, on the custom models section of the page and select the **Create a project** button.



- a. On the `Create Project` dialog, provide a name for your project, optionally a description, and select continue.
- b. Next, choose, or select create a Document Intelligence resource before you continue.

Custom models

X

<ul style="list-style-type: none"><li><input checked="" type="radio"/> Enter project details</li><li><input checked="" type="radio"/> Configure service resource</li><li><input type="radio"/> Connect training data source</li><li><input type="radio"/> Review and create</li></ul>	<p><b>Configure service resource</b></p> <p>To create a project in Form Recognizer Studio, you will need an Azure subscription containing a service resource for usage and billing. Resources are organized in resource groups. <a href="#">Learn more</a></p> <p>Subscription *</p> <p>Resource group *</p> <p>Create new</p> <p>Form Recognizer or Cognitive Service Resource *</p> <p><input type="checkbox"/> Create new resource</p> <p>API version *</p>
---	--

3. Next select the storage account you used to upload your custom model training dataset. The **Folder path** should be empty if your training documents are in the root of the container. If your documents are in a subfolder, enter the relative path from the container root in the **Folder path** field. Once your storage account is configured, select continue.

**Important**

You can either organize the training dataset by folders where the folder name is the label or class for documents or create a flat list of documents that you can assign a label to in the Studio.

Custom models

Enter project details

Configure service resource

Connect training data source

Review and create

Connect training data source

Link the Azure Blob Storage account and the folder that contains your training data. [Learn more](#)

Subscription \*

Resource group \*

Storage account \*

Blob container \*

Folder path

Back Continue Cancel

4. Training a custom classifier requires the output from the Layout model for each document in your dataset. Run layout on all documents before the model training process.
5. Finally, review your project settings and select **Create Project** to create a new project. You should now be in the labeling window and see the files in your dataset listed.

## Label your data

In your project, you only need to label each document with the appropriate class label.

The screenshot shows the Microsoft Form Recognizer Studio interface. At the top, it says "Applied AI | Form Recognizer Studio" and "Form Recognizer Studio > Custom classification model". Below this, the title "Custom classification models" is displayed. A message states "Custom classification model welcome message (To be update). Learn more about custom classification model." On the left, there's a "My Projects" sidebar with a "+ Create a project" button and a "Create a project" link. The main area shows a file list with documents named "simple form 1.pdf" through "simple form 10.pdf". A preview pane on the right shows a document with various labels applied to its fields. At the bottom, there's a search bar and a message "Create a new project to get started."

You see the files you uploaded to storage in the file list, ready to be labeled. You have a few options to label your dataset.

1. If the documents are organized in folders, the Studio prompts you to use the folder names as labels. This step simplifies your labeling down to a single select.
2. To assign a label to a document, select on the `add label selection mark` to assign a label.
3. Control select to multi-select documents to assign a label

You should now have all the documents in your dataset labeled. If you look at the storage account, you find `.ocr.json` files that correspond to each document in your training dataset and a new `class-name.jsonl` file for each class labeled. This training dataset is submitted to train the model.

## Train your model

With your dataset labeled, you're now ready to train your model. Select the train button in the upper-right corner.

1. On the train model dialog, provide a unique classifier ID and, optionally, a description. The classifier ID accepts a string data type.
2. Select **Train** to initiate the training process.
3. Classifier models train in a few minutes.

4. Navigate to the *Models* menu to view the status of the train operation.

## Test the model

Once the model training is complete, you can test your model by selecting the model on the models list page.

1. Select the model and select on the **Test** button.
2. Add a new file by browsing for a file or dropping a file into the document selector.
3. With a file selected, choose the **Analyze** button to test the model.
4. The model results are displayed with the list of identified documents, a confidence score for each document identified and the page range for each of the documents identified.
5. Validate your model by evaluating the results for each document identified.

## Training a custom classifier using the SDK or API

The Studio orchestrates the API calls for you to train a custom classifier. The classifier training dataset requires the output from the layout API that matches the version of the API for your training model. Using layout results from an older API version can result in a model with lower accuracy.

The Studio generates the layout results for your training dataset if the dataset doesn't contain layout results. When using the API or SDK to train a classifier, you need to add the layout results to the folders containing the individual documents. The layout results should be in the format of the API response when calling layout directly. The SDK object model is different. Make sure that the `layout results` are the API results and not the `SDK response`.

## Troubleshoot

The [classification model](#) requires results from the [layout model](#) for each training document. If you don't provide the layout results, the Studio attempts to run the layout model for each document before training the classifier. This process is throttled and can result in a 429 response.

In the Studio, before training with the classification model, run the [layout model](#) on each document and upload it to the same location as the original document. Once the layout results are added, you can train the classifier model with your documents.

## Next steps

[Learn about custom model types](#)

[Learn about accuracy and confidence with custom models](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Project sharing using Document Intelligence Studio

Article • 11/19/2024

This content applies to: v4.0 (GA) | Previous versions: v3.1 (GA) v3.0 (GA) v2.1 (GA)

Document Intelligence Studio is an online tool to visually explore, understand, train, and integrate features from the Document Intelligence service into your applications. Document Intelligence Studio enables project sharing feature within the custom extraction model. Projects can be shared easily via a project token. The same project token can also be used to import a project.

## Prerequisite

In order to share and import your custom projects seamlessly, both users (user who shares and user who imports) need an active [Azure account](#). If you don't have one, you can [create a free account](#). Also, both users need to configure permissions to grant access to the Document Intelligence and storage resources.

Generally, in the process of creating a custom model project, most of the requirements should be met for project sharing. However, in cases where the project sharing feature doesn't work, check [permissions](#).

## Granted access and permissions

### Important

Custom model projects can be imported only if you have the access to the storage account that is associated with the project you are trying to import. Check your storage account permission before starting to share or import projects with others.

## Virtual networks and firewalls

If your storage account virtual network (VNet) is enabled or if there are any firewall constraints, the project can't be shared. If you want to bypass those restrictions, ensure that those settings are turned off.

A workaround is to manually create a project using the same settings as the project being shared.

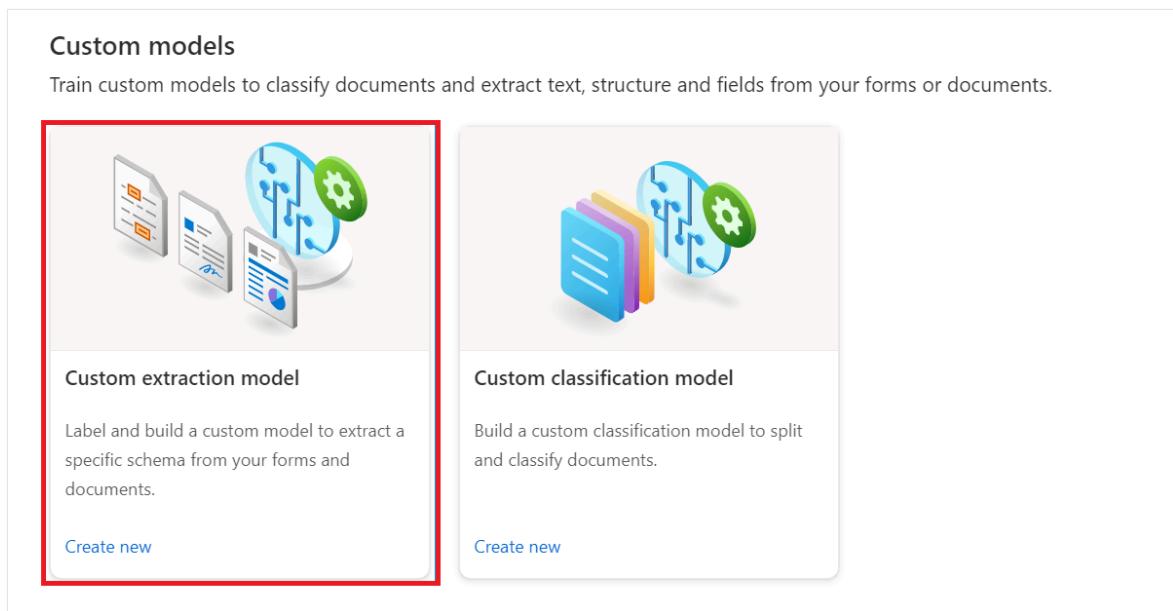
# Share a custom extraction model with Document Intelligence Studio

## ⓘ Note

Custom classification model projects can also be shared following the same step, starting with the following [page](#). In this guide, we use custom extraction project as an example to share projects.

Follow these steps to share your project using Document Intelligence Studio:

1. Start by navigating to the [Document Intelligence Studio](#).
2. In the Studio, select the **Custom extraction models** tile, under the custom models section.



3. On the custom extraction models page, select the desired model to share and then select the **Share** button.

**Template (Custom form) models**

Template models work well when the target documents share a common visual layout. Training only takes a few minutes, and more than 100 languages are supported.

**Request for Taxpayer Identification Number and Certification**

Give Form to the responder. Do not send to the API.

**HOUSE RENTAL AGREEMENT**

This House Rental Agreement ("Agreement," "rental agreement," or "lease") is entered into between **Doug Clegg** (Landlord) and **John Doe** (Guest, Tenant). If more than one person is named as Tenant they shall jointly and severally liable and responsible under the terms of this Agreement. This lease Agreement involves a residential house, yard, and related facilities located at **123 Main Street, Anytown, USA** (the "premises"). The date of this Agreement is **April 1, 2023**.

**My Projects**

+ Create a project  Share  Import  Delete

Project name	Description	Created	API Version
custom-extraction	custom test multipage	2023-04-04T02:58:29.809994Z	2023-02-28 preview
multipage	custom test multipage	2022-09-20T01:22:18.818865Z	2022-08-31
application12	custom test application12	2022-09-20T01:20:13.364718Z	2022-08-31
sec-10q	custom test sec-10q	2022-09-20T01:17:52.619394Z	2022-08-31
Studio Test	Custom Improvements	2022-08-30T05:53:32.884889Z	2022-08-31
Lienfuent	junking words	2022-06-09T18:19:34.882891Z	2022-06-30 preview
test-custom-2	For customer	2022-05-17T18:34:48.653232Z	2022-01-30 preview

#### 4. On the share project dialog, copy the project token for the selected project.

**Template (Custom form) models**

Template models work well when the target documents share a common visual layout. Training only takes a few minutes, and more than 100 languages are supported.

**Request for Taxpayer Identification Number and Certification**

Give Form to the responder. Do not send to the API.

**HOUSE RENTAL AGREEMENT**

This House Rental Agreement ("Agreement," "rental agreement," or "lease") is entered into between **Doug Clegg** (Landlord) and **John Doe** (Guest, Tenant). If more than one person is named as Tenant they shall jointly and severally liable and responsible under the terms of this Agreement. This lease Agreement involves a residential house, yard, and related facilities located at **123 Main Street, Anytown, USA** (the "premises"). The date of this Agreement is **April 1, 2023**.

**Share project**

Copy the below project token to share the selected project.

<project-token>

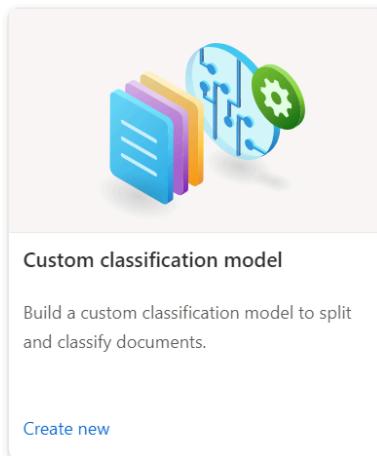
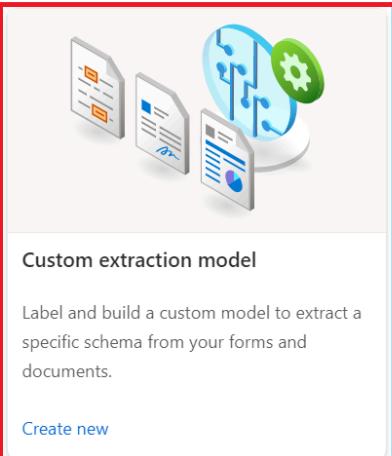
# Import custom extraction model with Document Intelligence Studio

Follow these steps to import a project using Document Intelligence Studio.

1. Start by navigating to the [Document Intelligence Studio](#).
2. In the Studio, select the **Custom extraction models** tile, under the custom models section.

## Custom models

Train custom models to classify documents and extract text, structure and fields from your forms or documents.



### 3. On the custom extraction models page, select the Import button.

Form Recognizer Studio > Custom extraction model

**Custom extraction models**

Extract information from forms and documents with custom extraction models. Train a model by labeling as few as 5 example documents. (The same labeled dataset can train all types of custom extraction models.) [Learn more about custom extraction models](#).

**Template (Custom form) models**

Template models work well when the target documents share a common visual layout. Training only takes a few minutes, and more than 100 languages are supported.

**Request for Taxpayer Identification Number and Certification**

Give Form to the responder. Do not send to the IRS.

**HOUSE RENTAL AGREEMENT**

This House Rental Agreement ("Agreement," "rental agreement," or "lease") is entered into between **JOHN DIAZ** (Landlord) and **SUE WILSON** **1234 FAIRFIELD DR. NEW YORK CITY** (Tenant). If more than one person is named as Tenant they shall be jointly and severally liable and responsible under the terms of this Agreement. This lease Agreement involves a residential house, yard, and related facilities located at **123 FAIRFIELD ST. NEW YORK CITY** (the "premises"). The date of this Agreement is **APRIL 15, 2023**. Tenant shall not assign, sublease, or allow anyone other than persons permitted under this

**Neural (Custom document) models**

Neural models can flexibly handle both structured and unstructured documents. Training takes up to half an hour, and currently only English language documents are supported. The current version can extract inline field data and checklists. Neural models are available only in select regions. [Click here for details](#).

**My Projects**

+ Create a project | Share | **Import** | Delete

Project name Description Created API Version

[Create a new project to get started.](#)

Search

### 4. On the import project dialog, paste the project token shared with you and select import.

Form Recognizer Studio > Custom extraction model

**Custom extraction models**

Extract information from forms and documents with custom extraction models. Train a model by labeling as few as 5 example documents. (The same labeled dataset can train all types of custom extraction models.) [Learn more about custom extraction models](#).

**Template (Custom form) models**

Template models work well when the target documents share a common visual layout. Training only takes a few minutes, and more than 100 languages are supported.

**Request for Taxpayer Identification Number and Certification**

Give Form to the responder. Do not send to the IRS.

**HOUSE RENTAL AGREEMENT**

This House Rental Agreement ("Agreement," "rental agreement," or "lease") is entered into between **JOHN DIAZ** (Landlord) and **SUE WILSON** **1234 FAIRFIELD DR. NEW YORK CITY** (Tenant). If more than one person is named as Tenant they shall be jointly and severally liable and responsible under the terms of this Agreement. This lease Agreement involves a residential house, yard, and related facilities located at **123 FAIRFIELD ST. NEW YORK CITY** (the "premises"). The date of this Agreement is **APRIL 15, 2023**. Tenant shall not assign, sublease, or allow anyone other than persons permitted under this

**Neural (Custom document) models**

Neural models can flexibly handle both structured and unstructured documents. Training takes up to half an hour, and currently only English language documents are supported. The current version can extract inline field data and checklists. Neural models are available only in select regions. [Click here for details](#).

**My Projects**

+ Create a project | Share | **Import** | Delete

Project name Description Created API Version

**Import project**

Enter the project token to continue working on an existing Form Recognizer project.

**<project-token>**

**Import** | Close

[Create a new project to get started.](#)

Search

# Next steps

[Back up and recover models](#)

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Compose custom models

Article • 11/19/2024

emphasis style

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)   
[v2.1 \(GA\)](#)

## Important

Model compose behavior is changed for api-version=2024-11-30 (GA). For more information refer to [composed custom models](#). The following behavior **only** applies to v3.1 and previous versions.

A composed model is created by taking a collection of custom models and assigning them to a single model ID. You can assign up to 200 trained custom models to a single composed model ID. When a document is submitted to a composed model, the service performs a classification step to decide which custom model accurately represents the form presented for analysis. Composed models are useful when you train several models and want to group them to analyze similar form types. For example, your composed model might include custom models trained to analyze your supply, equipment, and furniture purchase orders. Instead of manually trying to select the appropriate model, you can use a composed model to determine the appropriate custom model for each analysis and extraction.

To learn more, see [Composed custom models](#).

In this article, you learn how to create and use composed custom models to analyze your forms and documents.

## Prerequisites

To get started, you need the following resources:

- An Azure subscription. You can [create a free Azure subscription](#).
- A Document Intelligence instance. Once you have your Azure subscription, [create a Document Intelligence resource](#) in the Azure portal to get your key and endpoint. If you have an existing Document Intelligence resource, navigate directly to your resource page. You can use the free pricing tier (F0) to try the service, and upgrade later to a paid tier for production.

1. After the resource deploys, select **Go to resource**.
2. Copy the **Keys and Endpoint** values from the Azure portal and paste them in a convenient location, such as *Microsoft Notepad*. You need the key and endpoint values to connect your application to the Document Intelligence API.

The screenshot shows the Azure portal interface for a resource named "Contoso-DI". The left sidebar has a "Keys and Endpoint" section highlighted with a red box. The main content area shows a "Show Keys" button and two key fields: "KEY 1" and "KEY 2", each with a copy icon. Below the keys is a "Location/Region" field set to "westus2" and an "Endpoint" field containing the URL "https://contoso-di.cognitiveservices.azure.com/". A tip message at the top right says: "These keys are used to access your Azure AI service API. Do not share your keys. Store them securely—for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service."

### Tip

For more information, see [create a Document Intelligence resource](#).

- An **Azure storage account**. If you don't know how to create an Azure storage account, follow the [Azure Storage quickstart for Azure portal](#). You can use the free pricing tier (F0) to try the service, and upgrade later to a paid tier for production.

## Create your custom models

First, you need a set of custom models to compose. You can use the Document Intelligence Studio, REST API, or client libraries. The steps are as follows:

- [Assemble your training dataset](#)
- [Upload your training set to Azure blob storage](#)
- [Train your custom models](#)

# Assemble your training dataset

Building a custom model begins with establishing your training dataset. You need a minimum of five completed forms of the same type for your sample dataset. They can be of different file types (jpg, png, pdf, tiff) and contain both text and handwriting. Your forms must follow the [input requirements](#) for Document Intelligence.

## Tip

Follow these tips to optimize your data set for training:

- If possible, use text-based PDF documents instead of image-based documents. Scanned PDFs are handled as images.
- For filled-in forms, use examples that have all of their fields filled in.
- Use forms with different values in each field.
- If your form images are of lower quality, use a larger data set (10-15 images, for example).

See [Build a training data set](#) for tips on how to collect your training documents.

# Upload your training dataset

Once you gather a set of training documents, you need to [upload your training data](#) to an Azure blob storage container.

If you want to use manually labeled data, you have to upload the *.labels.json* and *.ocr.json* files that correspond to your training documents.

# Train your custom model

When you [train your model](#) with labeled data, the model uses supervised learning to extract values of interest, using the labeled forms you provide. Labeled data results in better-performing models and can produce models that work with complex forms or forms containing values without keys.

Document Intelligence uses the [prebuilt-layout model](#) API to learn the expected sizes and positions of typeface and handwritten text elements and extract tables. Then it uses user-specified labels to learn the key/value associations and tables in the documents. We recommend that you use five manually labeled forms of the same type (same structure) to get started with training a new model. Then, add more labeled data, as

needed, to improve the model accuracy. Document Intelligence enables training a model to extract key-value pairs and tables using supervised learning capabilities.

## Document Intelligence Studio

To create custom models, start with configuring your project:

1. From the Studio homepage, select [Create new](#) from the Custom model card.
2. Use the  **Create a project** command to start the new project configuration wizard.
3. Enter project details, select the Azure subscription and resource, and the Azure Blob storage container that contains your data.
4. Review, submit your settings, and create the project.



The screenshot shows the 'My Projects' section of the Document Intelligence Studio. At the top, there is a header with the title 'My Projects'. Below the header, there are two buttons: '+ Create a project' and 'Delete'. The '+ Create a project' button is highlighted with a red box. To the right of these buttons are two small icons: a gear and a trash can. Below the buttons, there is a table with four columns: 'Project name', 'Description', 'Created ↓', and 'API Version'. The table currently has no data rows.

While creating your custom models, you may need to extract data collections from your documents. The collections may appear one of two formats. Using tables as the visual pattern:

- Dynamic or variable count of values (rows) for a given set of fields (columns)
- Specific collection of values for a given set of fields (columns and/or rows)

See [Document Intelligence Studio: labeling as tables](#)

# Create a composed model

## ⓘ Note

the `create compose model` operation is only available for custom models trained **with labels**. Attempting to compose unlabeled models will produce an error.

With the **create compose model** operation, you can assign up to 100 trained custom models to a single model ID. When analyze documents with a composed model, Document Intelligence first classifies the form you submitted, then chooses the best matching assigned model, and returns results for that model. This operation is useful when incoming forms may belong to one of several templates.

### Document Intelligence Studio

Once the training process is successfully completed, you can begin to build your composed model. Here are the steps for creating and using composed models:

- [Gather your custom model IDs](#)
- [Compose your custom models](#)
- [Analyze documents](#)
- [Manage your composed models](#)

## Gather your model IDs

When you train models using the [Document Intelligence Studio](#), the model ID is located in the models menu under a project:

The screenshot shows the 'Cognitive Services | Form Recognizer Studio - Preview' interface. On the left, there's a sidebar with options: 'Custom Form' (highlighted with a red box), 'composed', 'Label data', 'Models' (highlighted with a grey box), 'Test', and 'Settings'. The main area is titled 'Models' and shows a list of three model IDs: '8aa16866-16fe-44ca-b13a-8bfc6ad1d', '773fb140-f173-47a2-8aa9-a5fce1ceb', and '4c493f98-87c3-4f6d-b0d8-3a1aab49e'. Each row has a checkbox next to the Model ID, followed by 'Model Description' and a 'Delete' button. Above the list, there are buttons for 'Compose' (highlighted with a red box), 'Test', 'Download', and 'Delete'.

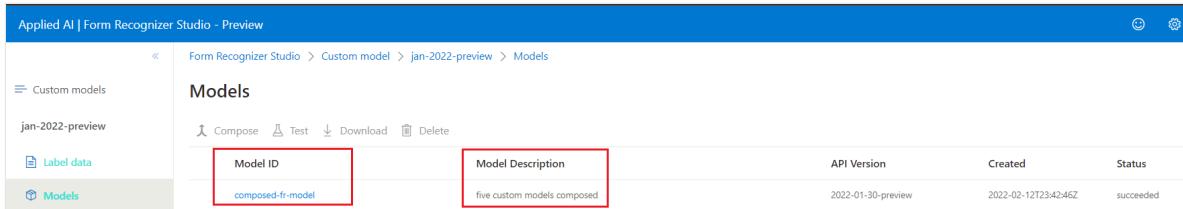
Model ID	Model Description
8aa16866-16fe-44ca-b13a-8bfc6ad1d	
773fb140-f173-47a2-8aa9-a5fce1ceb	
4c493f98-87c3-4f6d-b0d8-3a1aab49e	

## Compose your custom models

1. Select a custom models project.
2. In the project, select the `Models` menu item.
3. From the resulting list of models, select the models you wish to compose.
4. Choose the **Compose** button from the upper-left corner.
5. In the pop-up window, name your newly composed model and select **Compose**.
6. When the operation completes, your newly composed model appears in the list.
7. Once the model is ready, use the **Test** command to validate it with your test documents and observe the results.

## Analyze documents

The custom model **Analyze** operation requires you to provide the `modelID` in the call to Document Intelligence. You should provide the composed model ID for the `modelID` parameter in your applications.



The screenshot shows the 'Models' section of the Form Recognizer Studio. The left sidebar has 'Custom models' expanded, showing 'jan-2022-preview' and 'Models'. The 'Models' tab is selected. The main area shows a table with one row. The table columns are 'Model ID', 'Model Description', 'API Version', 'Created', and 'Status'. The 'Model ID' cell contains 'composed-fr-model' (highlighted with a red box). The 'Model Description' cell contains 'five custom models composed'. The 'API Version' cell contains '2022-01-30-preview'. The 'Created' cell contains '2022-02-12T23:42:46Z'. The 'Status' cell contains 'succeeded'.

Model ID	Model Description	API Version	Created	Status
composed-fr-model	five custom models composed	2022-01-30-preview	2022-02-12T23:42:46Z	succeeded

## Manage your composed models

You can manage your custom models throughout life cycles:

- Test and validate new documents.
- Download your model to use in your applications.
- Delete your model when its lifecycle is complete.

The screenshot shows a table within a software application window titled "composed-fr". The table has four columns: "Component model", "Build Mode", "Field Name", and "Accuracy". There are ten rows of data. The first row is highlighted with a red border around its first three columns. The data is as follows:

Component model	Build Mode	Field Name	Accuracy
fr-3	template	Receipt No	95 %
fr-3	template	Sold To	95 %
fr-3	template	ID #	83.3 %
fr-3	template	Live Delivery?	95 %
fr-3	template	Online Delivery?	95 %
fr-3	template	Video Delivery?	95 %
fr-1	template	Receipt No	95 %
fr-1	template	Sold To	95 %
fr-1	template	ID #	83.3 %
fr-1	template	Live Delivery?	95 %

Great! You learned the steps to create custom and composed models and use them in your Document Intelligence projects and applications.

## Next steps

Try one of our Document Intelligence quickstarts:

[Document Intelligence Studio](#)

[REST API](#)

[C#](#)

[Java](#)

[JavaScript](#)

[Python](#)

## Feedback

Was this page helpful?

 Yes

 No

Provide product feedback  | Get help at Microsoft Q&A

# Disaster recovery

Article • 02/27/2025

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)  v2.1 (GA)

When you create a Document Intelligence resource in the Azure portal, you specify a region. From then on, your resource and all of its operations stay associated with that particular Azure server region. It's rare, but not impossible, to encounter a network issue that hits an entire region. If your solution needs to always be available, then you should design it to either fail-over into another region or split the workload between two or more regions. Both approaches require at least two Document Intelligence resources in different regions and the ability to sync custom models and classifiers across regions.

The Copy API enables this scenario by allowing you to copy custom models and classifiers from one Document Intelligence account or into others, which can exist in any supported geographical region. This guide shows you how to use the Copy REST API with cURL for custom models. You can also use an HTTP request service to issue the requests.

## Note

The 2024-11-30 (GA) API custom classification model supports the Copy API. This guide specifically uses custom models to copy models. For classifier model copy, follow the [train a custom classifier guide](#).

## Business scenarios

If your app or business depends on the use of a Document Intelligence custom model, we recommend you copy your model to another Document Intelligence account in another region. If a regional outage occurs, you can then access your model in the region where it was copied.

## Prerequisites

1. Two Document Intelligence Azure resources in different Azure subscriptions or regions. If you don't have them, go to the Azure portal and [create a new Document Intelligence resource](#).
2. The key, endpoint URL, and subscription ID for your Document Intelligence resource. You can find these values on the resource's [Overview tab](#) in the [Azure portal](#).

# Copy API overview

The process for copying a custom model consists of the following steps:

1. First you issue a copy authorization request to the target resource—that is, the resource that receives the copied model. You receive back the URL of the newly created target model that receives the copied model.
2. Next you send the copy request to the source resource—the resource that contains the model to be copied with the payload (copy authorization) returned from the previous call. You receive back a URL that you can query to track the progress of the operation.
3. You use your source resource credentials to query the progress URL until the operation is a success. You can also query the new model ID in the target resource to get the status of the new model.

## Generate Copy authorization request

The following HTTP request gets copy authorization from your target resource. You need to enter the endpoint and key of your target resource as headers.

HTTP

```
POST https://<your-resource-
endpoint>/documentintelligence/documentModels:authorizeCopy?api-version=2024-11-30
Ocp-Apim-Subscription-Key: {<your-key>}
```

Request body

JSON

```
{
  "modelId": "target-model-name",
  "description": "Copied from SCUS"
}
```

You receive a `200` response code with response body that contains the JSON payload required to initiate the copy.

JSON

```
{
  "targetResourceId":
  "/subscriptions/{targetSub}/resourceGroups/{targetRG}/providers/Microsoft.Cognitiv
  eServices/accounts/{targetService}",
  "targetResourceRegion": "region",
```

```
"targetModelId": "target-model-name",
"targetModelLocation": "model path",
"accessToken": "access token",
"expirationDateTime": "timestamp"
}
```

## Start Copy operation

The following HTTP request starts the copy operation on the source resource. You need to enter the endpoint and key of your source resource as the url and header. Notice that the request URL contains the model ID of the source model you want to copy.

HTTP

```
POST https://<your-resource-
endpoint>/documentintelligence/documentModels/{modelId}:copyTo?api-version=2024-
11-30
Ocp-Apim-Subscription-Key: {<your-key>}
```

The body of your request is the response from the previous step.

JSON

```
{
  "targetResourceId":
    "/subscriptions/{targetSub}/resourceGroups/{targetRG}/providers/Microsoft.Cognitiv
eServices/accounts/{targetService}",
  "targetResourceRegion": "region",
  "targetModelId": "target-model-name",
  "targetModelLocation": "model path",
  "accessToken": "access token",
  "expirationDateTime": "timestamp"
}
```

You receive a `202\Accepted` response with an `Operation-Location` header. This value is the URL that you use to track the progress of the operation. Copy it to a temporary location for the next step.

HTTP

```
HTTP/1.1 202 Accepted
Operation-Location: https://<your-resource-
endpoint>.cognitiveservices.azure.com/documentintelligence/operations/{operation-
id}?api-version=2024-11-30
```

### (!) Note

The Copy API transparently supports the [AEK/CMK](#) feature. This action doesn't require any special treatment, but note that if you're copying between an unencrypted resource to an encrypted resource, you need to include the request header `x-ms-forms-copy-degrade: true`. If this header isn't included, the copy operation fails and returns a `DataProtectionTransformServiceError`.

## Track Copy progress

### Console

```
GET https://<your-resource-
endpoint>.cognitiveservices.azure.com/documentintelligence/operations/{<operation-
id>}?api-version=2024-11-30
Ocp-Apim-Subscription-Key: {<your-key>}
```

## Track the target model ID

You can also use the [Get model](#) API to track the status of the operation by querying the target model. Call the API using the target model ID that you copied down from the [Generate copy authorization request](#) response.

### HTTP

```
GET https://<your-resource-
endpoint>/documentintelligence/documentModels/{modelId}?api-version=2024-11-30" -H
"Ocp-Apim-Subscription-Key: <your-key>
```

In the response body, you see information about the model. Check the `"status"` field for the status of the model.

### HTTP

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
{"modelInfo": {"modelId": "33f4d42c-cd2f-4e74-b990-
a1aeafab5a5d", "status": "ready", "createdDateTime": "2020-02-
26T16:59:28Z", "lastUpdatedDateTime": "2020-02-26T16:59:34Z"}, "trainResult": {
"trainingDocuments": [{"documentName": "0.pdf", "pages": 1, "errors": [],
"status": "succeeded"}, {"documentName": "1.pdf", "pages": 1, "errors": [],
"status": "succeeded"}, {"documentName": "2.pdf", "pages": 1, "errors": [],
"status": "succeeded"}, {"documentName": "3.pdf", "pages": 1, "errors": [],
"status": "succeeded"}]}
```

```
[],"status":"succeeded"}, {"documentName": "4.pdf", "pages": 1, "errors": [], "status": "succeeded"}], "errors": []}]}
```

## cURL sample code

The following code snippets use cURL to make API calls. You also need to fill in the model IDs and subscription information specific to your own resources.

### Generate Copy authorization

#### Request

Bash

```
curl -i -X POST "<your-resource-endpoint>/documentintelligence/documentModels:authorizeCopy?api-version=2024-11-30"
-H "Content-Type: application/json"
-H "Ocp-Apim-Subscription-Key: <YOUR-KEY>"
--data-ascii "{"
    'modelId': '{modelId}',
    'description': '{description}'
}"
```

#### Successful response

JSON

```
{
    "targetResourceId": "string",
    "targetResourceRegion": "string",
    "targetModelId": "string",
    "targetModelLocation": "string",
    "accessToken": "string",
    "expirationDateTime": "string"
}
```

### Begin Copy operation

#### Request

Bash

```
curl -i -X POST "<your-resource-endpoint>/documentintelligence/documentModels/{modelId}:copyTo?api-version=2024-
```

```
11-30"
-H "Content-Type: application/json"
-H "Ocp-Apim-Subscription-Key: <YOUR-KEY>"
--data-ascii "{  
    'targetResourceId': '{targetResourceId}',  
    'targetResourceRegion': {targetResourceRegion},  
    'targetModelId': '{targetModelId}',  
    'targetModelLocation': '{targetModelLocation}',  
    'accessToken': '{accessToken}',  
    'expirationDateTime': '{expirationDateTime}'  
}"
```

## Successful response

HTTP

```
HTTP/1.1 202 Accepted
Operation-Location: https://<your-resource-
endpoint>.cognitiveservices.azure.com/documentintelligence/operations/{operation-
id}?api-version=2024-11-30
```

## Track copy operation progress

You can use the [GET operation API](#) to list all document model operations (succeeded, in-progress, or failed) associated with your Document Intelligence resource. Operation information only persists for 24 hours. Here's a list of the operations (operationId) that can be returned:

- documentModelBuild
- documentModelCompose
- documentModelCopyTo

## Track the target model ID

If the operation was successful, the document model can be accessed using the [getModel](#) (get a single model), or [GetModels](#) (get a list of models) APIs.

## Common error code messages

  Expand table

Error	Resolution
400 / Bad Request with "code: " "1002"	Indicates validation error or badly formed copy request. Common issues include: a) Invalid or modified <code>copyAuthorization</code> payload. b) Expired value for <code>expirationDateTimeTicks</code> token ( <code>copyAuthorization</code> payload is valid for 24 hours). c) Invalid or unsupported <code>targetResourceRegion</code> . d) Invalid or malformed <code>targetResourceId</code> string.
<i>Authorization failure due to missing or invalid authorization claims.</i>	Occurs when the <code>copyAuthorization</code> payload or content is modified from the <code>copyAuthorization</code> API. Ensure that the payload is the same exact content that was returned from the earlier <code>copyAuthorization</code> call.
<i>Couldn't retrieve authorization metadata.</i>	Indicates that the <code>copyAuthorization</code> payload is being reused with a copy request. A copy request that succeeds doesn't allow any further requests that use the same <code>copyAuthorization</code> payload. If you raise a separate error and you later retry the copy with the same authorization payload, this error gets raised. The resolution is to generate a new <code>copyAuthorization</code> payload and then reissue the copy request.
<i>Data transfer request isn't allowed as it downgrades to a less secure data protection scheme.</i>	Occurs when copying between an <code>AEK</code> enabled resource to a non- <code>AEK</code> enabled resource. To allow copying encrypted model to the target as unencrypted, specify <code>x-ms-forms-copy-degrade: true</code> header with the copy request.
"Couldn't fetch information for Cognitive resource with ID...".	Indicates that the Azure resource indicated by the <code>targetResourceId</code> isn't a valid Cognitive resource or doesn't exist. To resolve this issue, verify and reissue the copy request.  Ensure the resource is valid and exists in the specified region, such as, <code>westus2</code>

## Next steps

In this guide, you learned how to use the Copy API to back up your custom models to a secondary Document Intelligence resource. Next, explore the API reference docs to see what else you can do with Document Intelligence.

- [REST API reference documentation](#)

# Configure Azure AI services virtual networks

Article • 01/31/2025

Azure AI services provide a layered security model. This model enables you to secure your Azure AI services accounts to a specific subset of networks. When network rules are configured, only applications that request data over the specified set of networks can access the account. You can limit access to your resources with *request filtering*, which allows requests that originate only from specified IP addresses, IP ranges, or from a list of subnets in [Azure Virtual Networks](#).

An application that accesses an Azure AI services resource when network rules are in effect requires authorization. Authorization is supported with [Microsoft Entra ID](#) credentials or with a valid API key.

## Important

Turning on firewall rules for your Azure AI services account blocks incoming requests for data by default. To allow requests through, one of the following conditions needs to be met:

- The request originates from a service that operates within an Azure Virtual Network on the allowed subnet list of the target Azure AI services account. The endpoint request that originated from the virtual network needs to be set as the [custom subdomain](#) of your Azure AI services account.
- The request originates from an allowed list of IP addresses.

Requests that are blocked include those from other Azure services, from the Azure portal, and from logging and metrics services.

## Note

We recommend that you use the Azure Az PowerShell module to interact with Azure. To get started, see [Install Azure PowerShell](#). To learn how to migrate to the Az PowerShell module, see [Migrate Azure PowerShell from AzureRM to Az](#).

## Scenarios

To secure your Azure AI services resource, you should first configure a rule to deny access to traffic from all networks, including internet traffic, by default. Then, configure rules that grant access to traffic from specific virtual networks. This configuration enables you to build a secure network boundary for your applications. You can also configure rules to grant access to traffic from select public internet IP address ranges and enable connections from specific internet or on-premises clients.

Network rules are enforced on all network protocols to Azure AI services, including REST and WebSocket. To access data by using tools such as the Azure test consoles, explicit network rules must be configured. You can apply network rules to existing Azure AI services resources, or when you create new Azure AI services resources. After network rules are applied, they're enforced for all requests.

## Supported regions and service offerings

Virtual networks are supported in [regions where Azure AI services are available](#). Azure AI services support service tags for network rules configuration. The services listed here are included in the `CognitiveServicesManagement` service tag.

- ✓ Anomaly Detector
- ✓ Azure OpenAI
- ✓ Content Moderator
- ✓ Custom Vision
- ✓ Face
- ✓ Language Understanding (LUIS)
- ✓ Personalizer
- ✓ Speech service
- ✓ Language
- ✓ QnA Maker
- ✓ Translator

### Note

If you use Azure OpenAI, LUIS, Speech Services, or Language services, the `CognitiveServicesManagement` tag only enables you to use the service by using the SDK or REST API. To access and use the [Azure AI Foundry portal](#), LUIS portal, Speech Studio, or Language Studio from a virtual network, you need to use the following tags:

- `AzureActiveDirectory`
- `AzureFrontDoor.Frontend`

- AzureResourceManager
- CognitiveServicesManagement
- CognitiveServicesFrontEnd
- Storage (Speech Studio only)

For information on [Azure AI Foundry portal](#) configurations, see the [Azure AI Foundry documentation](#).

## Change the default network access rule

By default, Azure AI services resources accept connections from clients on any network. To limit access to selected networks, you must first change the default action.

### Warning

Making changes to network rules can impact your applications' ability to connect to Azure AI services. Setting the default network rule to *deny* blocks all access to the data unless specific network rules that *grant* access are also applied.

Before you change the default rule to deny access, be sure to grant access to any allowed networks by using network rules. If you allow listing for the IP addresses for your on-premises network, be sure to add all possible outgoing public IP addresses from your on-premises network.

## Manage default network access rules

You can manage default network access rules for Azure AI services resources through the Azure portal, PowerShell, or the Azure CLI.

Azure portal

1. Go to the Azure AI services resource you want to secure.
2. Select **Resource Management** to expand it, then select **Networking**.

Home > contoso-rg > contoso-custom-vision

**contoso-custom-vision** | Networking

Custom vision | Directory: Microsoft

Firewalls and virtual networks Private endpoint connections

Save Discard Refresh

Access control settings allowing access to Azure AI services account will remain in effect for up to three minutes after saving updated settings restricting access.

Allow access from

All networks  Selected Networks and Private Endpoints  Disabled

Configure network security for your Azure AI services account. [Learn more.](#)

Virtual networks

Secure your Azure AI services account with virtual networks. + Add existing virtual network + Add new virtual network

Virtual Network	Subnet	Address range	Endpoint Status	Resource group	Subscription
No network selected.					

Firewall

Add IP ranges to allow access from the internet or your on-premises networks. [Learn more.](#)

Add your client IP address

Address range

IP address or CIDR

3. To deny access by default, under **Firewalls and virtual networks**, select **Selected Networks and Private Endpoints**.

With this setting alone, unaccompanied by configured virtual networks or address ranges, all access is effectively denied. When all access is denied, requests that attempt to consume the Azure AI services resource aren't permitted. The Azure portal, Azure PowerShell, or the Azure CLI can still be used to configure the Azure AI services resource.

4. To allow traffic from all networks, select **All networks**.

Home > contoso-rg > contoso-custom-vision

**contoso-custom-vision** | Networking

Custom vision | Directory: Microsoft

Firewalls and virtual networks Private endpoint connections

Save Discard Refresh

Allow access from

All networks  Selected Networks and Private Endpoints  Disabled

All networks, including the internet, can access this resource. [Learn more.](#)

Keys and Endpoint

Encryption

Pricing tier

Networking

Identity

5. Select **Save** to apply your changes.

## Grant access from a virtual network

You can configure Azure AI services resources to allow access from specific subnets only. The allowed subnets might belong to a virtual network in the same subscription or in a different subscription. The other subscription can belong to a different Microsoft Entra tenant. When the subnet belongs to a different subscription, the Microsoft.CognitiveServices resource provider needs to be also registered for that subscription.

Enable a *service endpoint* for Azure AI services within the virtual network. The service endpoint routes traffic from the virtual network through an optimal path to the Azure AI service. For more information, see [Virtual Network service endpoints](#).

The identities of the subnet and the virtual network are also transmitted with each request. Administrators can then configure network rules for the Azure AI services resource to allow requests from specific subnets in a virtual network. Clients granted access by these network rules must continue to meet the authorization requirements of the Azure AI services resource to access the data.

Each Azure AI services resource supports up to 100 virtual network rules, which can be combined with IP network rules. For more information, see [Grant access from an internet IP range](#) later in this article.

## Set required permissions

To apply a virtual network rule to an Azure AI services resource, you need the appropriate permissions for the subnets to add. The required permission is the default *Contributor* role or the *Cognitive Services Contributor* role. Required permissions can also be added to custom role definitions.

The Azure AI services resource and the virtual networks that are granted access might be in different subscriptions, including subscriptions that are part of a different Microsoft Entra tenant.

### Note

Configuration of rules that grant access to subnets in virtual networks that are a part of a different Microsoft Entra tenant are currently supported only through PowerShell, the Azure CLI, and the REST APIs. You can view these rules in the Azure portal, but you can't configure them.

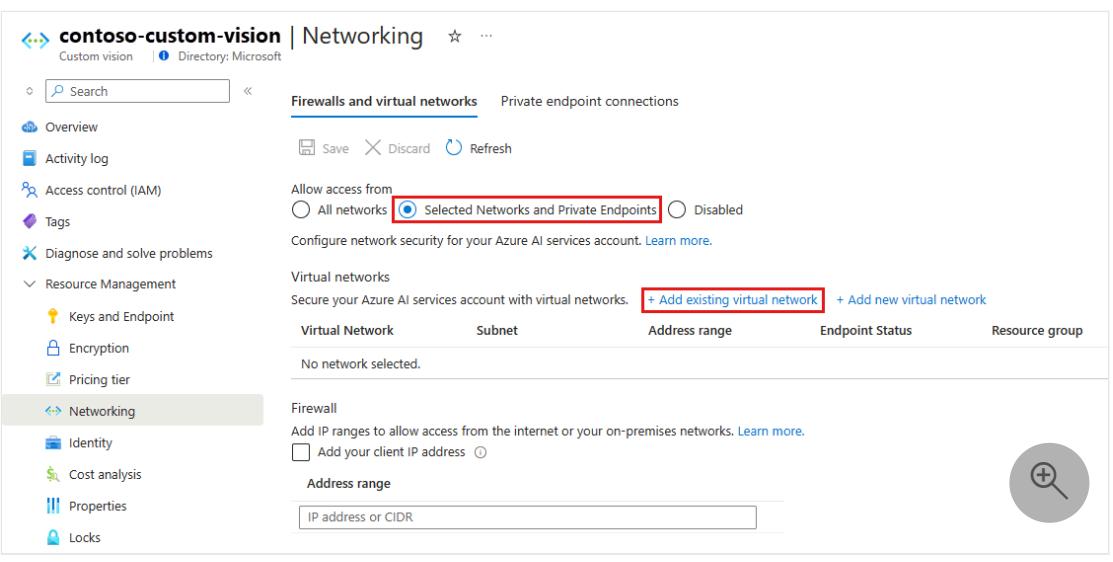
## Configure virtual network rules

You can manage virtual network rules for Azure AI services resources through the Azure portal, PowerShell, or the Azure CLI.

Azure portal

To grant access to a virtual network with an existing network rule:

1. Go to the Azure AI services resource you want to secure.
2. Select **Resource Management** to expand it, then select **Networking**.
3. Confirm that you selected **Selected Networks and Private Endpoints**.
4. Under **Allow access from**, select **Add existing virtual network**.



5. Select the **Virtual networks** and **Subnets** options, and then select **Enable**.

## Add networks

X

Subscription \*

Contoso Subscription

Virtual networks \*

contoso-rg

Subnets \*

default (Service endpoint required)

**i** The following networks don't have service endpoints enabled for 'Microsoft.CognitiveServices'. Enabling access will take up to 15 minutes to complete. After starting this operation, it is safe to leave and return later if you do not wish to wait.

Virtual network	Service endpoint status	
contoso-rg	Not enabled	...
default	Not enabled	...

Enable

### ⚠ Note

If a service endpoint for Azure AI services wasn't previously configured for the selected virtual network and subnets, you can configure it as part of this operation.

Currently, only virtual networks that belong to the same Microsoft Entra tenant are available for selection during rule creation. To grant access to a subnet in a virtual network that belongs to another tenant, use PowerShell, the Azure CLI, or the REST APIs.

6. Select **Save** to apply your changes.

To create a new virtual network and grant it access:

1. On the same page as the previous procedure, select **Add new virtual network**.

The screenshot shows the Azure portal interface for managing a Custom Vision service named "contoso-custom-vision". The left sidebar lists various management options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management (Keys and Endpoint, Encryption, Pricing tier), Networking (Identity, Cost analysis, Properties, Locks), and Firewall. The "Networking" section is currently selected. The main content area displays settings for firewalls and virtual networks. Under "Allow access from", the radio button for "Selected Networks and Private Endpoints" is selected. Below this, there's a note about securing the Azure AI services account with virtual networks and buttons for "+ Add existing virtual network" and "+ Add new virtual network", both of which are highlighted with a red box. A table below shows columns for Virtual Network, Subnet, Address range, Endpoint Status, and Resource group, with a message stating "No network selected.". The bottom right of the blade features a circular icon with a plus sign and a magnifying glass.

2. Provide the information necessary to create the new virtual network, and then select **Create**.

**Create virtual network**

\* Name  
widgets-vnet

\* Address space ⓘ  
10.1.0.0/16  
10.1.0.0 - 10.1.255.255 (65536 addresses)

\* Subscription  
widgets-subscription

\* Resource group  
widgets-resource-group  
[Create new](#)

\* Location  
(US) West US 2

Subnet

\* Name  
default

\* Address range ⓘ  
10.1.0.0/24  
10.1.0.0 - 10.1.0.255 (256 addresses)

DDoS protection ⓘ  
 Basic  Standard

Service endpoint ⓘ  
Microsoft.CognitiveServices

Firewall ⓘ  
 Disabled  Enabled

**Create**

3. Select **Save** to apply your changes.

To remove a virtual network or subnet rule:

1. On the same page as the previous procedures, select ...(**More options**) to open the context menu for the virtual network or subnet, and select **Remove**.

The screenshot shows the 'Firewalls and virtual networks' section of the Azure portal. At the top, there are buttons for 'Save', 'Discard', and 'Refresh'. Below that, there's a section for 'Allow access from' with three options: 'All networks' (radio button), 'Selected Networks and Private Endpoints' (selected radio button), and 'Disabled'. A note says 'Configure network security for your Azure AI services account. [Learn more](#)'. Under 'Virtual networks', there's a table with columns 'Virtual Network', 'Subnet', 'Address range', 'Endpoint Status', 'Resource group', and 'Subscription'. One row is shown for 'contoso-01-vnet' with subnet '1'. The 'Subscription' column shows 'contoso-rg'. To the right of the table are two buttons: 'Remove' (highlighted with a red box) and '...'. Below the table is a 'Firewall' section with a note about adding IP ranges and a checkbox for 'Add your client IP address'. There's also a 'Address range' section with a search icon.

2. Select **Save** to apply your changes.

### ⓘ Important

Be sure to [set the default rule](#) to *deny*, or network rules have no effect.

## Grant access from an internet IP range

You can configure Azure AI services resources to allow access from specific public internet IP address ranges. This configuration grants access to specific services and on-premises networks, which effectively block general internet traffic.

You can specify the allowed internet address ranges by using [CIDR format \(RFC 4632\)](#) in the form `192.168.0.0/16` or as individual IP addresses like `192.168.0.1`.

### 💡 Tip

Small address ranges that use `/31` or `/32` prefix sizes aren't supported. Configure these ranges by using individual IP address rules.

IP network rules are only allowed for *public internet* IP addresses. IP address ranges reserved for private networks aren't allowed in IP rules. Private networks include addresses that start with `10.*`, `172.16.*` - `172.31.*`, and `192.168.*`. For more information, see [Private Address Space \(RFC 1918\)](#).

Currently, only IPv4 addresses are supported. Each Azure AI services resource supports up to 100 IP network rules, which can be combined with [virtual network rules](#).

# Configure access from on-premises networks

To grant access from your on-premises networks to your Azure AI services resource with an IP network rule, identify the internet-facing IP addresses used by your network. Contact your network administrator for help.

If you use Azure ExpressRoute on-premises for Microsoft peering, you need to identify the NAT IP addresses. For more information, see [What is Azure ExpressRoute](#).

For Microsoft peering, the NAT IP addresses that are used are either customer provided or supplied by the service provider. To allow access to your service resources, you must allow these public IP addresses in the resource IP firewall setting.

## Managing IP network rules

You can manage IP network rules for Azure AI services resources through the Azure portal, PowerShell, or the Azure CLI.

Azure portal

1. Go to the Azure AI services resource you want to secure.
2. Select **Resource Management** to expand it, then select **Networking**.
3. Confirm that you selected **Selected Networks and Private Endpoints**.
4. Under **Firewalls and virtual networks**, locate the **Address range** option. To grant access to an internet IP range, enter the IP address or address range (in [CIDR format](#)). Only valid public IP (nonreserved) addresses are accepted.

Firewalls and virtual networks    Private endpoint connections

Save   Discard   Refresh

Allow access from

All networks  Selected Networks and Private Endpoints  Disabled

Configure network security for your Azure AI services account. [Learn more](#).

Virtual networks

Secure your Azure AI services account with virtual networks. [+ Add existing virtual network](#) [+ Add new virtual network](#)

Virtual Network	Subnet	Address range	Endpoint Status	Resource group
No network selected.				

Firewall

Add IP ranges to allow access from the internet or your on-premises networks. [Learn more](#).

Add your client IP address (1)

Address range

(1)

To remove an IP network rule, select the trash can  icon next to the address range.

5. Select **Save** to apply your changes.

### Important

Be sure to [set the default rule](#) to *deny*, or network rules have no effect.

## Use private endpoints

You can use [private endpoints](#) for your Azure AI services resources to allow clients on a virtual network to securely access data over [Azure Private Link](#). The private endpoint uses an IP address from the virtual network address space for your Azure AI services resource. Network traffic between the clients on the virtual network and the resource traverses the virtual network and a private link on the Microsoft Azure backbone network, which eliminates exposure from the public internet.

Private endpoints for Azure AI services resources let you:

- Secure your Azure AI services resource by configuring the firewall to block all connections on the public endpoint for the Azure AI service.
- Increase security for the virtual network, by enabling you to block exfiltration of data from the virtual network.

- Securely connect to Azure AI services resources from on-premises networks that connect to the virtual network by using [Azure VPN Gateway](#) or [ExpressRoutes](#) with private-peering.

## Understand private endpoints

A private endpoint is a special network interface for an Azure resource in your [virtual network](#). Creating a private endpoint for your Azure AI services resource provides secure connectivity between clients in your virtual network and your resource. The private endpoint is assigned an IP address from the IP address range of your virtual network. The connection between the private endpoint and the Azure AI service uses a secure private link.

Applications in the virtual network can connect to the service over the private endpoint seamlessly. Connections use the same connection strings and authorization mechanisms that they would use otherwise. The exception is Speech Services, which require a separate endpoint. For more information, see [Private endpoints with the Speech Services](#) in this article. Private endpoints can be used with all protocols supported by the Azure AI services resource, including REST.

Private endpoints can be created in subnets that use service endpoints. Clients in a subnet can connect to one Azure AI services resource using private endpoint, while using service endpoints to access others. For more information, see [Virtual Network service endpoints](#).

When you create a private endpoint for an Azure AI services resource in your virtual network, Azure sends a consent request for approval to the Azure AI services resource owner. If the user who requests the creation of the private endpoint is also an owner of the resource, this consent request is automatically approved.

Azure AI services resource owners can manage consent requests and the private endpoints through the **Private endpoint connection** tab for the Azure AI services resource in the [Azure portal](#) ↗.

## Specify private endpoints

When you create a private endpoint, specify the Azure AI services resource that it connects to. For more information on creating a private endpoint, see:

- [Create a private endpoint by using the Azure portal](#)
- [Create a private endpoint by using Azure PowerShell](#)
- [Create a private endpoint by using the Azure CLI](#)

# Connect to private endpoints

## ⓘ Note

Azure OpenAI Service uses a different private DNS zone and public DNS zone forwarder than other Azure AI services. For the correct zone and forwarder names, see [Azure services DNS zone configuration](#).

Clients on a virtual network that use the private endpoint use the same connection string for the Azure AI services resource as clients connecting to the public endpoint. The exception is the Speech service, which requires a separate endpoint. For more information, see [Use private endpoints with the Speech service](#) in this article. DNS resolution automatically routes the connections from the virtual network to the Azure AI services resource over a private link.

By default, Azure creates a [private DNS zone](#) attached to the virtual network with the necessary updates for the private endpoints. If you use your own DNS server, you might need to make more changes to your DNS configuration. For updates that might be required for private endpoints, see [Apply DNS changes for private endpoints](#) in this article.

## Use private endpoints with the Speech service

See [Use Speech service through a private endpoint](#).

## Apply DNS changes for private endpoints

When you create a private endpoint, the DNS `CNAME` resource record for the Azure AI services resource is updated to an alias in a subdomain with the prefix `privatelink`. By default, Azure also creates a private DNS zone that corresponds to the `privatelink` subdomain, with the DNS A resource records for the private endpoints. For more information, see [What is Azure Private DNS](#).

When you resolve the endpoint URL from outside the virtual network with the private endpoint, it resolves to the public endpoint of the Azure AI services resource. When it's resolved from the virtual network hosting the private endpoint, the endpoint URL resolves to the private endpoint's IP address.

This approach enables access to the Azure AI services resource using the same connection string for clients in the virtual network that hosts the private endpoints and clients outside the virtual network.

If you use a custom DNS server on your network, clients must be able to resolve the fully qualified domain name (FQDN) for the Azure AI services resource endpoint to the private endpoint IP address. Configure your DNS server to delegate your private link subdomain to the private DNS zone for the virtual network.

### 💡 Tip

When you use a custom or on-premises DNS server, you should configure your DNS server to resolve the Azure AI services resource name in the `privatelink` subdomain to the private endpoint IP address. Delegate the `privatelink` subdomain to the private DNS zone of the virtual network. Alternatively, configure the DNS zone on your DNS server and add the DNS A records.

For more information on configuring your own DNS server to support private endpoints, see the following resources:

- [Name resolution that uses your own DNS server](#)
- [DNS configuration](#)

## Grant access to trusted Azure services for Azure OpenAI

You can grant a subset of trusted Azure services access to Azure OpenAI, while maintaining network rules for other apps. These trusted services will then use managed identity to authenticate your Azure OpenAI service. The following table lists the services that can access Azure OpenAI if the managed identity of those services have the appropriate role assignment.

 [Expand table](#)

Service	Resource provider name
Azure AI Services	<code>Microsoft.CognitiveServices</code>
Azure Machine Learning	<code>Microsoft.MachineLearningServices</code>
Azure AI Search	<code>Microsoft.Search</code>

You can grant networking access to trusted Azure services by creating a network rule exception using the REST API or Azure portal:

# Using the Azure CLI

Bash

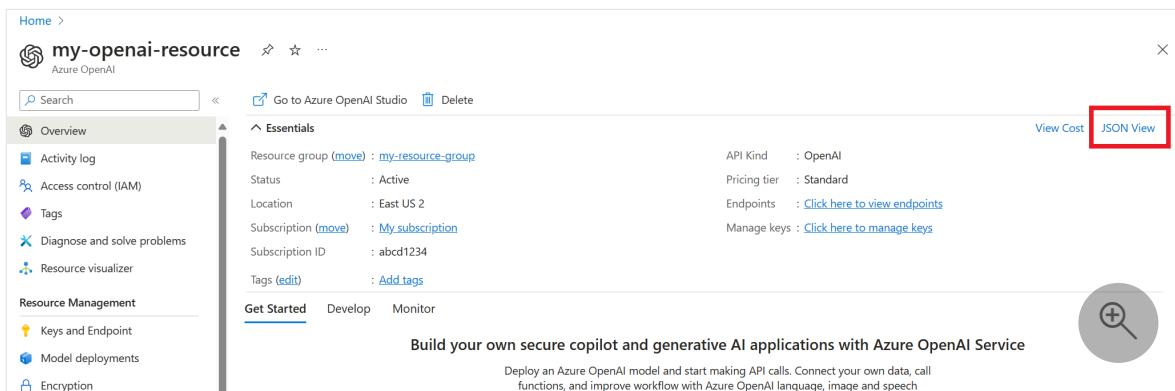
```
accessToken=$(az account get-access-token --resource https://management.azure.com --query "accessToken" --output tsv)
rid="/subscriptions/<your subscription id>/resourceGroups/<your resource group>/providers/Microsoft.CognitiveServices/accounts/<your Azure AI resource name>"

curl -i -X PATCH https://management.azure.com$rid?api-version=2023-10-01-preview \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $accessToken" \
-d \
'
{
    "properties": {
        "networkAcls": {
            "bypass": "AzureServices"
        }
    }
}'
```

To revoke the exception, set `networkAcls.bypass` to `None`.

To verify if the trusted service has been enabled from the Azure portal,

## 1. Use the JSON View from the Azure OpenAI resource overview page



The screenshot shows the Azure OpenAI resource overview page for a resource named "my-openai-resource". The left sidebar includes links for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource visualizer, Resource Management (Keys and Endpoint, Model deployments, Encryption), Get Started, Develop, and Monitor. The main content area displays the "Essentials" section with details like Resource group (my-resource-group), Status (Active), Location (East US 2), Subscription (My subscription), Subscription ID (abcd1234), and Tags (Add tags). To the right of the essentials, there are fields for API Kind (OpenAI), Pricing tier (Standard), Endpoints (Click here to view endpoints), and Manage keys (Click here to manage keys). At the top right, there are "View Cost" and "JSON View" buttons, with "JSON View" being highlighted by a red box. Below the essentials, there is a call-to-action: "Build your own secure copilot and generative AI applications with Azure OpenAI Service" and a note: "Deploy an Azure OpenAI model and start making API calls. Connect your own data, call functions, and improve workflow with Azure OpenAI language, image and speech".

## 2. Choose your latest API version under API versions. Only the latest API version is supported, 2023-10-01-preview.

## Resource JSON

Resource ID

/subscriptions/ /resourceGroups/ /providers/Microsoft [ ] 2023-10-01-preview [ ]

```
75 "networkAcls": {  
76     "bypass": "AzureServices",  
77     "defaultAction": "Deny",  
78     "virtualNetworkRules": [],  
79     "ipRules": []  
80 },
```

[ ]

## Using the Azure portal

1. Navigate to your Azure OpenAI resource, and select **Networking** from the navigation menu.
2. Under **Exceptions**, select **Allow Azure services on the trusted services list** to access this cognitive services account.



You can view the **Exceptions** option by selecting either **Selected networks and private endpoints** or **Disabled** under **Allow access from**.

Dashboard > **test** | Networking

Overview Activity log Access control (IAM) Tags Diagnose and solve problems Resource visualizer Favorites **Networking** Identity Resource Management Keys and Endpoint Model deployments Encryption Pricing tier Networking Identity Cost analysis Properties Locks

Firewalls and virtual networks Private endpoint connections

Save Discard Refresh

Access control settings allowing access to Azure AI services account will remain in effect for up to three minutes after saving.

Allow access from

All networks  Selected Networks and Private Endpoints  Disabled

Configure network security for your Azure AI services account. Learn more.

Virtual networks

Secure your Azure AI services account with virtual networks. + Add existing virtual network + Add new virtual network

Virtual Network	Subnet	Address range	Enc
No network selected.			

Firewall

Add IP ranges to allow access from the internet or your on-premises networks. Learn more.

Add your client IP address ( ) ⓘ

Address range

IP address or CIDR

Exceptions

Allow Azure services on the trusted services list to access this cognitive services account. ⓘ

# Pricing

For pricing details, see [Azure Private Link pricing](#).

## Next steps

- Explore the various [Azure AI services](#)
  - Learn more about [Virtual Network service endpoints](#)
- 

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Managed identities for Document Intelligence

Article • 11/19/2024

This content applies to: ✓ v4.0 (GA) ✓ v3.1 (GA) ✓ v3.0 (GA) ✓ v2.1 (GA)

Managed identities for Azure resources are service principals that create a Microsoft Entra identity and specific permissions for Azure managed resources:



- Managed identities grant access to any resource that supports Microsoft Entra authentication, including your own applications. Unlike security keys and authentication tokens, managed identities eliminate the need for developers to manage credentials.
- You can grant access to an Azure resource and assign an Azure role to a managed identity using [Azure role-based access control \(Azure RBAC\)](#). There's no added cost to use managed identities in Azure.

## ⓘ Important

- Managed identities eliminate the need for you to manage credentials, including Shared Access Signature (SAS) tokens.
- Managed identities are a safer way to grant access to data without having credentials in your code.

## Private storage account access

Private Azure storage account access and authentication support [managed identities for Azure resources](#). If you have an Azure storage account, protected by a Virtual Network (vNet) or firewall, Document Intelligence can't directly access your storage account data. However, once a managed identity is enabled, Document Intelligence can access your storage account using an assigned managed identity credential.

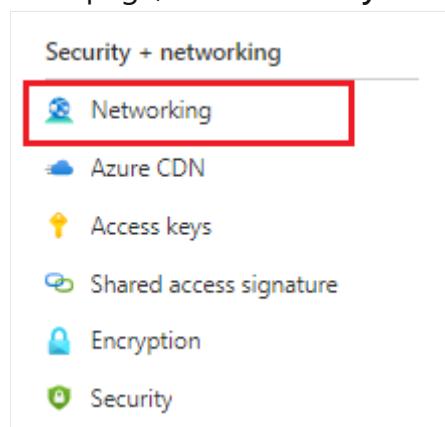
## ⚠ Note

- If you intend to analyze your storage data with the [Document Intelligence Sample Labeling tool \(FOTT\)](#), you must deploy the tool behind your VNet or firewall.
- The [Analyze Receipt, Business Card, Invoice, ID document, and Custom Form](#) APIs can extract data from a single document by posting requests as raw binary content. In these scenarios, there is no requirement for a managed identity credential.

## Prerequisites

To get started, you need:

- An active [Azure account](#)—if you don't have one, you can [create a free account](#).
- A [Document Intelligence](#) or [Azure AI services](#) resource in the Azure portal. For detailed steps, see [Create an Azure AI services resource](#).
- An [Azure blob storage account](#) in the same region as your Document Intelligence resource. You also need to create containers to store and organize your blob data within your storage account.
  - If your storage account is behind a firewall, **you must enable the following configuration:**
  - On your storage account page, select **Security + networking** → **Networking**



from the left menu.

- In the main window, select **Allow access from selected networks**.

The screenshot shows the 'Firewalls and virtual networks' section of the Azure Storage settings. At the top, there are tabs for 'Firewalls and virtual networks', 'Private endpoint connections', and 'Custom domain'. Below the tabs are buttons for 'Save', 'Discard', and 'Refresh'. A note says: 'Firewall settings allowing access to storage services will remain in effect for up to a minute after saving updated settings restricting access.' Under 'Allow access from', there are two options: 'All networks' (unchecked) and 'Selected networks' (checked). A red circle highlights the 'Selected networks' option. At the bottom, there's a note: 'Configure network security for your storage accounts. Learn more'.

- On the selected networks page, navigate to the **Exceptions** category and make certain that the **Allow Azure services on the trusted services list to access this storage account** checkbox is enabled.

The screenshot shows the 'Exceptions' section. It contains three checkboxes:
 

- Allow Azure services on the trusted services list to access this storage account. (This checkbox is highlighted with a red border.)
- Allow read access to storage logging from any network
- Allow read access to storage metrics from any network

- A brief understanding of **Azure role-based access control (Azure RBAC)** using the Azure portal.

## Managed identity assignments

There are two types of managed identity: **system-assigned** and **user-assigned**.

Currently, Document Intelligence only supports system-assigned managed identity:

- A system-assigned managed identity is **enabled** directly on a service instance. It isn't enabled by default; you must go to your resource and update the identity setting.
- The system-assigned managed identity is tied to your resource throughout its lifecycle. If you delete your resource, the managed identity is deleted as well.

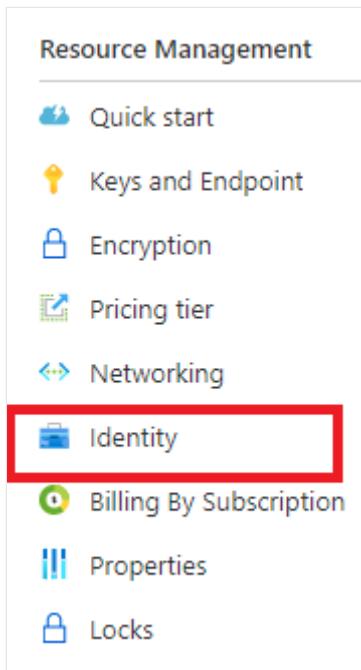
In the following steps, we enable a system-assigned managed identity and grant Document Intelligence limited access to your Azure blob storage account.

## Enable a system-assigned managed identity

**Important**

To enable a system-assigned managed identity, you need Microsoft.Authorization/roleAssignments/write permissions, such as [Owner](#) or [User Access Administrator](#). You can specify a scope at four levels: management group, subscription, resource group, or resource.

1. Sign in to the [Azure portal](#) using an account associated with your Azure subscription.
2. Navigate to your **Document Intelligence** resource page in the Azure portal.
3. In the left rail, Select **Identity** from the Resource Management list:



4. In the main window, toggle the **System assigned Status** tab to **On**.

## Grant access to your storage account

You need to grant Document Intelligence access to your storage account before it can read blobs. Now that Document Intelligence access is enabled with a system-assigned managed identity, you can use Azure role-based access control (Azure RBAC), to give Document Intelligence access to Azure storage. The **Storage Blob Data Reader** role gives Document Intelligence (represented by the system-assigned managed identity) read and list access to the blob container and data.

1. Under **Permissions** select **Azure role assignments**:

[System assigned](#)   [User assigned](#)

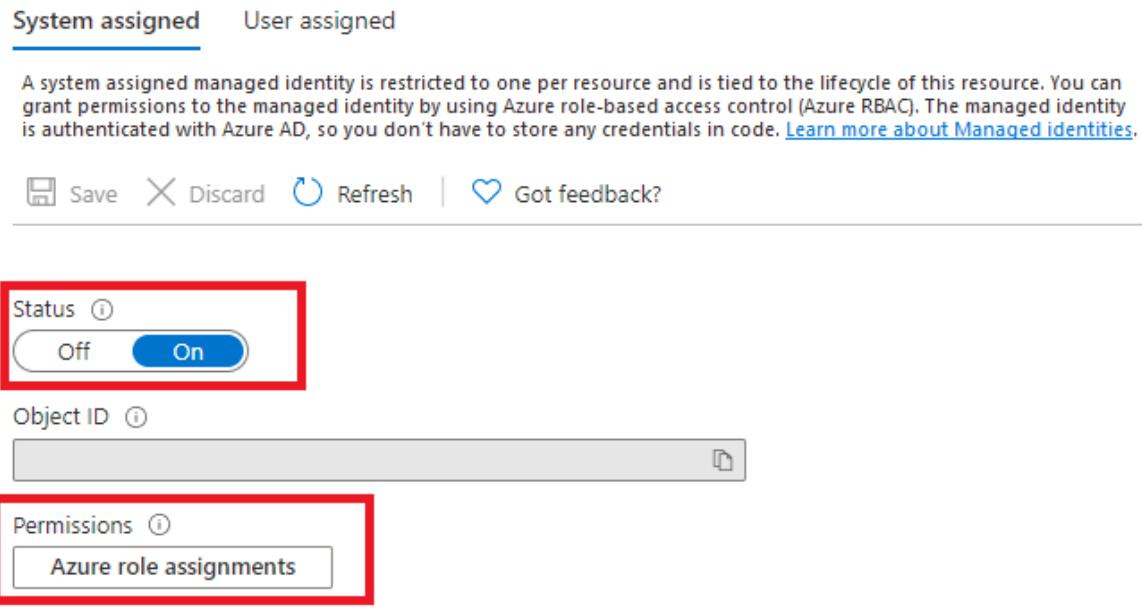
A system assigned managed identity is restricted to one per resource and is tied to the lifecycle of this resource. You can grant permissions to the managed identity by using Azure role-based access control (Azure RBAC). The managed identity is authenticated with Azure AD, so you don't have to store any credentials in code. [Learn more about Managed identities.](#)

Save Discard Refresh Got feedback?

Status: [Off](#) [On](#)

Object ID:

Permissions: [Azure role assignments](#)



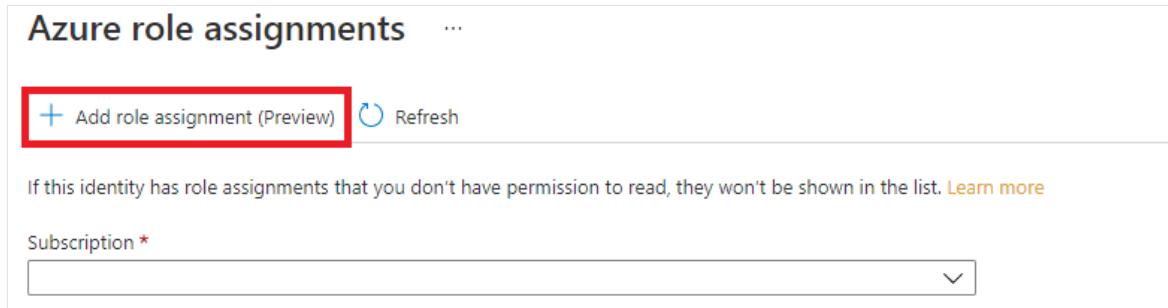
2. On the Azure role assignments page that opens, choose your subscription from the drop-down menu then select **+ Add role assignment**.

### Azure role assignments

+ Add role assignment (Preview) Refresh

If this identity has role assignments that you don't have permission to read, they won't be shown in the list. [Learn more](#)

Subscription \*



#### ⓘ Note

If you're unable to assign a role in the Azure portal because the Add > Add role assignment option is disabled or you get the permissions error, "you do not have permissions to add role assignment at this scope", check that you're currently signed in as a user with an assigned a role that has Microsoft.Authorization/roleAssignments/write permissions such as Owner or User Access Administrator at the Storage scope for the storage resource.

3. Next, you're going to assign a **Storage Blob Data Reader** role to your Document Intelligence service resource. In the **Add role assignment** pop-up window, complete the fields as follows and select **Save**:

[+] Expand table

Field	Value
Scope	<i>Storage</i>

Field	Value
Subscription	<i>The subscription associated with your storage resource.</i>
Resource	<i>The name of your storage resource</i>
Role	<b>Storage Blob Data Reader</b> —allows for read access to Azure Storage blob containers and data.

Add role assignment (Preview) X

Scope (i)  
Storage

Subscription  
▼

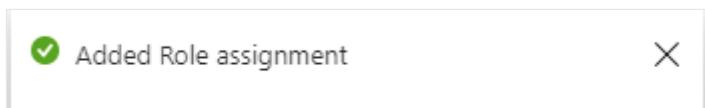
Resource (i)  
▼

Role (i)  
Storage Blob Data Reader ▼

[Learn more about RBAC](#)

Save Discard

4. After you receive the *Added Role assignment* confirmation message, refresh the page to see the added role assignment.



5. If you don't see the change right away, wait and try refreshing the page once more. When you assign or remove role assignments, it can take up to 30 minutes for changes to take effect.

Azure role assignments ...

+ Add role assignment (Preview) ↻ Refresh

If this identity has role assignments that you don't have permission to read, they won't be shown in the list. [Learn more](#)

Subscription \*  
▼

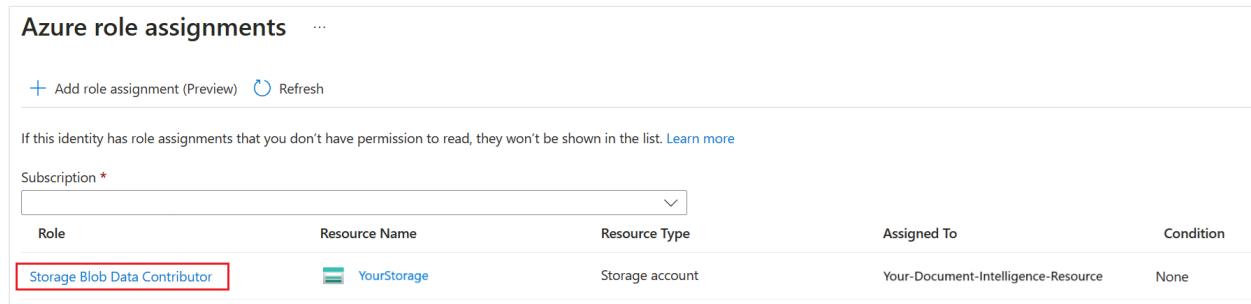
Role	Resource Name	Resource Type	Assigned To
Storage Blob Data Reader	YourStorage	Storage account	Your-Form-Recognizer-Service

That's it! You completed the steps to enable a system-assigned managed identity. With managed identity and Azure RBAC, you granted Document Intelligence specific access

rights to your storage resource without having to manage credentials such as SAS tokens.

## Other role assignments for Document Intelligence Studio

If you're going to use Document Intelligence Studio and your storage account is configured with network restriction such as firewall or virtual network, another role, **Storage Blob Data Contributor**, needs to be assigned to your Document Intelligence service. Document Intelligence Studio requires this role to write blobs to your storage account when you perform Auto label, Human in the loop, or Project sharing/upgrade operations.



The screenshot shows the 'Azure role assignments' page. At the top, there's a header with 'Azure role assignments' and a '...'. Below it are buttons for '+ Add role assignment (Preview)' and 'Refresh'. A note says: 'If this identity has role assignments that you don't have permission to read, they won't be shown in the list. [Learn more](#)'. Under 'Subscription \*', a dropdown menu is open. The main table has columns: 'Role', 'Resource Name', 'Resource Type', 'Assigned To', and 'Condition'. One row is visible: 'Storage Blob Data Contributor' (highlighted with a red border), 'YourStorage' (with a storage icon), 'Storage account', 'Your-Document-Intelligence-Resource', and 'None'.

## Next steps

Configure secure access with managed identities and private endpoints

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Configure secure access with managed identities and virtual networks

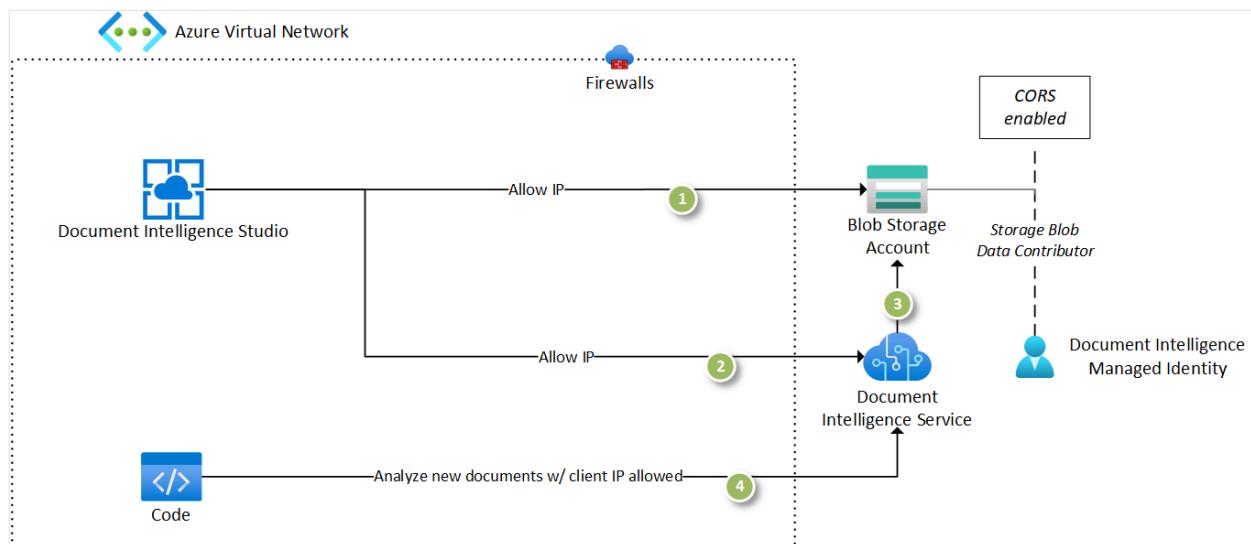
Article • 11/19/2024

This content applies to: ✓ v4.0 (GA) ✓ v3.1 (GA) ✓ v3.0 (GA) ✓ v2.1 (GA)

This how-to guide walks you through the process of enabling secure connections for your Document Intelligence resource. You can secure the following connections:

- Communication between a client application within a Virtual Network (VNET) and your Document Intelligence Resource.
- Communication between Document Intelligence Studio and your Document Intelligence resource.
- Communication between your Document Intelligence resource and a storage account (needed when training a custom model).

You're setting up your environment to secure the resources:



## Prerequisites

To get started, you need:

- An active [Azure account](#) —if you don't have one, you can [create a free account](#).
- A [Document Intelligence](#) or [Azure AI services](#) resource in the Azure portal.  
For detailed steps, see [Create an Azure AI services resource](#).

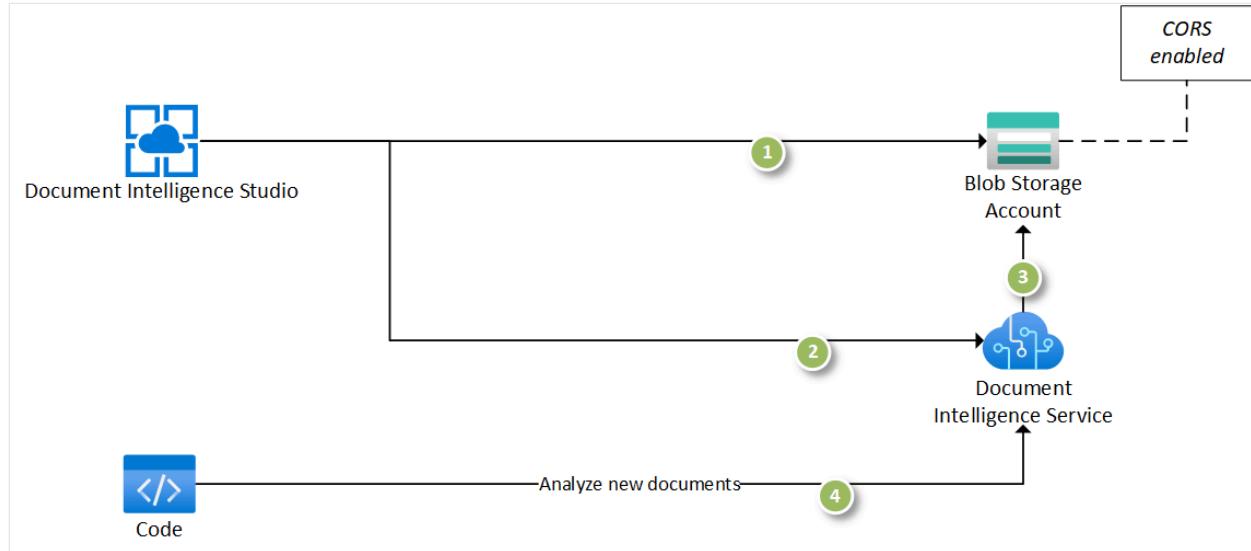
- An [Azure blob storage account](#) in the same region as your Document Intelligence resource. Create containers to store and organize your blob data within your storage account.
- An [Azure virtual network](#) in the same region as your Document Intelligence resource. Create a virtual network to deploy your application resources to train models and analyze documents.
- An **Azure data science VM** for [Windows](#) or [Linux/Ubuntu](#) to optionally deploy a data science VM in the virtual network to test the secure connections being established.

## Configure resources

Configure each of the resources to ensure that the resources can communicate with each other:

- Configure the Document Intelligence Studio to use the newly created Document Intelligence resource by accessing the settings page and selecting the resource.
- Ensure and validate that the configuration works by selecting the Read API and analyzing a sample document. If the resource was configured correctly, the request successfully completes.
- Add a training dataset to a container in the Storage account you created.
- Select the custom model tile to create a custom project. Ensure that you select the same Document Intelligence resource and the storage account you created in the previous step.
- Select the container with the training dataset you uploaded in the previous step. Ensure that if the training dataset is within a folder, the folder path is set appropriately.
- Ensure that you have the required permissions, the Studio sets the CORS setting required to access the storage account. If you don't have the permissions, you need to make certain that the CORS settings are configured on the Storage account before you can proceed.
- Ensure and validate that the Studio is configured to access your training data. If you can see your documents in the labeling experience, all the required connections are established.

You now have a working implementation of all the components needed to build a Document Intelligence solution with the default security model:



Next, complete the following steps:

- Configure managed identity on the Document Intelligence resource.
- Secure the storage account to restrict traffic from only specific virtual networks and IP addresses.
- Configure the Document Intelligence managed identity to communicate with the storage account.
- Disable public access to the Document Intelligence resource and create a private endpoint. Your resource is then only accessible from specific virtual networks and IP addresses.
- Add a private endpoint for the storage account in a selected virtual network.
- Ensure and validate that you can train models and analyze documents from within the virtual network.

## Setup managed identity for Document Intelligence

Navigate to the Document Intelligence resource in the Azure portal and select the **Identity** tab. Toggle the **System assigned** managed identity to **On** and save the changes:

The screenshot shows the Azure portal interface for a storage account named 'Form recognizer'. On the left, there's a navigation menu with sections like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, Identity (which is selected and highlighted in grey), Cost analysis, Properties, Locks, Monitoring, Alerts, and Metrics. At the top right, there are tabs for 'System assigned' (selected) and 'User assigned', with a note explaining system assigned identities. Below that is a toolbar with Save, Discard, Refresh, and Got feedback? buttons. Underneath the toolbar, there's a 'Status' section with a toggle switch set to 'On'.

## Secure the Storage account

Start configuring secure communications by navigating to the **Networking** tab on your **Storage account** in the Azure portal.

1. Under **Firewalls and virtual networks**, choose **Enabled** from **selected virtual networks and IP addresses** from the **Public network access** list.
2. Ensure that **Allow Azure services on the trusted services list to access this storage account** is selected from the **Exceptions** list.
3. **Save** your changes.

The screenshot shows the 'Firewalls and virtual networks' tab in the Azure Storage account settings. On the left, there's a navigation menu with sections like Overview, Activity log, Tags, Diagnose and solve problems, Access Control (IAM), Data migration, Events, Storage browser (preview), Data storage (Containers, File shares, Queues, Tables), Security + networking (Networking, Azure CDN, Access keys, Shared access signature, Encryption, Security), and Data management (Geo-replication). The Networking section is currently selected.

The main area displays configuration for public network access, virtual networks, and a firewall. Under 'Public network access', 'Enabled from selected virtual networks and IP addresses' is selected. Under 'Virtual networks', there are buttons for 'Add existing virtual network' and 'Add new virtual network'. A table lists 'Virtual Network', 'Subnet', 'Address range', and 'Endpoint Status' with a note 'No network selected.' Below this is a 'Firewall' section where you can add IP ranges or client IP addresses. An 'Address range' input field is present. Under 'Resource instances', it says 'Specify resource instances that will have access to your storage account based on their system-assigned managed identity'. It includes dropdowns for 'Resource type' (Select a resource type) and 'Instance name' (Select one or more instances). Finally, there's an 'Exceptions' section with checkboxes for allowing Azure services, read access to storage logging, and read access to storage metrics.

### ⓘ Note

Your storage account won't be accessible from the public internet.

Refreshing the custom model labeling page in the Studio will result in an error message.

## Enable access to storage from Document Intelligence

To ensure that the Document Intelligence resource can access the training dataset, you need to add a role assignment for your [managed identity](#).

1. Staying on the storage account window in the Azure portal, navigate to the **Access Control (IAM)** tab in the left navigation bar.
2. Select the **Add role assignment** button.

Add role assignment ...

Got feedback?

**Role** Members Review + assign

A role definition is a collection of permissions. You can use the built-in roles or you can create your own custom roles. [Learn more](#)

Showing 4 of 40 roles

Name ↑↓	Description ↑↓
Storage Blob Data Contributor	Allows for read, write and delete access to Azure Storage blob containers and data
Storage Blob Data Owner	Allows for full access to Azure Storage blob containers and data, including assigning POSIX access control.
Storage Blob Data Reader	Allows for read access to Azure Storage blob containers and data
Storage Blob Delegator	Allows for generation of a user delegation key which can be used to sign SAS tokens

3. On the **Role** tab, search for and select the **Storage Blob Data Contributor** permission and select **Next**.

4. On the **Members** tab, select the **Managed identity** option and choose + **Select members**

5. On the **Select managed identities** dialog window, select the following options:

- **Subscription.** Select your subscription.
- **Managed Identity.** Select **Form Recognizer**.
- **Select.** Choose the Document Intelligence resource you enabled with a managed identity.

The screenshot shows the 'Add role assignment' dialog in the Azure portal. The 'Members' tab is active. The 'Selected role' is 'Storage Blob Data Reader'. Under 'Assign access to', the 'Managed identity' option is selected. On the right, a sidebar titled 'Select managed identities' shows a subscription dropdown set to 'Visual Studio Enterprise Subscription' and a list of managed identities including 'Form recognizer (1)'. A 'Search by name' input field is also present.

6. Close the dialog window.

7. Finally, select **Review + assign** to save your changes.

Great! You configured your Document Intelligence resource to use a managed identity to connect to a storage account.

### 💡 Tip

When you try the [Document Intelligence Studio](#), you'll see the READ API and other prebuilt models don't require storage access to process documents. However, training a custom model requires additional configuration because the Studio can't directly communicate with a storage account. You can enable storage access by selecting **Add your client IP address** from the **Networking** tab of the storage account to configure your machine to access the storage account via IP allowlisting.

## Configure private endpoints for access from VNets

### ⚠ Note

- The resources are only accessible from the virtual network.
- Some Document Intelligence features in the Studio like auto label require the Document Intelligence Studio to have access to your storage account.
- Add our Studio IP address, 20.3.165.95, to the firewall allowlist for both Document Intelligence and Storage Account resources. This is Document Intelligence Studio's dedicated IP address and can be safely allowed.

When you connect to resources from a virtual network, adding private endpoints ensures both the storage account, and the Document Intelligence resource are accessible from the virtual network.

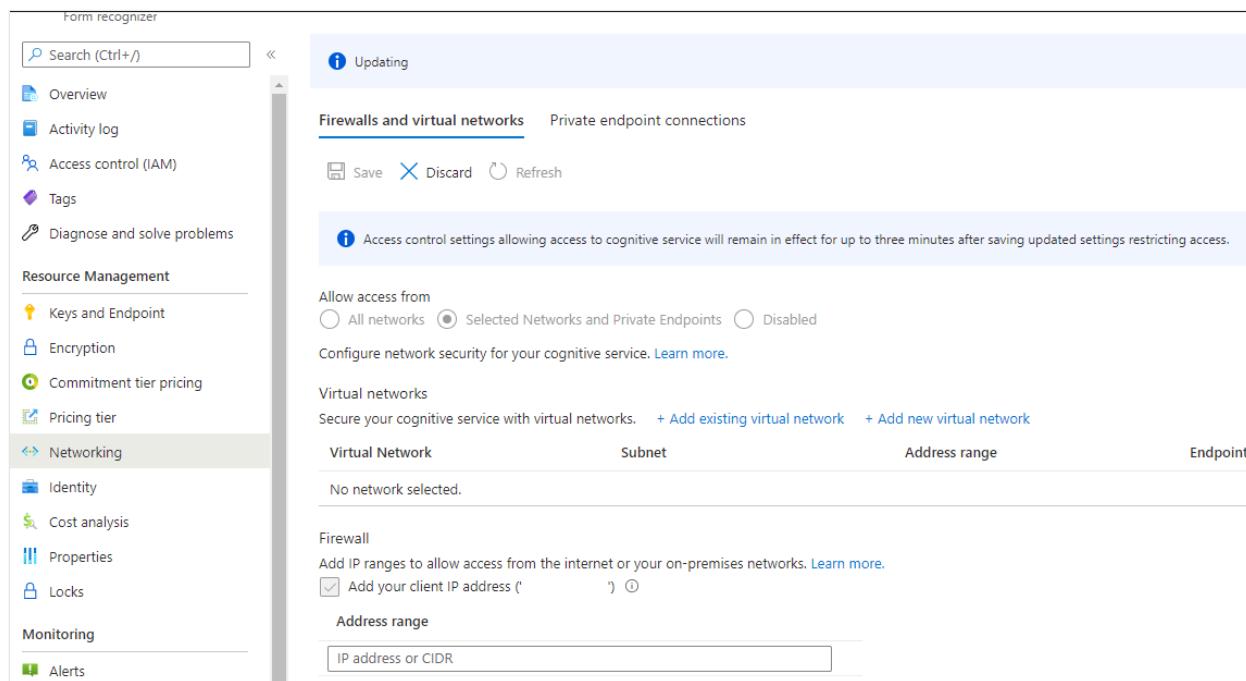
Next, configure the virtual network to ensure only resources within the virtual network or traffic router through the network have access to the Document Intelligence resource and the storage account.

## Enable your firewalls and virtual networks

1. In the Azure portal, navigate to your Document Intelligence resource.
2. Select the **Networking** tab from the left navigation bar.
3. Enable the **Selected Networking and Private Endpoints** option from the **Firewalls and virtual networks** tab and select save.

### Note

If you try accessing any of the Document Intelligence Studio features, you'll see an access denied message. To enable access from the Studio on your machine, select the **Add your client IP address** checkbox and **Save** to restore access.



The screenshot shows the Azure portal interface for a 'Form recognizer' resource. The left sidebar has a 'Networking' section highlighted. The main content area shows the 'Firewalls and virtual networks' tab selected. It displays a note about access control settings being effective for up to three minutes after saving. Below this, there's an 'Allow access from' section with a radio button for 'Selected Networks and Private Endpoints' selected. A 'Virtual networks' section allows adding existing or new virtual networks. A 'Firewall' section includes a checkbox for 'Add your client IP address' which is checked. An 'Address range' input field is also present.

## Configure your private endpoint

1. Navigate to the **Private endpoint connections** tab and select the **+ Private endpoint**. You're navigated to the **Create a private endpoint** dialog page.
2. On the **Create private endpoint** dialog page, select the following options:
  - **Subscription**. Select your billing subscription.

- **Resource group.** Select the appropriate resource group.
- **Name.** Enter a name for your private endpoint.
- **Region.** Select the same region as your virtual network.
- Select **Next: Resource.**

**Create a private endpoint** ...

**Basics**   **Resource**   **Virtual Network**   **Tags**   **Review + create**

Use private endpoints to privately connect to a service or resource. Your private endpoint must be in the same region as your virtual network, but can be in a different region from the private link resource that you are connecting to. [Learn more](#)

**Project details**

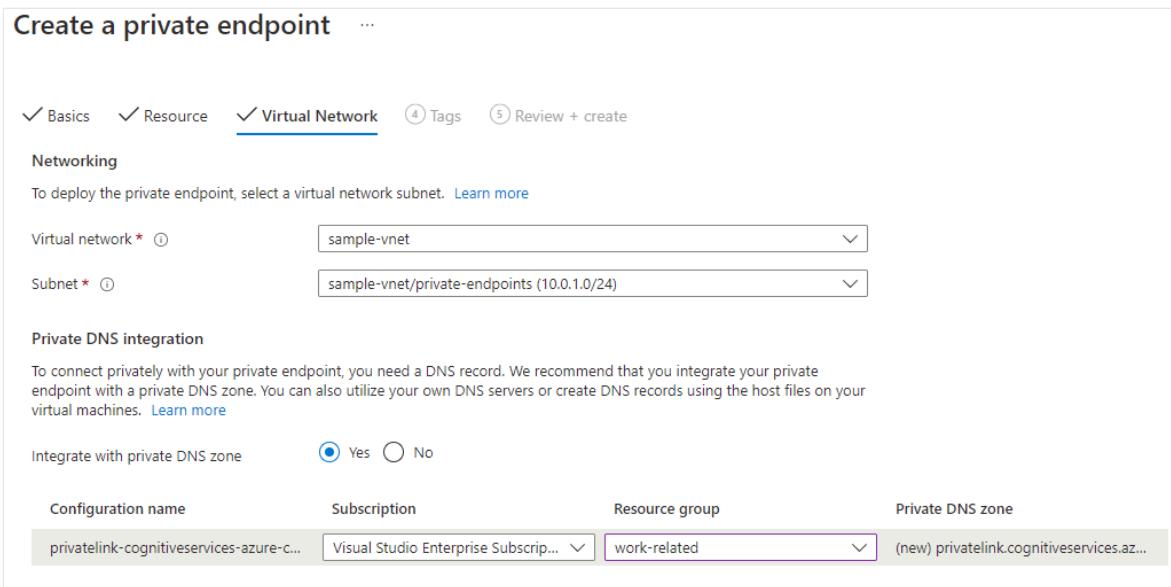
Subscription *	Visual Studio Enterprise Subscription
Resource group *	work-related
	<a href="#">Create new</a>

**Instance details**

Name *	my-fr-endpoint
Region *	Southeast Asia

## Configure your virtual network

1. On the **Resource** tab, accept the default values and select **Next: Virtual Network**.
2. On the **Virtual Network** tab, make sure that you select the virtual network that you created.
3. If you have multiple subnets, select the subnet where you want the private endpoint to connect. Accept the default value to **Dynamically allocate IP address**.
4. Select **Next: DNS**
5. Accept the default value **Yes** to **integrate with private DNS zone**.



6. Accept the remaining defaults and select **Next: Tags**.

7. Select **Next: Review + create**.

Well done! Your Document Intelligence resource now is only accessible from the virtual network and any IP addresses in the IP allowlist.

## Configure private endpoints for storage

Navigate to your **storage account** on the Azure portal.

1. Select the **Networking** tab from the left navigation menu.
2. Select the **Private endpoint connections** tab.
3. Choose **add + Private endpoint**.
4. Provide a name and choose the same region as the virtual network.
5. Select **Next: Resource**.

## Create a private endpoint

**Basics**   [Resource](#)   [Virtual Network](#)   [Tags](#)   [Review + create](#)

Use private endpoints to privately connect to a service or resource. Your private endpoint must be in the same region as your virtual network, but can be in a different region from the private link resource that you are connecting to. [Learn more](#)

**Project details**

Subscription *	Visual Studio Enterprise Subscription	▼
Resource group *	work-related	▼
	<a href="#">Create new</a>	

**Instance details**

Name *	my-stg-endpoint	✓
Region *	Southeast Asia	▼

6. On the resource tab, select **blob** from the Target sub-resource list.

7. select Next: Virtual Network.

# Create a private endpoint

Basics    **Resource**    Virtual Network    Tags    Review + create

Private Link offers options to create private endpoints for different Azure resources, like your private link service, a SQL server, or an Azure storage account. Select which resource you would like to connect to using this private endpoint. [Learn more](#)

Subscription	Visual Studio Enterprise Subscription (083694cc-cace-414f-9d55-d6b9071db956)
Resource type	Microsoft.Storage/storageAccounts
Resource	vikurpadwork
Target sub-resource *	<input type="text" value="blob"/> <div>         blob          table          queue          file          web          dfs       </div>

8. Select the **Virtual network** and **Subnet**. Make sure **Enable network policies for all private endpoints in this subnet** is selected and the **Dynamically allocate IP address** is enabled.

### 9. Select Next: DNS.

10. Make sure that **Yes** is enabled for **Integrate with private DNS zone**.

11. Select **Next: Tags**.

12. Select **Next: Review + create**.

Great work! You now have all the connections between the Document Intelligence resource and storage configured to use managed identities.

 **Note**

The resources are only accessible from the virtual network and allowed IPs.

Studio access and analyze requests to your Document Intelligence resource will fail unless the request originates from the virtual network or is routed via the virtual network.

## Validate your deployment

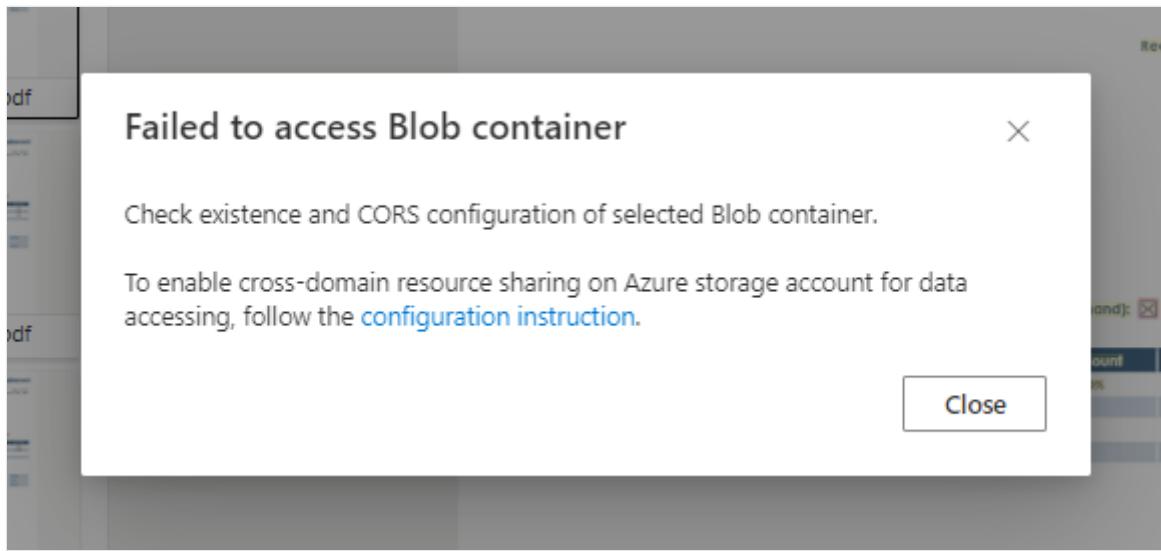
To validate your deployment, you can deploy a virtual machine (VM) to the virtual network and connect to the resources.

1. Configure a [Data Science VM](#) in the virtual network.
2. Remotely connect into the VM from your desktop and launch a browser session that accesses Document Intelligence Studio.
3. Analyze requests and the training operations should now work successfully.

That's it! You can now configure secure access for your Document Intelligence resource with managed identities and private endpoints.

## Common error messages

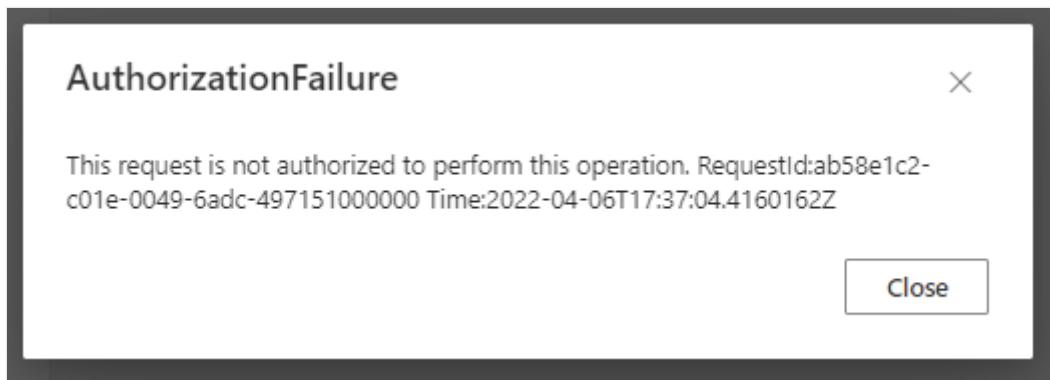
- Failed to access Blob container:



**Resolution:**

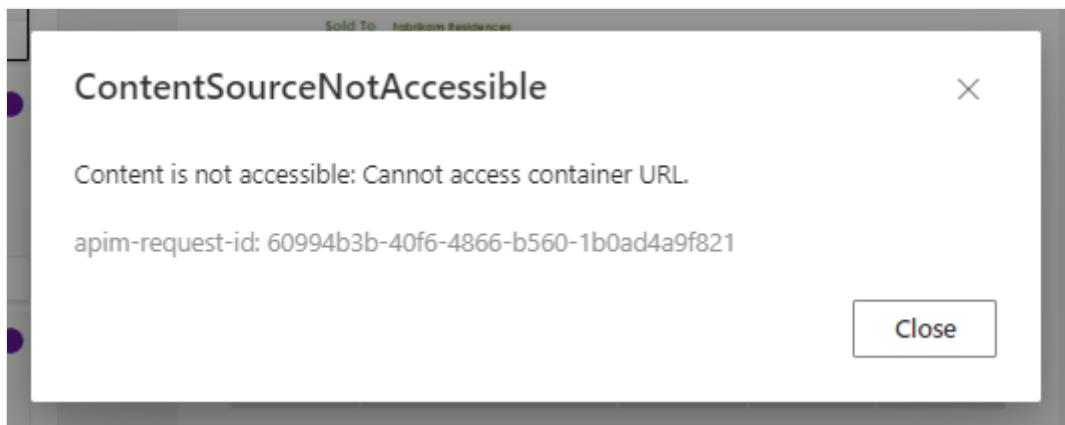
1. [Configure CORS](#).
2. Make sure the client computer can access Document Intelligence resource and storage account, either they are in the same **VNET**, or client IP address is allowed in **Networking > Firewalls and virtual networks** setting page of both Document Intelligence resource and storage account.

- **AuthorizationFailure:**



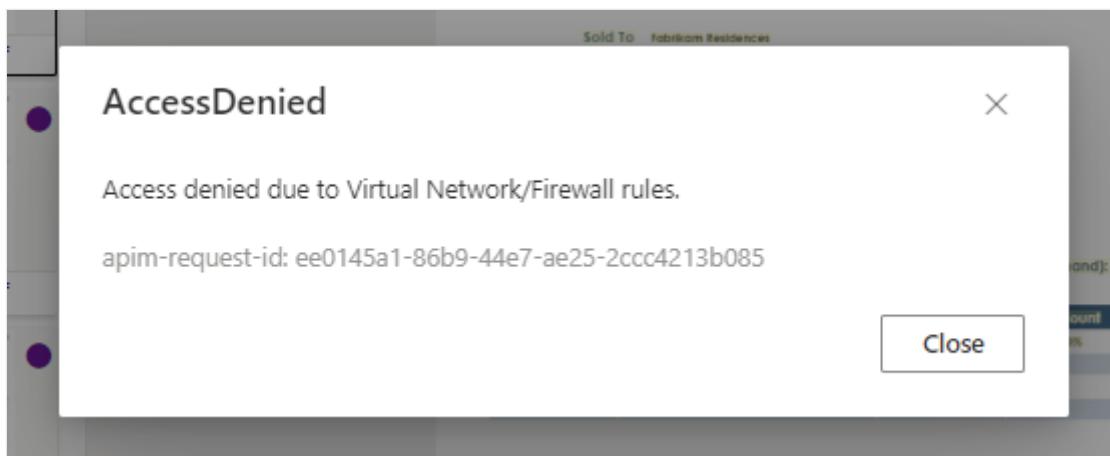
**Resolution:** Make sure the client computer can access Document Intelligence resource and storage account, either they are in the same **VNET**, or client IP address is allowed in **Networking > Firewalls and virtual networks** setting page of both Document Intelligence resource and storage account.

- **ContentSourceNotAccessible:**



**Resolution:** Make sure you grant your Document Intelligence managed identity the role of **Storage Blob Data Contributor** and enabled **Trusted services** access or **Resource instance** rules on the networking tab.

- **AccessDenied:**



**Resolution:** Make sure the client computer can access Document Intelligence resource and storage account, either they are in the same **VNET**, or client IP address is allowed in **Networking > Firewalls and virtual networks** setting page of both Document Intelligence resource and storage account.

## Next steps

[Access Azure Storage from a web app using managed identities](#)

## Feedback

Was this page helpful?

Yes

No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Encrypt data at rest

Article • 03/19/2025

This content applies to:  v4.0 (GA)  v3.1 (GA)  v3.0 (GA)  v2.1 (GA)

## Important

- Earlier versions of customer managed keys ( CMK ) only encrypted your models.
- Beginning with the **07/31/2023** release, all new resources utilize customer-managed keys to encrypt both models and document results.
- [Delete analyze response](#). the **analyze response** is stored for 24 hours from when the operation completes for retrieval. For scenarios where you want to delete the response sooner, use the delete analyze response API to delete the response.
- To upgrade an existing service to encrypt both the models and the data, disable and reenable the customer managed key.

Azure AI Document Intelligence automatically encrypts your data when persisting it to the cloud. Document Intelligence encryption protects your data to help you to meet your organizational security and compliance commitments.

## About Azure AI services encryption

Data is encrypted and decrypted using [FIPS 140-2](#)-compliant [256-bit AES](#) encryption. Encryption and decryption are transparent, meaning encryption and access are managed for you. Your data is secure by default. You don't need to modify your code or applications to take advantage of encryption.

## About encryption key management

By default, your subscription uses Microsoft-managed encryption keys. You can also manage your subscription with your own keys, which are called customer-managed keys. When you use customer-managed keys, you have greater flexibility in the way you create, rotate, disable, and revoke access controls. You can also audit the encryption keys that you use to protect your data. If customer-managed keys are configured for your subscription, double encryption is provided. With this second layer of protection, you can control the encryption key through your Azure Key Vault.

### Important

- Customer-managed keys are only available for resources created after May 11, 2020. To use customer-managed keys with Document Intelligence, you need to create a new Document Intelligence resource. Once the resource is created, you can use Azure Key Vault to set up your managed identity.
- The scope for data encrypted with customer-managed keys includes the `analysis response` stored for 24 hours, allowing the operation results to be retrieved during that 24-hour time period.

## Customer-managed keys with Azure Key Vault

When you use customer-managed keys, you must use Azure Key Vault to store them. You can either create your own keys and store them in a key vault, or you can use the Key Vault APIs to generate keys. The Azure AI services resource and the key vault must be in the same region and in the same Microsoft Entra tenant, but they can be in different subscriptions. For more information about Key Vault, see [What is Azure Key Vault?](#).

When you create a new Azure AI services resource, it's always encrypted by using Microsoft-managed keys. It's not possible to enable customer-managed keys when you create the resource. Customer-managed keys are stored in Key Vault. The key vault needs to be provisioned with access policies that grant key permissions to the managed identity that's associated with the Azure AI services resource. The managed identity is available only after the resource is created by using the pricing tier that's required for customer-managed keys.

Enabling customer-managed keys also enables a system-assigned [managed identity](#), a feature of Microsoft Entra ID. After the system-assigned managed identity is enabled, this resource is registered with Microsoft Entra ID. After being registered, the managed identity is given access to the key vault that's selected during customer-managed key setup.

### Important

If you disable system-assigned managed identities, access to the key vault is removed and any data that's encrypted with the customer keys is no longer accessible. Any features that depend on this data stop working.

### (i) Important

Managed identities don't currently support cross-directory scenarios. When you configure customer-managed keys in the Azure portal, a managed identity is automatically assigned behind the scenes. If you subsequently move the subscription, resource group, or resource from one Microsoft Entra directory to another, the managed identity that's associated with the resource isn't transferred to the new tenant, so customer-managed keys might no longer work. For more information, see [Transferring a subscription between Microsoft Entra directories](#) in [FAQs and known issues with managed identities for Azure resources](#).

## Configure Key Vault

When you use customer-managed keys, you need to set two properties in the key vault, **Soft Delete** and **Do Not Purge**. These properties aren't enabled by default, but you can enable them on a new or existing key vault by using the Azure portal, PowerShell, or Azure CLI.

### (i) Important

If the **Soft Delete** and **Do Not Purge** properties aren't enabled and you delete your key, you can't recover the data in your Azure AI services resource.

To learn how to enable these properties on an existing key vault, see [Azure Key Vault recovery management with soft delete and purge protection](#).

## Enable customer-managed keys for your resource

To enable customer-managed keys in the Azure portal, follow these steps:

1. Go to your Azure AI services resource.
2. On the left, select **Encryption**.
3. Under **Encryption type**, select **Customer Managed Keys**, as shown in the following screenshot.

The screenshot shows the Azure portal interface for managing encryption settings. On the left, there's a sidebar with various service management options like Overview, Activity log, Access control (IAM), Tags, and Diagnose and solve problems. Below that is a 'RESOURCE MANAGEMENT' section with Quick start, Keys and Endpoint, Encryption (which is currently selected), Pricing tier, Virtual network, Identity, Billing By Subscription, Properties, Locks, and Export template. The main content area is titled 'Encryption' and contains a 'Save' and 'Discard' button. It explains that cognitive services encryption protects data at rest using Microsoft Managed Keys by default, or Customer Managed Keys if chosen. A note states that after enabling encryption, only new data will be encrypted, and existing files will be retroactively encrypted. There's a link to 'Learn More about Azure Cognitive services Encryption'. Under 'Encryption type', the 'Customer Managed Keys' radio button is selected, with a tooltip explaining that the cognitive service account will be granted access to the selected key vault, enabling both soft delete and purge protection. Below that is an 'Encryption key' section with 'Enter key URI' selected. The 'Key URI \*' input field is empty. At the bottom, there's a 'Subscription' dropdown set to 'AICP-DEV'.

## Specify a key

After you enable customer-managed keys, you can specify a key to associate with the Azure AI services resource.

### Specify a key as a URI

To specify a key as a URI, follow these steps:

1. In the Azure portal, go to your key vault.
2. Under **Settings**, select **Keys**.
3. Select the desired key, and then select the key to view its versions. Select a key version to view the settings for that version.
4. Copy the **Key Identifier** value, which provides the URI.

The screenshot shows the 'Properties' section of a key in the Azure Key Vault. Key details include:

- Key Type:** RSA
- RSA Key Size:** 2048
- Created:** 4/9/2019, 12:50:38 PM
- Updated:** 4/9/2019, 12:50:38 PM

**Key Identifier:** <key-uri>

**Settings:**

- Set activation date?
- Set expiration date?
- Enabled?  Yes  No

**Tags:** 0 tags

**Permitted operations:**

<input checked="" type="checkbox"/> Encrypt	<input checked="" type="checkbox"/> Sign	<input checked="" type="checkbox"/> Wrap Key
<input checked="" type="checkbox"/> Decrypt	<input checked="" type="checkbox"/> Verify	<input checked="" type="checkbox"/> Unwrap Key

5. Go back to your Azure AI services resource, and then select **Encryption**.

6. Under **Encryption key**, select **Enter key URI**.

7. Paste the URI that you copied into the **Key URI** box.

The screenshot shows the 'Encryption' blade for a Cognitive Service named 'CMK-Test'. The left sidebar includes options like Overview, Activity log, Access control (IAM), Tags, and Keys and Endpoint. The 'Encryption' option is selected.

**Encryption**

**Encryption type:**  Customer Managed Keys  Microsoft Managed Keys

**Encryption key:**  Enter key URI  Select from Key Vault

**Key URI \***: <key uri>

**Subscription**: AICP-DEV

Information on the right side states:

- Cognitive services encryption protects your data at rest. Azure Cognitive services encrypts your data as it's written in our datacenters, and automatically decrypts it for you as you access it.
- By default, data in the cognitive service account is encrypted using Microsoft Managed Keys. You may choose to bring your own key.
- Please note that after enabling Cognitive Service Encryption, only new data will be encrypted, and any existing files in this cognitive service account will retroactively get encrypted by a background encryption process.

[Learn More about Azure Cognitive services Encryption](#)

8. Under **Subscription**, select the subscription that contains the key vault.

9. Save your changes.

# Specify a key from a key vault

To specify a key from a key vault, first make sure that you have a key vault that contains a key. Then follow these steps:

1. Go to your Azure AI services resource, and then select **Encryption**.
2. Under **Encryption key**, select **Select from Key Vault**.
3. Select the key vault that contains the key that you want to use.
4. Select the key that you want to use.

The screenshot shows the Microsoft Azure portal interface. At the top, there's a blue header bar with the Microsoft Azure logo and a search bar labeled 'Search resources, services, and docs (G+/)'. Below the header, the URL 'Home > CMKTest01-SB - Encryption > Select key from Azure Key Vault' is visible. The main content area has a title 'Select key from Azure Key Vault'. It contains four input fields with dropdown menus:

- 'Subscription \*' dropdown set to 'AICP-DEV'.
- 'Key vault \*' dropdown set to 'CMKTest-01SB', with a 'Create new' link below it.
- 'Key \*' dropdown set to 'CMKTest-01SB', with a 'Create new' link below it.
- 'Version \*' dropdown set to '19fc5cfacbd34e47b373709c1e400902', with a 'Create new' link below it.

5. Save your changes.

# Update the key version

When you create a new version of a key, update the Azure AI services resource to use the new version. Follow these steps:

1. Go to your Azure AI services resource, and then select **Encryption**.
2. Enter the URI for the new key version. Alternately, you can select the key vault and then select the key again to update the version.
3. Save your changes.

# Use a different key

To change the key that you use for encryption, follow these steps:

1. Go to your Azure AI services resource, and then select **Encryption**.

2. Enter the URI for the new key. Alternately, you can select the key vault and then select a new key.
3. Save your changes.

## Rotate customer-managed keys

You can rotate a customer-managed key in Key Vault according to your compliance policies. When the key is rotated, you must update the Azure AI services resource to use the new key URI. To learn how to update the resource to use a new version of the key in the Azure portal, see [Update the key version](#).

Rotating the key doesn't trigger re-encryption of data in the resource. No further action is required from the user.

## Revoke access to customer-managed keys

To revoke access to customer-managed keys, use PowerShell or Azure CLI. For more information, see [Azure Key Vault PowerShell](#) or [Azure Key Vault CLI](#). Revoking access effectively blocks access to all data in the Azure AI services resource, because the encryption key is inaccessible by Azure AI services.

## Disable customer-managed keys

When you disable customer-managed keys, your Azure AI services resource is then encrypted with Microsoft-managed keys. To disable customer-managed keys, follow these steps:

1. Go to your Azure AI services resource, and then select **Encryption**.
2. Clear the checkbox that's next to **Use your own key**.

## Next steps

- [Learn more about Azure Key Vault](#)

---

## Feedback

Was this page helpful?

 Yes

 No

# Authenticate requests to Azure AI services

Article • 02/07/2025

Each request to an Azure AI service must include an authentication header. This header passes along a resource key or authentication token, which is used to validate your subscription for a service or group of services. In this article, you'll learn about three ways to authenticate a request and the requirements for each.

- Authenticate with a [single-service](#) or [multi-service](#) resource key.
- Authenticate with a [token](#).
- Authenticate with [Microsoft Entra ID](#).

## Prerequisites

Before you make a request, you need an Azure account and an Azure AI services subscription. If you already have an account, go ahead and skip to the next section. If you don't have an account, we have a guide to get you set up in minutes: [Create an Azure AI services resource](#).

Go to your resource in the Azure portal. The **Keys & Endpoint** section can be found in the **Resource Management** section. Copy your endpoint and access key as you'll need both for authenticating your API calls. You can use either `KEY1` or `KEY2`. Always having two keys allows you to securely rotate and regenerate keys without causing a service disruption. The length of the key can vary depending on the API version used to create or regenerate the key.

## Authentication headers

Let's quickly review the authentication headers available for use with Azure AI services.

[+] Expand table

Header	Description
Ocp-Apim-Subscription-Key	Use this header to authenticate with a resource key for a specific service or a multi-service resource key.
Ocp-Apim-Subscription-	This header is only required when using a multi-service resource key with the <a href="#">Azure AI Translator service</a> . Use this header to specify the resource

Header	Description
Region	region.
Authorization	Use this header if you are using an access token. The steps to perform a token exchange are detailed in the following sections. The value provided follows this format: <code>Bearer &lt;TOKEN&gt;</code> .

## Authenticate with a single-service resource key

The first option is to authenticate a request with a resource key for a specific service, like Azure AI Translator. The keys are available in the Azure portal for each resource that you've created. Go to your resource in the Azure portal. The **Keys & Endpoint** section can be found in the **Resource Management** section. Copy your endpoint and access key as you'll need both for authenticating your API calls. You can use either `KEY1` or `KEY2`. Always having two keys allows you to securely rotate and regenerate keys without causing a service disruption.

To use a resource key to authenticate a request, it must be passed along as the `Ocp-Apim-Subscription-Key` header. This is a sample call to the Azure AI Translator service:

This is a sample call to the Translator service:

cURL

```
curl -X POST 'https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=de' \
-H 'Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY' \
-H 'Content-Type: application/json' \
--data-binary '[{"text": "How much for the cup of coffee?"}]' | json_pp
```

## Authenticate with a multi-service resource key

You can use a [multi-service](#) resource key to authenticate requests. The main difference is that the multi-service resource key isn't tied to a specific service, rather, a single key can be used to authenticate requests for multiple Azure AI services. See [Azure AI services pricing](#) for information about regional availability, supported features, and pricing.

The resource key is provided in each request as the `Ocp-Apim-Subscription-Key` header.

## Supported regions

When using the [Azure AI services multi-service](#) resource key to make a request to `api.cognitive.microsoft.com`, you must include the region in the URL. For example: `westus.api.cognitive.microsoft.com`.

When using a multi-service resource key with [Azure AI Translator](#), you must specify the resource region with the `Ocp-Apim-Subscription-Region` header.

Multi-service resource authentication is supported in these regions:

- `australiaeast`
- `brazilsouth`
- `canadacentral`
- `centralindia`
- `eastasia`
- `eastus`
- `japaneast`
- `northeurope`
- `southcentralus`
- `southeastasia`
- `uksouth`
- `westcentralus`
- `westeurope`
- `westus`
- `westus2`
- `francecentral`
- `koreacentral`
- `northcentralus`
- `southafricanorth`
- `uaenorth`
- `switzerlandnorth`

## Sample requests

This is a sample call to the Azure AI Translator service:

cURL

```
curl -X POST 'https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=de' \
-H 'Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY' \
-H 'Ocp-Apim-Subscription-Region: YOUR_SUBSCRIPTION_REGION' \
```

```
-H 'Content-Type: application/json' \
--data-raw '[{"text": "How much for the cup of coffee?" }]' | json_pp
```

# Authenticate with an access token

Some Azure AI services accept, and in some cases require, an access token. Currently, these services support access tokens:

- Text Translation API
- Speech Services: Speech to text API
- Speech Services: Text to speech API

## ⚠ Warning

The services that support access tokens may change over time, please check the API reference for a service before using this authentication method.

Both single service and multi-service resource keys can be exchanged for authentication tokens. Authentication tokens are valid for 10 minutes. They're stored in JSON Web Token (JWT) format and can be queried programmatically using the [JWT libraries ↗](#).

Access tokens are included in a request as the `Authorization` header. The token value provided must be preceded by `Bearer`, for example: `Bearer YOUR_AUTH_TOKEN`.

## Sample requests

Use this URL to exchange a resource key for an access token: `https://YOUR-REGION.api.cognitive.microsoft.com/sts/v1.0/issueToken`.

### cURL

```
curl -v -X POST \
"https://YOUR-REGION.api.cognitive.microsoft.com/sts/v1.0/issueToken" \
-H "Content-type: application/x-www-form-urlencoded" \
-H "Content-length: 0" \
-H "Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY"
```

These multi-service regions support token exchange:

- `australiaeast`
- `brazilsouth`
- `canadacentral`

- centralindia
- eastasia
- eastus
- japaneast
- northeurope
- southcentralus
- southeastasia
- uksouth
- westcentralus
- westeurope
- westus
- westus2

After you get an access token, you'll need to pass it in each request as the `Authorization` header. This is a sample call to the Azure AI Translator service:

#### cURL

```
curl -X POST 'https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=de' \
-H 'Authorization: Bearer YOUR_AUTH_TOKEN' \
-H 'Content-Type: application/json' \
--data-binary '[{"text": "How much for the cup of coffee?"}]' | json_pp
```

## Authenticate with Microsoft Entra ID

### ⓘ Important

Microsoft Entra authentication always needs to be used together with custom subdomain name of your Azure resource. [Regional endpoints](#) do not support Microsoft Entra authentication.

In the previous sections, we showed you how to authenticate against Azure AI services using a single-service or multi-service subscription key. While these keys provide a quick and easy path to start development, they fall short in more complex scenarios that require Azure [role-based access control \(Azure RBAC\)](#). Let's take a look at what's required to authenticate using Microsoft Entra ID.

In the following sections, you'll use either the Azure Cloud Shell environment or the Azure CLI to create a subdomain, assign roles, and obtain a bearer token to call the

Azure AI services. If you get stuck, links are provided in each section with all available options for each command in Azure Cloud Shell/Azure CLI.

### ⓘ Important

If your organization is doing authentication through Microsoft Entra ID, you should [disable local authentication](#) (authentication with keys) so that users in the organization must always use Microsoft Entra ID.

## Create a resource with a custom subdomain

The first step is to create a custom subdomain. If you want to use an existing Azure AI services resource which does not have custom subdomain name, follow the instructions in [Azure AI services custom subdomains](#) to enable custom subdomain for your resource.

1. Start by opening the Azure Cloud Shell. Then [select a subscription](#):

```
PowerShell
```

```
Set-AzContext -SubscriptionName <SubscriptionName>
```

2. Next, [create an Azure AI services resource](#) with a custom subdomain. The subdomain name needs to be globally unique and cannot include special characters, such as: ".", "!", ",", ".".

```
PowerShell
```

```
$account = New-AzCognitiveServicesAccount -ResourceGroupName  
<RESOURCE_GROUP_NAME> -name <ACCOUNT_NAME> -Type <ACCOUNT_TYPE> -  
SkuName <SUBSCRIPTION_TYPE> -Location <REGION> -CustomSubdomainName  
<UNIQUE_SUBDOMAIN>
```

3. If successful, the **Endpoint** should show the subdomain name unique to your resource.

## Assign a role to a service principal

Now that you have a custom subdomain associated with your resource, you're going to need to assign a role to a service principal.

### ⓘ Note

Keep in mind that Azure role assignments may take up to five minutes to propagate.

1. First, let's register an [Microsoft Entra application](#).

```
PowerShell
```

```
$SecureStringPassword = ConvertTo-SecureString -String <YOUR_PASSWORD>  
-AsPlainText -Force  
  
$app = New-AzureADApplication -DisplayName <APP_DISPLAY_NAME> -  
IdentifierUris <APP_URIS> -PasswordCredentials $SecureStringPassword
```

You're going to need the **ApplicationId** in the next step.

2. Next, you need to [create a service principal](#) for the Microsoft Entra application.

```
PowerShell
```

```
New-AzADServicePrincipal -ApplicationId <APPLICATION_ID>
```

① Note

If you register an application in the Azure portal, this step is completed for you.

3. The last step is to [assign the "Cognitive Services User" role](#) to the service principal (scoped to the resource). By assigning a role, you're granting service principal access to this resource. You can grant the same service principal access to multiple resources in your subscription.

① Note

The ObjectId of the service principal is used, not the ObjectId for the application. The ACCOUNT\_ID will be the Azure resource Id of the Azure AI services account you created. You can find Azure resource Id from "properties" of the resource in Azure portal.

```
Azure CLI
```

```
New-AzRoleAssignment -ObjectId <SERVICE_PRINCIPAL_OBJECTID> -Scope  
<ACCOUNT_ID> -RoleDefinitionName "Cognitive Services User"
```

## Sample request

In this sample, a password is used to authenticate the service principal. The token provided is then used to call the Computer Vision API.

1. Get your **TenantId**:

```
PowerShell
```

```
$context=Get-AzContext  
$context.Tenant.Id
```

2. Get a token:

```
PowerShell
```

```
$tenantId = $context.Tenant.Id  
$clientId = $app.ApplicationId  
$clientSecret = "<YOUR_PASSWORD>"  
$resourceUrl = "https://cognitiveservices.azure.com/"  
  
$tokenEndpoint =  
"https://login.microsoftonline.com/$tenantId/oauth2/token"  
$body = @{  
    grant_type      = "client_credentials"  
    client_id      = $clientId  
    client_secret   = $clientSecret  
    resource        = $resourceUrl  
}  
  
$responseToken = Invoke-RestMethod -Uri $tokenEndpoint -Method Post -  
Body $body  
$accessToken = $responseToken.access_token
```

 **Note**

Anytime you use passwords in a script, the most secure option is to use the PowerShell Secrets Management module and integrate with a solution such as Azure Key Vault.

3. Call the Computer Vision API:

```
PowerShell
```

```
$url = $account.Endpoint+"vision/v1.0/models"  
$result = Invoke-RestMethod -Uri $url -Method Get -Headers
```

```
@{ "Authorization" = "Bearer $accessToken" } -Verbose  
$result | ConvertTo-Json
```

Alternatively, the service principal can be authenticated with a certificate. Besides service principal, user principal is also supported by having permissions delegated through another Microsoft Entra application. In this case, instead of passwords or certificates, users would be prompted for two-factor authentication when acquiring token.

## Authorize access to managed identities

Azure AI services support Microsoft Entra authentication with [managed identities for Azure resources](#). Managed identities for Azure resources can authorize access to Azure AI services resources using Microsoft Entra credentials from applications running in Azure virtual machines (VMs), function apps, virtual machine scale sets, and other services. By using managed identities for Azure resources together with Microsoft Entra authentication, you can avoid storing credentials with your applications that run in the cloud.

### Enable managed identities on a VM

Before you can use managed identities for Azure resources to authorize access to Azure AI services resources from your VM, you must enable managed identities for Azure resources on the VM. To learn how to enable managed identities for Azure Resources, see:

- [Azure portal](#)
- [Azure PowerShell](#)
- [Azure CLI](#)
- [Azure Resource Manager template](#)
- [Azure Resource Manager client libraries](#)

For more information about managed identities, see [Managed identities for Azure resources](#).

### Use Azure key vault to securely access credentials

You can [use Azure Key Vault](#) to securely develop Azure AI services applications. Key Vault enables you to store your authentication credentials in the cloud, and reduces the

chances that secrets may be accidentally leaked, because you won't store security information in your application.

Authentication is done via Microsoft Entra ID. Authorization may be done via Azure role-based access control (Azure RBAC) or Key Vault access policy. Azure RBAC can be used for both management of the vaults and access data stored in a vault, while key vault access policy can only be used when attempting to access data stored in a vault.

## Related content

- [What are Azure AI services?](#)
  - [Azure AI services pricing ↗](#)
  - [Custom subdomains](#)
- 

## Feedback

Was this page helpful?



[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Install and run containers

Article • 04/04/2025

This content applies to:  v3.0 (GA)  v3.1 (GA)  v4.0 (GA)

Azure AI Document Intelligence is an Azure AI service that lets you build automated data processing software using machine-learning technology. Document Intelligence enables you to identify and extract text, key/value pairs, selection marks, table data, and more from your documents. The results are delivered as structured data that includes the relationships in the original file. Containers process only the data provided to them and solely utilize the resources they're permitted to access. Containers can't process data from other regions.

In this article you can learn how to download, install, and run Document Intelligence containers. Containers enable you to run the Document Intelligence service in your own environment. Containers are great for specific security and data governance requirements.

- Layout model is supported by Document Intelligence v4.0 containers.
- Read, Layout, ID Document, Receipt, and Invoice models are supported by Document Intelligence v3.1 containers.
- Read, Layout, General Document, Business Card, and Custom models are supported by Document Intelligence v3.0 containers.

## Version support

Support for containers is currently available with Document Intelligence version [v3.0: 2022-08-31 \(GA\)](#) for all models, [v3.1 2023-07-31 \(GA\)](#) for Read, Layout, ID Document, Receipt, and Invoice models, and [v4.0 2024-11-30 \(GA\)](#) for Layout:

- REST API v3.0: 2022-08-31 (GA)
- REST API v3.1: 2023-07-31 (GA)
- REST API v4.0: 2024-11-30 (GA)
- Client libraries targeting REST API v3.0: 2022-08-31 (GA)
- Client libraries targeting REST API v3.1: 2023-07-31 (GA)
- Client libraries targeting REST API v4.0: 2024-11-30 (GA)

## Prerequisites

To get started, you need an active [Azure account](#). If you don't have one, you can [create a free account](#).

You also need the following to use Document Intelligence containers:

 Expand table

Required	Purpose
Familiarity with Docker	You should have a basic understanding of Docker concepts, like registries, repositories, containers, and container images, as well as knowledge of basic <a href="#">Docker terminology and commands</a> .
Docker Engine installed	<ul style="list-style-type: none"> <li>You need the Docker Engine installed on a <a href="#">host computer</a>. Docker provides packages that configure the Docker environment on <a href="#">macOS</a>, <a href="#">Windows</a>, and <a href="#">Linux</a>. For a primer on Docker and container basics, see the <a href="#">Docker overview</a>.</li> <li>Docker must be configured to allow the containers to connect with and send billing data to Azure.</li> <li>On <a href="#">Windows</a>, Docker must also be configured to support <a href="#">Linux</a> containers.</li> </ul>
Document Intelligence resource	<p>A <a href="#">single-service Azure AI Document Intelligence</a> or <a href="#">multi-service</a> resource in the Azure portal. To use the containers, you must have the associated key and endpoint URI. Both values are available on the Azure portal <a href="#">Document Intelligence Keys and Endpoint</a> page:</p> <ul style="list-style-type: none"> <li>{FORM_RECOGNIZER_KEY}: one of the two available resource keys.</li> <li>{FORM_RECOGNIZER_ENDPOINT_URI}: the endpoint for the resource used to track billing information.</li> </ul>

[\[+\] Expand table](#)

Optional	Purpose
Azure CLI (command-line interface)	The <a href="#">Azure CLI</a> enables you to use a set of online commands to create and manage Azure resources. It's available to install in Windows, macOS, and Linux environments and can be run in a Docker container and Azure Cloud Shell.

## Host computer requirements

The host is a x64-based computer that runs the Docker container. It can be a computer on your premises or a Docker hosting service in Azure, such as:

- [Azure Kubernetes Service](#).
- [Azure Container Instances](#).
- A [Kubernetes](#) cluster deployed to [Azure Stack](#). For more information, see [Deploy Kubernetes to Azure Stack](#).

 **Note**

The Studio container can't be deployed and run in Azure Kubernetes Service. The Studio container is only supported to run on local machines.

## Container requirements and recommendations

### Required supporting containers

The following table lists one or more supporting containers for each Document Intelligence container you download. For more information, see the [Billing](#) section.

 Expand table

Feature container	Supporting containers
Read	Not required
Layout	Not required
Business Card	Read
General Document	Layout
Invoice	Layout
Receipt	Read or Layout
ID Document	Read
Custom Template	Layout

## Recommended CPU cores and memory

### Note

The minimum and recommended values are based on Docker limits and *not* the host machine resources.

## Document Intelligence containers

 Expand table

Container	Minimum	Recommended
Read	8 cores, 10-GB memory	8 cores, 24-GB memory
Layout	8 cores, 16-GB memory	8 cores, 24-GB memory
Business Card	8 cores, 16-GB memory	8 cores, 24-GB memory
General Document	8 cores, 12-GB memory	8 cores, 24-GB memory
ID Document	8 cores, 8-GB memory	8 cores, 24-GB memory
Invoice	8 cores, 16-GB memory	8 cores, 24-GB memory
Receipt	8 cores, 11-GB memory	8 cores, 24-GB memory
Custom Template	8 cores, 16-GB memory	8 cores, 24-GB memory

- Each core must be at least 2.6 gigahertz (GHz) or faster.
- Core and memory correspond to the `--cpus` and `--memory` settings, which are used as part of the `docker compose` or `docker run` command.

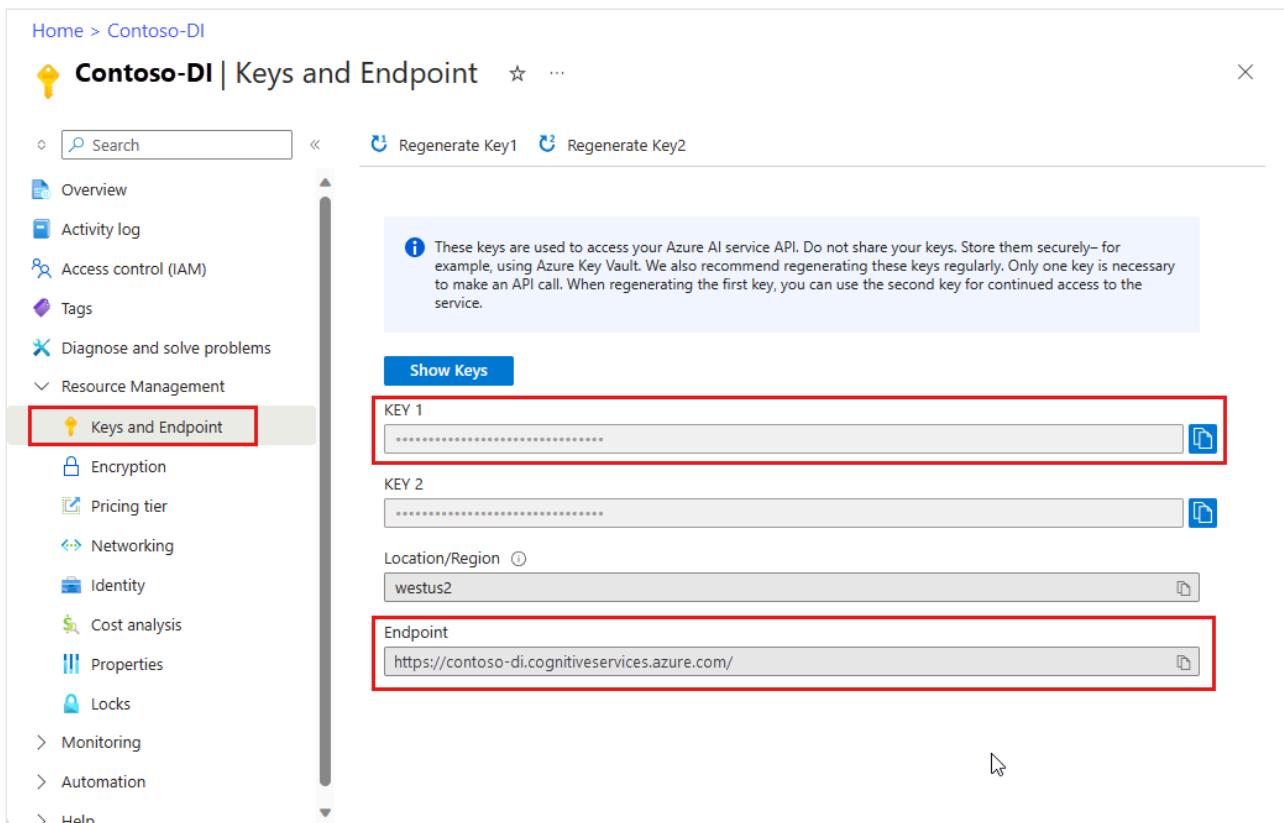
### Tip

You can use the [docker images](#) command to list your downloaded container images. For example, the following command lists the ID, repository, and tag of each downloaded container image, formatted as a table:

```
docker
docker images --format "table {{.ID}}\t{{.Repository}}\t{{.Tag}}"
IMAGE ID      REPOSITORY          TAG
<image-id>    <repository-path/name>  <tag-name>
```

## Run the container with the docker-compose up command

- Replace the {ENDPOINT\_URI} and {API\_KEY} values with your resource Endpoint URI and the key from the Azure resource page.



The screenshot shows the Azure portal interface for a resource named 'Contoso-DI'. The left sidebar shows various management options like Overview, Activity log, Access control (IAM), Tags, and Resource Management. Under Resource Management, the 'Keys and Endpoint' option is selected and highlighted with a red box. The main content area displays information about keys and endpoints. A callout box provides instructions: 'These keys are used to access your Azure AI service API. Do not share your keys. Store them securely—for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.' Below this, there are two input fields labeled 'KEY 1' and 'KEY 2', both of which are also highlighted with red boxes. Further down, there is a 'Location/Region' dropdown set to 'westus2' and an 'Endpoint' field containing the URL 'https://contoso-di.cognitiveservices.azure.com/'. There are also 'Regenerate Key1' and 'Regenerate Key2' buttons at the top right of the main content area.

- Ensure that the `EULA` value is set to `accept`.
- The `EULA`, `Billing`, and `ApiKey` values must be specified; otherwise the container can't start.

## Important

The keys are used to access your Document Intelligence resource. Don't share your keys. Store them securely, for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.

### Layout

The following code sample is a self-contained `docker compose` example to run the Document Intelligence Layout container. With `docker compose`, you use a YAML file to configure your application's services. Then, with `docker-compose up` command, you create and start all the services from your configuration. Enter `{FORM_RECOGNIZER_ENDPOINT_URI}` and `{FORM_RECOGNIZER_KEY}` values for your Layout container instance.

#### yml

```
version: "3.9"
services:
  azure-form-recognizer-layout:
    container_name: azure-form-recognizer-layout
    image: mcr.microsoft.com/azure-cognitive-services/form-recognizer/layout-4.0
    environment:
      - EULA=accept
      - billing={FORM_RECOGNIZER_ENDPOINT_URI}
      - apiKey={FORM_RECOGNIZER_KEY}
    ports:
      - "5000:5000"
    networks:
      - ocrvnet
networks:
  ocrvnet:
    driver: bridge
```

Now, you can start the service with the `docker compose` command:

#### Bash

```
docker-compose up
```

## Create a docker compose file

1. Name this file `docker-compose.yml`
2. The following code sample is a self-contained `docker compose` example to run Document Intelligence Layout, Studio, and Custom template containers together. With `docker compose`, you use a YAML file

to configure your application's services. Then, with `docker-compose up` command, you create and start all the services from your configuration.

yml

```
version: '3.3'
services:
  nginx:
    image: nginx:alpine
    container_name: reverseproxy
    depends_on:
      - layout
      - custom-template
    volumes:
      - ${NGINX_CONF_FILE}:/etc/nginx/nginx.conf
    ports:
      - "5000:5000"
  layout:
    container_name: azure-cognitive-service-layout
    image: mcr.microsoft.com/azure-cognitive-services/form-recognizer/layout-3.1:latest
    environment:
      eula: accept
      apikey: ${FORM_RECOGNIZER_KEY}
      billing: ${FORM_RECOGNIZER_ENDPOINT_URI}
      Logging:Console:LogLevel:Default: Information
      SharedRootFolder: /share
      Mounts:Shared: /share
      Mounts:Output: /logs
    volumes:
      - type: bind
        source: ${SHARED_MOUNT_PATH}
        target: /share
      - type: bind
        source: ${OUTPUT_MOUNT_PATH}
        target: /logs
    expose:
      - "5000"

  custom-template:
    container_name: azure-cognitive-service-custom-template
    image: mcr.microsoft.com/azure-cognitive-services/form-recognizer/custom-template-3.1:latest
    restart: always
    depends_on:
      - layout
    environment:
      AzureCognitiveServiceLayoutHost: http://azure-cognitive-service-layout:5000
      eula: accept
      apikey: ${FORM_RECOGNIZER_KEY}
      billing: ${FORM_RECOGNIZER_ENDPOINT_URI}
      Logging:Console:LogLevel:Default: Information
      SharedRootFolder: /share
      Mounts:Shared: /share
      Mounts:Output: /logs
    volumes:
      - type: bind
        source: ${SHARED_MOUNT_PATH}
        target: /share
      - type: bind
        source: ${OUTPUT_MOUNT_PATH}
```

```

    target: /logs
  expose:
    - "5000"

studio:
  container_name: form-recognizer-studio
  image: mcr.microsoft.com/azure-cognitive-services/form-recognizer/studio:3.1
  environment:
    ONPREM_LOCALFILE_BASEPATH: /onprem_folder
    STORAGE_DATABASE_CONNECTION_STRING: /onprem_db/Application.db
  volumes:
    - type: bind
      source: ${FILE_MOUNT_PATH} # path to your local folder
      target: /onprem_folder
    - type: bind
      source: ${DB_MOUNT_PATH} # path to your local folder
      target: /onprem_db
  ports:
    - "5001:5001"
  user: "1000:1000" # echo $(id -u):$(id -g)

```

The custom template container and Layout container can use Azure Storage queues or in memory queues. The `Storage:ObjectStore:AzureBlob:ConnectionString` and `queue:azure:connectionstring` environment variables only need to be set if you're using Azure Storage queues. When running locally, delete these variables.

## Ensure the service is running

To ensure that the service is up and running. Run these commands in an Ubuntu shell.

```
Bash

$cd <folder containing the docker-compose file>
$source .env
$docker-compose up
```

Custom template containers require a few different configurations and support other optional configurations.

[Expand table](#)

Setting	Required	Description
EULA	Yes	License acceptance Example: Eula=accept
Billing	Yes	Billing endpoint URI of the FR resource
ApiKey	Yes	The endpoint key of the FR resource
Queue:Azure:ConnectionString	No	Azure Queue connection string

Setting	Required	Description
Storage:ObjectStore:AzureBlob:ConnectionString	No	Azure Blob connection string
HealthCheck:MemoryUpperboundInMB	No	Memory threshold for reporting unhealthy to liveness. Default: Same as recommended memory
StorageTimeToLiveInMinutes	No	<code>TTL</code> duration to remove all intermediate and final files. Default: Two days, <code>TTL</code> can set between five minutes to seven days
Task:MaxRunningTimeSpanInMinutes	No	Maximum running time for treating request as time out. Default: 60 minutes
HTTP_PROXY_BYPASS_URLS	No	Specify URLs for bypassing proxy Example: <code>HTTP_PROXY_BYPASS_URLS = abc.com, xyz.com</code>
AzureCognitiveServiceReadHost (Receipt, IdDocument Containers Only)	Yes	Specify Read container uri Example: <code>AzureCognitiveServiceReadHost=http://onprem-fread:5000</code>
AzureCognitiveServiceLayoutHost (Document, Invoice Containers Only)	Yes	Specify Layout container uri Example: <code>AzureCognitiveServiceLayoutHost=http://onprem-frlayout:5000</code>

## Use the Document Intelligence Studio to train a model

- Gather a set of at least five forms of the same type. You use this data to train the model and test a form. You can use a [sample data set](#) (download and extract `sample_data.zip`).
- Once you can confirm that the containers are running, open a browser and navigate to the endpoint where you have the containers deployed. If this deployment is your local machine, the endpoint is `[http://localhost:5001](http://localhost:5001)`.
- Select the custom extraction model tile.
- Select the `Create project` option.
- Provide a project name and optionally a description
- On the "configure your resource" step, provide the endpoint to your custom template model. If you deployed the containers on your local machine, use this URL `[http://localhost:5000](http://localhost:5000)`.
- Provide a subfolder for where your training data is located within the files folder.
- Finally, create the project

You should now have a project created, ready for labeling. Upload your training data and get started labeling. If you're new to labeling, see [build and train a custom model](#).

## Using the API to train

If you plan to call the APIs directly to train a model, the custom template model train API requires a base64 encoded zip file that is the contents of your labeling project. You can omit the PDF or image files and submit only the JSON files.

Once you have your dataset labeled and \*.ocr.json, \*.labels.json and fields.json files added to a zip, use the PowerShell commands to generate the base64 encoded string.

#### PowerShell

```
$bytes = [System.IO.File]::ReadAllBytes("<your_zip_file>.zip")
$b64String = [System.Convert]::ToBase64String($bytes,
[System.Base64FormattingOptions]::None)
```

Use the build model API to post the request.

#### HTTP

```
POST http://localhost:5000/formrecognizer/documentModels:build?api-version=2023-07-31
```

```
{
  "modelId": "mymodel",
  "description": "test model",
  "buildMode": "template",

  "base64Source": "<Your base64 encoded string>",
  "tags": {
    "additionalProp1": "string",
    "additionalProp2": "string",
    "additionalProp3": "string"
  }
}
```

## Validate that the service is running

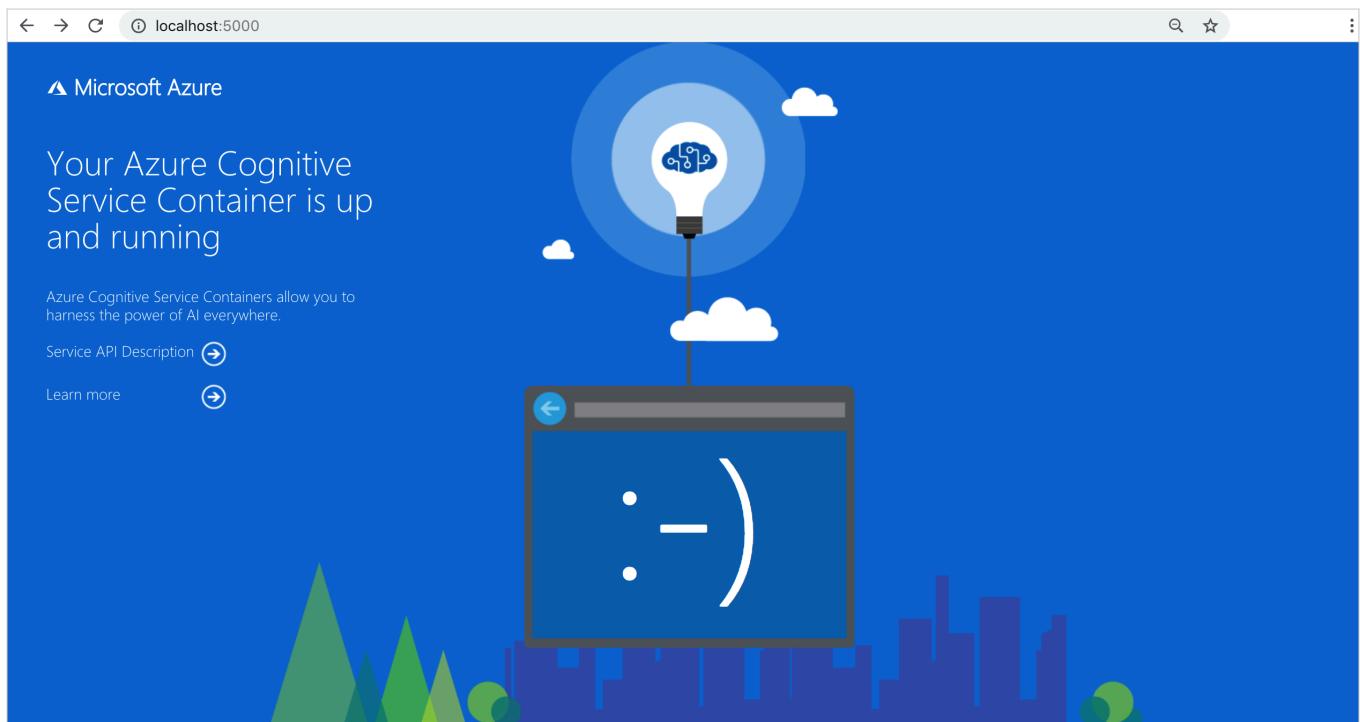
There are several ways to validate that the container is running:

- The container provides a homepage at `\` as a visual validation that the container is running.
- You can open your favorite web browser and navigate to the external IP address and exposed port of the container in question. Use the listed request URLs to validate the container is running. The listed example request URLs are `http://localhost:5000`, but your specific container can vary. Keep in mind that you're navigating to your container's **External IP address** and exposed port.

[ ] Expand table

Request URL	Purpose
<code>http://localhost:5000/</code>	The container provides a home page.

Request URL	Purpose
<a href="http://localhost:5000/ready">http://localhost:5000/ready</a>	Requested with GET, this request provides a verification that the container is ready to accept a query against the model. This request can be used for Kubernetes liveness and readiness probes.
<a href="http://localhost:5000/status">http://localhost:5000/status</a>	Requested with GET, this request verifies if the api-key used to start the container is valid without causing an endpoint query. This request can be used for Kubernetes liveness and readiness probes.
<a href="http://localhost:5000/swagger">http://localhost:5000/swagger</a>	The container provides a full set of documentation for the endpoints and a Try it out feature. With this feature, you can enter your settings into a web-based HTML form and make the query without having to write any code. After the query returns, an example CURL command is provided to demonstrate the required HTTP headers and body format.



## Stop the containers

To stop the containers, use the following command:

```
Console
docker-compose down
```

## Billing

The Document Intelligence containers send billing information to Azure by using a Document Intelligence resource on your Azure account.

Queries to the container are billed at the pricing tier of the Azure resource used for the API [Key](#). Billing is calculated for each container instance used to process your documents and images.

If you receive the following error: *Container isn't in a valid state. Subscription validation failed with status 'OutOfQuota'* API key is out of quota. It's an indicator that your containers aren't communicating with the billing endpoint.

## Connect to Azure

The container needs the billing argument values to run. These values allow the container to connect to the billing endpoint. The container reports usage about every 10 to 15 minutes. If the container doesn't connect to Azure within the allowed time window, the container continues to run, but doesn't serve queries until the billing endpoint is restored. The connection is attempted 10 times at the same time interval of 10 to 15 minutes. If it can't connect to the billing endpoint within the 10 tries, the container stops serving requests. See the [Azure AI container FAQ](#) for an example of the information sent to Microsoft for billing.

## Billing arguments

The [docker-compose up](#) command starts the container when all three of the following options are provided with valid values:

[Expand table](#)

Option	Description
<a href="#">ApiKey</a>	The key of the Azure AI services resource used to track billing information. The value of this option must be set to a key for the provisioned resource specified in <a href="#">Billing</a> .
<a href="#">Billing</a>	The endpoint of the Azure AI services resource used to track billing information. The value of this option must be set to the endpoint URL of a provisioned Azure resource.
<a href="#">Eula</a>	Indicates that you accepted the license for the container. The value of this option must be set to <code>accept</code> .

For more information about these options, see [Configure containers](#).

## Summary

That's it! In this article, you learned concepts and workflows for downloading, installing, and running Document Intelligence containers. In summary:

- Document Intelligence provides seven Linux containers for Docker.
- Container images are downloaded from mcr.
- Container images run in Docker.
- The billing information must be specified when you instantiate a container.

 **Important**

Azure AI containers aren't licensed to run without being connected to Azure for metering. Customers need to enable the containers to always communicate billing information with the metering service. Azure AI containers don't send customer data (for example, the image or text that is being analyzed) to Microsoft.

## Next steps

- [Document Intelligence container configuration settings](#)
- [Azure container instance recipe](#)

# Configure Document Intelligence containers

Article • 11/19/2024

Document Intelligence doesn't support containers for v4.0. Support for containers is currently available with Document Intelligence version [2022-08-31 \(GA\)](#) for all models and [2023-07-31 \(GA\)](#) for Read, Layout, Invoice, Receipt, and ID Document models:

- REST API [2022-08-31 \(GA\)](#)
- REST API [2023-07-31 \(GA\)](#)
- Client libraries targeting REST API [2022-08-31 \(GA\)](#)
- Client libraries targeting REST API [2023-07-31 \(GA\)](#)

✓ See [Configure Document Intelligence v3.0 containers](#) or [Configure Document Intelligence v3.1 containers](#) for supported versions of container documentation.

---

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

# Document Intelligence container tags

Article • 04/03/2025

This content applies to:  v4.0 (GA)

## Microsoft container registry (MCR)

Document Intelligence container images can be found within the [Microsoft Artifact Registry](#) (also know as Microsoft Container Registry(MCR))  , the primary registry for all Microsoft published container images.

The following containers support DocumentIntelligence v3.1 models and features:

 Expand table

Container name	image
<a href="#">Layout 4.0</a> 	<code>mcr.microsoft.com/azure-cognitive-services/form-recognizer/layout-4.0:latest</code>

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#)  | Get help at Microsoft Q&A

# Containers in disconnected (offline) environments

Article • 11/19/2024

Document Intelligence doesn't support containers for v4.0. Support for containers is currently available with Document Intelligence version [2022-08-31 \(GA\)](#) for all models and [2023-07-31 \(GA\)](#) for Read, Layout, Invoice, Receipt, and ID Document models:

- REST API [2022-08-31 \(GA\)](#)
- REST API [2023-07-31 \(GA\)](#)
- Client libraries targeting REST API [2022-08-31 \(GA\)](#)
- Client libraries targeting REST API [2023-07-31 \(GA\)](#)

✓ See [Document Intelligence v3.0 containers in disconnected environments](#) or [Document Intelligence v3.1 containers in disconnected environments](#) for supported versions of container documentation.

---

## Feedback

Was this page helpful?



[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Transparency note and use cases for Document Intelligence

Article • 03/12/2024

## What is a transparency note?

An AI system includes not only the technology, but also the people who will use it, the people who will be affected by it, and the environment in which it is deployed. Creating a system that is fit for its intended purpose requires an understanding of how the technology works, its capabilities and limitations, and how to achieve the best performance.

Microsoft provides *transparency notes* to help you understand how our AI technology works. This includes the choices system owners can make that influence system performance and behavior, and the importance of thinking about the whole system, including the technology, the people, and the environment. You can use transparency notes when developing or deploying your own system, or share them with the people who will use or be affected by your system.

Transparency notes are part of a broader effort at Microsoft to put our AI principles into practice. To find out more, see [Microsoft's AI principles](#).

## The basics of Document Intelligence

### Introduction

[Document Intelligence](#) is accessed via a set of APIs and allows developers to easily extract text, structure, and fields from their documents. It is composed of features like:

- Read for text extraction.
- Layout and General Documents for structural insights and general key-values and entities such as names, places, and things.
- Prebuilt models for specific document types like invoices, receipts, business cards, W2s, and IDs.
- Custom models for building models specific to your document types.

Document Intelligence supports one or more languages and locales for each of the features, as listed in the [Supported languages](#) article.

# Key terms

[+] [Expand table](#)

Term	Definition
Read	This feature extracts text lines, words, and their locations from images and documents, along with other information such as detected languages.
Layout	This feature extracts text, selection marks, and table structure (the row and column numbers associated with the text). See <a href="#">Document Intelligence Layout</a> .
General Documents	Analyze documents and associate values to keys and entries to tables that it discovers. For more information, see <a href="#">Document Intelligence General Documents</a> .
Prebuilt models	Prebuilt models are document-specific models for unique form types. These models don't require custom training before use. For example, the prebuilt invoice model extracts key fields from invoices. For more information, see <a href="#">Document Intelligence prebuilt invoice model</a> .
Custom models	Document Intelligence allows you to train a custom model that's tailored to your forms and documents. This model extracts text, key-value pairs, selection marks, and table data. Custom models can be improved with human feedback by applying human review, updating the labels, and retraining the model by using the API.
Confidence value	All Get Analysis Results operations return confidence values in the range between 0 and 1 for all extracted words and key-value mappings. This value represents the service's estimate of how many times it correctly extracts the word out of 100 or correctly maps the key-value pairs. For example, a word that's estimated to be extracted correctly 82% of the time results in a confidence value of 0.82.
Add-on features	Document Intelligence offers a set of add-on features to extend the results to include more elements from your documents. Some add-on features incur an extra cost and can be enabled and disabled depending on the scenario of the document extraction. We currently offer high resolution, formula, styleFont, barcodes, languages, keyValuePairs, and queryFields extraction capabilities. For more information, see <a href="#">Document Intelligence Add-on capabilities</a> .

# Capabilities

## System behavior

Azure AI Document Intelligence is a cloud-based Azure AI service that's built by using optical character recognition (OCR), Text Analytics, and Custom Text from Azure AI services. Custom models currently use Azure OpenAI service's GPT-3.5 model. OCR is used to extract typeface and handwritten text documents. Document Intelligence uses

OCR to detect and extract information from forms and documents supported by AI to provide more structure and information to the text extraction.

## Use cases

### Intended uses

Document Intelligence includes features that enable customers from various industries to extract data from their documents. The following scenarios are examples of appropriate use cases:

- **Accounts payable:** A company can increase the efficiency of its accounts payable clerks by using the prebuilt invoice model and custom forms to speed up invoice data entry with a human in the loop. The prebuilt invoice model can extract key fields, such as *Invoice Total* and *Shipping Address*.
- **Insurance form processing:** A customer can train a model by using custom forms to extract a key-value pair in insurance forms and then feeds the data to their business flow to improve the accuracy and efficiency of their process. For their unique forms, customers can build their own model that extracts key values by using custom forms. These extracted values then become actionable data for various workflows within their business.
- **Bank form processing:** A bank can use the prebuilt ID model and custom forms to speed up the data entry for "know your customer" documentation, or to speed up data entry for a mortgage packet. If a bank requires their customers to submit personal identification as part of a process, the prebuilt ID model can extract key values, such as *Name* and *Document Number*, speeding up the overall time for data entry.
- **Robotic process automation (RPA):** Using the custom extraction model, customers can extract specific data needed from various types of documents. The key-value pair extracted can then be entered into various systems such as databases, or CRM systems, through RPA, replacing manual data entry. Customers can also use custom classification model to categorize documents based on their content and file them in proper location. As such, an organized set of data extracted from the custom model can be an essential first step to document RPA scenarios for businesses that handle large volumes of documents regularly.

### Considerations when choosing other use cases

Consider the following factors when you choose a use case:

- **Carefully consider applying human review when sensitive data or scenarios are involved:** It's important to include a human in the loop for a manual review when you're dealing with high-stakes scenarios (e.g. affecting someone's consequential rights) or sensitive data. Machine learning models aren't perfect. Consider carefully when to include a manual review step for certain workflows. For example, identity verification at a port of entry such as airports should include human oversight.
- **Carefully consider when using for awarding or denying of benefits:** Document Intelligence was not designed or evaluated for the award or denial of benefits, and use in these scenarios may have unintended consequences. These scenarios include:
  - **Medical insurance:** This would include using healthcare records and medical prescriptions as the basis for decisions on insurance reward or denial.
  - **Loan approvals:** These include applications for new loans or refinancing of existing ones.
- **Carefully consider the supported document types and locales:** Prebuilt models have a predefined list of supported fields and are built for specific locales. Be sure to carefully check the officially supported locales and document types to ensure the best results. For example, see [Document Intelligence prebuilt receipt locales](#).
- **Legal and regulatory considerations:** Organizations need to evaluate potential specific legal and regulatory obligations when using any AI services and solutions, which may not be appropriate for use in every industry or scenario. Additionally, AI services or solutions are not designed for and may not be used in ways prohibited in applicable terms of service and relevant codes of conduct.

## Limitations

### Technical limitations, operational factors, and ranges

#### Prebuilt model limitations

Document Intelligence prebuilt models are used for processing specific document types and are pretrained on thousands of forms. This capability allows developers to get started and get results within minutes, with no training data or labeling required. For prebuilt models, it's important to note the list of input requirements, supported document types, and locales for each prebuilt model for optimal results. For example, refer to the prebuilt Invoice input requirements.

## Custom model limitations

Document Intelligence custom models are trained using your own training data so that the model can train to your specific forms and documents. This capability is heavily dependent on the way you label the data, as well as the type of training data set you provide. For custom models, it's important to note the limits of training data set size, document page limits, and minimum number of samples needed for each type of document. Custom models currently use Azure OpenAI Service's GPT-3.5 model. Further information on the Azure OpenAI models can be found in the [Azure OpenAI Transparency Note](#).

The [Service limits](#) page contains more information on Azure AI Document Intelligence service quotas and limits for all pricing tiers. It also contains model limitations and best practices for model usage and avoiding request throttling.

## Feature support

See the [Analysis features table](#) for a list of the different operations that Document Intelligence models can perform.

## System performance

### Accuracy

Text is composed of lines and words at the foundational level and entities such as names, prices, amounts, company names, and products at the document understanding level.

### Word-level accuracy

A popular measure of accuracy for OCR is word error rate (WER), or how many words were incorrectly output in the extracted results. The lower the WER, the higher the accuracy.

WER is defined as:

$$WER = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C}$$

Where:

Term	Definition	Example
S	Count of incorrect words ("substituted") in the output.	"Velvet" gets extracted as "Veivet" because "l" is detected as "i."
D	Count of missing ("deleted") words in the output.	For the text "Company Name: Microsoft," Microsoft isn't extracted because it's handwritten or hard to read.
I	Count of nonexistent ("inserted") words in the output.	"Department" gets incorrectly segmented into three words as "Dep artm ent." In this case, the result is one deleted word and three inserted words.
C	Count of correctly extracted words in the output.	All words that are correctly extracted.
N	Count of total words in the reference ( $N=S+D+C$ ) excluding I because those words were missing from the original reference and were incorrectly predicted as present.	Consider an image with the sentence, "Microsoft, headquartered in Redmond, WA announced a new product called Velvet for finance departments." Assume the OCR output is ", headquartered in Redmond, WA announced a new product called Veivet for finance dep artm ents." In this case, S (Velvet) = 1, D (Microsoft) = 1, I (dep artm ents) = 3, C (11), and N = S + D + C = 13. Therefore, WER = $(S + D + I) / N = 5 / 13 = 0.38$ or 38% (out of 100).

## Using a confidence value

As covered in an earlier section, the service provides a confidence value for each predicted word in the OCR output. Customers use this value to calibrate custom thresholds for their content and scenarios to route the content for straight-through processing or forwarding to the human-in-the-loop process. The resulting measurements determine the scenario-specific accuracy.

OCR system performance implications can vary by scenarios where the OCR technology is applied. We'll review a few examples to illustrate that concept.

- **Medical device compliance:** In this first example, a multinational pharmaceutical company with a diverse product portfolio of patents, devices, medications, and treatments needs to analyze FDA-compliant product label information and analysis results documents. The company might prefer a low confidence value threshold for applying human-in-the-loop because the cost of incorrectly extracted data can have significant impact for consumers and fines from regulatory agencies.
- **Image and documents processing:** In this second example, a company performs insurance and loan application processing. The customer using OCR might prefer a medium confidence value threshold because the automated text extraction is

combined downstream with other information inputs and human-in-the-loop steps for a holistic review of applications.

- **Content moderation:** For a large volume of e-commerce catalog data imported from suppliers at scale, the customer might prefer a high confidence value threshold with high accuracy because even a small percentage of falsely flagged content can generate a lot of overhead for their human review teams and suppliers.

## Document and entity-level accuracy

At the document level, for example, in the case of an invoice or receipt, an error of only one character in the entire document might be rated insignificant. But if that error is in the text that represents the paid amount, the entire invoice or receipt might get flagged as incorrect.

Another useful metric is the entity error rate (EER). It's the percentage of incorrectly extracted entities, such as names, prices, amounts, and phone numbers, out of the total number of the corresponding entities in one or more documents. For example, for a total of 30 words representing 10 names, 2 incorrect words out of 30 equals 0.06 (6%) WER. But if that results in 2 names out of 10 as incorrect, the Name EER is 0.20 (20%), which is much higher than the WER.

Measuring both WER and EER is a useful exercise to get a full perspective on document understanding accuracy.

## Best practices for improving system performance

Consider the following points about limitations and performance:

The service supports images and documents. For the allowable limits for number of pages, image sizes, paper sizes, and file sizes, see [What is Document Intelligence?](#).

- Many variables can affect the accuracy of the OCR results upon which Document Intelligence depends. These variables include document scan quality, resolution, contrast, light conditions, rotation, and text attributes such as size, color, and density. For example, we recommend that the image be at least 50 x 50 pixels. Refer to the product specifications and test the service on your documents to validate the fit for your situation.
- Note the limitations of each service with regard to currently supported inputs, languages and locales, and document types. For example, refer to the [Layout supported languages](#).

## Best practices to improve custom model quality

When you're using the Document Intelligence custom model, you provide your own training data so that the model can train to your specific forms and documents. The following list uses the custom form model type to share starter tips for improving your model quality.

- For filled-in forms, use examples that have all of their fields filled in.
- Use forms with real-world values that you expect to see for each field.
- If your form images are of lower quality, use a larger data set (at least 10-15 images, for example).

For a full guide and input requirements, see [Build a training data set for a custom model](#).

## Evaluation of Document Intelligence

Document Intelligence's performance will vary depending on the real-world solutions for which it's implemented. To ensure optimal performance in their scenarios, customers should conduct their own evaluations. The service provides a confidence value in the range between 0 and 1 for each extracted word and key-value mapping. Customers should run a pilot or a proof of concept representing their use case to understand the range of confidence values and the extraction quality from Document Intelligence. They can then estimate the confidence value thresholds for the results to be either sent for straight-through processing (STP) or reviewed by a human. For example, the customer might submit results with confidence values greater than or equal to .80 for straight-through processing and apply human review to results with confidence values less than .80.

## Evaluating and integrating Document Intelligence for your use

Microsoft wants to help you responsibly develop and deploy solutions that use Document Intelligence. We're taking a principled approach to upholding personal agency and dignity by considering the AI systems' fairness, reliability and safety, privacy and security, inclusiveness, transparency, and human accountability. These considerations are in line with our commitment to developing Responsible AI.

When you're getting ready to deploy AI-powered products or features, the following activities help to set you up for success:

- **Understand what it can do:** Fully assess the potential of Document Intelligence to understand its capabilities and limitations. Understand how it will perform in your particular scenario and context. For example, if you're using the prebuilt invoice model, test with real-world invoices from your business processes to analyze and benchmark the results against your existing process metrics.
- **Respect an individual's right to privacy:** Only collect data and information from individuals for lawful and justifiable purposes. Only use data and information that you have consent to use for this purpose.
- **Legal review:** Obtain appropriate legal review, particularly if you plan to use it in sensitive or high-risk applications. Understand what restrictions you might need to work within, and your responsibility to resolve any issues that might come up in the future.
- **Human-in-the-loop:** Keep a human in the loop, and include human oversight as a consistent pattern area to explore. This means ensuring constant human oversight of the AI-powered product or feature and to maintain the role of humans in decision-making. Ensure that you can have real-time human intervention in the solution to prevent harm. A human in the loop enables you to manage situations when Document Intelligence does not perform as required.
- **Security:** Ensure your solution is secure and that it has adequate controls to preserve the integrity of your content and prevent unauthorized access.

## Recommendations for preserving privacy

A successful privacy approach empowers individuals with information and provides controls and protection to preserve their privacy.

- If Document Intelligence is part of a solution designed to incorporate personally identifiable information (PII), think carefully about whether and how to record that data. Follow applicable national and regional regulations on privacy and sensitive data.
- Privacy managers should consider the retention policies on the extracted text and values, and the underlying documents or images of those documents. The retention policies will be tied to the intended use of each application.

## Learn more about responsible AI

- [Microsoft AI principles ↗](#)
- [Microsoft responsible AI resources ↗](#)

- Microsoft Azure Learning courses on responsible AI

## Learn more about Document Intelligence

- Document Intelligence overview
- Data, privacy, and security for Document Intelligence

# Data, privacy, and security for Document Intelligence

Article • 07/18/2023

This article provides details regarding how Document Intelligence processes your data. Document Intelligence is designed with compliance, privacy, and security in mind. However, you are responsible for its use and the implementation of this technology. It's your responsibility to comply with all applicable laws and regulations in your jurisdiction.

## How does Document Intelligence process data?

### Authenticate (with subscription or API keys)

The most common way to authenticate access to Document Intelligence is by using the customer's Document Intelligence API key. Each request to the service URL must include an authentication header. This header passes along an API key (or token if applicable), which is used to validate your subscription for a service or group of services. For more information, see [Authenticate requests to Azure AI services](#).

### Secure data in transit (for scanning)

All Azure AI services endpoints, including the Document Intelligence API URLs, use HTTPS URLs for encrypting data during transit. The client operating system needs to support Transport Layer Security (TLS) 1.2 for calling the endpoints. For more information, see [Azure AI services security](#).

### Encrypts input data for processing

The incoming data is processed in the same region where the Document Intelligence resource was created. When you submit your documents to a Document Intelligence operation, it starts the process of analyzing the document to extract all text and identify structure and key values in a document. Your data and results are then temporarily encrypted and stored in Azure Storage.

### Retrieve the results

The "Get Analyze Results" operation is authenticated against the same API key that was used to call the "Analyze" operation to ensure no other customer can access your data.

It returns the analysis job completion status, When the status shows as completed, the operation also returns the extracted results in JSON format.

## Data stored by Document Intelligence

**For all analysis:** To facilitate asynchronous analysis and checking the completion status and returning the extracted results to the customer upon completion, the data and extracted results are stored temporarily in Azure Storage in the same region. All customers in the same region share the temporary storage. The customer's data is logically isolated from other customers with their Azure subscription and API credentials.

**For customer trained models:** The Custom model feature allows customers to build custom models from training data stored in customer's Azure blob storage locations. The interim outputs after analysis and labeling are stored in the same location. The trained custom models are stored in Azure storage in the same region and logically isolated with their Azure subscription and API credentials.

**Deletes data:** For all features, the input data and results are deleted within 24 hours and not used for any other purpose. For customer trained models, the customers can delete their models and associated metadata at any time by using the API.

To learn more about privacy and security commitments, see the [Microsoft Trust Center](#).

# Create a Document Intelligence Logic Apps workflow

Article • 11/19/2024

This content applies to:  v4.0 (GA) | Previous versions:  v3.1 (GA)  v3.0 (GA)

## Important

This tutorial and the Logic App Document intelligence connector targets Document intelligence REST API v3.0 and forward.

Azure Logic Apps is a cloud-based platform that can be used to automate workflows without writing a single line of code. The platform enables you to easily integrate Microsoft and your applications with your apps, data, services, and systems. A Logic App is the Azure resource you create when you want to develop a workflow. Here are a few examples of what you can do with a Logic App:

- Create business processes and workflows visually.
- Integrate workflows with software as a service (SaaS) and enterprise applications.
- Automate enterprise application integration (EAI), business-to-business (B2B), and electronic data interchange (EDI) tasks.

For more information, see [Logic Apps Overview](#).

In this tutorial, we show you how to build a Logic App connector flow to automate the following tasks:

- ✓ Detect when an invoice has been added to a OneDrive folder.
- ✓ Process the invoice using the Document Intelligence prebuilt-invoice model.
- ✓ Send the extracted information from the invoice to a pre-specified email address.

Choose a workflow using a file from either your Microsoft OneDrive account or Microsoft ShareDrive site:

## Prerequisites

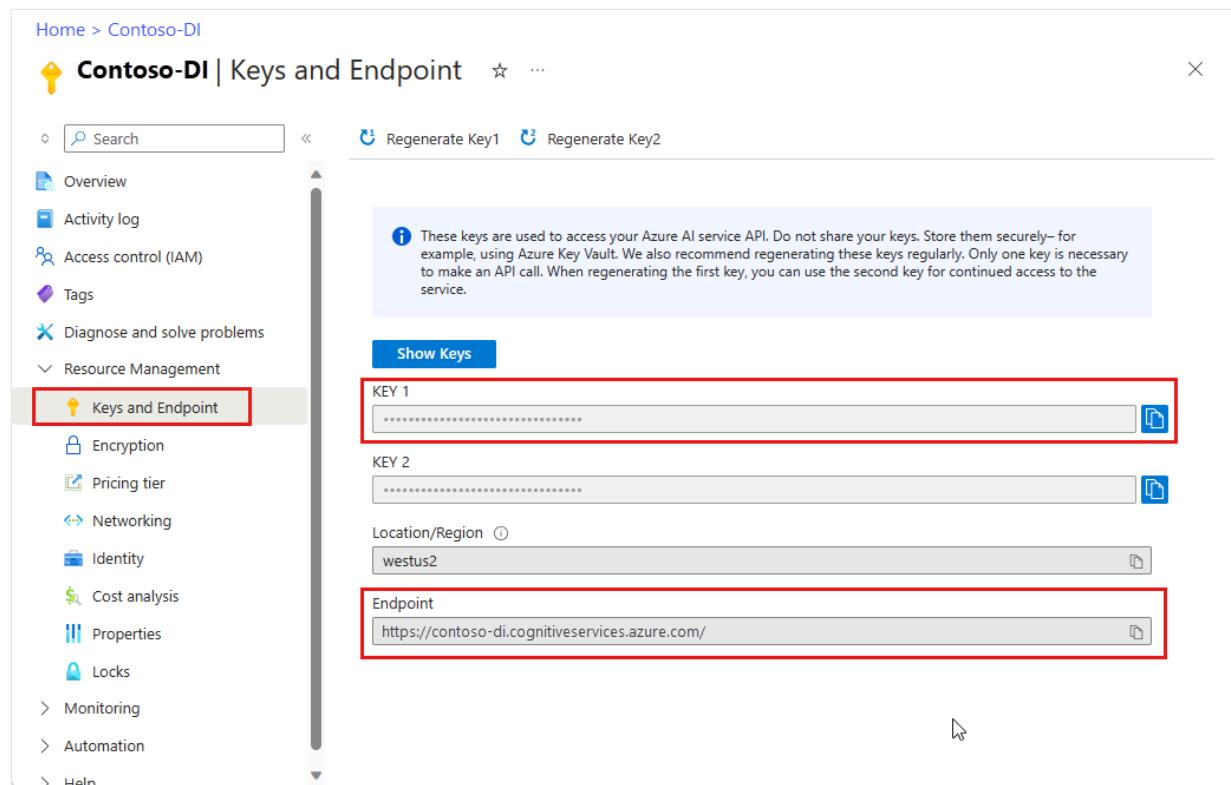
To complete this tutorial, you need the following resources:

- An Azure subscription. You can [create a free Azure subscription](#) ↗
- A free [OneDrive](#) ↗ or [OneDrive for Business](#) ↗ cloud storage account.

### ⚠ Note

- OneDrive is intended for personal storage.
- OneDrive for Business is part of Office 365 and is designed for organizations. It provides cloud storage where you can store, share, and sync all work files.

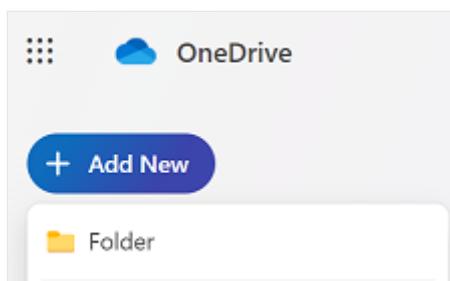
- A free [Outlook online](#) or [Office 365](#) email account\*\*.
- A sample invoice to test your Logic App. You can download and use our [sample invoice document](#) for this tutorial.
- A Document Intelligence resource. Once you have your Azure subscription, [create a Document Intelligence resource](#) in the Azure portal to get your key and endpoint. If you have an existing Document Intelligence resource, navigate directly to your resource page. You can use the free pricing tier (F0) to try the service, and upgrade later to a paid tier for production.
  - After the resource deploys, select **Go to resource**. Copy the **Keys and Endpoint** values from your resource in the Azure portal and paste them in a convenient location, such as *Microsoft Notepad*. You need the key and endpoint values to connect your application to the Document Intelligence API. For more information, see [create a Document Intelligence resource](#).



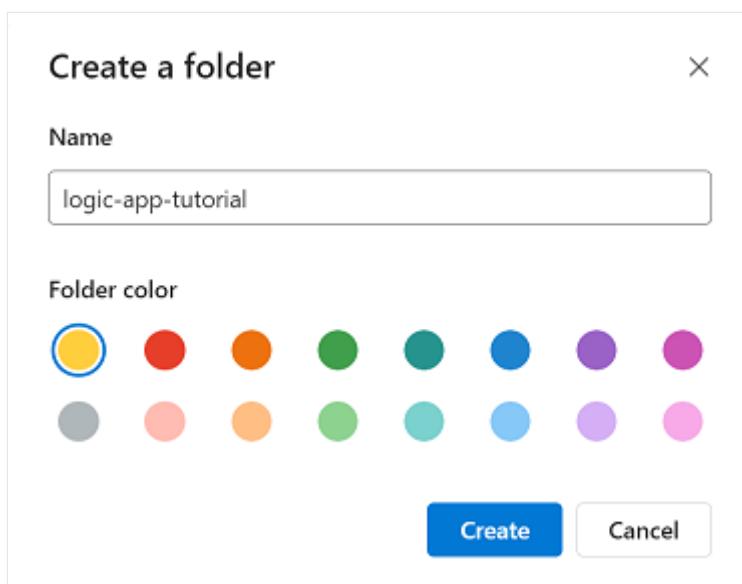
## Create a OneDrive folder

Before we jump into creating the Logic App, we have to set up a OneDrive folder.

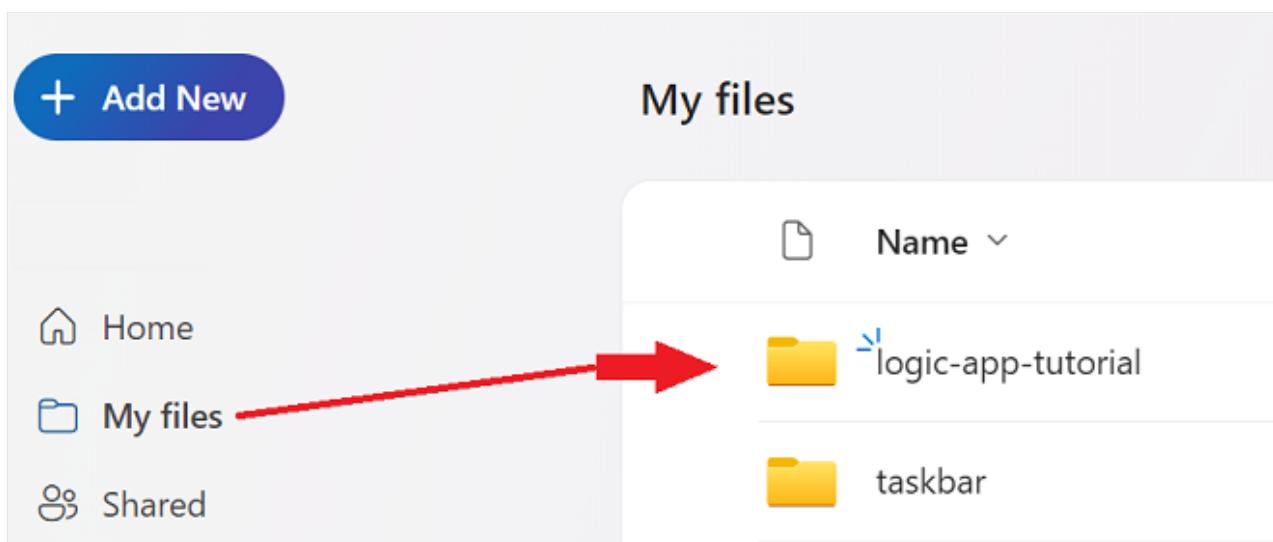
1. Sign in to your [OneDrive](#) or [OneDrive for Business](#) home page.
2. Select the **+ Add New** button in the upper-left corner sidebar and select **Folder**.



3. Enter a name for your new folder and select **Create**.



4. You see the new folder in your files.

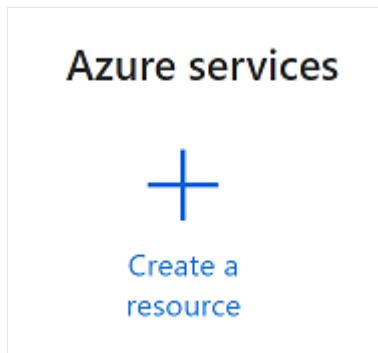


5. We're done with OneDrive for now.

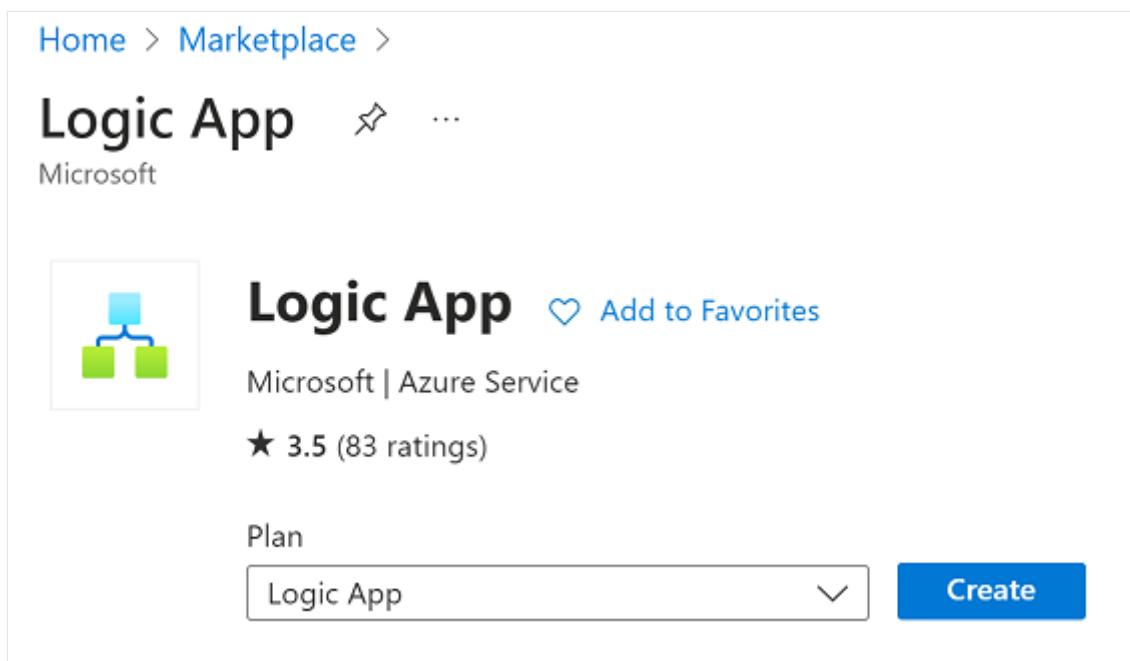
# Create a Logic App resource

At this point, you should have a Document Intelligence resource and a OneDrive folder all set. Now, it's time to create a Logic App resource.

1. Navigate to the [Azure portal ↗](#).
2. Select  **Create a resource** from the Azure home page.



3. Search for and choose **Logic App** from the search bar.
4. Select the create button



A screenshot of the Azure Marketplace. The URL in the address bar is "Home > Marketplace >". The search results show a listing for "Logic App" by Microsoft. The listing includes a thumbnail icon of a tree, the name "Logic App", a "Microsoft | Azure Service" badge, a "3.5 (83 ratings)" rating, and a "Plan" section with a dropdown menu set to "Logic App" and a "Create" button.

5. Next, you're going to fill out the **Create Logic App** fields with the following values:

- **Subscription**. Select your current subscription.
- **Resource group**. The [Azure resource group](#) that contains your resource. Choose the same resource group you have for your Document Intelligence resource.
- **Type**. Select **Consumption**. The Consumption resource type runs in global, multi-tenant Azure Logic Apps and uses the [Consumption billing model](#).

- **Logic App name.** Enter a name for your resource. We recommend using a descriptive name, for example *YourNameLogicApp*.
- **Publish.** Select **Workflow**.
- **Region.** Select your local region.
- **Enable log analytics.** For this project, select **No**.
- **Plan Type.** Select **Consumption**. The Consumption resource type runs in global, multi-tenant Azure Logic Apps and uses the [Consumption billing model](#).
- **Zone Redundancy.** Select **disabled**.

6. When you're done, you have something similar to the following image (Resource group, Logic App name, and Region may be different). After checking these values, select **Review + create** in the bottom-left corner.

Home > Create a resource > Logic App >

## Create Logic App

Create a logic app, which lets you group workflows as a logical unit for easier management, deployment and sharing of resources. Workflows let you connect your business-critical apps and services with Azure Logic Apps, automating your workflows without writing a single line of code.

### Project Details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	<input type="text" value="your-subscription"/>
Resource Group *	<input type="text" value="your-resource-group"/> <a href="#">Create new</a>

### Instance Details

Type \*

<input checked="" type="radio"/> Consumption	<input type="radio"/> Standard
<a href="#">Looking for the classic consumption create experience? Click here</a>	

Logic App name \*

Region \*

Enable log analytics \*

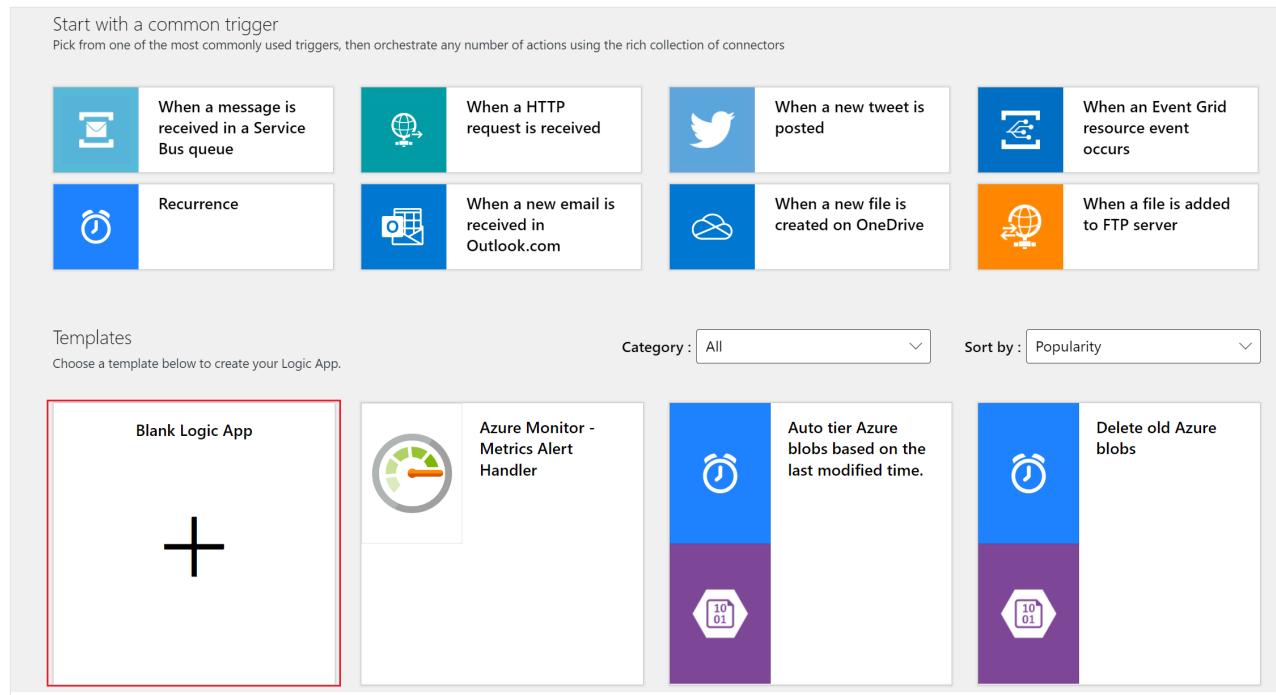
<input type="radio"/> Yes	<input checked="" type="radio"/> No
---------------------------	-------------------------------------

**Review + create** [< Previous](#) [Next : Tags >](#)

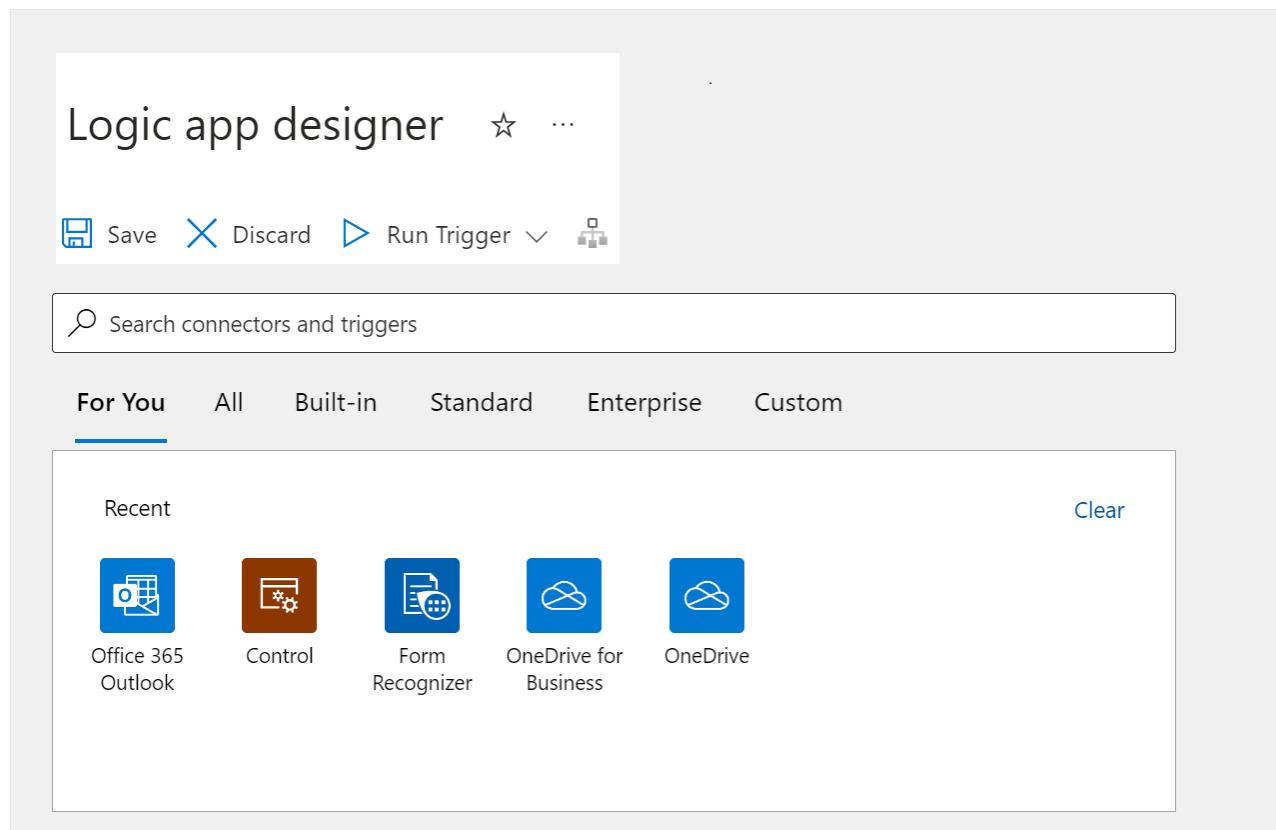
7. A short validation check runs. After it completes successfully, select **Create** in the bottom-left corner.
8. Next, you're redirected to a screen that says **Deployment in progress**. Give Azure some time to deploy; it can take a few minutes. After the deployment is complete, you see a

banner that says, **Your deployment is complete**. When you reach this screen, select **Go to resource**.

9. Finally, you're redirected to the **Logic Apps Designer** page. There's a short video for a quick introduction to Logic Apps available on the home screen. When you're ready to begin designing your Logic App, select the **Blank Logic App** button from the **Templates** section.



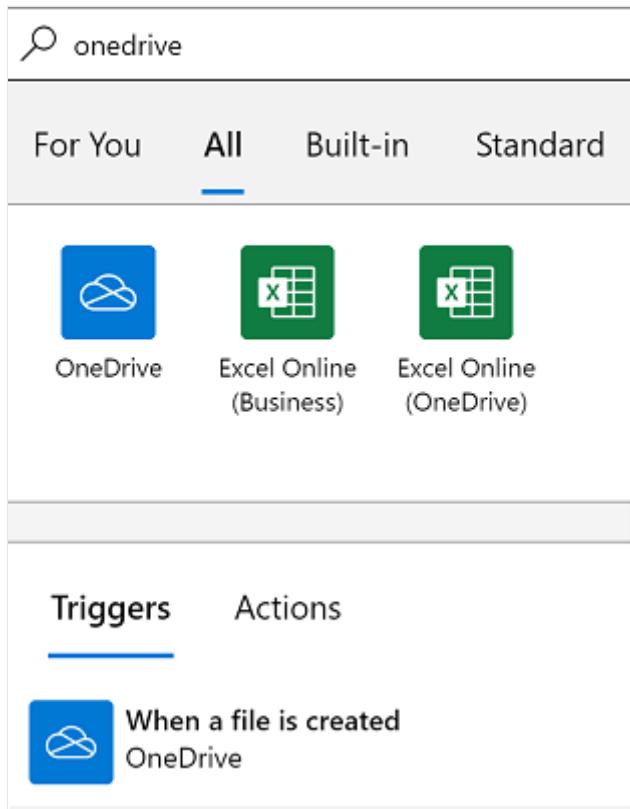
10. You see a screen that looks similar to the following image. Now, you're ready to start designing and implementing your Logic App.



# Create an automation flow

Now that you have the Logic App connector resource set up and configured, let's create the automation flow and test it out!

1. Search for and select OneDrive or OneDrive for Business in the search bar. Then, select the When a file is created trigger.



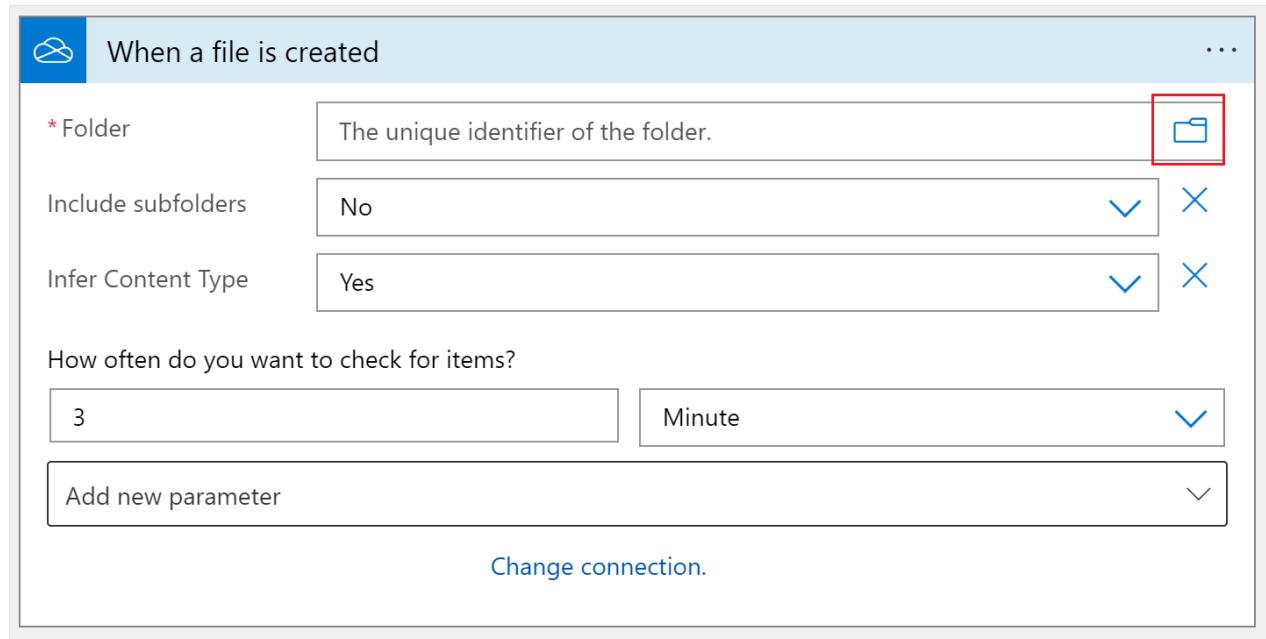
2. Next, a pop-up window appears, prompting you to log into your OneDrive account. Select **Sign in** and follow the prompts to connect your account.

## 💡 Tip

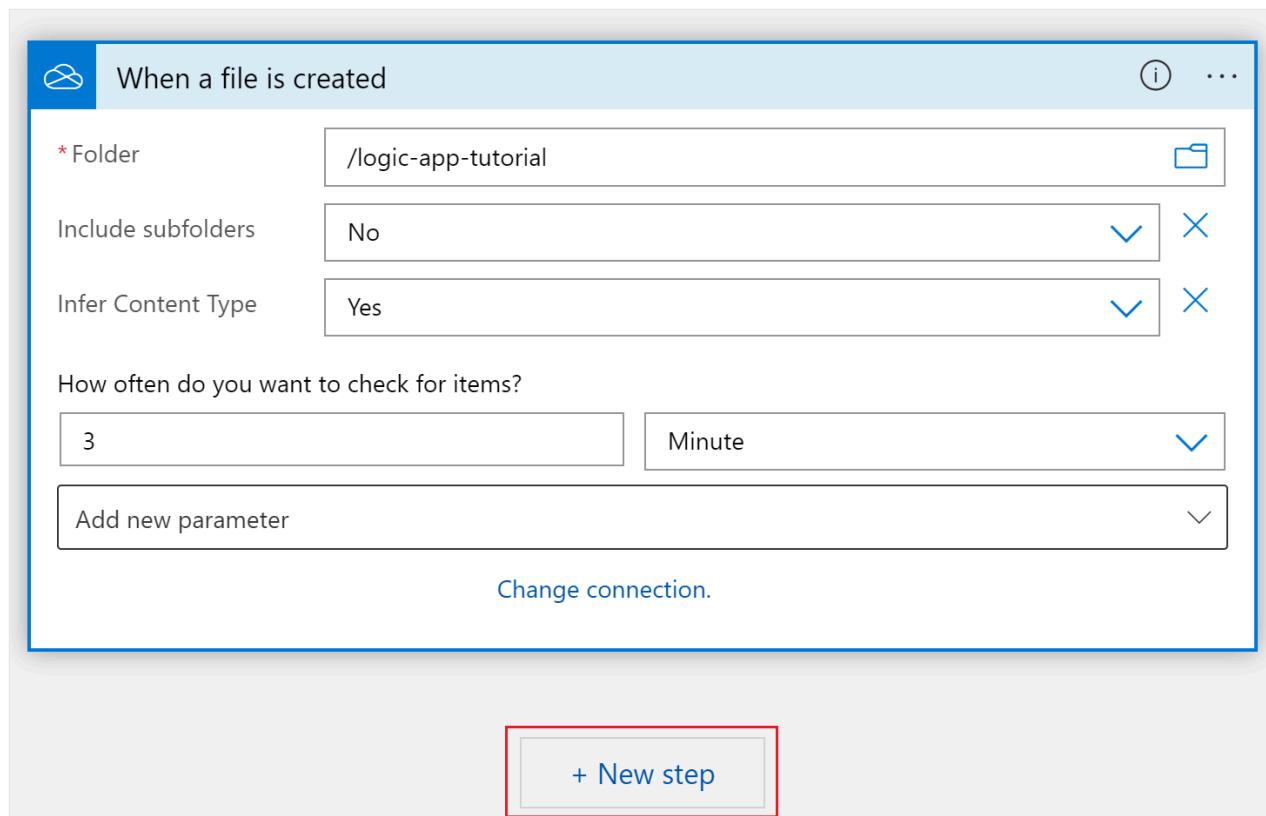
If you try to sign into the OneDrive connector using an Office 365 account, you may receive the following error: ***Sorry, we can't sign you in here with your @MICROSOFT.COM account.***

- This error happens because OneDrive is a cloud-based storage for personal use that can be accessed with an Outlook.com or Microsoft Live account not with Office 365 account.
- You can use the OneDrive for Business connector if you want to use an Office 365 account. Make sure that you have [created a OneDrive Folder](#) for this project in your OneDrive for Business account.

3. After your account is connected, select the folder you created earlier in your **OneDrive** or **OneDrive for Business** account. Leave the other default values in place.



4. Next, we're going to add a new step to the workflow. Select the **+ New step** button underneath the newly created OneDrive node.



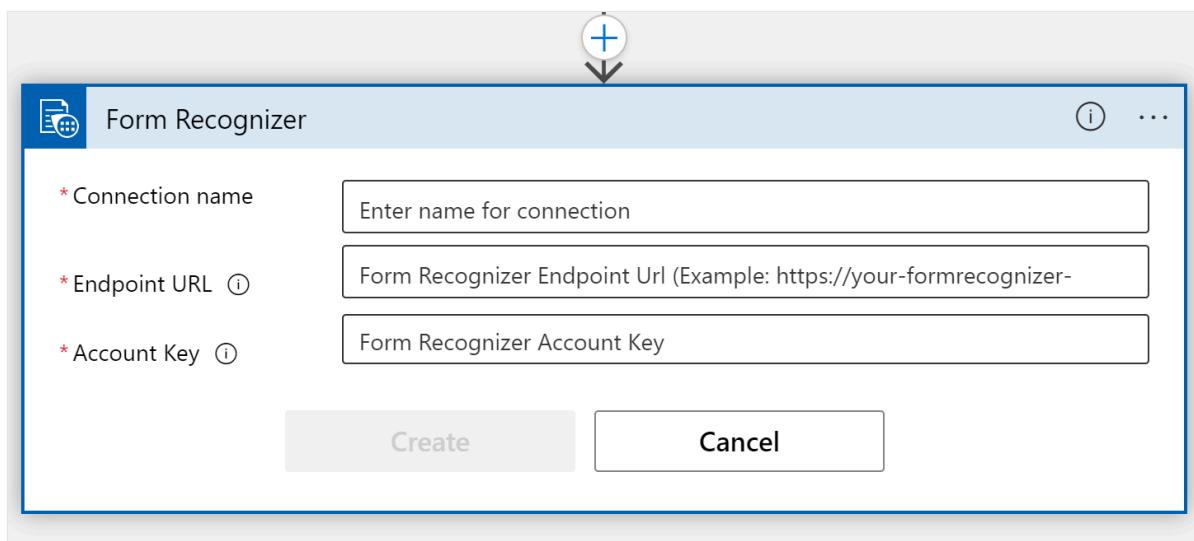
5. A new node is added to the Logic App designer view. Search for **Form Recognizer** (Document Intelligence forthcoming) in the **Choose an operation** search bar and select **Analyze Document for Prebuilt or Custom models (v3.0 API)** from the list.

The screenshot shows the Logic Apps Designer Studio interface. At the top, there are two tabs: 'Triggers' and 'Actions'. The 'Actions' tab is selected, indicated by a blue underline. Below the tabs, a list of actions is displayed in a table format. Each row contains an icon, the action name, a brief description, and an information button (i). The 'Analyze Document for Prebuilt or Custom models (v3.0 API)' action is highlighted with a red rectangular border around its row.

	Actions	
	Analyze Business Card	<a href="#">Form Recognizer</a> <a href="#">i</a>
	Analyze Custom Form	<a href="#">Form Recognizer</a> <a href="#">i</a>
	Analyze Document for Prebuilt or Custom models (v3.0 API)	<a href="#">Form Recognizer</a> <a href="#">i</a>
	Analyze ID Document	<a href="#">Form Recognizer</a> <a href="#">i</a>
	Analyze Invoice	<a href="#">Form Recognizer</a> <a href="#">i</a>
	Analyze Layout	<a href="#">Form Recognizer</a> <a href="#">i</a>

6. Now, you see a window to create your connection. Specifically, you're going to connect your Document Intelligence resource to the Logic Apps Designer Studio:

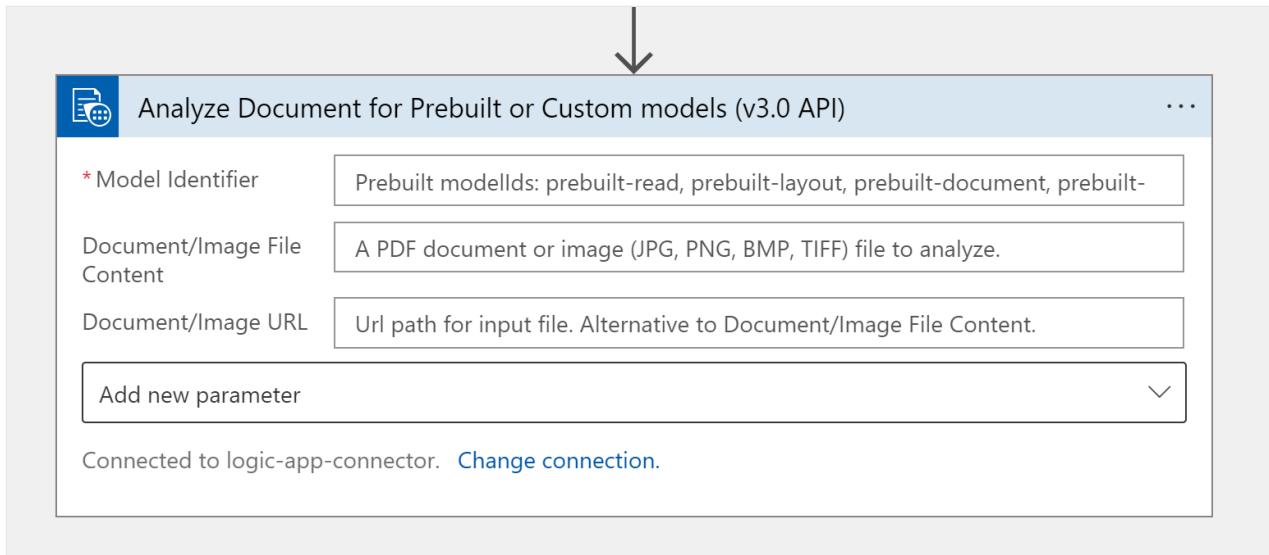
- Enter a **Connection name**. It should be something easy to remember.
- Enter the Document Intelligence resource **Endpoint URL** and **Account Key** that you copied previously. If you skipped this step earlier or lost the strings, you can navigate back to your Document Intelligence resource and copy them again. When you're done, select **Create**.



**Note**

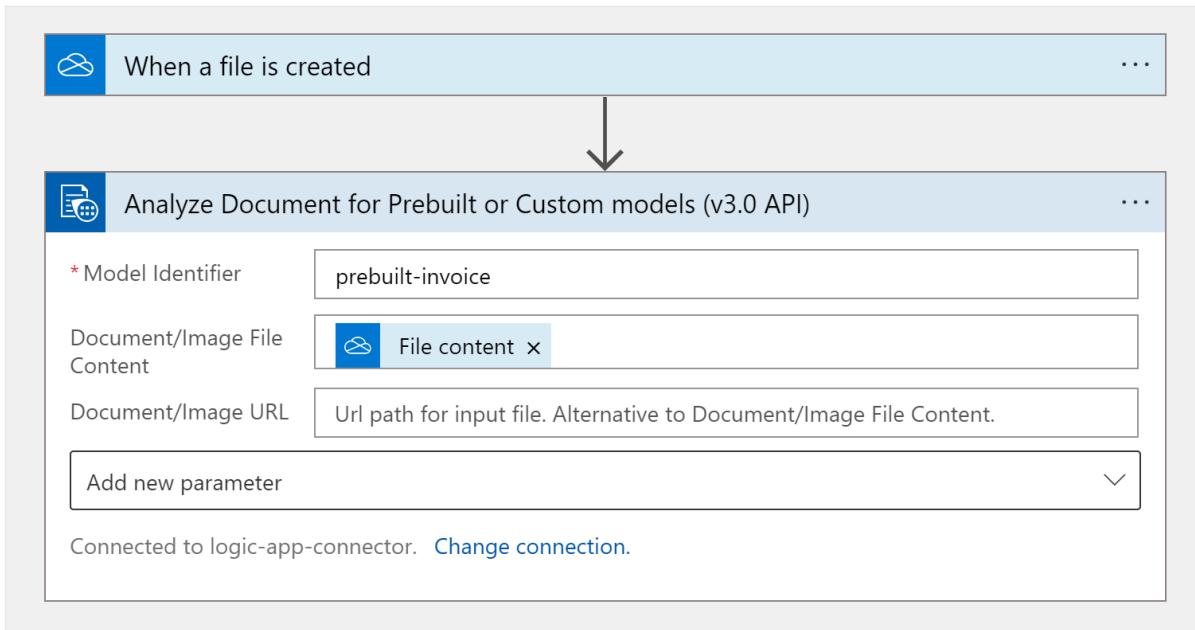
If you already logged in with your credentials, the prior step is skipped.

7. Next, you see the selection parameters window for the **Analyze Document for Prebuilt or Custom Models (v3.0 API)** connector.



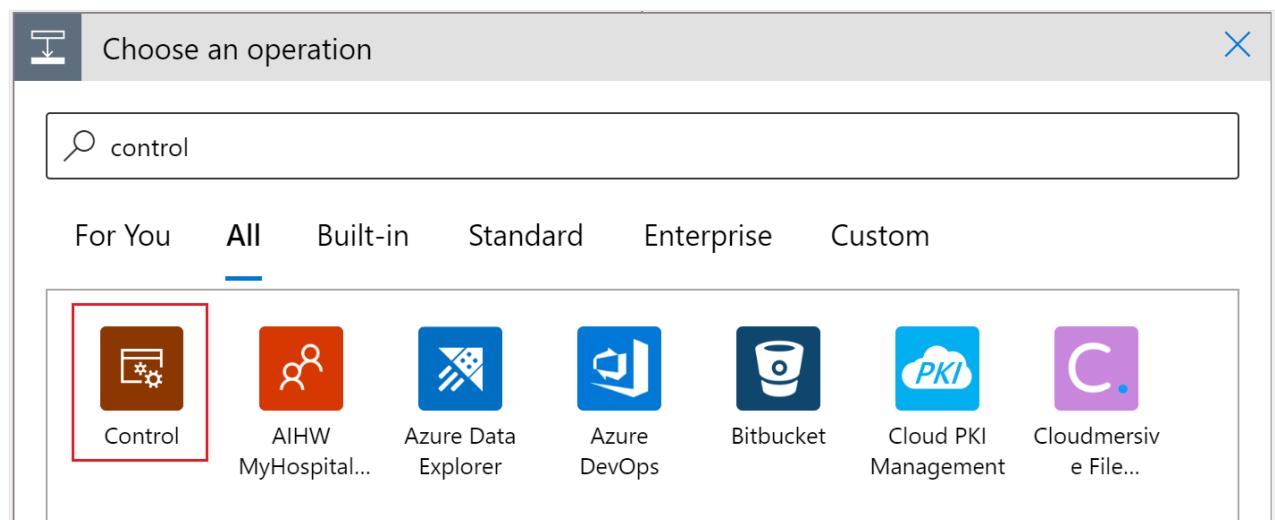
8. Complete the fields as follows:

- **Model Identifier.** Specify which model you want to call, in this case we're calling the prebuilt invoice model, so enter **prebuilt-invoice**.
- **Document/Image File Content.** Select this field. A dynamic content pop-up appears. If it doesn't, select the **Add dynamic content** button below the field and choose **File content**. This step is essentially sending the file(s) to be analyzed to the Document Intelligence prebuilt-invoice model. Once you see the **File content** badge show in the **Document /Image file content** field, you've completed this step correctly.
- **Document/Image URL.** Skip this field for this project because we're already pointing to the file content directly from the OneDrive folder.
- **Add new parameter.** Skip this field for this project.



9. We need to add a few more steps. Once again, select the **+ New step** button to add another action.

10. In the **Choose an operation** search bar, enter *Control* and select the **Control** tile.



11. Scroll down and select the **For each Control** tile from the **Control** list.

The screenshot shows the 'Actions' tab in the Microsoft Power Automate interface. It lists three actions under the 'Control' category:

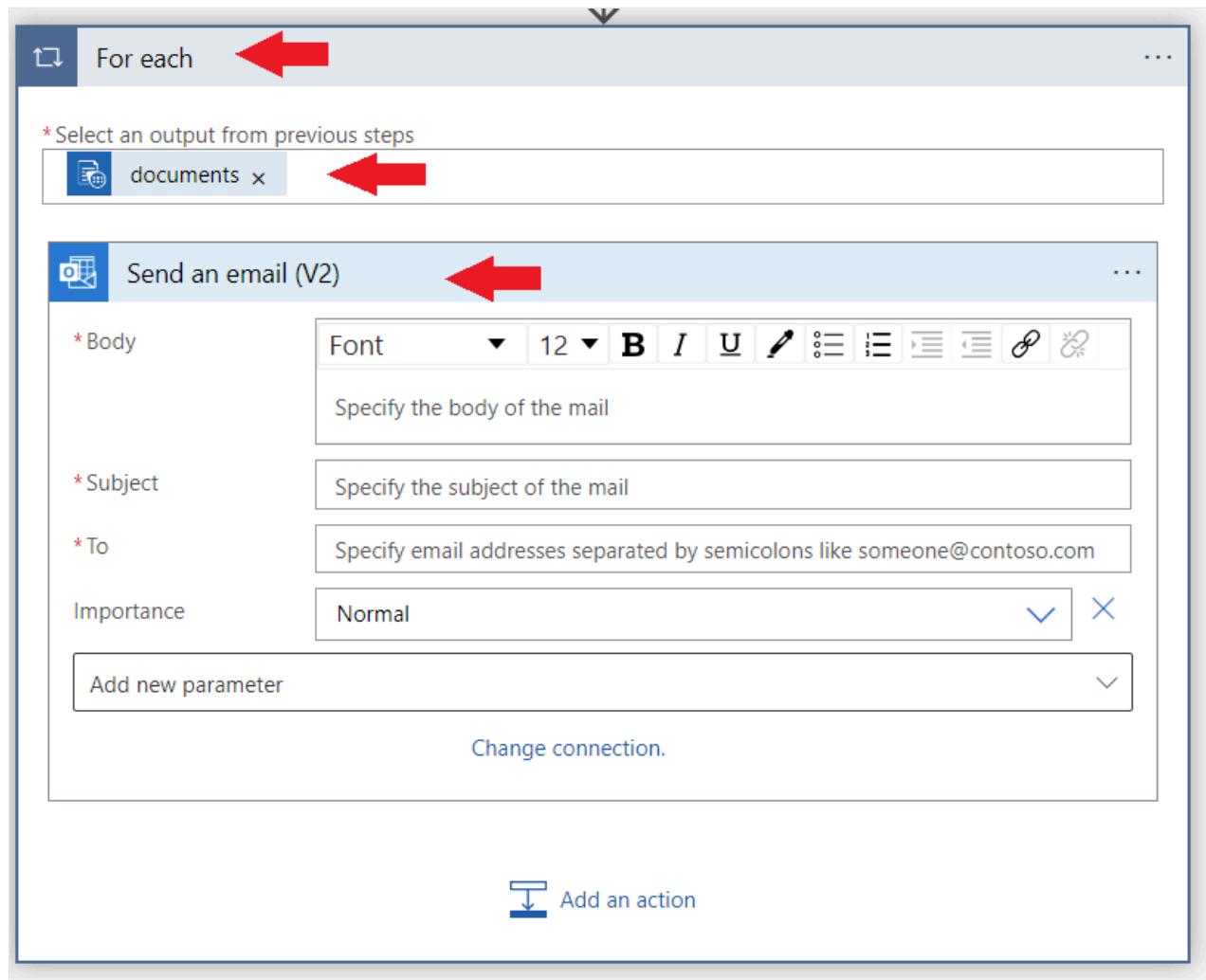
- Condition Control
- For each Control
- Scope Control

The 'For each Control' action is highlighted with a red box.

12. In the **For each** step window, there's a field labeled **Select an output from previous steps**. Select this field. A dynamic content pop-up appears. If it doesn't, select the **Add dynamic content** button below the field and choose **documents**.

The screenshot shows the 'Dynamic content' pop-up window. On the left, there is a list with an 'Add dynamic content' button. On the right, the 'documents' option is selected and highlighted with a red box. The 'documents' entry includes the sub-description 'Extracted documents.'

13. Now, select **Add an Action** from within the **For each** step window.
14. In the **Choose an operation** search bar, enter *Outlook* and select **Outlook.com (personal)** or **Office 365 Outlook (work)**.
15. In the actions list, scroll down until you find **Send an email (V2)** and select this action.



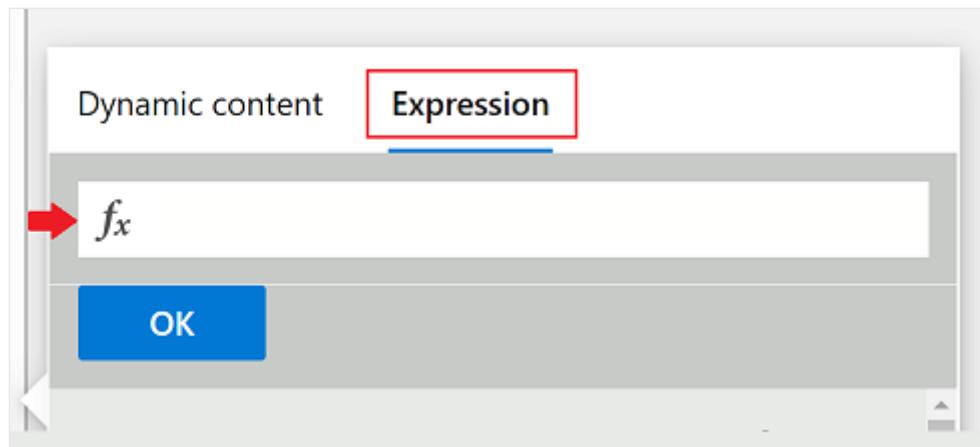
16. Just like with OneDrive, you're asked to sign into your Outlook or Office 365 Outlook account. After you sign in, you see a window where we're going to format the email with dynamic content that Document Intelligence extracts from the invoice.

17. We're going to use the following expression to complete some of the fields:

PowerApps Formula

```
items('For_each')?['fields']?['FIELD-NAME']?['content']
```

1. In order to access a specific field, we select the **add the dynamic content** button and select the **Expression** tab.



2. In the **fx** box, copy and paste the above formula and replace **FIELD-NAME** with the name of the field we want to extract. For the full list of available fields, refer to the concept page for the given API. In this case, we use the [prebuilt-invoice model field extraction values](#).
3. We're almost done! Make the following changes to the following fields:

- **To.** Enter your personal or business email address or any other email address you have access to.
- **Subject.** Enter ***Invoice received from:*** and then add the following expression:

PowerApps Formula

```
items('For_each')?['fields']?['VendorName']?['content']
```

- **Body.** We're going to add specific information about the invoice:
  - Type ***Invoice ID:*** and, using the same method as before, append the following expression:

PowerApps Formula

```
items('For_each')?['fields']?['InvoiceId']?['content']
```

- On a new line type ***Invoice due date:*** and append the following expression:

PowerApps Formula

```
items('For_each')?['fields']?['DueDate']?['content']
```

- Type ***Amount due:*** and append the following expression:

## PowerApps Formula

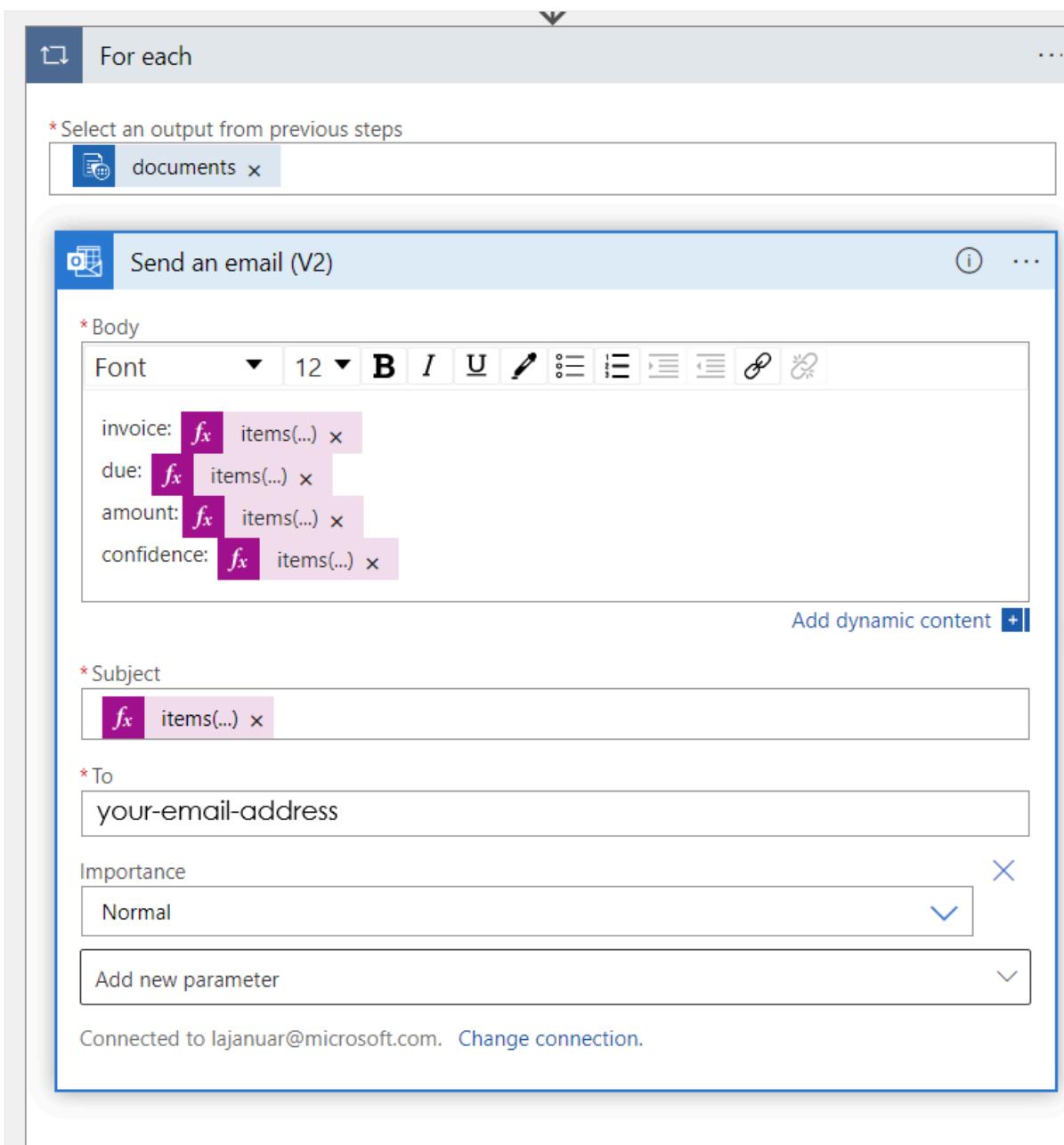
```
items('For_each')?['fields']?['AmountDue']?['content']
```

- Lastly, because the amount due is an important number, we also want to send the confidence score for this extraction in the email. To do this type **Amount due (confidence)**: and append the following expression:

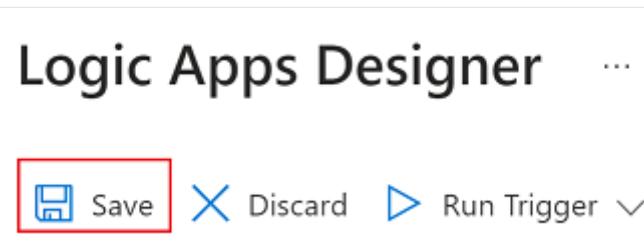
## PowerApps Formula

```
items('For_each')?['fields']?['AmountDue']?['confidence']
```

- When you're done, the window looks similar to the following image:



4. Select Save in the upper left corner.



#### ⚠ Note

- This current version only returns a single invoice per PDF.
- The "For each loop" is required around the send email action to enable an output format that may return more than one invoice from PDFs in the future.

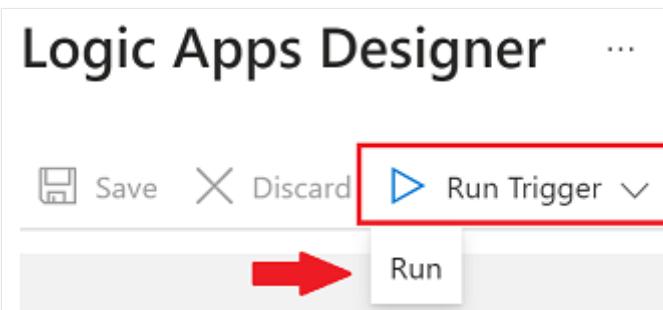
## Test the automation flow

Let's quickly review what we completed before we test our flow:

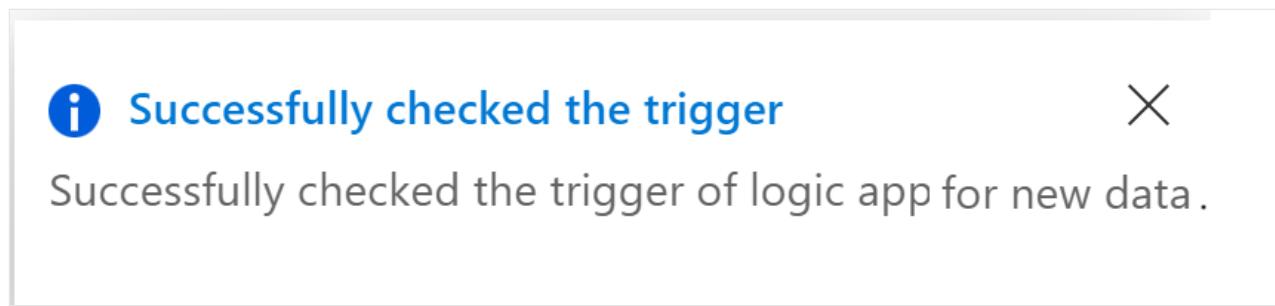
- ✓ We created a trigger—in this scenario. The trigger is activated when a file is created in a pre-specified folder in our OneDrive account.
- ✓ We added a Document Intelligence action to our flow. In this scenario, we decided to use the invoice API to automatically analyze an invoice from the OneDrive folder.
- ✓ We added an Outlook.com action to our flow. We sent some of the analyzed invoice data to a pre-determined email address.

Now that we created the flow, the last thing to do is to test it and make sure that we're getting the expected behavior.

1. To test the Logic App, first open a new tab and navigate to the OneDrive folder you set up at the beginning of this tutorial. Add this file to the OneDrive folder [Sample invoice](#).
2. Return to the Logic App designer tab and select the **Run trigger** button and select **Run** from the drop-down menu.



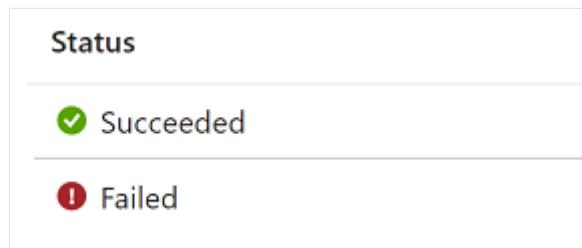
3. You see a message in the upper-right corner indicating that the trigger was successful:



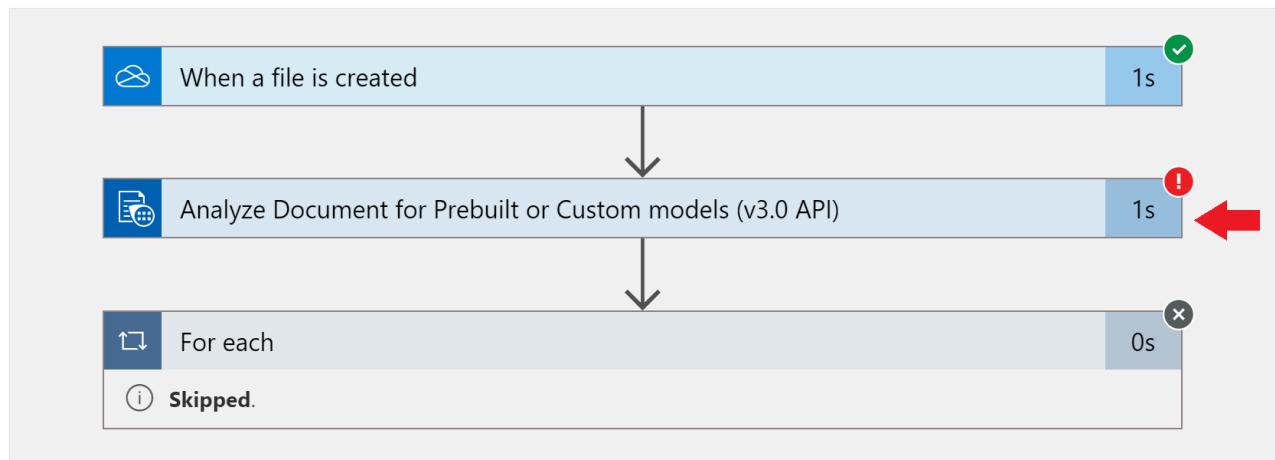
4. Navigate to your Logic App overview page by selecting your app name link in the upper-left corner.



5. Check the status, to see if the run succeeded or failed. You can select the status indicator to check which steps were successful.



6. If your run failed, check the failed step to ensure that you entered the correct information.



7. After a workflow run succeeds, check your email. There's a new email with the information we specified.

# Invoice received from:CONTOSO LTD.

Invoice ID: INV-100

Invoice due date:12/15/2019

Amount due: \$610.00

Amount due (confidence):0.967

8. After you're done, [disable or delete your logic app](#) so that usage stops.



Run Trigger



Refresh



Edit



Delete



Disable

Congratulations! You completed this tutorial.

## Next steps

[Learn more about Document Intelligence models](#)

# Create a document processing custom model

Article • 04/21/2025

After you review the [requirements](#), you can get started creating your document processing model.

## Create your model with the wizard

You can create a document processing model by using the **Create custom model** wizard. The wizard guides you through the process of creating a model to extract information from documents.

1. Sign in to [Power Apps](#) or [Power Automate](#).

2. On the left pane, select ... **More > AI hub**.

(Optional) To keep AI models permanently on the menu for easy access, select the pin icon next to **AI hub**.

3. Under **Discover an AI capability**, select **AI models**.

4. Select **Extract custom information from documents**.

5. Select **Create custom model**.

6. A step-by-step wizard walks you through the process by asking you to list all data you want to extract from your document.

Learn more in the [Select the type of document](#) section in this article.

If you want to create your model by using your own documents, make sure you have at least five examples that use the same layout. Otherwise, you can [use sample data](#) to create the model.

7. Select **Train**.

8. Test the model by selecting **Quick test**.

## Select the type of document

On the **Choose document type** step, select the type of document you want to build an AI model to automate data extraction. There are three options: **Fixed template documents**,

## General documents, and Invoices.

The screenshot shows the Power Apps | AI Builder interface. On the left, a vertical list of steps is shown with radio buttons: "Choose document type" (selected), "Choose information to extract", "Add collections of documents", "Tag documents", and "Model summary". On the right, a large text box says "Select the type of documents your model will process". Below it are three options: "Fixed template documents" (with a file icon), "General documents" (with a list icon), and "Invoices" (with a document icon). The "General documents" and "Invoices" boxes are highlighted with red rectangles.

- **Fixed template documents:** Previously known as *structured*, this option is ideal when, for a given layout, the fields, tables, checkboxes, signatures, and other items can be found in similar places. You can teach this model to extract data from structured documents that have different layouts. This model has a quick training time.
- **General documents:** Previously known as *unstructured*, this option is ideal for any kind of documents, especially when there's no set structure, or when the format is complex. You can teach this model to extract data from structured or unstructured documents that have different layouts. This model is powerful, but has long training time.
- **Invoices:** Augment the behaviors of the prebuilt invoice processing model by adding new fields to be extracted in addition to the ones by [default](#), or samples of documents not properly extracted.

## Understand document intelligence versions

The document intelligence model is available in two versions: v4.0 and v3.1. The version of your model depends on when you last edited the model.

### Document Intelligence v4.0 - General Availability (GA)

In addition to the features listed in this article, v4.0 retains all the capabilities of v3.1.

- **Overlapping fields:** v4.0 supports overlapping fields in custom models, which let you extract information more effectively from documents with complex layouts.
- **Signature detection:** v4.0 detects signatures in documents, which is especially useful for contracts, agreements, and other signed forms.
- **Confidence scores for tables:** v4.0 provides confidence scores for table and their cells.

- **OCR engine improvements:** v4.0 improves the optical character recognition (OCR) engine, enhancing text recognition accuracy and supporting more document types and formats.

## Document Intelligence v3.1 General Availability (GA)

- v3.1 supports custom models trained to recognize specific data patterns, such as unique text fields or structures.
- v3.1 includes custom template models that let users create templates based on their document layout and structure.

## Check the model version

You can verify the version used to train and publish your model. To do this, select **Settings > Published model version > Last trained model version.**

The screenshot shows the Microsoft Document Intelligence interface. On the left, there's a navigation bar with 'Edit model', 'Share', 'Settings', and 'Delete' buttons. Below that, it says 'Models > Cycles Invoices' and 'Document Processing • Fixed template documents • Published [REDACTED]'. There are two tabs: 'Published version' and 'Last trained version', with 'Last trained version' being active. Under 'Published version', there's an 'Accuracy score' section showing a gauge at 99% (Excellent). A note below says: 'This model correctly predicted 99% of actual results and may be ready to be used. To improve the accuracy score, [review full evaluation](#)'. At the bottom are 'Publish' and 'Quick test' buttons. On the right, under 'Model settings', it shows 'Model name: Cycles Invoices', 'Model id: [REDACTED]', 'Published model version: GA (v3.1)', 'Last trained model version: GA (v4.0)', and 'Created date: 1/9/2025, 1:26:45 PM'. The 'Model settings' tab is highlighted with a red border.

You can move a model from v3.1 to v4.0 by editing, retraining, and publishing it. Re-tagging and other specific modification aren't necessary. Learn more in [FAQ for document processing](#).

## Define information to extract

On the Choose information to extract screen, define the fields, tables, and checkboxes you want to teach your model to extract. To start defining these, select **+Add**.

## Add

Text field

Number field

Date field

Checkbox

Table

Recipient's / Lender's <b>Contoso, Ltd.</b> 4567 Main St. Buffalo, NY 90852		Mortgage Statement	
Recipient's / Lender's TIN <b>750-01-1829</b>	Payer's/Borrower's TIN <b>306-14-5298</b>	1.Mortgage interest <b>\$ 13,345.34</b>	2.Outstanding mortgage <b>\$ 764,321.33</b>
Payer's/Borrower's name <b>Helena Wilcox</b>		3.Mortgage origination date <b>10/08/2019</b>	4.Refund of overpaid <b>\$ 15.01</b>
Street address <b>123 Main St Apt. 10</b>		5.Mortgage insurance <b>\$ 1,222.05</b>	6.Address <b>1025 Fifth St. Sunnyvale, CA 27673</b>
City or Town, state or province, country, and ZIP or foreign postal code <b>Brooklyn, NY 90852</b>		8.Number of properties securing the mortgage <b>2</b>	9.Other

**Overview**

Use this to extract printed or handwritten text from your documents.  
For example: Name, Address, Phone number.

**Next** **Cancel**

1. For each text field, provide a name for the field to use in the model.

2. For each number field, provide a name for the field to use in the model.

Define the format dot (.) or comma (,) as a decimal separator.

3. For each **Date field**, provide a name for the field to use in the model.

Also, define the format (**Year, Month, Day**), or (**Monthly, Day, Year**), or (**Day, Month, Year**).

4. For each checkbox, provide a name for the checkbox to use in the model.

Define separate checkboxes for each item that can be checked in a document.

5. For each table, provide the name for the table.

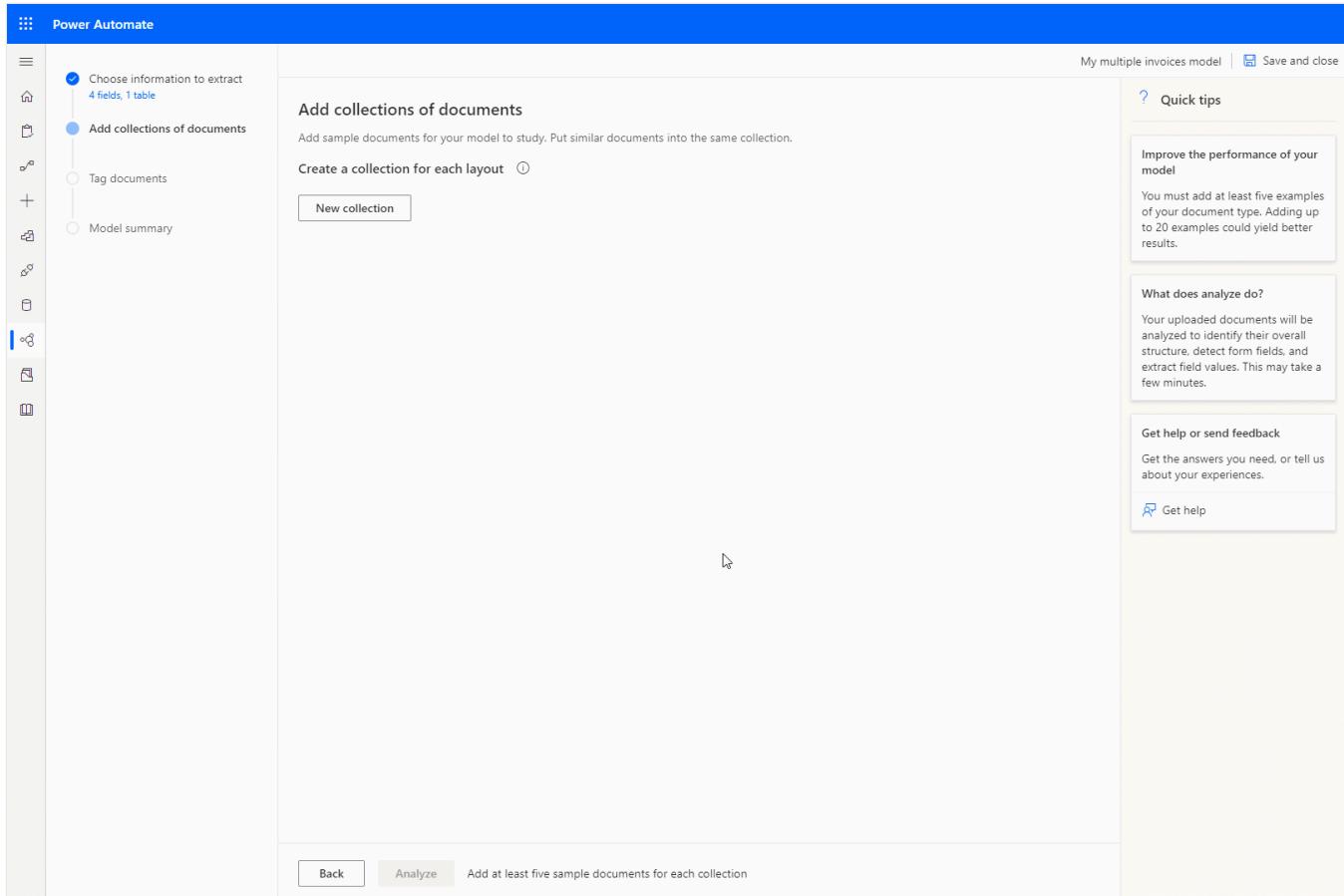
Define the different columns that the model should extract.

### ⚠ Note

The custom invoices model comes with default fields that can't be edited.

## Group documents by collections

A *collection* is a group of documents that share the same layout. Create as many collections as document layouts that you want your model to process. For example, if you're building an AI model to process invoices from two different vendors, each having their own invoice template, create two collections.



For each collection that you create, you need to upload at least five sample documents per collection. Files with formats JPG, PNG, and PDF files are accepted.

The screenshot shows the 'Power Automate' interface with a sidebar containing icons for Home, Model, Add collection, Tag documents, and Model summary. The main area displays a flow titled 'My multiple invoices model'. A step titled 'Add collections of documents' is currently selected. It includes a note about processing documents with different layouts, a 'New collection' button, and two collection boxes labeled 'Adatum' and 'Contoso', each showing 0 documents. To the right, there are 'Quick tips' sections for 'Improve the performance of your model' (which suggests adding at least five examples) and 'What does analyze do?' (explaining the analysis process). At the bottom, there are 'Back', 'Analyze', and a note to 'Add at least five sample documents for each collection'.

### ! Note

You can create up to 200 collections per model.

## Next step

[Tag documents in a document processing model](#)

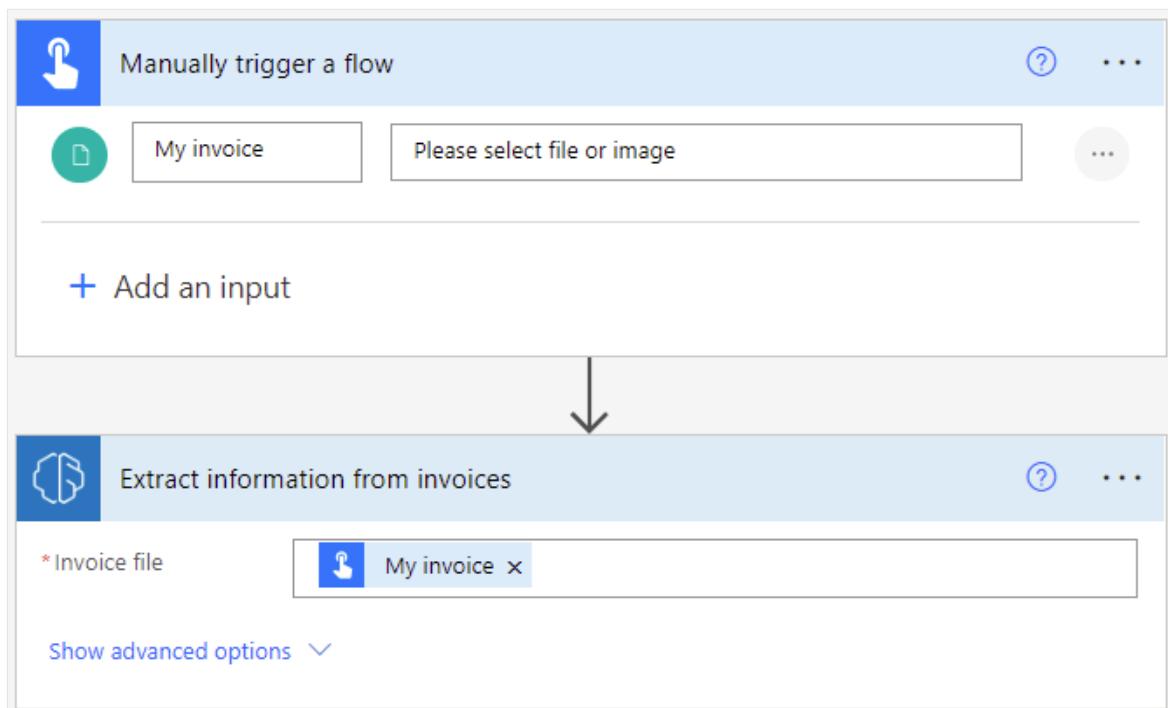
## Related information

- [Training: Process custom documents with AI Builder \(module\)](#)
- [Interpret confidence score for tables and table cells](#)
- [FAQ for document processing](#)

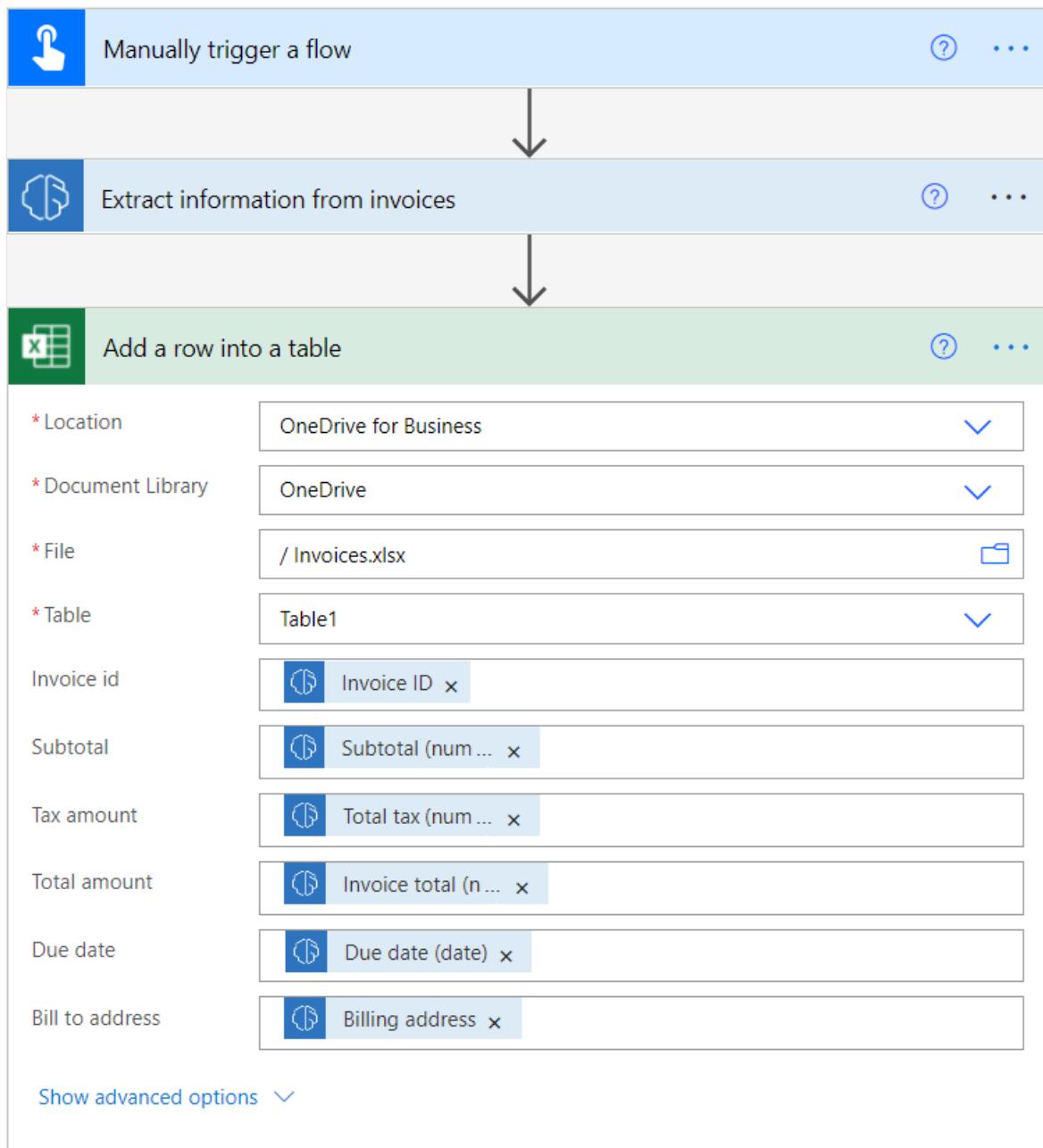
# Use the invoice processing prebuilt model in Power Automate

Article • 11/20/2024

1. Sign in to [Power Automate](#).
2. Select **My flows** in the left pane, and then select **New flow > Instant cloud flow**.
3. Name your flow, select **Manually trigger a flow** under **Choose how to trigger this flow**, and then select **Create**.
4. Expand **Manually trigger a flow**, and then select **+Add an input > File** as the input type.
5. Replace **File Content** with **My invoice** (also known as the title).
6. Select **+New step > AI Builder**, and then select **Extract information from invoices** in the list of actions.
7. Specify **My invoice** from the trigger in the **Invoice file** input.



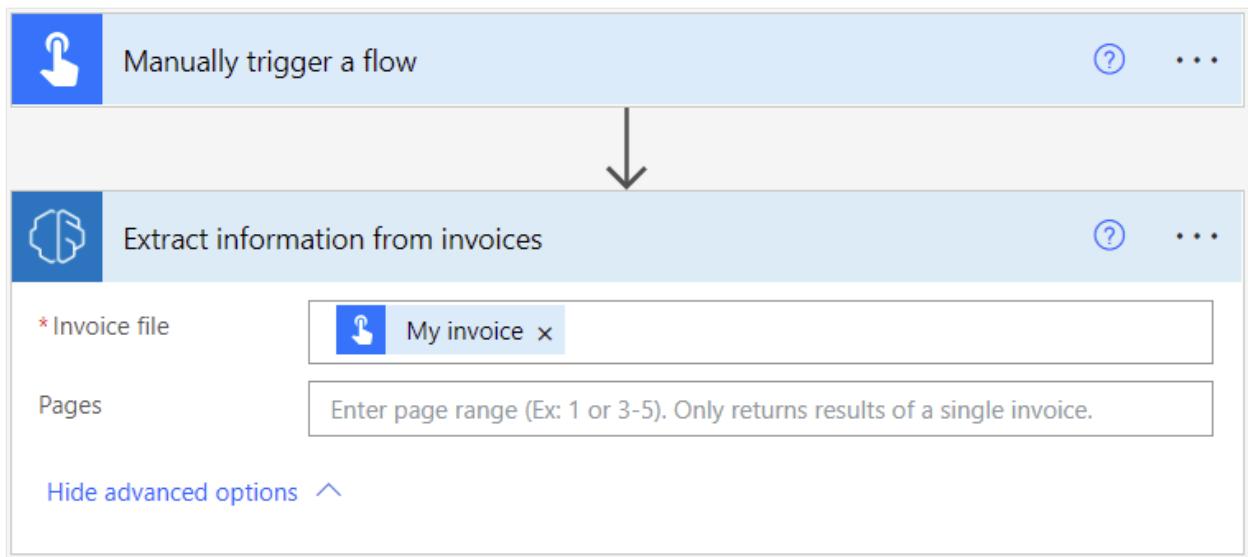
8. In the successive actions, you can use any of the invoice values from the [model output](#).



Congratulations! You've created a flow that uses the AI Builder invoice processing model. Select **Save** on the top right, and then select **Test** to try out your flow.

## Page range

For large documents, it's possible to specify the page range to process.



You can enter a page value or page range in the **Pages** parameter. Example: 1 or 3-5.

### ! Note

If you have a large document with only one invoice, we strongly recommend to **use the Pages parameter to aim at your invoice, and therefore reduce the cost of model prediction and increase performance**. However, the page range should contain a **unique invoice** for the action to return correct data.

Example: A document contains a first invoice in page 2 and a second invoice that spans over pages 3 and 4:

- If you enter page range 2, it will return the data of the first invoice.
- If you enter page range 3-4, it will only return the data of the second invoice.
- If you enter page range 2-4, it will return partial data of the first and second invoices (should be avoided).

## Parameters

### Input

[] Expand table

Name	Required	Type	Description
Receipt file	Yes	file	The invoice file to process
Pages	No	string	Page range to process

# Output

[Expand table](#)

Name	Type	Definition
Amount due (text)	string	Amount due as it's written on the invoice
Amount due (number)	float	Amount due in standardized number format. Example: 1234.98
Confidence of amount due	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Billing address	string	Billing address
Confidence of billing address	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Billing address recipient	string	Billing address recipient
Confidence of billing address recipient	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Customer address	string	Customer address
Confidence of customer address	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Customer address recipient	string	Customer address recipient
Confidence of customer address recipient	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Customer ID	string	Customer ID
Confidence of customer ID	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Customer name	string	Customer name
Confidence of customer name	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Due date (text)	string	Due date as it's written on the invoice
Due date (date)		Due date in standardized date format. Example: 2019-05-31T00:00:00Z
Confidence of due date	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).

Name	Type	Definition
Invoice date (text)	string	Invoice date as it's written on the invoice
Invoice date (date)	date	Invoice date in standardized date format. Example: 2019-05-31T00:00:00Z
Confidence of invoice date	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Invoice ID	string	Invoice ID
Confidence of invoice ID	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Invoice total (text)	string	Invoice total as it's written on the invoice
Invoice total (number)	float	Invoice total in standardized date format. Example: 2019-05-31T00:00:00Z
Confidence of invoice total	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Purchase order	string	Purchase order
Confidence of purchase order	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Remittance address	string	Remittance address
Confidence of remittance address	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Remittance address recipient	string	Remittance address recipient
Confidence of remittance address recipient	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Service address	string	Service address
Confidence of service address	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Service address recipient	string	Service address recipient
Confidence of service address recipient	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Shipping address	string	Shipping address
Confidence of shipping	float	How confident the model is in its prediction. Score

Name	Type	Definition
address		between 0 (low confidence) and 1 (high confidence).
Shipping address recipient	string	Shipping address recipient
Confidence of shipping address recipient	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Subtotal (text)	string	Subtotal as it's written on the invoice
Subtotal (number)	float	Subtotal in standardized number format. Example: 1234.98
Confidence of subtotal	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Total tax (text)	string	Total tax as it's written on the invoice
Total tax (number)	float	Total tax in standardized number format. Example: 1234.98
Confidence of total tax	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Vendor address	string	Vendor address
Confidence of vendor address	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Vendor address recipient	string	Vendor address recipient
Confidence of vendor address recipient	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Vendor name	string	Vendor name
Confidence of vendor name	float	How confident the model is in its prediction. Score between 0 (low confidence) and 1 (high confidence).
Detected text	string	Line of recognized text from running OCR on an invoice. Returned as a part of a list of text.
Page number of detected text	integer	Which page the line of recognized text is found on. Returned as a part of a list of text.

## Related topics

[Invoice processing overview](#)

---

## Feedback

Was this page helpful?

 Yes

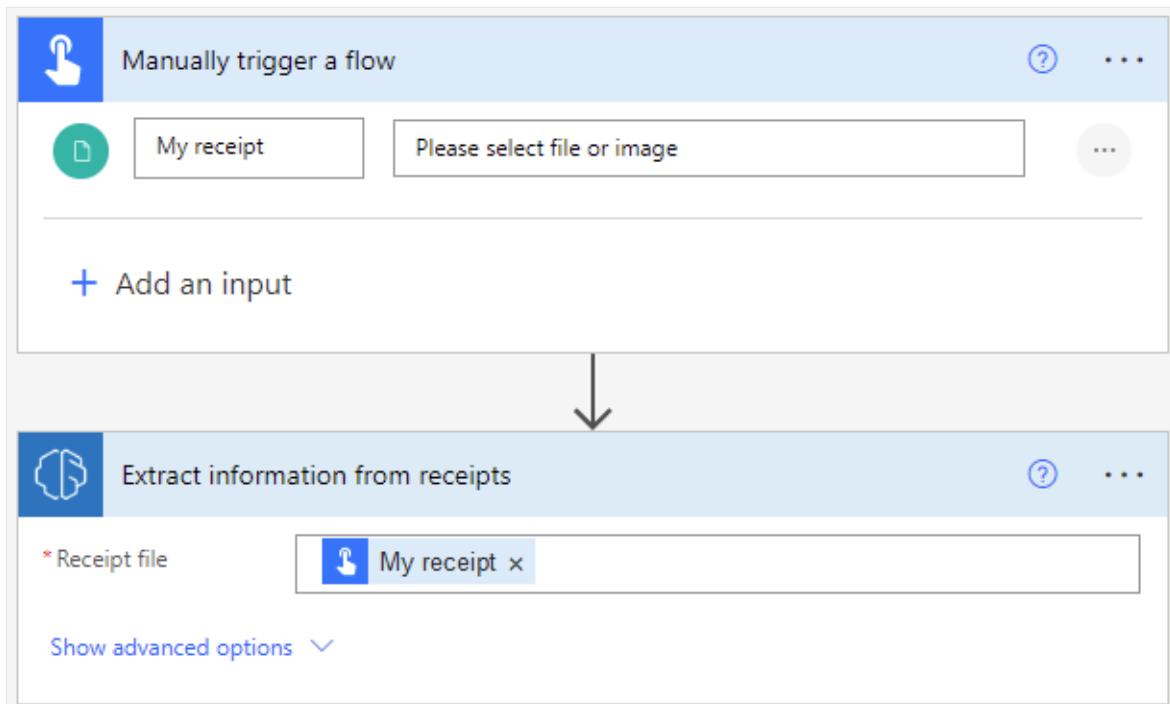
 No

Provide product feedback 

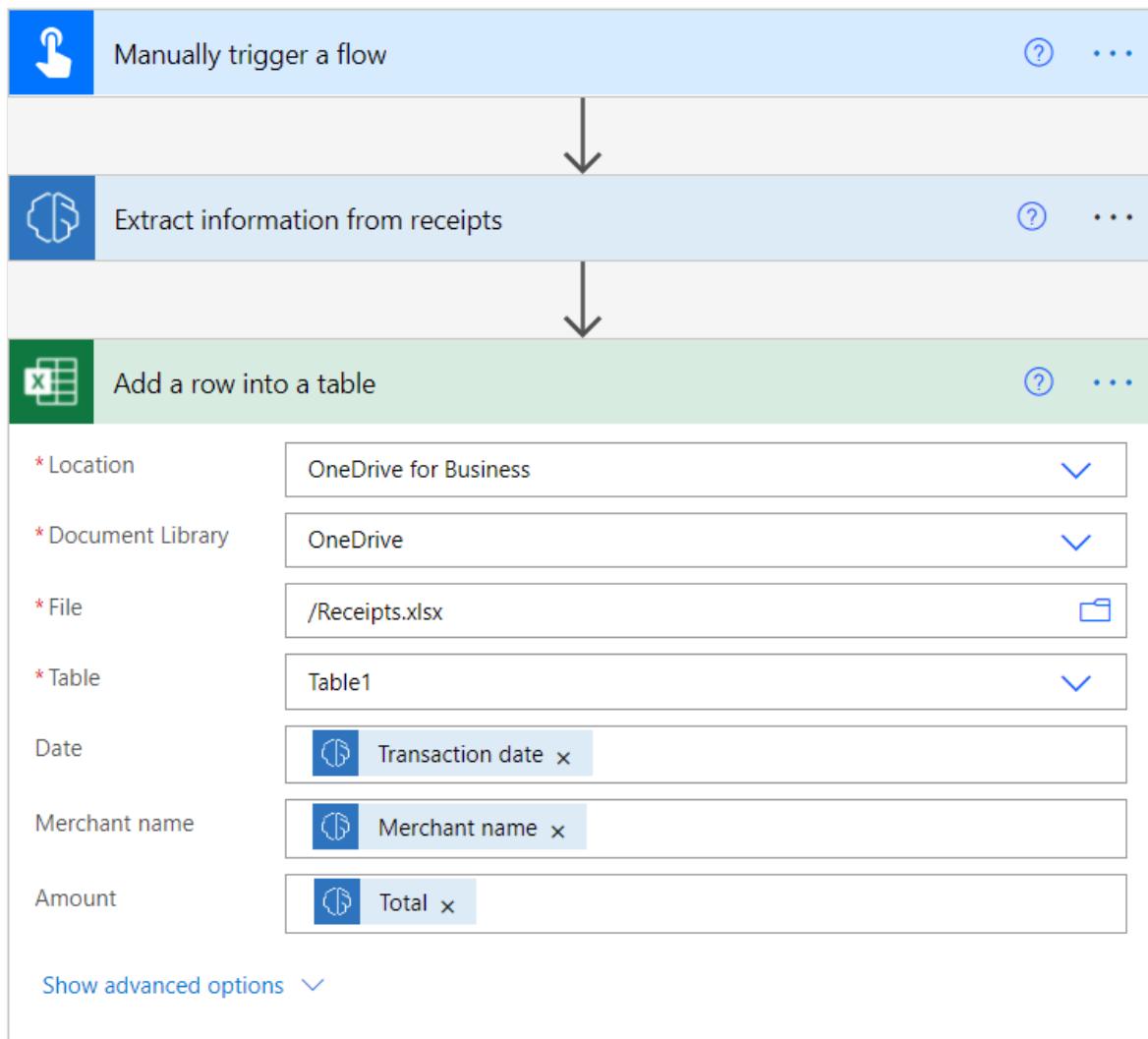
# Use the receipt processing prebuilt model in Power Automate

Article • 01/27/2025

1. Sign in to [Power Automate](#).
2. Select **My flows** in the left pane, and then select **New flow > Instant cloud flow**.
3. Name your flow, select **Manually trigger a flow** under **Choose how to trigger this flow**, and then select **Create**.
4. Expand **Manually trigger a flow**, and then select **+Add an input > File** as the input type.
5. Replace **File Content** with **My receipt** (also known as the title).
6. Select **+New step > AI Builder**, and then select **Extract information from receipts** in the list of actions.
7. Select the **Receipt file** input, and then select **My receipt** from the **Dynamic content** list:



8. In the successive actions, you can use any of the receipt values from the [model output](#) section below.



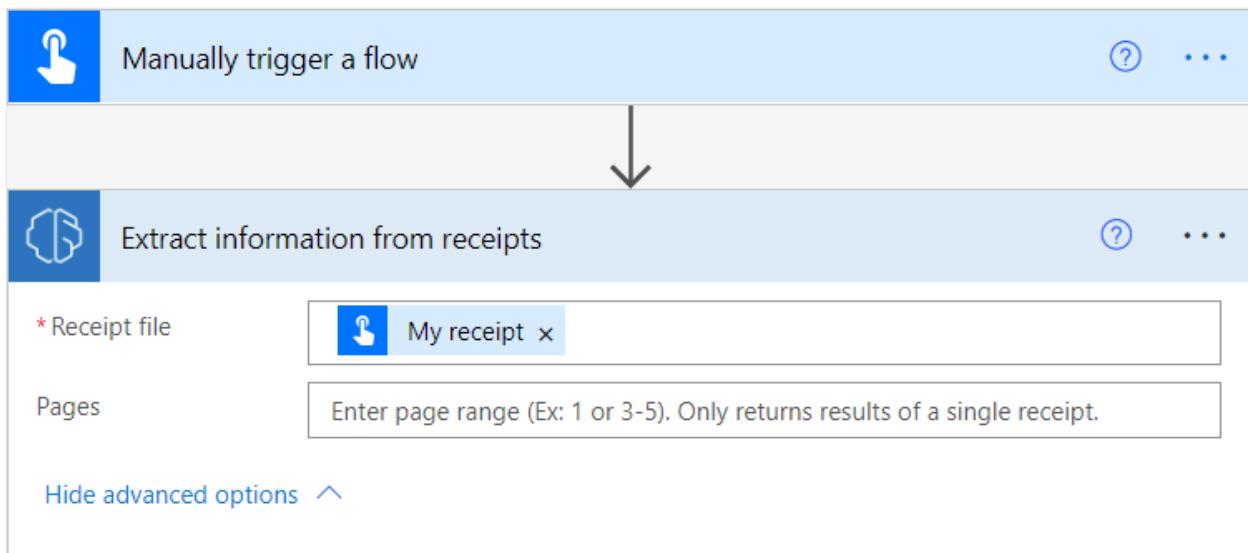
### ! Note

Receipt values are returned as strings. To manipulate them as numbers, you can use the [float](#) or [int](#) conversion functions.

Congratulations! You've created a flow that uses the AI Builder receipt processing model. Select **Save** on the top right, and then select **Test** to try out your flow.

## Page range

For large documents, it's possible to specify the page range to process.



You can enter a page value or page range in the **Pages** parameter. Example: 1 or 3-5.

#### ⓘ Note

If you have a large document with only one receipt, we strongly recommend to use the Pages parameter to aim at your receipt, and therefore reduce the cost of model prediction and increase performance. However, note that only the data of the **first receipt** within the page range will be returned and that multi-page receipts are not supported.

Example: A document contains a first receipt in page 2 and a second receipt that spans overs page 3 and 4:

- If you enter value 2, it will return the data of the first receipt
- If you enter value 3-4, it will only return the data of the first page of the second receipt
- If you enter value 2-4, it will only return data of the first receipt, not the data of the second receipt

## Parameters

### Input

[ ] [Expand table](#)

Name	Required	Type	Description
Receipt file	Yes	string	The receipt file to process

Name	Required	Type	Description
Pages	No	string	Page range to process

## Output

[\[\] Expand table](#)

Name	Type	Description
Merchant name	string	Merchant name
Merchant address	string	Merchant address
Merchant phone number	string	Merchant phone number
Transaction date	string	Transaction date
Transaction time	string	Transaction time
Purchased item name	string	Purchased item name. Returned as a part of a list of items.
Purchased item quantity	string	Purchased item quantity. Returned as a part of a list of items.
Purchased item price	string	Purchased item price. Returned as a part of a list of items.
Purchased item total price	string	Purchased item total price. Returned as a part of a list of items.
Subtotal	string	Subtotal
Tax	string	Tax
Tip	string	Tip
Total	string	Total
Confidence of merchant name	float	How confident the model is in its detection
Confidence of merchant address	float	How confident the model is in its detection
Confidence of merchant phone number	float	How confident the model is in its detection
Confidence of transaction	float	How confident the model is in its detection

Name	Type	Description
date		
Confidence of transaction time	float	How confident the model is in its detection
Confidence of purchased item name	float	How confident the model is in its detection. Returned as a part of a list of items.
Confidence of purchased item quantity	float	How confident the model is in its detection. Returned as a part of a list of items.
Confidence of purchased item price	float	How confident the model is in its detection. Returned as a part of a list of items.
Confidence of purchased item total price	float	How confident the model is in its detection. Returned as a part of a list of items.
Confidence of subtotal	float	How confident the model is in its detection
Confidence of tax	float	How confident the model is in its detection
Confidence of tip	float	How confident the model is in its detection
Confidence of total	float	How confident the model is in its detection
Detected text	string	Line of recognized text. Returned as a part of a list of text.
Page number of detected text	integer	Which page the line of recognized text is found on. Returned as a part of a list of text.
Height of detected text	float	Height of the line of text. Returned as a part of a list of text.
Left position of detected text	float	Left position of the line of text. Returned as a part of a list of text.
Top position of detected text	float	Top position of the line of text. Returned as a part of a list of text.
Width of detected text	float	Width of the line of text. Returned as a part of a list of text.

## Related information

- [Receipt processing overview](#)
- [Training: Process receipts with AI Builder \(module\)](#)

# Feedback

Was this page helpful?

 Yes

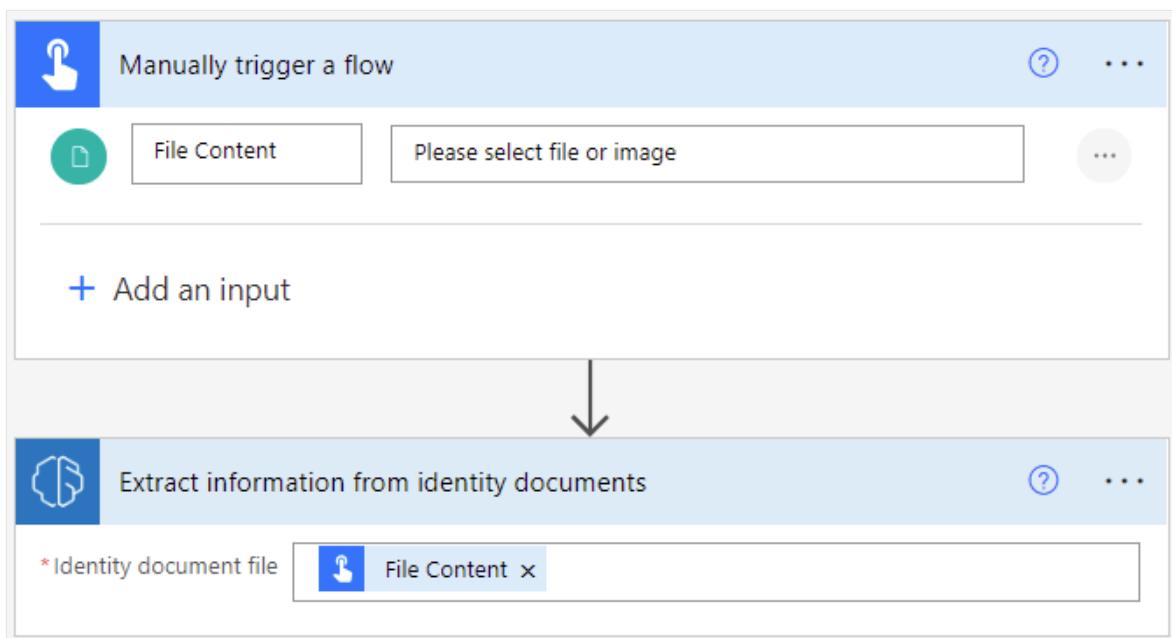
 No

[Provide product feedback ↗](#)

# Use the ID reader prebuilt model in Power Automate

Article • 01/27/2025

1. Sign in to [Power Automate](#).
2. In the left pane, select **My flows**, and then select **New flow > Instant cloud flow** in the menu at the top.
3. Name your flow, select **Manually trigger a flow** under **Choose how to trigger this flow**, and then select **Create**.
4. Expand **Manually trigger a flow**, and then select **+Add an input > File** as the input type.
5. Select **+New step > AI Builder > Extract information from identity documents**.
6. Specify **File Content** as the identity document file you want to process in your flow.



Congratulations! You've created a flow that uses the ID reader model. Select **Save**, and then select **Test** in the upper-right corner to try out your flow.

## Example flow that adds extracted information to an Excel worksheet

In the following example, you'll add steps to your flow to enter the extracted information in an Excel worksheet. First, you'll prepare a table to use in your flow. The

table must match the information you want to extract. Then you'll add an Excel connector to your flow.

## Create an Excel table

1. Create an Excel workbook in a Microsoft OneDrive or SharePoint folder.
2. In the first row of the worksheet, enter the following values, one to a column: **First name**, **Last name**, **Identity document number**, and **Country**. These values are the column headers for your table.
3. Select the cells and format them as a table, with the first row as the header.

	A	B	C	D
1	<b>First name</b>	<b>Last name</b>	<b>Identity document number</b>	<b>Country</b>
2				
3				

4. Save and close the workbook.

## Enter the extracted data in the table

1. Use the ID reader flow you created, or create another one for this example.
2. Select **+New step > Excel Online (Business) > Add a row into a table**.
3. Select a **Location**, **Document Library**, and **File** to specify where to find your Excel workbook.
4. Select the **Table** that you created in the previous step.
5. In **First name**, **Last name**, and **Identity document number**, select the matching value in the dynamic content list.
6. In **Country**, select **Country/Region** in the dynamic content list.

Add a row into a table

\* Location: OneDrive for Business

\* Document Library: OneDrive

\* File: /ID Documents/ID Documents.xlsx

\* Table: Table1

First name: First name x

Last name: Last name x

Identity document number: Identity docum... x

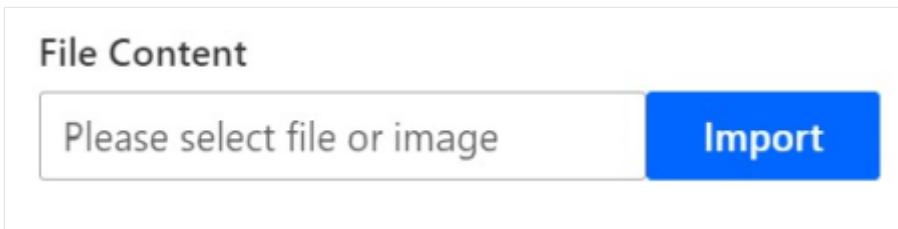
Country: Country/Region x

Show advanced options ▾

7. Select Save.

## Test the flow

1. Select **Test**, select **Manually**, and then select **Test** to trigger the action.
2. In **File Content**, select an identity document file or image, and then select **Import**.



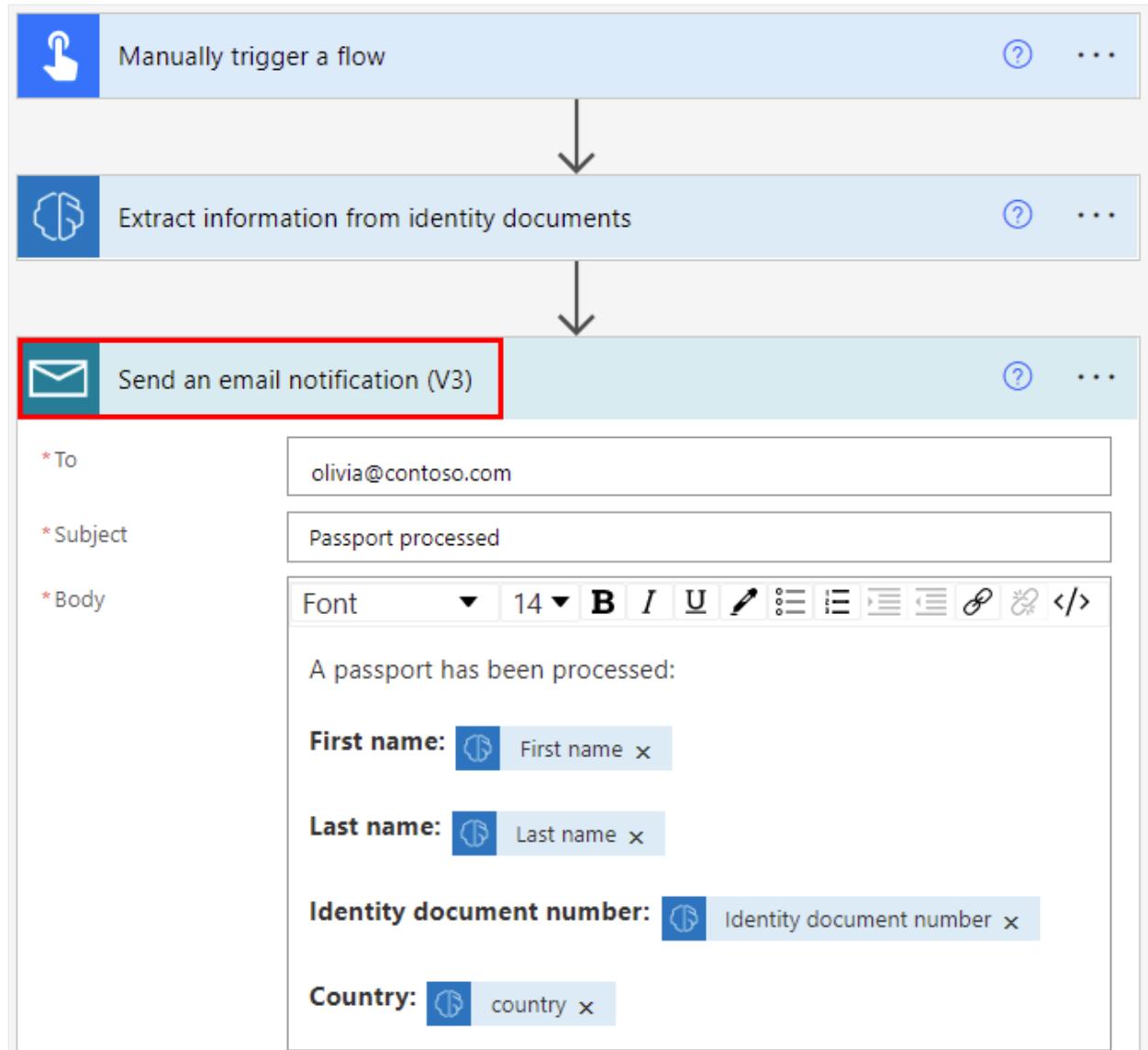
3. Select **Run Flow**.

The flow may take a few seconds to execute while AI Builder extracts the data and adds a new entry to the table in Excel. Open your Excel workbook to confirm the extracted information has been entered.

	A	B	C	D
1	First name	Last name	Identity document number	Country
2	JENNIFER	BROOKS	340020013	USA
3				

# Example flow that sends extracted information in an email

The following example shows how to set up a flow to send the extracted information in an email. You can add the **Send an email notification** connector to the flow you created earlier or create an ID reader flow for this example.



## Related information

[ID reader prebuilt model overview](#)

## Feedback

Was this page helpful?

Yes

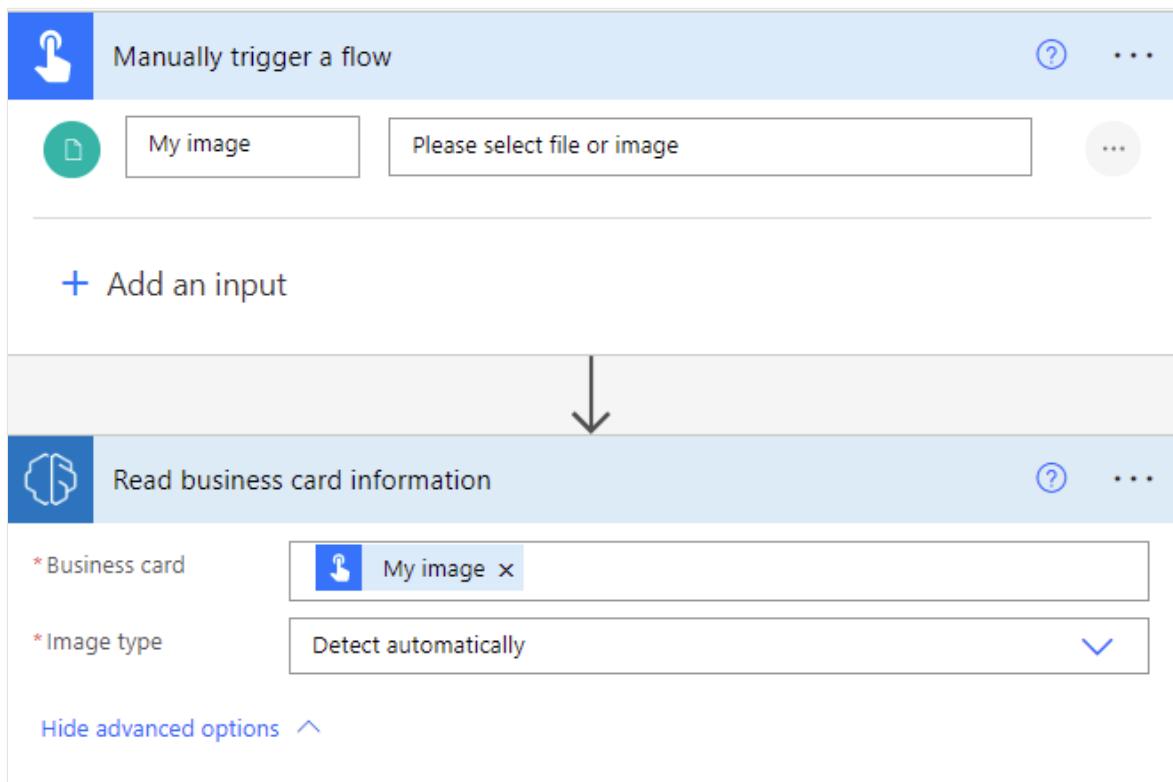
No

Provide product feedback ↗

# Use the business card reader prebuilt model in Power Automate

Article • 01/27/2025

1. Sign in to [Power Automate](#).
2. Select **My flows** in the left pane, and then select **New flow > Instant cloud flow**.
3. Name your flow, select **Manually trigger a flow** under **Choose how to trigger this flow**, and then select **Create**.
4. Expand **Manually trigger a flow**, and then select **+Add an input > File** as the input type.
5. Replace **File Content** with **My image** (also known as the title).
6. Select **+ New step > AI Builder**, and then select **Read business card information** in the list of actions.
7. Specify **My Image** from the trigger in the **Business card** input for your flow.
8. Select **Show advanced options** and verify that **Detect automatically** is in the **Image type** input.

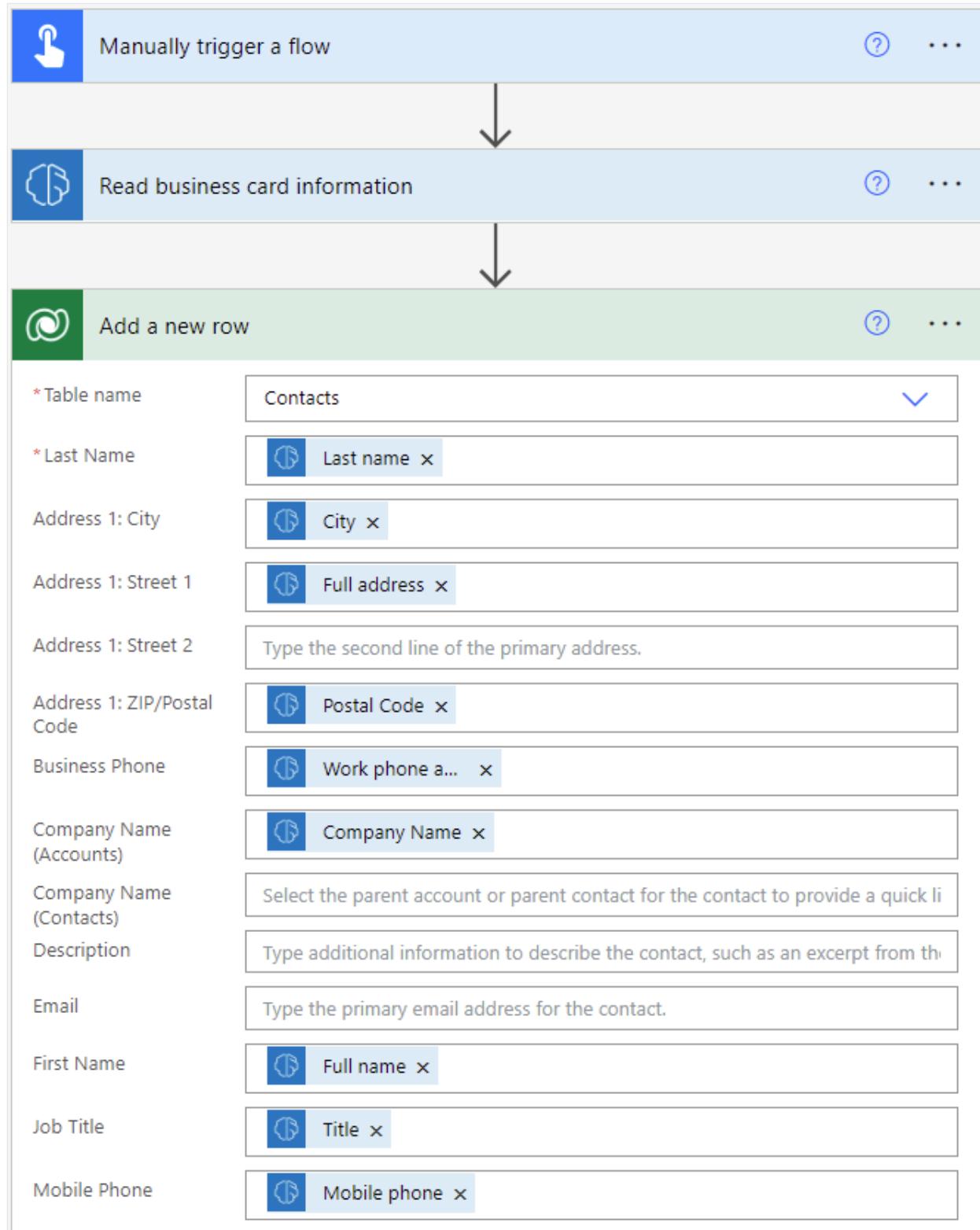


Congratulations! You've created a flow that uses the business card reader AI model. Select **Save**, and then select **Test** in the upper-right corner to try out your flow.

# Example business card reader flow

The following example shows a new contact being created in Microsoft Dataverse using the business card data.

To add the **Add a new row** step, select **+ New step > Microsoft Dataverse > Add a new row**.



## Parameters

## Input

[\[\] Expand table](#)

Name	Required	Type	Description	Values
Image type	Yes	string	Mime type of the image	"auto" as default value. This column being obsolete, any value will be accepted.
Image	Yes	file	Image file to analyze	

## Output

[\[\] Expand table](#)

Name	Type	Description
City	string	The city address
Country	string	The country address
Postal Code	string	The postal code address
PO Box	string	The post office box address
State	string	The state address
Street	string	The street address
Work phone or other phone	string	The first phone or fax number
Company name	string	The company name
Department	string	The organization department found
Email	string	The contact email found in the business card, if any
Fax	string	The third phone or fax number
First name	string	The contact first name
Full address	string	The contact full address
Full name	string	The contact full name
Title	string	The contact job title
Last name	string	The contact last name

Name	Type	Description
Mobile phone	string	The second phone or fax number
Website	string	The website

## Related information

[Business card reader overview](#)

---

## Feedback

Was this page helpful?

 Yes

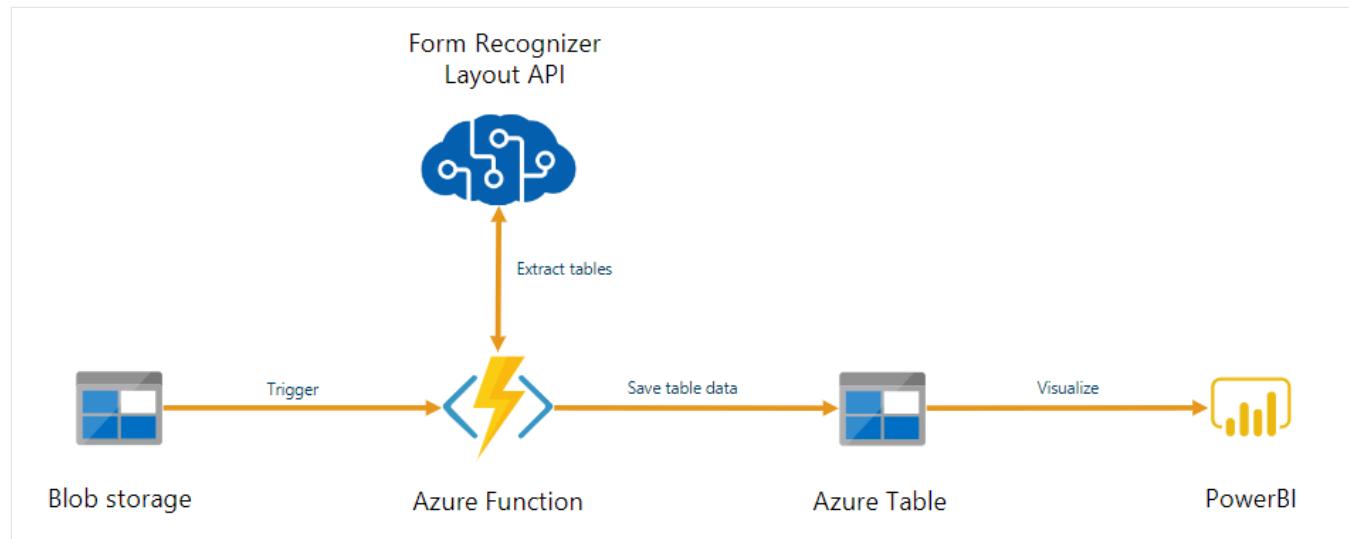
 No

[Provide product feedback ↗](#)

# Tutorial: Use Azure Functions and Python to process stored documents

Article • 03/19/2025

Document Intelligence can be used as part of an automated data processing pipeline built with Azure Functions. This guide will show you how to use Azure Functions to process documents that are uploaded to an Azure blob storage container. This workflow extracts table data from stored documents using the Document Intelligence layout model and saves the table data in a .csv file in Azure. You can then display the data using Microsoft Power BI (not covered here).



In this tutorial, you learn how to:

- ✓ Create an Azure Storage account.
- ✓ Create an Azure Functions project.
- ✓ Extract layout data from uploaded forms.
- ✓ Upload extracted layout data to Azure Storage.

## Prerequisites

- Azure subscription - [Create one for free](#)
- A Document Intelligence resource. Once you have your Azure subscription, create a [Document Intelligence resource](#) in the Azure portal to get your key and endpoint. You can use the free pricing tier (`F0`) to try the service, and upgrade later to a paid tier for production.
  - After your resource deploys, select **Go to resource**. You need the key and endpoint from the resource you create to connect your application to the Document Intelligence API. You'll paste your key and endpoint into the code below later in the tutorial:

The screenshot shows the Azure portal interface for managing a Cognitive Services resource. The left sidebar has a tree view with 'Keys and Endpoint' selected. The main pane shows the keys and endpoint configuration. A tooltip at the top right provides instructions on key management.

- [Python 3.6.x, 3.7.x, 3.8.x or 3.9.x](#) (Python 3.10.x isn't supported for this project).
- The latest version of [Visual Studio Code](#) (VS Code) with the following extensions installed:
  - [Azure Functions extension](#). Once it's installed, you should see the Azure logo in the left-navigation pane.
  - [Azure Functions Core Tools](#) version 3.x (Version 4.x isn't supported for this project).
  - [Python Extension](#) for Visual Studio code. For more information, see [Getting Started with Python in VS Code](#)
- [Azure Storage Explorer](#) installed.
- A local PDF document to analyze. You can use our [sample pdf document](#) for this project.

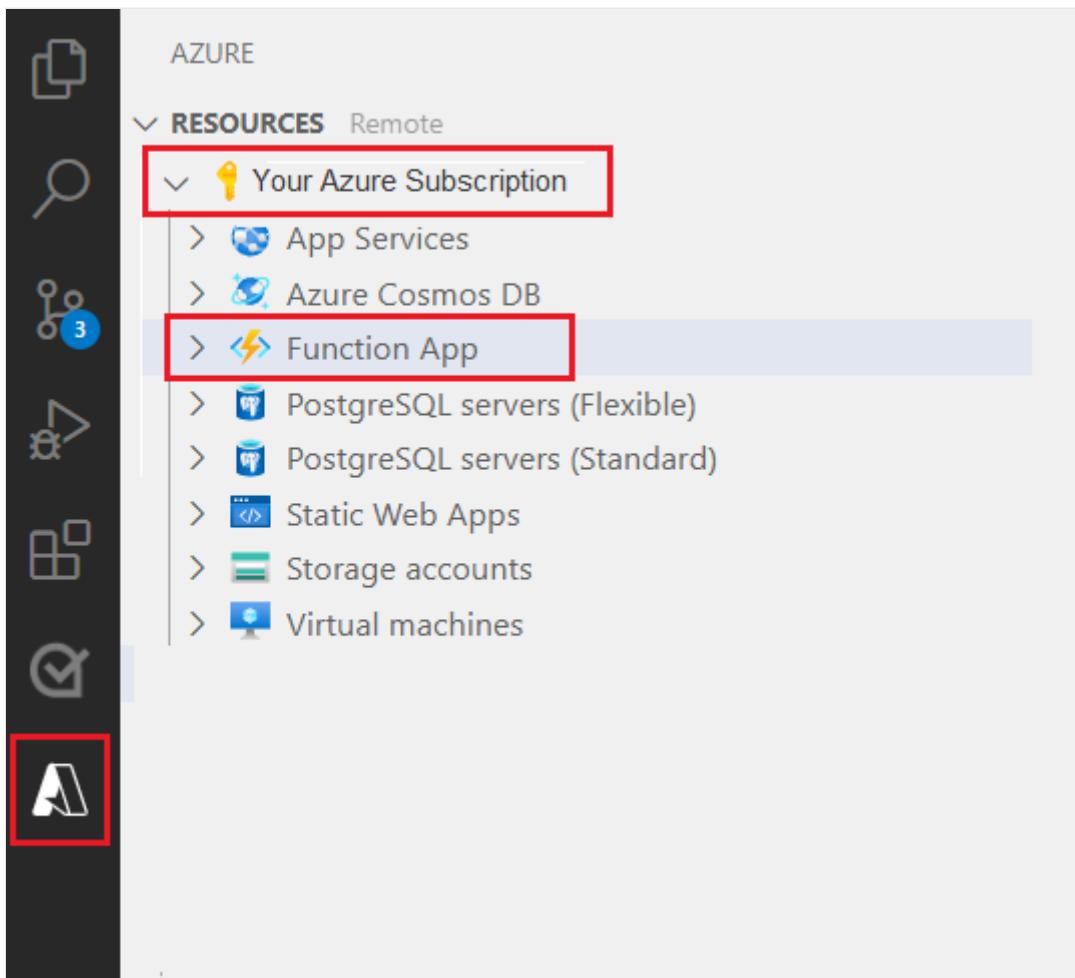
## Create an Azure Storage account

1. [Create a general-purpose v2 Azure Storage account](#) in the Azure portal. If you don't know how to create an Azure storage account with a storage container, follow these quickstarts:
  - [Create a storage account](#). When you create your storage account, select **Standard** performance in the **Instance details > Performance** field.

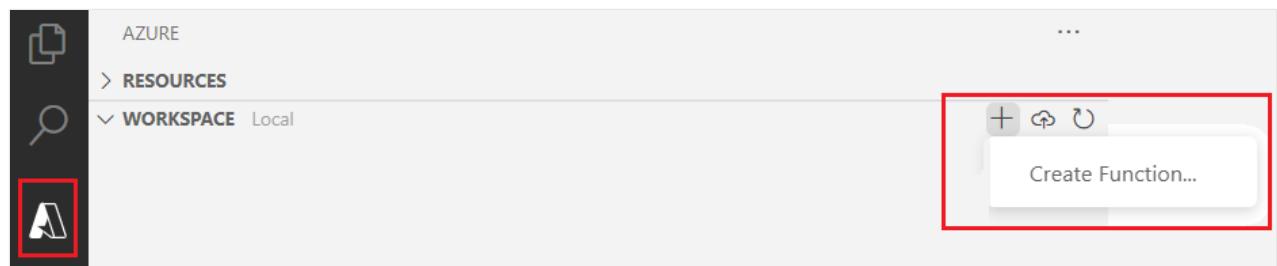
- [Create a container](#). When you create your container, set **Public access level** to **Container** (anonymous read access for containers and files) in the **New Container** window.
2. On the left pane, select the **Resource sharing (CORS)** tab, and remove the existing CORS policy if any exists.
  3. Once your storage account has deployed, create two empty blob storage containers, named **input** and **output**.

## Create an Azure Functions project

1. Create a new folder named **functions-app** to contain the project and choose **Select**.
2. Open Visual Studio Code and open the Command Palette (Ctrl+Shift+P). Search for and choose **Python:Select Interpreter** → choose an installed Python interpreter that is version 3.6.x, 3.7.x, 3.8.x or 3.9.x. This selection will add the Python interpreter path you selected to your project.
3. Select the Azure logo from the left-navigation pane.
  - You'll see your existing Azure resources in the Resources view.
  - Select the Azure subscription that you're using for this project and below you should see the Azure Function App.



4. Select the **Workspace (Local)** section located below your listed resources. Select the plus symbol and choose the **Create Function** button.



5. When prompted, choose **Create new project** and navigate to the **function-app** directory. Choose **Select**.

6. You'll be prompted to configure several settings:

- **Select a language** → choose Python.
- **Select a Python interpreter to create a virtual environment** → select the interpreter you set as the default earlier.
- **Select a template** → choose **Azure Blob Storage trigger** and give the trigger a name or accept the default name. Press **Enter** to confirm.
- **Select setting** → choose **+ Create new local app setting** from the dropdown menu.

- **Select subscription** → choose your Azure subscription with the storage account you created → select your storage account → then select the name of the storage input container (in this case, `input/{name}`). Press **Enter** to confirm.
- **Select how you would like to open your project** → choose **Open the project in the current window** from the dropdown menu.

7. Once you've completed these steps, VS Code will add a new Azure Function project with a `__init__.py` Python script. This script will be triggered when a file is uploaded to the **input** storage container:

Python

```
import logging

import azure.functions as func

def main(myblob: func.InputStream):
    logging.info(f"Python blob trigger function processed blob \n"
                 f"Name: {myblob.name}\n"
                 f"Blob Size: {myblob.length} bytes")
```

## Test the function

1. Press F5 to run the basic function. VS Code will prompt you to select a storage account to interface with.
2. Select the storage account you created and continue.
3. Open Azure Storage Explorer and upload the sample PDF document to the **input** container. Then check the VS Code terminal. The script should log that it was triggered by the PDF upload.

The screenshot shows a terminal window with the following log output:

```
[19-03-2020 13:26:48] Executing 'Functions.BlobTrigger1' (Reason='New blob detected: test/1025872, batch J5UAC91.pdf', Id=ba6e1371-be13-4f7f-8b99-094659600047)
[19-03-2020 13:26:48] INFO: Received FunctionInvocationRequest, request ID: 75e17627-bb9c-4506-baf2-7eec06b557e0, function ID: 9a00bc1f-ac2a-42a4-80fa-fda26ec6b29f, invocation ID: ba6e1371-be13-4f7f-8b99-094659600047
[19-03-2020 13:26:48] Python blob trigger function processed blob
Name: test/1025872, batch J5UAC91.pdf
Blob Size: 626862 bytes
[19-03-2020 13:26:48] INFO: Successfully processed FunctionInvocationRequest, request ID: 75e17627-bb9c-4506-baf2-7eec06b557e0, function ID: 9a00bc1f-ac2a-42a4-80fa-fda26ec6b29f, invocation ID: ba6e1371-be13-4f7f-8b99-094659600047
[19-03-2020 13:26:48] Executed 'Functions.BlobTrigger1' (Succeeded, Id=ba6e1371-be13-4f7f-8b99-094659600047)
```

4. Stop the script before continuing.

## Add document processing code

Next, you'll add your own code to the Python script to call the Document Intelligence service and parse the uploaded documents using the Document Intelligence [layout model](#).

1. In VS Code, navigate to the function's `requirements.txt` file. This file defines the dependencies for your script. Add the following Python packages to the file:

```
txt

cryptography
azure-functions
azure-storage-blob
azure-identity
requests
pandas
numpy
```

2. Then, open the `__init__.py` script. Add the following `import` statements:

```
Python

import logging
from azure.storage.blob import BlobServiceClient
import azure.functions as func
import json
import time
from requests import get, post
```

```
import os
import requests
from collections import OrderedDict
import numpy as np
import pandas as pd
```

3. You can leave the generated `main` function as-is. You'll add your custom code inside this function.

Python

```
# This part is automatically generated
def main(myblob: func.InputStream):
    logging.info(f"Python blob trigger function processed blob \n"
    f"Name: {myblob.name}\n"
    f"Blob Size: {myblob.length} bytes")
```

4. The following code block calls the Document Intelligence [Analyze Layout API](#) on the uploaded document. Fill in your endpoint and key values.

Python

```
# This is the call to the Document Intelligence endpoint
endpoint = r"Your Document Intelligence Endpoint"
apim_key = "Your Document Intelligence Key"
post_url = endpoint + "/formrecognizer/v2.1/layout/analyze"
source = myblob.read()

headers = {
# Request headers
'Content-Type': 'application/pdf',
'Ocp-Apim-Subscription-Key': apim_key,
}

text1=os.path.basename(myblob.name)
```

### ⓘ Important

Remember to remove the key from your code when you're done, and never post it publicly. For production, use a secure way of storing and accessing your credentials like [Azure Key Vault](#). For more information, see Azure AI services [security](#).

5. Next, add code to query the service and get the returned data.

Python

```

resp = requests.post(url=post_url, data=source, headers=headers)

if resp.status_code != 202:
    print("POST analyze failed:\n%s" % resp.text)
    quit()
print("POST analyze succeeded:\n%s" % resp.headers)
get_url = resp.headers["operation-location"]

wait_sec = 25

time.sleep(wait_sec)
# The layout API is async therefore the wait statement

resp = requests.get(url=get_url, headers={"Ocp-Apim-Subscription-Key": apim_key})

resp_json = json.loads(resp.text)

status = resp_json["status"]

if status == "succeeded":
    print("POST Layout Analysis succeeded:\n%s") 
    results = resp_json
else:
    print("GET Layout results failed:\n%s")
    quit()

results = resp_json

```

6. Add the following code to connect to the Azure Storage **output** container. Fill in your own values for the storage account name and key. You can get the key on the **Access keys** tab of your storage resource in the Azure portal.

Python

```

# This is the connection to the blob storage, with the Azure Python SDK
blob_service_client =
BlobServiceClient.from_connection_string("DefaultEndpointsProtocol=https;Acco
untName="Storage Account Name";AccountKey="storage account
key";EndpointSuffix=core.windows.net")
container_client=blob_service_client.get_container_client("output")

```

The following code parses the returned Document Intelligence response, constructs a .csv file, and uploads it to the **output** container.

 **Important**

You will likely need to edit this code to match the structure of your own documents.

Python

```
# The code below extracts the json format into tabular data.
# Please note that you need to adjust the code below to your form
structure.
# It probably won't work out-of-the-box for your specific form.
pages = results["analyzeResult"]["pageResults"]

def make_page(p):
    res=[]
    res_table=[]
    y=0
    page = pages[p]
    for tab in page["tables"]:
        for cell in tab["cells"]:
            res.append(cell)
            res_table.append(y)
        y=y+1

    res_table=pd.DataFrame(res_table)
    res=pd.DataFrame(res)
    res["table_num"]=res_table[0]
    h=res.drop(columns=["boundingBox","elements"])
    h.loc[:, "rownum"] = range(0, len(h))
    num_table=max(h["table_num"])
    return h, num_table, p

h, num_table, p= make_page(0)

for k in range(num_table+1):
    new_table=h[h.table_num==k]
    new_table.loc[:, "rownum"] = range(0, len(new_table))
    row_table=pages[p]["tables"][k]["rows"]
    col_table=pages[p]["tables"][k]["columns"]
    b=np.zeros((row_table,col_table))
    b=pd.DataFrame(b)
    s=0
    for i,j in zip(new_table["rowIndex"],new_table["columnIndex"]):
        b.loc[i,j]=new_table.loc[new_table.loc[s,"rownum"],"text"]
        s=s+1
```

7. Finally, the last block of code uploads the extracted table and text data to your blob storage element.

Python

```
# Here is the upload to the blob storage
tab1_csv=b.to_csv(header=False,index=False,mode='w')
```

```
name1=(os.path.splitext(text1)[0]) + '.csv'  
container_client.upload_blob(name=name1,data=tab1_csv)
```

## Run the function

1. Press F5 to run the function again.
2. Use Azure Storage Explorer to upload a sample PDF form to the **input** storage container. This action should trigger the script to run, and you should then see the resulting .csv file (displayed as a table) in the **output** container.

You can connect this container to Power BI to create rich visualizations of the data it contains.

## Next steps

In this tutorial, you learned how to use an Azure Function written in Python to automatically process uploaded PDF documents and output their contents in a more data-friendly format. Next, learn how to use Power BI to display the data.

### Microsoft Power BI

- [What is Document Intelligence?](#)
- Learn more about the [layout model](#)

# Document Models - Analyze Document

Reference

Service: Azure AI Services

API Version: 2024-11-30

Analyzes document with document model.

HTTP

```
POST {endpoint}/documentintelligence/documentModels/{modelId}:analyze?  
_overload=analyzeDocument&api-version=2024-11-30
```

With optional parameters:

HTTP

```
POST {endpoint}/documentintelligence/documentModels/{modelId}:analyze?  
_overload=analyzeDocument&api-version=2024-11-30&pages={pages}&locale=  
{locale}&stringIndexType={stringIndexType}&features={features}&queryFields=  
{queryFields}&outputContentFormat={outputContentFormat}&output={output}
```

## URI Parameters

[Expand table](#)

Name	In	Required	Type	Description
<b>endpoint</b>	path	True	string (uri)	The Document Intelligence service endpoint.
<b>modelId</b>	path	True	string maxLength: 64 pattern: ^[a-zA-Z0-9][a-zA-Z0-9._~-]{1,63}\$	Unique document model name.
<b>api-version</b>	query	True	string minLength: 1	The API version to use for this operation.
<b>features</b>	query		Document Analysis Feature[]	List of optional analysis features.
<b>locale</b>	query		string	Locale hint for text recognition and document analysis. Value may contain only the language

			code (ex. "en", "fr") or BCP 47 language tag (ex. "en-US").
<b>output</b>	query	AnalyzeOutputOption[]	Additional outputs to generate during analysis.
<b>output Content Format</b>	query	DocumentContentFormat	Format of the analyze result top-level content.
<b>pages</b>	query	string pattern: ^(\d+(-\d+)?)(,\s*(\d+(-\d+)?))*\$	1-based page numbers to analyze. Ex. "1-3,5,7-9"
<b>query Fields</b>	query	string[]	List of additional fields to extract. Ex. "NumberOfGuests,StoreNumber"
<b>StringIndexType</b>	query	StringIndexType	Method used to compute string offset and length.

## Request Body

[Expand table](#)

Name	Type	Description
base64Source	string (byte)	Base64 encoding of the document to analyze. Either urlSource or base64Source must be specified.
urlSource	string (uri)	Document URL to analyze. Either urlSource or base64Source must be specified.

## Responses

[Expand table](#)

Name	Type	Description
202 Accepted		<p>The request has been accepted for processing, but processing has not yet completed.</p> <p>Headers</p> <ul style="list-style-type: none"> <li>• Operation-Location: string</li> <li>• Retry-After: integer</li> </ul>

Other Status Codes	<a href="#">Document Intelligence</a>	An unexpected error response.
	<a href="#">Error</a>	
	<a href="#">Response</a>	

# Security

## Ocp-Apim-Subscription-Key

Type: apiKey

In: header

## OAuth2Auth

Type: oauth2

Flow: accessCode

Authorization URL: <https://login.microsoftonline.com/common/oauth2/authorize>

Token URL: <https://login.microsoftonline.com/common/oauth2/token>

## Scopes

[\[ \]](#) [Expand table](#)

Name	Description
<a href="https://cognitiveservices.azure.com/.default">https://cognitiveservices.azure.com/.default</a>	

# Examples

[\[ \]](#) [Expand table](#)

<a href="#">Analyze Document from Base64</a>
<a href="#">Analyze Document from Url</a>

## Analyze Document from Base64

### Sample request

HTTP

HTTP

POST

```
https://myendpoint.cognitiveservices.azure.com/documentintelligence/documentModels/prebuilt-layout:analyze?_overload=analyzeDocument&api-version=2024-11-30&pages=1-2,4&locale=en-US&stringIndexType=textElements
```

```
{
```

```
    "base64Source": "e2Jhc2U2NEVuY29kZWRQZGZ9"
```

```
}
```

## Sample response

Status code: 202

HTTP

**Operation-Location:**

```
https://myendpoint.cognitiveservices.azure.com/documentintelligence/documentModels/prebuilt-layout/analyzeResults/3b31320d-8bab-4f88-b19c-2322a7f11034?api-version=2024-11-30
```

## Analyze Document from Url

### Sample request

HTTP

HTTP

POST

```
https://myendpoint.cognitiveservices.azure.com/documentintelligence/documentModels/customModel:analyze?_overload=analyzeDocument&api-version=2024-11-30&pages=1-2,4&locale=en-US&stringIndexType=textElements
```

```
{
```

```
    "urlSource": "http://host.com/doc.pdf"
```

```
}
```

## Sample response

Status code: 202

HTTP

**Operation-Location:**

<https://myendpoint.cognitiveservices.azure.com/documentintelligence/documentModels/customModel/analyzeResults/3b31320d-8bab-4f88-b19c-2322a7f11034?api-version=2024-11-30>

## Definitions

 Expand table

Name	Description
AnalyzeDocumentRequest	Document analysis parameters.
AnalyzeOutputOption	Additional outputs to generate during analysis.
DocumentAnalysisFeature	Document analysis features to enable.
DocumentContentFormat	Format of the content in analyzed result.
DocumentIntelligenceError	The error object.
DocumentIntelligenceErrorResponse	Error response object.
DocumentIntelligenceInnerError	An object containing more specific information about the error.
StringIndexType	Method used to compute string offset and length.

## AnalyzeDocumentRequest

Object

Document analysis parameters.

 Expand table

Name	Type	Description
base64Source	string (byte)	Base64 encoding of the document to analyze. Either urlSource or base64Source must be specified.
urlSource	string (uri)	Document URL to analyze. Either urlSource or base64Source must be specified.

## AnalyzeOutputOption

Enumeration

Additional outputs to generate during analysis.

[+] Expand table

Value	Description
figures	Generate cropped images of detected figures.
pdf	Generate searchable PDF output.

## DocumentAnalysisFeature

Enumeration

Document analysis features to enable.

[+] Expand table

Value	Description
barcodes	Enable the detection of barcodes in the document.
formulas	Enable the detection of mathematical expressions in the document.
keyValuePairs	Enable the detection of general key value pairs (form fields) in the document.
languages	Enable the detection of the text content language.
ocrHighResolution	Perform OCR at a higher resolution to handle documents with fine print.
queryFields	Enable the extraction of additional fields via the queryFields query parameter.
styleFont	Enable the recognition of various font styles.

## DocumentContentFormat

Enumeration

Format of the content in analyzed result.

[+] Expand table

Value	Description
markdown	Markdown representation of the document content with section headings, tables, etc.

text	Plain text representation of the document content without any formatting.
------	---

## DocumentIntelligenceError

Object

The error object.

[ ] [Expand table](#)

Name	Type	Description
code	string	One of a server-defined set of error codes.
details	<a href="#">Document Intelligence Error[]</a>	An array of details about specific errors that led to this reported error.
innererror	<a href="#">Document Intelligence InnerError</a>	An object containing more specific information than the current object about the error.
message	string	A human-readable representation of the error.
target	string	The target of the error.

## DocumentIntelligenceErrorResponse

Object

Error response object.

[ ] [Expand table](#)

Name	Type	Description
error	<a href="#">Document Intelligence Error</a>	Error info.

## DocumentIntelligenceInnerError

Object

An object containing more specific information about the error.

[ ] [Expand table](#)

Name	Type	Description
code	string	One of a server-defined set of error codes.
innererror	Document Intelligence InnerError	Inner error.
message	string	A human-readable representation of the error.

## StringIndexType

Enumeration

Method used to compute string offset and length.

[ ] [Expand table](#)

Value	Description
textElements	User-perceived display character, or grapheme cluster, as defined by Unicode 8.0.0.
unicodeCodePoint	Character unit represented by a single unicode code point. Used by Python 3.
utf16CodeUnit	Character unit represented by a 16-bit Unicode code unit. Used by JavaScript, Java, and .NET.

# Error guide v4.0, v3.1, and v3.0

Article • 11/19/2024

Document Intelligence uses a unified design to represent all errors encountered in the REST APIs. Whenever an API operation returns a 4xx or 5xx status code, additional information about the error is returned in the response JSON body as follows:

JSON

```
{  
  "error": {  
    "code": "InvalidRequest",  
    "message": "Invalid request.",  
    "innererror": {  
      "code": "InvalidContent",  
      "message": "The file format is unsupported or corrupted. Refer to  
documentation for the list of supported formats."  
    }  
  }  
}
```

For long-running operations where multiple errors are encountered, the top-level error code is set to the most severe error, with the individual errors listed under the *error.details* property. In such scenarios, the *target* property of each individual error specifies the trigger of the error.

JSON

```
{  
  "status": "Failed",  
  "createdDateTime": "2021-07-14T10:17:51Z",  
  "lastUpdatedDateTime": "2021-07-14T10:17:51Z",  
  "error": {  
    "code": "InternalServerError",  
    "message": "An unexpected error occurred.",  
    "details": [  
      {  
        "code": "InternalServerError",  
        "message": "An unexpected error occurred."  
      },  
      {  
        "code": "InvalidContentDimensions",  
        "message": "The input image dimensions are out of range.  
Refer to documentation for supported image dimensions.",  
        "target": "2"  
      }  
    ]  
  }  
}
```

```
    }  
}
```

The top-level `error.code` property can be one of the following error code messages:

[Expand table](#)

Error Code	Message	Http Status
InvalidRequest	Invalid request.	400
InvalidArgument	Invalid argument.	400
Forbidden	Access forbidden due to policy or other configuration.	403
NotFound	Resource not found.	404
MethodNotAllowed	The requested HTTP method isn't allowed.	405
Conflict	The request couldn't be completed due to a conflict.	409
UnsupportedMediaType	Request content type isn't supported.	415
InternalServerError	An unexpected error occurred.	500
ServiceUnavailable	A transient error occurred. Try again.	503

When possible, more details are specified in the `inner.error` property.

[Expand table](#)

Top Error Code	Inner Error Code	Message
Conflict	ModelExists	A model with the provided name already exists.
Forbidden	AuthorizationFailed	Authorization failed: {details}
Forbidden	InvalidDataProtectionKey	Data protection key is invalid: {details}
Forbidden	OutboundAccessForbidden	The request contains a disallowed domain name or violates the current access control policy.
InternalServerError	Unknown	Unknown error.
InvalidArgument	InvalidContentSourceFormat	Invalid content source: {details}
InvalidArgument	InvalidParameter	The parameter {parameterName} is invalid: {details}

Top Error Code	Inner Error Code	Message
InvalidArgumentException	InvalidParameterLength	Parameter {parameterName} length must not exceed {maxChars} characters.
InvalidArgumentException	InvalidSasToken	The shared access signature (SAS) is invalid: {details}
InvalidArgumentException	ParameterMissing	The parameter {parameterName} is required.
InvalidRequest	ContentSourceNotAccessible	Content isn't accessible: {details}
InvalidRequest	ContentSourceTimeout	Timeout while receiving the file from client.
InvalidRequest	DocumentModelLimit	Account can't create more than {maximumModels} models.
InvalidRequest	DocumentModelLimitNeural	Account can't create more than 10 custom neural models per month. Contact support to request more capacity.
InvalidRequest	DocumentModelLimitComposed	Account can't create a model with more than {details} component models.
InvalidRequest	InvalidContent	The file is corrupted or format is unsupported. Refer to documentation for the list of supported formats.
InvalidRequest	InvalidContentDimensions	The input image dimensions are out of range. Refer to documentation for supported image dimensions.
InvalidRequest	InvalidContentSize	The input image is too large. Refer to documentation for the maximum file size.
InvalidRequest	InvalidFieldsDefinition	Invalid fields: {details}
InvalidRequest	InvalidTrainingContentSize	Training content contains {bytes} bytes. Training is limited to {maxBytes} bytes.
InvalidRequest	InvalidTrainingContentPageCount	Training content contains {pages} pages. Training is limited to {pages} pages.
InvalidRequest	ModelError	Couldn't analyze using a custom model: {details}
InvalidRequest	ModelError	Couldn't build the model: {details}

Top Error Code	Inner Error Code	Message
InvalidRequest	ModelComposeError	Couldn't compose the model: {details}
InvalidRequest	ModelNotReady	Model isn't ready for the requested operation. Wait for training to complete or check for operation errors.
InvalidRequest	ModelReadOnly	The requested model is read-only.
InvalidRequest	NotSupportedApiVersion	The requested operation requires {minimumApiVersion} or later.
InvalidRequest	OperationNotCancellable	The operation can no longer be canceled.
InvalidRequest	TrainingContentMissing	Training data is missing: {details}
InvalidRequest	UnsupportedContent	Content isn't supported: {details}
NotFound	ModelNotFound	The requested model wasn't found. It was deleted or still building.
NotFound	OperationNotFound	The requested operation wasn't found. The identifier is invalid or the operation is expired.

## Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback ↗](#) | [Get help at Microsoft Q&A](#)

# Analyzer

Reference

Service: Azure AI Services

API Version: 2.1

## Operations

 Expand table

<a href="#">Analyze Business Card</a>	Extract field text and semantic values from a given business card document. The input document must be of one of the supported content types - 'application/pdf'...
<a href="#">Analyze Id Document</a>	Extract field text and semantic values from a given ID document. The input document must be of one of the supported content types - 'application/pdf', 'image/jp...
<a href="#">Analyze Invoice</a>	Extract field text and semantic values from a given invoice document. The input document must be of one of the supported content types - 'application/pdf', 'ima...
<a href="#">Analyze Layout</a>	Extract text and layout information from a given document. The input document must be of one of the supported content types - 'application/pdf', 'image/jpeg', '...
<a href="#">Analyze Receipt</a>	Extract field text and semantic values from a given receipt document. The input document must be of one of the supported content types - 'application/pdf', 'ima...
<a href="#">Get Analyze Business Card Result</a>	Track the progress and obtain the result of the analyze business card operation.
<a href="#">Get Analyze Id Document Result</a>	Track the progress and obtain the result of the analyze ID operation.
<a href="#">Get Analyze Invoice Result</a>	Track the progress and obtain the result of the analyze invoice operation.
<a href="#">Get Analyze Layout Result</a>	Track the progress and obtain the result of the analyze layout operation
<a href="#">Get Analyze Receipt Result</a>	Track the progress and obtain the result of the analyze receipt operation.

# Azure Document Intelligence client library for .NET - version 1.0.0-beta.3

Article • 08/15/2024

Note: on July 2023, the Azure Cognitive Services Form Recognizer service was renamed to Azure AI Document Intelligence. Any mentions of Form Recognizer or Document Intelligence in documentation refer to the same Azure service.

Azure AI Document Intelligence is a cloud service that uses machine learning to analyze text and structured data from your documents. It includes the following main features:

- Layout - Extract text, selection marks, table structures, styles, and paragraphs, along with their bounding region coordinates from documents.
- Read - Read information about textual elements, such as page words and lines in addition to text language information.
- Prebuilt - Analyze data from certain types of common documents using prebuilt models. Supported documents include receipts, invoices, business cards, identity documents, US W2 tax forms, and more.
- Custom analysis - Build custom document models to analyze text, field values, selection marks, table structures, styles, and paragraphs from documents. Custom models are built with your own data, so they're tailored to your documents.
- Custom classification - Build custom classifier models that combine layout and language features to accurately detect and identify documents you process within your application.

[Source code ↗](#) | [Package \(NuGet\) ↗](#) | [API reference documentation ↗](#) | [Product documentation](#) | [Samples ↗](#)

## Getting started

This section should include everything a developer needs to do to install and create their first client connection *very quickly*.

### Install the package

Install the Azure Document Intelligence client library for .NET with [NuGet ↗](#):

.NET CLI

```
dotnet add package Azure.AI.DocumentIntelligence --prerelease
```

Note: This version of the client library defaults to the 2024-07-31-preview version of the service.

## Prerequisites

- An [Azure subscription ↗](#).
- A [Cognitive Services or Document Intelligence resource](#) to use this package.

## Create a Cognitive Services or Document Intelligence resource

Document Intelligence supports both [multi-service and single-service access](#). Create a Cognitive Services resource if you plan to access multiple cognitive services under a single endpoint and key. For Document Intelligence access only, create a Document Intelligence resource. Please note that you will need a single-service resource if you intend to use [Azure Active Directory authentication](#).

You can create either resource using:

- Option 1: [Azure Portal](#).
- Option 2: [Azure CLI](#).

Below is an example of how you can create a Document Intelligence resource using the CLI:

PowerShell

```
# Create a new resource group to hold the Document Intelligence resource
# If using an existing resource group, skip this step
az group create --name <your-resource-name> --location <location>
```

PowerShell

```
# Create the Form Recognizer resource
az cognitiveservices account create \
    --name <resource-name> \
    --resource-group <resource-group-name> \
    --kind FormRecognizer \
    --sku <sku> \
    --location <location> \
    --yes
```

For more information about creating the resource or how to get the location and sku information see [here](#).

## Authenticate the client

In order to interact with the Document Intelligence service, you'll need to create an instance of the [DocumentIntelligenceClient](#) class. An **endpoint** and a **credential** are necessary to instantiate the client object.

## Get the endpoint

You can find the endpoint for your Document Intelligence resource using the [Azure Portal](#) or the [Azure CLI](#):

PowerShell

```
# Get the endpoint for the Document Intelligence resource
az cognitiveservices account show --name "<resource-name>" --resource-group
"<resource-group-name>" --query "properties.endpoint"
```

Either a regional endpoint or a custom subdomain can be used for authentication. They are formatted as follows:

```
Regional endpoint: https://<region>.api.cognitive.microsoft.com/
Custom subdomain: https://<resource-name>.cognitiveservices.azure.com/
```

A regional endpoint is the same for every resource in a region. A complete list of supported regional endpoints can be consulted [here](#). Please note that regional endpoints do not support AAD authentication.

A custom subdomain, on the other hand, is a name that is unique to the Document Intelligence resource. They can only be used by [single-service resources](#).

## Get the API Key

The API key can be found in the [Azure Portal](#) or by running the following Azure CLI command:

PowerShell

```
az cognitiveservices account keys list --name "<resource-name>" --resource-group "<resource-group-name>"
```

## Create DocumentIntelligenceClient with AzureKeyCredential

Once you have the value for the API key, create an `AzureKeyCredential`. With the endpoint and key credential, you can create the [DocumentIntelligenceClient](#):

C#

```
string endpoint = "<endpoint>";
string apiKey = "<apiKey>";
var client = new DocumentIntelligenceClient(new Uri(endpoint), new
AzureKeyCredential(apiKey));
```

## Create DocumentIntelligenceClient with Azure Active Directory Credential

`AzureKeyCredential` authentication is used in the examples in this getting started guide, but you can also authenticate with Azure Active Directory using the [Azure Identity library](#). Note that regional endpoints do not support AAD authentication. Create a [custom subdomain](#) for your resource in order to use this type of authentication.

To use the [DefaultAzureCredential](#) provider shown below, or other credential providers provided with the Azure SDK, please install the `Azure.Identity` package:

.NET CLI

```
dotnet add package Azure.Identity
```

You will also need to [register a new AAD application](#) and [grant access](#) to Document Intelligence by assigning the `"Cognitive Services User"` role to your service principal.

Set the values of the client ID, tenant ID, and client secret of the AAD application as environment variables: AZURE\_CLIENT\_ID, AZURE\_TENANT\_ID, AZURE\_CLIENT\_SECRET.

C#

```
string endpoint = "<endpoint>";
var client = new DocumentIntelligenceClient(new Uri(endpoint), new
DefaultAzureCredential());
```

# Key concepts

## DocumentIntelligenceClient

`DocumentIntelligenceClient` provides operations for:

- Analyzing input documents using prebuilt and custom models through the `AnalyzeDocument` API.
- Detecting and identifying custom input documents with the `ClassifyDocument` API.

Sample code snippets are provided to illustrate using a `DocumentIntelligenceClient` [here](#).

More information about analyzing documents, including supported features, locales, and document types can be found in the [service documentation](#).

## DocumentIntelligenceAdministrationClient

`DocumentIntelligenceAdministrationClient` provides operations for:

- Building custom models to analyze specific fields you specify by labeling your custom documents.
- Compose a model from a collection of existing models.
- Managing models created in your account.
- Copying a custom model from one Document Intelligence resource to another.
- Getting or listing operations created within the last 24 hours.
- Building and managing document classification models to accurately detect and identify documents you process within your application.

See examples for [Build a Custom Model](#), [Manage Models](#), and [Build a Document Classifier](#).

Please note that models and classifiers can also be built using a graphical user interface such as the [Document Intelligence Studio](#).

## Thread safety

We guarantee that all client instance methods are thread-safe and independent of each other ([guideline](#)). This ensures that the recommendation of reusing client instances is always safe, even across threads.

## Additional concepts

[Client options ↗](#) | [Accessing the response ↗](#) | [Long-running operations ↗](#) | [Handling failures ↗](#) | [Diagnostics ↗](#) | [Mocking ↗](#) | [Client lifetime ↗](#)

## Examples

The following section provides several code snippets illustrating common patterns used in the Document Intelligence .NET API. Most of the snippets below make use of asynchronous service calls, but keep in mind that the Azure.AI.DocumentIntelligence package supports both synchronous and asynchronous APIs.

- [Extract Layout](#)
- [Use Prebuilt Models](#)
- [Build a Custom Model](#)
- [Manage Models](#)
- [Build a Document Classifier](#)
- [Classify a Document](#)

### Extract Layout

Extract text, selection marks, table structures, styles, and paragraphs, along with their bounding region coordinates from documents.

C#

```
Uri uriSource = new Uri("<uriSource>");

var content = new AnalyzeDocumentContent()
{
    UrlSource = uriSource
};

Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-layout",
content);
AnalyzeResult result = operation.Value;

foreach (DocumentPage page in result.Pages)
{
    Console.WriteLine($"Document Page {page.PageNumber} has
{page.Lines.Count} line(s), {page.Words.Count} word(s)," +
    $" and {page.SelectionMarks.Count} selection mark(s.");

    for (int i = 0; i < page.Lines.Count; i++)
    {
```

```

        DocumentLine line = page.Lines[i];

        Console.WriteLine($"  Line {i}:");
        Console.WriteLine($"    Content: '{line.Content}'");

        Console.Write("      Bounding polygon, with points ordered
clockwise:");
        for (int j = 0; j < line.Polygon.Count; j += 2)
        {
            Console.Write($" ({line.Polygon[j]}, {line.Polygon[j + 1]})");
        }

        Console.WriteLine();
    }

    for (int i = 0; i < page.SelectionMarks.Count; i++)
    {
        DocumentSelectionMark selectionMark = page.SelectionMarks[i];

        Console.WriteLine($"  Selection Mark {i} is
{selectionMark.State}");
        Console.WriteLine($"    State: {selectionMark.State}");

        Console.Write("      Bounding polygon, with points ordered
clockwise:");
        for (int j = 0; j < selectionMark.Polygon.Count; j++)
        {
            Console.Write($" ({selectionMark.Polygon[j]},
{selectionMark.Polygon[j + 1]})");
        }

        Console.WriteLine();
    }
}

for (int i = 0; i < result.Paragraphs.Count; i++)
{
    DocumentParagraph paragraph = result.Paragraphs[i];

    Console.WriteLine($"Paragraph {i}:");
    Console.WriteLine($"  Content: {paragraph.Content}");

    if (paragraph.Role != null)
    {
        Console.WriteLine($"    Role: {paragraph.Role}");
    }
}

foreach (DocumentStyle style in result.Styles)
{
    // Check the style and style confidence to see if text is handwritten.
    // Note that value '0.8' is used as an example.

    bool isHandwritten = style.IsHandwritten.HasValue && style.IsHandwritten
== true;
}

```

```

if (isHandwritten && style.Confidence > 0.8)
{
    Console.WriteLine($"Handwritten content found:");

    foreach (DocumentSpan span in style.Spans)
    {
        var handwrittenContent = result.Content.Substring(span.Offset,
span.Length);
        Console.WriteLine($"  {handwrittenContent}");
    }
}

for (int i = 0; i < result.Tables.Count; i++)
{
    DocumentTable table = result.Tables[i];

    Console.WriteLine($"Table {i} has {table.RowCount} rows and
{table.ColumnCount} columns.");

    foreach (DocumentTableCell cell in table.Cells)
    {
        Console.WriteLine($"  Cell ({cell.RowIndex}, {cell.ColumnIndex}) is
a '{cell.Kind}' with content: {cell.Content}");
    }
}

```

For more information, see [here](#).

## Use Prebuilt Models

Analyze data from certain types of common documents using prebuilt models provided by the Document Intelligence service.

For example, to analyze fields from an invoice, use the prebuilt Invoice model provided by passing the `prebuilt-invoice` model ID to the `AnalyzeDocumentAsync` method:

C#

```

Uri uriSource = new Uri("<uriSource>");

var content = new AnalyzeDocumentContent()
{
    UrlSource = uriSource
};

Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-invoice",
content);
AnalyzeResult result = operation.Value;

```

```
// To see the list of all the supported fields returned by service and its
// corresponding types for the
// prebuilt-invoice model, see:
// https://aka.ms/azsdk/formrecognizer/invoicefieldschema

for (int i = 0; i < result.Documents.Count; i++)
{
    Console.WriteLine($"Document {i}:");

    AnalyzedDocument document = result.Documents[i];

    if (document.Fields.TryGetValue("VendorName", out DocumentField vendorNameField)
        && vendorNameField.Type == DocumentFieldType.String)
    {
        string vendorName = vendorNameField.ValueString;
        Console.WriteLine($"Vendor Name: '{vendorName}', with confidence
{vendorNameField.Confidence}");
    }

    if (document.Fields.TryGetValue("CustomerName", out DocumentField customerNameField)
        && customerNameField.Type == DocumentFieldType.String)
    {
        string customerName = customerNameField.ValueString;
        Console.WriteLine($"Customer Name: '{customerName}', with confidence
{customerNameField.Confidence}");
    }

    if (document.Fields.TryGetValue("Items", out DocumentField itemsField)
        && itemsField.Type == DocumentFieldType.List)
    {
        foreach (DocumentField itemField in itemsField.ValueList)
        {
            Console.WriteLine("Item:");

            if (itemField.Type == DocumentFieldType.Dictionary)
            {
                IReadOnlyDictionary<string, DocumentField> itemFields =
itemField.ValueDictionary;

                if (itemFields.TryGetValue("Description", out DocumentField itemDescriptionField)
                    && itemDescriptionField.Type ==
DocumentFieldType.String)
                {
                    string itemDescription =
itemDescriptionField.ValueString;
                    Console.WriteLine($" Description: '{itemDescription}', with confidence
{itemDescriptionField.Confidence}");
                }

                if (itemFields.TryGetValue("Amount", out DocumentField itemAmountField)
```

```

        && itemAmountField.Type == DocumentFieldType.Currency)
    {
        CurrencyValue itemAmount =
itemAmountField.ValueCurrency;
        Console.WriteLine($" Amount:
'{itemAmount.CurrencySymbol}{itemAmount.Amount}', with confidence
{itemAmountField.Confidence}");
    }
}
}

if (document.Fields.TryGetValue("SubTotal", out DocumentField
subTotalField)
    && subTotalField.Type == DocumentFieldType.Currency)
{
    CurrencyValue subTotal = subTotalField.ValueCurrency;
    Console.WriteLine($"Sub Total: '{subTotal.CurrencySymbol}
{subTotal.Amount}', with confidence {subTotalField.Confidence}");
}

if (document.Fields.TryGetValue("TotalTax", out DocumentField
totalTaxField)
    && totalTaxField.Type == DocumentFieldType.Currency)
{
    CurrencyValue totalTax = totalTaxField.ValueCurrency;
    Console.WriteLine($"Total Tax: '{totalTax.CurrencySymbol}
{totalTax.Amount}', with confidence {totalTaxField.Confidence}");
}

if (document.Fields.TryGetValue("InvoiceTotal", out DocumentField
invoiceTotalField)
    && invoiceTotalField.Type == DocumentFieldType.Currency)
{
    CurrencyValue invoiceTotal = invoiceTotalField.ValueCurrency;
    Console.WriteLine($"Invoice Total: '{invoiceTotal.CurrencySymbol}
{invoiceTotal.Amount}', with confidence {invoiceTotalField.Confidence}");
}
}

```

You are not limited to invoices! There are a couple of prebuilt models to choose from, each of which has its own set of supported fields. More information about the supported document types can be found in the [service documentation](#).

For more information, see [here](#).

## Build a Custom Model

Build a custom model on your own document type. The resulting model can be used to analyze values from the types of documents it was built on.

C#

```
// For this sample, you can use the training documents found in the
`trainingFiles` folder.
// Upload the documents to your storage container and then generate a
container SAS URL. Note
// that a container URI without SAS is accepted only when the container is
public or has a
// managed identity configured.

// For instructions to set up documents for training in an Azure Blob
Storage Container, please see:
// https://aka.ms/azsdk/formrecognizer/buildcustommodel

string modelId = "<modelId>";
Uri blobContainerUri = new Uri("<blobContainerUri>");

// We are selecting the Template build mode in this sample. For more
information about the available
// build modes and their differences, see:
// https://aka.ms/azsdk/formrecognizer/buildmode

var content = new BuildDocumentModelContent(modelId,
DocumentBuildMode.Template)
{
    AzureBlobSource = new AzureBlobContentSource(blobContainerUri)
};

Operation<DocumentModelDetails> operation = await
client.BuildDocumentModelAsync(WaitUntil.Completed, content);
DocumentModelDetails model = operation.Value;

Console.WriteLine($"Model ID: {model.ModelId}");
Console.WriteLine($"Created on: {model.CreatedOn}");

Console.WriteLine("Document types the model can recognize:");
foreach (KeyValuePair<string, DocumentTypeDetails> docType in
model.DocTypes)
{
    Console.WriteLine($" Document type: '{docType.Key}', which has the
following fields:");
    foreach (KeyValuePair<string, DocumentFieldSchema> schema in
docType.Value.FieldSchema)
    {
        Console.WriteLine($" Field: '{schema.Key}', with confidence
{docType.Value.FieldConfidence[schema.Key]}");
    }
}
```

For more information, see [here](#).

## Manage Models

Manage the models stored in your account.

C#

```
// Check number of custom models in the Document Intelligence resource, and  
// the maximum number  
// of custom models that can be stored.  
  
ResourceDetails resourceDetails = await client.GetResourceInfoAsync();  
  
Console.WriteLine($"Resource has  
{resourceDetails.CustomDocumentModels.Count} custom models.");  
Console.WriteLine($"It can have at most  
{resourceDetails.CustomDocumentModels.Limit} custom models.");  
  
// Get a model by ID.  
string modelId = "<modelId>";  
DocumentModelDetails model = await client.GetModelAsync(modelId);  
  
Console.WriteLine($"Details about model with ID '{model.ModelId}':");  
Console.WriteLine($" Created on: {model.CreatedOn}");  
Console.WriteLine($" Expires on: {model.ExpiresOn}");  
  
// List up to 10 models currently stored in the resource.  
int count = 0;  
  
await foreach (DocumentModelDetails modelItem in client.GetModelsAsync())  
{  
    Console.WriteLine($"Model details:");  
    Console.WriteLine($" Model ID: {modelItem.ModelId}");  
    Console.WriteLine($" Description: {modelItem.Description}");  
    Console.WriteLine($" Created on: {modelItem.CreatedOn}");  
    Console.WriteLine($" Expires on: {model.ExpiresOn}");  
  
    if (++count == 10)  
    {  
        break;  
    }  
}
```

For more information, see [here](#).

## Build a Document Classifier

Build a document classifier by uploading custom training documents.

C#

```
// For this sample, you can use the training documents found in the  
`classifierTrainingFiles` folder.  
// Upload the documents to your storage container and then generate a
```

```

container SAS URL. Note
// that a container URI without SAS is accepted only when the container is
public or has a
// managed identity configured.

// For instructions to set up documents for training in an Azure Blob
Storage Container, please see:
// https://aka.ms/azsdk/formrecognizer/buildclassifiermodel

string classifierId = "<classifierId>";
Uri blobContainerUri = new Uri("<blobContainerUri>");
var sourceA = new AzureBlobContentSource(blobContainerUri) { Prefix = "IRS-
1040-A/train" };
var sourceB = new AzureBlobContentSource(blobContainerUri) { Prefix = "IRS-
1040-B/train" };
var docTypeA = new ClassifierDocumentTypeDetails() { AzureBlobSource =
sourceA };
var docTypeB = new ClassifierDocumentTypeDetails() { AzureBlobSource =
sourceB };
var docTypes = new Dictionary<string, ClassifierDocumentTypeDetails>()
{
    { "IRS-1040-A", docTypeA },
    { "IRS-1040-B", docTypeB }
};

var content = new BuildDocumentClassifierContent(classifierId, docTypes);

Operation<DocumentClassifierDetails> operation = await
client.BuildClassifierAsync(WaitUntil.Completed, content);
DocumentClassifierDetails classifier = operation.Value;

Console.WriteLine($"Classifier ID: {classifier.ClassifierId}");
Console.WriteLine($"Created on: {classifier.CreatedOn}");

Console.WriteLine("Document types the classifier can recognize:");
foreach (KeyValuePair<string, ClassifierDocumentTypeDetails> docType in
classifier.DocTypes)
{
    Console.WriteLine($"  {docType.Key}");
}

```

For more information, see [here](#).

## Classify a Document

Use document classifiers to accurately detect and identify documents you process within your application.

C#

```

string classifierId = "<classifierId>";
Uri uriSource = new Uri("<uriSource>");

var content = new ClassifyDocumentContent()
{
    UrlSource = uriSource
};

Operation<AnalyzeResult> operation = await
client.ClassifyDocumentAsync(WaitUntil.Completed, classifierId, content);
AnalyzeResult result = operation.Value;

Console.WriteLine($"Input was classified by the classifier with ID
'{result.ModelId}'.");

foreach (AnalyzedDocument document in result.Documents)
{
    Console.WriteLine($"Found a document of type: {document.DocType}");
}

```

For more information, see [here](#).

## Troubleshooting

### General

When you interact with the Document Intelligence client library using the .NET SDK, errors returned by the service will result in a `RequestFailedException` with the same HTTP status code returned by the [REST API](#) request.

For example, if you submit a receipt image with an invalid `Uri`, a `400` error is returned, indicating "Bad Request".

C#

```

var content = new AnalyzeDocumentContent()
{
    UrlSource = new Uri("http://invalid.uri")
};

try
{
    Operation<AnalyzeResult> operation = await
client.AnalyzeDocumentAsync(WaitUntil.Completed, "prebuilt-receipt",
content);
}
catch (RequestFailedException e)
{

```

```
        Console.WriteLine(e.ToString());
    }
```

You will notice that additional information is logged, like the client request ID of the operation.

#### Message:

```
Azure.RequestFailedException: Service request failed.  
Status: 400 (Bad Request)  
ErrorCode: InvalidRequest
```

#### Content:

```
{"error": {"code": "InvalidRequest", "message": "Invalid  
request.", "innererror": {"code": "InvalidContent", "message": "The file is  
corrupted or format is unsupported. Refer to documentation for the list of  
supported formats."}}}
```

#### Headers:

```
Transfer-Encoding: chunked  
x-envoy-upstream-service-time: REDACTED  
apim-request-id: REDACTED  
Strict-Transport-Security: REDACTED  
X-Content-Type-Options: REDACTED  
Date: Fri, 01 Oct 2021 02:55:44 GMT  
Content-Type: application/json; charset=utf-8
```

Error codes and messages raised by the Document Intelligence service can be found in the [service documentation](#).

## Setting up console logging

The simplest way to see the logs is to enable the console logging.

To create an Azure SDK log listener that outputs messages to console use the `AzureEventSourceListener.CreateConsoleLogger` method.

C#

```
// Setup a listener to monitor logged events.  
using AzureEventSourceListener listener =  
AzureEventSourceListener.CreateConsoleLogger();
```

To learn more about other logging mechanisms see [Diagnostics Samples](#).

# Next steps

Samples showing how to use the Document Intelligence library are available in this GitHub repository. Samples are provided for each main functional area:

- [Extract the layout of a document ↗](#)
- [Analyze a document with a prebuilt model ↗](#)
- [Build a custom model ↗](#)
- [Manage models ↗](#)
- [Classify a document ↗](#)
- [Build a document classifier ↗](#)
- [Get and List document model operations ↗](#)
- [Compose a model ↗](#)
- [Copy a custom model between Document Intelligence resources ↗](#)
- [Analyze a document with add-on capabilities ↗](#)
- [Extract the layout of a document as Markdown ↗](#)

# Contributing

This project welcomes contributions and suggestions. Most contributions require you to agree to a Contributor License Agreement (CLA) declaring that you have the right to, and actually do, grant us the rights to use your contribution. For details, visit [cla.microsoft.com](https://cla.microsoft.com) ↗.

When you submit a pull request, a CLA-bot will automatically determine whether you need to provide a CLA and decorate the PR appropriately (e.g., label, comment). Simply follow the instructions provided by the bot. You will only need to do this once across all repos using our CLA.

This project has adopted the [Microsoft Open Source Code of Conduct](#) ↗. For more information see the [Code of Conduct FAQ](#) ↗ or contact [opencode@microsoft.com](mailto:opencode@microsoft.com) with any additional questions or comments.

# Azure DocumentIntelligence client library for Java - version 1.0.0-beta.4

Article • 08/15/2024

Azure Document Intelligence ([previously known as Form Recognizer](#)) is a cloud service that uses machine learning to analyze text and structured data from your documents. It includes the following main features:

- Layout - Analyze text, table structures, and selection marks, along with their bounding region coordinates, from documents.
- Prebuilt - Analyze data from certain types of common documents (such as receipts, invoices, identity documents or US W2 tax forms) using prebuilt models.
- Custom - Build custom models to extract text, field values, selection marks, and table data from documents. Custom models are built with your own data, so they're tailored to your documents.
- Read - Read information about textual elements, such as page words and lines in addition to text language information.
- Classifiers - Build custom classifiers to categorize documents into predefined classes.

[Source code](#) | [Package \(Maven\)](#) | [API reference documentation](#) | [Product Documentation](#) | [Samples](#)

## Getting started

### Prerequisites

- [Java Development Kit \(JDK\)](#) with version 8 or above
  - Here are details about [Java 8 client compatibility with Azure Certificate Authority](#).
- [Azure Subscription](#)
- [AI Services or Document Intelligence account](#) to use this package.

### Adding the package to your product

XML

```
<dependency>
    <groupId>com.azure</groupId>
    <artifactId>azure-ai-documentintelligence</artifactId>
```

```
<version>1.0.0-beta.4</version>
</dependency>
```

Note: This version of the client library defaults to the "2024-07-31-preview" version of the service.

This table shows the relationship between SDK versions and supported API versions of the service:

[+] Expand table

SDK version	Supported API version of service
1.0.0-beta.1	2023-10-31-preview
1.0.0-beta.2	2024-02-29-preview
1.0.0-beta.3	2024-02-29-preview
1.0.0-beta.4	2024-07-31-preview

Note: Please rely on the older `azure-ai-formrecognizer` library through the older service API versions for retired models, such as "`prebuilt-businessCard`" and "`prebuilt-document`". For more information, see [Changelog](#). The below table describes the relationship of each client and its supported API version(s):

[+] Expand table

API version	Supported clients
2023-10-31-preview, 2024-02-29-preview, 2024-07-31-preview	DocumentIntelligenceClient and DocumentIntelligenceAsyncClient
2023-07-31	DocumentAnalysisClient and DocumentModelAdministrationClient in <code>azure-ai-formrecognizer</code> SDK

Please see the [Migration Guide](#) for more information about migrating from `azure-ai-formrecognizer` to `azure-ai-documentintelligence`.

## Authentication

In order to interact with the Azure Document Intelligence Service you'll need to create an instance of client class, [DocumentIntelligenceAsyncClient](#) or [DocumentIntelligenceClient](#) by using [DocumentIntelligenceClientBuilder](#). To configure a client for use with Azure DocumentIntelligence, provide a valid endpoint URI to an Azure DocumentIntelligence resource along with a corresponding key credential, token credential, or [Azure Identity](#) credential that's authorized to use the Azure DocumentIntelligence resource.

## Create an Azure DocumentIntelligence client with key credential

Get Azure DocumentIntelligence `key` credential from the Azure Portal.

Java

```
DocumentIntelligenceClient documentIntelligenceClient = new  
DocumentIntelligenceClientBuilder()  
    .credential(new AzureKeyCredential("{key}"))  
    .endpoint("{endpoint}")  
    .buildClient();
```

or

Java

```
DocumentIntelligenceAdministrationClient client =  
    new DocumentIntelligenceAdministrationClientBuilder()  
        .credential(new AzureKeyCredential("{key}"))  
        .endpoint("{endpoint}")  
        .buildClient();
```

## Create an Azure DocumentIntelligence client with Azure Active Directory credential

Azure SDK for Java supports an Azure Identity package, making it easy to get credentials from Microsoft identity platform.

Authentication with AAD requires some initial setup:

- Add the Azure Identity package

XML

```
<dependency>  
    <groupId>com.azure</groupId>  
    <artifactId>azure-identity</artifactId>
```

```
<version>1.13.2</version>
</dependency>
```

After setup, you can choose which type of [credential](#) from azure-identity to use. As an example, [DefaultAzureCredential](#) can be used to authenticate the client: Set the values of the client ID, tenant ID, and client secret of the AAD application as environment variables: `AZURE_CLIENT_ID`, `AZURE_TENANT_ID`, `AZURE_CLIENT_SECRET`.

Authorization is easiest using [DefaultAzureCredential](#). It finds the best credential to use in its running environment. For more information about using Azure Active Directory authorization with DocumentIntelligence service, please refer to [the associated documentation](#).

Java

```
DocumentIntelligenceAsyncClient documentIntelligenceAsyncClient = new
DocumentIntelligenceClientBuilder()
    .credential(new DefaultAzureCredentialBuilder().build())
    .endpoint("{endpoint}")
    .buildAsyncClient();
```

## Key concepts

### DocumentAnalysisClient

The [DocumentAnalysisClient](#) and [DocumentAnalysisAsyncClient](#) provide both synchronous and asynchronous operations for analyzing input documents using custom and prebuilt models through the `beginAnalyzeDocument` API. See a full list of supported models [here](#).

Sample code snippets to illustrate using a DocumentAnalysisClient [here](#). More information about analyzing documents, including supported features, locales, and document types can be found [here](#).

### DocumentModelAdministrationClient

The [DocumentModelAdministrationClient](#) and [DocumentModelAdministrationAsyncClient](#) provide both synchronous and asynchronous operations

- Build custom document analysis models to analyze text content, fields, and values found in your custom documents. See example [Build a document model](#). A

`DocumentModelDetails` is returned indicating the document types that the model can analyze, along with the fields and schemas it will extract.

- Managing models created in your account by building, listing, deleting, and see the limit of custom models your account. See example [Manage models](#).
- Copying a custom model from one Document Intelligence resource to another.
- Creating a composed model from a collection of existing built models.
- Listing document model operations associated with the Document Intelligence resource.

Sample code snippets are provided to illustrate using a `DocumentModelAdministrationClient` [here](#).

## Long-running operations

Long-running operations are operations that consist of an initial request sent to the service to start an operation, followed by polling the service at intervals to determine whether the operation has completed or failed, and if it has succeeded, to get the result.

Methods that build models, analyze values from documents, or copy and compose models are modeled as long-running operations. The client exposes a

`begin<MethodName>` method that returns a `SyncPoller` or `PollerFlux` instance. Callers should wait for the operation to be completed by calling `getFinalResult()` on the returned operation from the `begin<MethodName>` method. Sample code snippets are provided to illustrate using long-running operations [below](#).

## Examples

The following section provides several code snippets covering some of the most common Document Intelligence tasks, including:

- [Analyze Layout](#)
- [Use Prebuilt Models](#)
- [Build a Document Model](#)
- [Analyze Documents using a Custom Model](#)
- [Manage Your Models](#)

## Analyze Layout

Analyze text, table structures, and selection marks like radio buttons and check boxes, along with their bounding box coordinates from documents without the need to build a model.

Java

```
File layoutDocument = new File("local/file_path/filename.png");
Path filePath = layoutDocument.toPath();
BinaryData layoutDocumentData = BinaryData.fromFile(filePath, (int)
layoutDocument.length());

SyncPoller<AnalyzeResultOperation, AnalyzeResult> analyzeLayoutResultPoller
=
    documentIntelligenceClient.beginAnalyzeDocument("prebuilt-layout",
        null,
        null,
        null,
        null,
        null,
        null,
        null,
        null,
        new
AnalyzeDocumentRequest().setBase64Source(Files.readAllBytes(layoutDocument.t
oPath())));

AnalyzeResult analyzeLayoutResult =
analyzeLayoutResultPoller.getFinalResult();

// pages
analyzeLayoutResult.getPages().forEach(documentPage -> {
    System.out.printf("Page has width: %.2f and height: %.2f, measured with
unit: %s%n",
        documentPage.getWidth(),
        documentPage.getHeight(),
        documentPage.getUnit());

    // lines
    documentPage.getLines().forEach(documentLine ->
        System.out.printf("Line '%s' is within a bounding box %s.%n",
            documentLine.getContent(),
            documentLine.getPolygon().toString()));

    // selection marks
    documentPage.getSelectionMarks().forEach(documentSelectionMark ->
        System.out.printf("Selection mark is '%s' and is within a bounding
box %s with confidence %.2f.%n",
            documentSelectionMark.getState().toString(),
            documentSelectionMark.getPolygon().toString(),
            documentSelectionMark.getConfidence()));
});

// tables
List<DocumentTable> tables = analyzeLayoutResult.getTables();
for (int i = 0; i < tables.size(); i++) {
    DocumentTable documentTable = tables.get(i);
    System.out.printf("Table %d has %d rows and %d columns.%n", i,
documentTable.getRowCount(),
    documentTable.getColumnCount());
```

```

documentTable.getCells().forEach(documentTableCell -> {
    System.out.printf("Cell '%s', has row index %d and column index
%d.%n", documentTableCell.getContent(),
documentTableCell.getRowIndex(),
documentTableCell.getColumnIndex());
});
System.out.println();
}

```

## Use Prebuilt Models

Extract fields from select document types such as receipts, invoices, and identity documents using prebuilt models provided by the Document Intelligence service. Supported prebuilt models are:

- Analyze receipts using the `prebuilt-receipt` model (fields recognized by the service can be found [here](#))
- Analyze invoices using the `prebuilt-invoice` model (fields recognized by the service can be found [here](#)).
- Analyze identity documents using the `prebuilt-idDocuments` model (fields recognized by the service can be found [here](#)).
- Analyze US W2 tax forms using the `prebuilt-tax.us.w2` model. [Supported fields](#).

For example, to analyze fields from a sales receipt, into the `beginAnalyzeDocumentFromUrl` method:

Java

```

File sourceFile = new File("../documentintelligence/azure-ai-
documentintelligence/src/samples/resources/"
+ "sample-forms/receipts/contoso-allinone.jpg");

SyncPoller<AnalyzeResultOperation, AnalyzeResult> analyzeReceiptPoller =
    documentIntelligenceClient.beginAnalyzeDocument("prebuilt-receipt",
        null,
        null,
        null,
        null,
        null,
        null,
        null,
        new
    AnalyzeDocumentRequest().setBase64Source(Files.readAllBytes(sourceFile.toPath())));
}

AnalyzeResult receiptResults = analyzeReceiptPoller.getFinalResult();

for (int i = 0; i < receiptResults.getDocuments().size(); i++) {

```

```

Document analyzedReceipt = receiptResults.getDocuments().get(i);
Map<String, DocumentField> receiptFields = analyzedReceipt.getFields();
System.out.printf("----- Analyzing receipt info %d -----%n",
i);
DocumentField merchantNameField = receiptFields.get("MerchantName");
if (merchantNameField != null) {
    if (DocumentFieldType.STRING == merchantNameField.getType()) {
        String merchantName = merchantNameField.getValueString();
        System.out.printf("Merchant Name: %s, confidence: %.2f%n",
            merchantName, merchantNameField.getConfidence());
    }
}

DocumentField merchantPhoneNumberField =
receiptFields.get("MerchantPhoneNumber");
if (merchantPhoneNumberField != null) {
    if (DocumentFieldType.PHONE_NUMBER ==
merchantPhoneNumberField.getType()) {
        String merchantAddress =
merchantPhoneNumberField.getValuePhoneNumber();
        System.out.printf("Merchant Phone number: %s, confidence:
%.2f%n",
            merchantAddress, merchantPhoneNumberField.getConfidence());
    }
}

DocumentField merchantAddressField =
receiptFields.get("MerchantAddress");
if (merchantAddressField != null) {
    if (DocumentFieldType.STRING == merchantAddressField.getType()) {
        String merchantAddress = merchantAddressField.getValueString();
        System.out.printf("Merchant Address: %s, confidence: %.2f%n",
            merchantAddress, merchantAddressField.getConfidence());
    }
}

DocumentField transactionDateField =
receiptFields.get("TransactionDate");
if (transactionDateField != null) {
    if (DocumentFieldType.DATE == transactionDateField.getType()) {
        LocalDate transactionDate = transactionDateField.getValueDate();
        System.out.printf("Transaction Date: %s, confidence: %.2f%n",
            transactionDate, transactionDateField.getConfidence());
    }
}
}

```

For more information and samples using prebuilt models, see:

- [Identity Documents ↗](#)
- [Invoices ↗](#)
- [Receipts sample ↗](#)

# Build a document model

Build a machine-learned model on your own document type. The resulting model will be able to analyze values from the types of documents it was built on. Provide a container SAS url to your Azure Storage Blob container where you're storing the training documents. See details on setting this up in the [service quickstart documentation](#).

## Note

You can use the [Document Intelligence Studio preview](#) for creating a labeled file for your training forms. More details on setting up a container and required file structure can be found in [here](#).

Java

```
// Build custom document analysis model
String blobContainerUrl = "{SAS_URL_of_your_container_in_blob_storage}";
// The shared access signature (SAS) Url of your Azure Blob Storage
// container with your forms.
SyncPoller<DocumentModelBuildOperationDetails, DocumentModelDetails>
buildOperationPoller =
    administrationClient.beginBuildDocumentModel(new
BuildDocumentModelRequest("modelID", DocumentBuildMode.TEMPLATE)
    .setAzureBlobSource(new AzureBlobContentSource(blobContainerUrl)));

DocumentModelDetails documentModelDetails =
buildOperationPoller.getFinalResult();

// Model Info
System.out.printf("Model ID: %s%n", documentModelDetails.getModelId());
System.out.printf("Model Description: %s%n",
documentModelDetails.getDescription());
System.out.printf("Model created on: %s%n%n",
documentModelDetails.getCreatedDateTime());

System.out.println("Document Fields:");
documentModelDetails.getDocTypes().forEach((key, documentTypeDetails) -> {
    documentTypeDetails.getFieldSchema().forEach((field,
documentFieldSchema) -> {
        System.out.printf("Field: %s", field);
        System.out.printf("Field type: %s", documentFieldSchema.getType());
        System.out.printf("Field confidence: %.2f",
documentTypeDetails.getFieldConfidence().get(field));
    });
});
```

# Analyze Documents using a Custom Model

Analyze the key/value pairs and table data from documents. These models are built with your own data, so they're tailored to your documents. You should only analyze documents of the same doc type that the custom model was built on.

Java

```
String documentUrl = "{document-url}";
String modelId = "{custom-built-model-ID}";
SyncPoller<AnalyzeResultOperation, AnalyzeResult> analyzeDocumentPoller =
documentIntelligenceClient.beginAnalyzeDocument(modelId,
    "1",
    "en-US",
    StringIndexType.TEXT_ELEMENTS,
    Arrays.asList(DocumentAnalysisFeature.LANGUAGES),
    null,
    ContentFormat.TEXT,
    null,
    new AnalyzeDocumentRequest().setUrlSource(documentUrl));

AnalyzeResult analyzeResult = analyzeDocumentPoller.getFinalResult();

for (int i = 0; i < analyzeResult.getDocuments().size(); i++) {
    final Document analyzedDocument = analyzeResult.getDocuments().get(i);
    System.out.printf("----- Analyzing custom document %d -----%n", i);
    System.out.printf("Analyzed document has doc type %s with confidence : %.2f%n",
        analyzedDocument.getDocType(), analyzedDocument.getConfidence());
}

analyzeResult.getPages().forEach(documentPage -> {
    System.out.printf("Page has width: %.2f and height: %.2f, measured with
unit: %s%n",
        documentPage.getWidth(),
        documentPage.getHeight(),
        documentPage.getUnit());

    // lines
    documentPage.getLines().forEach(documentLine ->
        System.out.printf("Line '%s' is within a bounding polygon %s.%n",
            documentLine.getContent(),
            documentLine.getPolygon()));

    // words
    documentPage.getWords().forEach(documentWord ->
        System.out.printf("Word '%s' has a confidence score of %.2f.%n",
            documentWord.getContent(),
            documentWord.getConfidence()));
});

// tables
List<DocumentTable> tables = analyzeResult.getTables();
for (int i = 0; i < tables.size(); i++) {
```

```

DocumentTable documentTable = tables.get(i);
System.out.printf("Table %d has %d rows and %d columns.%n", i,
documentTable.getRowCount(),
documentTable.getColumnCount());
documentTable.getCells().forEach(documentTableCell -> {
System.out.printf("Cell '%s', has row index %d and column index
%d.%n",
documentTableCell.getContent(),
documentTableCell.getRowIndex(),
documentTableCell.getColumnIndex());
});
System.out.println();
}

```

## Manage your models

Manage the models in your Document Intelligence account.

Java

```

ResourceDetails resourceDetails = administrationClient.getResourceInfo();
System.out.printf("The resource has %s models, and we can have at most %s
models.%n",
resourceDetails.getCustomDocumentModels().getCount(),
resourceDetails.getCustomDocumentModels().getLimit());

// Next, we get a paged list of all of our models
PagedIterable<DocumentModelDetails> customDocumentModels =
administrationClient.listModels();
System.out.println("We have following models in the account:");
customDocumentModels.forEach(documentModelInfo -> {
System.out.println();
// get custom document analysis model info
DocumentModelDetails documentModel =
administrationClient.getModel(documentModelInfo.getModelId());
System.out.printf("Model ID: %s%n", documentModel.getModelId());
System.out.printf("Model Description: %s%n",
documentModel.getDescription());
System.out.printf("Model created on: %s%n",
documentModel.getCreatedDateTime());
if (documentModel.getDocTypes() != null) {
documentModel.getDocTypes().forEach((key, documentTypeDetails) -> {
documentTypeDetails.getFieldSchema().forEach((field,
documentFieldSchema) -> {
System.out.printf("Field: %s, ", field);
System.out.printf("Field type: %s, ",
documentFieldSchema.getType());
if (documentTypeDetails.getFieldConfidence() != null) {
System.out.printf("Field confidence: %.2f%n",
documentTypeDetails.getFieldConfidence().get(field));
}
}
)
}
}
}

```

```
        }
    });
}
});
```

For more detailed examples, refer to [samples ↗](#).

## Troubleshooting

### Enable client logging

You can set the `AZURE_LOG_LEVEL` environment variable to view logging statements made in the client library. For example, setting `AZURE_LOG_LEVEL=2` would show all informational, warning, and error log messages. The log levels can be found here: [log levels ↗](#).

### Default HTTP Client

All client libraries by default use the Netty HTTP client. Adding the above dependency will automatically configure the client library to use the Netty HTTP client. Configuring or changing the HTTP client is detailed in the [HTTP clients wiki ↗](#).

### Default SSL library

All client libraries, by default, use the Tomcat-native Boring SSL library to enable native-level performance for SSL operations. The Boring SSL library is an uber jar containing native libraries for Linux / macOS / Windows, and provides better performance compared to the default SSL implementation within the JDK. For more information, including how to reduce the dependency size, refer to the [performance tuning ↗](#) section of the wiki.

## Next steps

- Samples are explained in detail [here ↗](#).

## Contributing

For details on contributing to this repository, see the [contributing guide ↗](#).

1. Fork it
2. Create your feature branch (`git checkout -b my-new-feature`)
3. Commit your changes (`git commit -am 'Add some feature'`)
4. Push to the branch (`git push origin my-new-feature`)
5. Create new Pull Request

# Azure DocumentIntelligence (formerly FormRecognizer) REST client library for JavaScript - version 1.0.0-beta.3

Article • 08/21/2024

Extracts content, layout, and structured data from documents.

Please rely heavily on our [REST client docs](#) to use this library

NOTE: Form Recognizer has been rebranded to Document Intelligence. Please check the [Migration Guide from @azure/ai-form-recognizer to @azure-rest/ai-document-intelligence](#).

Key links:

- [Source code](#)
- [Package \(NPM\)](#)
- [API reference documentation](#)
- [Samples](#)
- [Changelog](#)
- [Migration Guide from Form Recognizer](#)

This version of the client library defaults to the "2024-07-31-preview" version of the service.

This table shows the relationship between SDK versions and supported API versions of the service:

expand Expand table

SDK version	Supported API version of service
1.0.0-beta.3	2024-07-31-preview
1.0.0-beta.2	2024-02-29-preview
1.0.0-beta.1	2023-10-31-preview

Please rely on the older `@azure/ai-form-recognizer` library through the older service API versions for retired models, such as `"prebuilt-businessCard"` and `"prebuilt-document"`. For more information, see [Changelog](#).

The below table describes the relationship of each client and its supported API version(s):

[Expand table](#)

Service API version	Supported clients	Package
2024-07-31-preview	DocumentIntelligenceClient	@azure-rest/ai-document-intelligence version 1.0.0-beta.3
2024-02-29-preview	DocumentIntelligenceClient	@azure-rest/ai-document-intelligence version 1.0.0-beta.2
2023-10-31-preview	DocumentIntelligenceClient	@azure-rest/ai-document-intelligence version 1.0.0-beta.1
2023-07-31	DocumentAnalysisClient and DocumentModelAdministrationClient	@azure/ai-form-recognizer version ^5.0.0
2022-08-01	DocumentAnalysisClient and DocumentModelAdministrationClient	@azure/ai-form-recognizer version ^4.0.0

## Getting started

### Currently supported environments

- LTS versions of Node.js

### Prerequisites

- You must have an [Azure subscription](#) to use this package.

### Install the `@azure-rest/ai-document-intelligence` package

Install the Azure DocumentIntelligence(formerlyFormRecognizer) REST client REST client library for JavaScript with `npm`:

Bash

```
npm install @azure-rest/ai-document-intelligence
```

### Create and authenticate a `DocumentIntelligenceClient`

To use an [Azure Active Directory \(AAD\) token credential](#), provide an instance of the desired credential type obtained from the [@azure/identity](#) library.

To authenticate with AAD, you must first `npm` install [@azure/identity](#)

After setup, you can choose which type of [credential](#) from [@azure/identity](#) to use. As an example, [DefaultAzureCredential](#) can be used to authenticate the client.

Set the values of the client ID, tenant ID, and client secret of the AAD application as environment variables: AZURE\_CLIENT\_ID, AZURE\_TENANT\_ID, AZURE\_CLIENT\_SECRET

## Using a Token Credential

ts

```
import DocumentIntelligence from "@azure-rest/ai-document-intelligence";

const client = DocumentIntelligence(
  process.env["DOCUMENT_INTELLIGENCE_ENDPOINT"],
  new DefaultAzureCredential(),
);
```

## Using an API KEY

ts

```
import DocumentIntelligence from "@azure-rest/ai-document-intelligence";

const client = DocumentIntelligence(process.env["DOCUMENT_INTELLIGENCE_ENDPOINT"],
{
  key: process.env["DOCUMENT_INTELLIGENCE_API_KEY"],
});
```

## Document Models

### Analyze prebuilt-layout (urlSource)

ts

```
const initialResponse = await client
  .path("/documentModels/{modelId}:analyze", "prebuilt-layout")
  .post({
    contentType: "application/json",
    body: {
```

```
urlSource:  
    "https://raw.githubusercontent.com/Azure/azure-sdk-for-  
js/6704eff082aaaf2d97c1371a28461f512f8d748a/sdk/formrecognizer/ai-form-  
recognizer/assets/forms/Invoice_1.pdf",  
},  
queryParameters: { locale: "en-IN" },  
});
```

## Analyze prebuilt-layout (base64Source)

ts

```
import fs from "fs";  
import path from "path";  
  
const filePath = path.join(ASSET_PATH, "forms", "Invoice_1.pdf");  
const base64Source = fs.readFileSync(filePath, { encoding: "base64" });  
const initialResponse = await client  
    .path("/documentModels/{modelId}:analyze", "prebuilt-layout")  
    .post({  
        contentType: "application/json",  
        body: {  
            base64Source,  
        },  
        queryParameters: { locale: "en-IN" },  
    });
```

Continue creating the poller from initial response

ts

```
import {  
    getLongRunningPoller,  
    AnalyzeResultOperationOutput,  
    isUnexpected,  
} from "@azure-rest/ai-document-intelligence";  
  
if (isUnexpected(initialResponse)) {  
    throw initialResponse.body.error;  
}  
const poller = await getLongRunningPoller(client, initialResponse);  
const result = (await poller.pollUntilDone()).body as  
AnalyzeResultOperationOutput;  
console.log(result);  
// {  
//     status: 'succeeded',  
//     createdDateTime: '2023-11-10T13:31:31Z',  
//     lastUpdatedDateTime: '2023-11-10T13:31:34Z',  
//     analyzeResult: {  
//         apiVersion: '2023-10-31-preview',  
//         .
```

```
// .
// .
//   contentFormat: 'text'
// }
// }
```

## Markdown content format

Supports output with Markdown content format along with the default plain *text*. For now, this is only supported for "prebuilt-layout". Markdown content format is deemed a more friendly format for LLM consumption in a chat or automation use scenario.

Service follows the GFM spec ([GitHub Flavored Markdown ↗](#)) for the Markdown format. Also introduces a new *contentFormat* property with value "text" or "markdown" to indicate the result content format.

ts

```
import DocumentIntelligence from "@azure-rest/ai-document-intelligence";
const client = DocumentIntelligence(process.env["DOCUMENT_INTELLIGENCE_ENDPOINT"],
{
  key: process.env["DOCUMENT_INTELLIGENCE_API_KEY"],
});

const initialResponse = await client
  .path("/documentModels/{modelId}:analyze", "prebuilt-layout")
  .post({
    contentType: "application/json",
    body: {
      urlSource:
        "https://raw.githubusercontent.com/Azure/azure-sdk-for-
js/6704eff082aaaf2d97c1371a28461f512f8d748a/sdk/formrecognizer/ai-form-
recognizer/assets/forms/Invoice_1.pdf",
    },
    queryParameters: { outputContentFormat: "markdown" }, // <-- new query
parameter
  });

```

## Query Fields

When this feature flag is specified, the service will further extract the values of the fields specified via the *queryFields* query parameter to supplement any existing fields defined by the model as fallback.

ts

```
await client.path("/documentModels/{modelId}:analyze", "prebuilt-layout").post({
  contentType: "application/json",
  body: { urlSource: "..." },
  queryParameters: {
    features: ["queryFields"],
    queryFields: ["NumberOfGuests", "StoreNumber"],
  }, // <-- new query parameter
});
```

## Split Options

In the previous API versions supported by the older `@azure/ai-form-recognizer` library, document splitting and classification operation (`"/documentClassifiers/{classifierId}:analyze"`) always tried to split the input file into multiple documents.

To enable a wider set of scenarios, service introduces a "split" query parameter with the new "2023-10-31-preview" service version. The following values are supported:

- `split: "auto"`

Let service determine where to split.

- `split: "none"`

The entire file is treated as a single document. No splitting is performed.

- `split: "perPage"`

Each page is treated as a separate document. Each empty page is kept as its own document.

## Document Classifiers #Build

ts

```
import {
  DocumentClassifierBuildOperationDetailsOutput,
  getLongRunningPoller,
  isUnexpected,
} from "@azure-rest/ai-document-intelligence";

const containerSasUrl = (): string =>
  process.env["DOCUMENT_INTELLIGENCE_TRAINING_CONTAINER_SAS_URL"];
const initialResponse = await client.path("/documentClassifiers:build").post({
  body: {
```

```

classifierId: `customClassifier${getRandomNumber()}`,
description: "Custom classifier description",
docTypes: {
  foo: {
    azureBlobSource: {
      containerUrl: containerSasUrl(),
    },
  },
  bar: {
    azureBlobSource: {
      containerUrl: containerSasUrl(),
    },
  },
},
});

if (isUnexpected(initialResponse)) {
  throw initialResponse.body.error;
}
const poller = await getLongRunningPoller(client, initialResponse);
const response = (await poller.pollUntilDone())
  .body as DocumentClassifierBuildOperationDetailsOutput;
console.log(response);
// {
//   operationId: '31466834048_f3ee629e-73fb-48ab-993b-1d55d73ca460',
//   kind: 'documentClassifierBuild',
//   status: 'succeeded',
//   .
//   .
//   result: {
//     classifierId: 'customClassifier10978',
//     createdDateTime: '2023-11-09T12:45:56Z',
//     .
//     .
//     description: 'Custom classifier description'
//   },
//   apiVersion: '2023-10-31-preview'
// }

```

## Get Info

ts

```

const response = await client.path("/info").get();
if (isUnexpected(response)) {
  throw response.body.error;
}
console.log(response.body.customDocumentModels.limit);
// 20000

```

# List Document Models

ts

```
import { paginate } from "@azure-rest/ai-document-intelligence";
const response = await client.path("/documentModels").get();
if (isUnexpected(response)) {
    throw response.body.error;
}

const modelsInAccount: string[] = [];
for await (const model of paginate(client, response)) {
    console.log(model.modelId);
}
```

## Troubleshooting

### Logging

Enabling logging may help uncover useful information about failures. In order to see a log of HTTP requests and responses, set the `AZURE_LOG_LEVEL` environment variable to `info`.

Alternatively, logging can be enabled at runtime by calling `setLogLevel` in the `@azure/logger`:

JavaScript

```
const { setLogLevel } = require("@azure/logger");

setLogLevel("info");
```

For more detailed instructions on how to enable logs, you can look at the [@azure/logger package docs](#).

# Azure AI Document Intelligence client library for Python - version 1.0.0b4

Article • 09/06/2024

Azure AI Document Intelligence ([previously known as Form Recognizer](#)) is a cloud service that uses machine learning to analyze text and structured data from your documents. It includes the following main features:

- Layout - Extract content and structure (ex. words, selection marks, tables) from documents.
- Document - Analyze key-value pairs in addition to general layout from documents.
- Read - Read page information from documents.
- Prebuilt - Extract common field values from select document types (ex. receipts, invoices, business cards, ID documents, U.S. W-2 tax documents, among others) using prebuilt models.
- Custom - Build custom models from your own data to extract tailored field values in addition to general layout from documents.
- Classifiers - Build custom classification models that combine layout and language features to accurately detect and identify documents you process within your application.
- Add-on capabilities - Extract barcodes/QR codes, formulas, font/style, etc. or enable high resolution mode for large documents with optional parameters.

[Source code](#) | [Package \(PyPI\)](#) | [API reference documentation](#) | [Product documentation](#) | [Samples](#)

## Disclaimer

The latest service API is currently only available in some Azure regions, the available regions can be found from [here](#).

## Getting started

### Installing the package

Bash

```
python -m pip install azure-ai-documentintelligence
```

This table shows the relationship between SDK versions and supported API service versions:

[+] Expand table

SDK version	Supported API service version
1.0.0b1	2023-10-31-preview
1.0.0b2	2024-02-29-preview

Older API versions are supported in `azure-ai-formrecognizer`, please see the [Migration Guide](#) for detailed instructions on how to update application.

## Prerequisites

- Python 3.8 or later is required to use this package.
- You need an [Azure subscription](#) to use this package.
- An existing Azure AI Document Intelligence instance.

## Create a Cognitive Services or Document Intelligence resource

Document Intelligence supports both [multi-service and single-service access](#). Create a Cognitive Services resource if you plan to access multiple cognitive services under a single endpoint/key. For Document Intelligence access only, create a Document Intelligence resource. Please note that you will need a single-service resource if you intend to use [Azure Active Directory authentication](#).

You can create either resource using:

- Option 1: [Azure Portal](#).
- Option 2: [Azure CLI](#).

Below is an example of how you can create a Document Intelligence resource using the CLI:

PowerShell

```
# Create a new resource group to hold the Document Intelligence resource
# if using an existing resource group, skip this step
az group create --name <your-resource-name> --location <location>
```

PowerShell

```
# Create the Document Intelligence resource
az cognitiveservices account create \
    --name <your-resource-name> \
    --resource-group <your-resource-group-name> \
    --kind FormRecognizer \
    --sku <sku> \
    --location <location> \
    --yes
```

For more information about creating the resource or how to get the location and sku information see [here](#).

## Authenticate the client

In order to interact with the Document Intelligence service, you will need to create an instance of a client. An **endpoint** and **credential** are necessary to instantiate the client object.

### Get the endpoint

You can find the endpoint for your Document Intelligence resource using the [Azure Portal](#) or [Azure CLI](#):

Bash

```
# Get the endpoint for the Document Intelligence resource
az cognitiveservices account show --name "resource-name" --resource-group
"resource-group-name" --query "properties.endpoint"
```

Either a regional endpoint or a custom subdomain can be used for authentication. They are formatted as follows:

```
Regional endpoint: https://<region>.api.cognitive.microsoft.com/
Custom subdomain: https://<resource-name>.cognitiveservices.azure.com/
```

A regional endpoint is the same for every resource in a region. A complete list of supported regional endpoints can be consulted [here](#). Please note that regional endpoints do not support AAD authentication.

A custom subdomain, on the other hand, is a name that is unique to the Document Intelligence resource. They can only be used by [single-service resources](#).

## Get the API key

The API key can be found in the [Azure Portal](#) or by running the following Azure CLI command:

Bash

```
az cognitiveservices account keys list --name "<resource-name>" --resource-group "<resource-group-name>"
```

## Create the client with AzureKeyCredential

To use an [API key](#) as the `credential` parameter, pass the key as a string into an instance of [AzureKeyCredential](#).

Python

```
from azure.core.credentials import AzureKeyCredential
from azure.ai.documentintelligence import DocumentIntelligenceClient

endpoint = "https://<my-custom-subdomain>.cognitiveservices.azure.com/"
credential = AzureKeyCredential("<api_key>")
document_intelligence_client = DocumentIntelligenceClient(endpoint,
    credential)
```

## Create the client with an Azure Active Directory credential

`AzureKeyCredential` authentication is used in the examples in this getting started guide, but you can also authenticate with Azure Active Directory using the [azure-identity](#) library. Note that regional endpoints do not support AAD authentication. Create a [custom subdomain](#) name for your resource in order to use this type of authentication.

To use the [DefaultAzureCredential](#) type shown below, or other credential types provided with the Azure SDK, please install the `azure-identity` package:

```
pip install azure-identity
```

You will also need to [register a new AAD application](#) and grant access to Document Intelligence by assigning the `"Cognitive Services User"` role to your service principal.

Once completed, set the values of the client ID, tenant ID, and client secret of the AAD application as environment variables: `AZURE_CLIENT_ID`, `AZURE_TENANT_ID`, `AZURE_CLIENT_SECRET`.

Python

```
"""DefaultAzureCredential will use the values from these environment
variables: AZURE_CLIENT_ID, AZURE_TENANT_ID, AZURE_CLIENT_SECRET
"""

from azure.ai.documentintelligence import DocumentIntelligenceClient
from azure.identity import DefaultAzureCredential

endpoint = os.environ["DOCUMENTINTELLIGENCE_ENDPOINT"]
credential = DefaultAzureCredential()

document_intelligence_client = DocumentIntelligenceClient(endpoint,
credential)
```

## Key concepts

### DocumentIntelligenceClient

`DocumentIntelligenceClient` provides operations for analyzing input documents using prebuilt and custom models through the `begin_analyze_document` API. Use the `model_id` parameter to select the type of model for analysis. See a full list of supported models [here](#). The `DocumentIntelligenceClient` also provides operations for classifying documents through the `begin_classify_document` API. Custom classification models can classify each page in an input file to identify the document(s) within and can also identify multiple documents or multiple instances of a single document within an input file.

Sample code snippets are provided to illustrate using a `DocumentIntelligenceClient` [here](#). More information about analyzing documents, including supported features, locales, and document types can be found in the [service documentation](#).

### DocumentIntelligenceAdministrationClient

`DocumentIntelligenceAdministrationClient` provides operations for:

- Building custom models to analyze specific fields you specify by labeling your custom documents. A `DocumentModelDetails` is returned indicating the document type(s) the model can analyze, as well as the estimated confidence for each field. See the [service documentation](#) for a more detailed explanation.
- Creating a composed model from a collection of existing models.
- Managing models created in your account.

- Listing operations or getting a specific model operation created within the last 24 hours.
- Copying a custom model from one Document Intelligence resource to another.
- Build and manage a custom classification model to classify the documents you process within your application.

Please note that models can also be built using a graphical user interface such as [Document Intelligence Studio](#).

Sample code snippets are provided to illustrate using a [DocumentIntelligenceAdministrationClient](#) here.

## Long-running operations

Long-running operations are operations which consist of an initial request sent to the service to start an operation, followed by polling the service at intervals to determine whether the operation has completed or failed, and if it has succeeded, to get the result.

Methods that analyze documents, build models, or copy/compose models are modeled as long-running operations. The client exposes a `begin_<method-name>` method that returns an `LROPoller` or `AsyncLROPoller`. Callers should wait for the operation to complete by calling `result()` on the poller object returned from the `begin_<method-name>` method. Sample code snippets are provided to illustrate using long-running operations [below](#).

## Examples

The following section provides several code snippets covering some of the most common Document Intelligence tasks, including:

- [Extract Layout](#)
- [Extract Figures from Documents](#)
- [Analyze Documents Result in PDF](#)
- [Using the General Document Model](#)
- [Using Prebuilt Models](#)
- [Build a Custom Model](#)
- [Analyze Documents Using a Custom Model](#)
- [Manage Your Models](#)
- [Add-on capabilities](#)

## Extract Layout

Extract text, selection marks, text styles, and table structures, along with their bounding region coordinates, from documents.

Python

```
from azure.core.credentials import AzureKeyCredential
from azure.ai.documentintelligence import DocumentIntelligenceClient
from azure.ai.documentintelligence.models import AnalyzeResult

def _in_span(word, spans):
    for span in spans:
        if word.span.offset >= span.offset and (word.span.offset + word.span.length) <= (span.offset + span.length):
            return True
    return False

def _format_polygon(polygon):
    if not polygon:
        return "N/A"
    return ", ".join([f"[{polygon[i]}, {polygon[i + 1]}]" for i in range(0, len(polygon), 2)])

endpoint = os.environ["DOCUMENTINTELLIGENCE_ENDPOINT"]
key = os.environ["DOCUMENTINTELLIGENCE_API_KEY"]

document_intelligence_client = DocumentIntelligenceClient(endpoint=endpoint,
    credential=AzureKeyCredential(key))
with open(path_to_sample_documents, "rb") as f:
    poller = document_intelligence_client.begin_analyze_document(
        "prebuilt-layout", analyze_request=f,
        content_type="application/octet-stream")
result: AnalyzeResult = poller.result()

if result.styles and any([style.is_handwritten for style in result.styles]):
    print("Document contains handwritten content")
else:
    print("Document does not contain handwritten content")

for page in result.pages:
    print(f"----Analyzing layout from page #{page.page_number}----")
    print(f"Page has width: {page.width} and height: {page.height}, measured with unit: {page.unit}")

    if page.lines:
        for line_idx, line in enumerate(page.lines):
            words = []
            if page.words:
                for word in page.words:
                    print(f".....Word '{word.content}' has a confidence of {word.confidence}")
                    if _in_span(word, line.spans):
                        words.append(word)
            print(f"Line {line_idx} has {len(words)} words: {words}")
```

```

        f"...Line # {line_idx} has word count {len(words)} and text
'{line.content}'"
            f"within bounding polygon '{_format_polygon(line.polygon)}'"
        )

if page.selection_marks:
    for selection_mark in page.selection_marks:
        print(
            f"Selection mark is '{selection_mark.state}' within bounding
            polygon "
            f"'{_format_polygon(selection_mark.polygon)}' and has a
            confidence of {selection_mark.confidence}"
        )

if result.paragraphs:
    print(f"---Detected #{len(result.paragraphs)} paragraphs in the
document---")
    # Sort all paragraphs by span's offset to read in the right order.
    result.paragraphs.sort(key=lambda p: (p.spans.sort(key=lambda s:
s.offset), p.spans[0].offset))
    print("----Print sorted paragraphs----")
    for paragraph in result.paragraphs:
        if not paragraph.bounding_regions:
            print(f"Found paragraph with role: '{paragraph.role}' within N/A
bounding region")
        else:
            print(f"Found paragraph with role: '{paragraph.role}' within")
            print(
                ", ".join(
                    f" Page #{region.page_number}:
{_format_polygon(region.polygon)} bounding region"
                    for region in paragraph.bounding_regions
                )
            )
            print(f"...with content: '{paragraph.content}'")
            print(f"...with offset: {paragraph.spans[0].offset} and length:
{paragraph.spans[0].length}")

if result.tables:
    for table_idx, table in enumerate(result.tables):
        print(f"Table # {table_idx} has {table.row_count} rows and {table.column_count} columns")
        if table.bounding_regions:
            for region in table.bounding_regions:
                print(
                    f"Table # {table_idx} location on page:
{region.page_number} is {_format_polygon(region.polygon)}"
                )
                for cell in table.cells:
                    print(f"...Cell[{cell.row_index}][{cell.column_index}] has text
'{cell.content}'")
                    if cell.bounding_regions:
                        for region in cell.bounding_regions:
                            print(
                                f"....content on page {region.page_number} is within
")

```

```
        bounding_polygon ' {_format_polygon(region.polygon)}' "
    )
print("-----")
```

## Extract Figures from Documents

Extract figures from the document as cropped images.

Python

```
from azure.core.credentials import AzureKeyCredential
from azure.ai.documentintelligence import DocumentIntelligenceClient
from azure.ai.documentintelligence.models import AnalyzeOutputOption,
AnalyzeResult

endpoint = os.environ["DOCUMENTINTELLIGENCE_ENDPOINT"]
key = os.environ["DOCUMENTINTELLIGENCE_API_KEY"]

document_intelligence_client = DocumentIntelligenceClient(endpoint=endpoint,
credential=AzureKeyCredential(key))

with open(path_to_sample_documents, "rb") as f:
    poller = document_intelligence_client.begin_analyze_document(
        "prebuilt-layout",
        analyze_request=f,
        output=[AnalyzeOutputOption.FIGURES],
        content_type="application/octet-stream",
    )
    result: AnalyzeResult = poller.result()
    operation_id = poller.details["operation_id"]

    if result.figures:
        for figure in result.figures:
            if figure.id:
                response =
document_intelligence_client.get_analyze_result_figure(
                model_id=result.model_id, result_id=operation_id,
                figure_id=figure.id
            )
            with open(f"{figure.id}.png", "wb") as writer:
                writer.writelines(response)
    else:
        print("No figures found.")
```

## Analyze Documents Result in PDF

Convert an analog PDF into a PDF with embedded text. Such text can enable text search within the PDF or allow the PDF to be used in LLM chat scenarios.

*Note: For now, this feature is only supported by `prebuilt-read`. All other models will return error.*

Python

```
from azure.core.credentials import AzureKeyCredential
from azure.ai.documentintelligence import DocumentIntelligenceClient
from azure.ai.documentintelligence.models import AnalyzeOutputOption,
AnalyzeResult

endpoint = os.environ["DOCUMENTINTELLIGENCE_ENDPOINT"]
key = os.environ["DOCUMENTINTELLIGENCE_API_KEY"]

document_intelligence_client = DocumentIntelligenceClient(endpoint=endpoint,
credential=AzureKeyCredential(key))

with open(path_to_sample_documents, "rb") as f:
    poller = document_intelligence_client.begin_analyze_document(
        "prebuilt-read",
        analyze_request=f,
        output=[AnalyzeOutputOption.PDF],
        content_type="application/octet-stream",
    )
    result: AnalyzeResult = poller.result()
    operation_id = poller.details["operation_id"]

    response =
    document_intelligence_client.get_analyze_result_pdf(model_id=result.model_id
    , result_id=operation_id)
    with open("analyze_result.pdf", "wb") as writer:
        writer.writelines(response)
```

## Using the General Document Model

Analyze key-value pairs, tables, styles, and selection marks from documents using the general document model provided by the Document Intelligence service. Select the General Document Model by passing `model_id="prebuilt-document"` into the `begin_analyze_document` method:

Python

```
from azure.core.credentials import AzureKeyCredential
from azure.ai.documentintelligence import DocumentIntelligenceClient
from azure.ai.documentintelligence.models import DocumentAnalysisFeature,
AnalyzeResult

def _in_span(word, spans):
    for span in spans:
        if word.span.offset >= span.offset and (word.span.offset +
```

```
word.span.length) <= (span.offset + span.length):
    return True
return False

def _format_bounding_region(bounding_regions):
    if not bounding_regions:
        return "N/A"
    return ", ".join(
        f"Page #{region.page_number}: {_format_polygon(region.polygon)}" for
region in bounding_regions
    )

def _format_polygon(polygon):
    if not polygon:
        return "N/A"
    return ", ".join([f"[{polygon[i]}, {polygon[i + 1]}]" for i in range(0,
len(polygon), 2)])

endpoint = os.environ["DOCUMENTINTELLIGENCE_ENDPOINT"]
key = os.environ["DOCUMENTINTELLIGENCE_API_KEY"]

document_intelligence_client = DocumentIntelligenceClient(endpoint=endpoint,
credential=AzureKeyCredential(key))
with open(path_to_sample_documents, "rb") as f:
    poller = document_intelligence_client.begin_analyze_document(
        "prebuilt-layout",
        analyze_request=f,
        features=[DocumentAnalysisFeature.KEY_VALUE_PAIRS],
        content_type="application/octet-stream",
    )
result: AnalyzeResult = poller.result()

if result.styles:
    for style in result.styles:
        if style.is_handwritten:
            print("Document contains handwritten content: ")
            print(", ".join([result.content[span.offset : span.offset +
span.length] for span in style.spans]))

print("----Key-value pairs found in document----")
if result.key_value_pairs:
    for kv_pair in result.key_value_pairs:
        if kv_pair.key:
            print(
                f"Key '{kv_pair.key.content}' found within "
                f"'{_format_bounding_region(kv_pair.key.bounding_regions)}' "
                "bounding regions"
            )
        if kv_pair.value:
            print(
                f"Value '{kv_pair.value.content}' found within "
                f"'{_format_bounding_region(kv_pair.value.bounding_regions)}' bounding
regions\n"
            )
```

```

for page in result.pages:
    print(f"---Analyzing document from page #{page.page_number}---")
    print(f"Page has width: {page.width} and height: {page.height}, measured
with unit: {page.unit}")

    if page.lines:
        for line_idx, line in enumerate(page.lines):
            words = []
            if page.words:
                for word in page.words:
                    print(f".....Word '{word.content}' has a confidence of
{word.confidence}")
                    if _in_span(word, line.spans):
                        words.append(word)
            print(
                f"...Line #{line_idx} has {len(words)} words and text
'{line.content}' within "
                f"bounding polygon '{_format_polygon(line.polygon)}'"
            )

    if page.selection_marks:
        for selection_mark in page.selection_marks:
            print(
                f"Selection mark is '{selection_mark.state}' within bounding
polygon "
                f"'{_format_polygon(selection_mark.polygon)}' and has a
confidence of "
                f"{selection_mark.confidence}"
            )

if result.tables:
    for table_idx, table in enumerate(result.tables):
        print(f"Table # {table_idx} has {table.row_count} rows and
{table.column_count} columns")
        if table.bounding_regions:
            for region in table.bounding_regions:
                print(
                    f"Table # {table_idx} location on page:
{region.page_number} is {_format_polygon(region.polygon)}"
                )
            for cell in table.cells:
                print(f"....Cell[{cell.row_index}][{cell.column_index}] has text
'{cell.content}'")
                if cell.bounding_regions:
                    for region in cell.bounding_regions:
                        print(
                            f"....content on page {region.page_number} is within
bounding polygon '{_format_polygon(region.polygon)}'\n"
                        )
print("-----")

```

- Read more about the features provided by the `prebuilt-document` model [here](#).

# Using Prebuilt Models

Extract fields from select document types such as receipts, invoices, business cards, identity documents, and U.S. W-2 tax documents using prebuilt models provided by the Document Intelligence service.

For example, to analyze fields from a sales receipt, use the prebuilt receipt model provided by passing `model_id="prebuilt-receipt"` into the `begin_analyze_document` method:

Python

```
from azure.core.credentials import AzureKeyCredential
from azure.ai.documentintelligence import DocumentIntelligenceClient
from azure.ai.documentintelligence.models import AnalyzeResult

def _format_price(price_dict):
    return "".join([f"{p}" for p in price_dict.values()])

endpoint = os.environ["DOCUMENTINTELLIGENCE_ENDPOINT"]
key = os.environ["DOCUMENTINTELLIGENCE_API_KEY"]

document_intelligence_client = DocumentIntelligenceClient(endpoint=endpoint,
    credential=AzureKeyCredential(key))
with open(path_to_sample_documents, "rb") as f:
    poller = document_intelligence_client.begin_analyze_document(
        "prebuilt-receipt", analyze_request=f, locale="en-US",
        content_type="application/octet-stream"
    )
receipts: AnalyzeResult = poller.result()

if receipts.documents:
    for idx, receipt in enumerate(receipts.documents):
        print(f"-----Analysis of receipt #{idx + 1}-----")
        print(f"Receipt type: {receipt.doc_type if receipt.doc_type else 'N/A'}")
        if receipt.fields:
            merchant_name = receipt.fields.get("MerchantName")
            if merchant_name:
                print(
                    f"Merchant Name: {merchant_name.get('valueString')} has
confidence: "
                    f"{merchant_name.confidence}"
                )
            transaction_date = receipt.fields.get("TransactionDate")
            if transaction_date:
                print(
                    f"Transaction Date: {transaction_date.get('valueDate')} has
confidence: "
                    f"{transaction_date.confidence}"
                )
            items = receipt.fields.get("Items")
```

```

if items:
    print("Receipt items:")
    for idx, item in enumerate(items.get("valueArray")):
        print(f"...Item #{idx + 1}")
        item_description =
item.get("valueObject").get("Description")
        if item_description:
            print(
                f".....Item Description:
{item_description.get('valueString')} has confidence: "
                f"{item_description.confidence}"
            )
        item_quantity = item.get("valueObject").get("Quantity")
        if item_quantity:
            print(
                f".....Item Quantity:
{item_quantity.get('valueString')} has confidence: "
                f"{item_quantity.confidence}"
            )
        item_total_price =
item.get("valueObject").get("TotalPrice")
        if item_total_price:
            print(
                f".....Total Item Price:
{_format_price(item_total_price.get('valueCurrency'))} has confidence: "
                f"{item_total_price.confidence}"
            )
        subtotal = receipt.fields.get("Subtotal")
        if subtotal:
            print(
                f"Subtotal:
{_format_price(subtotal.get('valueCurrency'))} has confidence:
{subtotal.confidence}"
            )
        tax = receipt.fields.get("TotalTax")
        if tax:
            print(f"Total tax: {_format_price(tax.get('valueCurrency'))}
has confidence: {tax.confidence}")
        tip = receipt.fields.get("Tip")
        if tip:
            print(f"Tip: {_format_price(tip.get('valueCurrency'))} has
confidence: {tip.confidence}")
        total = receipt.fields.get("Total")
        if total:
            print(f"Total: {_format_price(total.get('valueCurrency'))}
has confidence: {total.confidence}")
        print("-----")

```

You are not limited to receipts! There are a few prebuilt models to choose from, each of which has its own set of supported fields. See other supported prebuilt models [here](#).

## Build a Custom Model

Build a custom model on your own document type. The resulting model can be used to analyze values from the types of documents it was trained on. Provide a container SAS URL to your Azure Storage Blob container where you're storing the training documents.

More details on setting up a container and required file structure can be found in the [service documentation](#).

Python

```
# Let's build a model to use for this sample
import uuid
from azure.ai.documentintelligence import
DocumentIntelligenceAdministrationClient
from azure.ai.documentintelligence.models import (
    DocumentBuildMode,
    BuildDocumentModelRequest,
    AzureBlobContentSource,
    DocumentModelDetails,
)
from azure.core.credentials import AzureKeyCredential

endpoint = os.environ["DOCUMENTINTELLIGENCE_ENDPOINT"]
key = os.environ["DOCUMENTINTELLIGENCE_API_KEY"]
container_sas_url =
os.environ["DOCUMENTINTELLIGENCE_STORAGE_CONTAINER_SAS_URL"]

document_intelligence_admin_client =
DocumentIntelligenceAdministrationClient(endpoint, AzureKeyCredential(key))
poller = document_intelligence_admin_client.begin_build_document_model(
    BuildDocumentModelRequest(
        model_id=str(uuid.uuid4()),
        build_mode=DocumentBuildMode.TEMPLATE,

    azure_blob_source=AzureBlobContentSource(container_url=container_sas_url),
        description="my model description",
    )
)
model: DocumentModelDetails = poller.result()

print(f"Model ID: {model.model_id}")
print(f"Description: {model.description}")
print(f"Model created on: {model.created_date_time}")
print(f"Model expires on: {model.expiration_date_time}")
if model.doc_types:
    print("Doc types the model can recognize:")
    for name, doc_type in model.doc_types.items():
        print(f"Doc Type: '{name}' built with '{doc_type.build_mode}' mode
which has the following fields:")
        if doc_type.field_schema:
            for field_name, field in doc_type.field_schema.items():
                if doc_type.field_confidence:
                    print(
                        f"Field: '{field_name}' has type '{field['type']}'"
                    )
```

```
        and confidence score "
                f"{doc_type.field_confidence[field_name]}"
            )
```

## Analyze Documents Using a Custom Model

Analyze document fields, tables, selection marks, and more. These models are trained with your own data, so they're tailored to your documents. For best results, you should only analyze documents of the same document type that the custom model was built with.

Python

```
from azure.core.credentials import AzureKeyCredential
from azure.ai.documentintelligence import DocumentIntelligenceClient
from azure.ai.documentintelligence.models import AnalyzeResult

def _print_table(header_names, table_data):
    # Print a two-dimensional array like a table.
    max_len_list = []
    for i in range(len(header_names)):
        col_values = list(map(lambda row: len(str(row[i])), table_data))
        col_values.append(len(str(header_names[i])))
        max_len_list.append(max(col_values))

    row_format_str = "".join(map(lambda len: f"{{:{len + 4}}}", max_len_list))

    print(row_format_str.format(*header_names))
    for row in table_data:
        print(row_format_str.format(*row))

endpoint = os.environ["DOCUMENTINTELLIGENCE_ENDPOINT"]
key = os.environ["DOCUMENTINTELLIGENCE_API_KEY"]
model_id = os.getenv("CUSTOM_BUILT_MODEL_ID", custom_model_id)

document_intelligence_client = DocumentIntelligenceClient(endpoint=endpoint,
credential=AzureKeyCredential(key))

# Make sure your document's type is included in the list of document types
# the custom model can analyze
with open(path_to_sample_documents, "rb") as f:
    poller = document_intelligence_client.begin_analyze_document(
        model_id=model_id, analyze_request=f,
        content_type="application/octet-stream"
    )
result: AnalyzeResult = poller.result()

if result.documents:
    for idx, document in enumerate(result.documents):
        print(f"-----Analyzing document #{idx + 1}-----")
```

```

        print(f"Document has type {document.doc_type}")
        print(f"Document has document type confidence
{document.confidence}")
        print(f"Document was analyzed with model with ID {result.model_id}")
        if document.fields:
            for name, field in document.fields.items():
                field_value = field.get("valueString") if
field.get("valueString") else field.content
                print(
                    f".....found field of type '{field.type}' with value
'{field_value}' and with confidence {field.confidence}"
                )

# Extract table cell values
SYMBOL_OF_TABLE_TYPE = "array"
SYMBOL_OF_OBJECT_TYPE = "object"
KEY_OF_VALUE_OBJECT = "valueObject"
KEY_OF_CELL_CONTENT = "content"

for doc in result.documents:
    if not doc.fields is None:
        for field_name, field_value in doc.fields.items():
            # Dynamic Table cell information store as array in document
field.
            if field_value.type == SYMBOL_OF_TABLE_TYPE and
field_value.value_array:
                col_names = []
                sample_obj = field_value.value_array[0]
                if KEY_OF_VALUE_OBJECT in sample_obj:
                    col_names =
list(sample_obj[KEY_OF_VALUE_OBJECT].keys())
                    print("----Extracting Dynamic Table Cell Values----")
                    table_rows = []
                    for obj in field_value.value_array:
                        if KEY_OF_VALUE_OBJECT in obj:
                            value_obj = obj[KEY_OF_VALUE_OBJECT]
                            extract_value_by_col_name = lambda key: (
                                value_obj[key].get(KEY_OF_CELL_CONTENT)
                                if key in value_obj and KEY_OF_CELL_CONTENT
in value_obj[key]
                                else "None"
                            )
                            row_data = list(map(extract_value_by_col_name,
col_names))
                            table_rows.append(row_data)
                            _print_table(col_names, table_rows)

            elif (
                field_value.type == SYMBOL_OF_OBJECT_TYPE
                and KEY_OF_VALUE_OBJECT in field_value
                and field_value[KEY_OF_VALUE_OBJECT] is not None
            ):
                rows_by_columns =
list(field_value[KEY_OF_VALUE_OBJECT].values())
                is_fixed_table = all(

```

```

(
    rows_of_column["type"] == SYMBOL_OF_OBJECT_TYPE
    and Counter(list(rows_by_columns[0]
[KEY_OF_VALUE_OBJECT].keys()))
    ==
Counter(list(rows_of_column[KEY_OF_VALUE_OBJECT].keys())))
)
for rows_of_column in rows_by_columns
)

# Fixed Table cell information store as object in
document field.
if is_fixed_table:
    print("----Extracting Fixed Table Cell Values----")
    col_names =
list(field_value[KEY_OF_VALUE_OBJECT].keys())
    row_dict: dict = {}
    for rows_of_column in rows_by_columns:
        rows = rows_of_column[KEY_OF_VALUE_OBJECT]
        for row_key in list(rows.keys()):
            if row_key in row_dict:
                row_dict[row_key].append(rows[row_key].get(KEY_OF_CELL_CONTENT))
            else:
                row_dict[row_key] = [
                    row_key,
                    rows[row_key].get(KEY_OF_CELL_CONTENT),
                ]
                col_names.insert(0, "")
                _print_table(col_names, list(row_dict.values()))

print("-----")

```

Additionally, a document URL can also be used to analyze documents using the `begin_analyze_document` method.

Python

```

from azure.core.credentials import AzureKeyCredential
from azure.ai.documentintelligence import DocumentIntelligenceClient
from azure.ai.documentintelligence.models import AnalyzeDocumentRequest,
AnalyzeResult

endpoint = os.environ["DOCUMENTINTELLIGENCE_ENDPOINT"]
key = os.environ["DOCUMENTINTELLIGENCE_API_KEY"]

document_intelligence_client = DocumentIntelligenceClient(endpoint=endpoint,
credential=AzureKeyCredential(key))
url = "https://raw.githubusercontent.com/Azure/azure-sdk-for-
python/main/sdk/documentintelligence/azure-ai-
documentintelligence/samples/sample_forms/receipt/contoso-receipt.png"

```

```
    poller = document_intelligence_client.begin_analyze_document(
        "prebuilt-receipt", AnalyzeDocumentRequest(url_source=url)
    )
receipts: AnalyzeResult = poller.result()
```

## Manage Your Models

Manage the custom models attached to your account.

Python

```
# Let's build a model to use for this sample
import uuid
from azure.ai.documentintelligence import
DocumentIntelligenceAdministrationClient
from azure.ai.documentintelligence.models import (
    DocumentBuildMode,
    BuildDocumentModelRequest,
    AzureBlobContentSource,
    DocumentModelDetails,
)
from azure.core.credentials import AzureKeyCredential

endpoint = os.environ["DOCUMENTINTELLIGENCE_ENDPOINT"]
key = os.environ["DOCUMENTINTELLIGENCE_API_KEY"]
container_sas_url =
os.environ["DOCUMENTINTELLIGENCE_STORAGE_CONTAINER_SAS_URL"]

document_intelligence_admin_client =
DocumentIntelligenceAdministrationClient(endpoint, AzureKeyCredential(key))
poller = document_intelligence_admin_client.begin_build_document_model(
    BuildDocumentModelRequest(
        model_id=str(uuid.uuid4()),
        build_mode=DocumentBuildMode.TEMPLATE,
        azure_blob_source=AzureBlobContentSource(container_url=container_sas_url),
        description="my model description",
    )
)
model: DocumentModelDetails = poller.result()

print(f"Model ID: {model.model_id}")
print(f"Description: {model.description}")
print(f"Model created on: {model.created_date_time}")
print(f"Model expires on: {model.expiration_date_time}")
if model.doc_types:
    print("Doc types the model can recognize:")
    for name, doc_type in model.doc_types.items():
        print(f"Doc Type: '{name}' built with '{doc_type.build_mode}' mode
which has the following fields:")
        if doc_type.field_schema:
            for field_name, field in doc_type.field_schema.items():
```

```
        if doc_type.field_confidence:
            print(
                f"Field: '{field_name}' has type '{field['type']}'"
                and confidence score "
                f"{doc_type.field_confidence[field_name]}"
            )
    )
```

Python

```
account_details = document_intelligence_admin_client.get_resource_info()
print(
    f"Our resource has {account_details.custom_document_models.count} custom
models, "
    f"and we can have at most {account_details.custom_document_models.limit}
custom models"
)
```

Python

```
# Next, we get a paged list of all of our custom models
models = document_intelligence_admin_client.list_models()

print("We have the following 'ready' models with IDs and descriptions:")
for model in models:
    print(f"{model.model_id} | {model.description}")
```

Python

```
my_model =
document_intelligence_admin_client.get_model(model_id=model.model_id)
print(f"\nModel ID: {my_model.model_id}")
print(f"Description: {my_model.description}")
print(f"Model created on: {my_model.created_date_time}")
print(f"Model expires on: {my_model.expiration_date_time}")
if my_model.warnings:
    print("Warnings encountered while building the model:")
    for warning in my_model.warnings:
        print(f"warning code: {warning.code}, message: {warning.message},
target of the error: {warning.target}")
```

Python

```
# Finally, we will delete this model by ID
document_intelligence_admin_client.delete_model(model_id=my_model.model_id)

from azure.core.exceptions import ResourceNotFoundError

try:
    document_intelligence_admin_client.get_model(model_id=my_model.model_id)
```

```
except ResourceNotFoundError:  
    print(f"Successfully deleted model with ID {my_model.model_id}")
```

## Add-on Capabilities

Document Intelligence supports more sophisticated analysis capabilities. These optional features can be enabled and disabled depending on the scenario of the document extraction.

The following add-on capabilities are available in this SDK:

- [barcode/QR code ↗](#)
- [formula ↗](#)
- [font/style ↗](#)
- [high resolution mode ↗](#)
- [language ↗](#)
- [query fields ↗](#)

Note that some add-on capabilities will incur additional charges. See pricing: <https://azure.microsoft.com/pricing/details/ai-document-intelligence/> ↗.

## Troubleshooting

### General

Document Intelligence client library will raise exceptions defined in [Azure Core](#) ↗. Error codes and messages raised by the Document Intelligence service can be found in the [service documentation](#) ↗.

### Logging

This library uses the standard [logging](#) ↗ library for logging.

Basic information about HTTP sessions (URLs, headers, etc.) is logged at `INFO` level.

Detailed `DEBUG` level logging, including request/response bodies and `unredacted` headers, can be enabled on the client or per-operation with the `logging_enable` keyword argument.

See full SDK logging documentation with examples [here](#).

# Optional Configuration

Optional keyword arguments can be passed in at the client and per-operation level. The azure-core [reference documentation](#) describes available configurations for retries, logging, transport protocols, and more.

## Next steps

### More sample code

See the [Sample README](#) for several code snippets illustrating common patterns used in the Document Intelligence Python API.

### Additional documentation

For more extensive documentation on Azure AI Document Intelligence, see the [Document Intelligence documentation](#) on docs.microsoft.com.

## Contributing

This project welcomes contributions and suggestions. Most contributions require you to agree to a Contributor License Agreement (CLA) declaring that you have the right to, and actually do, grant us the rights to use your contribution. For details, visit <https://cla.microsoft.com>.

When you submit a pull request, a CLA-bot will automatically determine whether you need to provide a CLA and decorate the PR appropriately (e.g., label, comment). Simply follow the instructions provided by the bot. You will only need to do this once across all repos using our CLA.

This project has adopted the [Microsoft Open Source Code of Conduct](#). For more information, see the Code of Conduct FAQ or contact [opencode@microsoft.com](mailto:opencode@microsoft.com) with any additional questions or comments.

# Azure AI services support and help options

Article • 05/02/2025

Here are the options for getting support, staying up to date, giving feedback, and reporting bugs for Azure AI services.

## Get solutions to common issues

In the Azure portal, you can find answers to common AI service issues.

1. Go to your Azure AI services resource in the Azure portal. You can find it on the list on this page: [Azure AI services](#). If you're a United States government customer, use the [Azure portal for the United States government](#).
2. In the left pane, under **Help**, select **Support + Troubleshooting**.
3. Describe your issue in the text box, and answer the remaining questions in the form.
4. You'll find Learn articles and other resources that might help you resolve your issue.

## Create an Azure support request



Explore the range of Azure support options and [choose the plan](#) that best fits, whether you're a developer just starting your cloud journey or a large organization deploying business-critical, strategic applications. Azure customers can create and manage support requests in the Azure portal.

To submit a support request for Azure AI services, follow the instructions on the [New support request](#) page in the Azure portal. After choosing your **Issue type**, select **Cognitive Services** in the **Service type** dropdown field.

## Post a question on Microsoft Q&A

For quick and reliable answers on your technical product questions from Microsoft Engineers, Azure Most Valuable Professionals (MVPs), or our expert community, engage with us on [Microsoft Q&A](#), Azure's preferred destination for community support.

If you can't find an answer to your problem using search, submit a new question to Microsoft Q&A. Use one of the following tags when you ask your question:

- [Azure AI services](#)

- [Azure OpenAI](#)

## Vision

- [Azure AI Vision](#)
- [Azure AI Custom Vision](#)
- [Azure Face](#)
- [Azure AI Document Intelligence](#)
- [Video Indexer](#)

## Language

- [Azure AI Immersive Reader](#)
- [Language Understanding \(LUIS\)](#)
- [Azure QnA Maker](#)
- [Azure AI Language](#)
- [Azure Translator](#)

## Speech

- [Azure AI Speech](#)

## Decision

- [Azure AI Anomaly Detector](#)
- [Content Moderator](#)
- [Azure AI Metrics Advisor](#)
- [Azure AI Personalizer](#)

# Post a question to Stack Overflow



For answers to your developer questions from the largest community developer ecosystem, ask your question on Stack Overflow.

If you submit a new question to Stack Overflow, use one or more of the following tags when you create the question:

- [Azure AI services ↗](#)

## Azure OpenAI

- [Azure OpenAI ↗](#)

## Vision

- [Azure AI Vision ↗](#)
- [Azure AI Custom Vision ↗](#)
- [Azure Face ↗](#)
- [Azure AI Document Intelligence ↗](#)
- [Video Indexer ↗](#)

## Language

- [Azure AI Immersive Reader ↗](#)
- [Language Understanding \(LUIS\) ↗](#)
- [Azure QnA Maker ↗](#)
- [Azure AI Language service ↗](#)
- [Azure Translator ↗](#)

## Speech

- [Azure AI Speech service ↗](#)

## Decision

- [Azure AI Anomaly Detector ↗](#)
- [Content Moderator ↗](#)
- [Azure AI Metrics Advisor ↗](#)
- [Azure AI Personalizer ↗](#)

# Submit feedback

To request new features, post them on <https://feedback.azure.com>. Share your ideas for making Azure AI services and its APIs work better for the applications you develop.

- [Azure AI services ↗](#)

## Vision

- [Azure AI Vision ↗](#)
- [Azure AI Custom Vision ↗](#)
- [Azure Face ↗](#)
- [Azure AI Document Intelligence ↗](#)
- [Video Indexer ↗](#)

## Language

- [Azure AI Immersive Reader ↗](#)
- [Language Understanding \(LUIS\) ↗](#)

- [Azure QnA Maker ↗](#)
- [Azure AI Language ↗](#)
- [Azure Translator ↗](#)

## Speech

- [Azure AI Speech service ↗](#)

## Decision

- [Azure AI Anomaly Detector ↗](#)
- [Content Moderator ↗](#)
- [Azure AI Metrics Advisor ↗](#)
- [Azure AI Personalizer ↗](#)

# Stay informed

You can learn about the features in a new release or get the latest news on the Azure blog. Staying informed can help you find the difference between a programming error, a service bug, or a feature not yet available in Azure AI services.

- Learn more about product updates, roadmap, and announcements in [Azure Updates ↗](#).
- News about Azure AI services is shared in the [Azure AI blog ↗](#).
- [Join the conversation on Reddit ↗](#) about Azure AI services.

# Next step

[What are Azure AI services?](#)

# Azure AI Document Intelligence release history

Article • 03/10/2025

Azure AI Document Intelligence is an innovative cloud-based service that utilizes machine learning to streamline data processing within applications and workflows. This service is essential for boosting data-driven strategies and improving document search functionalities. Here, discover key milestones and enhancements that have contributed to the evolution of Azure AI Document Intelligence. For more information on recent advances, see [What's new?](#).

## July 2023

### Note

Form Recognizer is now **Azure AI Document Intelligence!**

- No changes to pricing.
- The names *Cognitive Services* and *Azure Applied AI* continue to be used in Azure billing, cost analysis, price list, and price APIs.
- No breaking changes to application programming interfaces (APIs) or client libraries.
- Some platforms are still awaiting the renaming update. All mention of Form Recognizer or Document Intelligence in our documentation refers to the same Azure service.

### Document Intelligence v3.1 (GA)

The Document Intelligence version 3.1 API is now generally available (GA)! The API version corresponds to `2023-07-31`. The v3.1 API introduces new and updated capabilities:

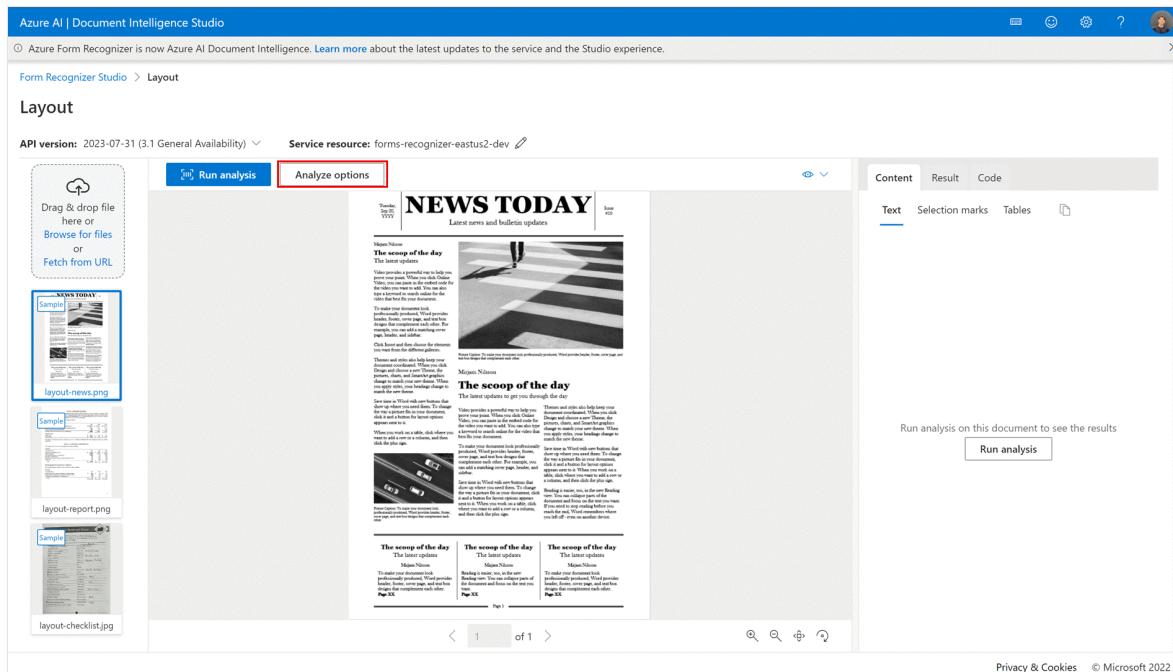
- Document Intelligence APIs are now more modular and with support for optional features. You can now customize the output to specifically include the features you need. Learn more about the [optional parameters](#).
- Document classification API for splitting a single file into individual documents. [Learn more](#) about document classification.
- [Prebuilt contract model](#).

- Prebuilt US tax form 1098 model.
- Support for [Office file types](#) with Read API.
- [Barcode recognition](#) in documents.
- Formula recognition [add-on capability](#).
- Font recognition [add-on capability](#).
- Support for [high resolution documents](#).
- Custom neural models now require a single labeled sample to train.
- Custom neural models language expansion. Train a neural model for documents in 30 languages. See [language support](#) for the complete list of supported languages.
- **NEW** [Prebuilt health insurance card model](#).
- [Prebuilt invoice model locale expansion](#).
- [Prebuilt receipt model language and locale expansion](#) with more than 100 languages supported.
- [Prebuilt ID model](#) now supports European IDs.

## Document Intelligence Studio UX Updates

### ✓ Analyze Options

- Document Intelligence now supports more sophisticated analysis capabilities and the Studio allows one entry point (Analyze options button) for configuring the add-on capabilities with ease.
- Depending on the document extraction scenario, configure the analysis range, document page range, optional detection, and premium detection features.

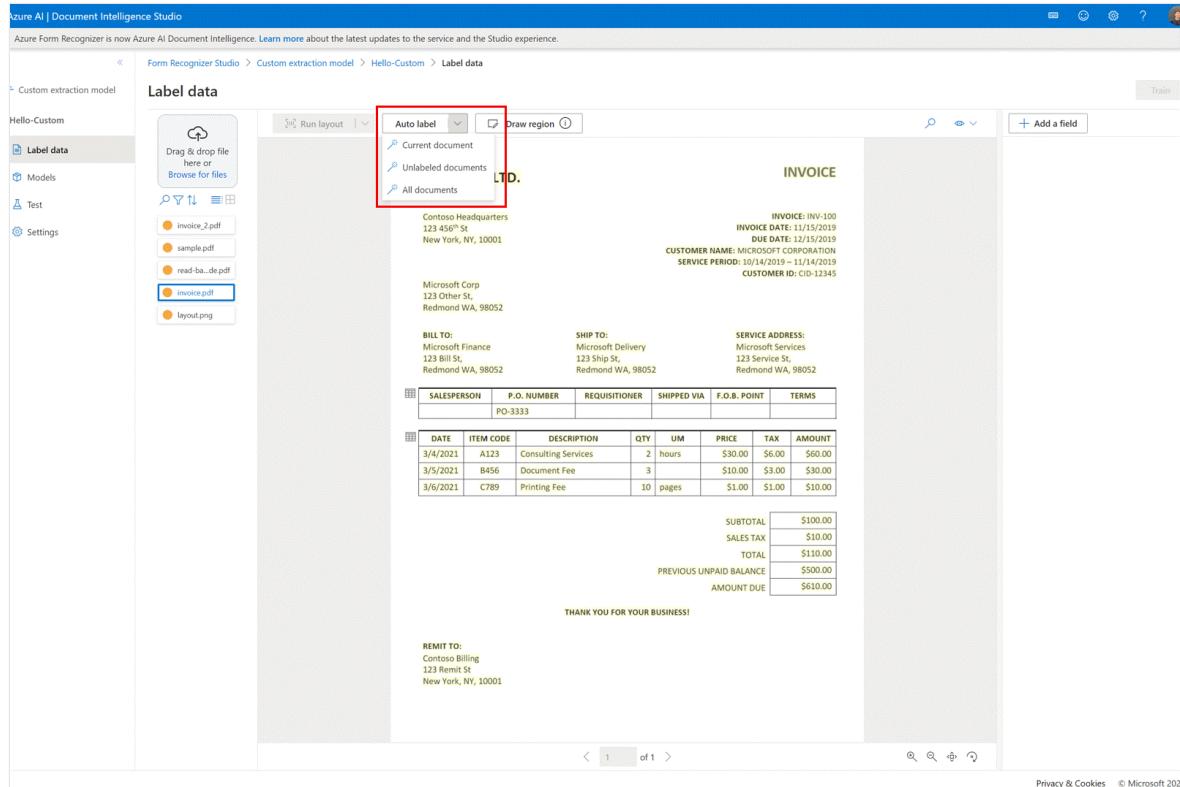


### ⓘ Note

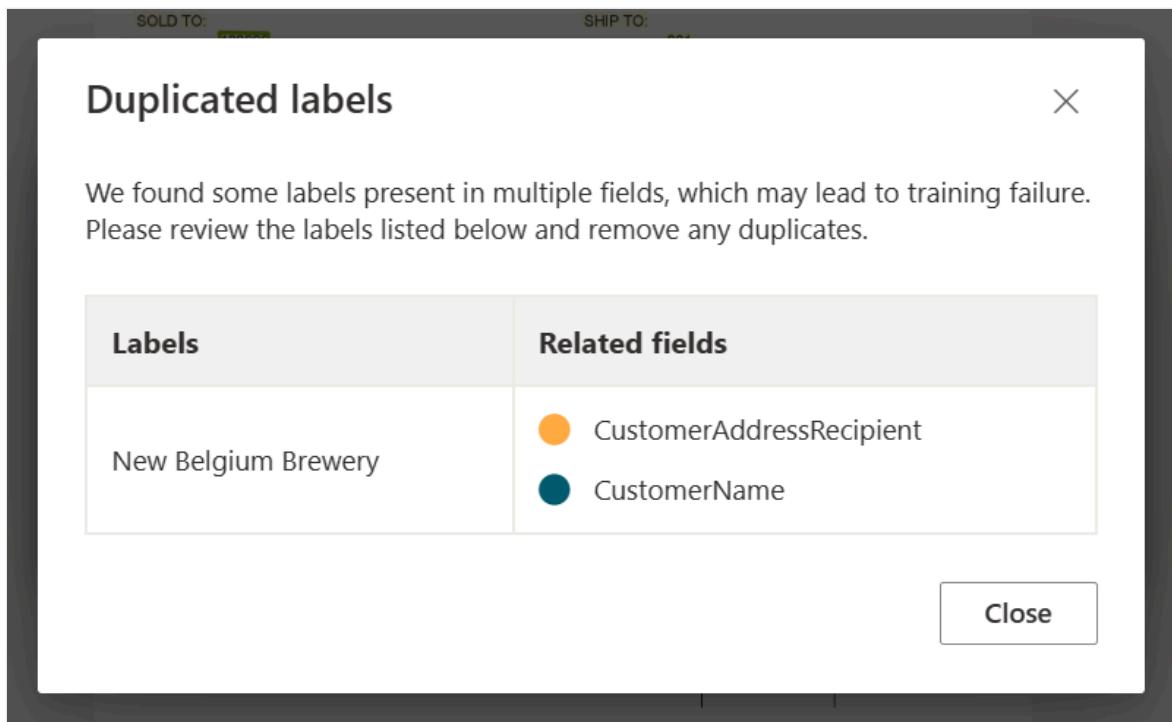
Font extraction isn't visualized in Document Intelligence Studio. However, you can check the styles section of the JSON output for the font detection results.

## ✓ Auto labeling documents with prebuilt models or one of your own models

- In custom extraction model labeling page, you can now auto label your documents using one of Document Intelligent Service prebuilt models or models you previously trained.



- For some documents, there can be duplicate labels after running auto label. Make sure to modify the labels so that there are no duplicate labels in the labeling page afterwards.



## ✓ Auto labeling tables

- In custom extraction model labeling page, you can now auto label the tables in the document without having to label the tables manually.

From	To	By	ETD	ETA
18 Queen Street Hoboken, NJ 07030	52 West Trenton St. Harleysville, PA 19438	cause science slow	09-Dec-2018 19:00	09-Dec-2020 11:00
9 Ketch Harbour Ave	75 Fawn Street Peabody, MA 01960	tone late spoken	12-Dec-2018 16:00	19-Dec-2020
Vincentown, NJ				

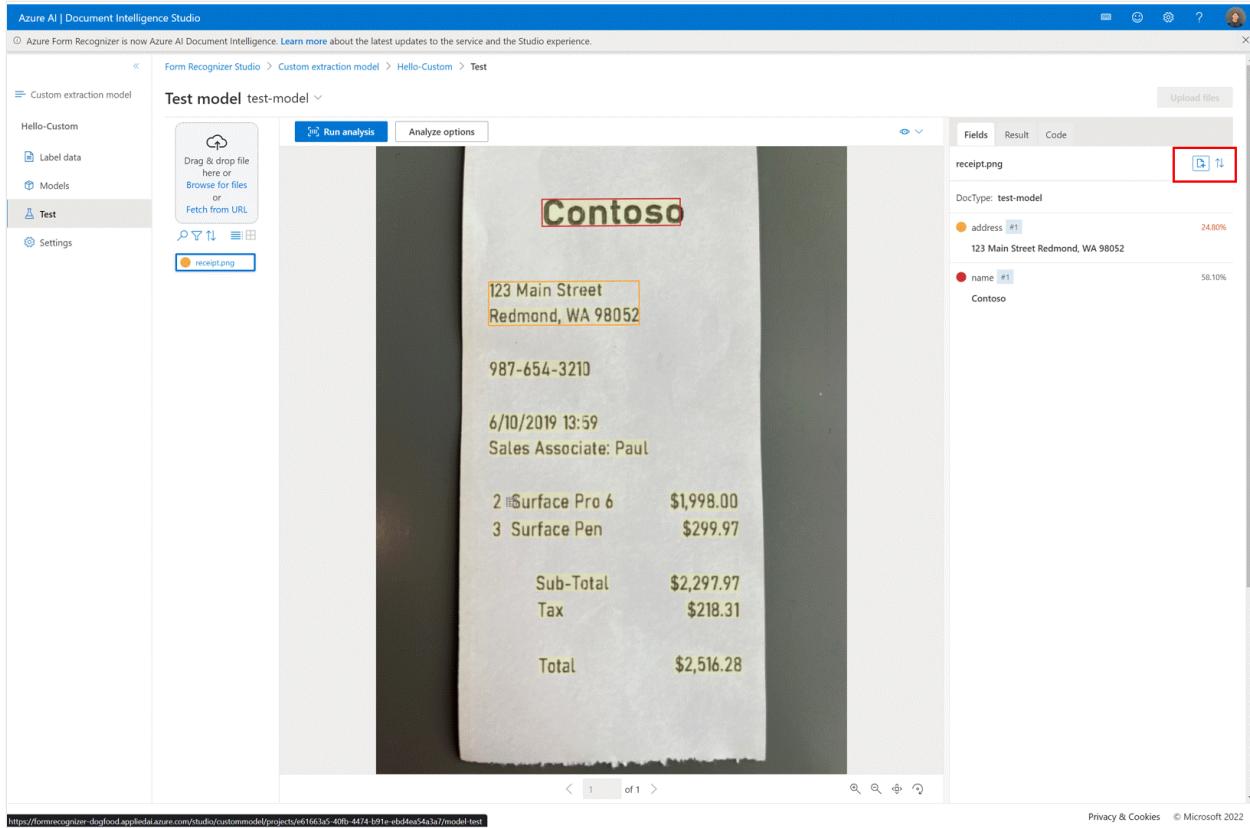
  

Deadline	Location	Date/Time (local)	Required action
Table	Harleysville (PA)	09-Dec-2019 13:00	Nobody likes being wearing lipstick.
Flight	Harleysville (PA)	09-Dec-2019 13:00	Two more days and all his problems would be solved.
Round	Harleysville (PA)	09-Dec-2019	
Accent	Harleysville (PA)	10-Dec-2019	
Monkey	Harleysville (PA)	11-Dec-2019	
Route	Harleysville (PA)	11-Dec-2019	Peanuts don't grow on trees.

## ✓ Add test files directly to your training dataset

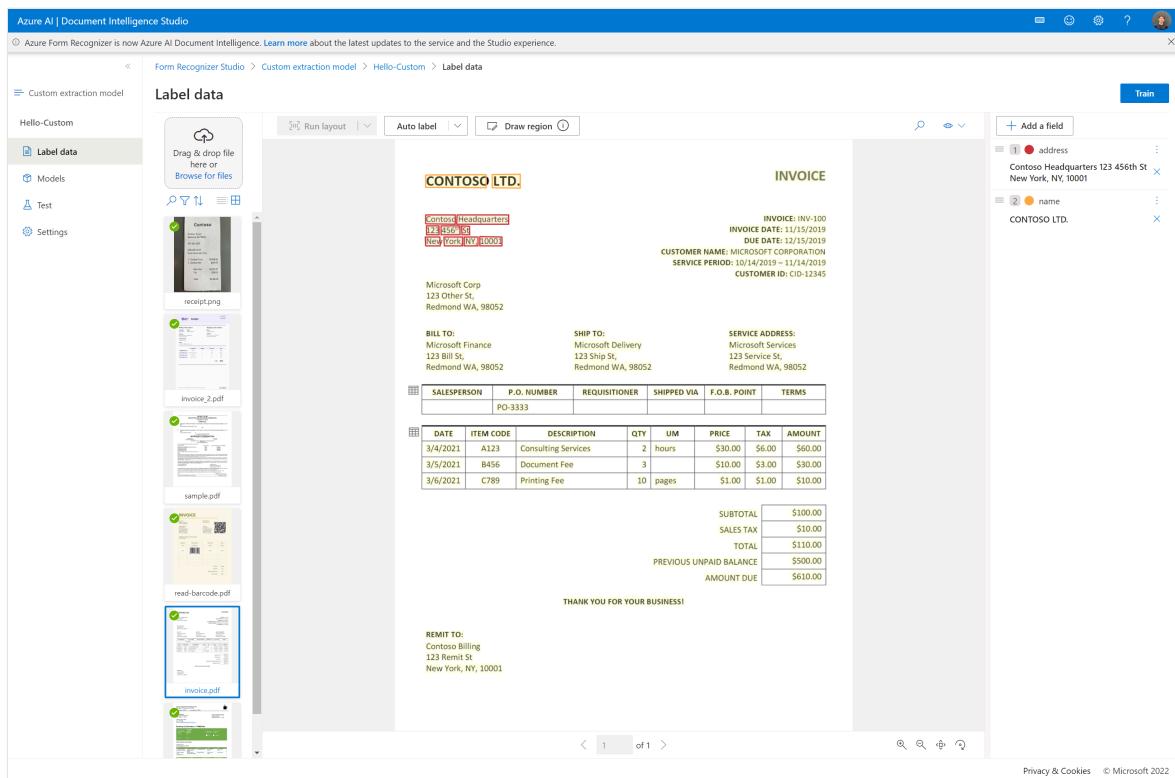
- Once you train a custom extraction model, make use of the test page to improve your model quality by uploading test documents to training dataset if needed.

- If a low confidence score is returned for some labels, make sure your labels are correct. If not, add them to the training dataset and relabel to improve the model quality.



## ✓ Make use of the document list options and filters in custom projects

- Use the custom extraction model labeling page. You can now navigate through your training documents with ease by making use of the search, filter, and sort by feature.
- Utilize the grid view to preview documents or use the list view to scroll through the documents more easily.



## ✓ Project sharing

- Share custom extraction projects with ease. For more information, see [Project sharing with custom models](#).

## May 2023

### Introducing refreshed documentation for Build 2023

- **NEW** [Document Intelligence Overview](#) enhanced navigation, structured access points, and enriched images.
- **NEW** [Choose a Document Intelligence model](#) provides guidance for choosing the best Document Intelligence solution for your projects and workflows.

## April 2023

### Announcing the latest Document Intelligence client-library public preview release

- Document Intelligence REST API Version **2023-02-28-preview** supports the public preview release client libraries. This release includes the following new features and capabilities available for .NET/C# (4.1.0-beta-1), Java (4.1.0-beta-1), JavaScript (4.1.0-beta-1), and Python (3.3.0b1) client libraries:
  - [Custom classification model](#)

- [Query fields extraction](#)
- [Add-on capabilities](#)
- For more information, see [Document Intelligence SDK \(../public preview\)](#) and [March 2023 release](#) notes

## March 2023

### ⓘ Important

[2023-02-28-preview](#) capabilities are currently only available in the following regions:

- West Europe
- West US2
- East US

- **Custom classification model** is a new capability within Document Intelligence starting with the [2023-02-28-preview](#) API.
- **Query fields** capabilities added to the General Document model, use Azure OpenAI models to extract specific fields from documents. Try the **General documents with query fields** feature using the [Document Intelligence Studio ↗](#). Query fields are currently only active for resources in the [East US](#) region.
- **Add-on capabilities:**
  - **Font extraction** is now recognized with the [2023-02-28-preview](#) API.
  - **Formula extraction** is now recognized with the [2023-02-28-preview](#) API.
  - **High resolution extraction** is now recognized with the [2023-02-28-preview](#) API.
- **Custom extraction model updates:**
  - **Custom neural model** now supports added languages for training and analysis. Train neural models for Dutch, French, German, Italian, and Spanish.
  - **Custom template model** now has an improved signature detection capability.
- **Document Intelligence Studio ↗** updates:
  - In addition to support for all the new features like classification and query fields, the Studio now enables project sharing for custom model projects.
  - New model additions in gated preview: **Vaccination cards**, **Contracts**, **US Tax 1098**, **US Tax 1098-E**, and **US Tax 1098-T**. To request access to gated preview models, complete and submit the [Document Intelligence private preview request form ↗](#).
- **Receipt model updates:**

- Receipt model adds support for thermal receipts.
  - Receipt model now adds language support for 18 languages and three regional languages (English, French, Portuguese).
  - Receipt model now supports `TaxDetails` extraction.
  - **Layout model** now improves table recognition.
  - **Read model** now adds improvement for single-digit character recognition.
- 

## February 2023

- Select Document Intelligence containers for v3.0 are now available for use!
- Currently **Read v3.0** and **Layout v3.0** containers are available.

For more information, see [Install and run Document Intelligence containers](#).

---

## January 2023

- Prebuilt receipt model - added languages supported. The receipt model now supports these added languages and locales
  - Japanese - Japan (ja-JP)
  - French - Canada (fr-CA)
  - Dutch - Netherlands (nl-NL)
  - English - United Arab Emirates (en-AE)
  - Portuguese - Brazil (pt-BR)
- Prebuilt invoice model - added languages supported. The invoice model now supports these added languages and locales
  - English - United States (en-US), Australia (en-AU), Canada (en-CA), United Kingdom (en-UK), India (en-IN)
  - Spanish - Spain (es-ES)
  - French - France (fr-FR)
  - Italian - Italy (it-IT)
  - Portuguese - Portugal (pt-PT)
  - Dutch - Netherlands (nl-NL)
- Prebuilt invoice model - added fields recognized. The invoice model now recognizes these added fields
  - Currency code
  - Payment options

- Total discount
- Tax items (en-IN only)
- Prebuilt ID model - added document types supported. The ID model now supports these added document types
  - US Military ID

### 💡 Tip

All January 2023 updates are available with [REST API version 2022-08-31 \(GA\)](#).

- **Prebuilt receipt model**—additional language support:

The **prebuilt receipt model** adds support for the following languages:

- English - United Arab Emirates (en-AE)
- Dutch - Netherlands (nl-NL)
- French - Canada (fr-CA)
- German - (de-DE)
- Italian - (it-IT)
- Japanese - Japan (ja-JP)
- Portuguese - Brazil (pt-BR)

- **Prebuilt invoice model**—additional language support and field extractions

The **prebuilt invoice model** adds support for the following languages:

- English - Australia (en-AU), Canada (en-CA), United Kingdom (en-UK), India (en-IN)
- Portuguese - Brazil (pt-BR)

The **prebuilt invoice model** now adds support for the following field extractions:

- Currency code
- Payment options
- Total discount
- Tax items (en-IN only)

- **Prebuilt ID document model**—additional document types support

The **prebuilt ID document model** now adds support for the following document types:

- Driver's license expansion supporting India, Canada, United Kingdom, and Australia
- US military ID cards and documents
- India ID cards and documents (PAN and Aadhaar)

- Australia ID cards and documents (photo card, Key-pass ID)
  - Canada ID cards and documents (identification card, Maple card)
  - United Kingdom ID cards and documents (national/regional identity card)
- 

## December 2022

- **Document Intelligence Studio updates** ↗

The December Document Intelligence Studio release includes the latest updates to Document Intelligence Studio. There are significant improvements to user experience, primarily with custom model labeling support.

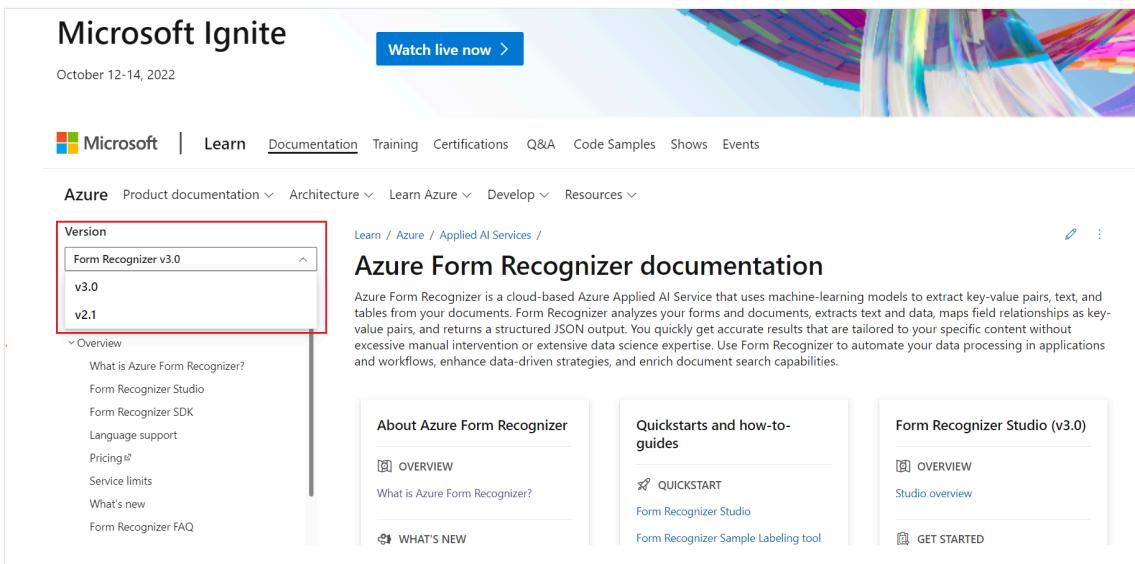
- **Page range.** The Studio now supports analyzing specified pages from a document.
  - **Custom model labeling:**
    - **Run Layout API automatically.** You can opt to run the Layout API for all documents automatically in your blob storage during the setup process for custom model.
    - **Search.** The Studio now includes search functionality to locate words within a document. This improvement allows for easier navigation while labeling.
    - **Navigation.** You can select labels to target labeled words within a document.
    - **Auto table labeling.** After you select the table icon within a document, you can opt to autolabel the extracted table in the labeling view.
    - **Label subtypes and second-level subtypes** The Studio now supports subtypes for table columns, table rows, and second-level subtypes for types such as dates and numbers.
  - Building custom neural models is now supported in the US Gov Virginia region.
  - Preview API versions `2022-01-30-preview` and `2021-09-30-preview` will be retired January 31 2023. Update to the `2022-08-31` API version to avoid any service disruptions.
- 

## November 2022

- Announcing the latest stable release of Azure AI Document Intelligence libraries
    - This release includes important changes and updates for .NET, Java, JavaScript, and Python client libraries. For more information, see [Azure SDK DevBlog](#).
    - The most significant enhancements are the introduction of two new clients, the `DocumentAnalysisClient` and the `DocumentModelAdministrationClient`.
- 

## October 2022

- Document Intelligence versioned content
  - Document Intelligence documentation is updated to present a versioned experience. Now, you can choose to view content targeting the `v3.0 GA` experience or the `v2.1 GA` experience. The v3.0 experience is the default.



The screenshot shows the Microsoft Ignite website for October 12-14, 2022. The navigation bar includes links for Microsoft, Learn, Documentation, Training, Certifications, Q&A, Code Samples, Shows, and Events. Below the navigation is a sub-navigation for Azure, with options for Product documentation, Architecture, Learn Azure, Develop, and Resources. A dropdown menu titled 'Version' is open, showing 'Form Recognizer v3.0' as the selected option, with 'v3.0' and 'v2.1' listed below it. The main content area is titled 'Azure Form Recognizer documentation'. It includes sections for 'About Azure Form Recognizer' (with 'OVERVIEW' and 'What is Azure Form Recognizer?'), 'Quickstarts and how-to-guides' (with 'QUICKSTART' and 'Form Recognizer Studio'), and 'Form Recognizer Studio (v3.0)' (with 'OVERVIEW', 'Studio overview', and 'GET STARTED'). The URL in the browser is 'Learn / Azure / Applied AI Services / Azure Form Recognizer documentation'.

- Document Intelligence Studio Sample Code
  - Sample code for the [Document Intelligence Studio labeling experience](#) is now available on GitHub. Customers can develop and integrate Document Intelligence into their own UX or build their own new UX using the Document Intelligence Studio sample code.
- Language expansion
  - With the latest preview release, Document Intelligence's Read (OCR), Layout, and Custom template models support 134 new languages. These language additions include Greek, Latvian, Serbian, Thai, Ukrainian, and Vietnamese, along with several Latin, and Cyrillic languages. Document Intelligence now has a total of 299 supported languages across the most recent GA and new preview versions. Refer to the supported languages pages to see all supported languages.

- Use the REST API parameter `api-version=2022-06-30-preview` when using the API or the corresponding SDK to support the new languages in your applications.
- **New Prebuilt Contract model**
  - A new prebuilt that extracts information from contracts such as parties, title, contract ID, execution date and more. the contracts model is currently in preview, request access [here ↗](#).
- **Region expansion for training custom neural models**
  - Training custom neural models now supported in added regions.
    - ✓ East US
    - ✓ East US2
    - ✓ US Gov Arizona

---

## September 2022

### ⓘ Note

Starting with version 4.0.0, a new set of clients is introduced to apply the newest features of the Document Intelligence service.

SDK version 4.0.0 GA release includes the following updates:

C#

- Version 4.0.0 GA (2022-09-08)
- Supports REST API v3.0 and v2.0 clients

[Package \(NuGet\) ↗](#)

[Changelog/Release History ↗](#)

[Migration guide ↗](#)

[ReadMe ↗](#)

[Samples ↗](#)

- Region expansion for training custom neural models now supported in six new regions

- ✓ Australia East
- ✓ Central US
- ✓ East Asia
- ✓ France Central
- ✓ UK South
- ✓ West US2

- For a complete list of regions where training is supported see [custom neural models](#).
- Document Intelligence SDK version 4.0.0 GA release:
  - **Document Intelligence client libraries version 4.0.0 (.NET/C#, Java, JavaScript) and version 3.2.0 (Python)** are generally available and ready for use in production applications!
  - For more information on Document Intelligence client libraries, see the [SDK overview](#).
  - Update your applications using your programming language's **migration guide**.

---

## August 2022

Document Intelligence SDK beta August 2022 preview release includes the following updates:

C#

[Version 4.0.0-beta.5 \(2022-08-09\)](#)

[Changelog/Release History ↗](#)

[Package \(NuGet\) ↗](#)

[SDK reference documentation](#)

- Document Intelligence v3.0 generally available
  - **Document Intelligence REST API v3.0 is now generally available and ready for use in production applications!** Update your applications with [REST API version 2022-08-31](#).
- Document Intelligence Studio updates

- ✓ **Next steps.** Under each model page, the Studio now has a next steps section. Users can quickly reference sample code, troubleshooting guidelines, and pricing information.
  - ✓ **Custom models.** The Studio now includes the ability to reorder labels in custom model projects to improve labeling efficiency.
  - ✓ **Copy Models** Custom models can be copied across Document Intelligence services from within the Studio. The operation enables the promotion of a trained model to other environments and regions.
  - ✓ **Delete documents.** The Studio now supports deleting documents from labeled dataset within custom projects.
- Document Intelligence service updates
    - **prebuilt-read.** Read OCR model is now also available in Document Intelligence with paragraphs and language detection as the two new features. Document Intelligence Read targets advanced document scenarios aligned with the broader document intelligence capabilities in Document Intelligence.
    - **prebuilt-layout.** The Layout model extracts paragraphs and whether the extracted text is a paragraph, title, section heading, footnote, page header, page footer, or page number.
    - **prebuilt-invoice.** The TotalVAT and Line/VAT fields now resolves to the existing fields TotalTax and Line/Tax respectively.
    - **prebuilt-idDocument.** Data extraction support for US state ID, social security, and green cards. Support for passport visa information.
    - **prebuilt-receipt.** Expanded locale support for French (fr-FR), Spanish (es-ES), Portuguese (pt-PT), Italian (it-IT) and German (de-DE).
    - **prebuilt-businessCard.** Address parse support to extract subfields for address components like address, city, state, country/region, and zip code.
  - AI quality improvements
    - **prebuilt-read.** Enhanced support for single characters, handwritten dates, amounts, names, other key data commonly found in receipts and invoices and improved processing of digital PDF documents.
    - **prebuilt-layout.** Support for better detection of cropped tables, borderless tables, and improved recognition of long spanning cells.
    - **prebuilt-document.** Improved value and check box detection.
    - **custom-neural.** Improved accuracy for table detection and extraction.

- Document Intelligence SDK beta June 2022 preview release includes the following updates:

C#

**Version 4.0.0-beta.4 (2022-06-08)**

[Changelog/Release History ↗](#)

[Package \(NuGet\) ↗](#)

[SDK reference documentation](#)

- [Document Intelligence Studio ↗](#) June release is the latest update to the Document Intelligence Studio. There are considerable user experience and accessibility improvements addressed in this update:
  - **Code sample for JavaScript and C#.** The Studio code tab now adds JavaScript and C# code samples in addition to the existing Python one.
  - **New document upload UI.** Studio now supports uploading a document with drag & drop into the new upload user interface.
  - **New feature for custom projects.** Custom projects now support creating storage account and blobs when configuring the project. In addition, custom project now supports uploading training files directly within the Studio and copying the existing custom model.
- Document Intelligence v3.0 [2022-06-30-preview](#) release presents extensive updates across the feature APIs:
  - **Layout extends structure extraction.** Layout now includes added structure elements including sections, section headers, and paragraphs. This update enables finer grain document segmentation scenarios. For a complete list of structure elements identified, see [enhanced structure](#).
  - **Custom neural model tabular fields support.** Custom document models now support tabular fields. Tabular fields by default are also multi page. To learn more about tabular fields in custom neural models, see [tabular fields](#).
  - **Custom template model tabular fields support for cross page tables.** Custom form models now support tabular fields across pages. To learn more about tabular fields in custom template models, see [tabular fields](#).
  - **Invoice model output now includes general document key-value pairs.** Where invoices contain required fields beyond the fields included in the prebuilt model, the general document model supplements the output with key-value pairs. See [key value pairs](#).

- **Invoice language expansion.** The invoice model includes expanded language support. See [supported languages](#).
- **Prebuilt business card** now includes Japanese language support. See [supported languages](#).
- **Prebuilt ID document model.** The ID document model now extracts DateOfIssue, Height, Weight, EyeColor, HairColor, and DocumentDiscriminator from US driver's licenses. See [field extraction](#).
- **Read model now supports common Microsoft Office document types.** Document types like Word (docx), Excel (xlsx), and PowerPoint (pptx) are now supported with the Read API. See [Read data extraction](#).

## February 2022

C#

Version 4.0.0-beta.3 (2022-02-10)

[Changelog/Release History](#)

[Package \(NuGet\)](#)

[SDK reference documentation](#)

- Document Intelligence v3.0 preview release introduces several new features, capabilities, and enhancements:
  - **Custom neural model** or custom document model is a new custom model to extract text and selection marks from structured forms, semi-structured and unstructured documents.
  - **W-2 prebuilt model** is a new prebuilt model to extract fields from W-2 forms for tax reporting and income verification scenarios.
  - **Read** API extracts printed text lines, words, text locations, detected languages, and handwritten text, if detected.
  - **General document** pretrained model is now updated to support selection marks in addition to API text, tables, structure, and key-value pairs from forms and documents.
  - **Invoice API** Invoice prebuilt model expands support to Spanish invoices.
  - **Document Intelligence Studio** adds new demos for Read, W2, Hotel receipt samples, and support for training the new custom neural models.
  - **Language Expansion** Document Intelligence Read, Layout, and Custom Form add support for 42 new languages including Arabic, Hindi, and other languages

using Arabic and Devanagari scripts to expand the coverage to 164 languages. Handwritten language support expands to Japanese and Korean.

- Get started with the new v3.0 preview API.
- Document Intelligence model data extraction:

[\[+\] Expand table](#)

Model	Text extraction	Key-Value pairs	Selection Marks	Tables	Signatures
Read	✓				
General document	✓	✓	✓	✓	
Layout	✓		✓	✓	
Invoice	✓	✓	✓	✓	
Receipt	✓	✓			✓
ID document	✓	✓			
Business card	✓	✓			
Custom template	✓	✓	✓	✓	✓
Custom neural	✓	✓	✓	✓	

- Document Intelligence SDK beta preview release includes the following updates:
  - [Custom Document models and modes](#):
    - [Custom template](#) (formerly custom form).
    - [Custom neural](#).
    - [Custom model—build mode](#).
  - [W-2 prebuilt model](#) ([./prebuilt-tax.us.w2](#)).
  - [Read prebuilt model](#) ([./prebuilt-read](#)).
  - [Invoice prebuilt model \(Spanish\)](#) ([./prebuilt-invoice](#)).

---

---

## Feedback

Was this page helpful?

 Yes

 No

Provide product feedback  | Get help at Microsoft Q&A