5.3 - Extract data from forms with Azure Document intelligence

---

Overview

Document intelligence uses machine learning technology to identify and extract key-value pairs and table data from form documents with accuracy, at scale. This module teaches you how to use the *Azure Document intelligence* cognitive service.

In this module, you learn how to:

- Identify how Document intelligence's layout service, prebuilt models, and custom models can automate processes.
- Use Document intelligence's capabilities with SDKs, REST API, and Document Intelligence Studio.
- Develop and test custom models.

Introduction

Forms are used to communicate information in every industry, every day. Many people still manually extract data from forms to exchange information.

Consider some of the instances when a person needs to process form data:

- When filing claims
- When enrolling new patients in an online management system
- When entering data from receipts to an expense report
- When reviewing an operations report for anomalies
- When selecting data from a report to give to a stakeholder

Without AI services, people need to manually sort through form documents to identify important information and then manually reenter data to record it. Some may also need to complete these tasks in real-time with a customer.

Azure Document Intelligence services provide the building blocks for automation by using intelligent services to extract data at scale and with accuracy. *Azure Document Intelligence* is a Vision API that extracts key-value pairs and table data from form documents.

Uses of the Azure Document Intelligence service include:

- Process automation
- Knowledge mining
- Industry-specific applications

In this module, you'll learn how to:

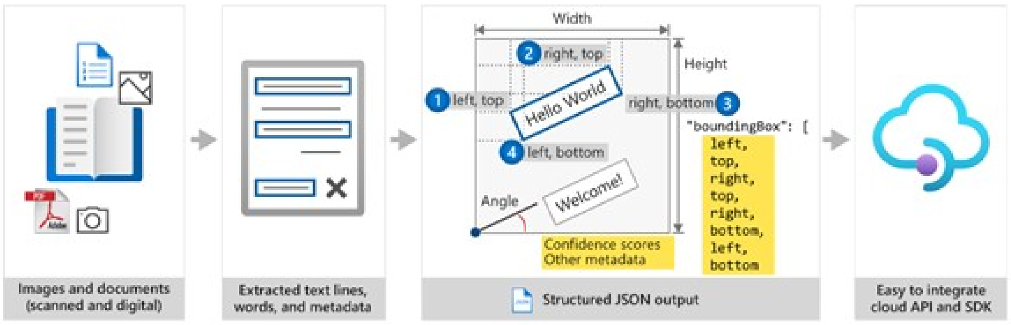
- Identify how Azure Document Intelligence's document analysis, prebuilt, and custom models can automate processes
- Use Azure Document Intelligence's Optical Character Recognition (OCR) capabilities with SDKs and REST API
- Develop and test a custom Azure Document Intelligence model

To complete this module, you'll need a Microsoft Azure subscription. If you don't already have one, you can sign up for a free trial at https://azure.microsoft.com.

What is Azure Document Intelligence?

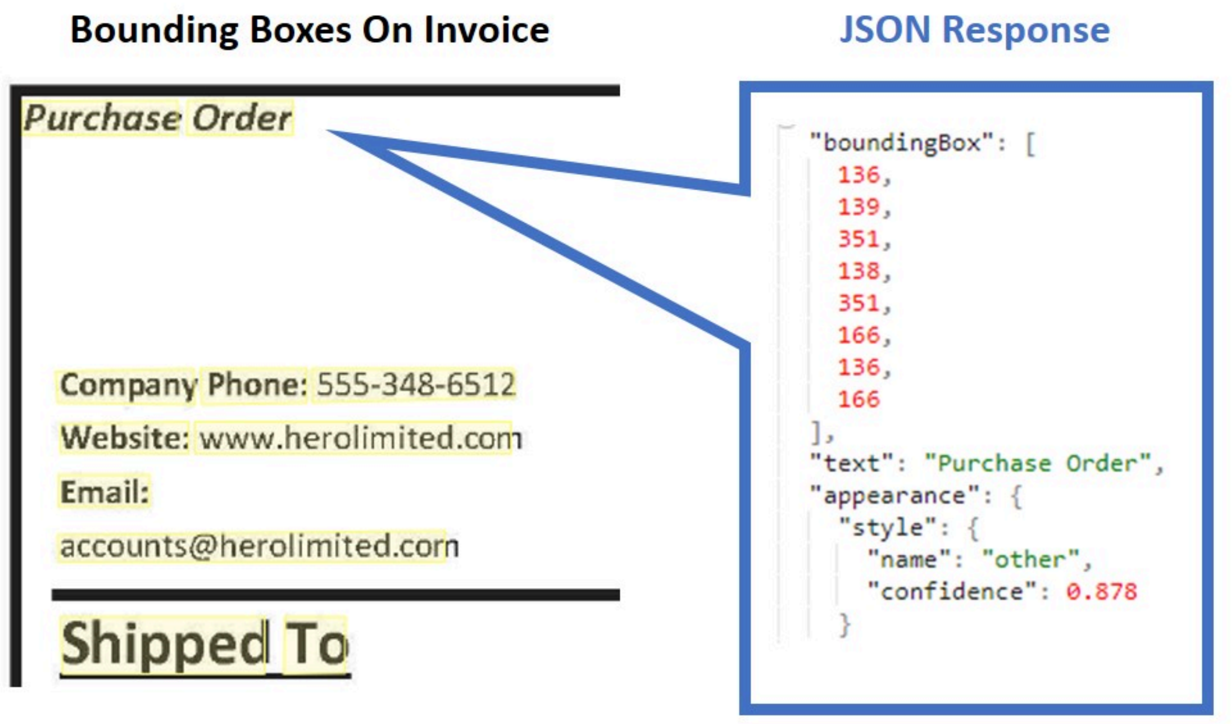
Azure Document Intelligence is one of many Azure AI Services, cloud-based artificial intelligence (AI) services with REST APIs and client library SDKs that can be used to build intelligence into your applications.

Azure Document Intelligence uses Optical Character Recognition (OCR) capabilities and deep learning models to **extract text, key-value pairs, selection marks, and tables from documents.**


Images and documents (scanned and digital)
Extracted text lines, words, and metadata
Width
Height
1 left, top
2 right, top
3 right, bottom
4 left, bottom
Hello World
Welcome!
Angle
Confidence scores
Other metadata
"boundingBox": [left, top, right, top, right, bottom, left, bottom
Structured JSON output
Easy to integrate cloud API and SDK

OCR captures document structure by creating bounding boxes around detected objects in an image. The locations of the bounding boxes are recorded as coordinates in relation to the rest of the page.

Azure Document Intelligence services return bounding box data and other information in a structured form with the relationships from the original file.



Purchase Order

Company Phone: 555-348-6512
Website: www.herolimited.com
Email:
accounts@herolimited.com

Shipped To

```
"boundingBox": [
  136,
  139,
  351,
  138,
  351,
  166,
  136,
  166
],
"text": "Purchase Order",
"appearance": {
  "style": {
    "name": "other",
    "confidence": 0.878
  }
```

To build a high-accuracy model from scratch, people need to build deep learning models, use a large amount of compute resources, and face long model training times. These factors could make a project infeasible.

Azure Document Intelligence provides **underlying models that have been trained on thousands of form examples. The underlying models enable you to do high-accuracy data extraction from your forms with little to no model training.**

Azure Document Intelligence service components

Azure Document Intelligence is composed of the following services:

- **Document analysis models**: which take an input of JPEG, PNG, PDF, and TIFF files and return a JSON file with the location of text in bounding boxes, text content, tables, selection marks (also known as checkboxes or radio buttons), and document structure.
- **Prebuilt models**: which detect and extract information from document images and return the extracted data in a structured JSON output. Azure Document Intelligence currently supports prebuilt models for several forms, including:
  - W-2 forms
  - Invoices
  - Receipts
  - ID documents
  - Business cards
- **Custom models**: custom models extract data from forms specific to your business. Custom models can be trained through the Azure Document Intelligence Studio.

Access services with the client library SDKs or REST API

You can access Azure Document Intelligence services by using a REST API, client library SDKs, and through the Azure Document Intelligence Studio to integrate the services into your workflow or application.

> Note: This module's exercise focuses on the Python and .NET SDKs. The underlying REST services can be used by any language.

Check out the documentation for quick start guides on all the available SDKs and the REST API.

Get started with Azure Document Intelligence

To start a project with Azure Document Intelligence services, you need to prepare the following:

- An Azure resource subscription
- A selection of form files for data extraction

Subscribe to a resource

You can access Azure Document Intelligence services via:

- An **Azure AI Service resource**: a multi-service subscription key (used across multiple Azure AI Services)
  OR

- An **Azure Document Intelligence resource**: a single-service subscription key (used only with a specific Azure AI Service)

> Note: **Create an Azure AI Services resource if you plan to access multiple Azure AI services under a single endpoint/key.** For Azure Document Intelligence access only, create an Azure Document Intelligence resource. Please note that you'll need a single-service resource if you intend to use Microsoft Entra authentication.

You can subscribe to a service in the Azure portal or with the Azure Command Line Interface (CLI). You can learn more about the CLI commands here.

Understand Azure Document Intelligence file input requirements

Azure Document Intelligence works on input documents that **meet these requirements:**

- Format must be JPG, PNG, BMP, PDF (text or scanned), or TIFF.
- 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 10000 x 10000 pixels.
- The total size of the training data set must be 500 pages or less.

More input requirements can be found in the documentation for specific models.

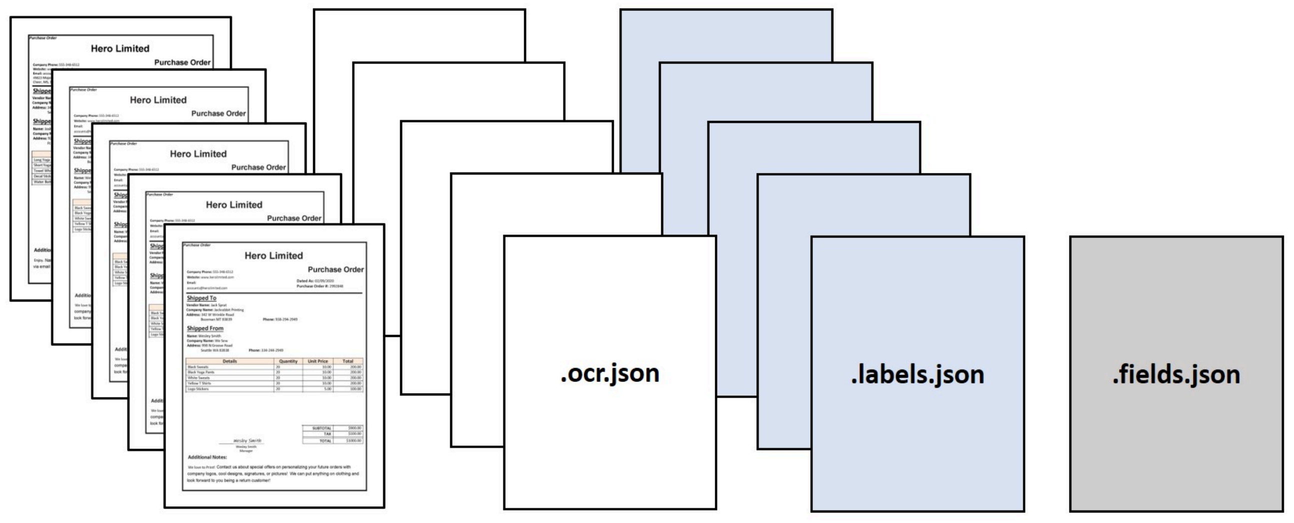
Decide what component of Azure Document Intelligence to use

After you have collected your files, decide what you need to accomplish.

- To use OCR capabilities to capture **document analysis**, use the Layout model, Read model, or General Document model.
- To create an application that extracts data from W-2s, Invoices, Receipts, ID documents, Health insurance, vaccination, and business cards, use a **prebuilt model.** These models do not need to be trained. Azure Document Intelligence services analyze the documents and return a JSON output.
- To create an application to extract data from your industry-specific forms, **create a custom model.** This model needs to be trained on sample documents. After training, the custom model can analyze new documents and return a JSON output.

Train custom models

Azure's Azure Document Intelligence service supports supervised machine learning. You can train custom models and create composite models with form documents *and* JSON documents that contain labeled fields.



Examples of files needed for supervised (labeled) custom form model training

To train a custom model:

1. **Store sample forms in an Azure blob container, along with JSON files containing layout and label field information.**
   - You can generate an **ocr.json** file for **each sample form** using the Azure Document Intelligence's **Analyze document** function.
   - Additionally, you need a single **fields.json** file describing the fields you want to extract, and a **labels.json** file for each sample form **mapping the fields to their location in that form.
   - Examples: Form_1.jpg.ocr.json , Form_1.jpg.labels.json, fields.json
2. Generate a shared access security (SAS) URL for the container.
3. Use the **Build model** REST API function (or equivalent SDK method).
4. Use the **Get model** REST API function (or equivalent SDK method) to get the trained **model ID**.

OR

5. Use the Azure Document Intelligence Studio to label and train. There are two types of underlying models for custom forms *custom template models* or *custom neural models*.
   - **Custom template models** accurately extract labeled key-value pairs, selection marks, tables, regions, and signatures from documents. **Training only takes a few minutes**, and more than 100 languages are supported.
   - **Custom neural models** are deep learned models that combine layout and language features to accurately extract labeled fields from documents. This model is best for **semi-structured or unstructured documents.**

Use Azure Document Intelligence models

Using the API

To extract form data using a custom model, use the **analyze document** function of either a supported SDK, or the REST API, while supplying model ID (generated during model training). This function starts the form analysis. which you can then request the result to get the analysis.

Example code to call your model:

C#

```
string endpoint = "<endpoint>";
string apiKey = "<apiKey>";
AzureKeyCredential credential = new AzureKeyCredential(apiKey);
DocumentAnalysisClient client = new DocumentAnalysisClient(new Uri(endpoint),
credential);

string modelId = "<modelId>";
Uri fileUri = new Uri("<fileUri>");

AnalyzeDocumentOperation operation = await
client.AnalyzeDocumentFromUriAsync(WaitUntil.Completed, modelId, fileUri);
AnalyzeResult result = operation.Value;
```

Python

```
endpoint = "YOUR_DOC_INTELLIGENCE_ENDPOINT"
key = "YOUR_DOC_INTELLIGENCE_KEY"

model_id = "YOUR_CUSTOM_BUILT_MODEL_ID"
formUrl = "YOUR_DOCUMENT"

document_analysis_client = DocumentAnalysisClient(
    endpoint=endpoint, credential=AzureKeyCredential(key)
)

# Make sure your document's type is included in the list of document types the custom
model can analyze
task = document_analysis_client.begin_analyze_document_from_url(model_id, formUrl)
result = task.result()
```

A successful JSON response contains **analyzeResult** that contains the content extracted and an array of pages containing information about the document content.

Example **analyze document** JSON response:

```
{
    "status": "succeeded",
    "createdDateTime": "2023-10-18T23:39:50Z",
    "lastUpdatedDateTime": "2023-10-18T23:39:54Z",
    "analyzeResult": {
        "apiVersion": "2022-08-31",
        "modelId": "DocIntelModel",
        "stringIndexType": "utf16CodeUnit",
        "content": "Purchase Order\nHero Limited\nCompany Phone: 555-348-6512 Website:
www.herolimited.com Email: accounts@herolimited.com\nPurchase Order\nDated As:
12/20/2020 Purchase Order #: 948284\nShipped To Vendor Name: Balozi Khamisi Company
Name: Higgly Wiggly Books Address: 938 NE Burner Road Boulder City, CO 92848 Phone: 938-
294-2949\nShipped From Name: Kidane Tsehaye Company Name: Jupiter Book Supply Address:
383 N Kinnick Road Seattle, WA 38383\nPhone: 932-299-0292\nDetails\nQuantity\nUnit
Price\nTotal\nBindings\n20\n1.00\n20.00\nCovers Small\n20\n1.00\n20.00\nFeather
```

```
Bookmark\n20\n5.00\n100.00\nCopper Swirl
Marker\n20\n5.00\n100.00\nSUBTOTAL\n$140.00\nTAX\n$4.00\nTOTAL\n$144.00\nKidane
Tsehaye\nManager\nKidane Tsehaye\nAdditional Notes: Do not Jostle Box. Unpack carefully.
Enjoy. Jupiter Book Supply will refund you 50% per book if returned within 60 days of
reading and offer you 25% off you next total purchase.",
        "pages": [
            {
                "pageNumber": 1,
                "angle": 0,
                "width": 1159,
                "height": 1486,
                "unit": "pixel",
                "words": [
                    {
                        "content": "Purchase",
                        "polygon": [
                            89,
                            90,
                            174,
                            91,
                            174,
                            112,
                            88,
                            112
                        ],
                        "confidence": 0.996,
                        "span": {
                            "offset": 0,
                            "length": 8
                        }
                    },
                    {
                        "content": "Order",
                        "polygon": [
                            178,
                            91,
                            237,
                            91,
                            236,
                            113,
                            178,
                            112
                        ],
                        "confidence": 0.997,
                        "span": {
                            "offset": 9,
                            "length": 5
                        }
                    },
                    ...
```

Understanding confidence scores

If the confidence values of the **analyzeResult** are low, try to improve the quality of your input documents.

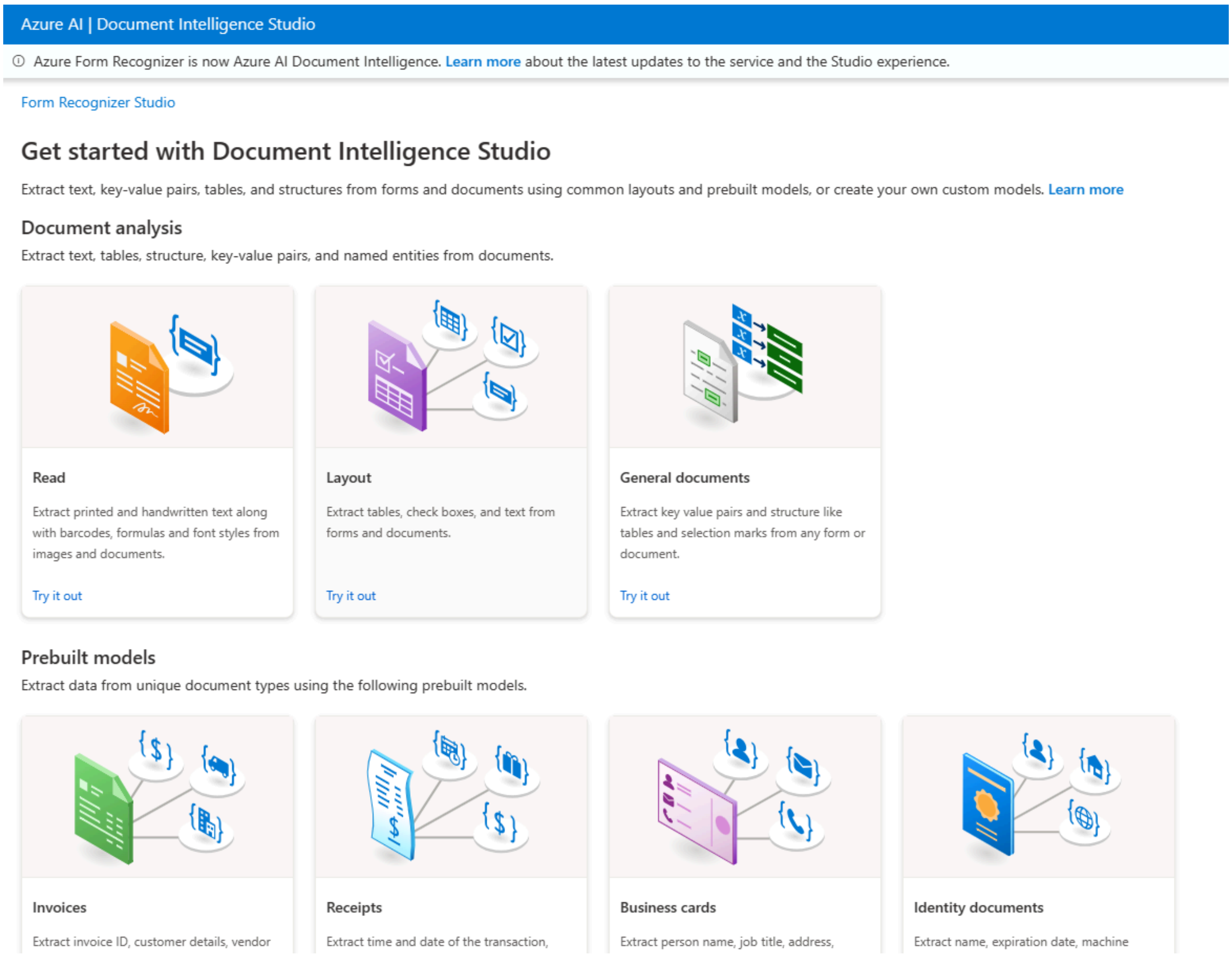
You want to make sure that the form you're analyzing has a similar appearance to forms in the training set if the confidence values are low.

If the form appearance varies, consider training more than one model, with each model focused on one form format.

Depending on the use case, **you might find that a confidence score of 80% or higher is acceptable for a low-risk application. For more sensitive cases, like reading medical records or billing statements, a score of 100% is recommended.**

Use the Azure Document Intelligence Studio

In addition to SDKs and the REST API, Azure Document Intelligence services can be accessed through a user interface called the Azure Document Intelligence Studio, an online tool for visually exploring, understanding, and integrating features from the Azure Document Intelligence service. The Studio can be used to analyze form layouts, extract data from prebuilt models, and train custom models.



Azure AI | Document Intelligence Studio

Azure Form Recognizer is now Azure AI Document Intelligence. Learn more about the latest updates to the service and the Studio experience.

Form Recognizer Studio

Get started with Document Intelligence Studio

Extract text, key-value pairs, tables, and structures from forms and documents using common layouts and prebuilt models, or create your own custom models. Learn more

Document analysis

Extract text, tables, structure, key-value pairs, and named entities from documents.

Read

Extract printed and handwritten text along with barcodes, formulas and font styles from images and documents.

Try it out

Layout

Extract tables, check boxes, and text from forms and documents.

Try it out

General documents

Extract key value pairs and structure like tables and selection marks from any form or document.

Try it out

Prebuilt models

Extract data from unique document types using the following prebuilt models.

Invoices

Extract invoice ID, customer details, vendor

Receipts

Extract time and date of the transaction,

Business cards

Extract person name, job title, address,

Identity documents

Extract name, expiration date, machine

The Azure Document Intelligence Studio currently supports the following projects:

- **Document analysis models**
  - Read: Extract printed and handwritten text lines, words, locations, and detected languages from documents and images.

    - Layout: Extract text, tables, selection marks, and structure information from documents (PDF and TIFF) and images (JPG, PNG, and BMP).
    - General Documents: Extract key-value pairs, selection marks, and entities from documents.
- **Prebuilt models**
- **Custom models**

Build Document analysis model projects

To extract text, tables, structure, key-value pairs, and named entities with document analysis models:

- Create an Azure Document Intelligence or Azure AI Services resource
- Select either **"Read", "Layout", or "General Documents"** under the **Document analysis models** category
- Analyze your document. You'll need your Azure Document Intelligence or Azure AI service endpoint and key.

Build prebuilt model projects

To extract data from common forms with prebuilt models:

- Create an Azure Document Intelligence or Azure AI Services resource
- Select one of the "prebuilt models" including W-2s, Invoices, Receipts, ID documents, Health insurance, vaccination, and business cards.
- Analyze your document. You'll need your Azure Document Intelligence or Azure AI service endpoint and key.

Build custom model projects

You can use Azure Document Intelligence Studio's custom service for the entire process of training and testing custom models.

When you use Azure Document Intelligence Studio to build custom models, the **ocr.json** files, **labels.json** files, and **fields.json** file needed for training are **automatically created and stored in your storage account.**

To train a custom model and use it to extract data with custom models:

- Create an Azure Document Intelligence or Azure AI Services resource
- Collect **at least 5-6 sample forms** for training and upload them to your storage account container.
- Configure **cross-domain resource sharing (CORS)**. CORS enables Azure Document Intelligence Studio to store labeled files in your storage container.
- Create a custom model project in Azure Document Intelligence Studio. You'll need to provide configurations linking your storage container and Azure Document Intelligence or Azure AI Service resource to the project.
- Use Azure Document Intelligence Studio to apply labels to text.
- Train your model. Once the model is trained, you'll receive a Model ID and Average Accuracy for tags.
- Test your model by analyzing a new form that wasn't used in training.

Exercise - Extract data from custom forms

Suppose a company currently requires employees to manually purchase order sheets and enter the data into a database. They would like you to utilize AI services to improve the data entry process. You decide to build a machine learning model that will read the form and produce structured data that can be used to automatically update a database.

Azure AI Document Intelligence is an Azure AI service that enables users to build automated data processing software. This software can extract text, key/value pairs, and tables from form documents using optical character recognition (OCR). Azure AI Document Intelligence has pre-built models for recognizing invoices, receipts, and business cards. The service also provides the capability to train custom models. In this exercise, we will focus on building custom models.

Prepare to develop an app in Visual Studio Code

Now let's use the service SDK to develop an app using Visual Studio Code. The code files for your app have been provided in a GitHub repo.

> **Tip**: If you have already cloned the **mslearn-ai-document-intelligence** repo, open it in Visual Studio code. Otherwise, follow these steps to clone it to your development environment.

1. Start Visual Studio Code.
2. Open the palette (SHIFT+CTRL+P) and run a **Git: Clone** command to clone the `https://github.com/MicrosoftLearning/mslearn-ai-document-intelligence` repository to a local folder (it doesn't matter which folder).
3. When the repository has been cloned, open the folder in Visual Studio Code.

   > **Note**: If Visual Studio Code shows you a pop-up message to prompt you to trust the code you are opening, click on **Yes, I trust the authors** option in the pop-up.

4. Wait while additional files are installed to support the C# code projects in the repo.

   > **Note**: If you are prompted to add required assets to build and debug, select **Not Now**. If you are prompted with the Message *Detected an Azure Function Project in folder*, you can safely close that message.

Create a Azure AI Document Intelligence resource

To use the Azure AI Document Intelligence service, you need a Azure AI Document Intelligence or Azure AI Services resource in your Azure subscription. You'll use the Azure portal to create a resource.

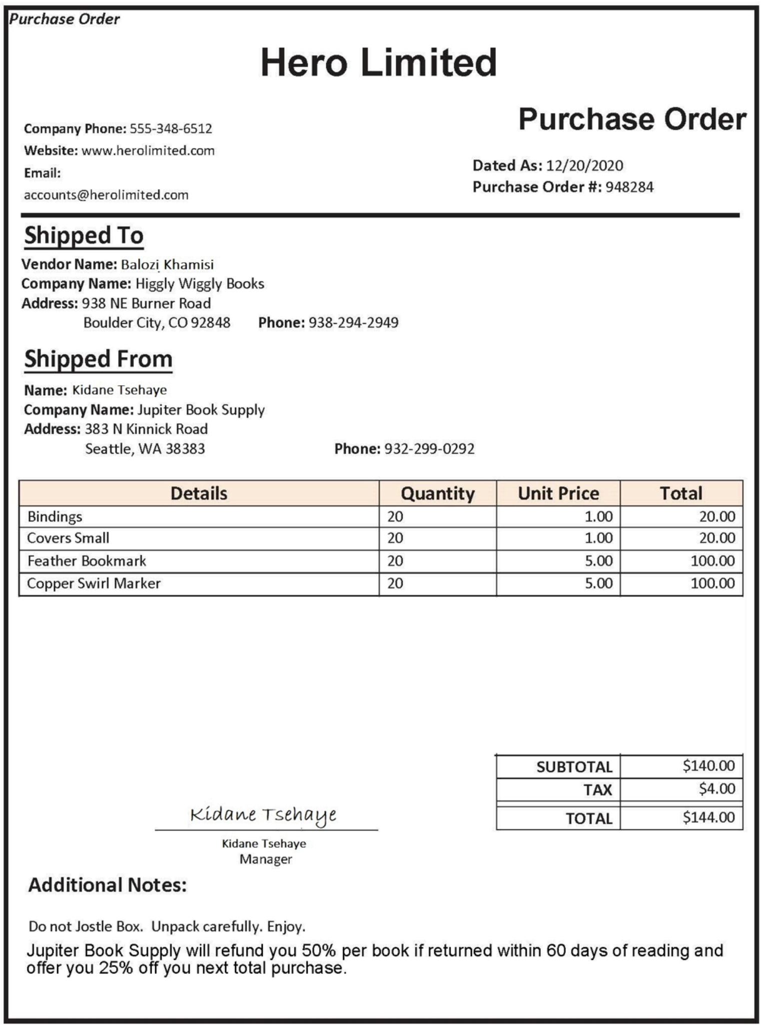
1. In a browser tab, open the Azure portal at `https://portal.azure.com`, signing in with the Microsoft account associated with your Azure subscription.
2. On the Azure portal home page, navigate to the top search box and type **Document Intelligence** and then press **Enter**.
3. On the **Document Intelligence** page, select **Create**.
4. On the **Create Document Intelligence** page, use the following to configure your resource:
   - **Subscription**: Your Azure subscription.
   - **Resource group**: Select or create a resource group with a unique name such as *DocIntelligenceResources*.
   - **Region**: Select a region near you.

- **Name**: Enter a globally unique name.
- **Pricing tier**: Select **Free F0** (if you don't have a Free tier available, select **Standard S0**).

5. Then select **Review + create**, and **Create**. Wait while Azure creates the Azure AI Document Intelligence resource.
6. When the deployment is complete, select **Go to resource** to view the resource's **Overview** page.

Gather documents for training

You'll use the sample forms such as this one to train a test a model:


Purchase Order

Hero Limited

Purchase Order

Company Phone: 555-348-6512
Website: www.herolimited.com
Email:
accounts@herolimited.com

Dated As: 12/20/2020
Purchase Order #: 948284

Shipped To

Vendor Name: Balozi Khamisi
Company Name: Higgly Wiggly Books
Address: 938 NE Burner Road
Boulder City, CO 92848 **Phone:** 938-294-2949

Shipped From

Name: Kidane Tsehaye
Company Name: Jupiter Book Supply
Address: 383 N Kinnick Road
Seattle, WA 38383 **Phone:** 932-299-0292

Details	Quantity	Unit Price	Total
Bindings	20	1.00	20.00
Covers Small	20	1.00	20.00
Feather Bookmark	20	5.00	100.00
Copper Swirl Marker	20	5.00	100.00

SUBTOTAL	$140.00
TAX	$4.00
TOTAL	$144.00

Kidane Tsehaye

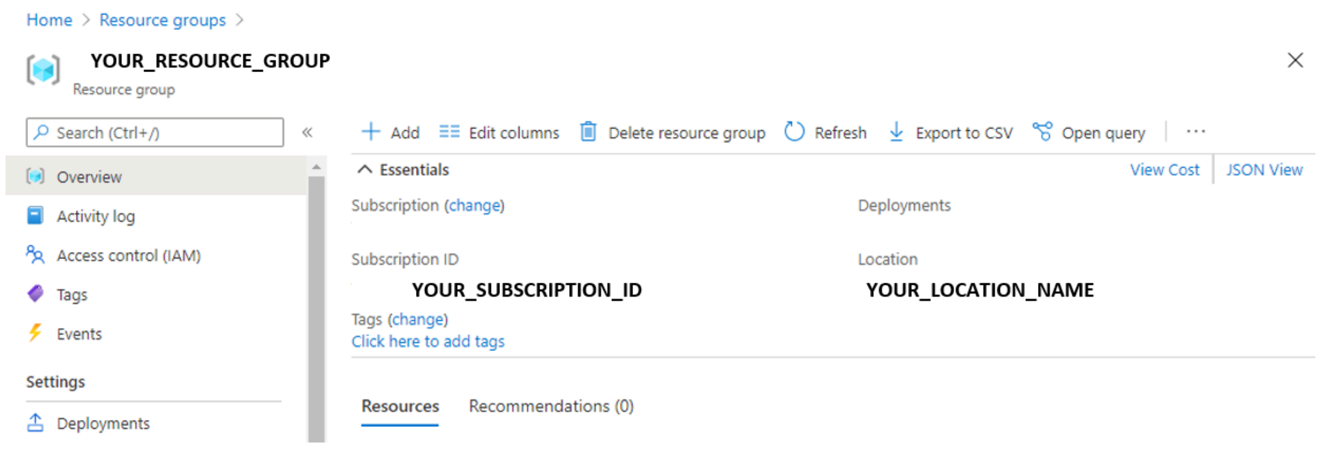
Kidane Tsehaye
Manager

Additional Notes:

Do not Jostle Box. Unpack carefully. Enjoy.
Jupiter Book Supply will refund you 50% per book if returned within 60 days of reading and offer you 25% off you next total purchase.

1. Return to **Visual Studio Code**. In the *Explorer* pane, open the **Labfiles/02-custom-document-intelligence** folder and expand the **sample-forms** folder. Notice there are files ending in **.json** and **.jpg** in the folder. You will use the **.jpg** files to train your model.

   The **.json** files have been generated for you and contain label information. The files will be uploaded into your blob storage container alongside the forms. You can view the images we are using in the *sample-forms* folder by selecting them on Visual Studio Code.
2. Return to the **Azure portal** and navigate to your resource's **Overview** page if you're not already there. Under the *Essentials* section, view the **Resource group** in which you created the Document Intelligence resource.
3. On the **Overview** page for your resource group, note the **Subscription ID** and **Location**. You will need these values, along with your **resource group** name in subsequent steps.



YOUR_RESOURCE_GROUP
Resource group

Subscription (change)

Deployments

Subscription ID
YOUR_SUBSCRIPTION_ID

Location
YOUR_LOCATION_NAME

Tags (change)
Click here to add tags

4. In Visual Studio Code, in the *Explorer* pane, browse to the **Labfiles/02-custom-document-intelligence** folder and expand the **CSharp** or **Python** folder depending on your language preference. Each folder contains the language-specific files for an app into which you're you're going to integrate Azure OpenAI functionality.
5. Right-click the **CSharp** or **Python** folder containing your code files and select **Open an integrated terminal**.
6. In the terminal, run the following command to login to Azure. The **az login** command will open the Microsoft Edge browser, login with the same account you used to create the Azure AI Document Intelligence resource. Once you are logged in, close the browser window.
   ```
   az login
   ```
7. Return to Visual Studio Code. In the terminal pane, run the following command to list the Azure locations.
   ```
   az account list-locations -o table
   ```
8. In the output, find the **Name** value that corresponds with the location of your resource group (for example, for *East US* the corresponding name is *eastus*).

   > **Important**: Record the **Name** value and use it in Step 11.

9. In Visual Studio Code, in the **Labfiles/02-custom-document-intelligence** folder, select **setup.cmd**. You will use this script to run the Azure command line interface (CLI) commands required to create the other Azure resources you need.
10. In the **setup.cmd** script, review the commands. The program will:
    - Create a storage account in your Azure resource group
    - Upload files from your local *sampleforms* folder to a container called *sampleforms* in the storage account (See *mslearn-ai-document-intelligence\Labfiles\02-custom-document-intelligence\sample-forms*)
    - Print a Shared Access Signature URI
11. Modify the **subscription_id**, **resource_group**, and **location** variable declarations with the appropriate values for the subscription, resource group, and location name where you deployed the Document Intelligence resource. Then **save** your changes.
    Leave the **expiry_date** variable as it is for the exercise. This variable is used when generating the Shared Access Signature (SAS) URI. In practice, you will want to set an appropriate expiry date for your SAS. You can learn more about SAS here.
12. In the terminal for the **Labfiles/02-custom-document-intelligence** folder, enter the following command to run the script:

```
$currentdir=(Get-Item .).FullName
cd ..
./setup.cmd
cd $currentdir
```

13. When the script completes, review the displayed output.
14. In the Azure portal, refresh your resource group and verify that it contains the Azure Storage account just created. Open the storage account and in the pane on the left, select **Storage browser**. Then in Storage Browser, expand **Blob containers** and select the **sampleforms** container to verify that the files have been uploaded from your local **02-custom-document-intelligence/sample-forms** folder.

Train the model using Document Intelligence Studio

Now you will train the model using the files uploaded to the storage account.

1. In your browser, navigate to the Document Intelligence Studio at `https://documentintelligence.ai.azure.com/studio`.
2. Scroll down to the **Custom models** section and select the **Custom extraction model** tile.
3. If you are asked to sign into your account, use your Azure credentials.
4. If you are asked which Azure AI Document Intelligence resource to use, select the subscription and resource name you used when you created the Azure AI Document Intelligence resource.
5. Under **My Projects**, select **Create a project**. Use the following configurations:
   - **Project name**: Enter a unique name.
     - Select *Continue*.
   - **Configure service resource**: Select the subscription, resource group, and document intelligence resource you created previously in this lab. Check the *Set as default* box. Keep the default API version.
     - Select *Continue*.
   - **Connect training data source**: Select the subscription, resource group, and storage account that was created by the setup script. Check the *Set as default* box. Select the `sampleforms` blob container, and leave the folder path blank.
     - Select *Continue*.
   - Select *Create project*
6. Once your project is created, on the top right of the screen, select **Train** to train your model. Use the following configurations:
   - **Model ID**: *Provide a globally unique name (you'll need the model ID name in the next step)*.
   - **Build Mode**: Template.
7. Select **Go to Models**.
8. Training can take some time. You'll see a notification when it's complete.

Test your custom Document Intelligence model

1. Return to Visual Studio Code. In the terminal, install the Document Intelligence package by running the appropriate command for your language preference:

   C#:
   `dotnet add package Azure.AI.FormRecognizer --version 4.1.0`

Python:

`pip install azure-ai-formrecognizer==3.3.3`

2. In Visual Studio Code, in the **Labfiles/02-custom-document-intelligence** folder, select the language you are using. Edit the configuration file (**appsettings.json** or **.env**, depending on your language preference) with the following values:
   - Your Document Intelligence endpoint.
   - Your Document Intelligence key.
   - The Model ID generated you provided when training your model. You can find this on the **Models** page of the Document Intelligence Studio. **Save** your changes.
3. In Visual Studio Code, open the code file for your client application (*Program.cs* for C#, test-model.py for Python) and review the code it contains, particularly that the image in the URL refers to the file in this GitHub repo on the web.
4. Return the terminal, and enter the following command to run the program:

   C#

   `dotnet build dotnet run`

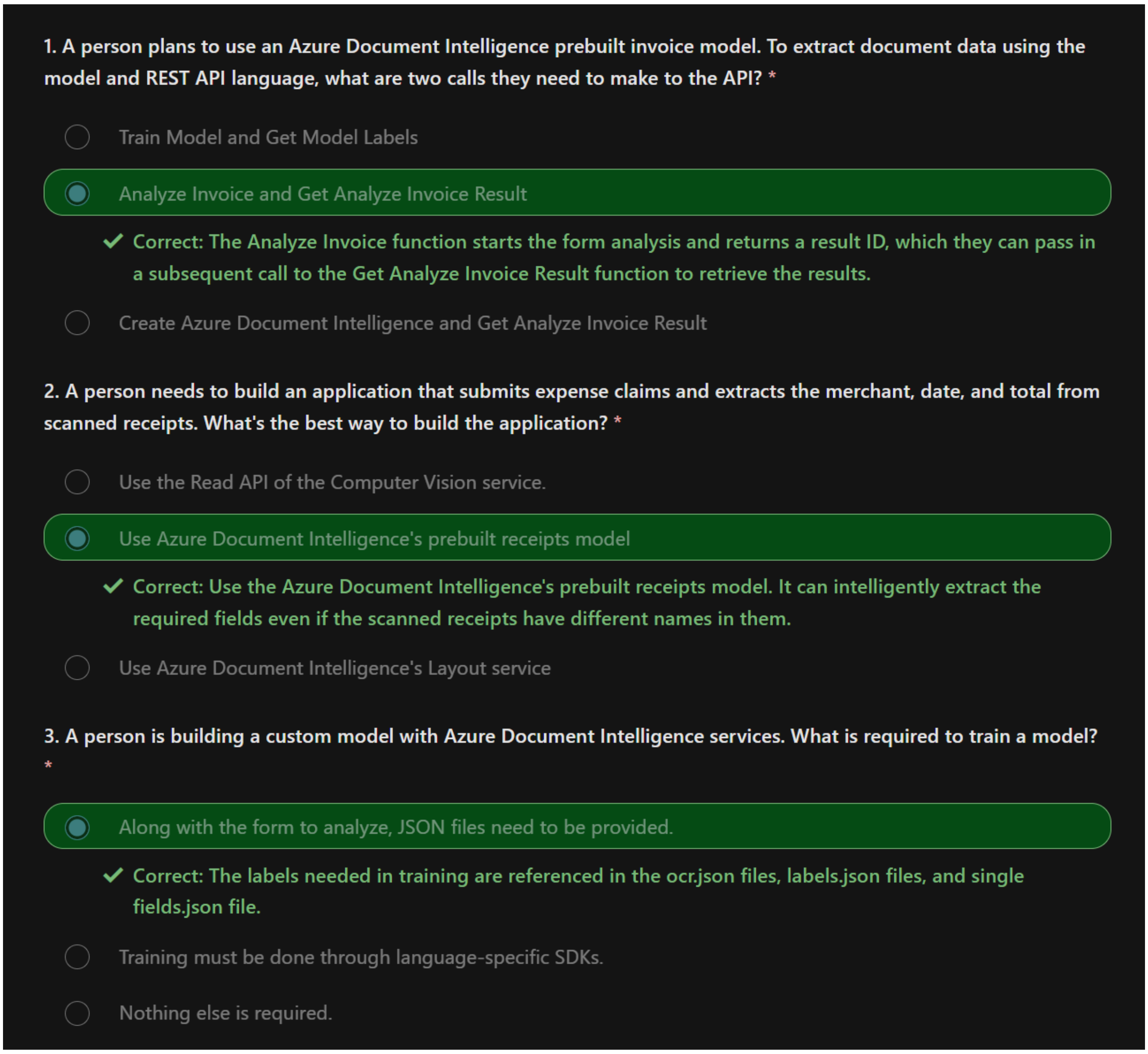
   Python

   `python test-model.py`
5. View the output and observe how the output for the model provides field names like `Merchant` and `CompanyPhoneNumber`.

Clean up

If you're done with your Azure resource, remember to delete the resource in the Azure portal to avoid further charges.

Knowledge Check


1. A person plans to use an Azure Document Intelligence prebuilt invoice model. To extract document data using the model and REST API language, what are two calls they need to make to the API? *

- ○ Train Model and Get Model Labels
- [✓ Analyze Invoice and Get Analyze Invoice Result]

  ✓ Correct: The Analyze Invoice function starts the form analysis and returns a result ID, which they can pass in a subsequent call to the Get Analyze Invoice Result function to retrieve the results.
- ○ Create Azure Document Intelligence and Get Analyze Invoice Result

2. A person needs to build an application that submits expense claims and extracts the merchant, date, and total from scanned receipts. What's the best way to build the application? *

- ○ Use the Read API of the Computer Vision service.
- [✓ Use Azure Document Intelligence's prebuilt receipts model]

  ✓ Correct: Use the Azure Document Intelligence's prebuilt receipts model. It can intelligently extract the required fields even if the scanned receipts have different names in them.
- ○ Use Azure Document Intelligence's Layout service

3. A person is building a custom model with Azure Document Intelligence services. What is required to train a model? *

- [✓ Along with the form to analyze, JSON files need to be provided.]

  ✓ Correct: The labels needed in training are referenced in the ocr.json files, labels.json files, and single fields.json file.
- ○ Training must be done through language-specific SDKs.
- ○ Nothing else is required.

Summary

This module focused on Azure Document Intelligence's prebuilt service, custom service, and its client library SDKs and REST API. We also introduced the Azure Document Intelligence Studio to label and train your model.

Azure Document Intelligence services can be integrated with other Azure AI Services. For example, you can try using this tutorial with Azure Document Intelligence and AI Search.

Document intelligence is just one part of the overall Vision API in Azure AI Services. Azure Document Intelligence services are ever evolving. You can read about the latest updates here.

✍️ Compiled by Kenneth Leung (2025)