Create a Azure AI Document Intelligence resource

Prepare to develop an app in Cloud Shell

Gather documents for training

Train the model using Document Intelligence Studio

Test your custom Document Intelligence model

Clean up

More information

Analyze forms with custom Azure AI Document Intelligence models

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.

While this exercise is based on Python, you can develop similar applications using multiple language-specific SDKs; including:

- Azure AI Document Intelligence client library for Python
- Azure AI Document Intelligence client library for Microsoft .NET
- Azure AI Document Intelligence client library for JavaScript

This exercise takes approximately **30** minutes.

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, create a new resource with the following settings:
    - **Subscription**: Your Azure subscription.
    - **Resource group**: Create or select a resource group
    - **Region**: Any available region
    - **Name**: A valid name for your Document Intelligence resource
    - **Pricing tier**: Free F0 (*if you don't have a Free tier available, select* Standard S0).
5. When the deployment is complete, select **Go to resource** to view the resource's **Overview** page.

Prepare to develop an app in Cloud Shell

You'll develop your text translation app using Cloud Shell. The code files for your app have been provided in a GitHub repo.

1. In the Azure Portal, use the **[>_]** button to the right of the search bar at the top of the page to create a new Cloud Shell in the Azure portal, selecting a ***PowerShell*** environment. The cloud shell provides a command line interface in a pane at the bottom of the Azure portal.

    > **Note**: If you have previously created a cloud shell that uses a *Bash* environment, switch it to ***PowerShell***.

2. Size the cloud shell pane so you can see both the command line console and the Azure portal. You'll need to use the the split bar to switch as you switch between the two panes.

3. In the cloud shell toolbar, in the **Settings** menu, select **Go to Classic version** (this is required to use the code editor).

   Ensure you've switched to the classic version of the cloud shell before continuing.

4. In the PowerShell pane, enter the following commands to clone the GitHub repo for this exercise:

   Code Copy

   ```
   rm -r mslearn-ai-info -f
   git clone https://github.com/microsoftlearning/mslearn-ai-information-extraction mslearn-ai-info
   ```

   > **Tip**: As you paste commands into the cloudshell, the ouput may take up a large amount of the screen buffer. You can clear the screen by entering the `cls` command to make it easier to focus on each task.

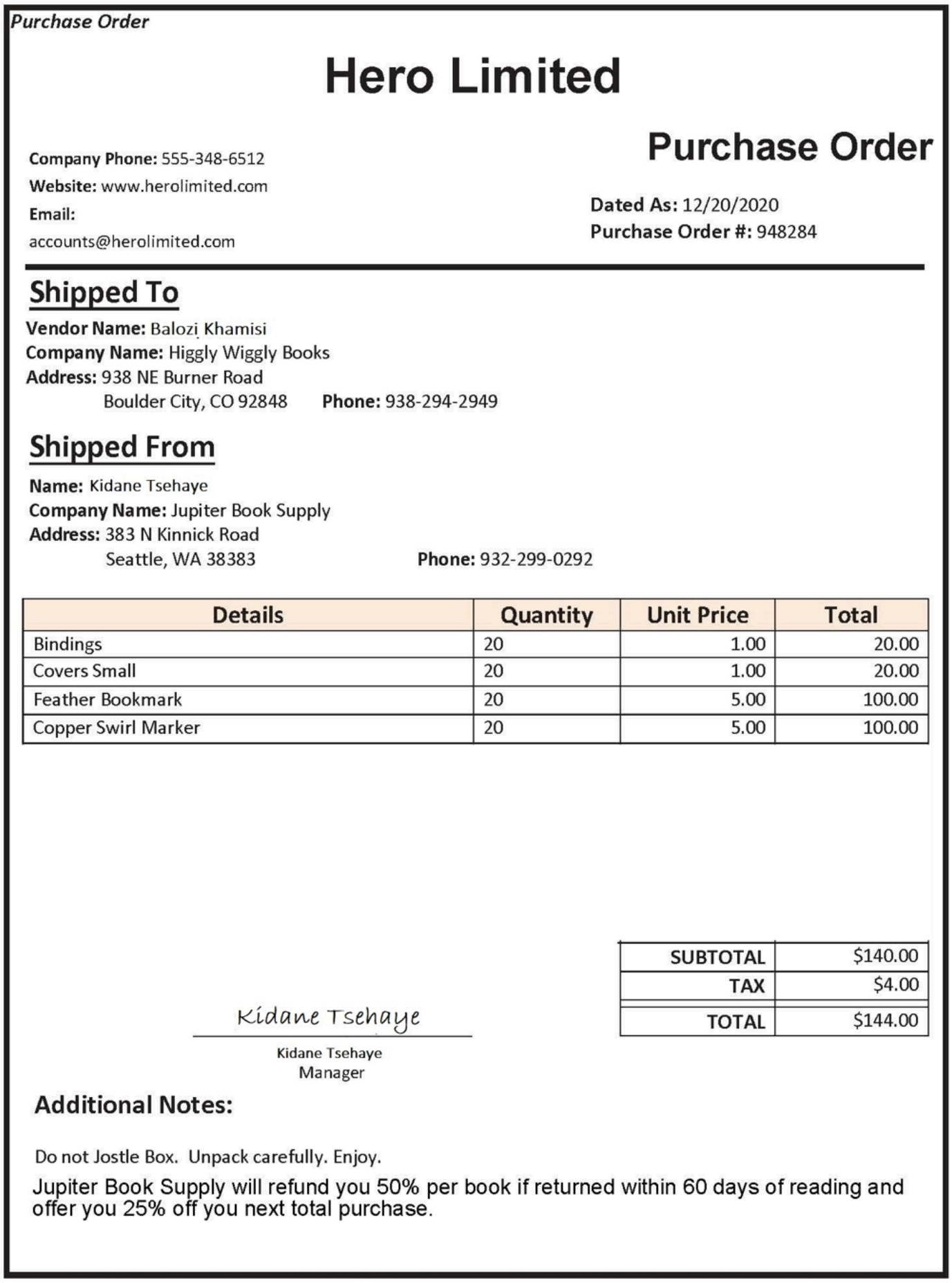
5. After the repo has been cloned, navigate to the folder containing the application code files:

   Code Copy

   ```
   cd mslearn-ai-info/Labfiles/custom-doc-intelligence
   ```

Gather documents for training

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



Purchase Order

Hero Limited

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

Purchase Order

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. In the command line, run `ls ./sample-forms` to list the content in 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.
2. In the **Azure portal** and navigate to your resource's **Overview** page if you're not already there. Under the *Essentials* section, note the **Resource group**, **Subscription ID**, and **Location**. You will need these values in subsequent steps.

3. Run the command `code setup.sh` to open **setup.sh** in a code editor. You will use this script to run the Azure command line interface (CLI) commands required to create the other Azure resources you need.
4. In the **setup.sh** 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
   - Print a Shared Access Signature URI

5. 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.

   > **Important**: For your **location** string, be sure to use the code version of your location. For example, if your location is "East US", the string in your script should be `eastus`. You can see that version is the **JSON View** button on the right side of the **Essentials** tab of your resource group in Azure portal.

   If the **expiry_date** variable is in the past, update it to a future date. 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.

6. After you've replaced the placeholders, within the code editor, use the **CTRL+S** command or **Right-click > Save** to save your changes and then use the **CTRL+Q** command or **Right-click > Quit** to close the code editor while keeping the cloud shell command line open.

7. Enter the following commands to make the script executable and to run it:

   Code Copy

   ```
   chmod +x ./setup.sh
   ./setup.sh
   ```

8. When the script completes, review the displayed output.
9. 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 **custom-doc-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. Open a new browser tab, and 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 prompted, sign in with 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**, Create a new project with the following configuration:
   - **Enter project details**:
     - **Project name**: A valid name for your project
   - **Configure service resource**:
     - **Subscription**: Your Azure subscription
     - **Resource group**: The resource group where you deployed your Document Intelligence resource
     - **Document intelligence resource** Your Document Intelligence resource (select the *Set as default* option and use the default API version)
   - **Connect training data source**:
     - **Subscription**: Your Azure subscription
     - **Resource group**: The resource group where you deployed your Document Intelligence resource
     - **Storage account**: The storage account that was created by the setup script (select the *Set as default* option, select the `sampleforms` blob container, and leave the folder path blank)
6. When your project is created, on the top right of the page, select **Train** to train your model. Use the following configurations:
   - **Model ID**: A valid name for your model (*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. Wait until the status is **succeeded**.

Test your custom Document Intelligence model

1. Return to the browser tab containing the Azure Portal and cloud shell. In the command line, run the following command to change to the folder containing the application code files:

```
cd Python
```

2. Install the Document Intelligence package by running the following command:

```
python -m venv labenv
./labenv/bin/Activate.ps1
pip install -r requirements.txt azure-ai-formrecognizer==3.3.3
```

3. Enter the following command to edit the configuration file that has been provided:

```
code .env
```

4. In the pane containing the Azure portal, on the **Overview** page for your Document Intelligence resource, select **Click here to manage keys** to see the endpoint and keys for your resource. Then edit the configuration file with the following values:
   - Your Document Intelligence endpoint
   - Your Document Intelligence key
   - The Model ID you specified when training your model
5. After you've replaced the placeholders, within the code editor, use the **CTRL+S** command to save your changes and then use the **CTRL+Q** command to close the code editor while keeping the cloud shell command line open.
6. Open the code file for your client application (`code Program.cs` for C#, `code 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. Close the file without making any changes.
7. In the command line, and enter the following command to run the program:

```
python test-model.py
```

8. 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.

More information

For more information about the Document Intelligence service, see the Document Intelligence documentation.