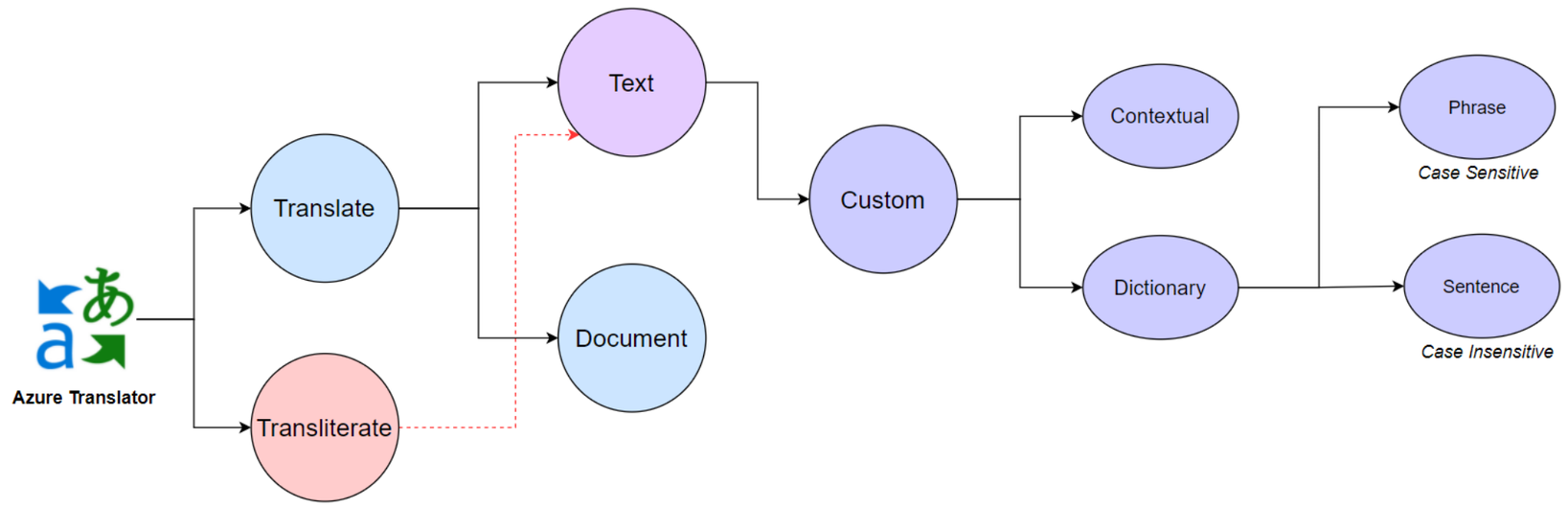
Document Name	HOL – Azure Translator Service v3.2
Author	Shiva S Tomar & Anupreet Kaur
Reviewer	
Executive Summary	Azure Cognitive APIs enable the developers of all skill levels to add human intelligence in their applications. The services are designed for developers interested in pursuing DS/AI/ML skills and people who want to acquire the deep technical knowledge on the Cognitive APIs of Azure, despite not having Machine Learning expertise.
Purpose	This document is created to help you gain level 350 working knowledge on Azure Translator Service. You will be able to explore each functionality offered by the service through the GUI Portal & REST APIs and observe the outcomes. We have also shared a sample dataset to replicate what we have used to create the content of this workshop. Once you complete these labs, you'll go from ***Zero to Hero*** on the respective Azure Cognitive service and should be able to ***D**emo, **D**evelop and **D**eploy* your own custom use cases. The important thing to note here is that you don't need to refer any other documents to complete this workshop.
Intent of Guide	This workshop is designed to help you explore all the features of a service offered through their APIs. The diagram shown in the beginning of the document is its functional Architecture; talking about the functionalities offered by the service in a flow. It also covers the Concepts, How-to and best practices about the service. This document is not intended to enable you with scenarios of deployment in production.

Service brief: Azure Translator Service

Azure's Translator Service allows you to translate & transliterate text to the language of your choice.

Diagram: Functional Architecture



The service offers 2 main capabilities :

1. **Translate** : Translate specified source language text into the target language text.
2. **Transliterate** : Map source language script or alphabet to a target language script or alphabet.

Translate works on raw text (Text Translator API) as well as any document containing text (Document Translator API), for Eg PowerPoint, Word Document, PDF etc.
Raw text is used to process real time text by leveraging the Text translator API. Document Translator API opens the document, translates the content and publishes the output translated document while maintaining the format. All of this is carried out in batch mode.

Azure Translator also allows you to create custom models for your use case or business domain to process real time text. Custom models can be created using 2 approaches :

1. **Contextual** : By passing sentence-based training document. This model takes context into consideration
2. **Dictionary** : This is like a 1:1 mapping for source to target translation. Phrase model is used for words & phrases and is Case Sensitive hence will input text to trained outcome only if there is a 100% match. Sentence model is used for sentence level mapping and is case insensitive. Use a dictionary in your training, when you want Microsoft Translator to always translate any instances of the source phrase or sentence, using the translation you've provided in the dictionary.

These APIs are available both as REST APIs and language specific SDKs.
You can deploy the models on the cloud or on the edge.

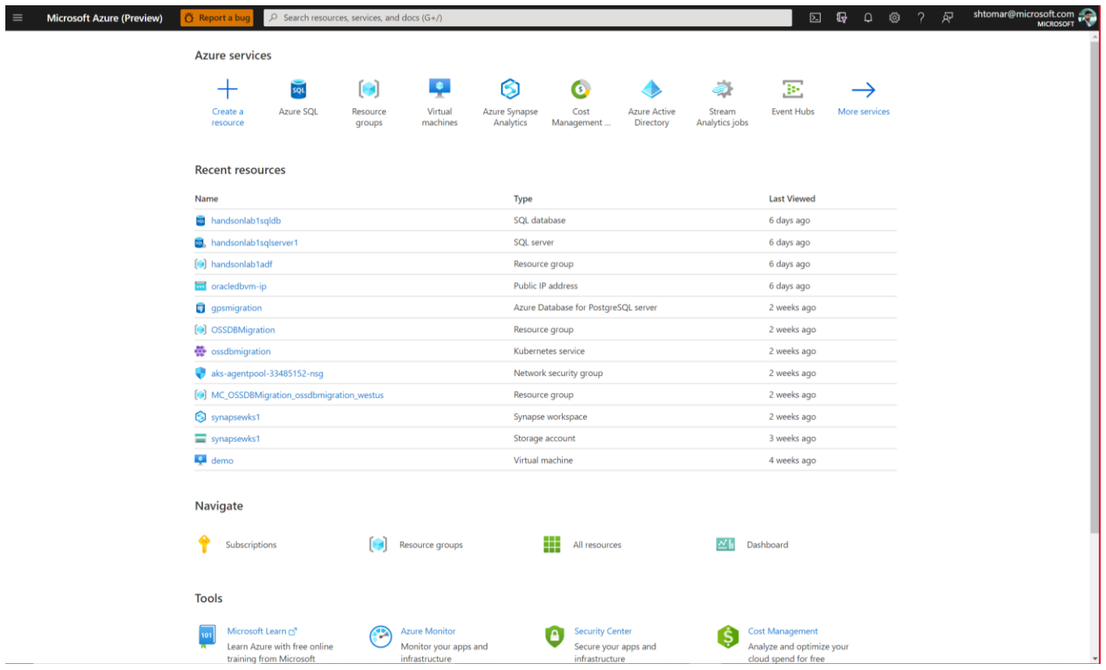
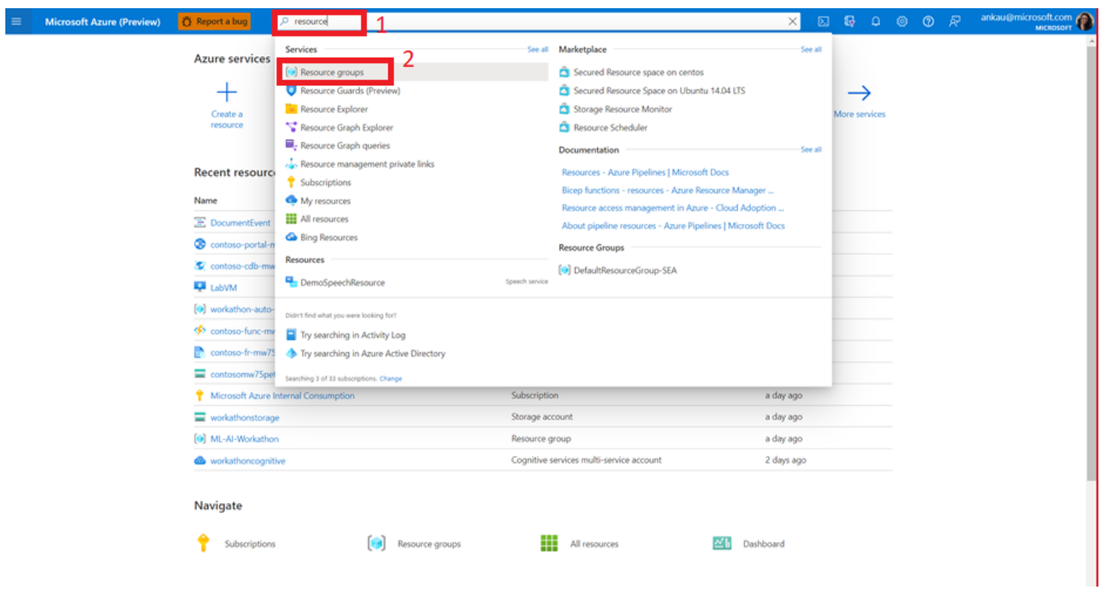
Step by step hands on guide to go from *Zero to Hero*

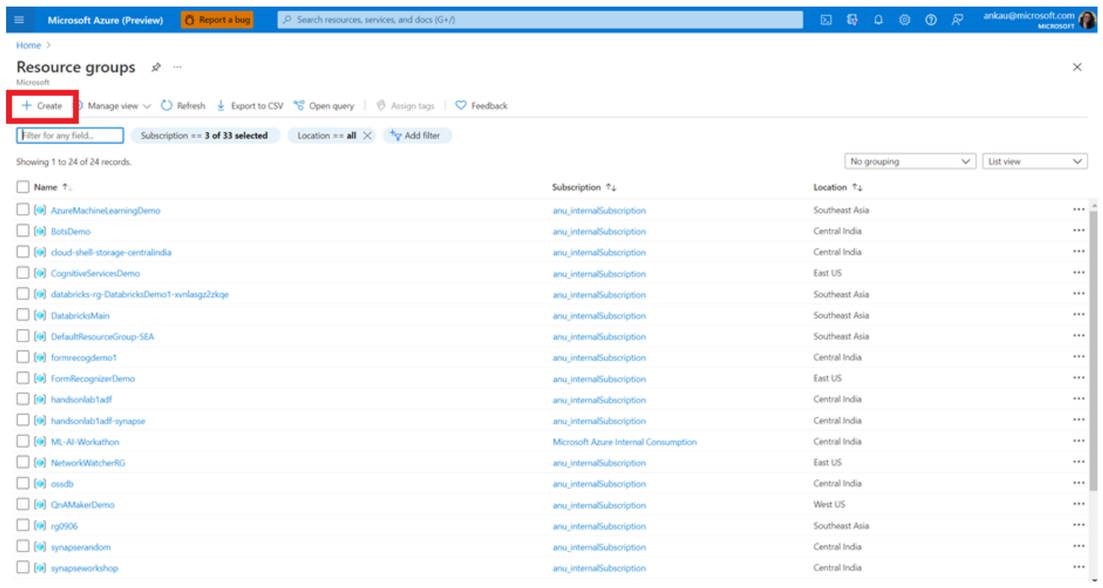
Pre-requisites

- Download & Install Postman
  - Postman is a free tool which allows you to make API calls
  - You can download the desktop application or get started using the web version (Download Postman | Try Postman for Free)

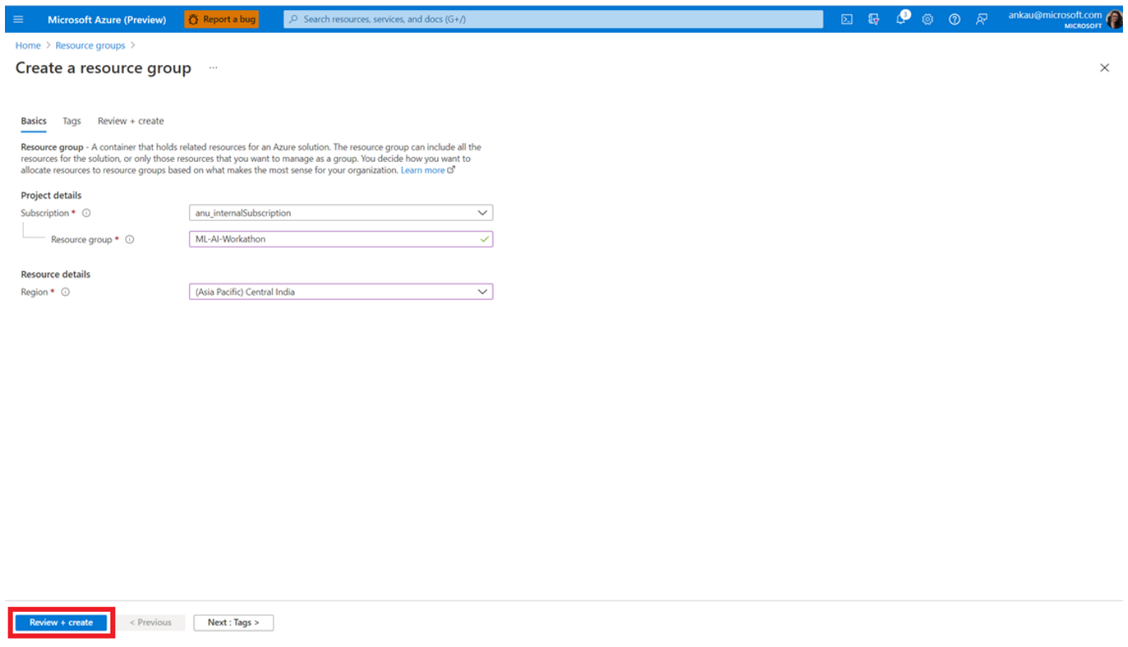
- An active Azure Account
    - You can use your current Azure Subscription or get started by creating a free trial account (https://azure.microsoft.com/en-in/free)
- Download the data from Data folder, if any.

Let's get started!

Screenshots	Steps & Significance
	Sign into your Azure Portal.
	<u>Create a Resource Group</u> Follow steps 1 & 2 to create a resource group. You can skip this step if you already have a Resource Group in place.



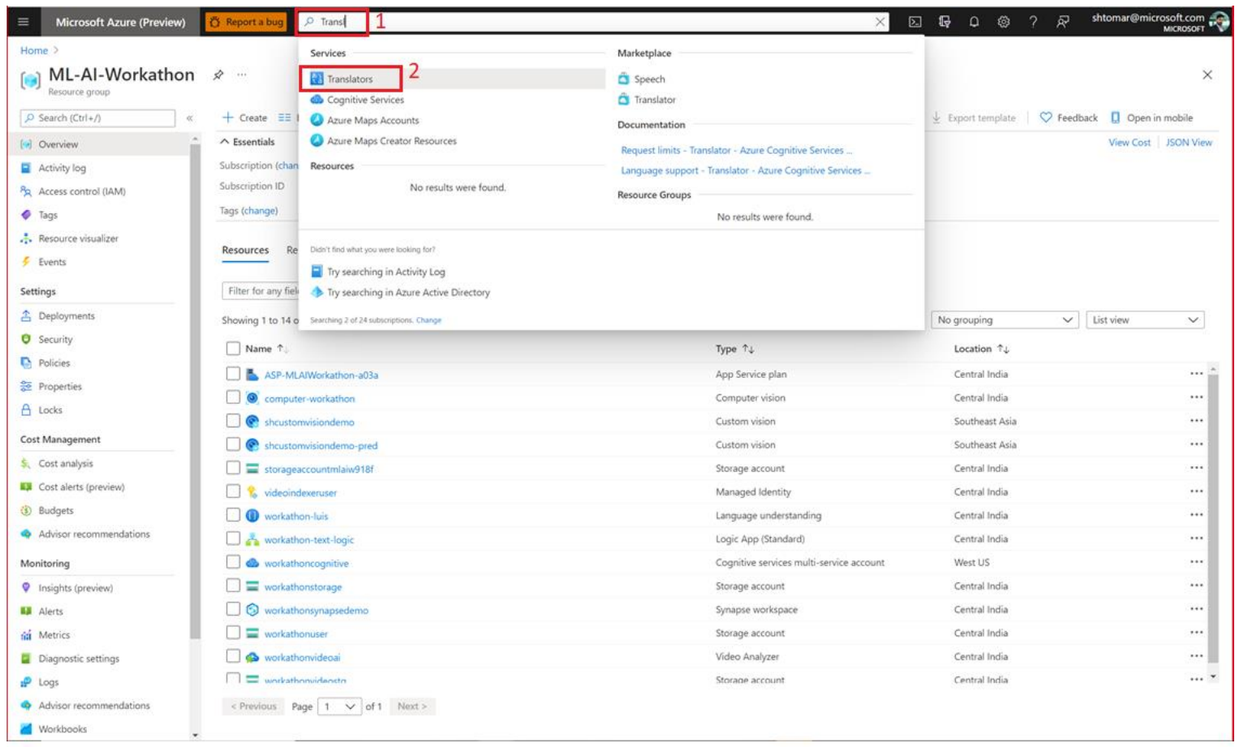
Click create to create a new resource group.



Enter the details –

1. Subscription : Azure subscription in which you want to deploy the resource group
2. Resource Group : Name of your choice for the resource group
3. Region : Region where you want to deploy the resource group

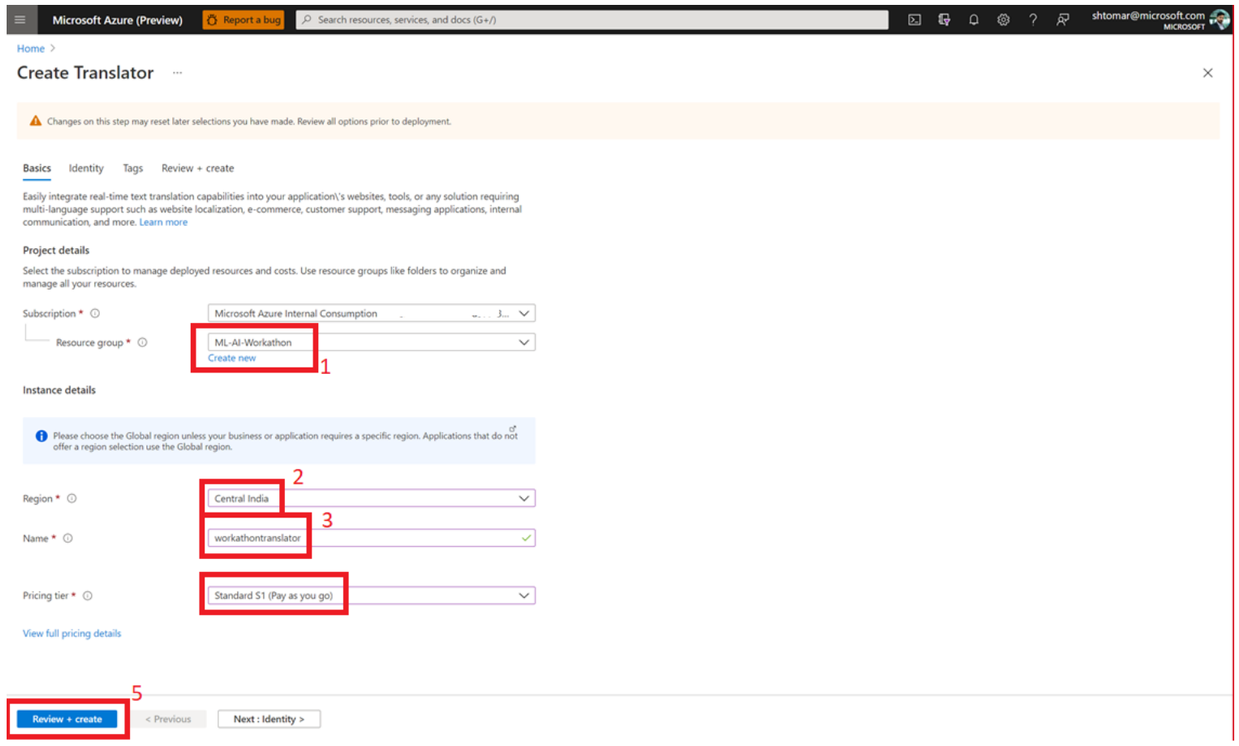
Click Review + Create.



Create Translator Service

Once the resource group is created, search for Translator in the search bar above and select Translator resource.

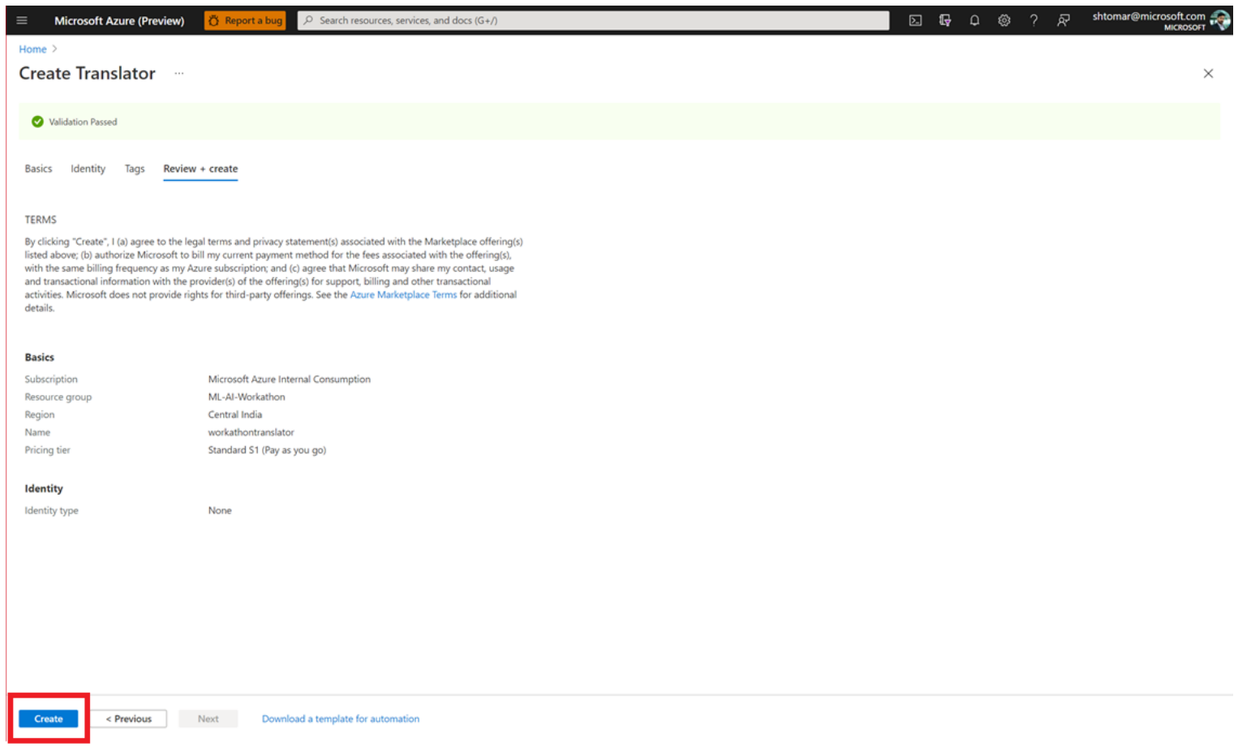
The multipurpose Cognitive Service account does not support advanced translator features like Document Translation. Thus, we have created a separate Translator Service account.



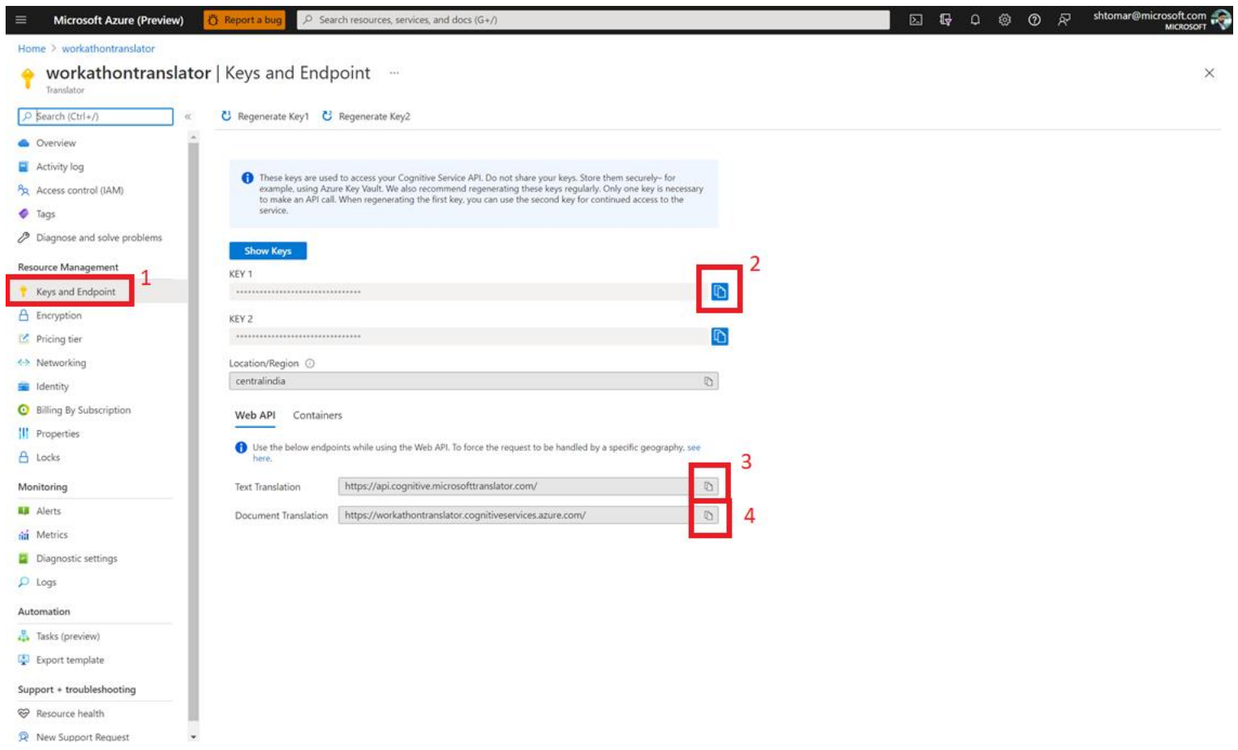
Enter the details to create a new cognitive service as follows -

Project details	Description
Subscription	Select one of your available Azure subscriptions.
Resource group	The Azure resource group that will contain your Cognitive Services resource. You can create a new group or add it to a pre-existing group.
Region	The location of your cognitive service instance. Different locations may introduce latency but have no impact on the runtime availability of your resource.
Name	A descriptive name for your cognitive services resource.
Pricing tier	The cost of your Cognitive Services account depends on the options you choose and your usage.

Click Review + Create.



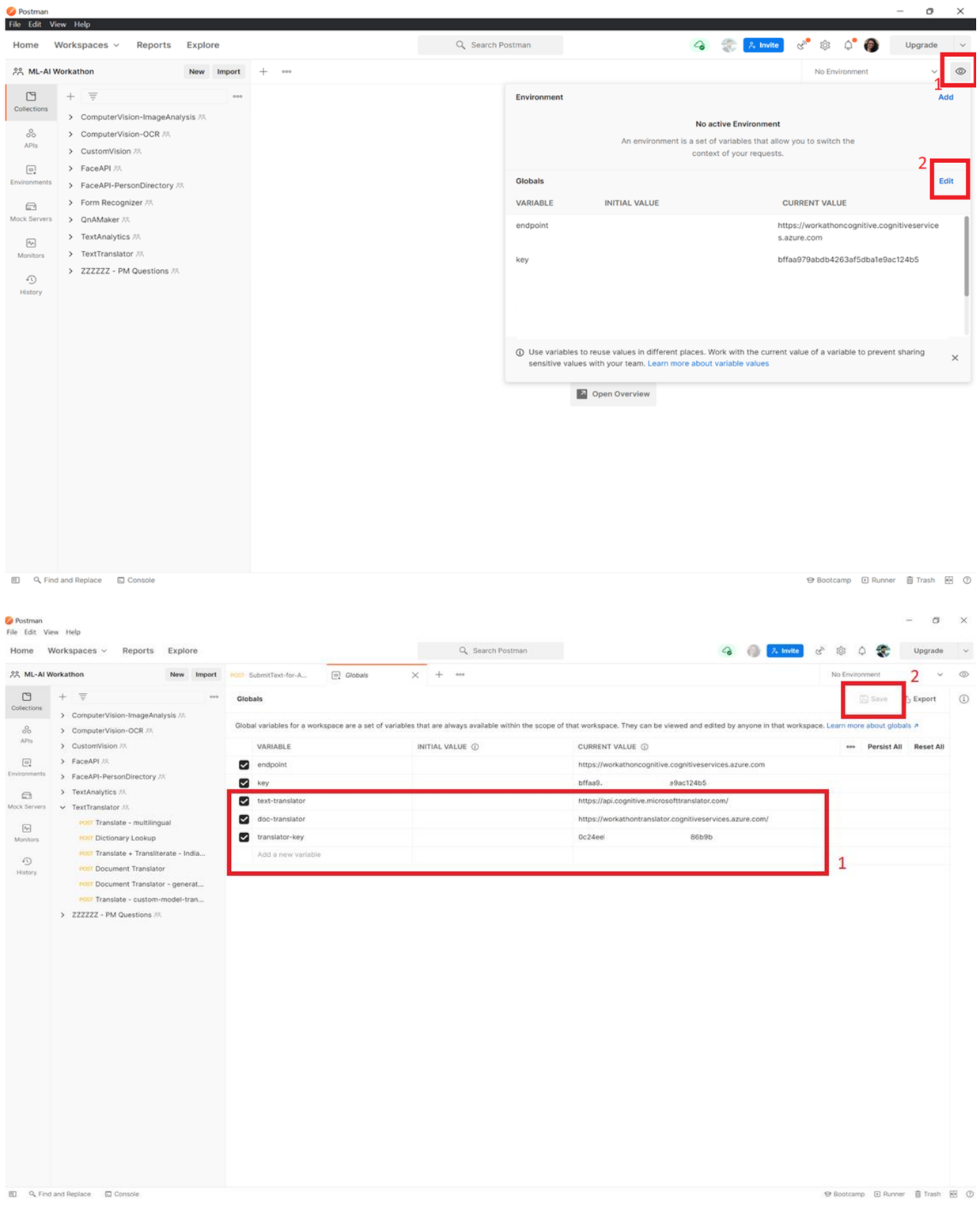
Verify the details and click Create.



<u>Copy keys & endpoints</u>

Click Key and Endpoints.

Copy the Key and Endpoint for Text Translation & Document Translation. Paste these in a notepad. You will leverage these in the next step, while setting up the global variables in Postman.



We have now switched the interface to Postman to explore the Text Analytics Service. If you haven't downloaded the Postman client, you can use web version.

Configure global variables in Postman

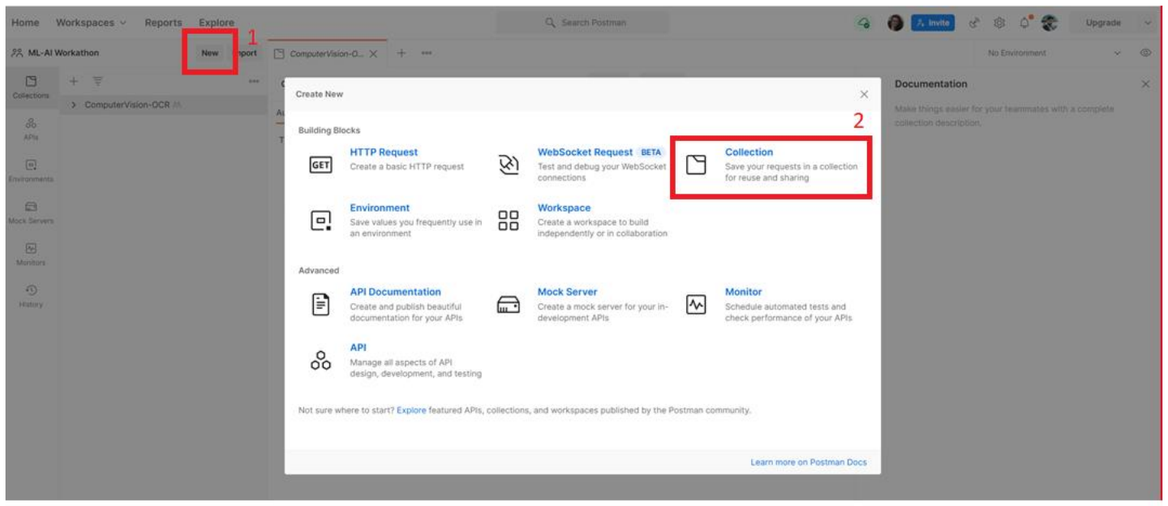
Significance :
The global variables will be created once and leveraged time & again, in each API request that we make through Postman.
This way, you will not have to hard code the Endpoint & Key for every request you make, thereby, making it more secure. This will also save time and effort.

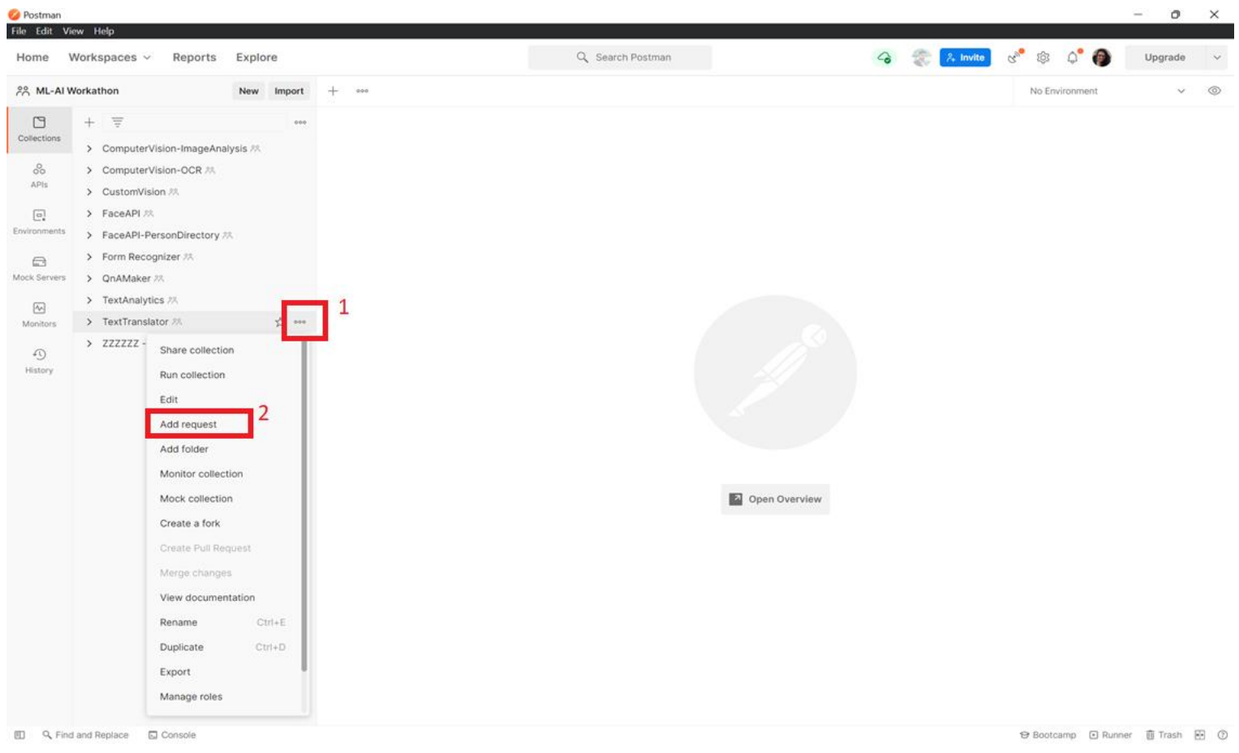
Follow step 1 & 2 to add global variables for :

1. Text-translator : Paste the endpoint of the Text Translator copied earlier
2. Document-translator : Paste the endpoint of the Document Translator copied earlier
3. Key - Paste the Key of the Translator service account copied earlier

Once you have added the global variables, they will appear in your global variables section, as shown in image 2.

Select Save.

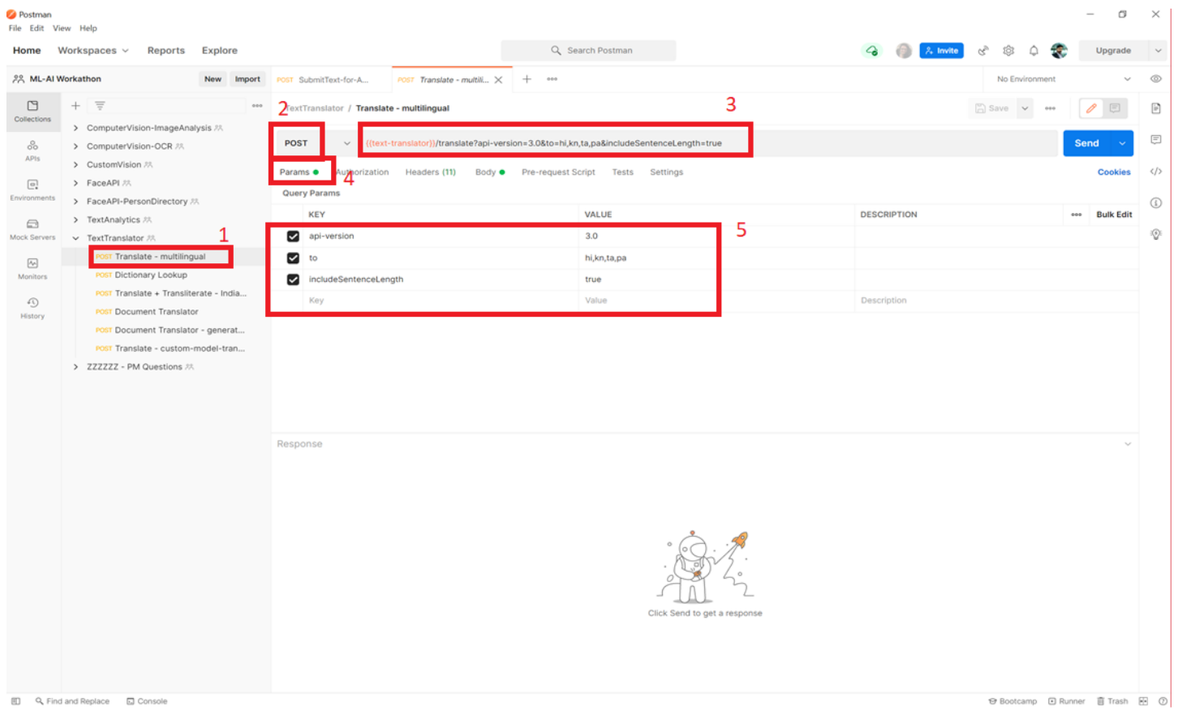


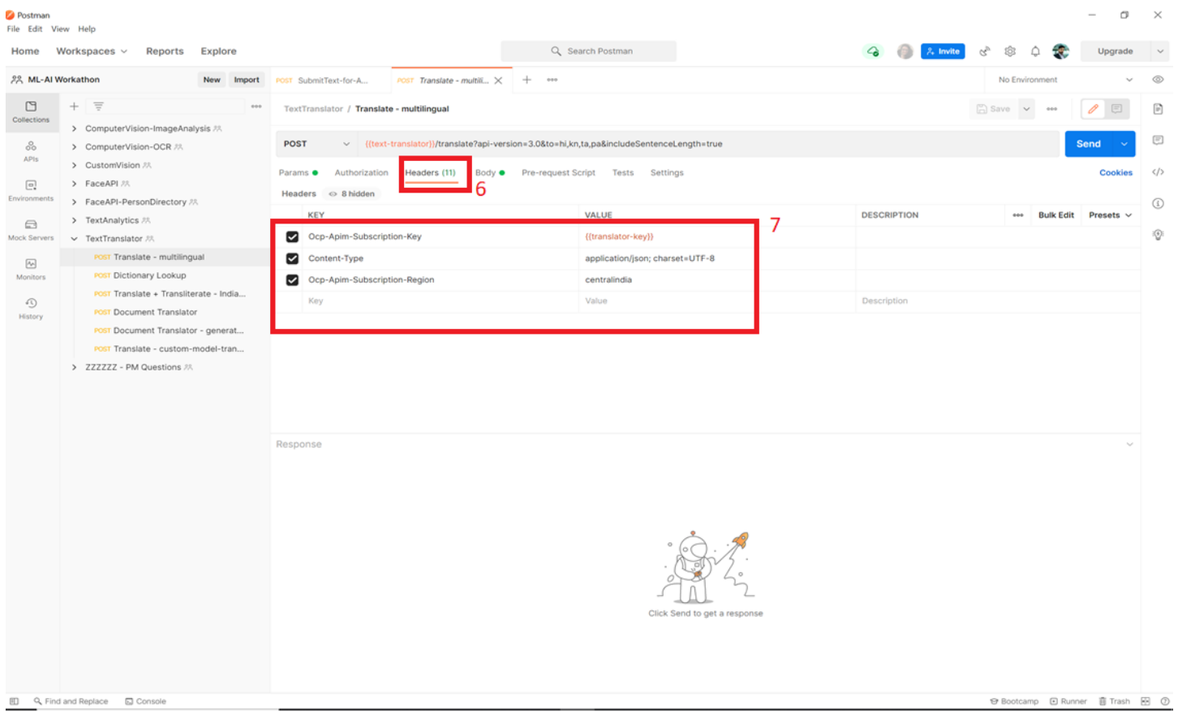


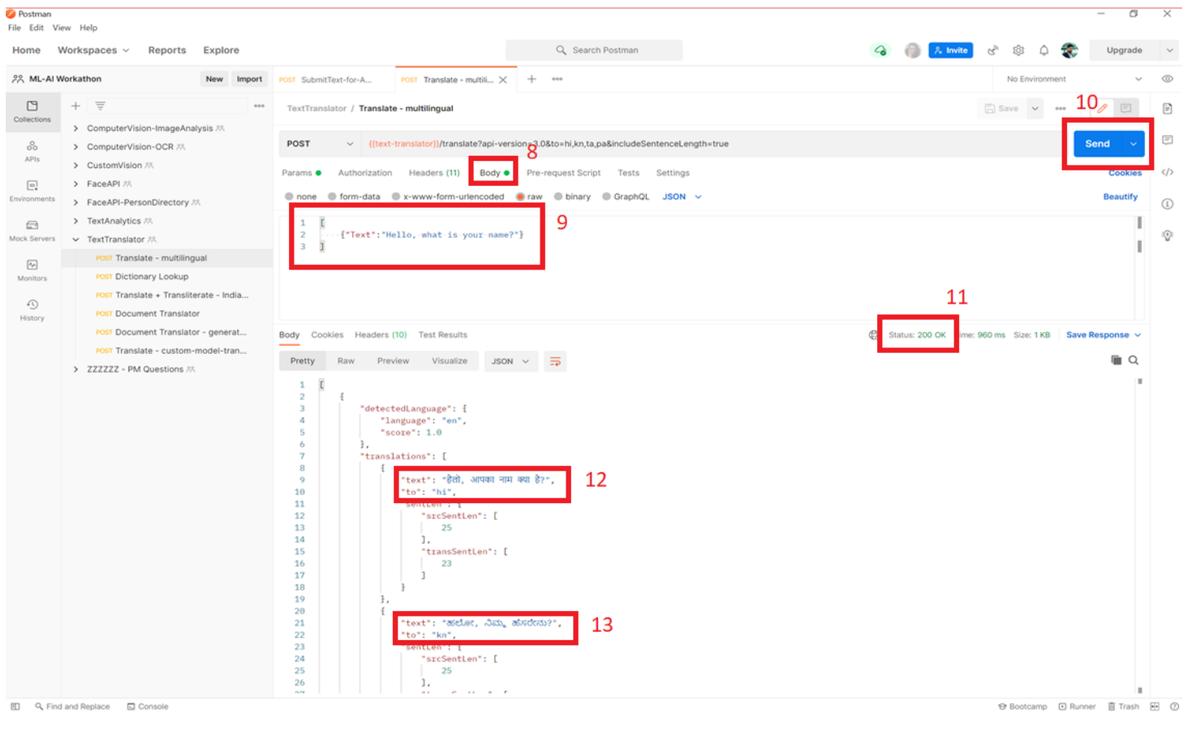
Create new collection in Postman

Open Postman > select New.
On the pop up select Collection.
Name the collection Text Translator.

Collection is like a folder for managing the API call requests.







Translate text to multiple languages

We will be translating text from English to multiple Indian Languages such as Hindi, Kannada, Tamil & Punjabi

This request upon successful execution will return text in each of the chosen languages.

URL : {{text-translator}}/translate?api-version=3.0&to=hi,kn,ta,pa&includeSentenceLength=true

Headers :
Ocp-Apim-Subscription-Key : {{key}}
Content-Type : application/json
Ocp-Apim-Subscription-Region : centralindia (change this if you provisioned your resource in a different region)

Body :

```
[
   {"Text":"Hello, what is your name?"}
]
```

You should also try exploring with different input text. You can add multiple text inputs within the array.

Params :
These will be auto-populated from the URL

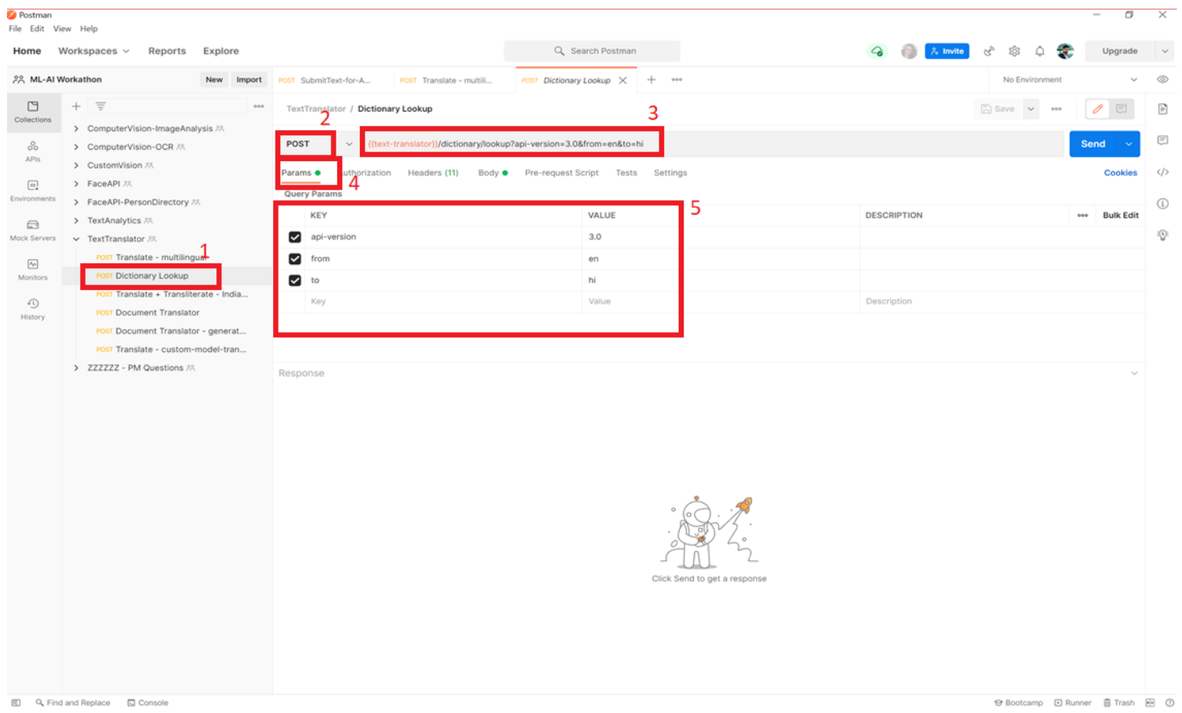
Significance of input & output

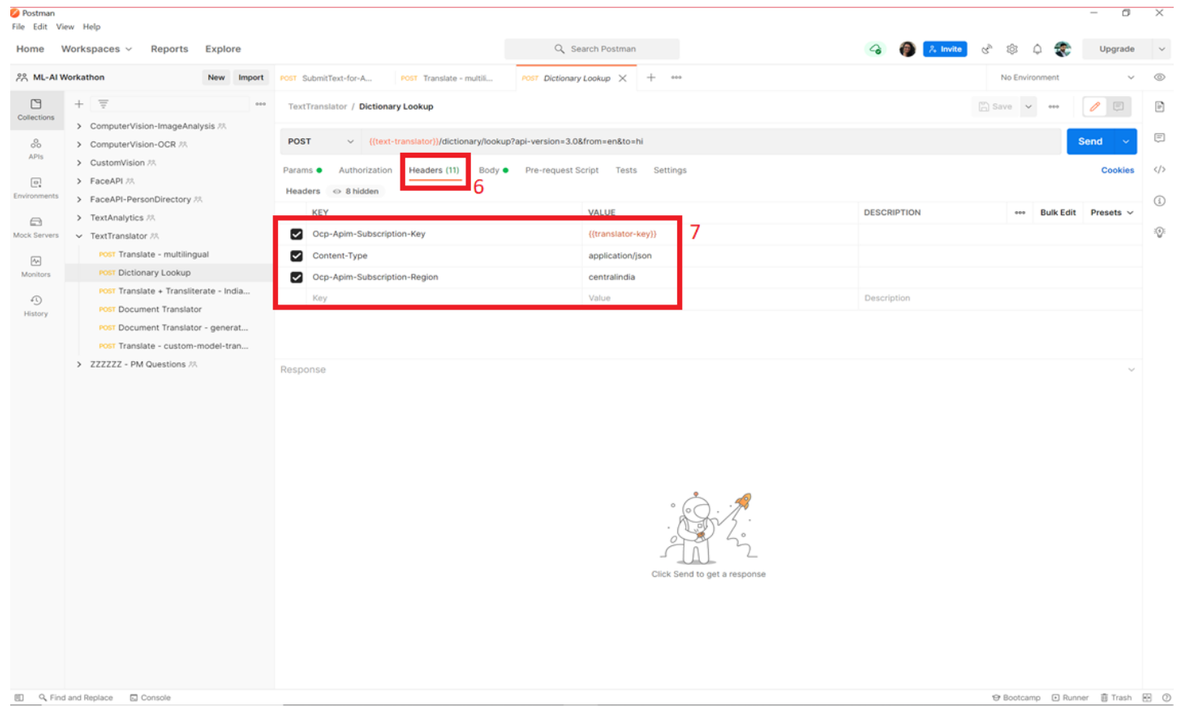
1. {{endpoint}} , {{key}} : Values being picked from global variables

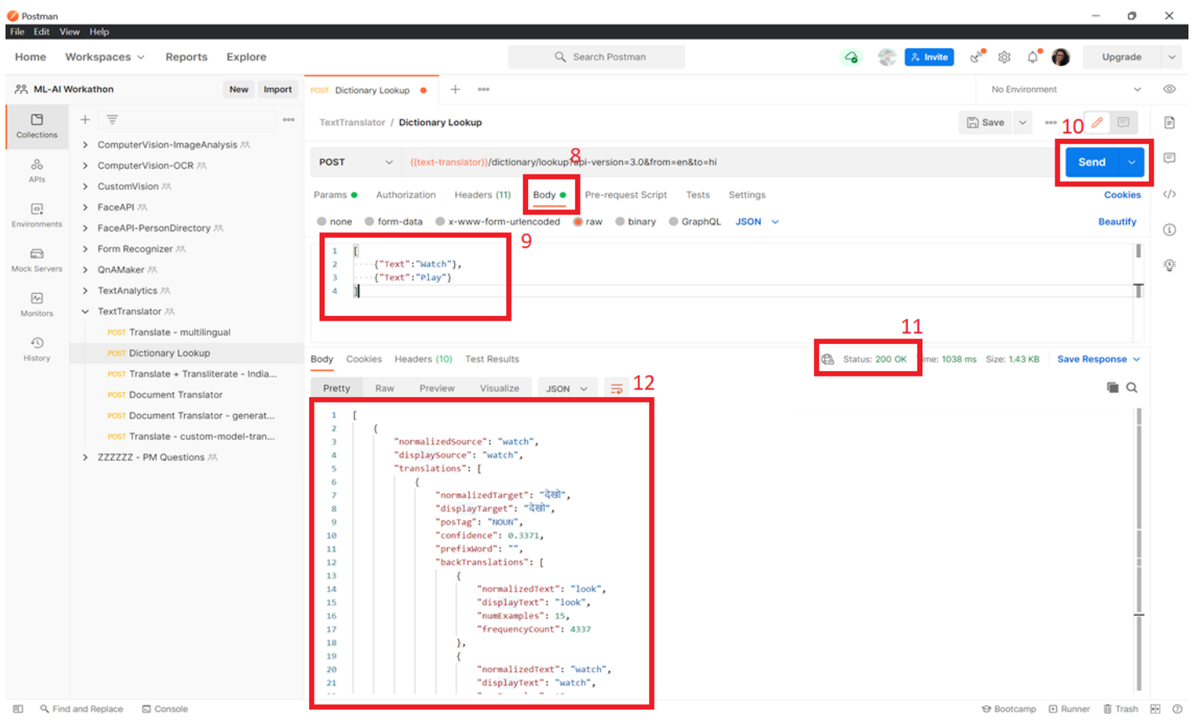
2. Ocp-Apim-Subscription-Key : This is the Azure Cognitive service key, that will authenticate the request.
   Content-Type : This refers to the input type that you provide in the body, for eg application/json allows you to enter body text in JSON format. Change the content-type on the basis of input you provide.
   Ocp-Apim-Subscription-Region : region where your resource is deployed.

3. Params :
   a. To : this takes as input the codes for the target languages, separated by a comma.
   b. includeSentenceLength : Setting this to true will return the length of the input & translated tedxt

4. After you execute the call, observe the status returned, as shown in step 11. This should reflect 200 OK. Observe the translated sentences.







<u>Dictionary Lookup API</u>

This returns a dictionary for individual words in the target language and reverse translation from target language to source language along with synonyms and related words.

URL : {{text-translator}}/dictionary/lookup

Headers :
Ocp-Apim-Subscription-Key : {{key}}
Content-Type : application/json
Ocp-Apim-Subscription-Region : centralindia (change this if you provisioned your resource in a different region)

Body :

```
[
   {"Text":"Watch"},
   {"Text":"Play"}
]
```

You should also try exploring with different input text. You can add multiple text inputs within the array. The value for each text should be a single word. This will not return any results upon adding more than 1 word in a line item.

Params :
api-version=3.0
from=en
to=hi

Once you have added the Params, they will be appended to the URL and it will look similar to the one in the screenshot.

<u>Significance of input & output</u>

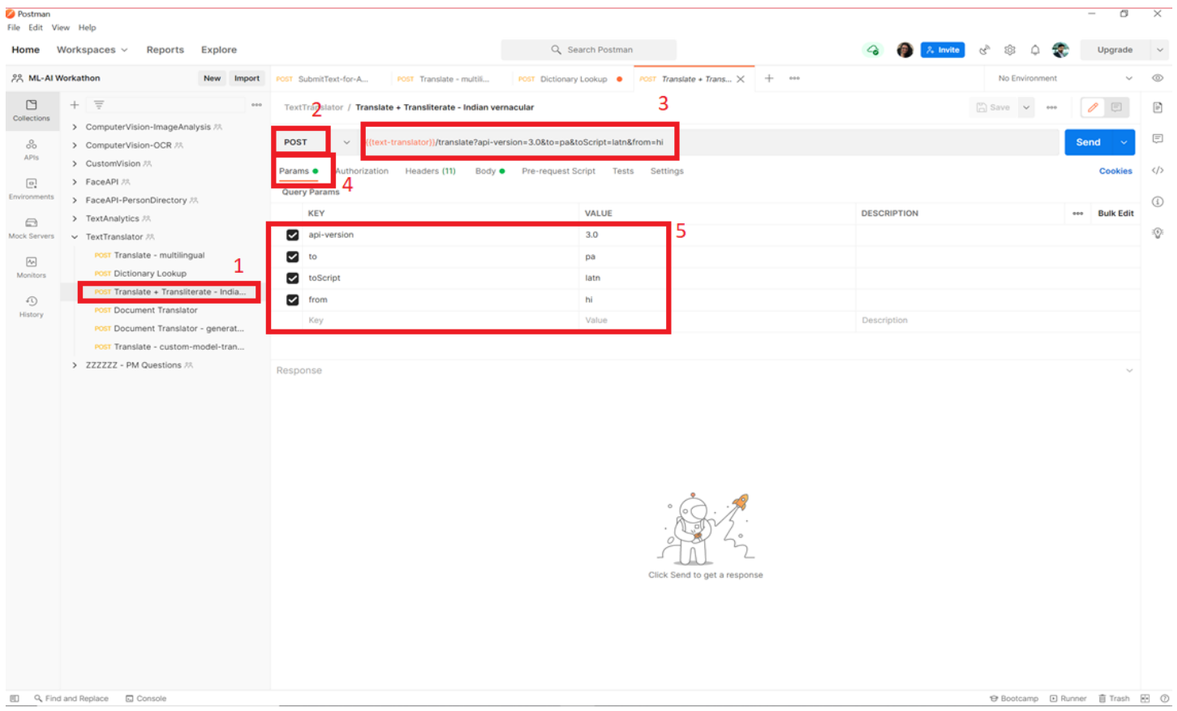
1. {{endpoint}} , {{key}} : Values being picked from global variables

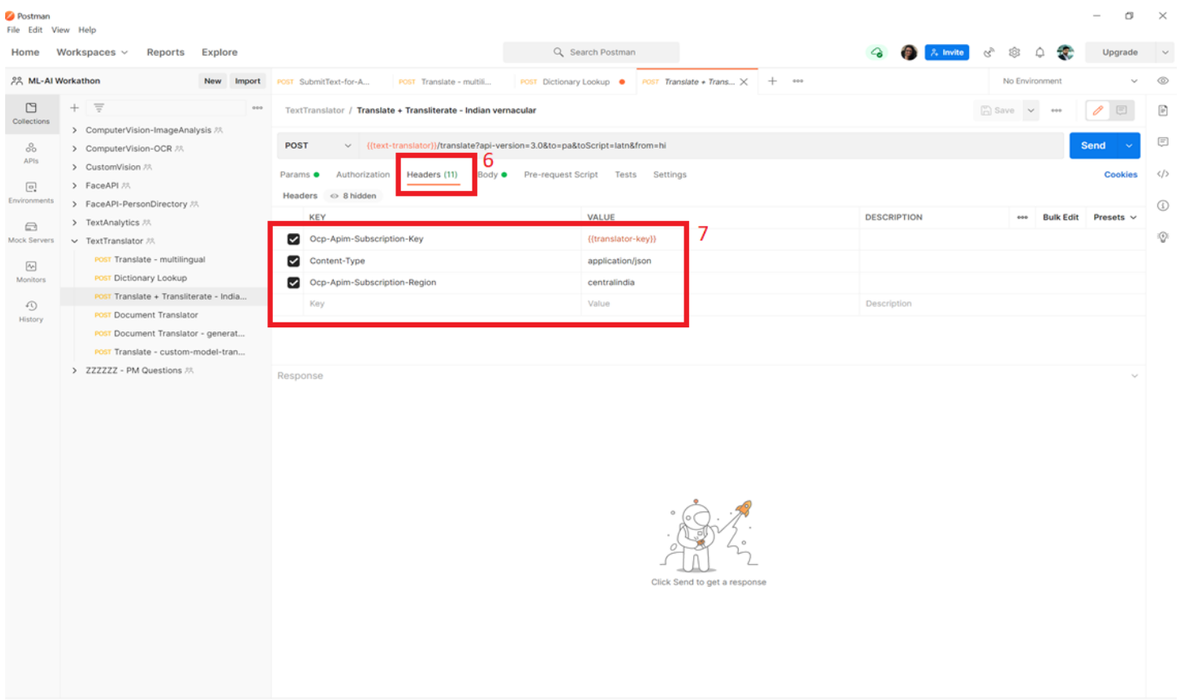
2. Ocp-Apim-Subscription-Key : This is the Azure Cognitive service key, that will authenticate the request.
   Content-Type : This refers to the input type that you provide in the body, for eg application/json allows you to enter body text in JSON format. Change the content-type on the basis of input you provide.
   Ocp-Apim-Subscription-Region : region where your resource is deployed.

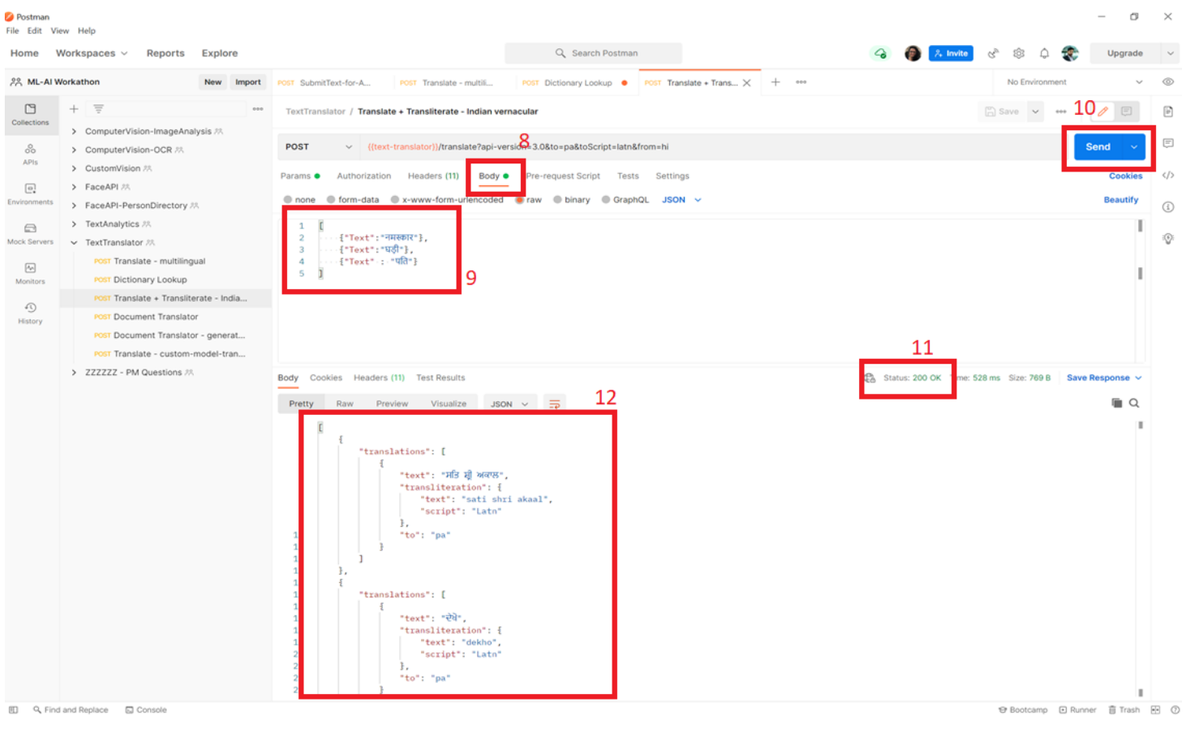
3. Params :
   c. from : this takes as input the codes for the source languages
   d. To : this takes as input the codes for the target languages

After you execute the call, observe the status returned, as shown in step 11. This should reflect 200 OK. Observe the translated sentences.

As part of the output, notice the translations & corresponding backTranslations for each of them.







Translate & Transliterate

We will use this API to translate between different Indian Vernacular Languages & transliterate the same to English.

URL : {{text-translator}}/translate

Headers :
Ocp-Apim-Subscription-Key : {{key}}
Content-Type : application/json
Ocp-Apim-Subscription-Region : centralindia (change this if you provisioned your resource in a different region)

Body :

```
[
   {"Text":"नमस्कार"},
   {"Text":"घड़ी"},
   {"Text" : "पति"}
]
```

You should also try exploring with different input text. You can add multiple text inputs within the array.

Params :
api-version=3.0
from=hi
to=pa
toScript=latn

Once you have added the Params, they will be appended to the URL and it will look similar to the one in the screenshot.

Significance of input & output

1. {{endpoint}} , {{key}} : Values being picked from global variables

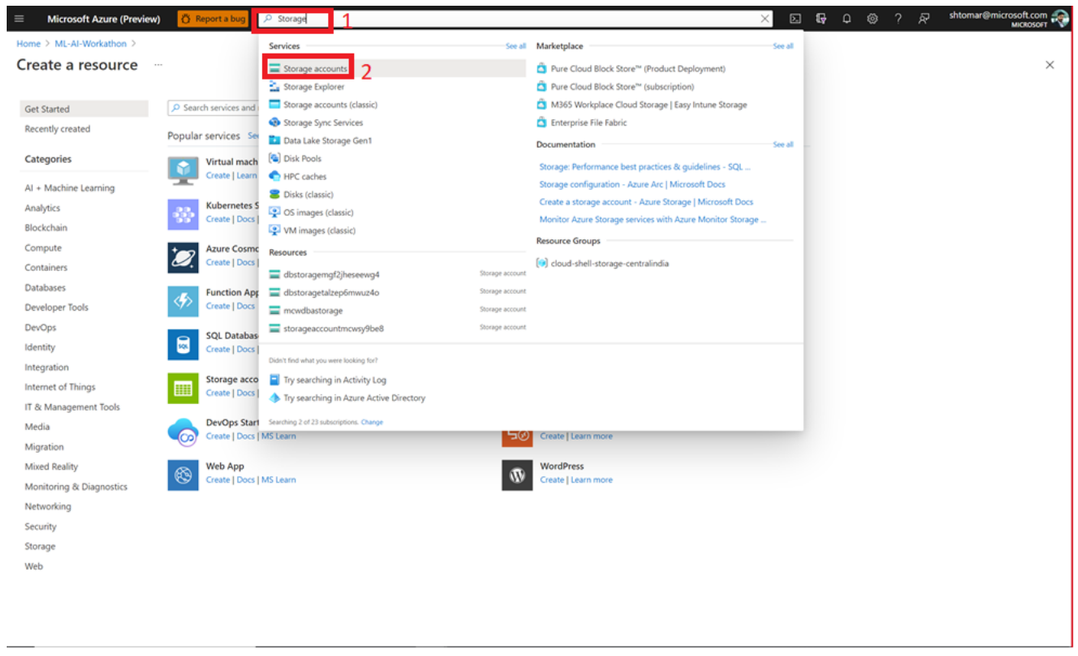
2. Ocp-Apim-Subscription-Key : This is the Azure Cognitive service key, that will authenticate the request.
    a. Content-Type : This refers to the input type that you provide in the body, for eg application/json allows you to enter body text in JSON format. Change the content-type on the basis of input you provide.
    b. Ocp-Apim-Subscription-Region : region where your resource is deployed.

3. Params :
    a. from : this takes as input the code for the source language
    b. To : this takes as input the codes for the target languages
    c. toScript : this takes as code the script for transliterated language

After you execute the call, observe the status returned, as shown in step 11. This should reflect 200 OK. Observe the translated sentences.

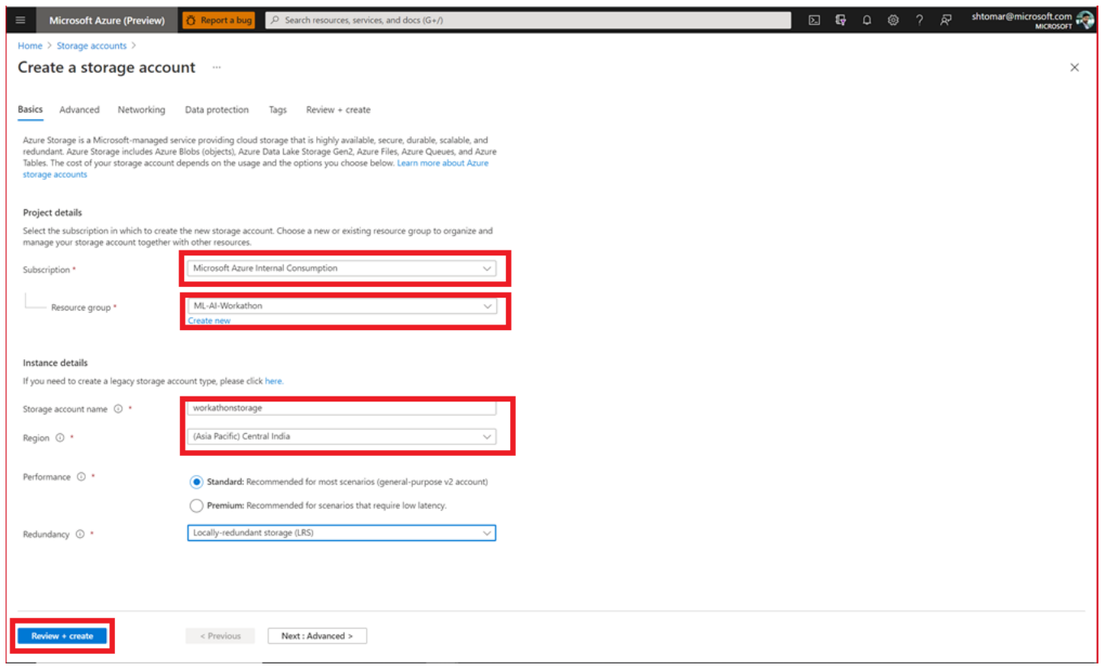
As part of the output, notice the translations & corresponding transliteration for each of them.

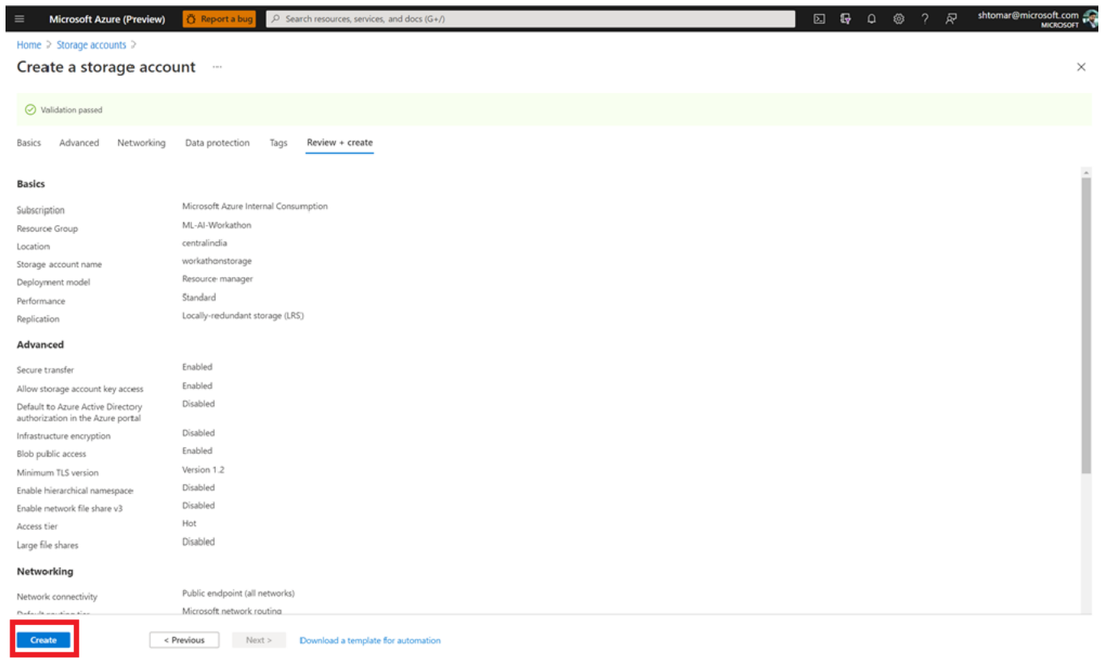
Document Translator



Create Storage Account in Azure Portal

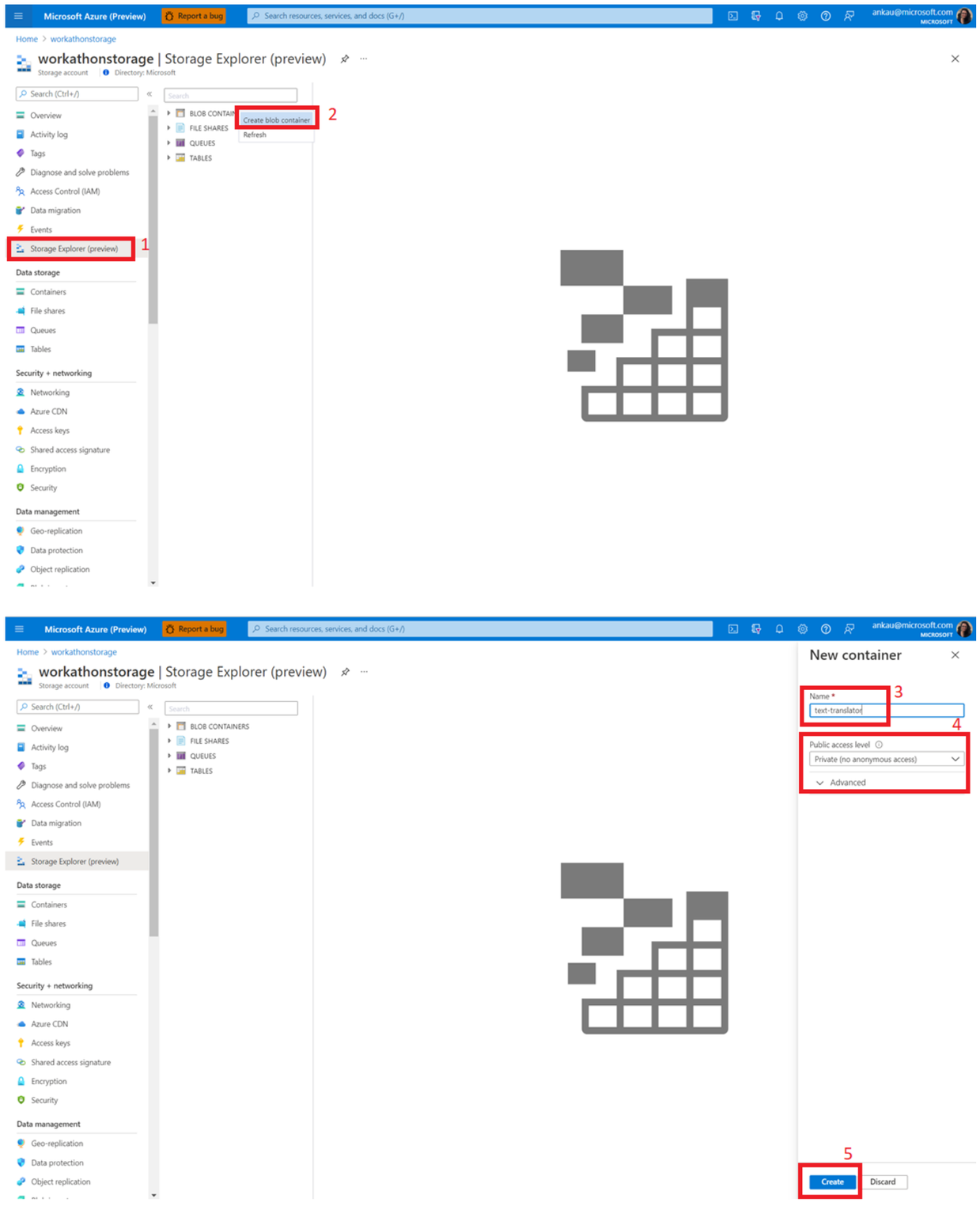
As pre-requisites for Document Translation API, we will be creating Azure blob storage container to store the source documents & a placeholder folder for target documents.





Enter the details highlighted in the screenshot.
Click Review + Create.

Verify the details and hit Create.



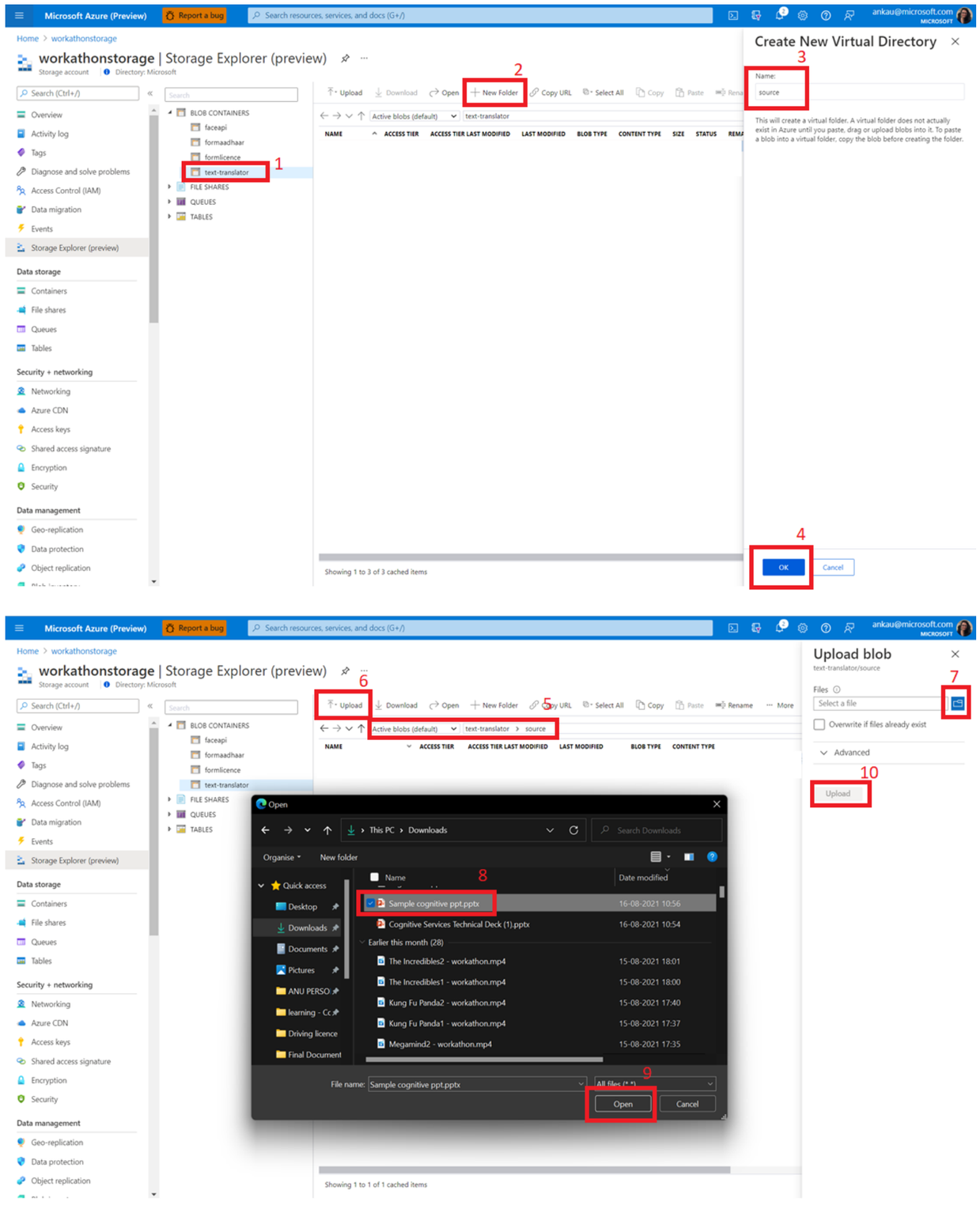
Create blob container

Go to the Storage Explorer (preview) to create a blob container.
Follows the steps as numbered in the screenshot. (step 1,2)

Create a container with the following details –

1. Name : **text-translator** [You can choose a different name, however, that will change the path to your images while making API calls from postman. Make sure to change the image paths accordingly]
2. Public access level : Private (no anonymous access)

Click Create.

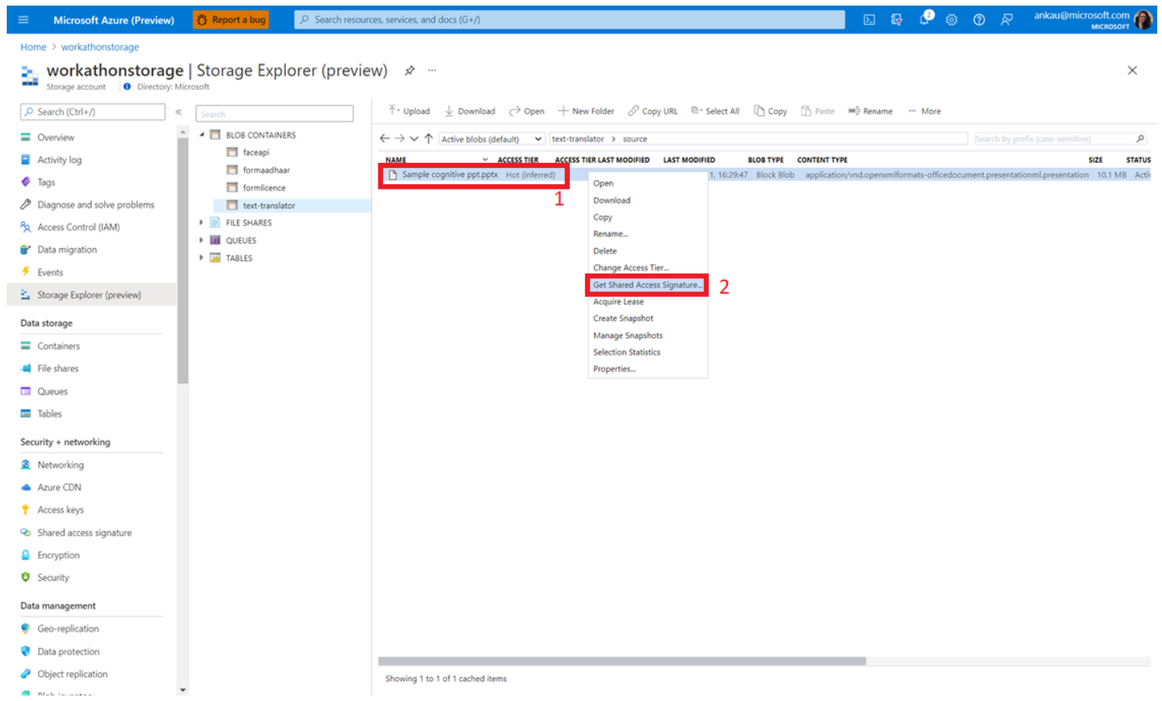


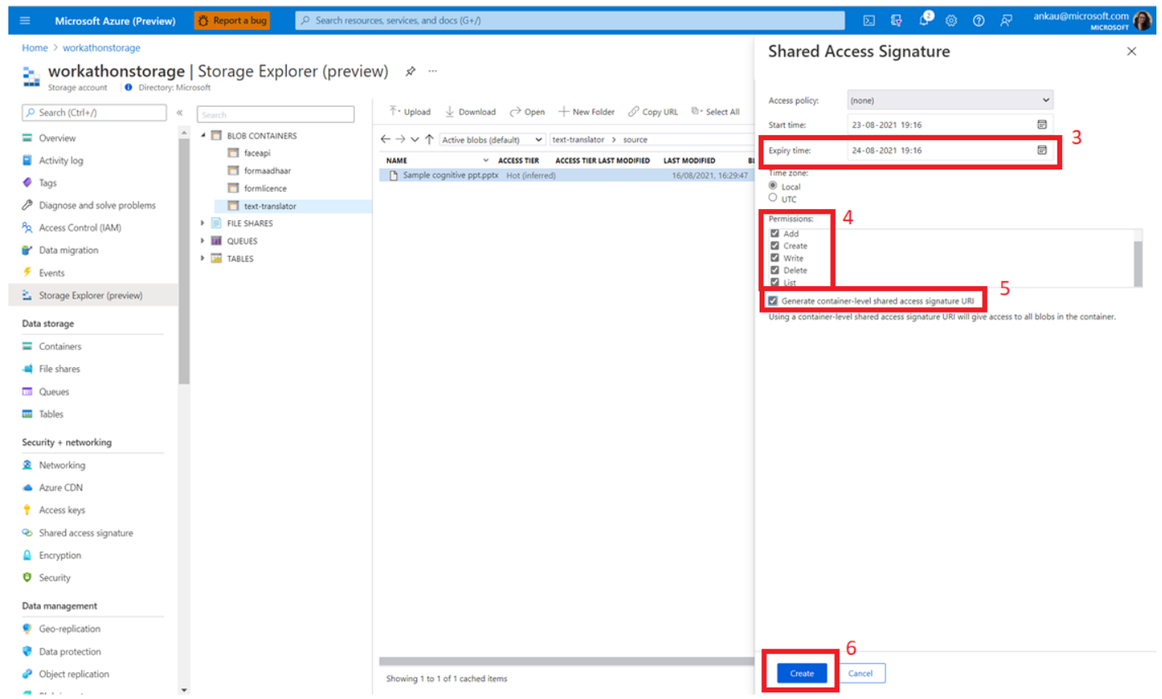
Create folder

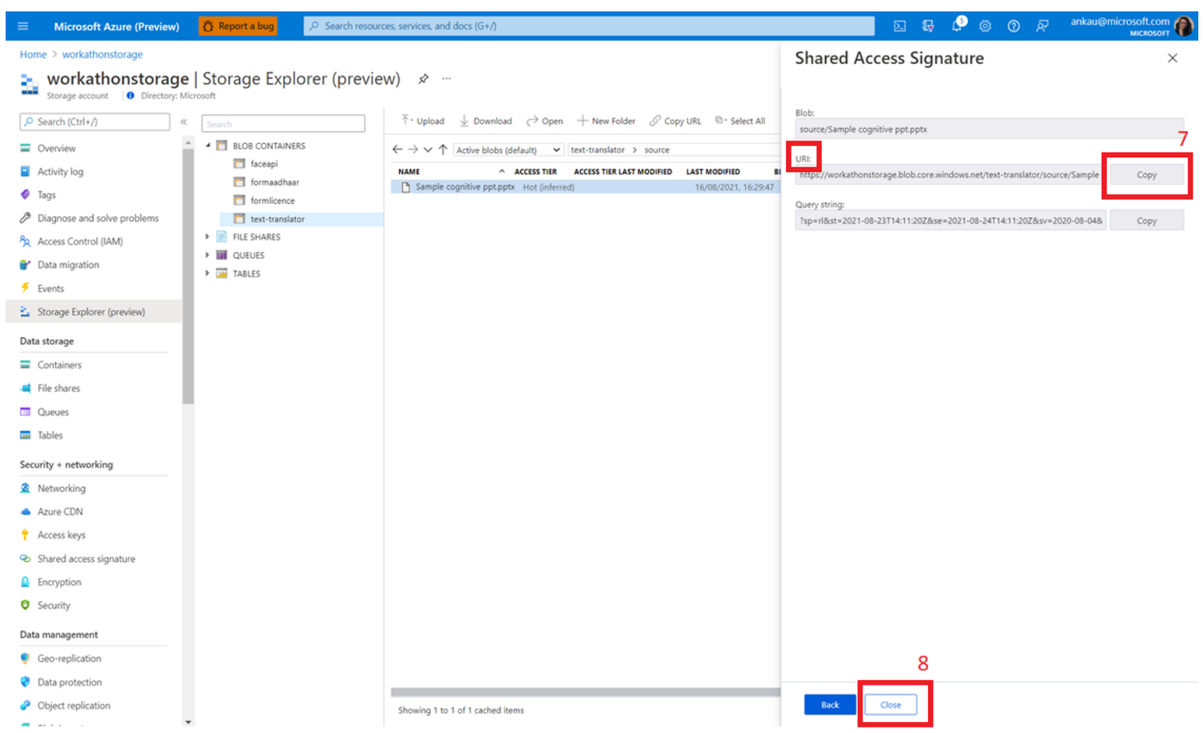
Make sure you are in the **text-translator** container, as shown in steps 1.
Select + New Folder (step 2)
Name the folder – **source**. [You can choose a different name, however, that will change the path to your images while making API calls from postman. Make sure to change the image paths accordingly]
Click Ok.

Once you click Ok, make sure to stay on the same screen and upload files to it.
As this is a virtual folder and will not be accessible if you leave it without adding any files. Once you add a file / folder, it will persist and you will be able to access the path.

Click Upload (step 6)
Select file icon and browse to the file you want to translate (step 7-9)
Click Upload Button (step 10)







Get SAS URI of the source file

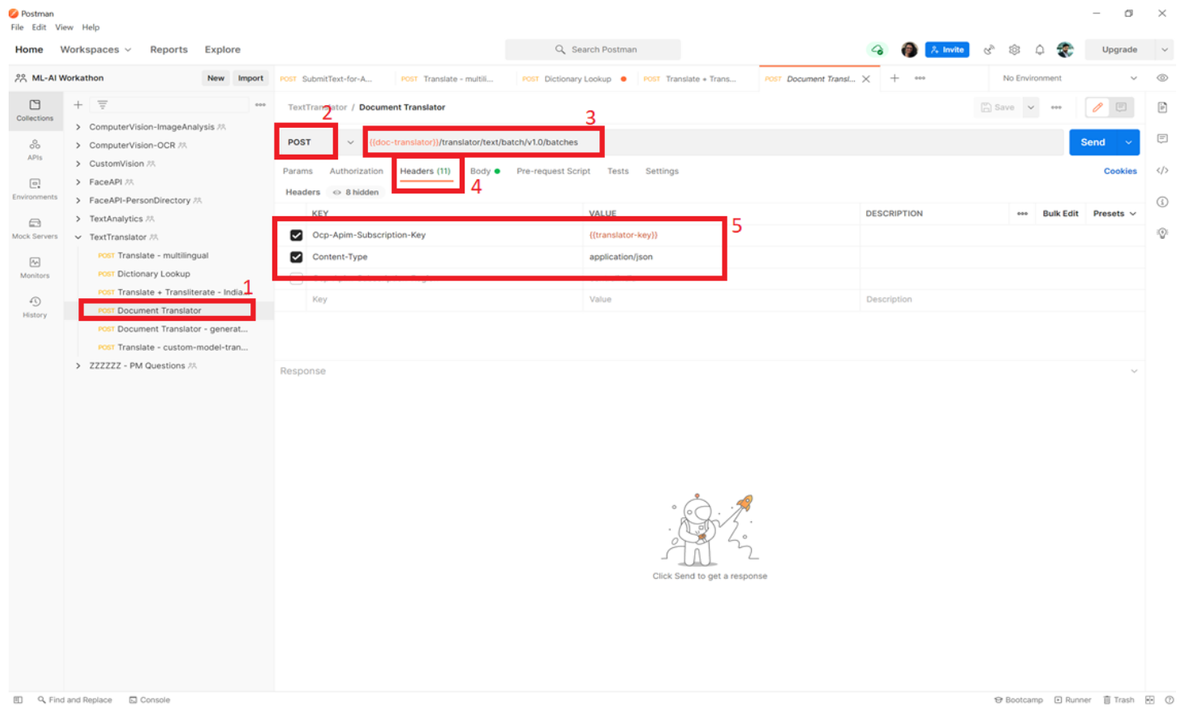
A shared access signature (SAS) provides secure delegated access to resources in your storage account. With a SAS, you have granular control over how a client can access your data. For example:

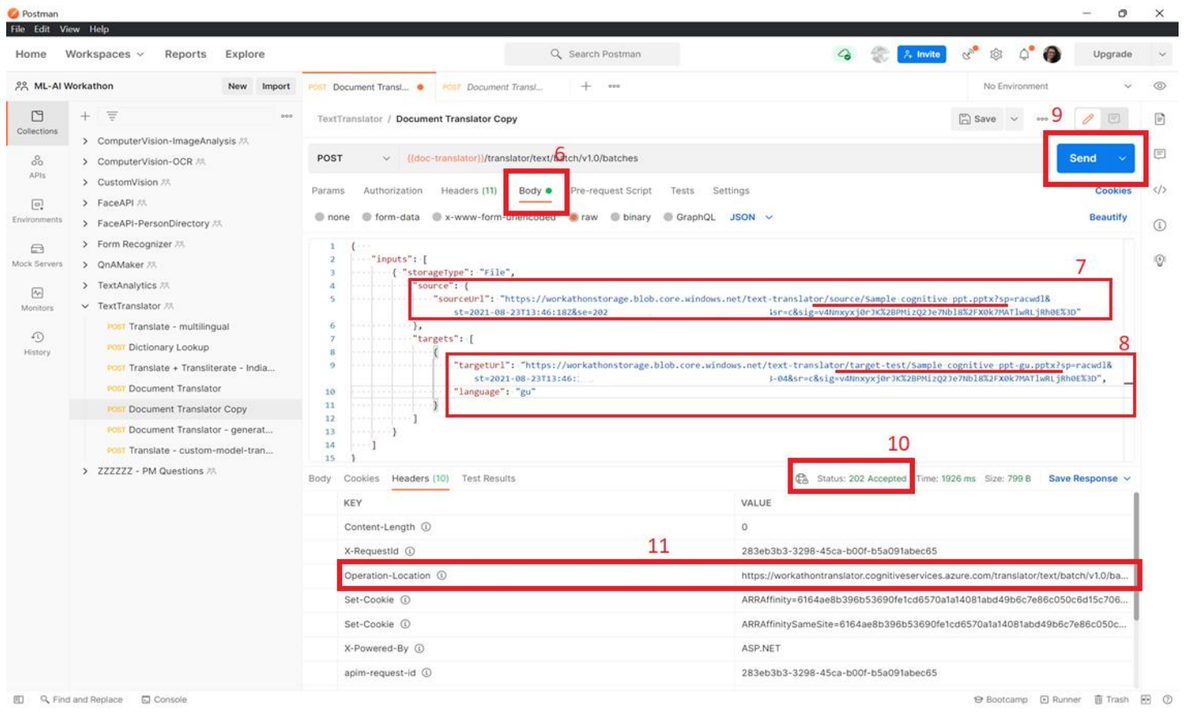
a. What resources the client may access
b. What permissions they have to those resources.
c. How long the SAS is valid

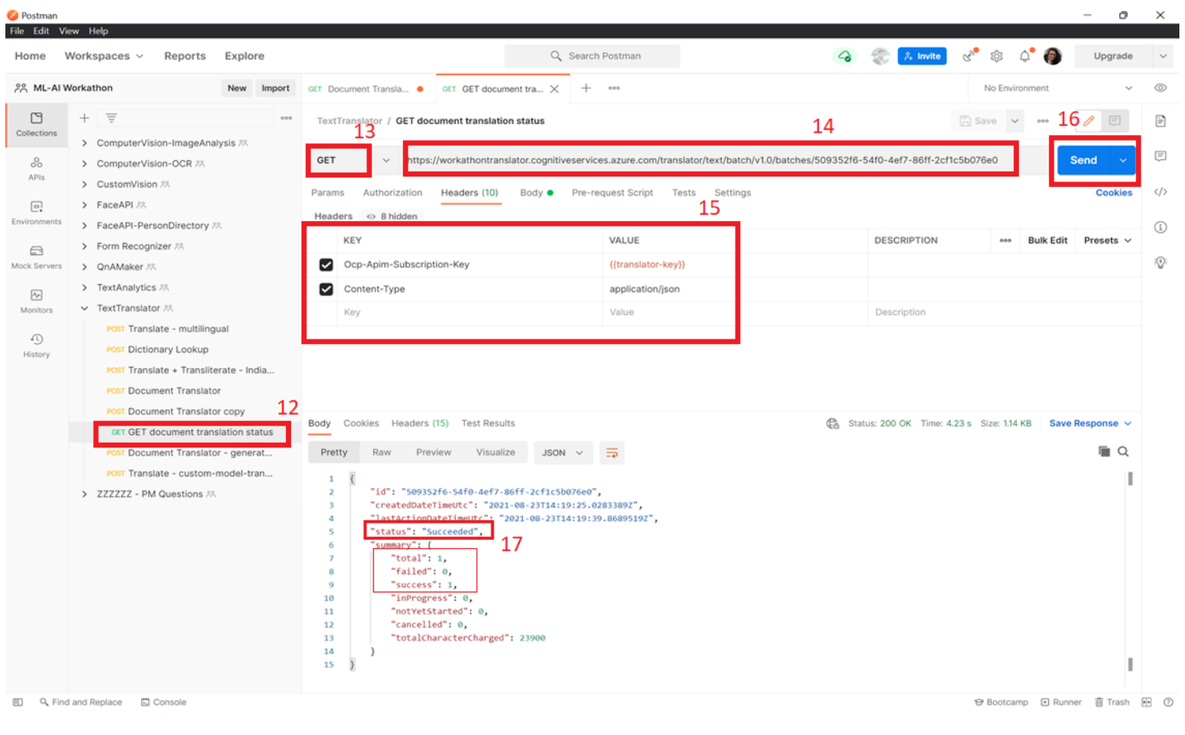
We require a SAS URI to grant access to the source file.

1. In the containers section, right click on file
2. Select Generate SAS
3. Set up the SAS Expiry (Make sure to set this up to a later date since you will have to generate a new SAS key in case the keys expire)
4. For Permissions select all**
5. Tick check box ‘Generate container level SAS’
6. Click Generate SAS URL & Token (step 6)
7. Copy the file SAS URL and paste it in a notepad. We will leverage this at a later step

** You should follow least access privilege however, for sake of simplicity, we have selected all for this Workshop.
This workshop is not considering security set up as this is designed to familiarise you with service capabilities.
We will be sharing the security best practices towards the end of the workshop.







<u>Document Translation API</u>

We will use this API to translate a Power Point document to Gujarati.

URL : {{doc-translator}}/translator/text/batch/v1.0/batches

Headers :
Ocp-Apim-Subscription-Key : {{key}}
Content-Type : application/json

Body :

```
{
  "inputs": [
    { "storageType": "File",
      "source": {
        "sourceUrl": " <<Paste the sas uri copied above within the quotes>>"
      },
      "targets": [
        {
          "targetUrl": " <<copy the sas uri here as well & replace source folder name with target folder name & file name >>",
          "language": "gu"
        }
      ]
    }
  ]
}
```

If you want to place the target file in a different container, fetch the uri for that container as well.
Make sure to match the uri to input with the screenshots in terms of folder & file name compatibility. (step 7,8)

<u>Significance of input & output</u>

1. {{endpoint}} , {{key}} : Values being picked from global variables

2. Ocp-Apim-Subscription-Key : This is the Azure Cognitive service key, that will authenticate the request.

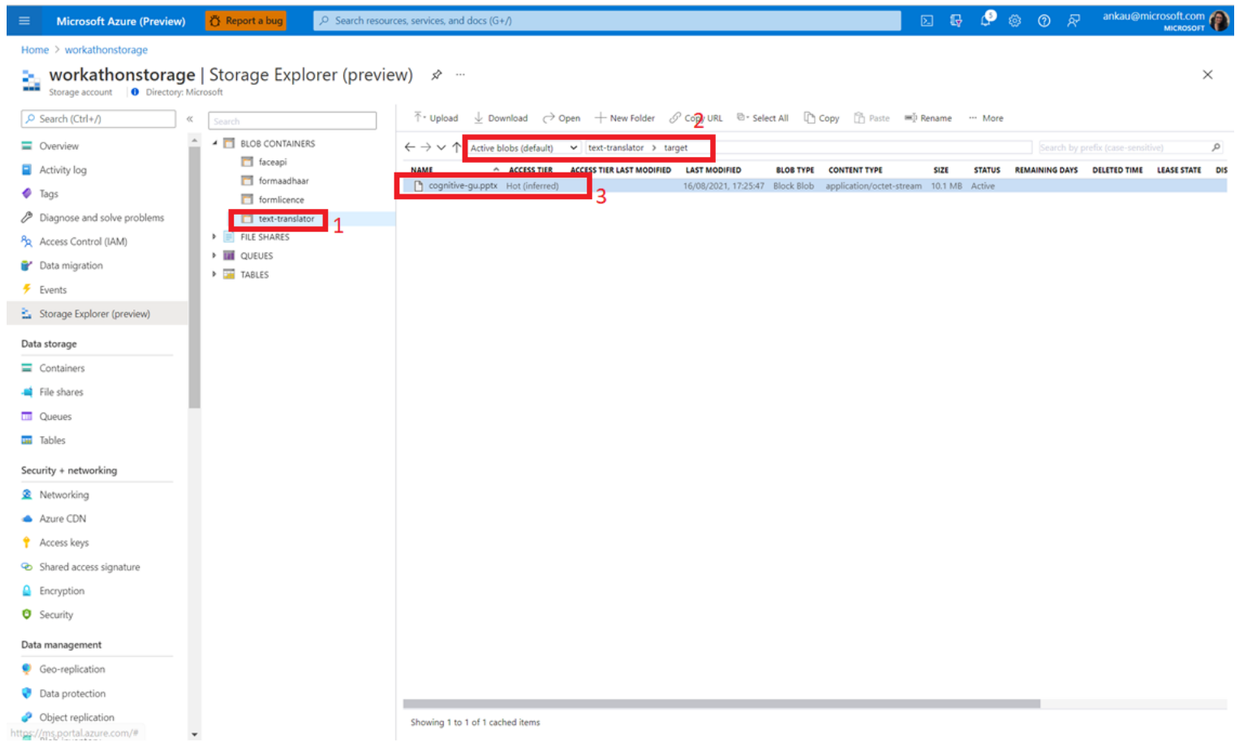
   Content-Type : This refers to the input type that you provide in the body, for eg application/json allows you to enter body text in JSON format. Change the content-type on the basis of input you provide.

You will see the translated file in the target location after a while, once complete document is translated.

Optionally, to check the status of the translation, you can option the Operation-location from Headers section in the results (step 11).

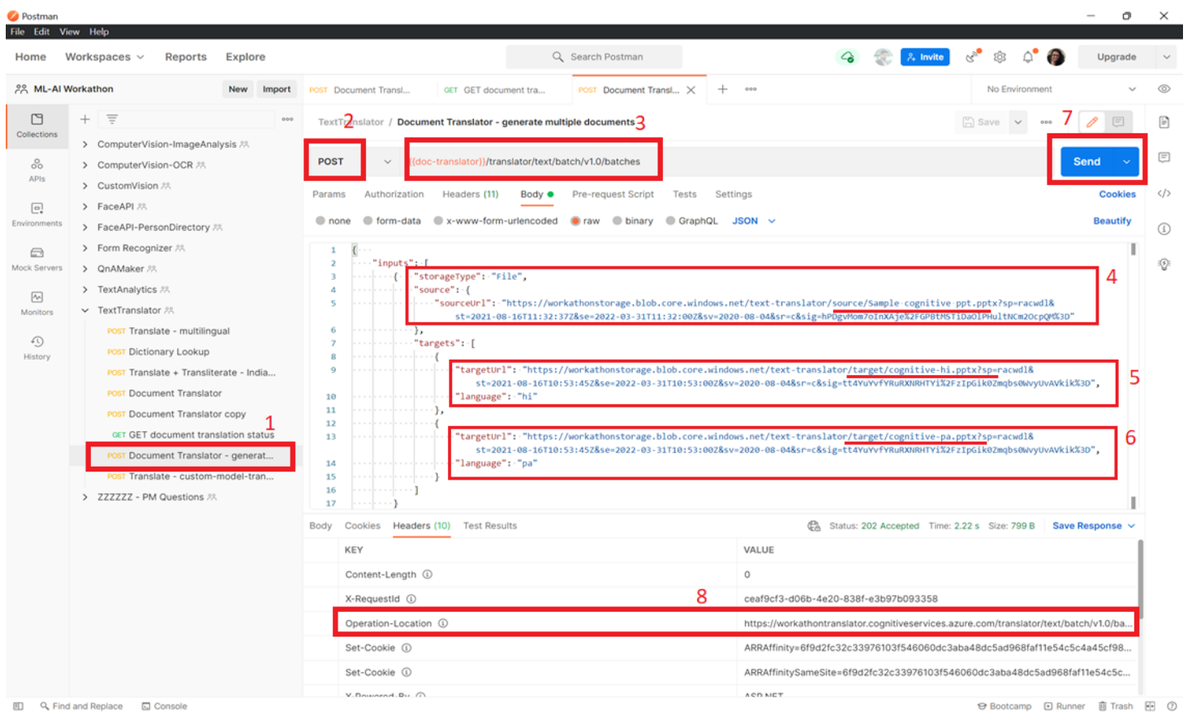
Make a GET call to the Operation-location with the headers same as previous call.
Observe the status. This will change to succeeded once the operation is completed.

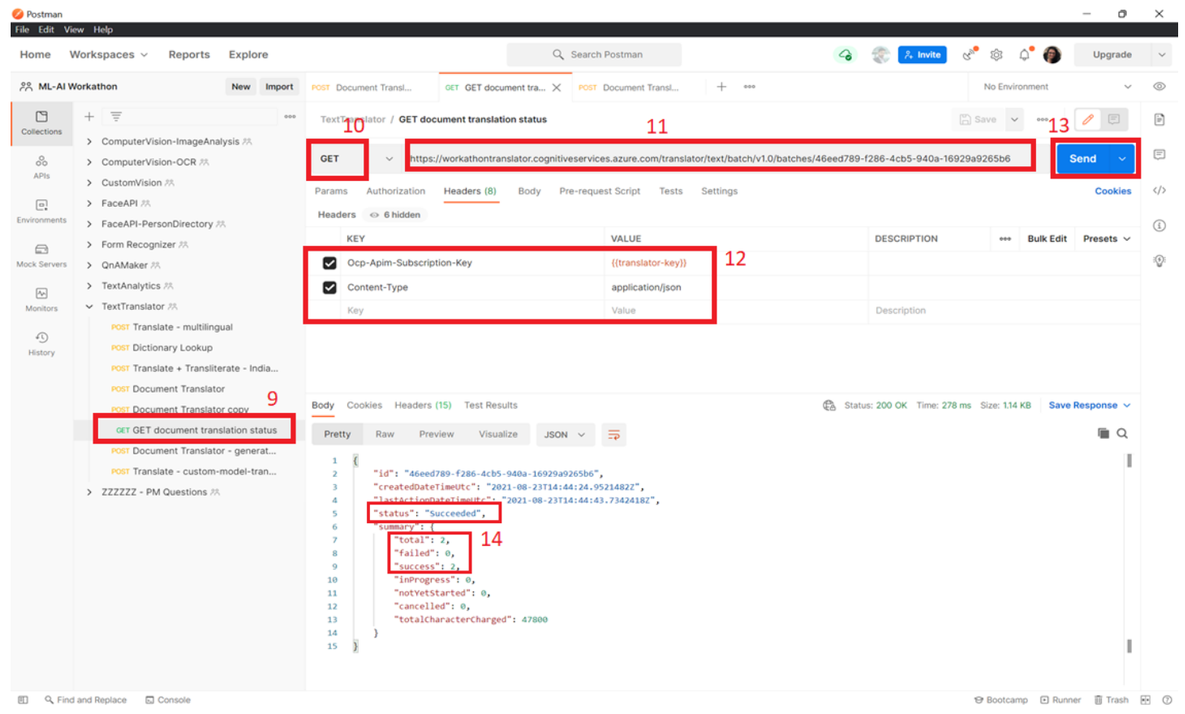
This will also give you status for number of files succeeded & failed.

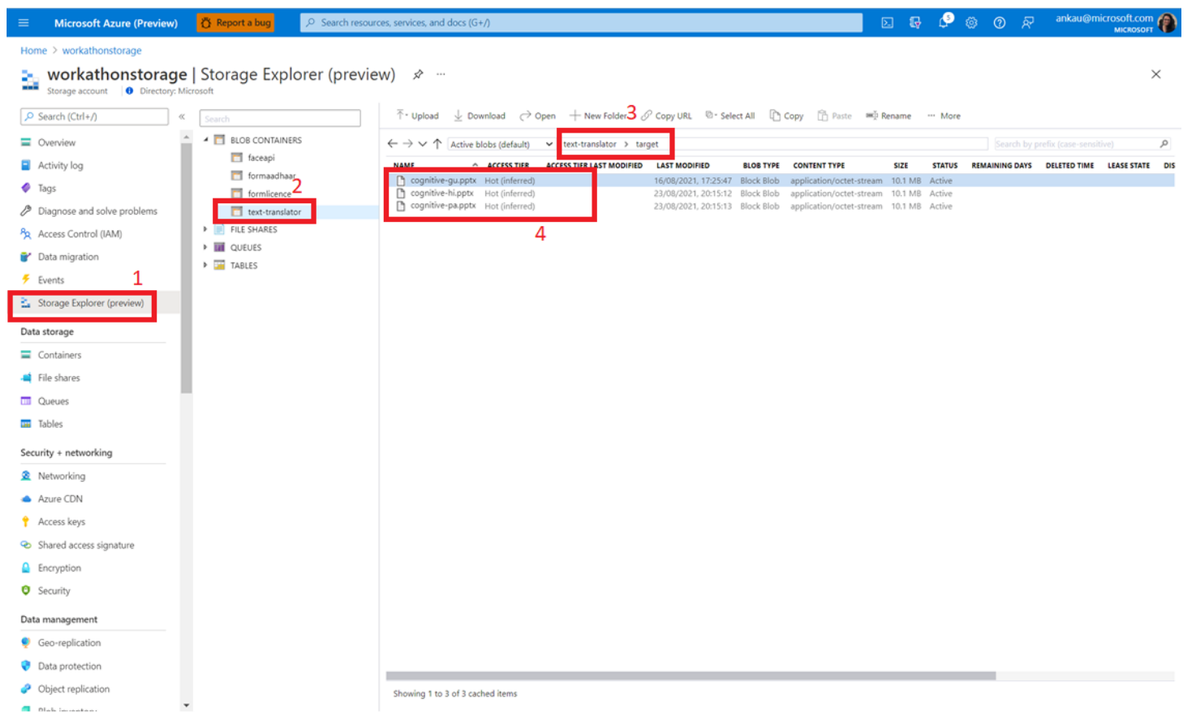


View the translated file

Navigate to the folder path you mentioned in Target URL to see the translated file. You can download this file to view the contents.







<u>To translate document into multiple languages at once</u>

We will now translate a Power Point document to multiple Indian languages.

URL : {{doc-translator}}/translator/text/batch/v1.0/batches

Headers :
Ocp-Apim-Subscription-Key : {{key}}
Content-Type : application/json

Body :

```
{
  "inputs": [
    { "storageType": "File",
      "source": {
        "sourceUrl": " Paste the sas uri copied above within the quotes"
      },
      "targets": [
        {
          "targetUrl": " copy the sas uri here as well & replace source folder name with target folder name & file name ",
          "language": "hi"
        },
        {
          "targetUrl": " copy the sas uri here as well & replace source folder name with target folder name & file name ",
          "language": "pa"
        }
      ]
    }
  ]
}
```

If you want to place the target file in a different container, fetch the uri for that container as well.
Make sure to match the uri to input with the screenshots in terms of folder & file name compatibility. (step 4,5,6)

<u>Significance of input & output</u>

1. {{endpoint}} , {{key}} : Values being picked from global variables

2. Ocp-Apim-Subscription-Key : This is the Azure Cognitive service key, that will authenticate the request.

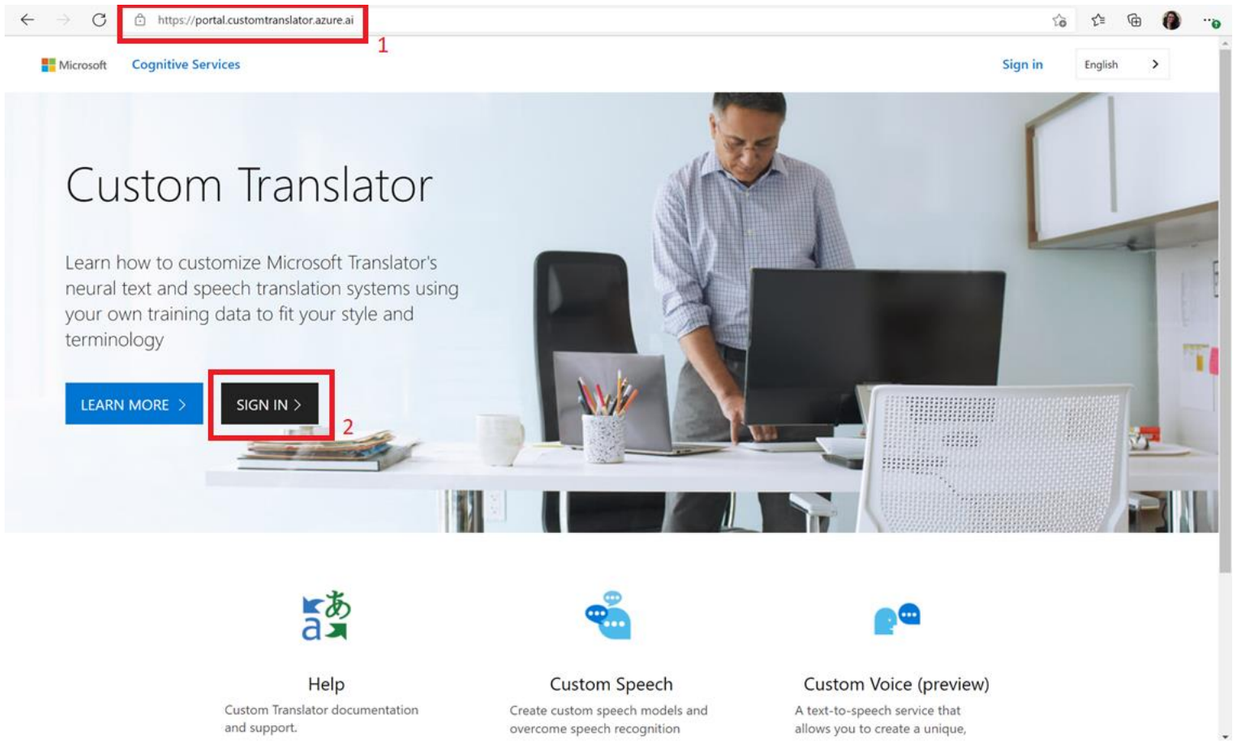
   Content-Type : This refers to the input type that you provide in the body, for eg application/json allows you to enter body text in JSON format. Change the content-type on the basis of input you provide.

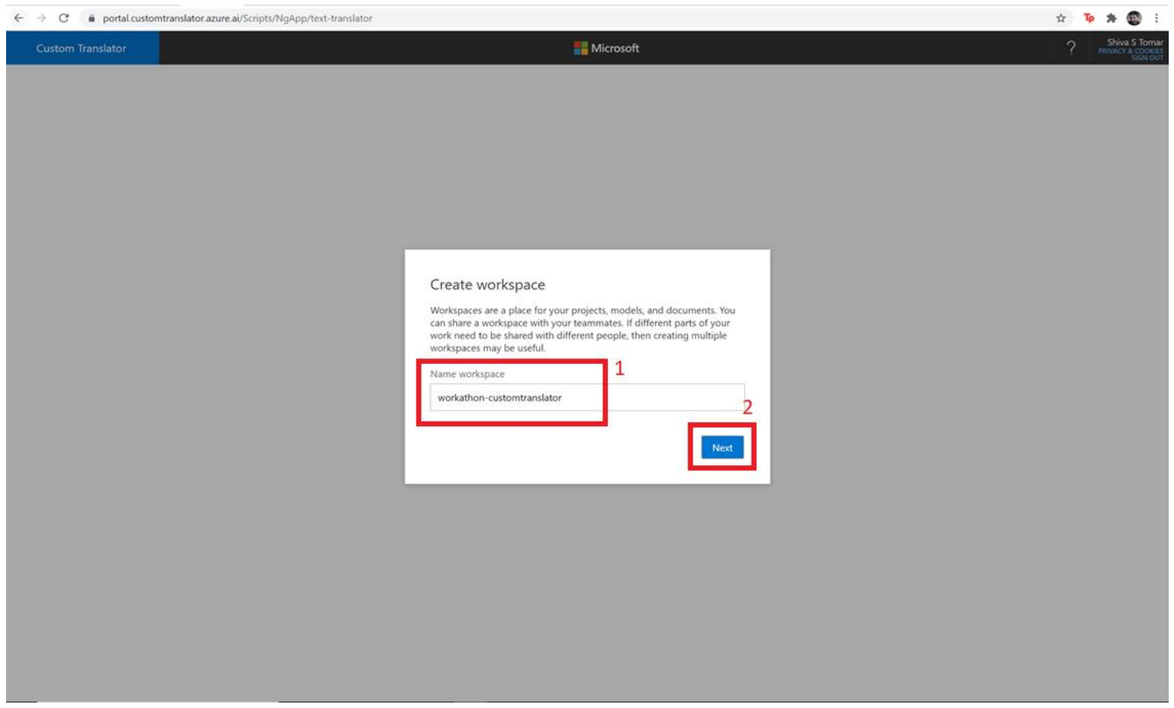
You will see the translated file in the target location after a while once complete document is translated.

Optionally, to check the status of the translation, you can option the Operation-location from Headers section in the results (step 11).

Make a GET call to the Operation-location with the headers same as previous call.
Observe the status. This will change to succeeded once the operation is completed.
This will also give you status for number of files succeeded & failed.

Navigate to the folder path you mentioned in Target URL to see the translated file. You can download this file to view the contents.

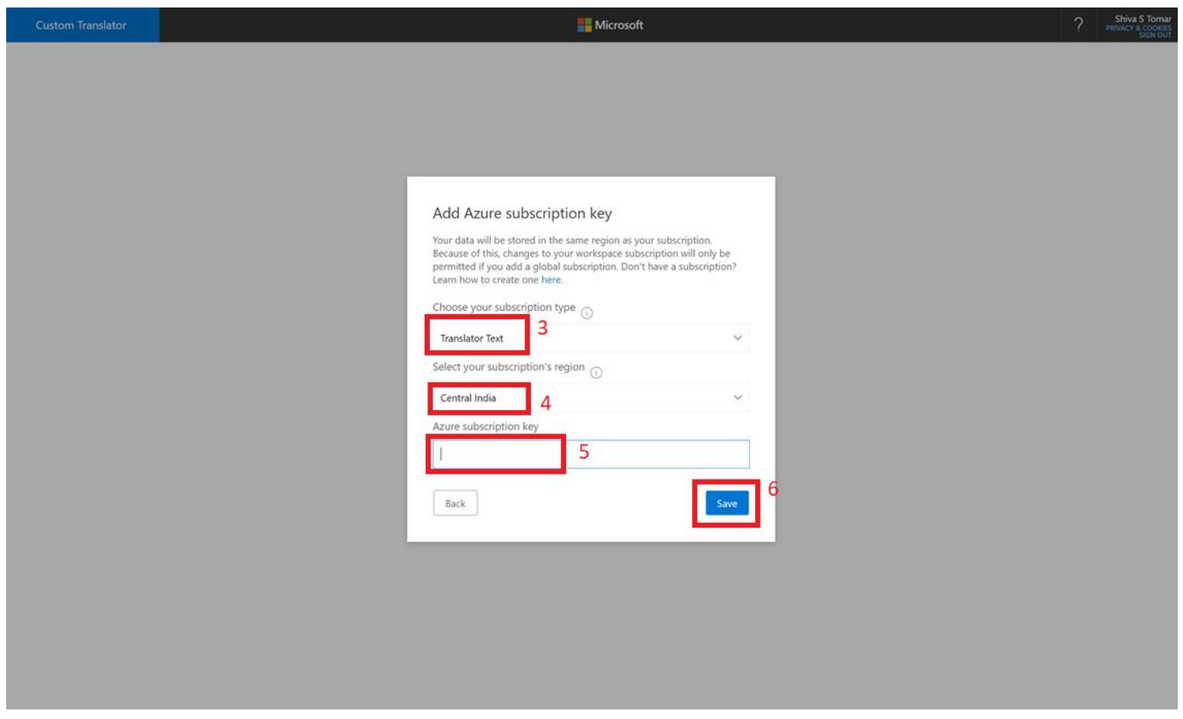
CUSTOM TRANSLATOR	
	Go to Custom Translator Portal. We will be creating a custom model for English to Hindi translation for e-commerce domain, using Phrase Dictionary document type.



Create workspace & set up account

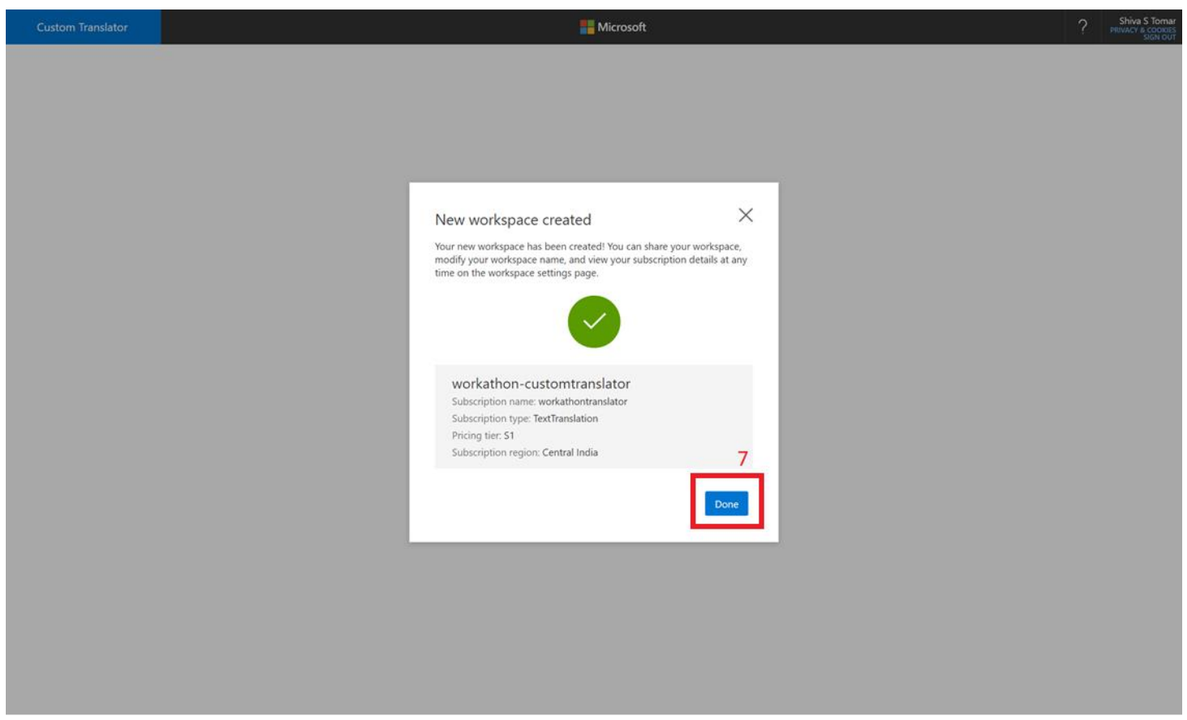
Create a workspace & select next. (step 1,2)

Workspace is like a logical container to hold & manage projects & documents for custom model you will build.

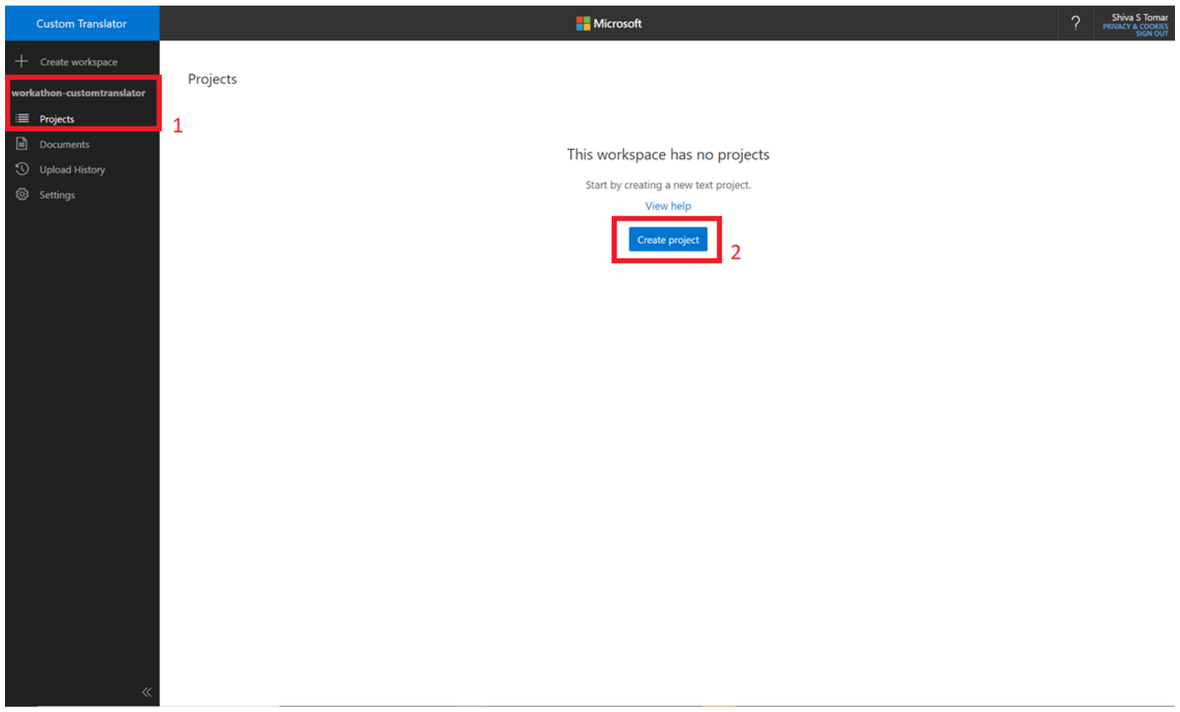


For steps 3 through 5, select the fields as shown and paste the key for the translator service we created in the beginning of the workshop.

Click Save button (step 6)

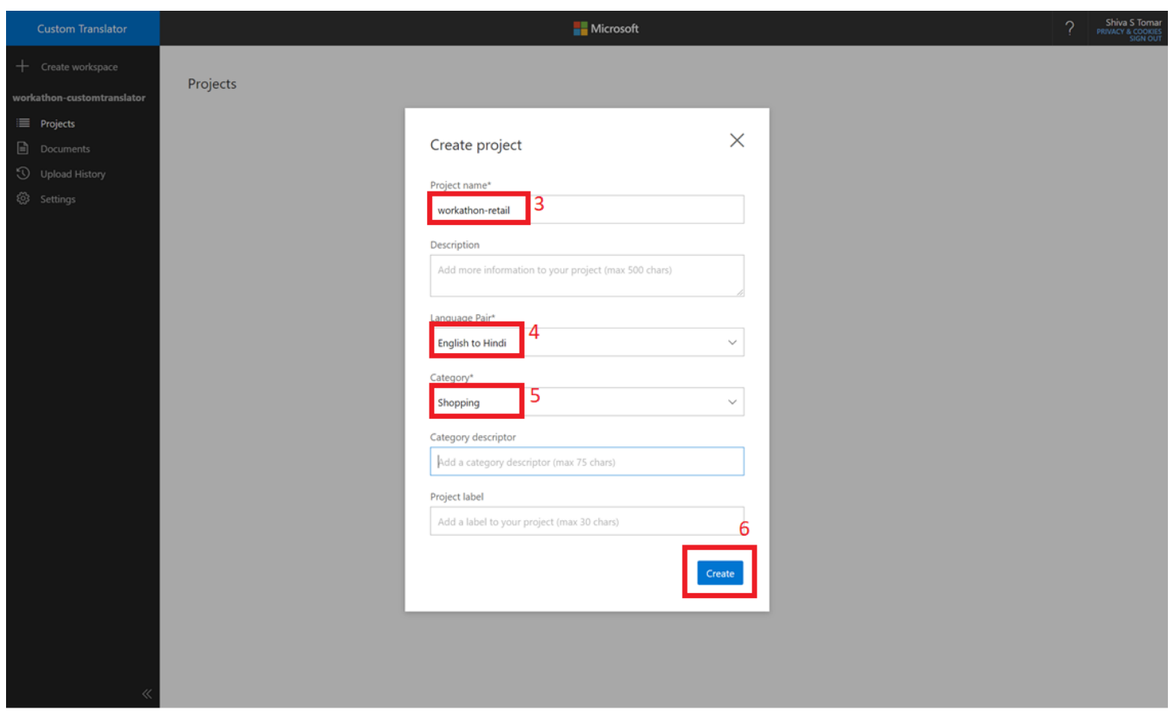


Verify the details and click Done (step 7)



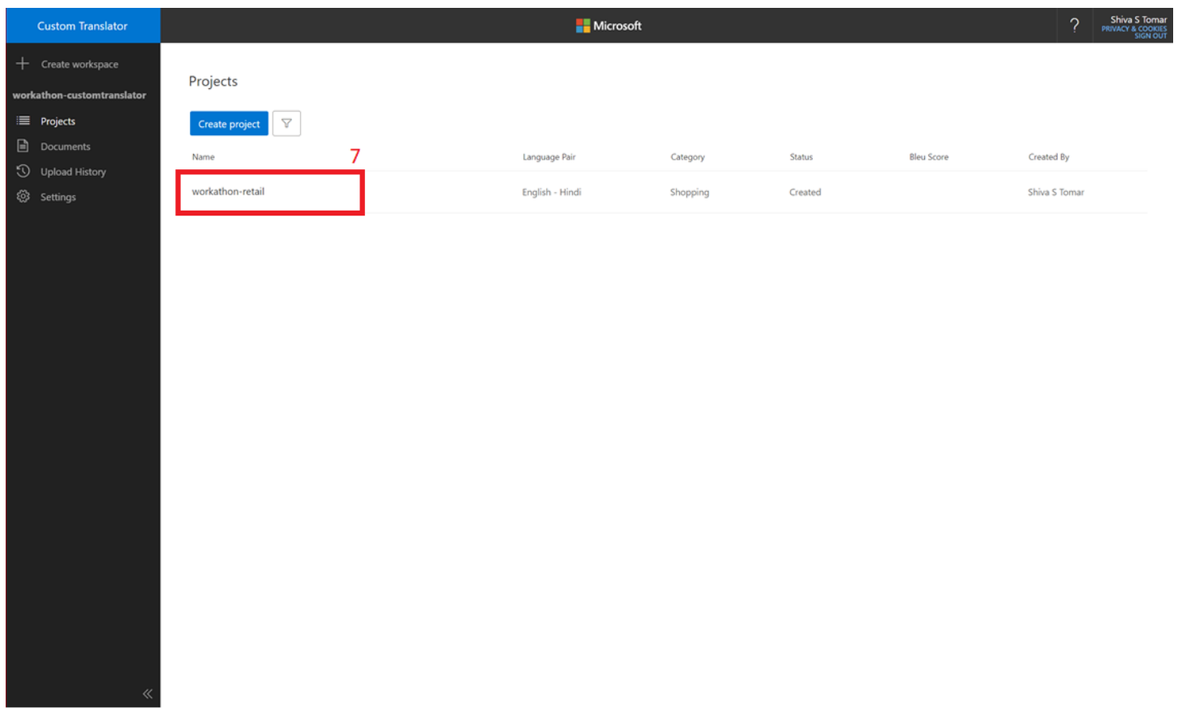
Create Project

Go to Projects & select Create Project (step 1,2)

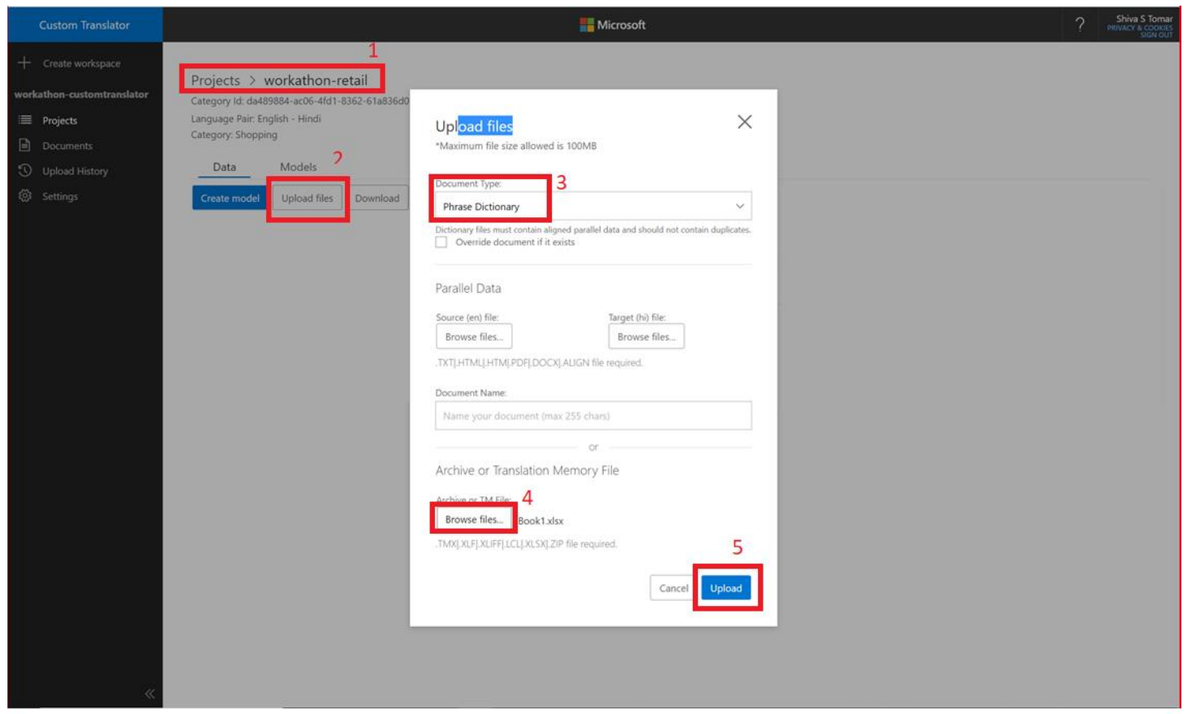


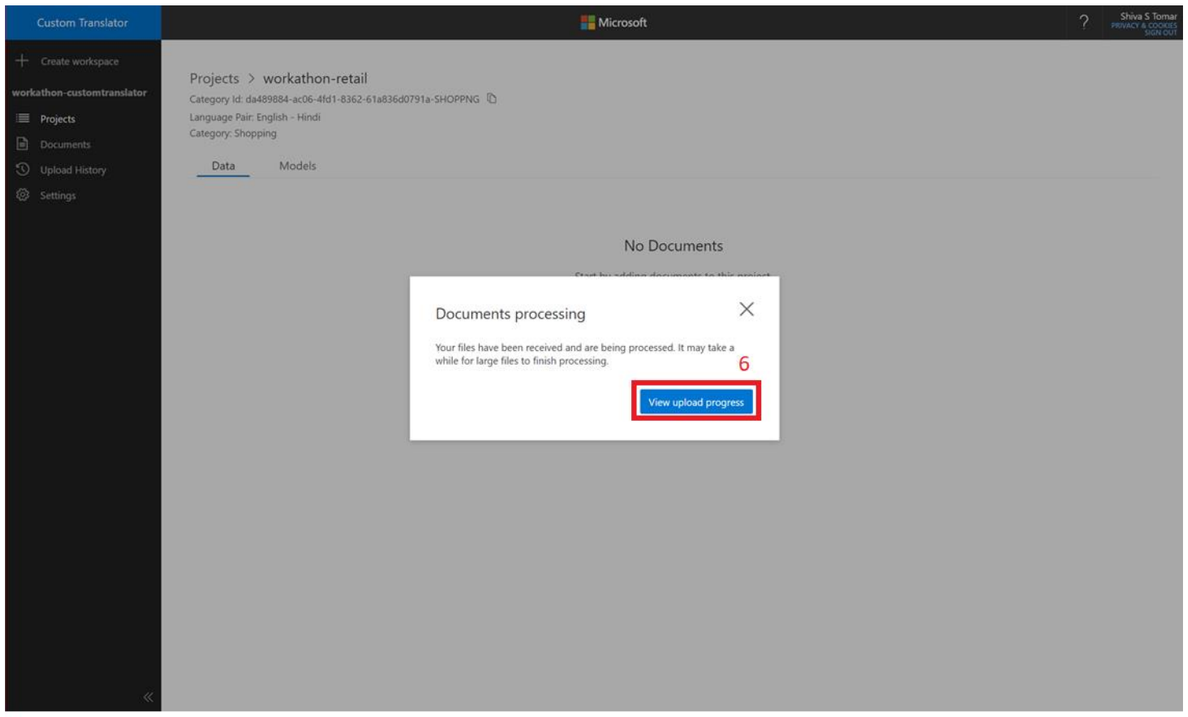
For steps 3 through 5, enter the details pertaining to the project as shown.
We will be creating a custom model for English to Hindi translation.

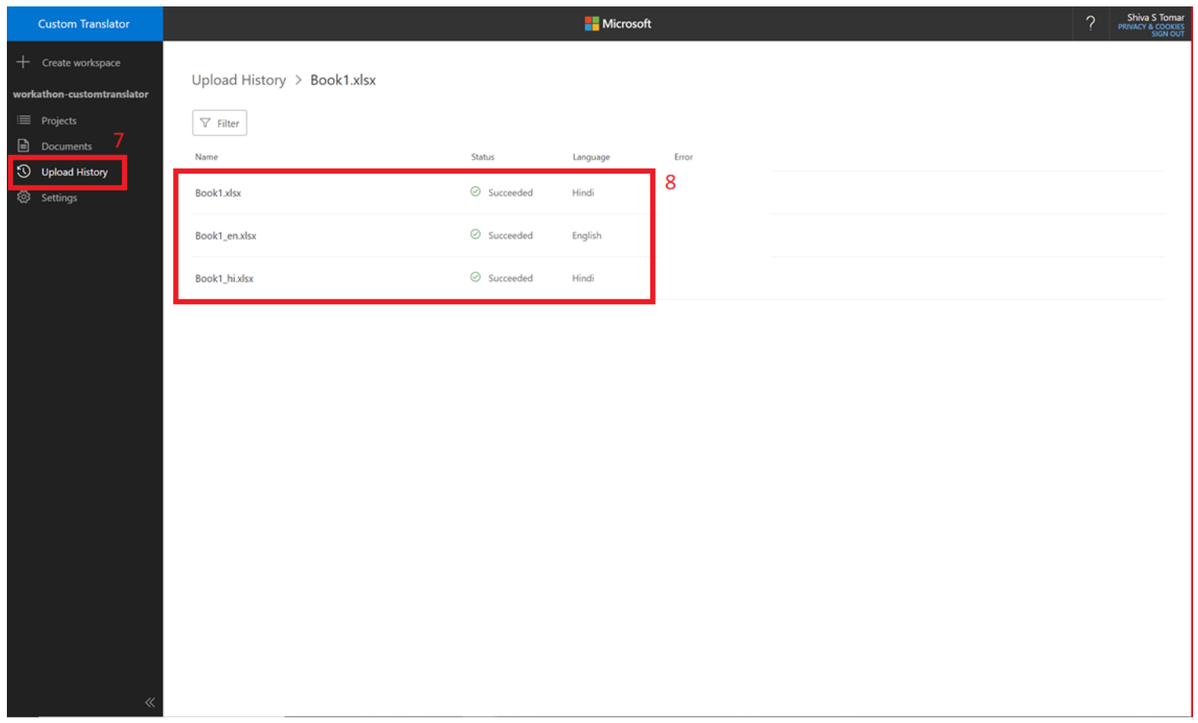
Click Create (step 6)



Once created, click on the Project Name (step 7)







<u>Upload training document</u>

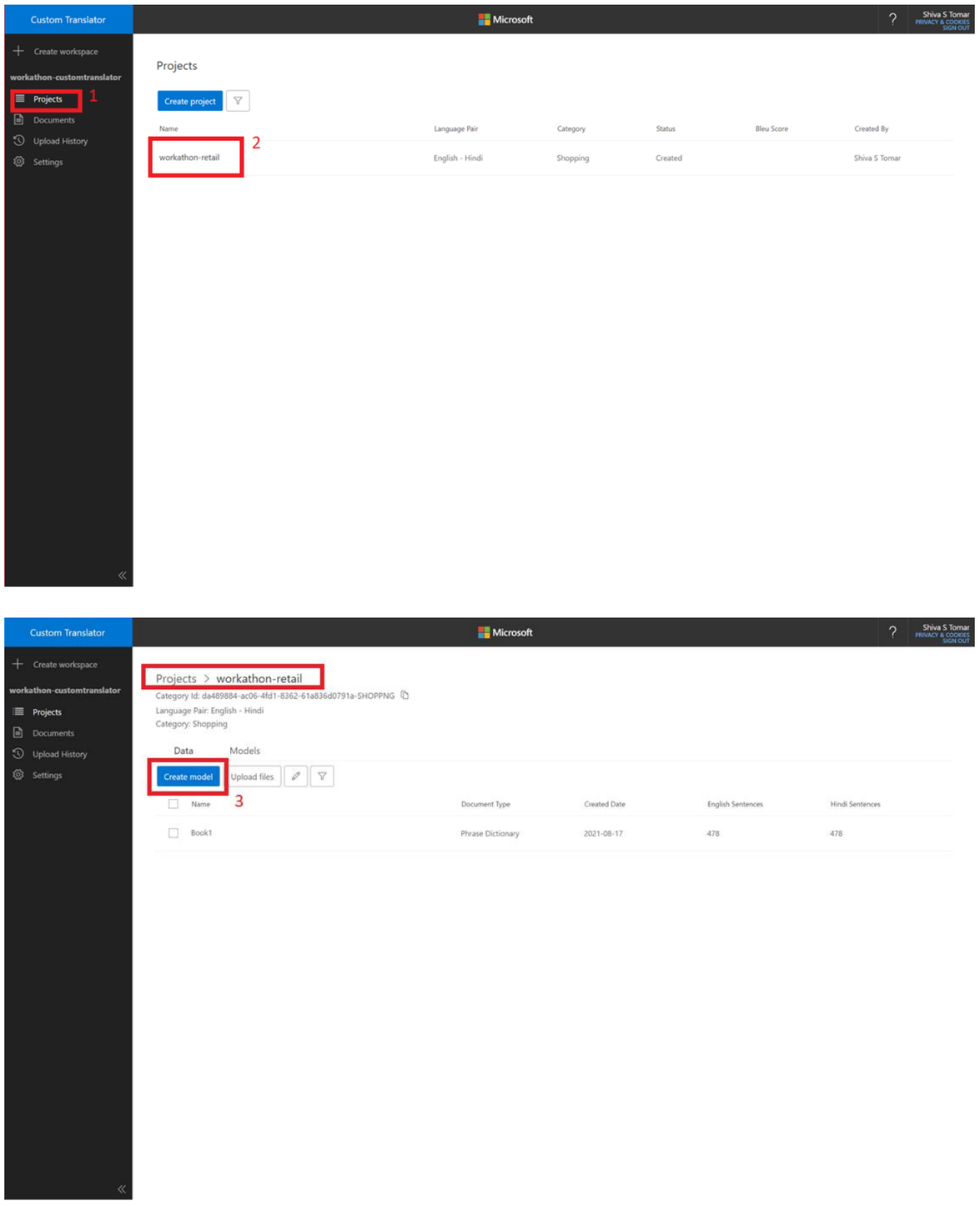
While in the project, click Upload Files (step 2)
Select Phrase Dictionary from Document Type dropdown.

Browse files as shown in step 4 and select the mapped Dictionary file you downloaded in the beginning of the workshop.

Click Upload (step 5)

Click View upload progress (step 6)

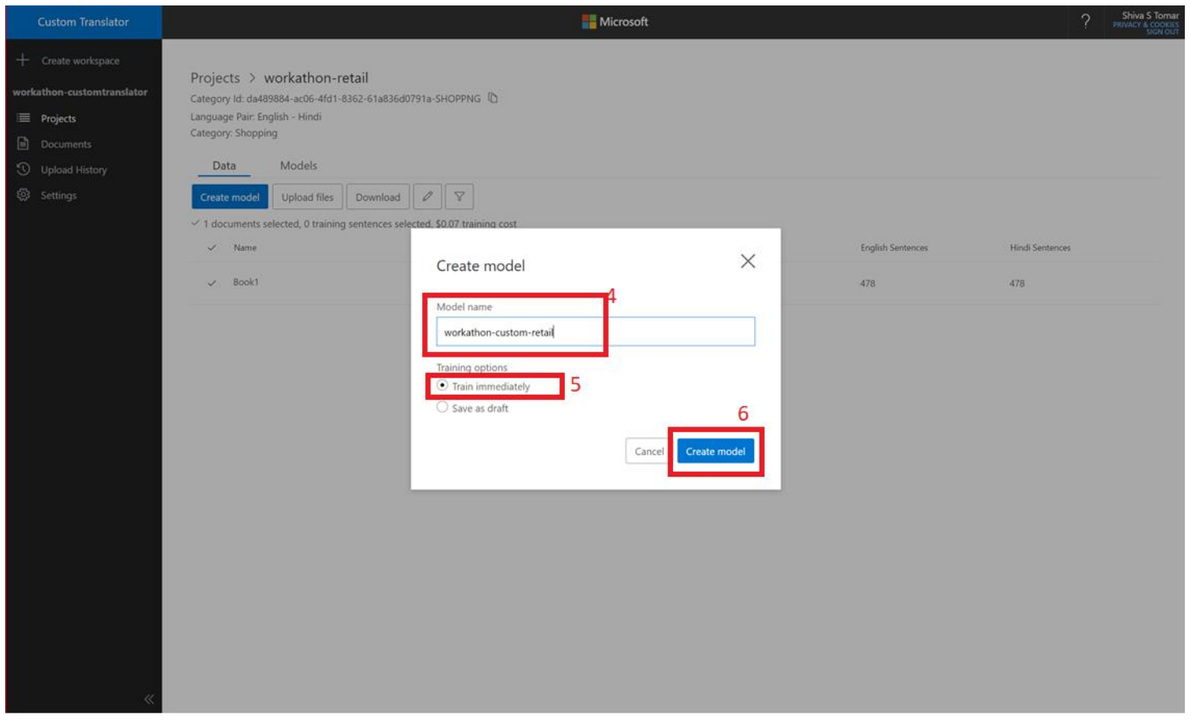
Once uploaded, you can view the documents in Upload History, as shown in steps 7 & 8.



Train Project

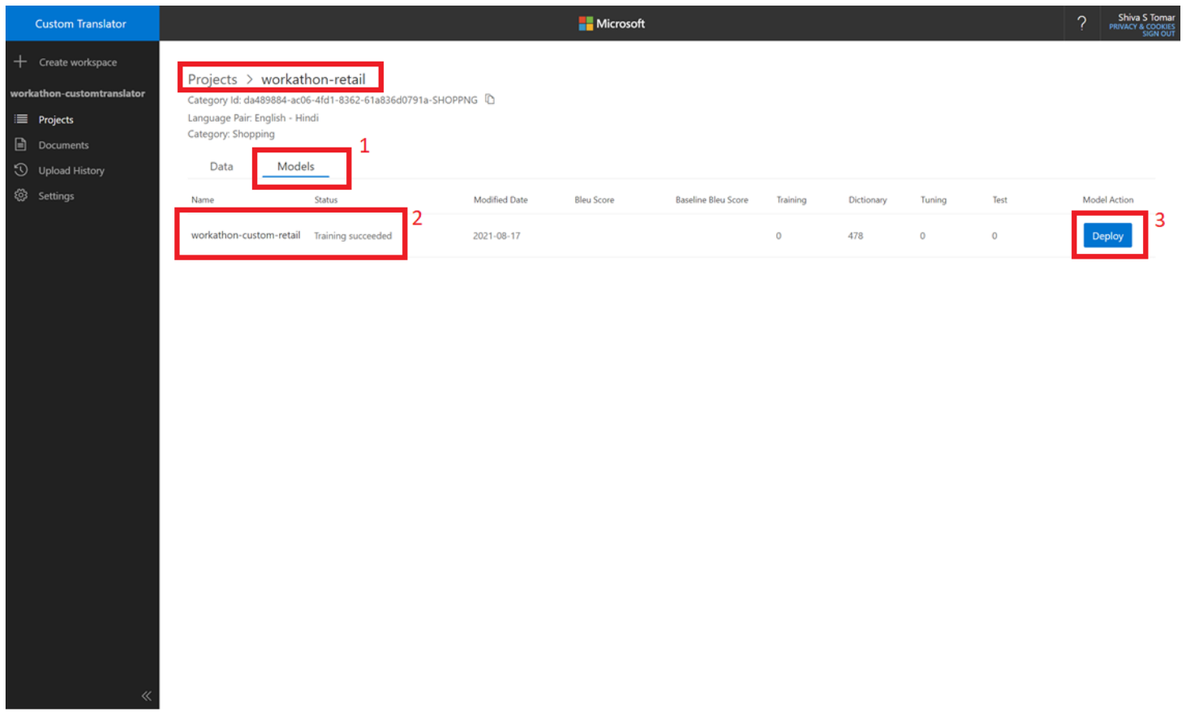
Come back to the Project home page and select the project (step 1,2)

Click create model (step 3)



Provide the Model name and select train immediately. (Step 4,5)

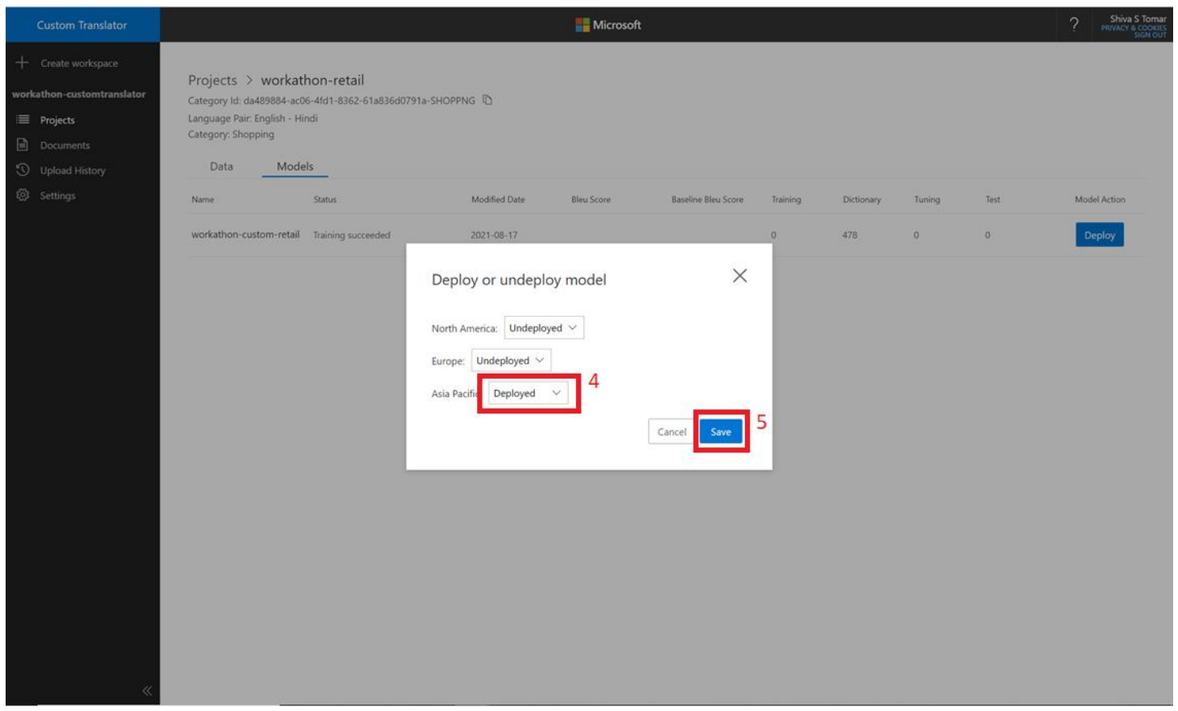
Click Create Model (step 6)



Deploy Custom Model

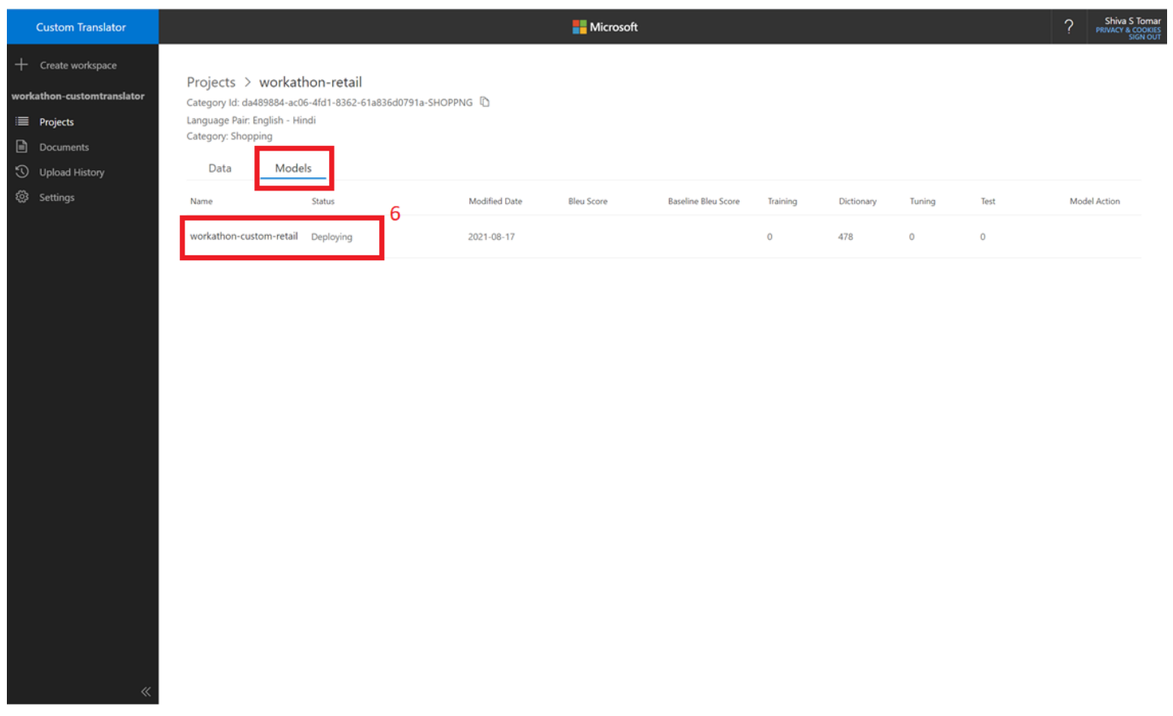
Once training is trained, you can see the status as shown in step 2.

Now, let's deploy the model to be able to leverage it to make real time translation calls. Click Deploy (Step 3)

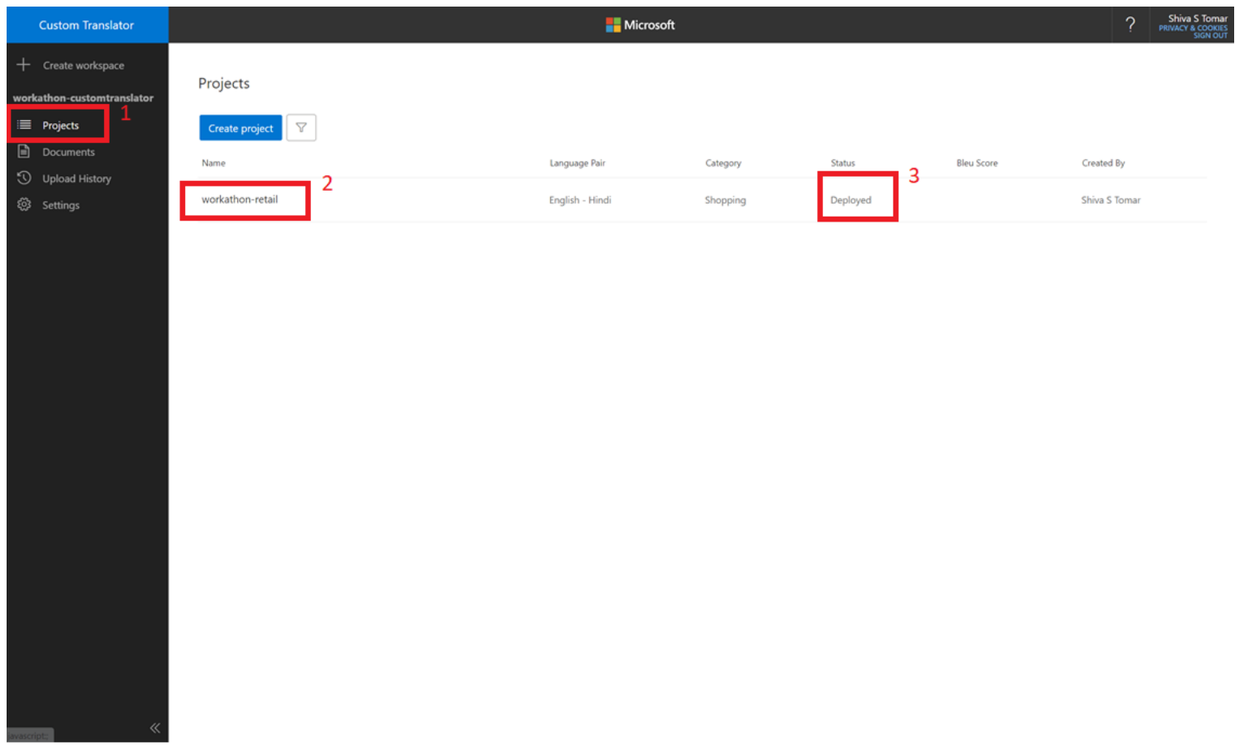


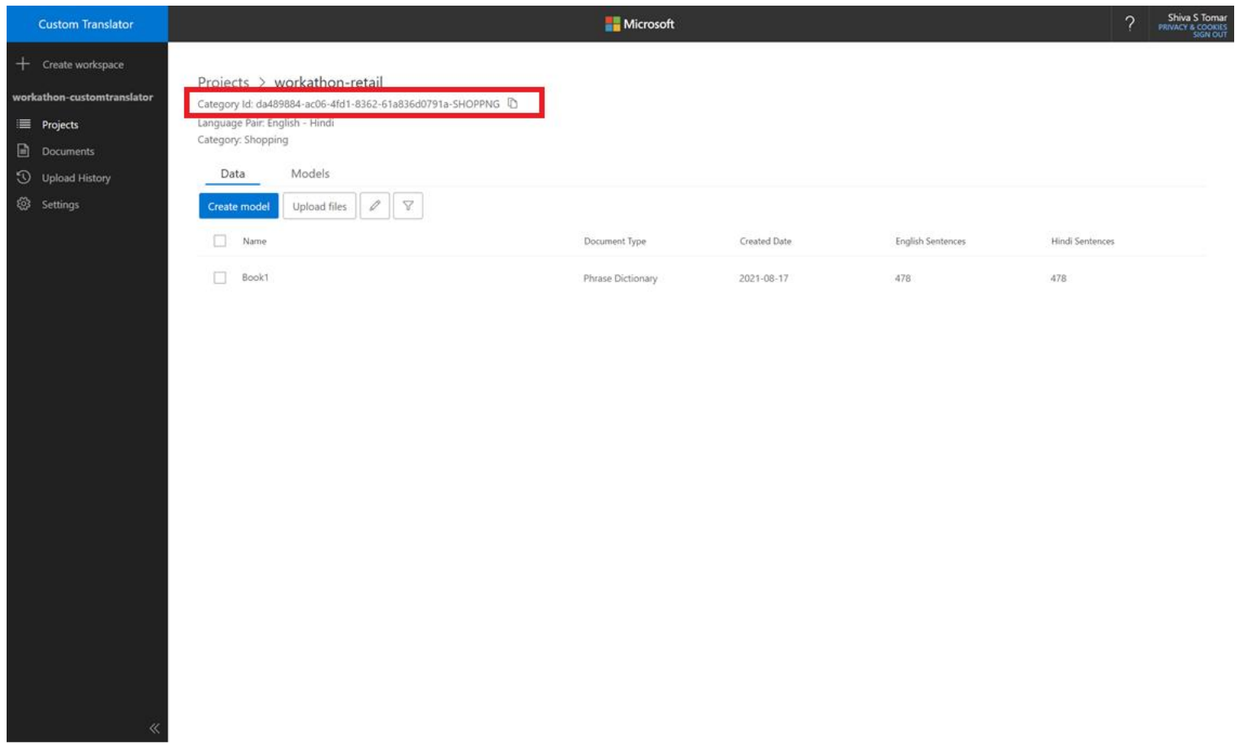
Change the drop downs to Deployed for Regions where you want to deploy your model in (step 4)

Select save (step 5)



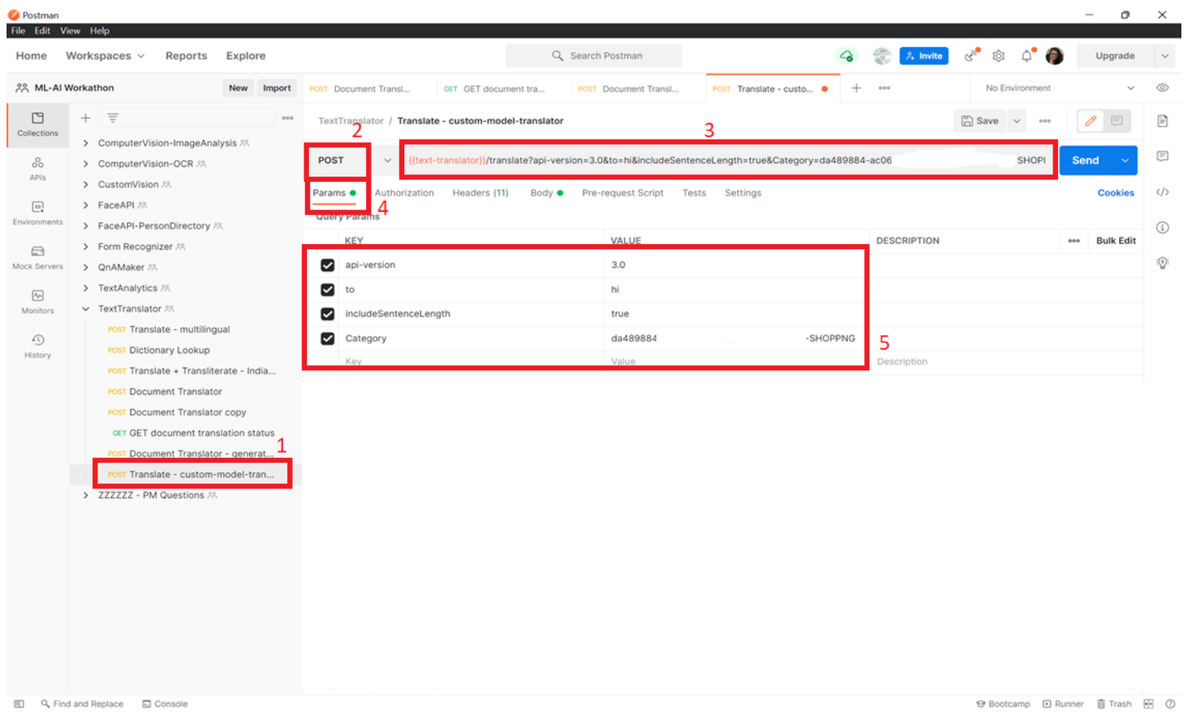
Go to the Models tab and view the status. This will be set to deploying for a while till model deployment isn't completed (Step 6)

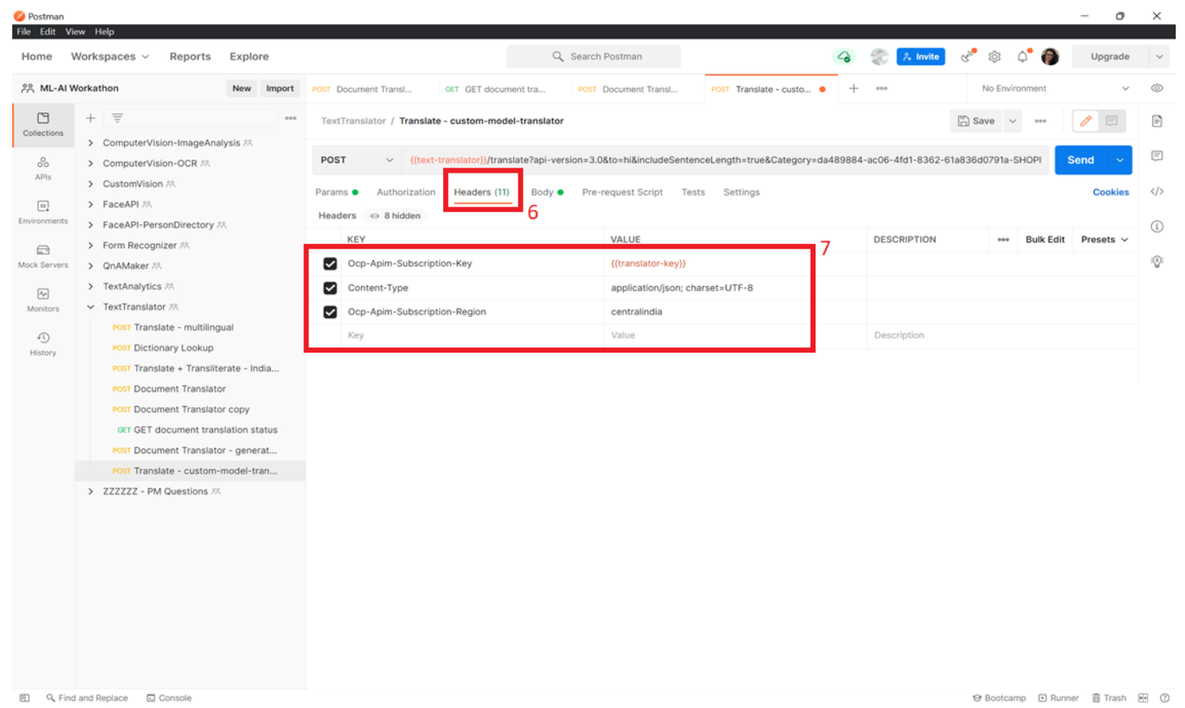


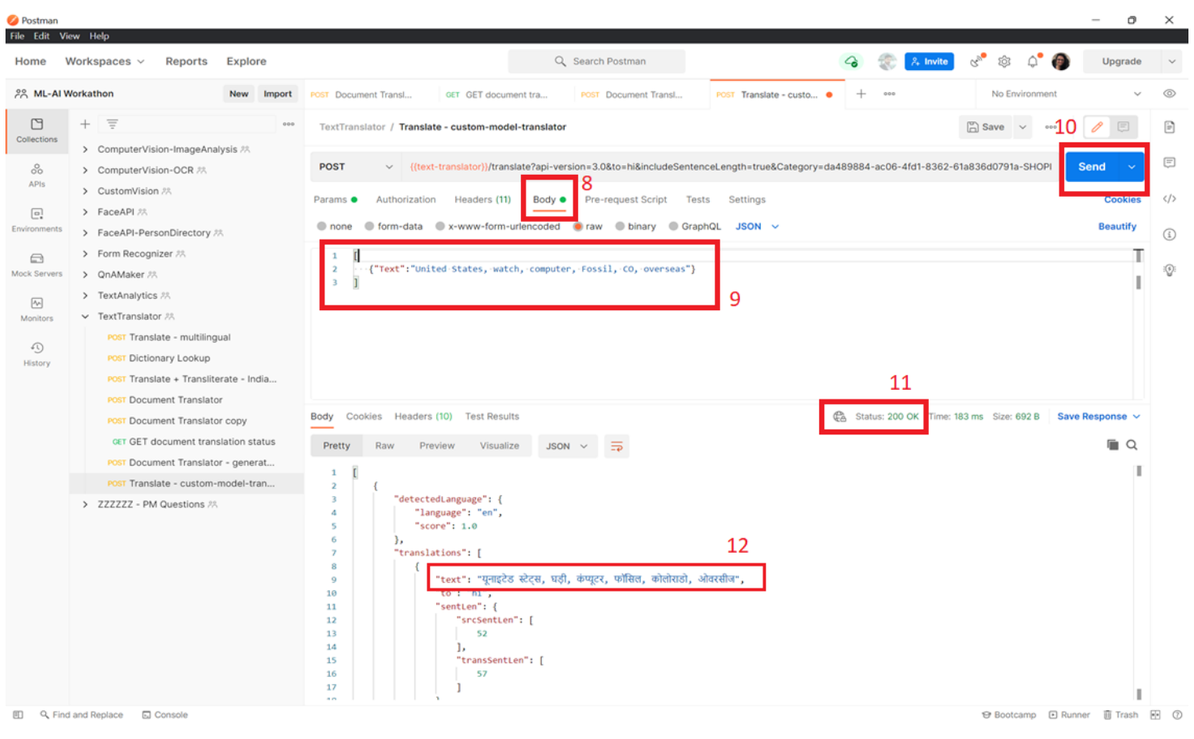


Once the Model is deployed, you can view the change in status as shown in step 3.

Go to the project page and copy the Category ID as shown in the below image.
We will require this to make real time API calls to our custom model.







<u>To translate text using Custom Model</u>

We will now translate text using the Custom Model just created. Note that custom models just work with Text Translation API & not Document Translation API.

URL : {{text-translator}}/translate

Params :
api-version=3.0
to=hi
includeSentenceLength=true
Category= *<<paste the CategoryID we copied above>>*

Once you add the params, these will be appended to the URL & you will see a URL similar to the one in the screenshot.

Headers :
Ocp-Apim-Subscription-Key : {{key}}
Content-Type : application/json
Ocp-Apim-Subscription-Region : centralindia (change this if you provisioned your resource in a different region)

Body :

```
[
  {"Text":"United States, watch, computer, Fossil, CO, overseas"}
]
```

<u>Significance of input & output</u>

1. {{endpoint}} , {{key}} : Values being picked from global variables

2. Ocp-Apim-Subscription-Key : This is the Azure Cognitive service key, that will authenticate the request.

   Content-Type : This refers to the input type that you provide in the body, for eg application/json allows you to enter body text in JSON format. Change the content-type on the basis of input you provide.

   Ocp-Apim-Subscription-Region : region where your resource is deployed.

Observe the obtained translations.

Now, try executing the call by removing the category parameter & executing the pre-built translator model and observe the difference in outcome.

Homework

1. Try the pre-built models for other languages that you're familiar with.
2. Try creating the custom model using other document types.

Additional recommended resources

Service Name	Category	Links
Translator Service	Programming Language	C#, Java, Python, Node.js, Go, PHP, Ruby
	Tiers	Free, S1, S2, S3, S4, C2, C3, C4, D3 | (Difference between Tiers)
	Pricing	https://azure.microsoft.com/en-in/pricing/details/cognitive-services/translator/
	Limits	https://docs.microsoft.com/en-us/azure/cognitive-services/translator/request-limits
	Language Support	https://docs.microsoft.com/en-us/azure/cognitive-services/translator/language-support
	Sample Apps	Sample Applications
	Regional Availability & Support	https://azure.microsoft.com/en-us/global-infrastructure/services/?products=cognitive-services&regions=all
	SLAs for Cognitive Services	https://azure.microsoft.com/en-in/support/legal/sla/cognitive-services/v1_1/
	Compliance & Certificates	https://azure.microsoft.com/en-us/support/legal/cognitive-services-compliance-and-privacy/
	Cognitive Services Updates	https://azure.microsoft.com/en-us/updates/?product=cognitive-services

Security best practices

1. Azure Cognitive Services security
2. Networking
3. Authentication
4. Key Management
5. Data loss prevention
6. Azure security baseline
7. Regulatory Compliance controls

Responsible AI being a part of best practices, we encourage you to read this.

Translator Documentation

Text Translator APIs

Custom Translator APIs