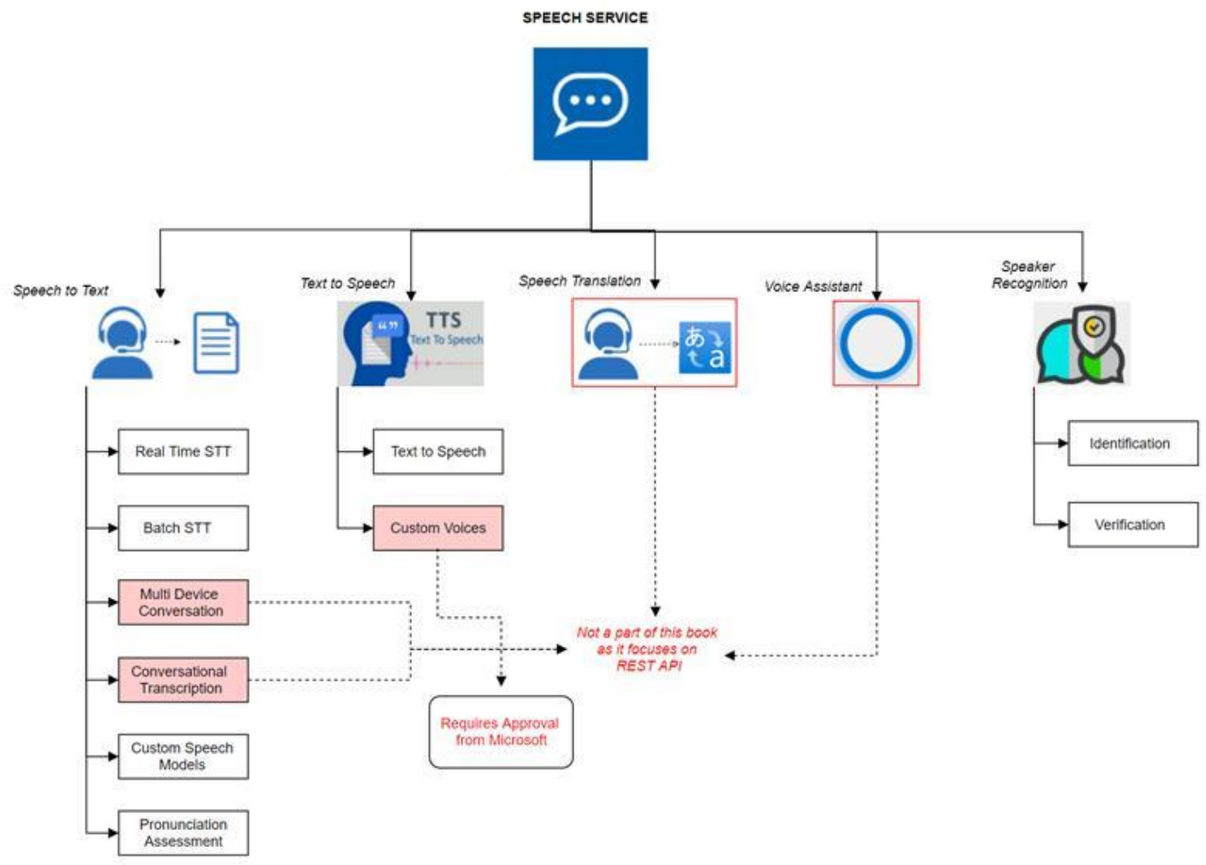
Document Name	HOL – Azure Speech Service
Author	Shiva S Tomar & Anupreet Kaur
Reviewer	
Executive Summary	Azure Cognitive APIs enable the developers of all skill levels to add human intelligence in their applications. The services are designed for developers interested in pursuing DS/AI/ML skills and 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 Speech Service. You will be able to explore each functionality offered by the service through the GUI & API 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 and Speech Portal. 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 Speech Service

Azure's Speech service allows users to add speech capabilities such as Speech-to-text transcription, Text-to-speech translation, Speaker recognition, Speech translation and Voice assistants in their applications.

Diagram: Functional Architecture



Azure Speech service can be broadly classified into the following features -

1. Speech-to-text : This functionality generates transcription for any given audio files. Speech to text capability is available in more than 85 languages and variants and can further be classified into the following features -
    a. Real time STT : This functionality transcribes audio streams or local files to text in real time that your applications, tools, or devices can consume or display. This takes up audio files in chunks of 60s or less and is a synchronous API call.
    b. Batch STT : This functionality enables asynchronous speech-to-text transcription of large volumes of speech audio data stored in Azure Blob Storage. Additionally, this also allows for diarization and sentiment-analysis of the spoken audio.
    c. Multi-device conversation : This functionality enables you to connect multiple devices or clients in a conversation to send speech- or text-based messages, with easy support for transcription and translation
    d. Conversational transcription : This functionality enables real-time speech recognition, speaker identification and diarization. It's perfect for transcribing in-person meetings with the ability to distinguish speakers.
    e. Custom Speech models : In case you have a specific use case that uses industry-specific vocabulary [Eg : in a Healthcare scenario] or the audio contains ambient noise [Eg : in a Manufacturing scenario] and the out-of-the-box Speech-to-text functionality doesn't provide the expected outcome, you can create Custom Speech Models for your use case. This functionality enables you to create customised

models by providing training data [Audio and corresponding text] to address ambient noise or industry-specific vocabulary. You can train & deploy a custom translation system without requiring machine learning expertise.

f. Pronunciation assessment : This functionality enables you to evaluate the pronunciation in the input audio and give scores for accuracy, fluency, completeness & pronunciation, at the overall speech level and at an individual word level, basis the intended text input that is provided.

2. Text-to-speech : This functionality generates spoken audio for any input text. You can choose from more than 250 voices in over 70 languages & variants, select male or female voices and select different speaking styles such as Newsreader, cheerful, empathetic etc. In addition to the out-of-the box functionality, you can add customisations to the spoken audio in the following ways -
    a. Audio content creation : This functionality allows you to customise the output by adjusting pronunciation, speaking rate, volume & pitch, adjust the breaks between words. This can be done in real time using SSML or in batch using the Speech Portal.
    b. Custom voices : This functionality allows you to create custom voice fonts unique to your brand or product.

3. Speech Translation : This functionality enables real-time, multi-language translation of speech in your applications, tools, and devices. This provides the capability for both speech-to-speech and speech-to-text translation. You can choose either of the 2 approaches –
    a. Pre-built : This is the translation functionality available out-of-the-box.
    b. Customised : This allows you to create custom models to recognise domain-specific terminology and unique speaking styles. You can train & deploy a custom translation system without requiring machine learning expertise.

4. Voice Assistant : This functionality allows you to integrate Speech service in your applications to create natural, human-like conversational interfaces & experiences. This provides you 2 major functionalities :
    a. Direct Line Speech (via Azure Bot Service) : This is used in scenarios where you want an open-ended conversation with robust skills integration and full deployment control.
    b. Custom Commands : This is used in scenarios where you want voice commanding or simple task-oriented conversations with simplified authoring and hosting.

5. Speaker Recognition : This functionality allows you to verify and identify speakers by their unique voice characteristics.

Since this workshop focuses on GUI and REST APIs, we have covered the following functionalities that are available as REST APIs :

1. Speech to text [Real time STT, Batch STT, Custom Speech models, Pronunciation assessment]
2. Text to speech [Real time audio generation, Batch content creation]

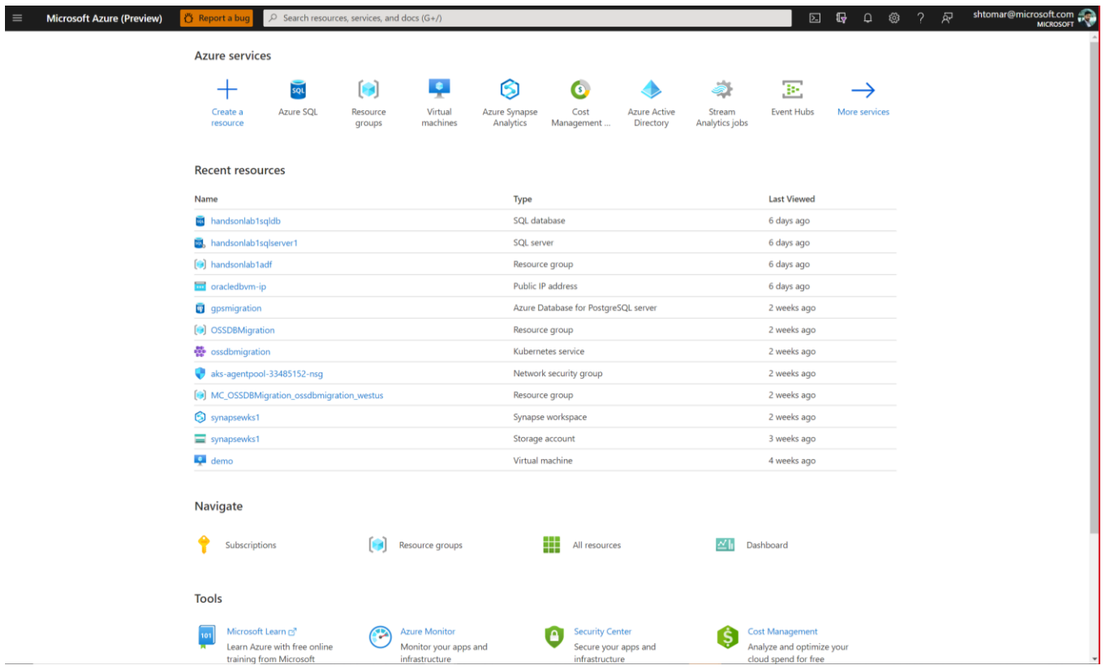
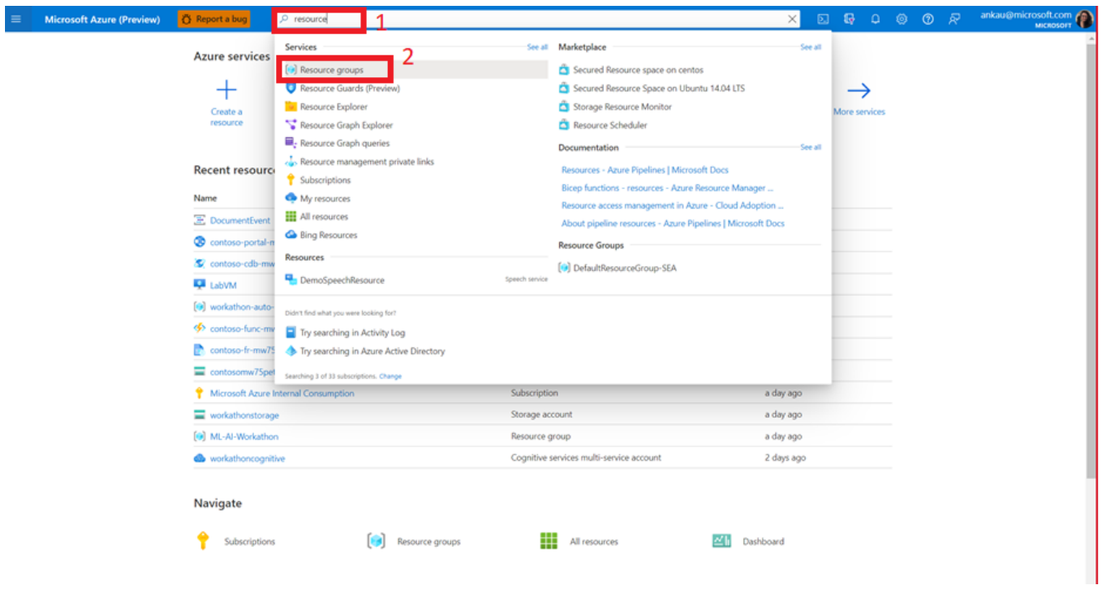
Services like Custom Voice are gated in accordance with Responsible AI principles, hence were skipped.

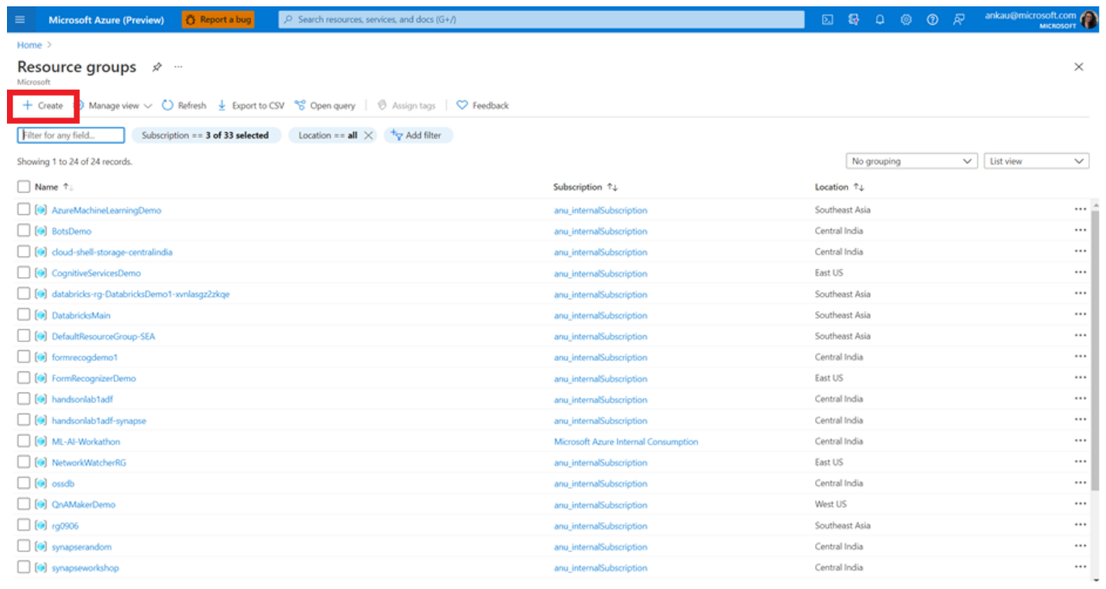
<u>Step by step hands on guide to go from *Zero to Hero*</u>

<u>Pre-requisites</u>

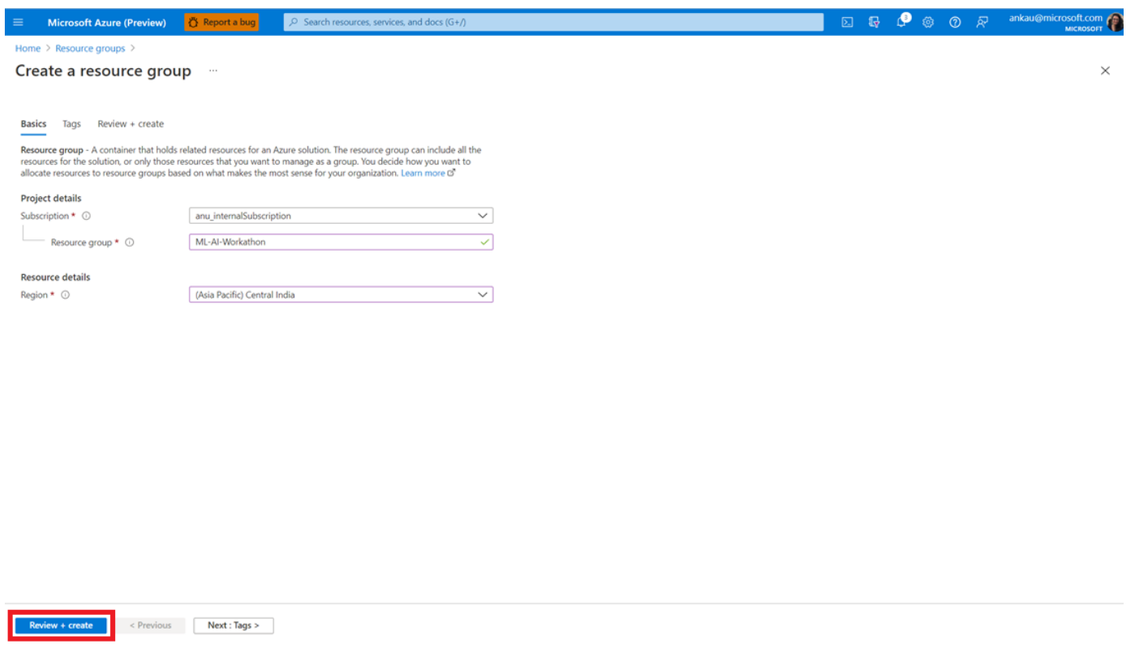
- Download & Install Postman
    - Postman is a free tool which allows you to make API calls
    - You can download the desktop application or get started using the web version (Download Postman | Try Postman for Free)
- An active Azure Account
    - You can use your current Azure Subscription or get started by creating a free trial account (https://azure.microsoft.com/en-in/free)
- Download the data for STT & TTS from the Data Folder.

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.



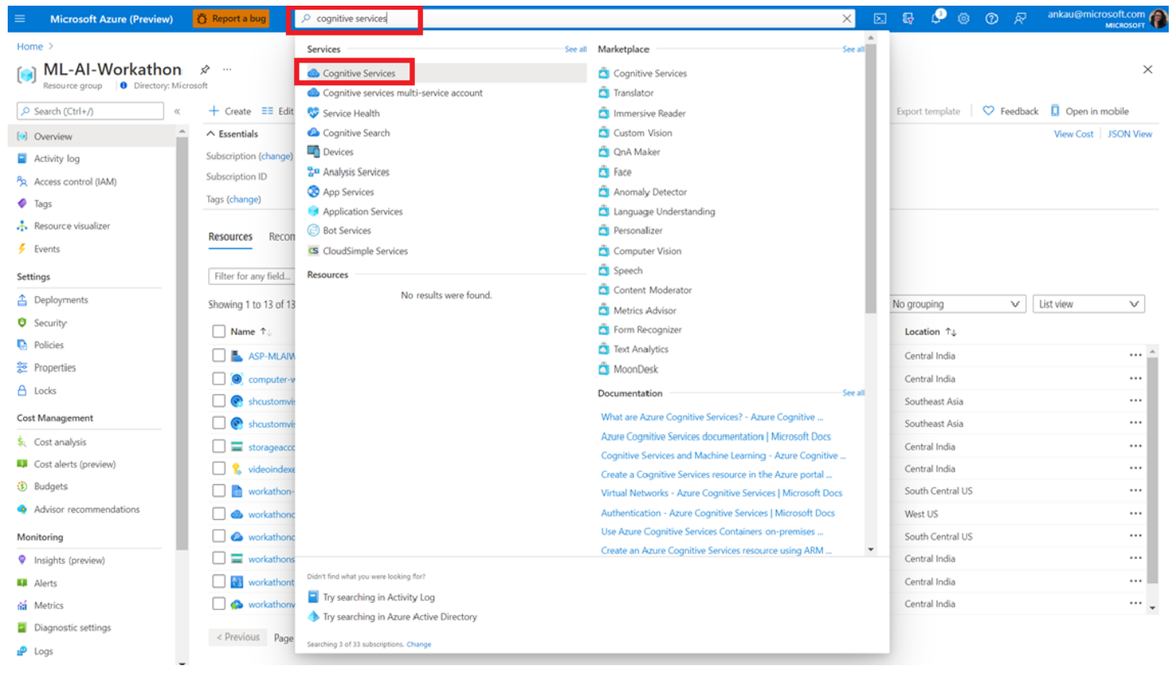
Click create to create a new resource group.



Enter the details –

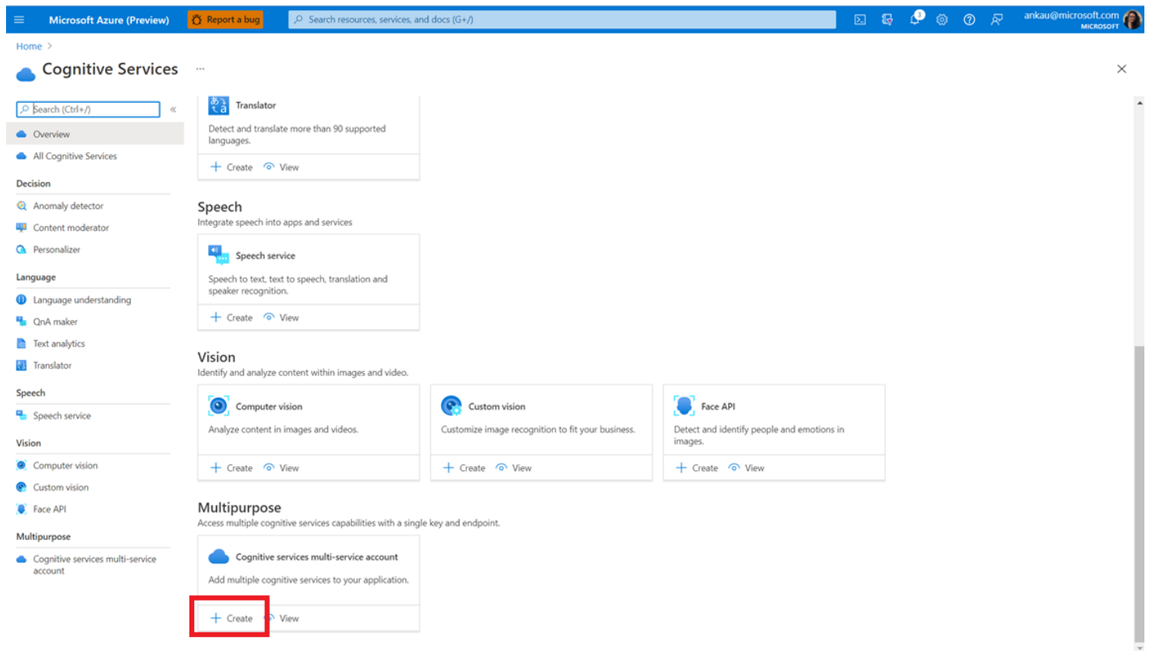
1. Subscription : Azure subscription in which you want to deploy the resource group
2. Resource Group : Name of your choice for the resource group
3. Region : Region where you want to deploy the resource group

Click Review + Create.



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

You can skip this step if you already have a Cognitive Service in place for Speech Service. This can be a multipurpose Azure Cognitive Resource or a Speech Resource.

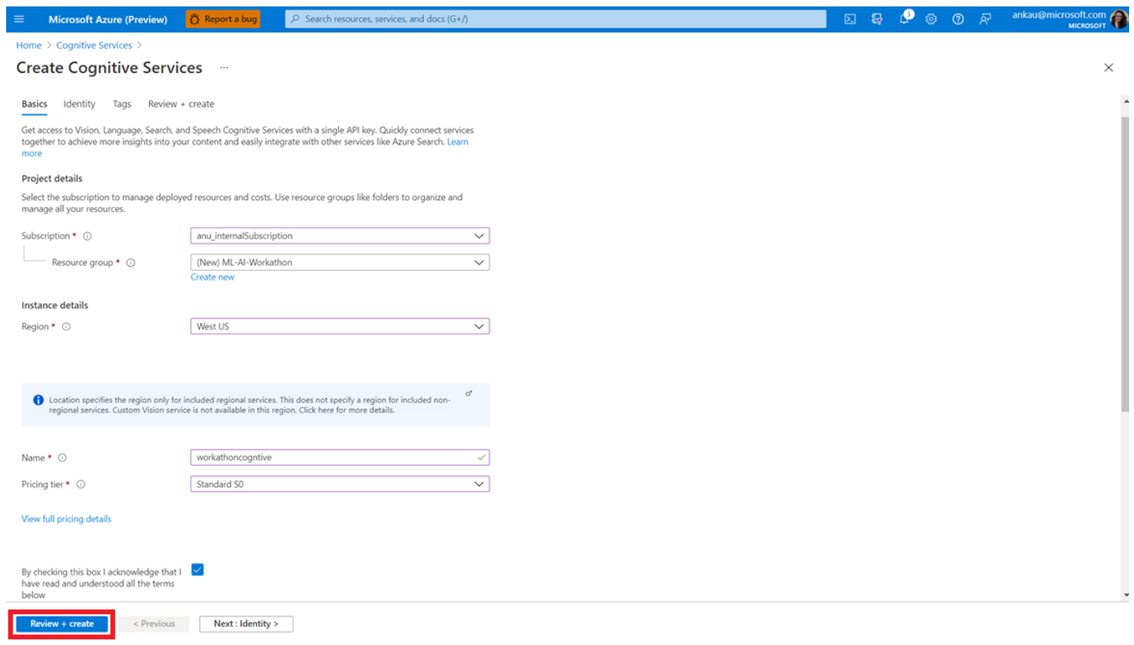


Create a multipurpose cognitive service

Significance : A multipurpose Cognitive Service account allows you to leverage the same resource for many cognitive services, which include :

Computer Vision - Analyze images
Content Moderator - Check text, image or videos for offensive or undesirable content
Face - Recognize people and their attributes in an image
Form Recognizer - Identify and extract text, key/value pairs and table data from form documents
Language Understanding - Extract meaning from natural language
Speech - Transform speech-to-text, text-to-speech and recognize speakers
Text Analytics - Detect sentiment, key phrases, entities and human language type in text

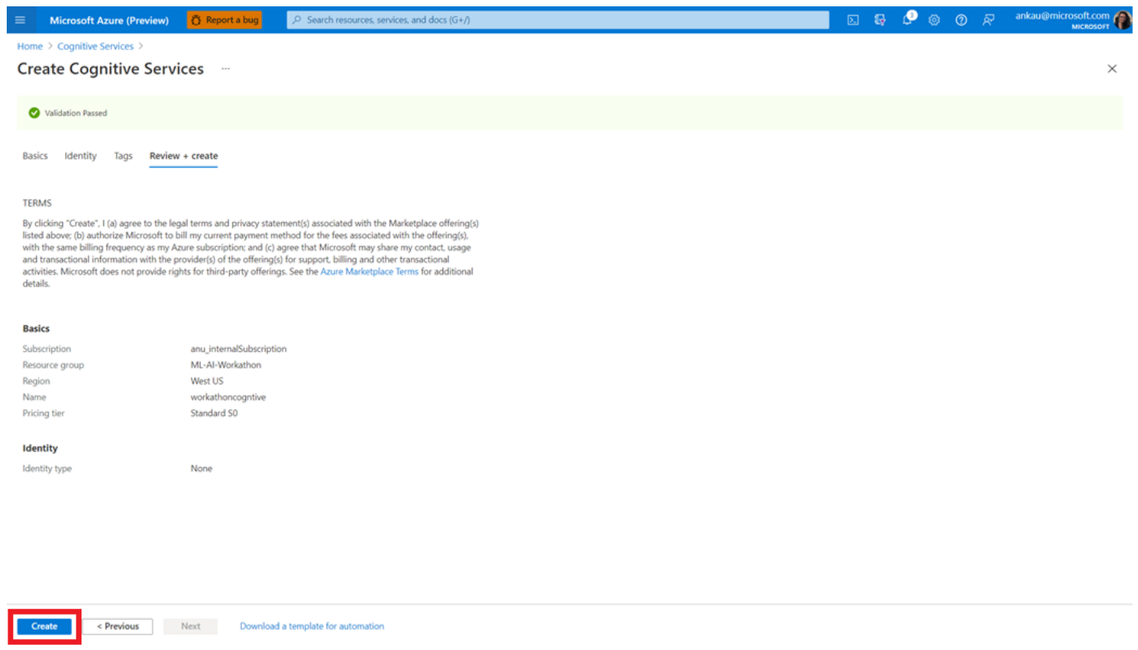
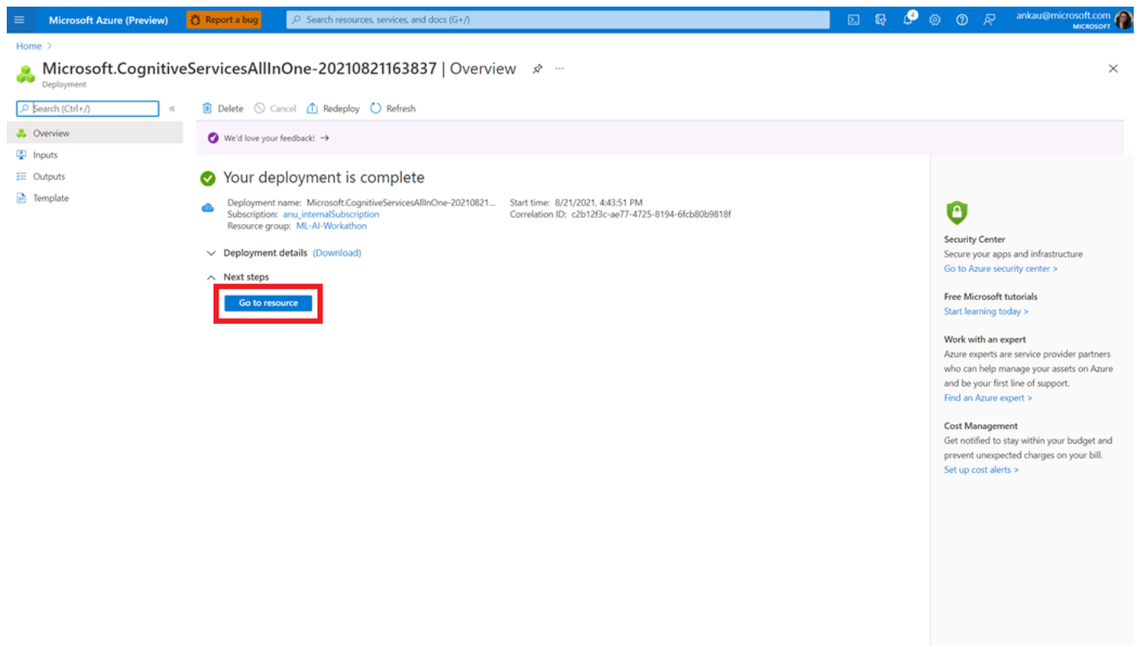
In this lab, we used a multipurpose Cognitive Service account since we would be learning about all the above-mentioned services.
However, you can also spin up individual services to execute these labs or for your development / production scenarios. The only difference is spinning up individual services allows logical separation from workspace stand point and easy monitoring of billability.

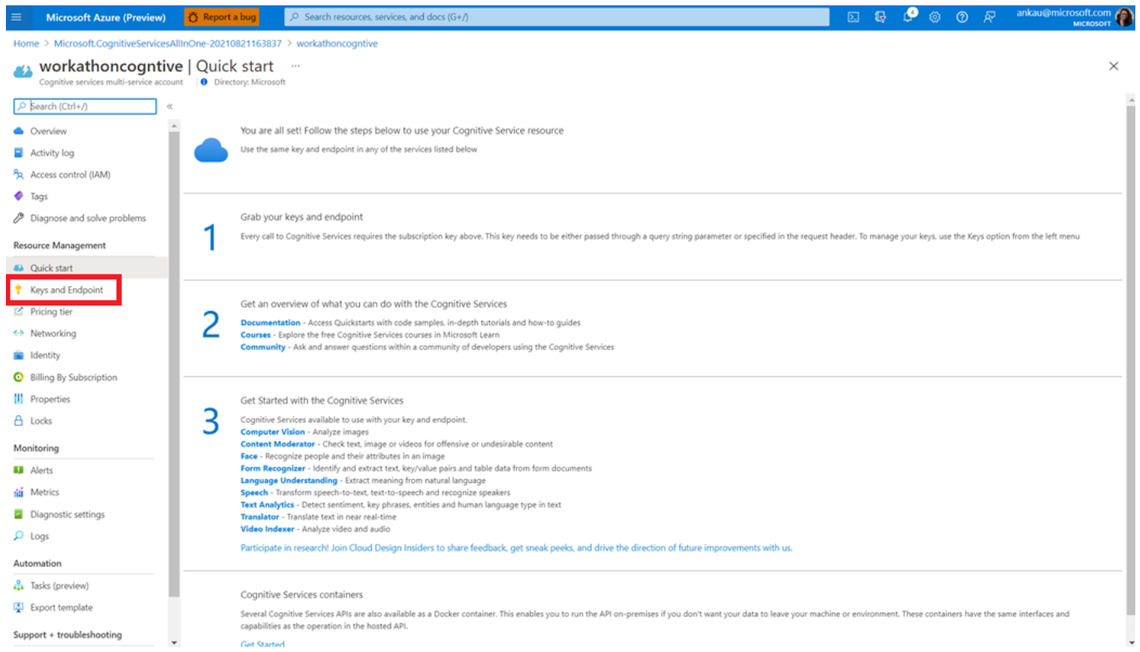


Enter the details to create a new cognitive service as follows -

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.

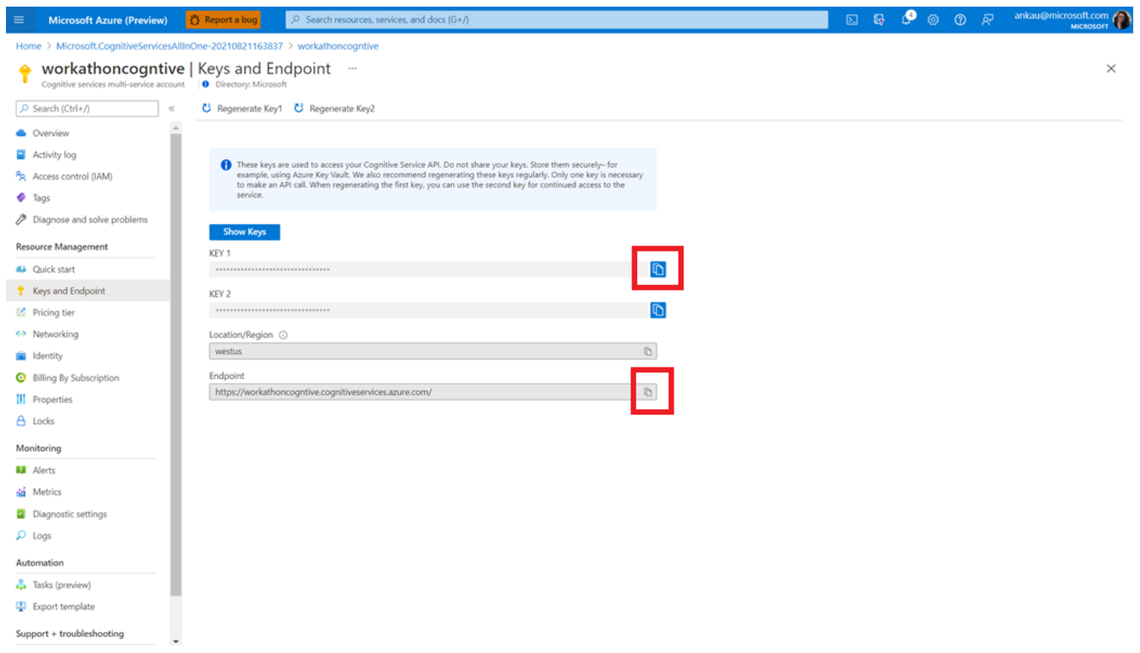
	Verify the details and click Create.
	After the resource has been deployed, click Go to Resource.



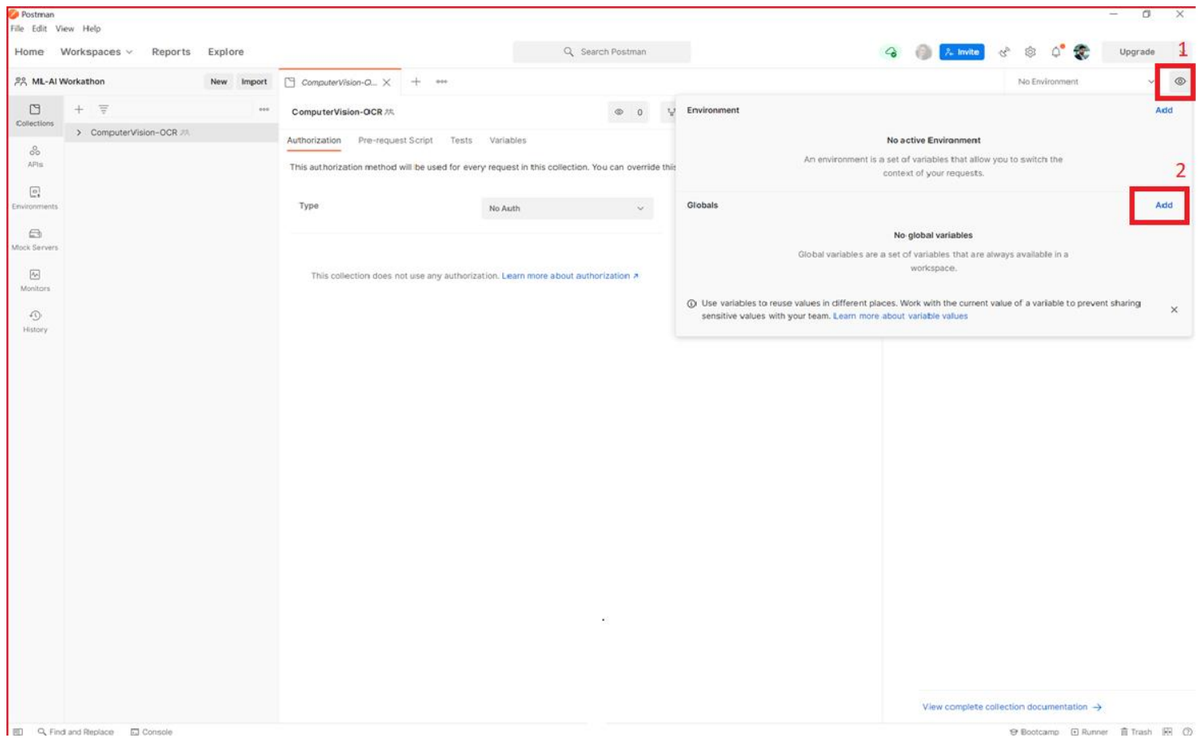
<u>Copy keys & endpoints</u>

On the Quick start page, you can find details about different cognitive services and can click the hyperlinks to learn more.

Click Key and Endpoints.



Copy the Key and Endpoint. Paste these in a notepad. You will leverage these in the next steps.



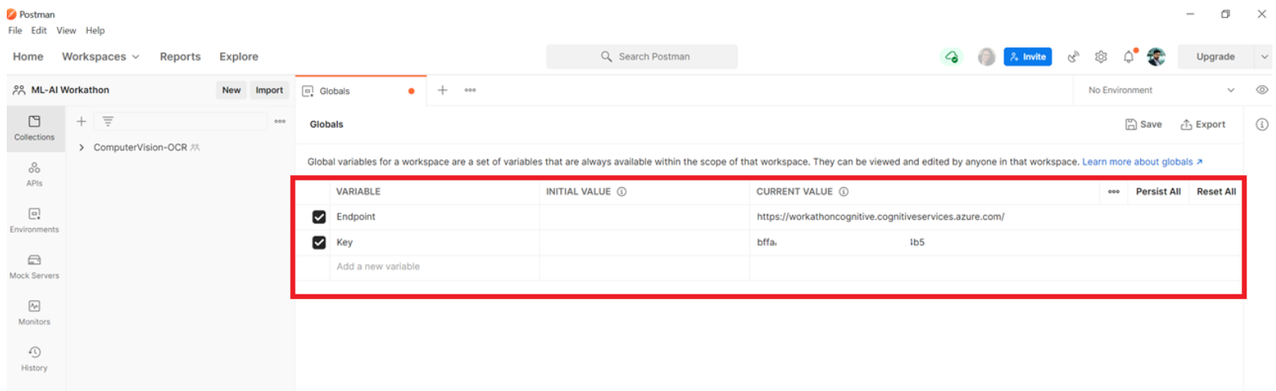
We have now switched the interface to Postman to explore the Speech service. If you haven't downloaded the Postman client, you can use web version.

<u>Configure global variables in Postman</u>

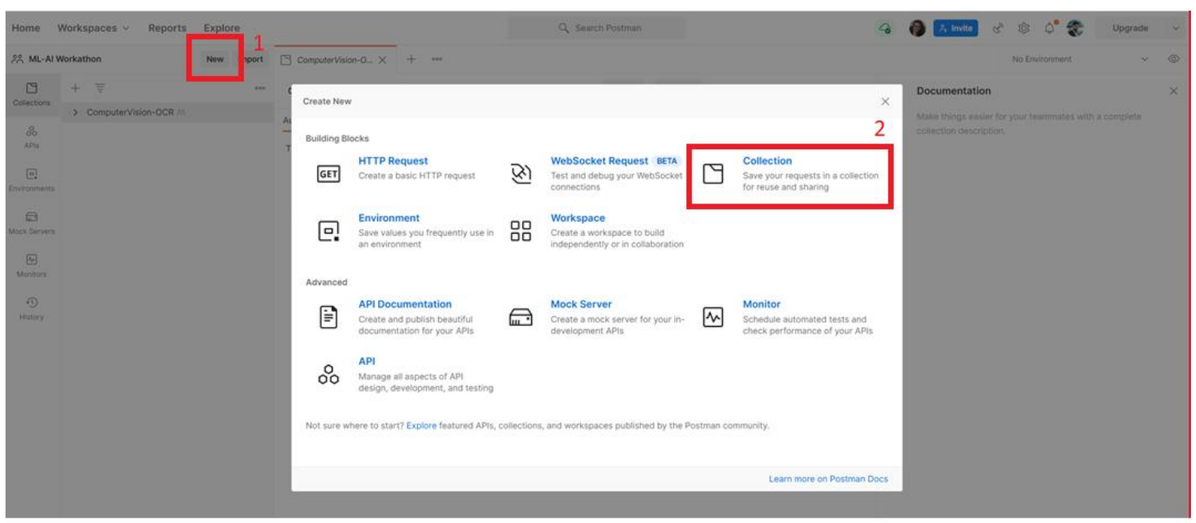
Significance :
The global variables will be created once and leveraged time & again, in each API request that we make through Postman.
This way, you will not have to hard code the Endpoint & Key for every request you make, thereby, making it more secure. This will also save time and effort.

Follow step 1 & 2 to add global variables for :

1. Endpoint – Paste the endpoint of the Multipurpose service account copied earlier
2. Key - Paste the Key of the Multipurpose service account copied earlier



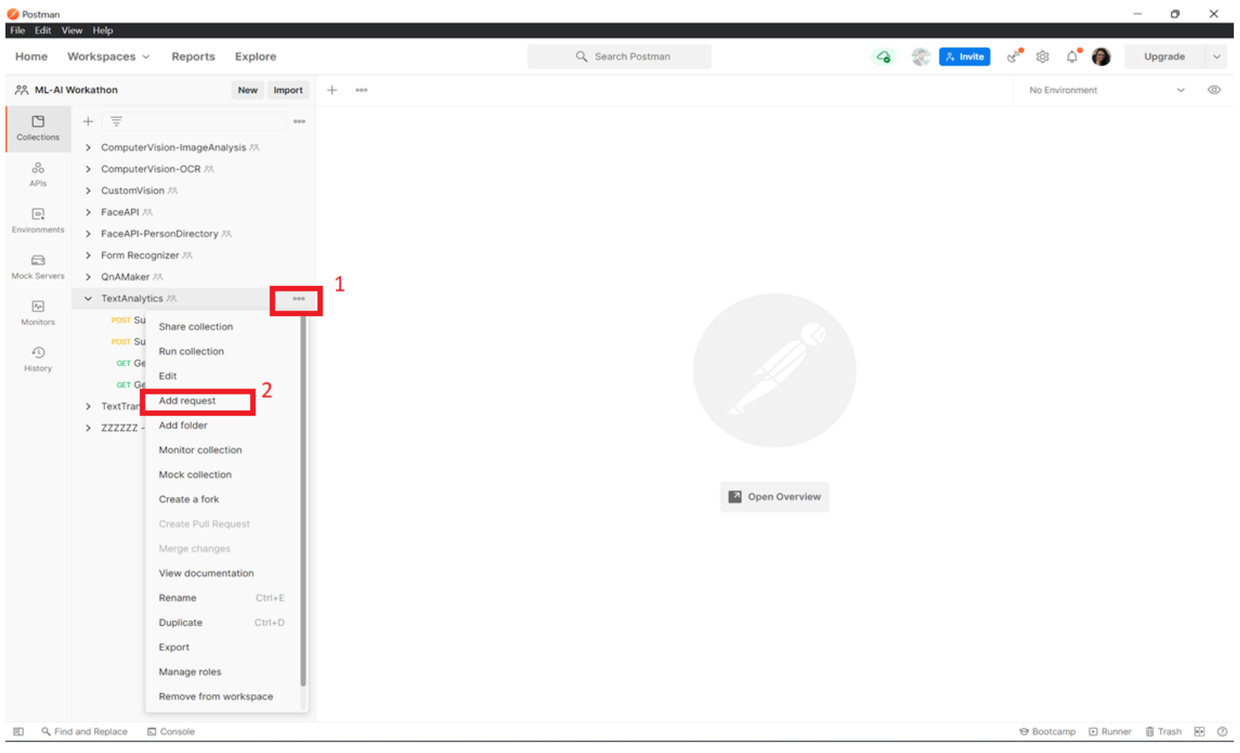
Once you have added the global variables, they will appear this way in your global variables section.



<u>Create new collection in Postman</u>

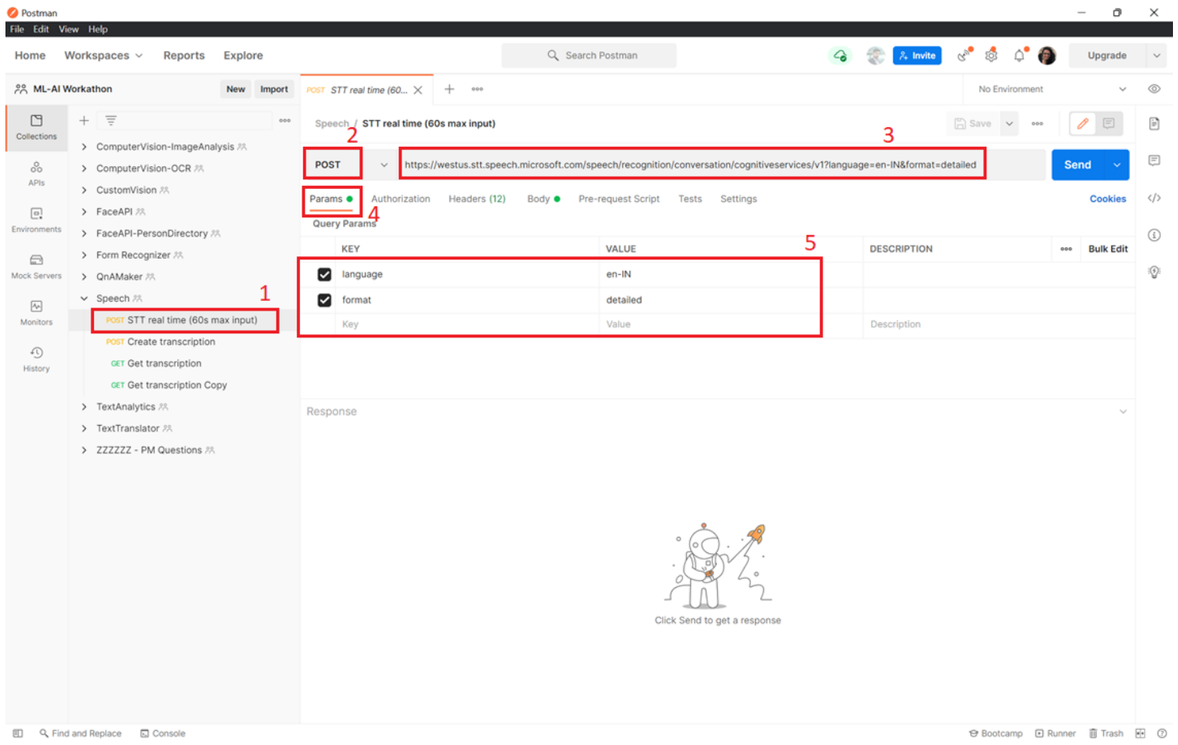
Open Postman > select New.
On the pop up select Collection.
Name the collection Speech.

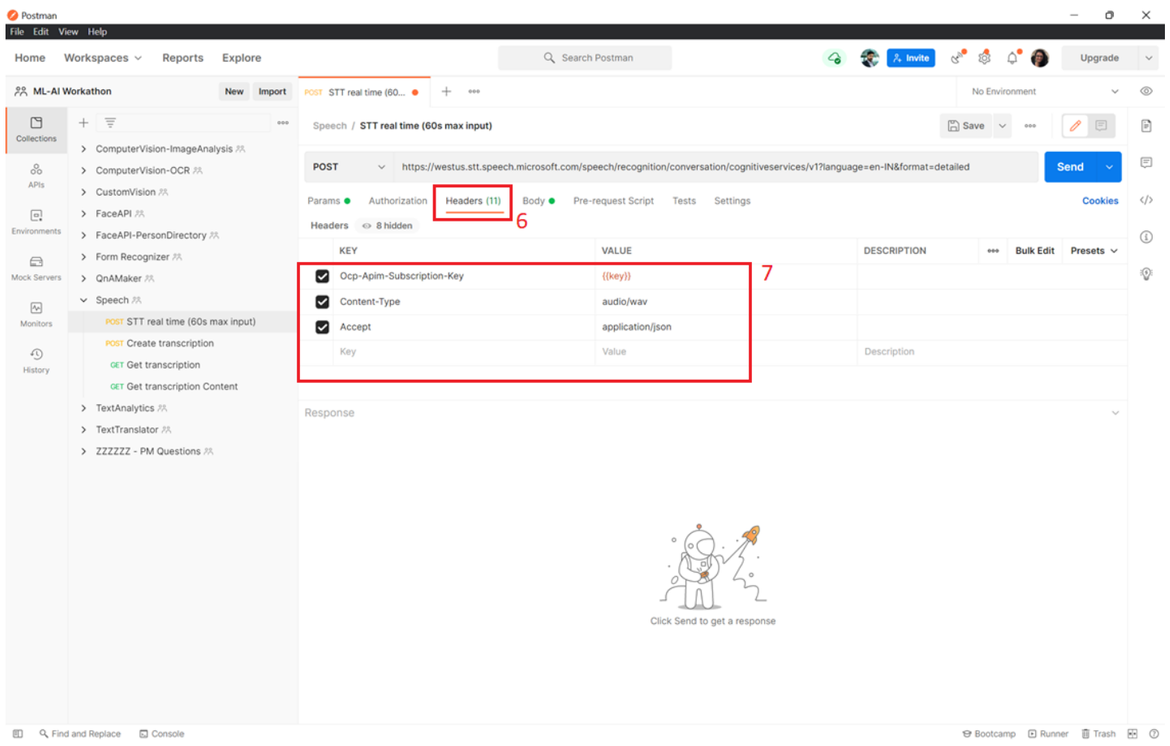
Collection is like a folder for managing the API call requests.



Once you have created the collection, follow steps 1 & 2, to create a new request.

SPEECH TO TEXT





Realtime STT API Call (sync)

This request upon successful execution will return the text for the input speech file.
This uses an audio file as input and the file should be less than 60s in length.
This is a synchronous single API call and can be leveraged in real time Speech to Text scenarios.

URL :
https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-IN&format=detailed

Params :
Language : en-IN
Format : detailed
These will be auto populated from the URL. You can change them if you want.

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

Body :
Select the binary input format and choose a local file from your machine.

Significance of input & output

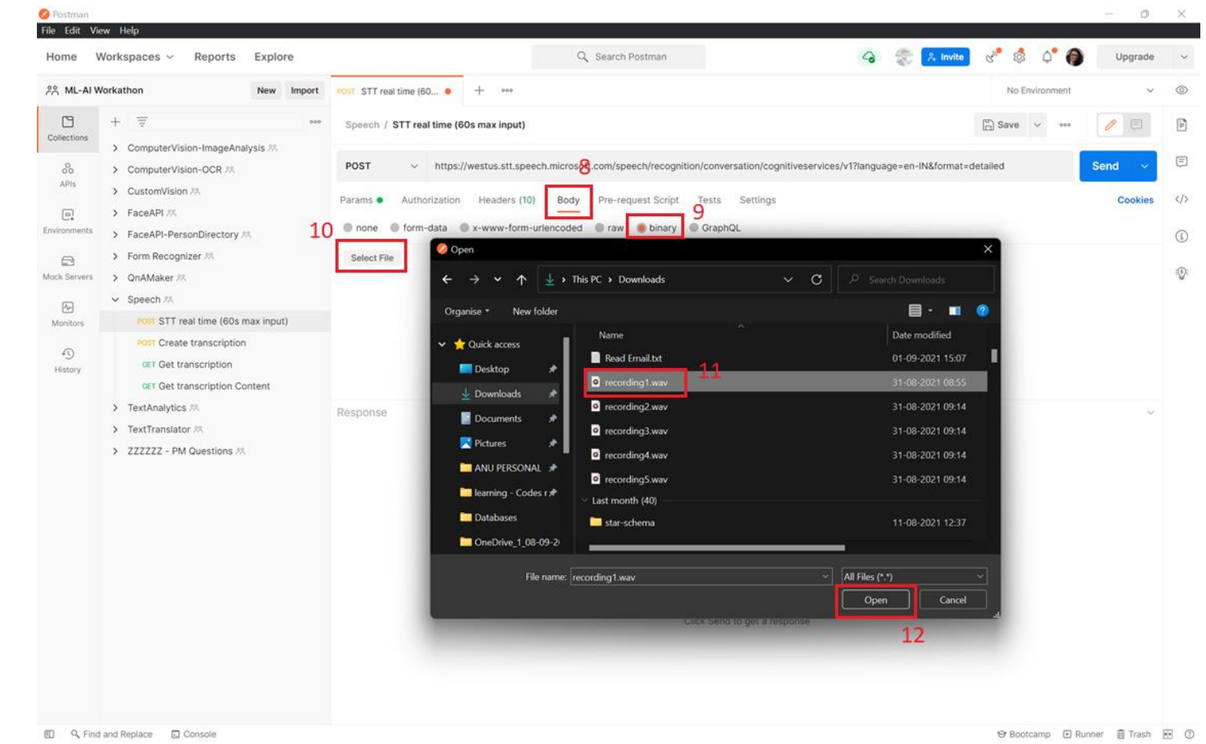
1. {{key}} : Value being picked from global variables

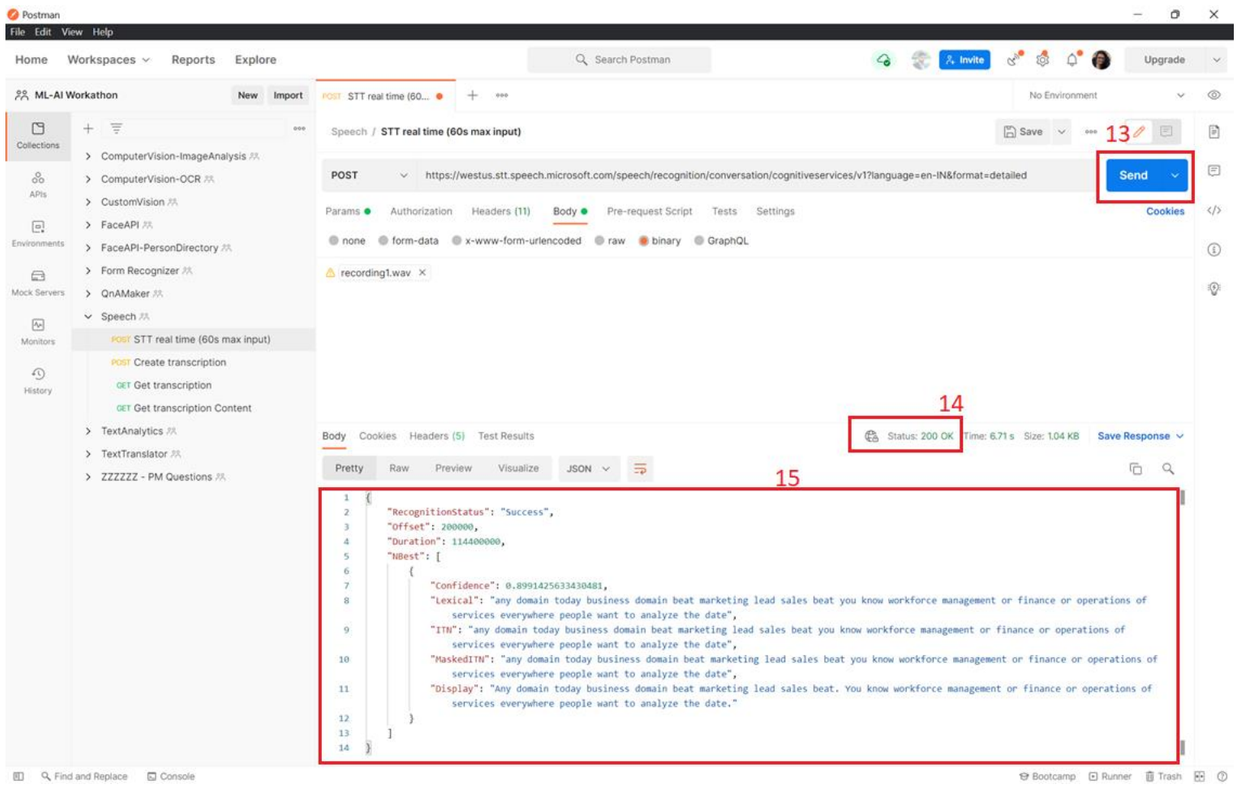
2. Headers:
   Ocp-Apim-Subscription-Key : This is the Azure Cognitive service key, that will authenticate the request.
   Content-Type : This refers to the input type that you provide in the body.
   Accept : This refers to the output type that you are expecting.

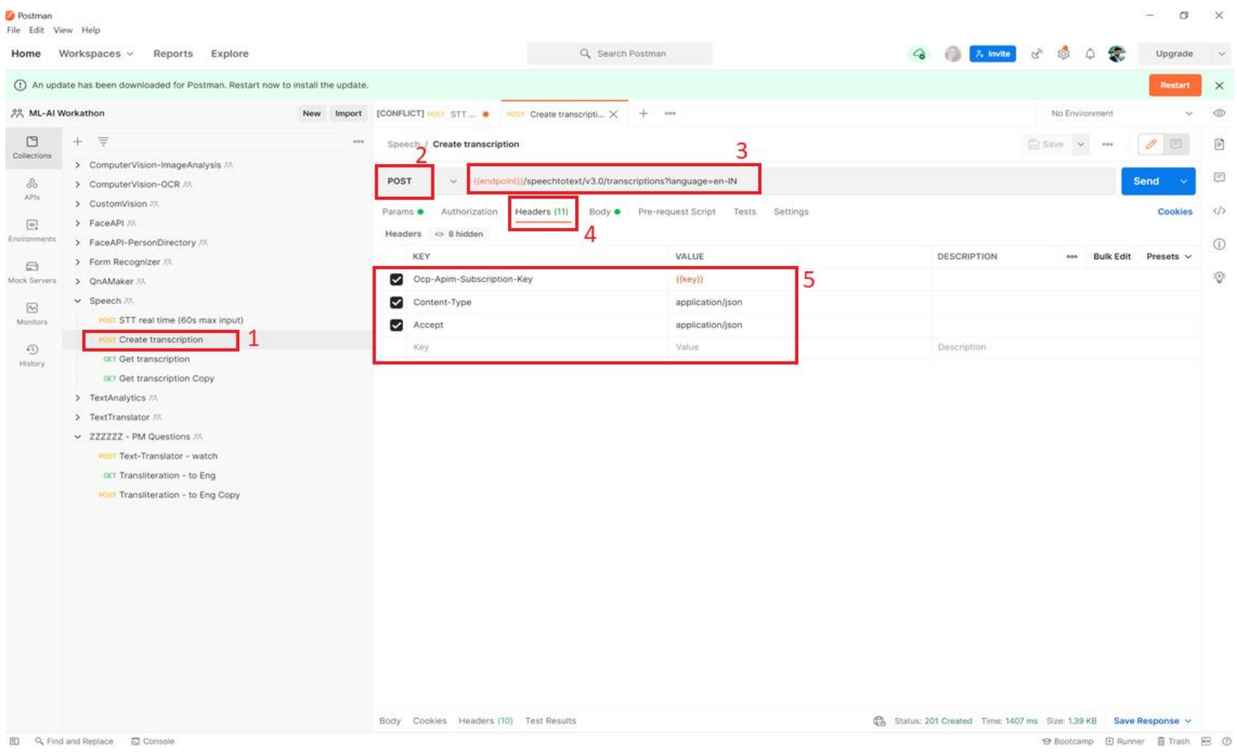
3. Params
   language : this refers to the input language of the audio file and the expected output
   format : setting this to detailed fetched you the punctuated display text, lexical text, masked text in case there is any content moderation happening. Leaving out this parameter just fetches you the punctuated display text.

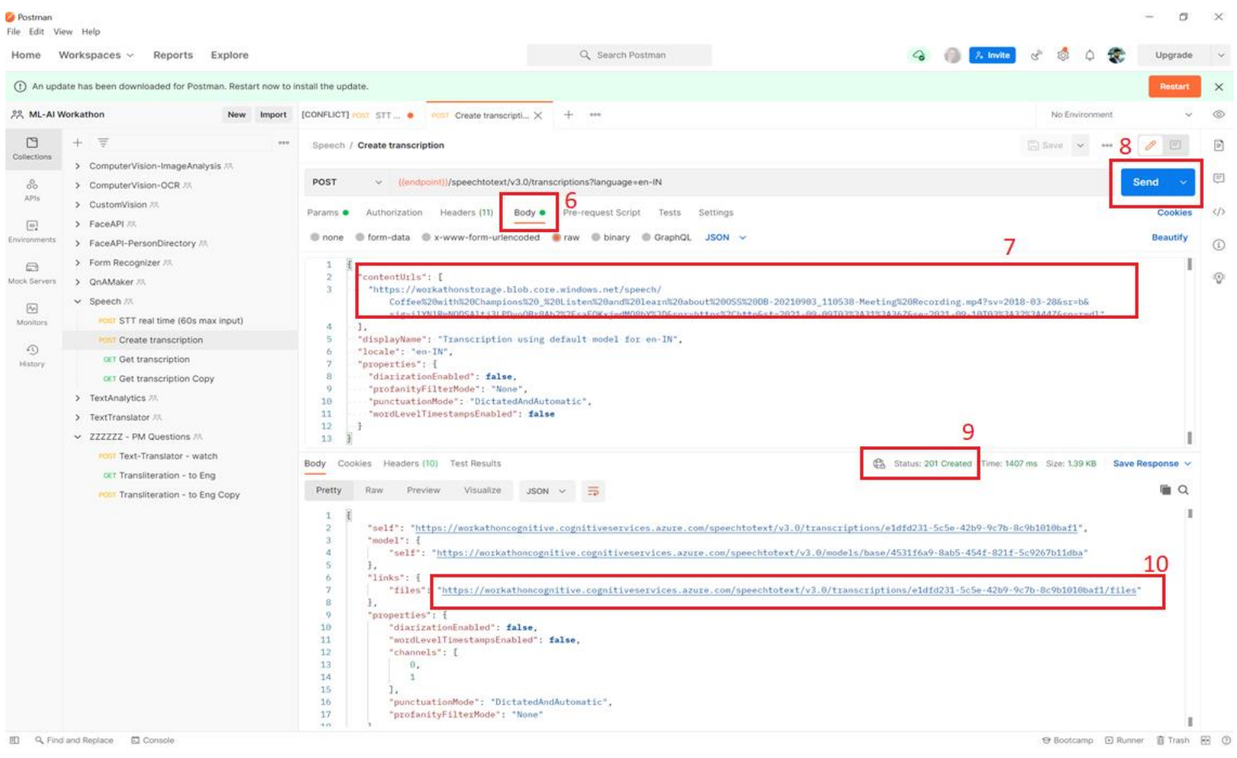
4. After you execute the call, observe the status returned, as shown in step 14. This should reflect 200 OK.

5. Observe the JSON output and observe the different text formats returned.


Postman
File Edit View Help
Home Workspaces Reports Explore
Search Postman
Invite
Upgrade
ML-AI Workathon
New Import
POST STT real time (60...
No Environment
Speech / STT real time (60s max input)
Save
POST
https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-IN&format=detailed
Send
Params Authorization Headers (10) Body Pre-request Script Tests Settings
Cookies
none form-data x-www-form-urlencoded raw binary GraphQL
Select File
8
9
10
Collections
APIs
Environments
Mock Servers
Monitors
History
ComputerVision-ImageAnalysis
ComputerVision-OCR
CustomVision
FaceAPI
FaceAPI-PersonDirectory
Form Recognizer
QnAMaker
Speech
STT real time (60s max input)
Create transcription
Get transcription
Get transcription Content
TextAnalytics
TextTranslator
ZZZZZZ - PM Questions
Response
Open
This PC › Downloads
Search Downloads
Organise
New folder
Quick access
Desktop
Downloads
Documents
Pictures
ANU PERSONAL
learning - Codes r
Databases
OneDrive_1_08-09-2
Name
Date modified
Read Email.txt
01-09-2021 15:07
recording1.wav
31-08-2021 08:55
recording2.wav
31-08-2021 09:14
recording3.wav
31-08-2021 09:14
recording4.wav
31-08-2021 09:14
recording5.wav
31-08-2021 09:14
Last month (40)
star-schema
11-08-2021 12:37
11
File name: recording1.wav
All Files (*.*)
Open
Cancel
12
Click Send to get a response
Find and Replace
Console
Bootcamp
Runner
Trash


Postman
File Edit View Help
Home Workspaces Reports Explore
Search Postman
Invite
Upgrade
ML-AI Workathon
New Import
POST STT real time (60...
No Environment
Speech / STT real time (60s max input)
Save
13
POST
https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-IN&format=detailed
Send
Params Authorization Headers (11) Body Pre-request Script Tests Settings
Cookies
none form-data x-www-form-urlencoded raw binary GraphQL
recording1.wav
Collections
APIs
Environments
Mock Servers
Monitors
History
ComputerVision-ImageAnalysis
ComputerVision-OCR
CustomVision
FaceAPI
FaceAPI-PersonDirectory
Form Recognizer
QnAMaker
Speech
STT real time (60s max input)
Create transcription
Get transcription
Get transcription Content
TextAnalytics
TextTranslator
ZZZZZZ - PM Questions
14
Body Cookies Headers (5) Test Results
Status: 200 OK
Time: 6.71 s
Size: 1.04 KB
Save Response
Pretty Raw Preview Visualize JSON
15
{
"RecognitionStatus": "Success",
"Offset": 200000,
"Duration": 114400000,
"NBest": [
{
"Confidence": 0.8991425633430481,
"Lexical": "any domain today business domain beat marketing lead sales beat you know workforce management or finance or operations of services everywhere people want to analyze the date",
"ITN": "any domain today business domain beat marketing lead sales beat you know workforce management or finance or operations of services everywhere people want to analyze the date",
"MaskedITN": "any domain today business domain beat marketing lead sales beat you know workforce management or finance or operations of services everywhere people want to analyze the date",
"Display": "Any domain today business domain beat marketing lead sales beat. You know workforce management or finance or operations of services everywhere people want to analyze the date."
}
]
}
Find and Replace
Console
Bootcamp
Runner
Trash





Batch API call (async)

This request upon successful execution will return the text for the input speech file.
This uses a SAS key to the file in the blob storage as input and can include files of longer length.
This is an asynchronous API call, that gives output in 3 API calls (1 POST call, 2 GET calls) and can be used in batch Speech to Text scenarios.

API Call 1 – POST (Create Transcription)

URL : {{endpoint}}/speechtotext/v3.0/transcriptions?language=en-IN

Params :
Language : en-IN
These will be auto populated from the URL. You can change them if you want.

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

Body :

```
{
 "contentUrls": [
  "<<Paste the SAS URL of the blob file within quotes. See step 7 in screenshot for reference.>>"
 ],
 "displayName": "Transcription using default model for en-IN",
 "locale": "en-IN",
 "properties": {
  "diarizationEnabled": false,
  "profanityFilterMode": "None",
  "punctuationMode": "DictatedAndAutomatic",
  "wordLevelTimestampsEnabled": false
 }
}
```

Significance of input & output

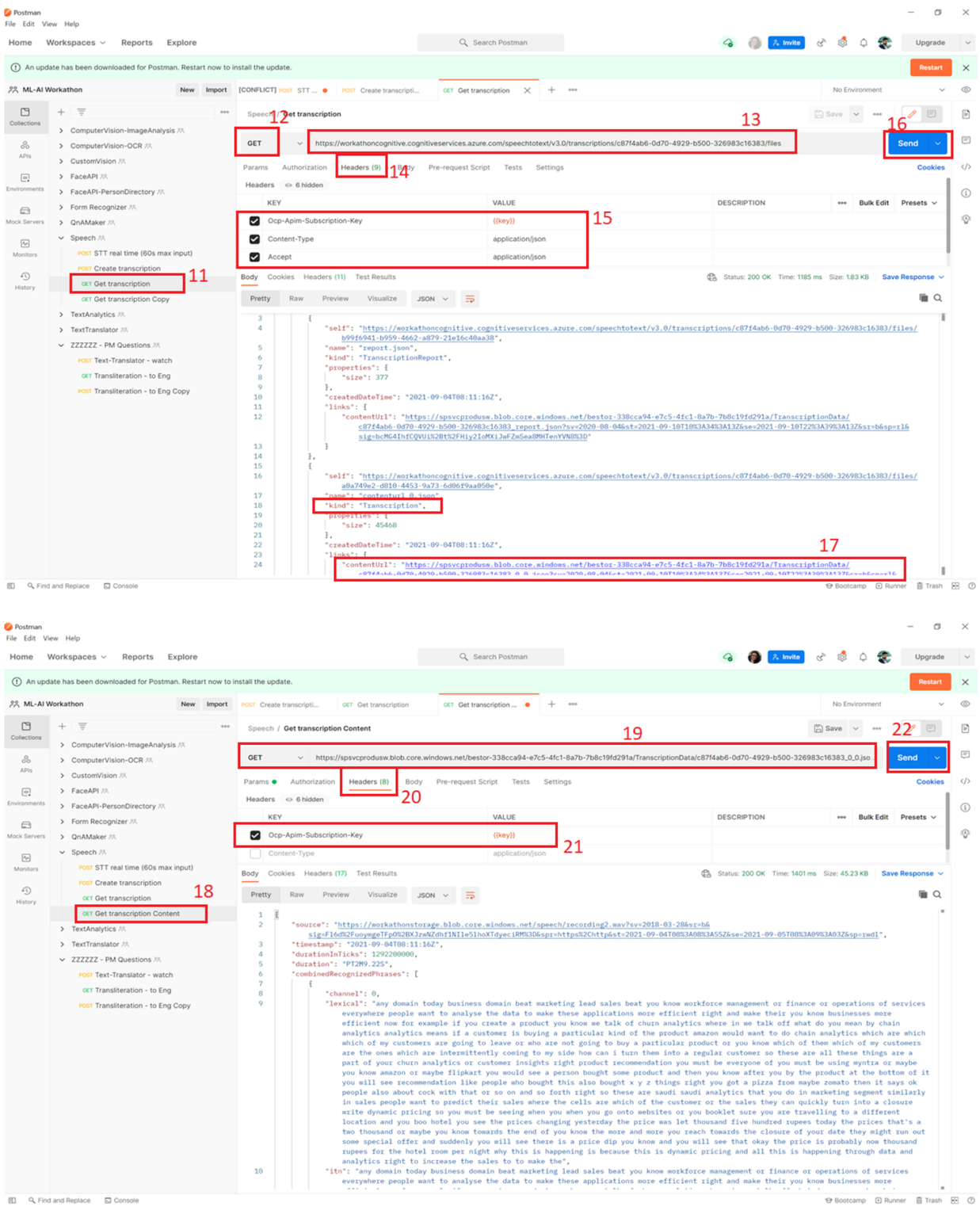
1. {{endpoint}}, {{key}} : Value being picked from global variables

2. Headers
   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.
   Accept : This refers to the output type that you are expecting.

3. Params
   language : this refers to the input language of the audio file and the expected output

4. After you execute the call, observe the status returned, as shown in step 9. This should reflect 201 Created.

5. Copy the value for (links.files) hierarchy as shown. This will be the URL for the next call.



API Call 2 – GET transcription

URL : The value for (links.files) hierarchy copied above.

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

Significance of input & output

1. {{key}} : Value being picked from global variables

2. Headers
   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.
   Accept : This refers to the output type that you are expecting.

3. After you execute the call, Copy the value for (links.contentUrl) hierarchy as shown. This will be the URL for the next call. Make sure you copy the value for the 2nd set, where the value for Kind is Transcription and not TranscriptionReport. See screenshot for reference (step 17).

API Call 3 – GET transcription content

URL : The value for (links. contentUrl) hierarchy copied above.

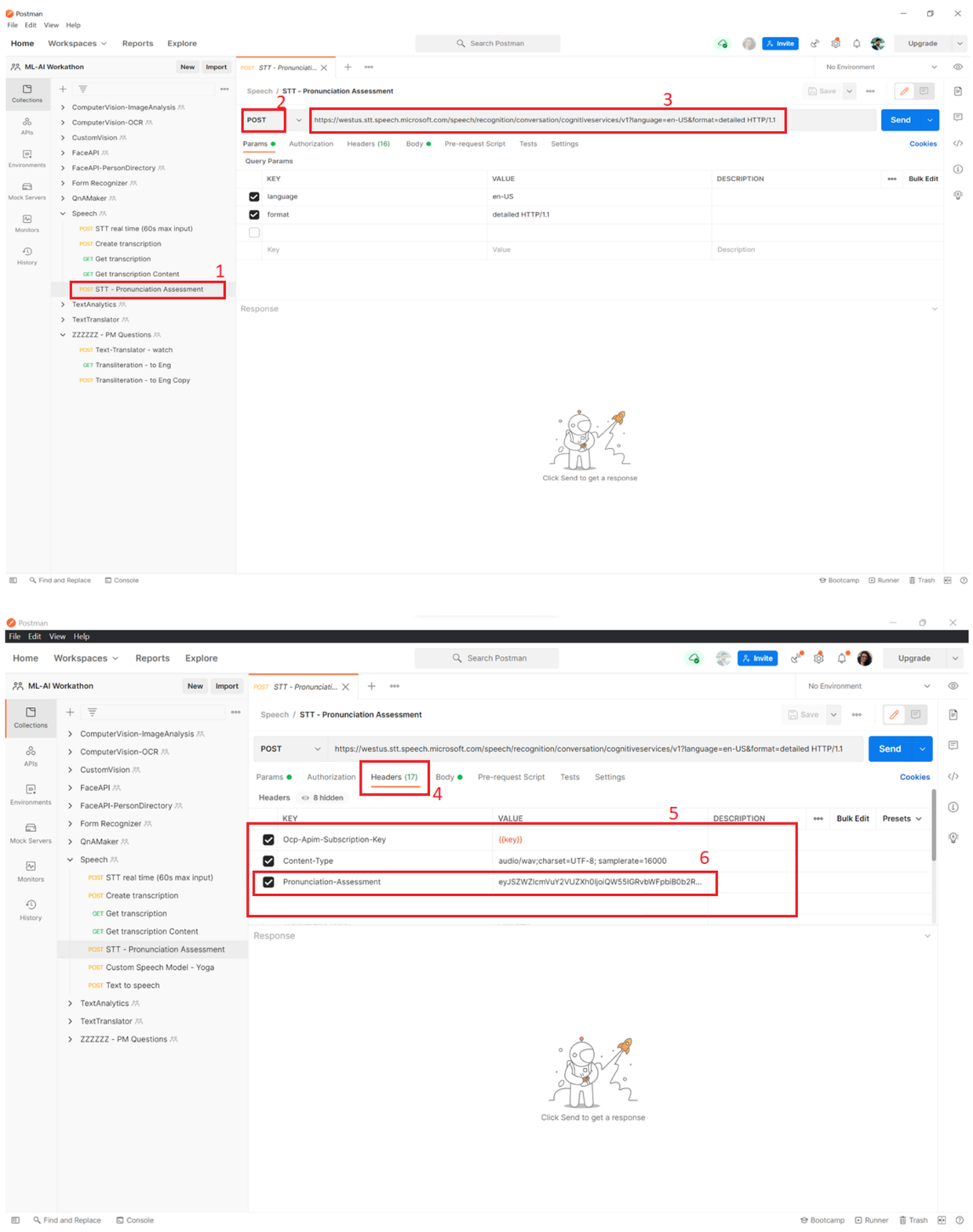
Headers :
Ocp-Apim-Subscription-Key : {{key}}

Significance of input & output

1. {{key}} : Value being picked from global variables

2. Headers
   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.
   Accept : This refers to the output type that you are expecting.

3. Observe the text translation output that you receive. The result gives the text output under 4 keys :
   - Lexical : this is the normalised text without any punctuations or letter capitalisation
   - Itn : this is the inverse-text-normalized ("canonical") form of the recognized text. It transforms entities like phone numbers, numbers, abbreviations, address to logical text (Eg : "doctor smith" to "dr smith", “November twenty fourth twenty seventeen” to “November 24, 2017.”)
   - maskeditn : this is the ITN form with profanity masking applied, if requested in the parameters [to try this, add ‘profanity’ as key in parameters with value ‘masked’]
   - Display : this is the properly formatted text with punctuations and letter capitalisation



Pronunciation Assessment

URL :
https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US&format=detailed HTTP/1.1

Params :
These will be auto populated from the URL. You can change them if you want.

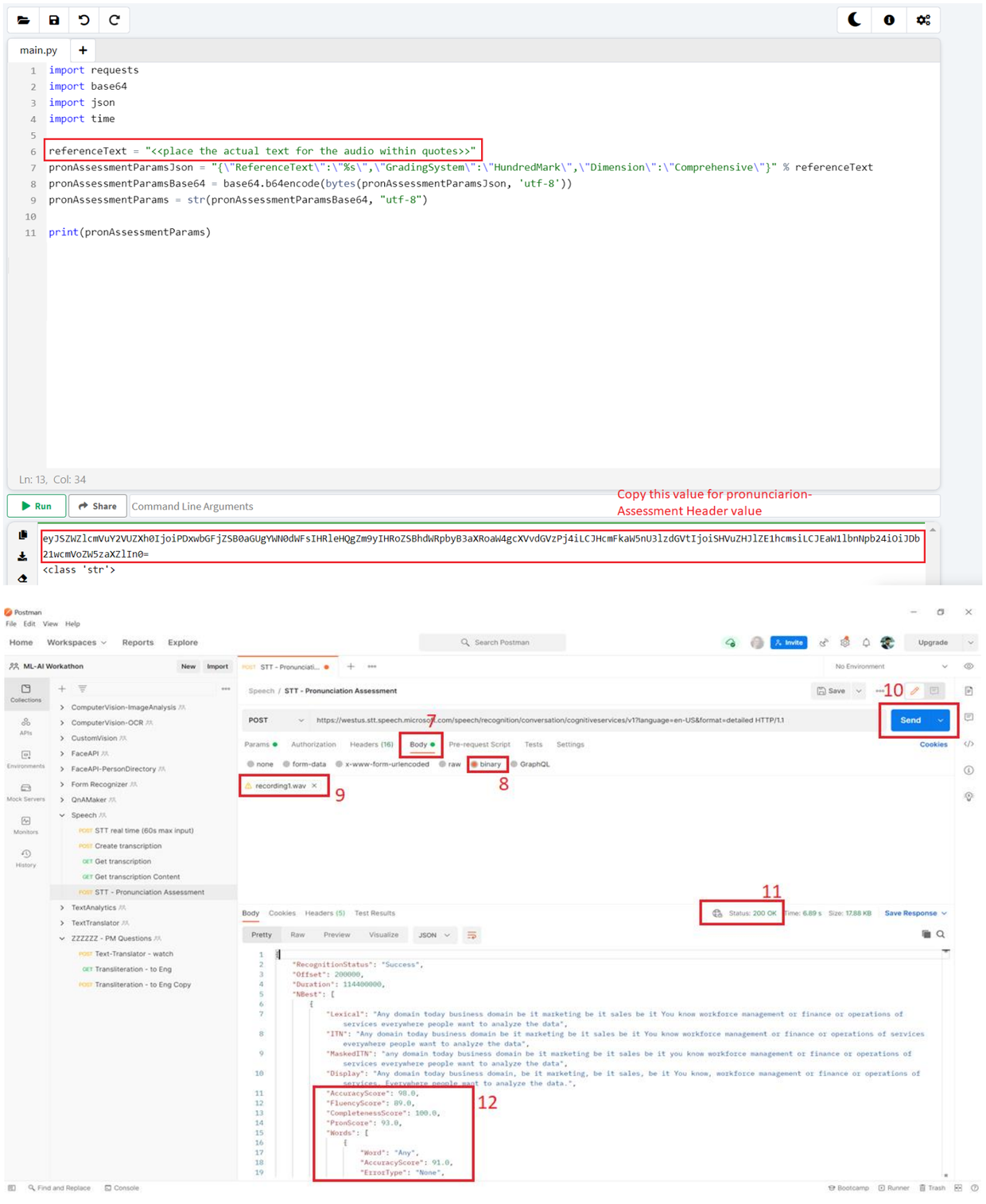
Body :
Select the binary input format and choose a local file from your machine.

Headers :
Ocp-Apim-Subscription-Key : {{key}}
Content-Type : audio/wav;charset=UTF-8; samplerate=16000
Pronunciation-Assessment : *<< This is a base64 encoded json containing multiple detailed parameters. How to obtain this is explained in the screenshot below>>*

'**Pronunciation-Assessment'** parameter specifies the sub-parameters for showing pronunciation scores in recognition results, which assess the pronunciation quality of speech input, with scores for accuracy, fluency, completeness, etc. This parameter is a base64 encoded json constituting multiple detailed sub-parameters :

1. Reference Text : The text that the pronunciation will be evaluated against
2. Grading System : The point system for score calibration. The FivePoint system gives a 0-5 floating point score and HundredMark gives a 0-100 floating point score. The Default value is FivePoint .
3. Granularity : The evaluation granularity. Accepted values are *Phoneme* which shows the score on the full text, word and phoneme level; *Word* which shows the score on the full text and word level; *FullText* which shows the score on the full text level only. The default setting is Phoneme.
4. Dimension : Defines the output criteria. Accepted values are - *Basic* which shows the accuracy score only; *Comprehensive* which shows scores on more dimensions (e.g. fluency score and completeness score on the full text level, error type on word level). The default setting is Basic .

Follow steps below to create a JSON using the above parameters and then encode it to base64, to be used as the value for 'Pronunciation-Assessment' parameter.



How to obtain Pronunciation Assessment parameter

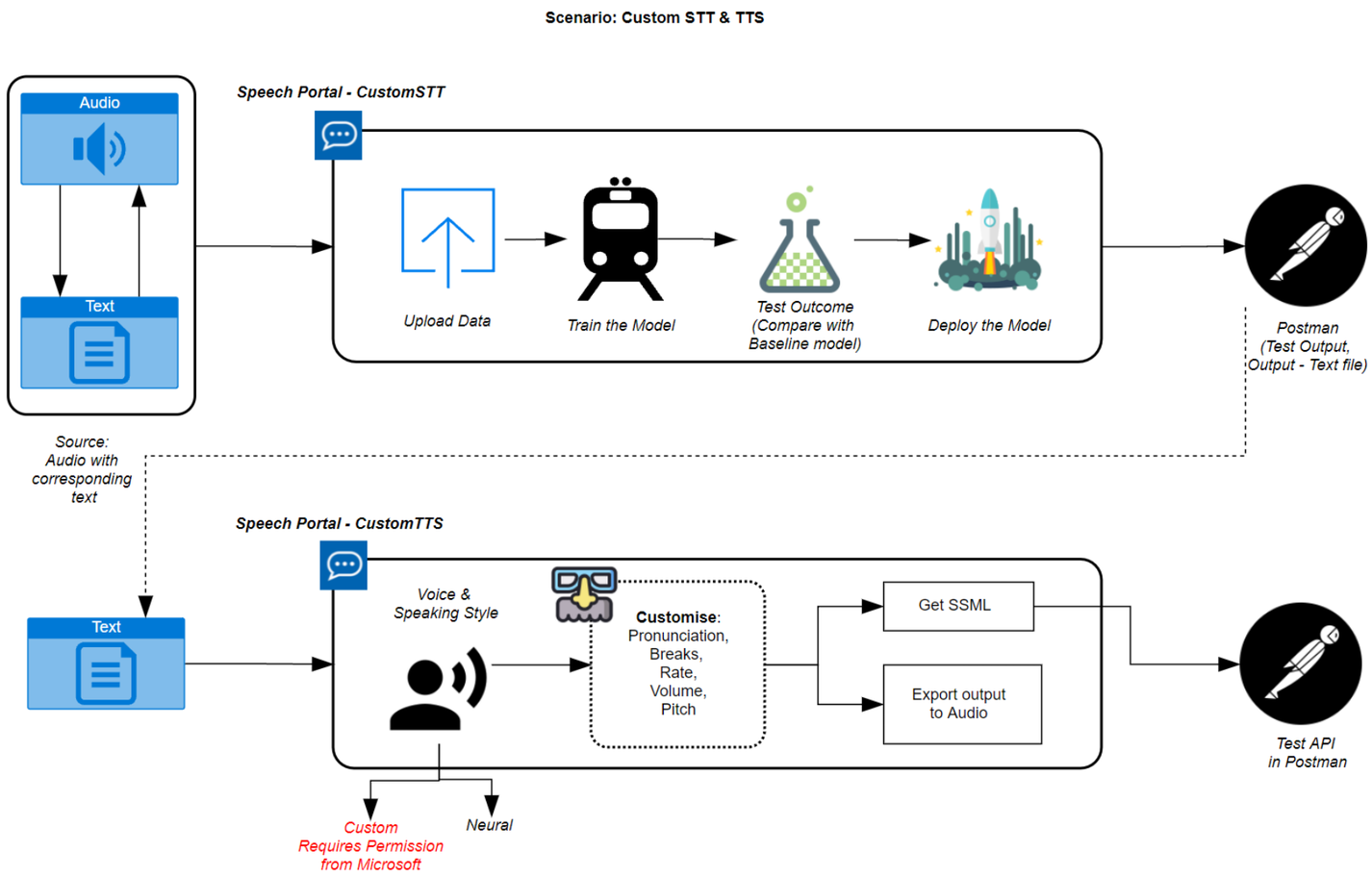
You will need some pre work to obtain the pronunciation parameter. In a production scenario, you will do this on the fly, however, since Postman doesn't support converting normal text to utf-8 and base 64, we will write some bit of Python code to obtain this parameter and then use the value in the API call.

1. Open a Python interpreter locally or online. Eg : https://www.online-python.com/
2. Use the following code to obtain the pronunciation assessment header. Replace the referenceText with the actual text for your audio. See the screenshot for reference.

```
import requests
import base64
import json
import time

referenceText = "<<place the actual text for the audio within quotes>>"

pronAssessmentParamsJson = "{\"ReferenceText\":\"%s\",\"GradingSystem\":\"HundredMark\",\"Dimension\":\"Comprehensive\"}" % referenceText

pronAssessmentParamsBase64 = base64.b64encode(bytes(pronAssessmentParamsJson, 'utf-8'))

pronAssessmentParams = str(pronAssessmentParamsBase64, "utf-8")

print(pronAssessmentParams)
```

This code converts the required JSON (pronAssessmentParamsJson) first to utf-8 encoding and then to base64 encoding as required by Pronunciation Assessment header.

Custom speech model – yoga for diabetes data



In this use case, we will build 2 custom model :

1. **Custom Speech-to-text Model** : Here we will use the 'Custom Speech Model' functionality of Speech Portal to create a custom model to convert the Audio files of an Indian Yoga Instructor to text in English. These audios contain Yoga related terms that are not so commonly used in day-to-day language and are difficult to spell, thus we require a custom model to get better results.
2. **Custom Text-to speech Model** : Here we will use the 'Audio Content Creation' functionality of Speech Portal to create a custom model to covert the text obtained above to a foreign dialect of English. The text generated above contains Yoga related terms that are difficult to pronounce, thus we require a custom model to get better results. The Audio Content Creation also allows us to add more customisations, which we have leveraged in the model.

Custom Speech-to-text Model

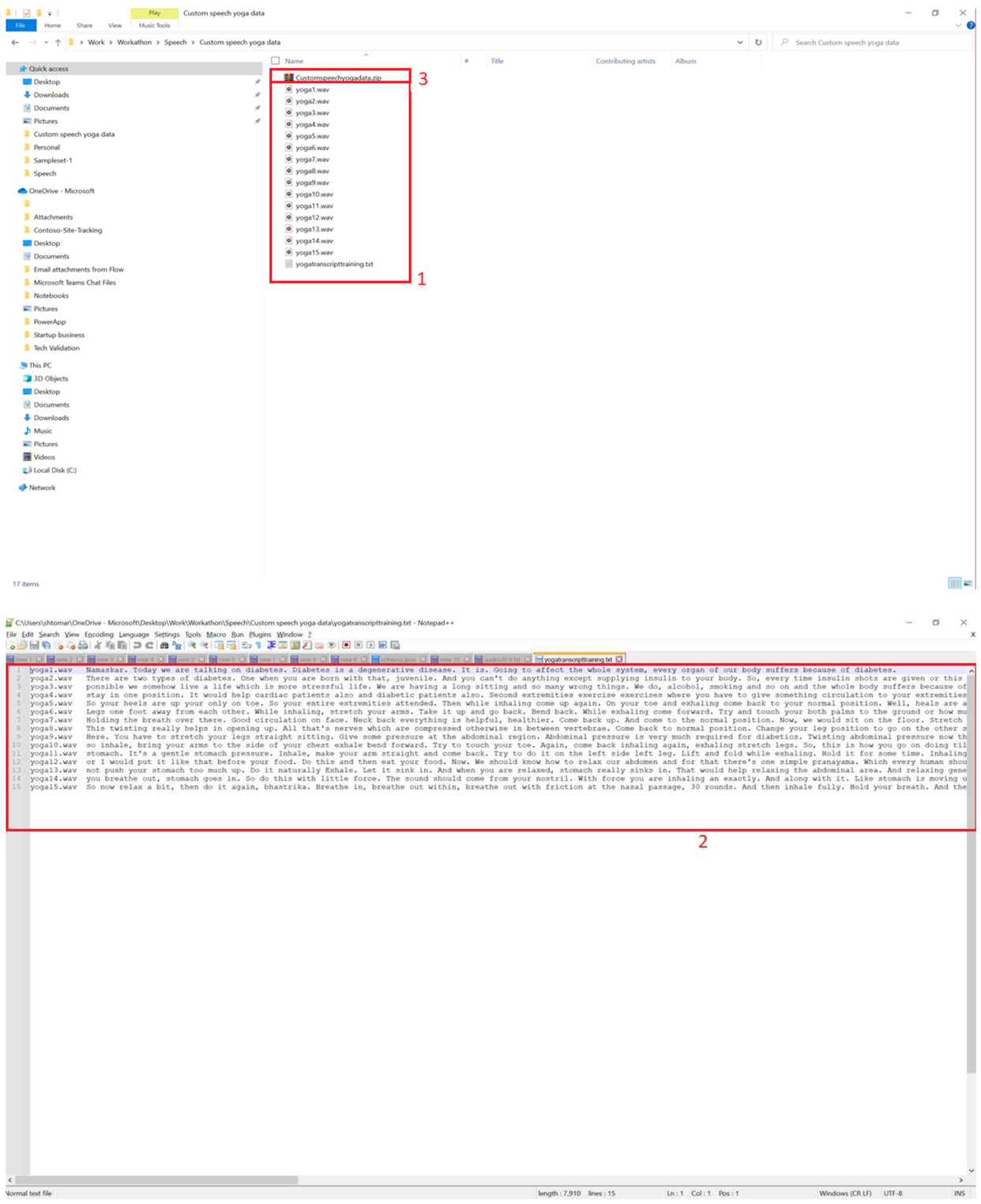
We will follow the steps as shown in the diagram above :

1. Prepare the training & testing audio content with corresponding text
2. Use the Speech Portal to upload the training & testing audio content, train a custom model, test & analyse the results and deploy the model
3. Test the URL obtained after deployment using Speech-to-text REST APIs in Postman

Custom Text-to speech Model

We will follow the steps as shown in the diagram above :

1. Obtain the text for which we want to customise the speech output. This text is obtained from the outputs of the previous model
2. Use the Speech Portal to upload the text content, add customisations such as pronunciation, phonemes, intonations, breaks, volume, pitch etc
3. Export the output audio in a batch format
4. We will also obtain the SSML for the custom content that we created, to be leveraged while making Text-to-speech REST API calls in Postman



Custom Speech-to-text Model

We will now start building the custom Speech-to-text Model.

Preparing training data

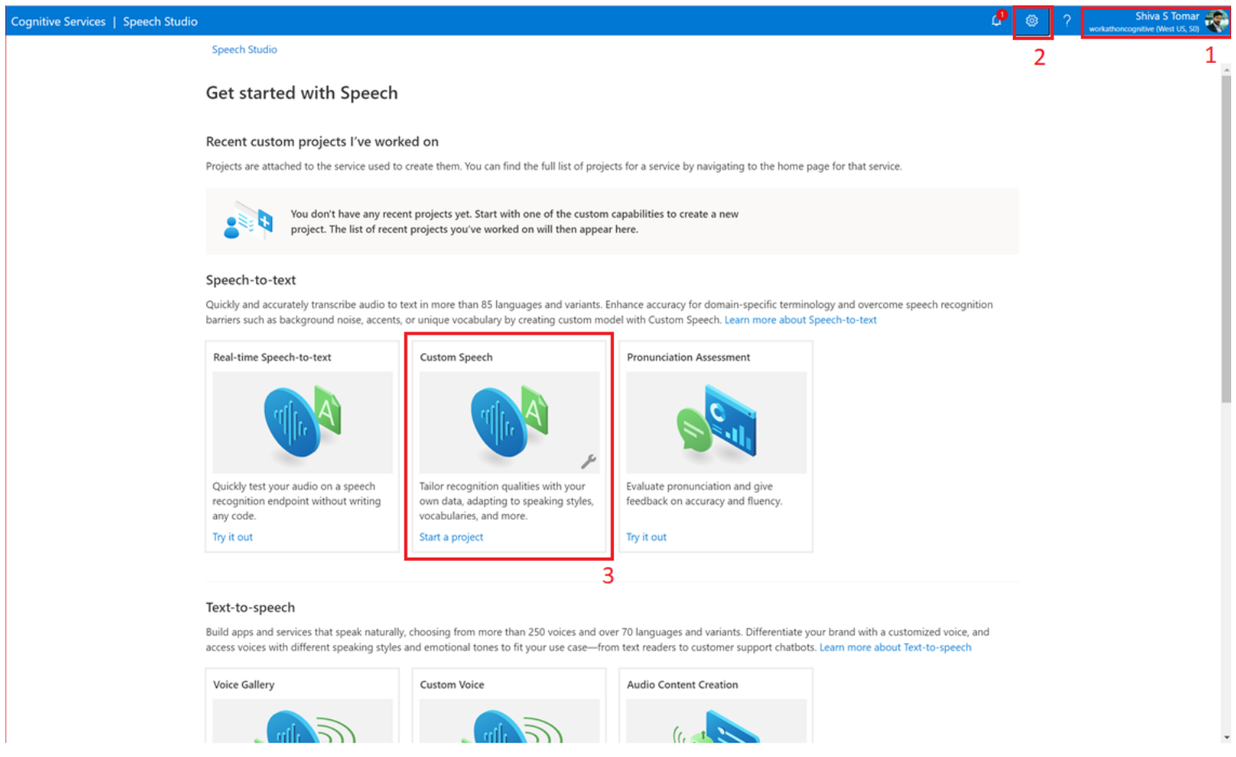
The training data consists of all the audio recordings and a text file that contains the actual text corresponding to each audio file.

Audio Files : The default audio streaming format is WAV (16 kHz or 8 kHz, 16-bit, and mono PCM). Each audio file should be of 60 seconds or less in length.
Files longer than this will give an error while training.

Text File : This is a .txt file that contains the actual text corresponding to each audio file. Each line in the text file contains the audio file name with extension and the corresponding text separated by a tab from the file name. (Step 2 as highlighted)

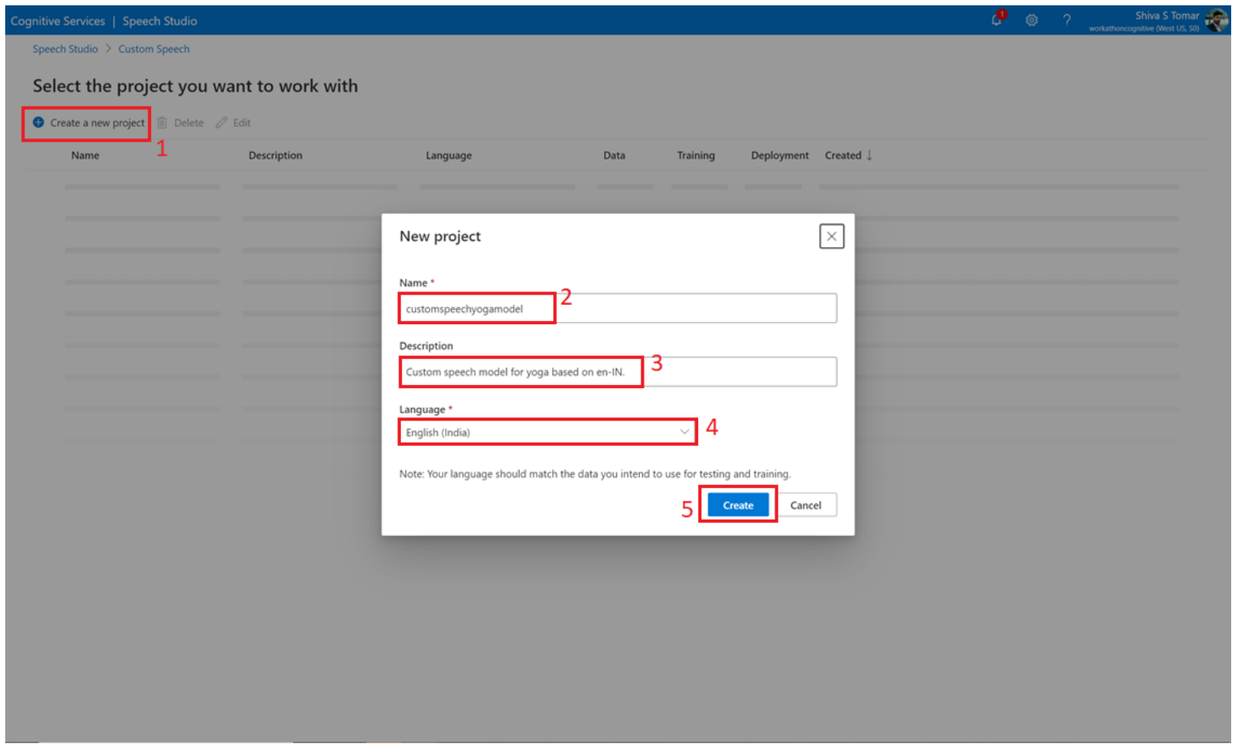
The audio files and the text file are then zipped together to be uploaded to the Speech Portal at a later step. (Step 3)

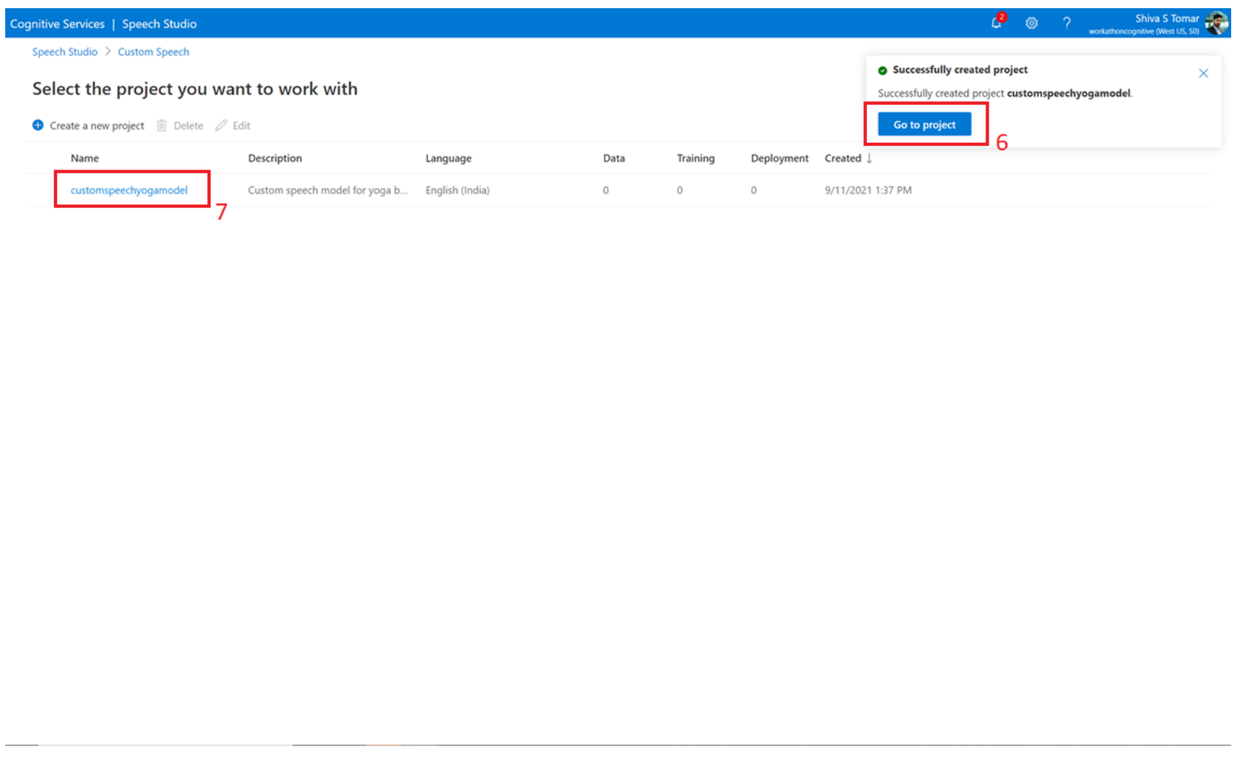
You can use the zip file from the STT Folder in Data Folder or create 1 for your own data.



Exploring Azure Speech Portal

1. Go to speech portal
2. Ensure that you are connected to the right cognitive service account. If not, click the settings gear to select the cognitive service account we created in the beginning. (Step 1,2)
3. Select Custom Speech module, as part of Speech-to-text functionality.



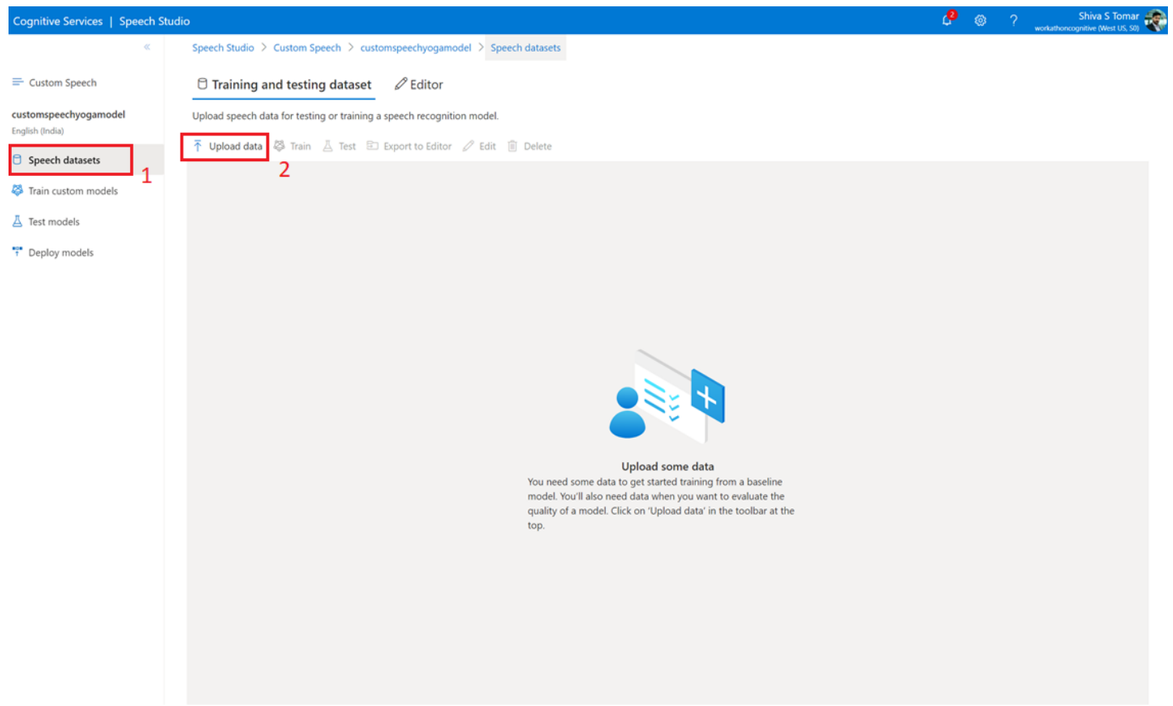


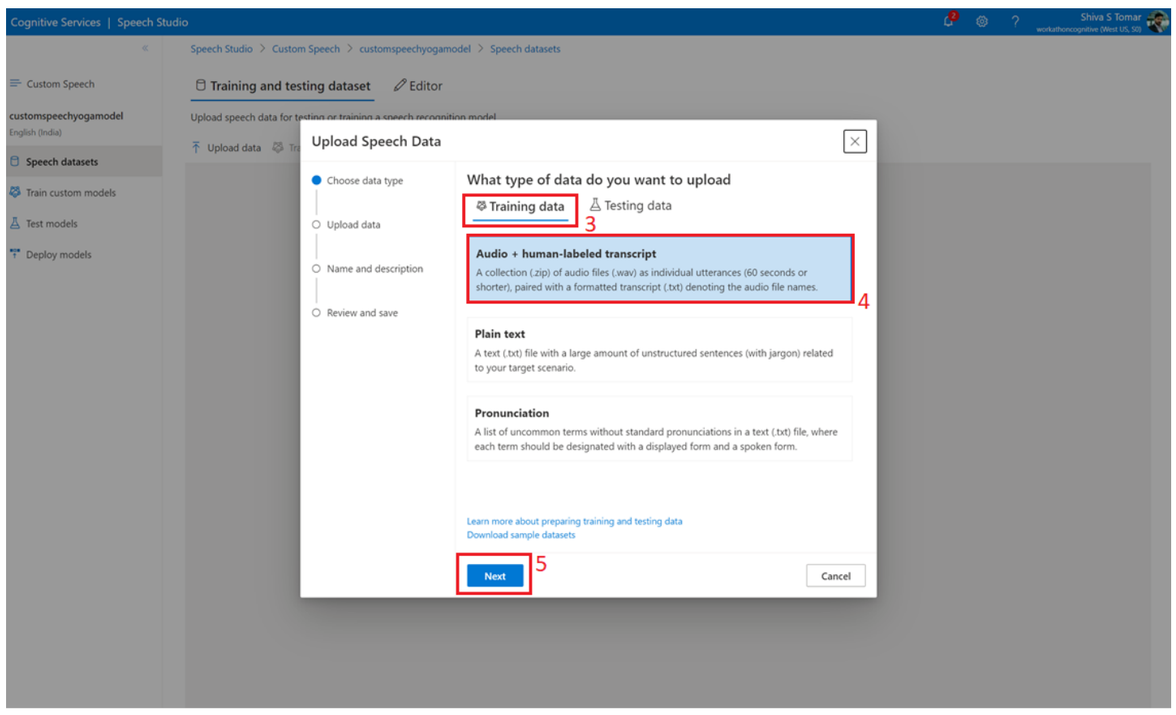
Create a new Project

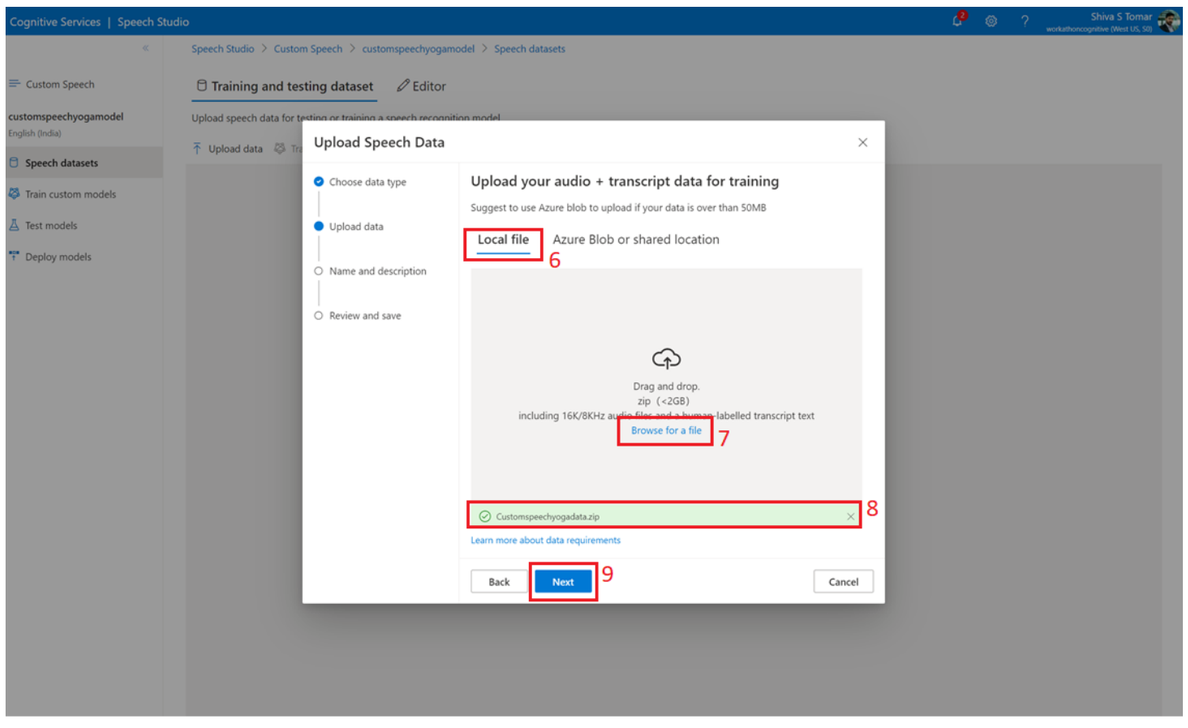
Project is like a collection for your data, custom model, model metadata, endpoints etc.
Each project can have multiple datasets and models within it.
As a best practice, you should create separate projects for each use case.

1. Select ‘Create a new project’
2. Give your project a name (customspeechyogamodel)
3. Give an appropriate description
4. Select the language your audio files are in (English India for this workshop)
5. Click ‘Create’

6. Once the project is created successfully, click ‘Go to project’
7. Click the project name to start building the model



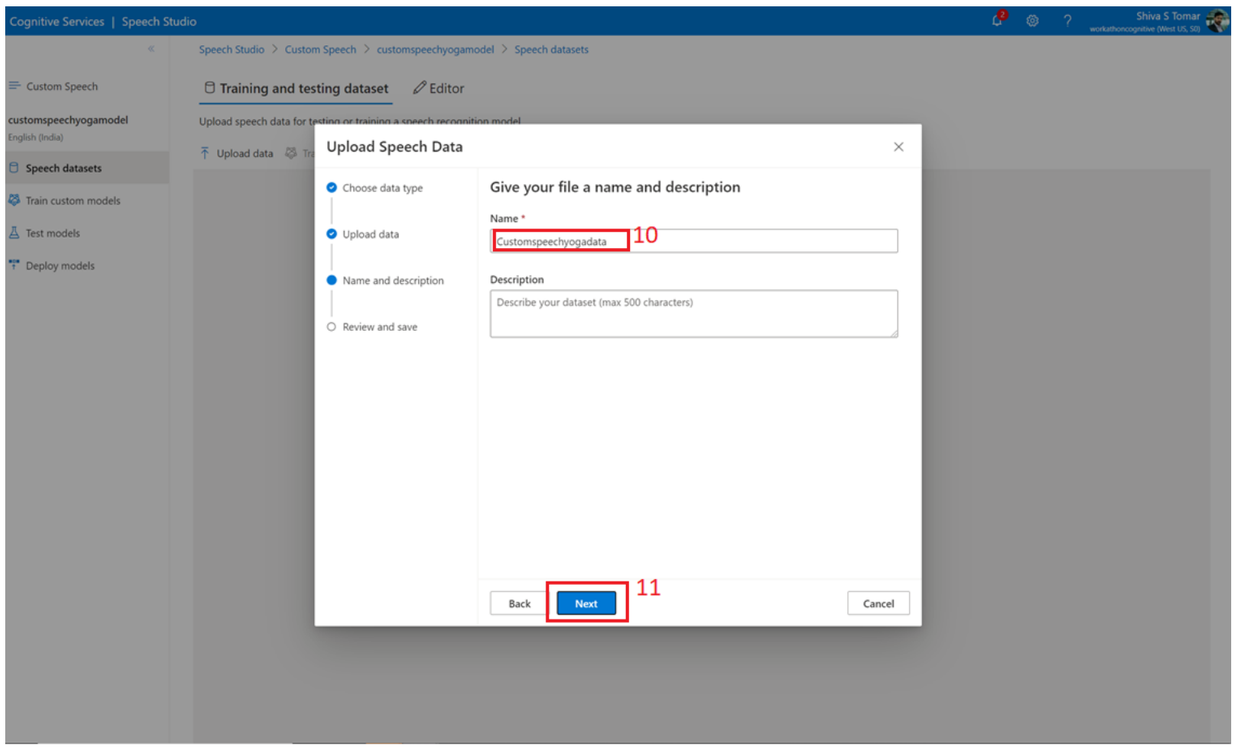


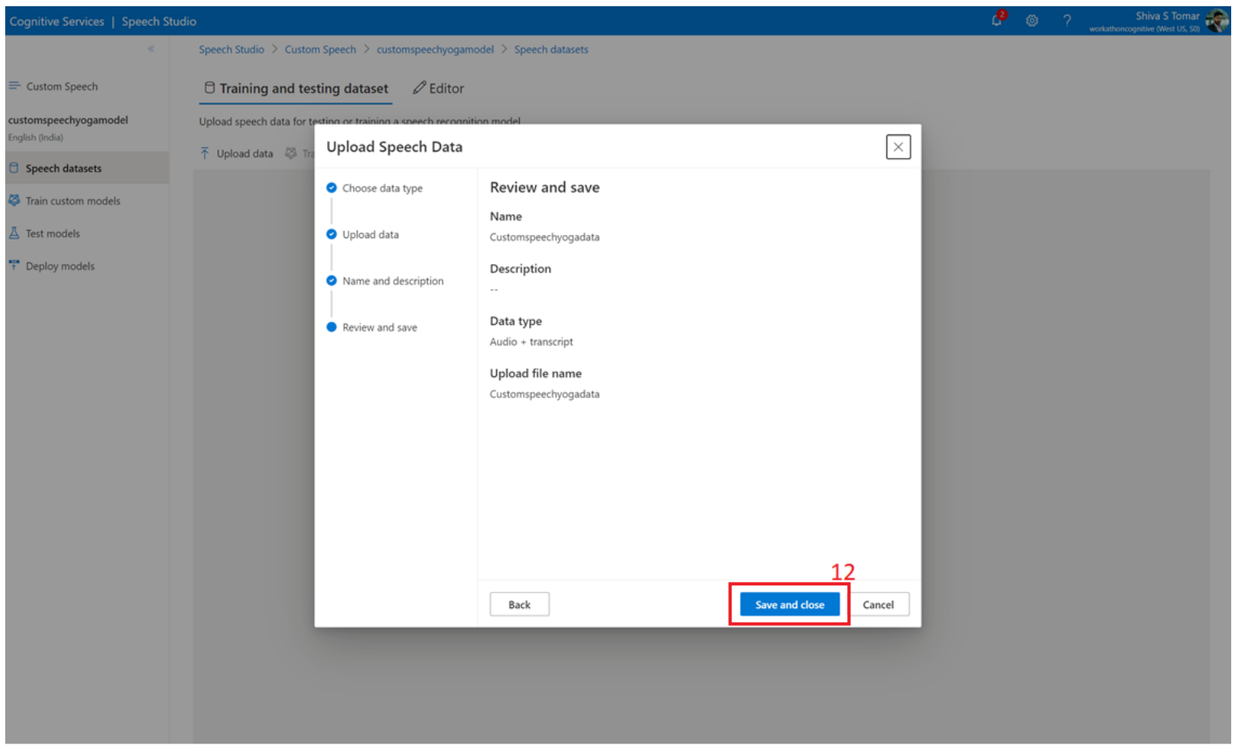


Upload a new Dataset for training the model

Dataset is a collection of audio files and their corresponding text.
You can also upload data containing just audio or text content, depending on how you want to train the model. In this use case, we will be using a dataset that contains both the audio & the corresponding text.

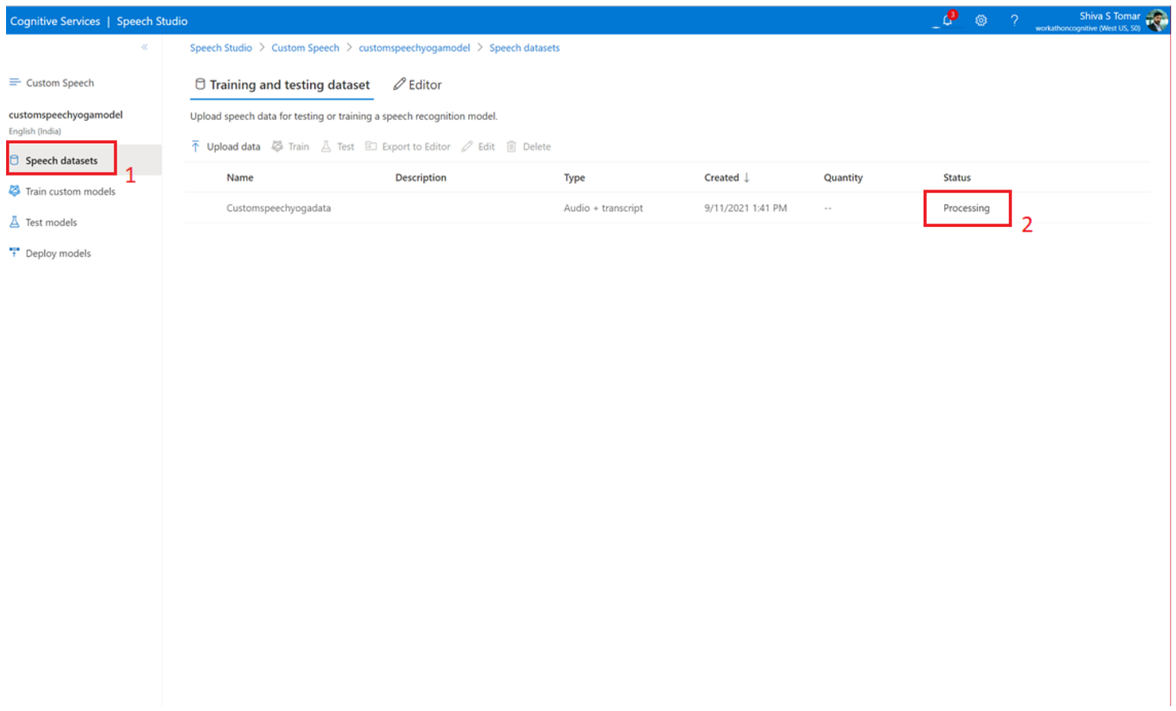
1. Make sure you are in the 'Speech datasets' tab
2. Select Upload data
3. Select Training data tab
4. Select 'Audio + Human-labelled transcript' [Explore the different options you get when you select the training data type]
5. Click Next
6. Select 'Local file'
7. In your computer, browse the zipped file that we created above
8. You will see the success message in green once the file is uploaded
9. Click Next

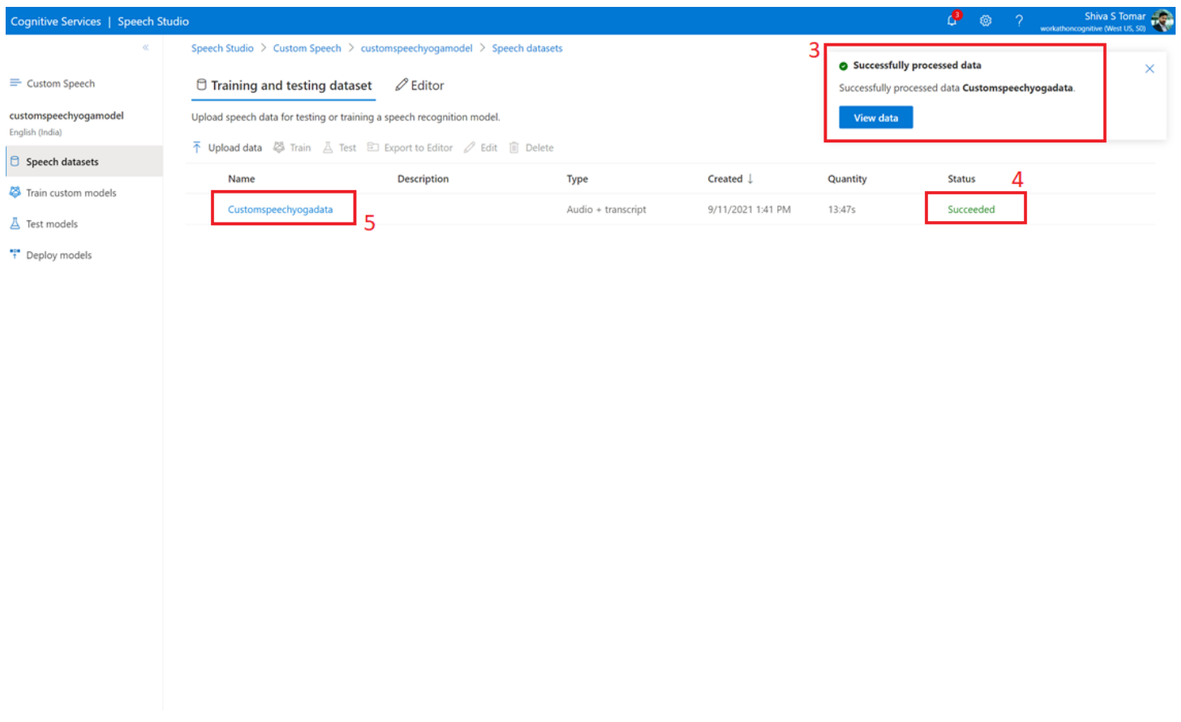


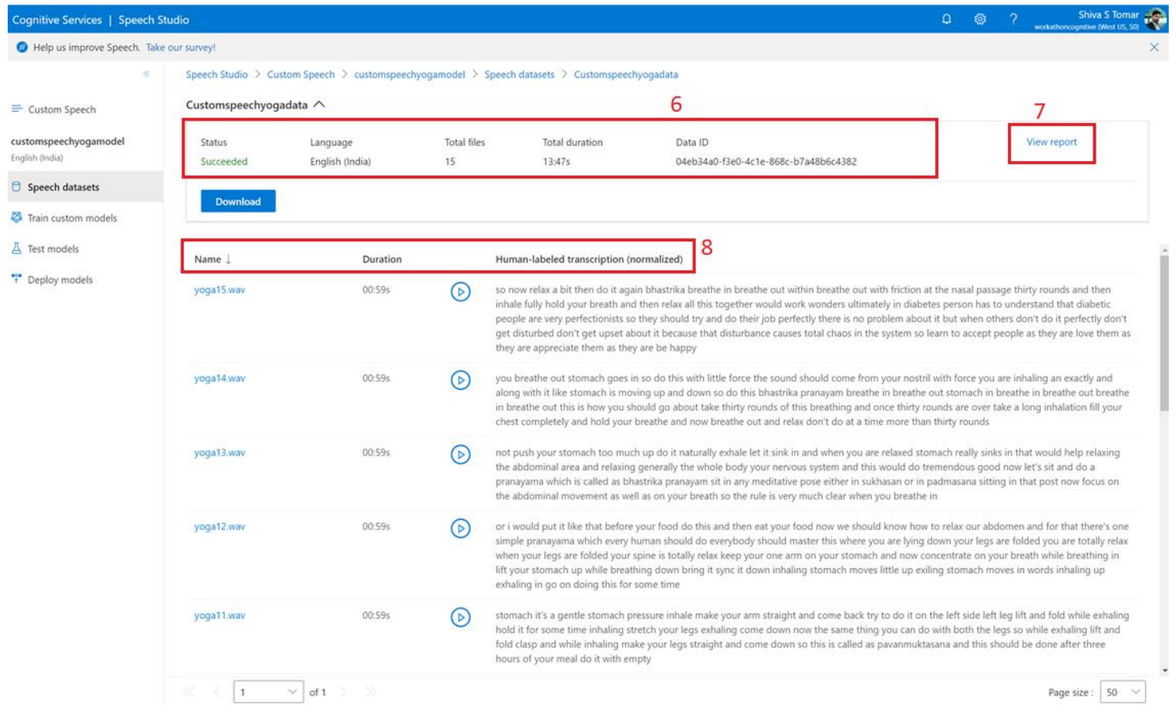


10. Give your dataset a name (Customspeechyogadata) and optionally a description
11. Click Next

12. Click Save and close

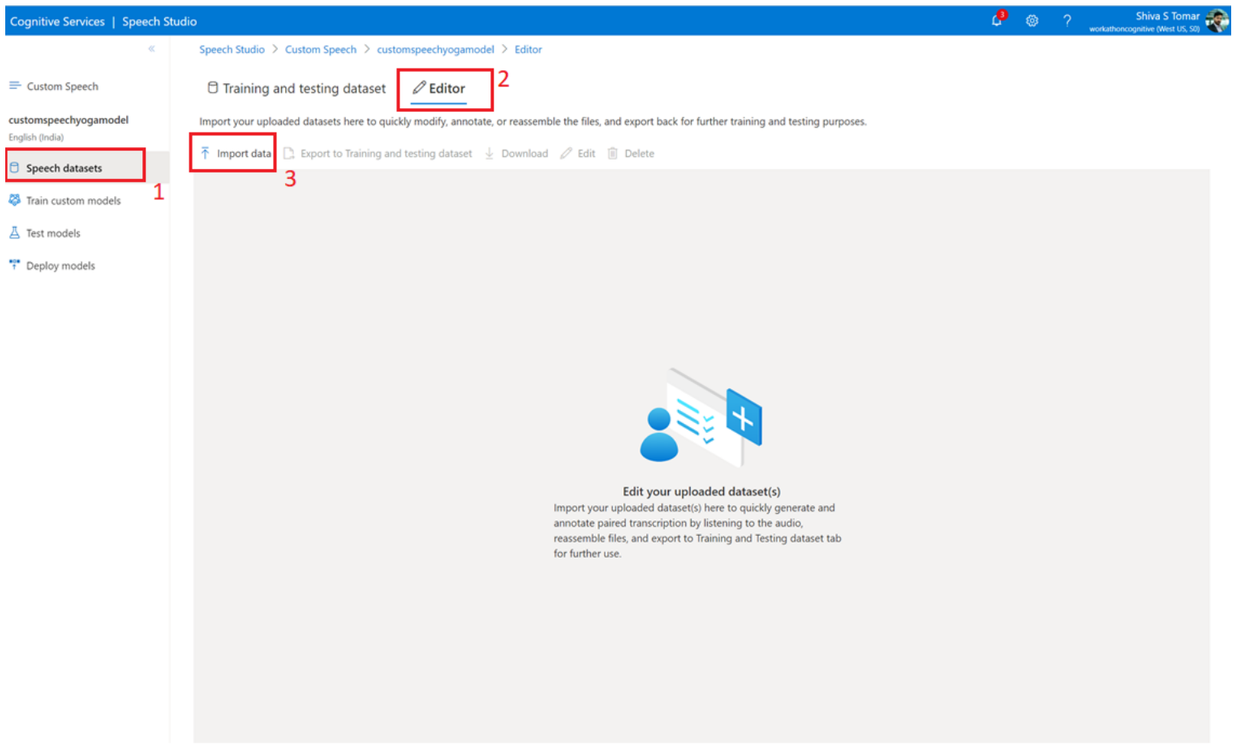






Exploring the dataset

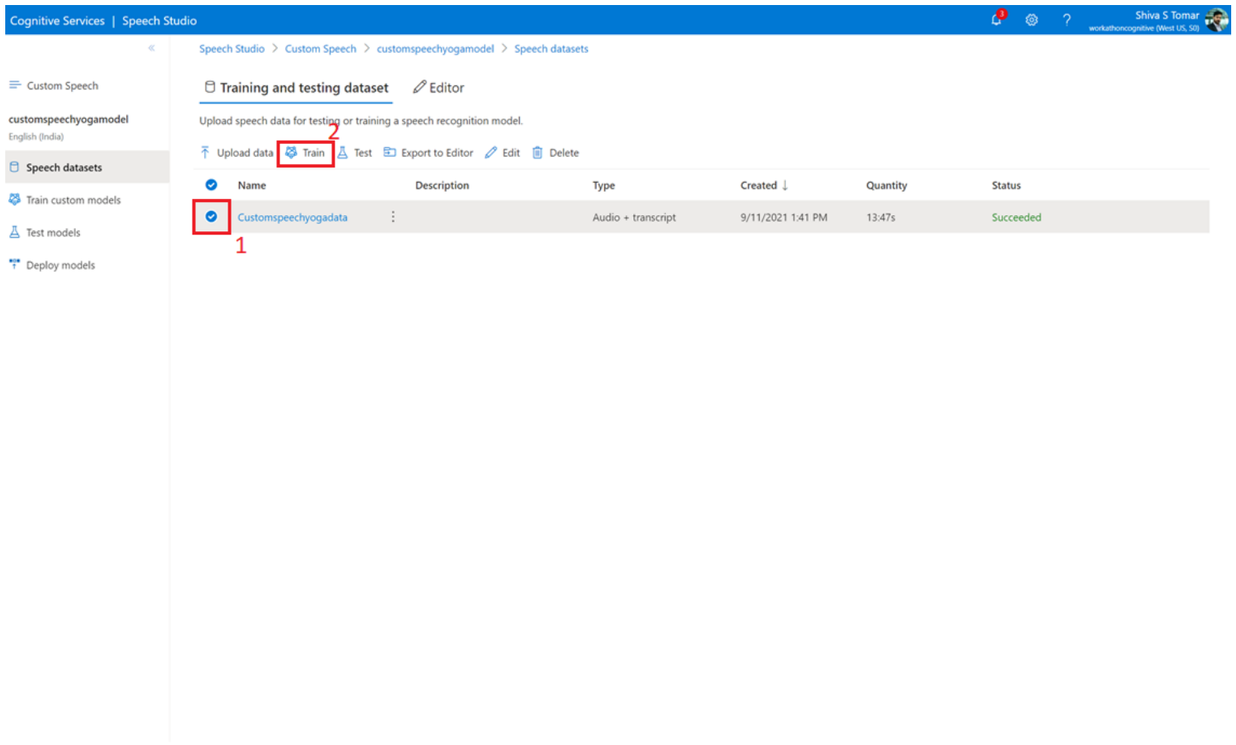
1. Navigate back to the Speech datasets tab
2. Wait while the Status of your dataset is set to Processing
3. You will receive a notification once data processing is complete
4. The Status will change to Succeeded
5. Click on the dataset name to explore the processed data
6. You can see the metadata information such as Status, Language, Total Files, Total Duration and Dataset ID
7. You can also view the normalised data for each file and play the recording if you like.
   View Report shows you the number of files uploaded successfully and the number of files that failed to process.
8. You can listen to each individual audio file you uploaded and read the transcriptions you added.



Editor Functionality

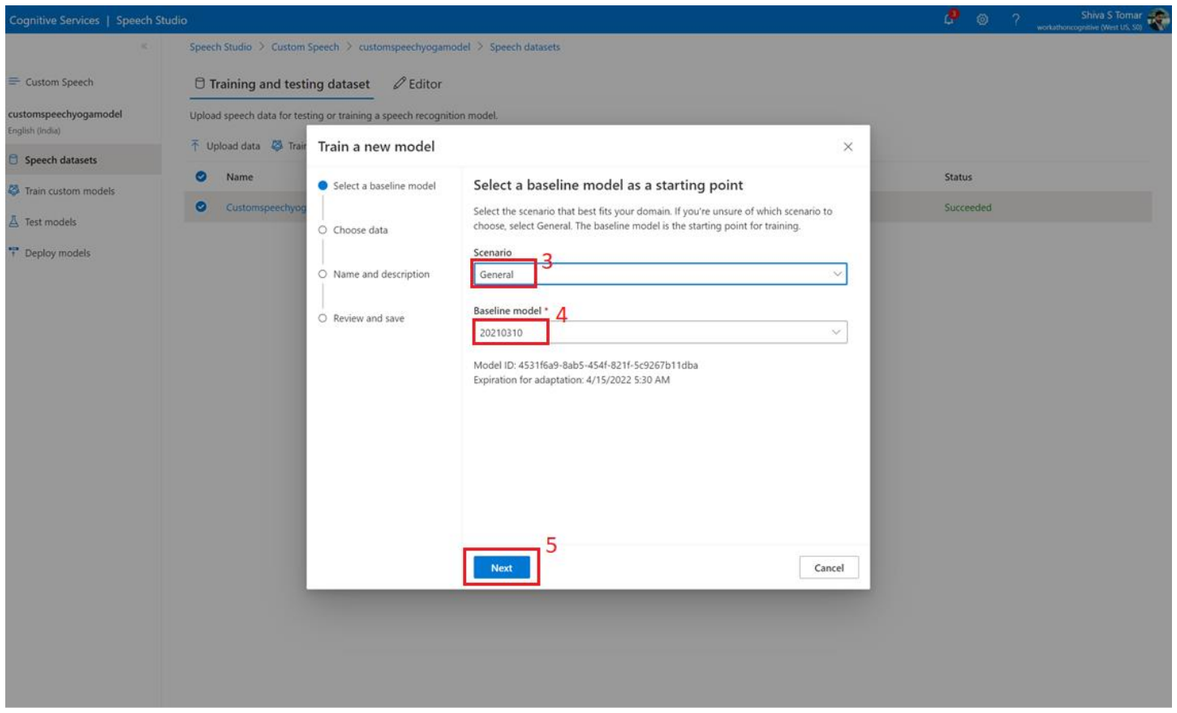
The Editor allows us to Import our uploaded datasets here to quickly modify, annotate, or reassemble the files, and export back for further training and testing purposes.

In case you want to edit the text for any of the files or have files with missing text, you can just upload the audio files here and the editor will generate the text transcript which you can edit for training purpose.

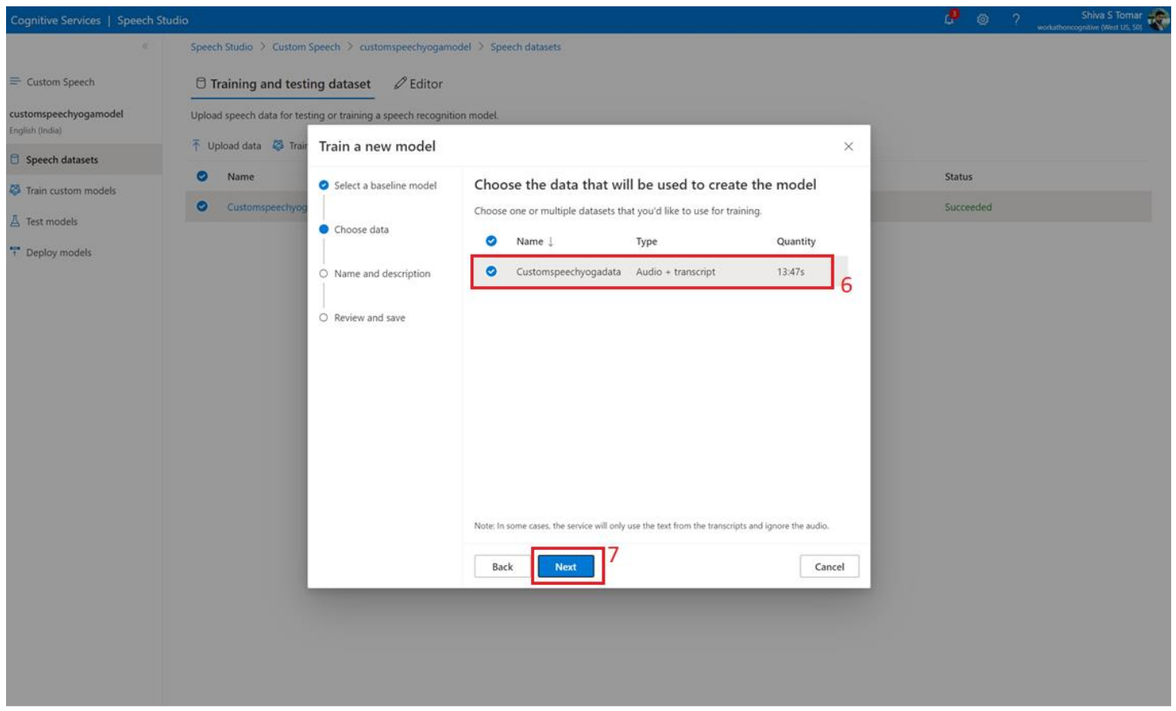


Training the model

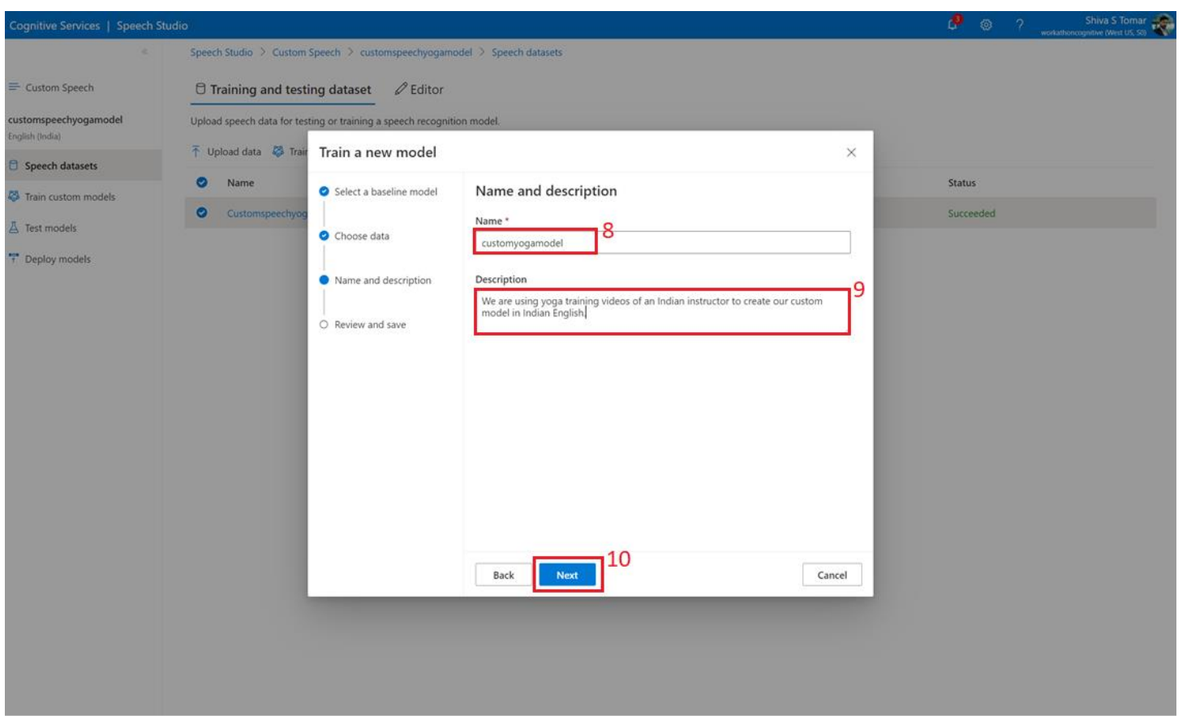
1. In the Speech datasets tab, select the training data you uploaded earlier
2. Select Train



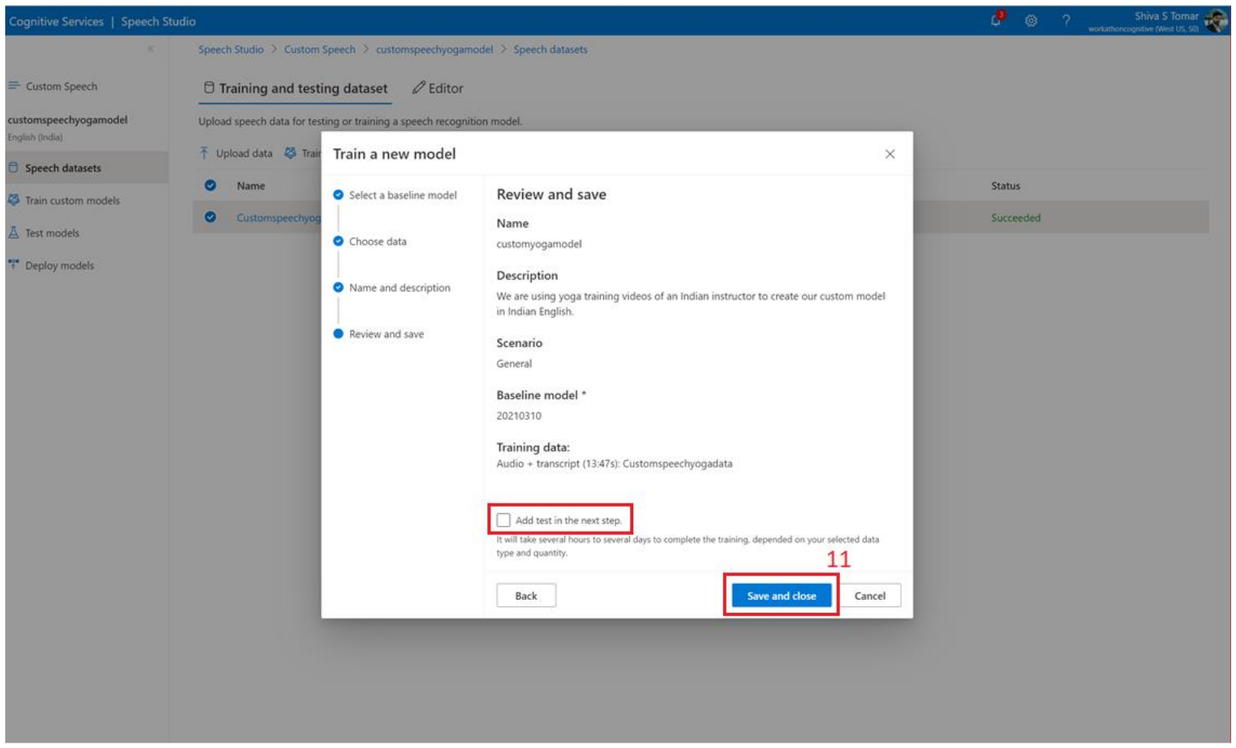
3. Select the Scenario (General)
4. Select any baseline model. (Baseline models are the pre-built Speech-to-text models available out of the box. The custom models are built on top of the baseline models. You can try training different models using different baseline models)
5. Click Next



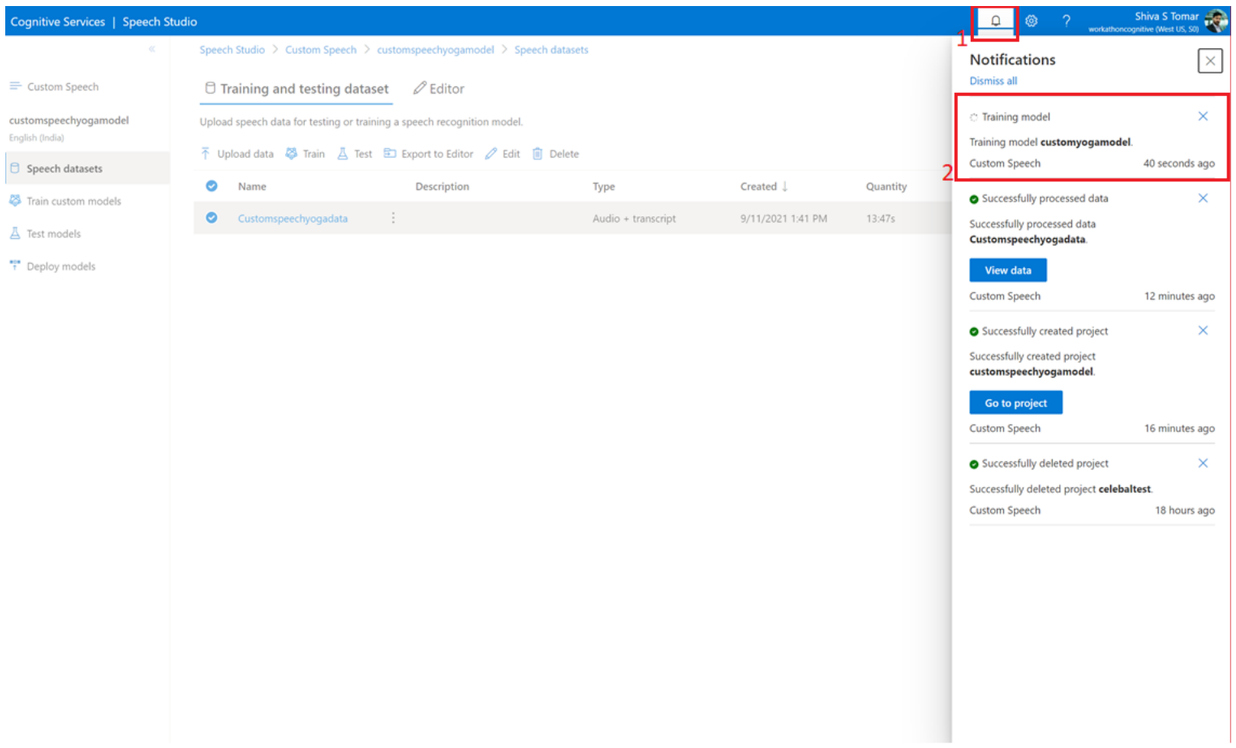
6. In the Choose data tab, Select the dataset we uploaded earlier
7. Click Next



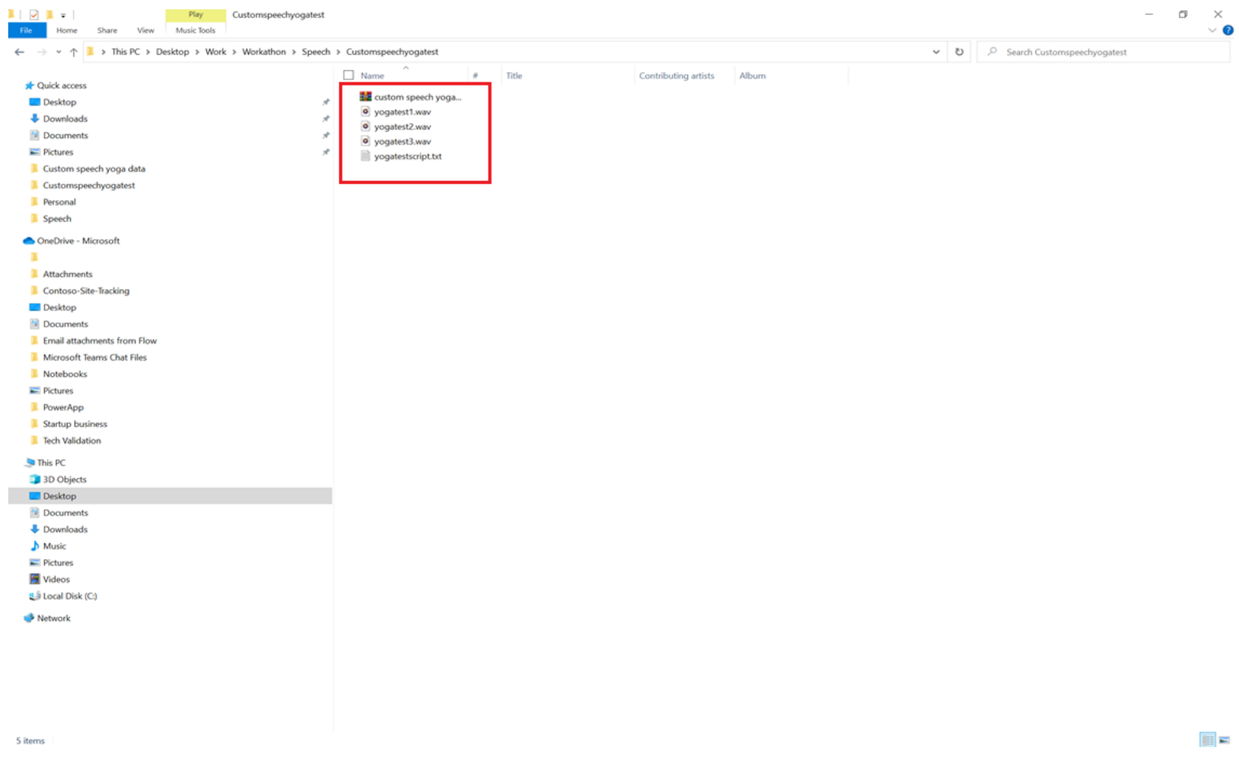
8. Give your custom model a name (customyogamodel)
9. Optionally, give your model a description
10. Click Next



11. Select Save and Close
12. You can optionally check ‘Add test in next step’ checkbox, this will additionally add a test step to test the accuracy of the model that is trained. However, we will be doing this separately, so are leaving the box unchecked

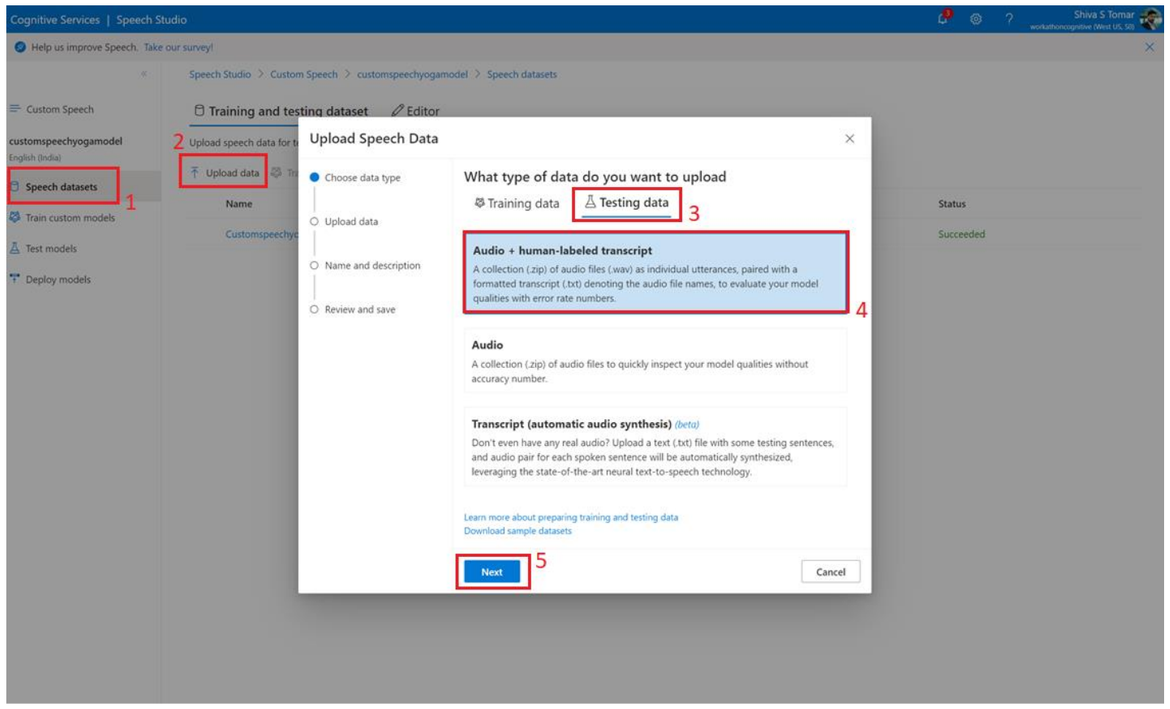


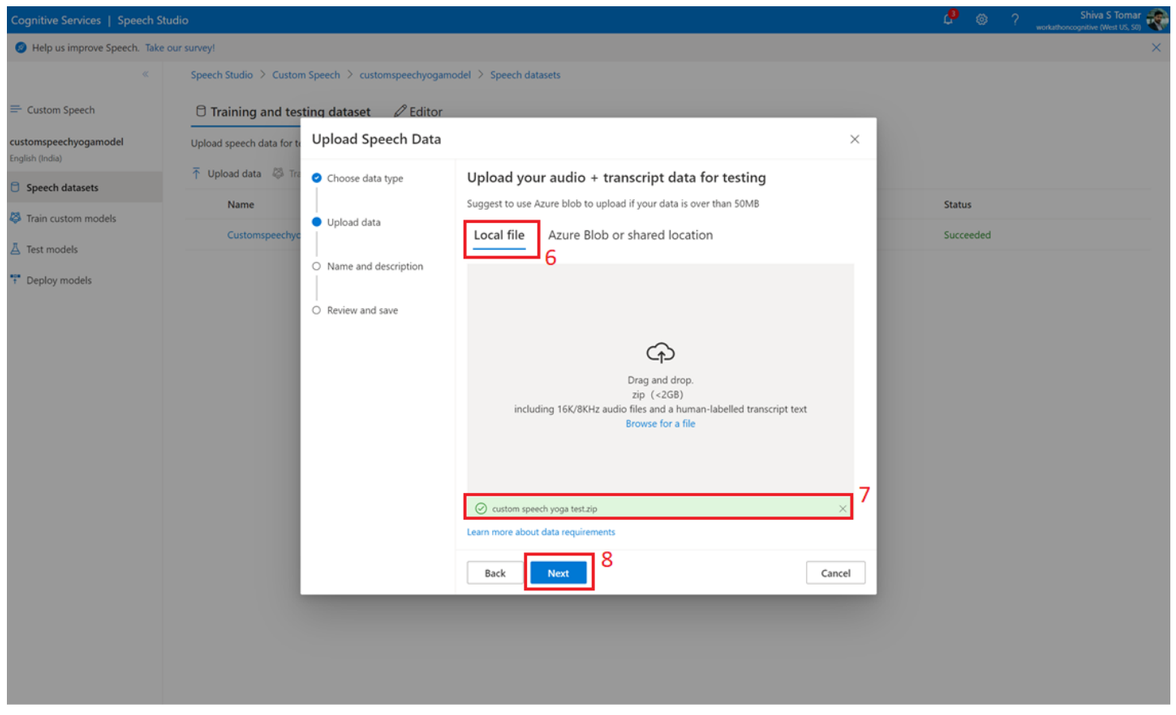
1. Select the notification bar
2. Wait while the model is still training. Meanwhile, proceed to the next steps, to prepare the testing data

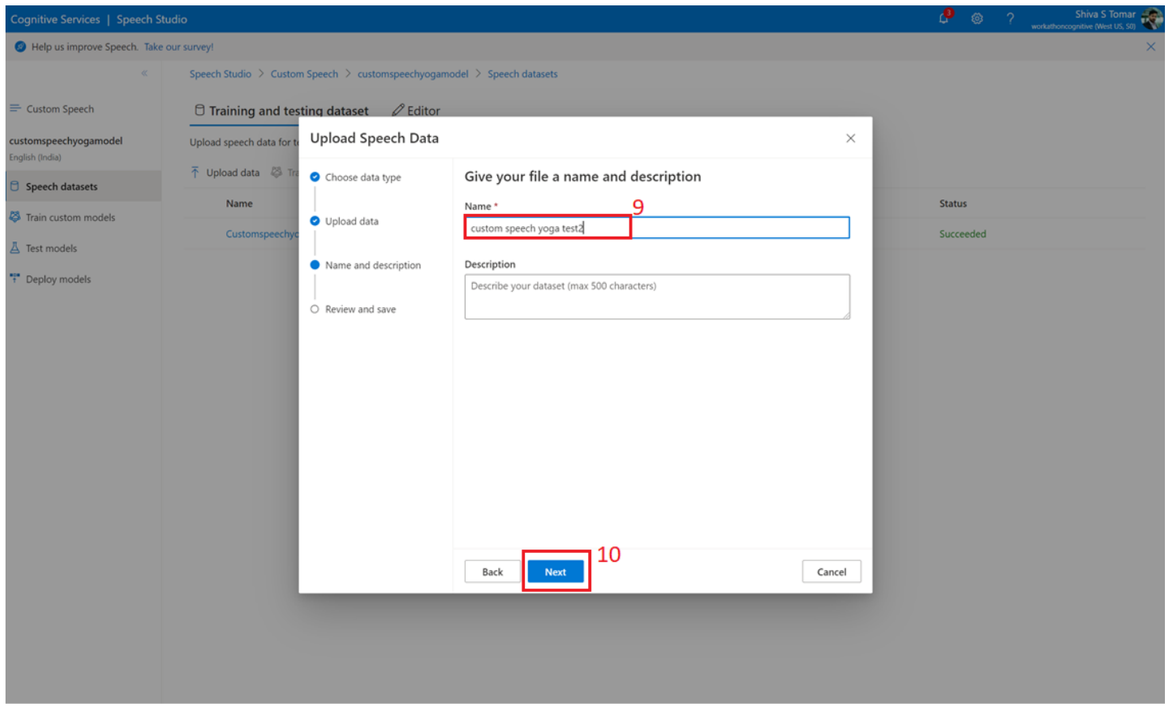


Prepare testing data

1. Like we created the training dataset, similarly, prepare the test set.
2. Make sure each audio file is in .wav format and is not more than 60s in length.
3. Each line in the text file should contain the name of the file and the corresponding text, separated by a tab.
4. Zip the audio and text files together.
5. Make sure the test audio files are related to the same use case, in case you use your own data.





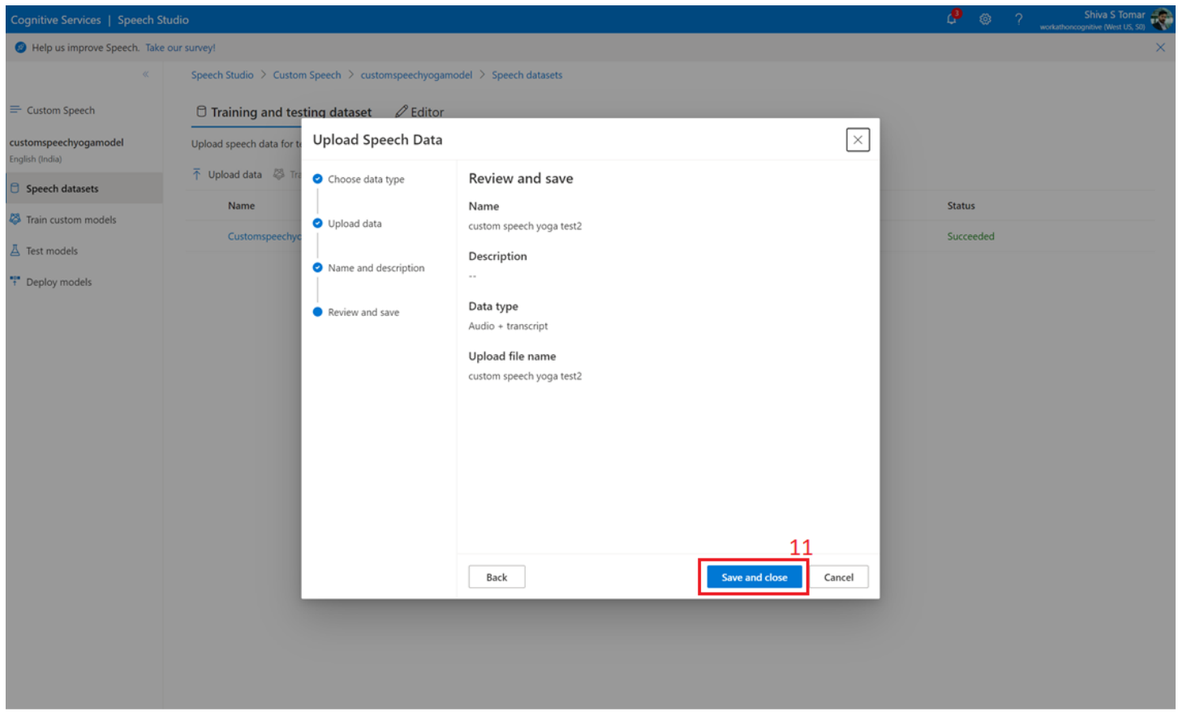


<u>Create a new Dataset for testing the model</u>

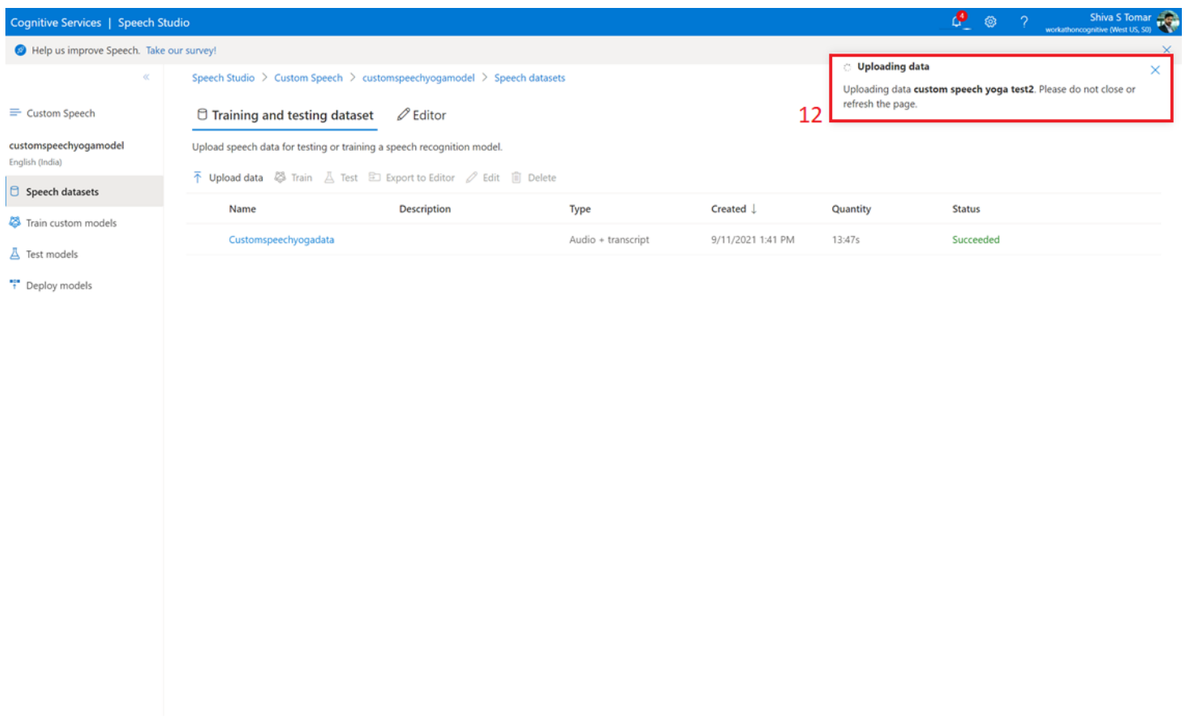
1. Make sure you are in the 'Speech datasets' tab
2. Select Upload data
3. Select Testing data tab
4. Select 'Audio + Human-labelled transcript'
   [Using 'Audio + Human-labelled transcript' allows you to quantitively measure how your model performed, since it can compare the transcriptions it generated with the actual transcription and find errors in terms of deletions, insertions & substitutions in the output text.]
5. Click Next

6. Select 'Local file' and in your computer, browse the zipped file that we created above
7. You will see the success message in green once the file is uploaded
8. Click Next

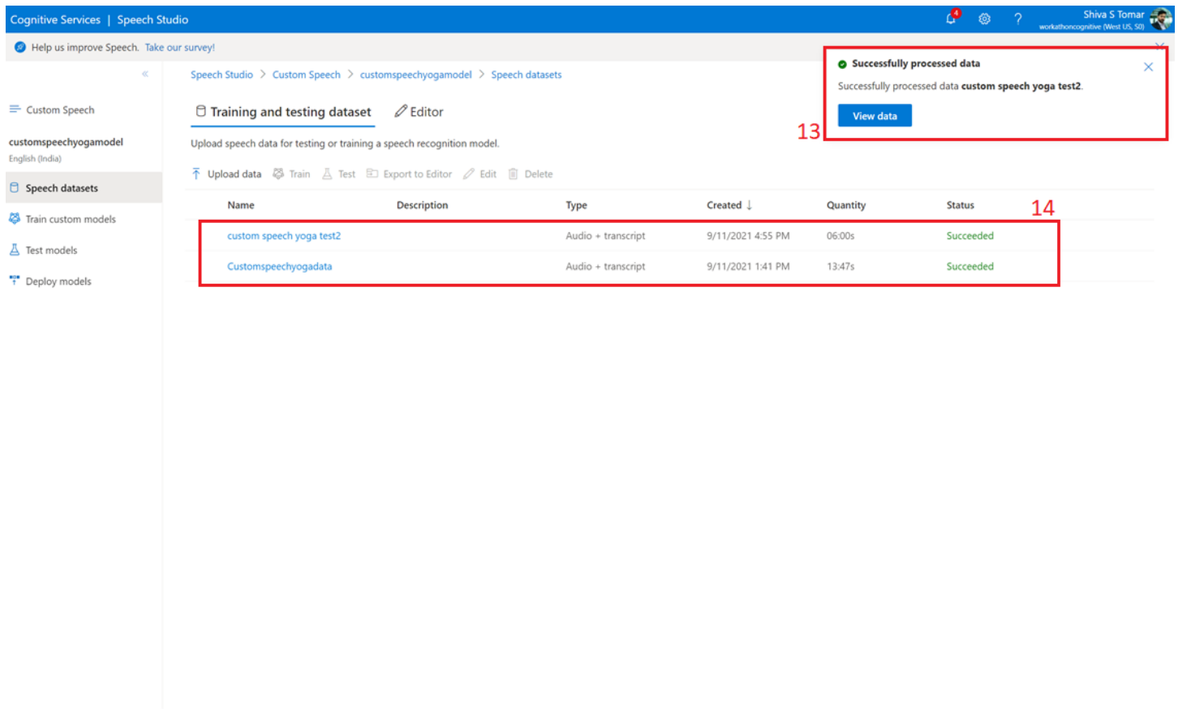
9. Give your dataset a name and optionally a description
10. Click Next



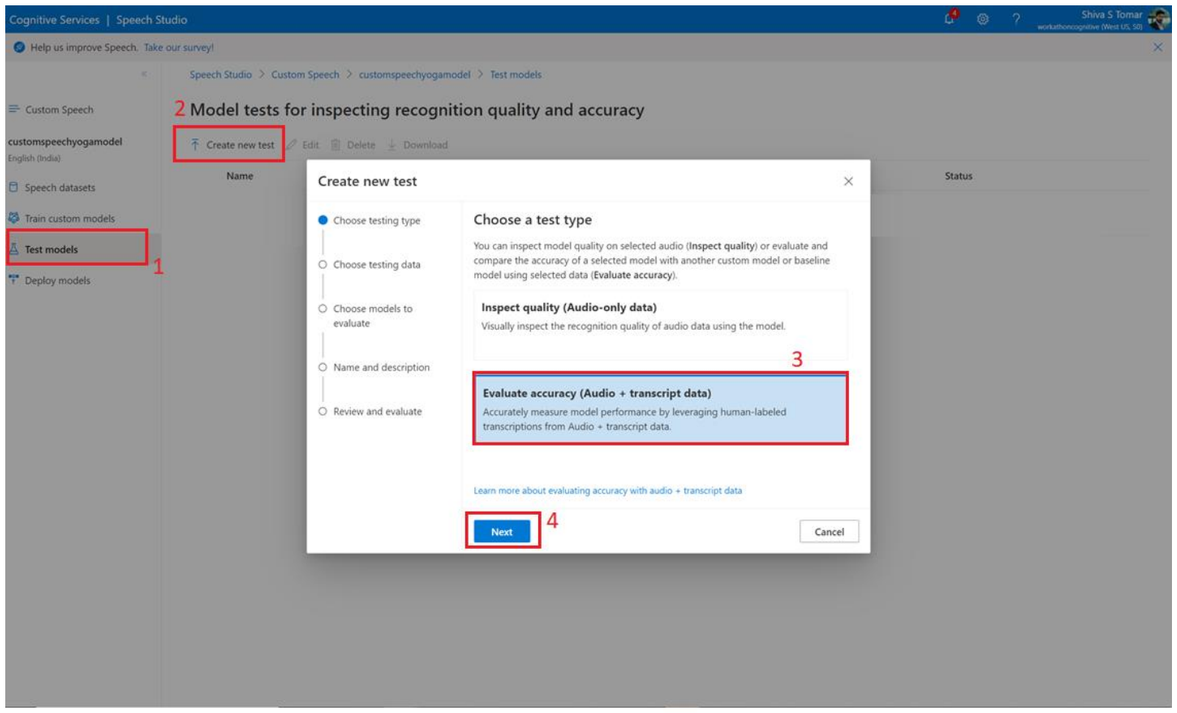
11. Click Save and close

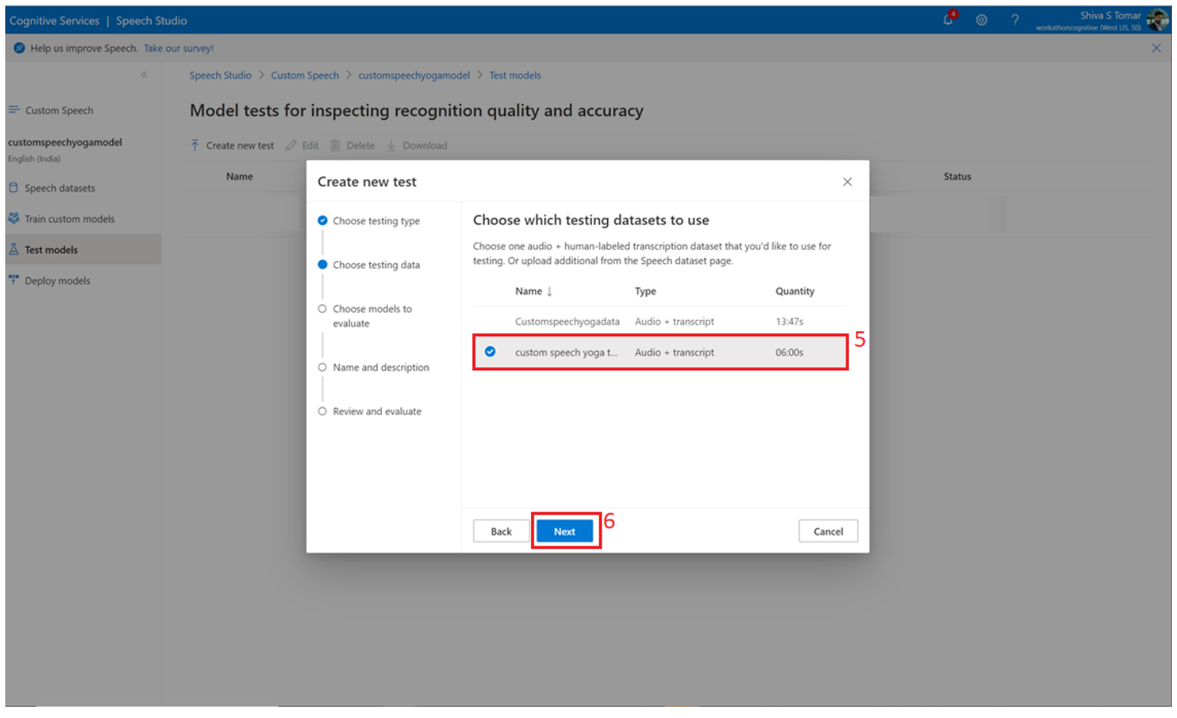


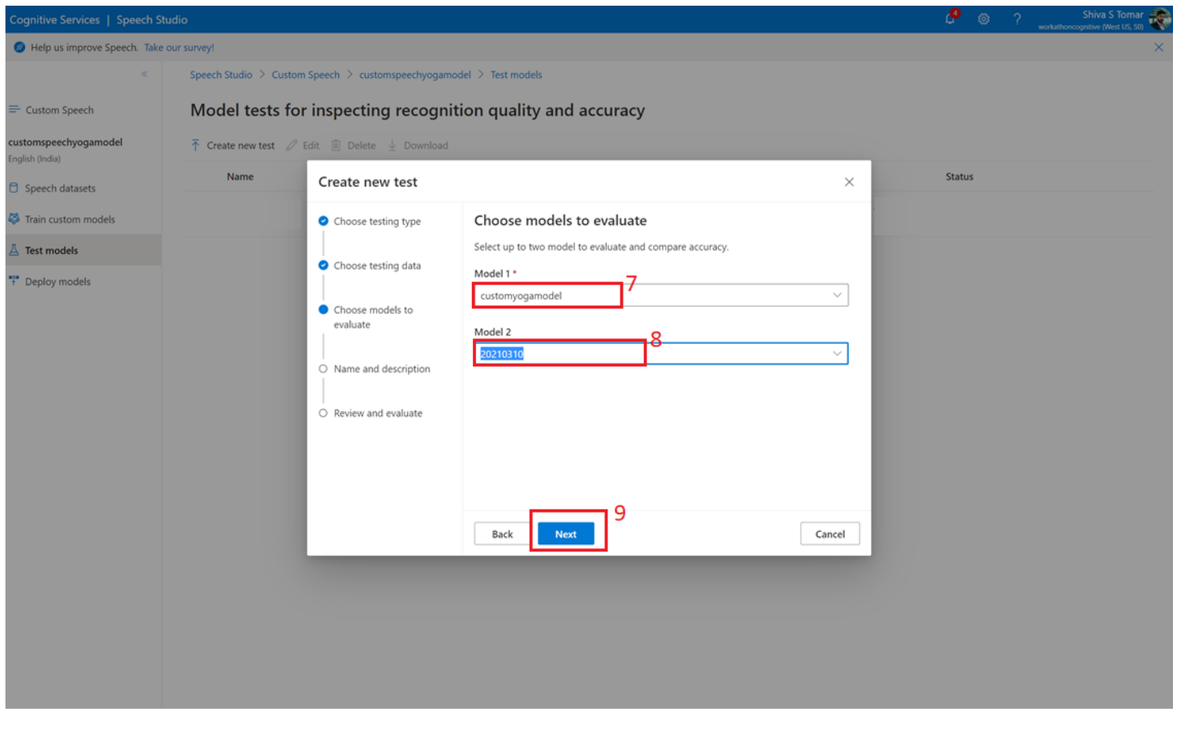
12. Wait while the testing data is uploaded and processed



13. Once data is processed, you will get a success notification
14. You now have 2 datasets uploaded (one for training, one for testing) and should have a view as highlighted in step 14

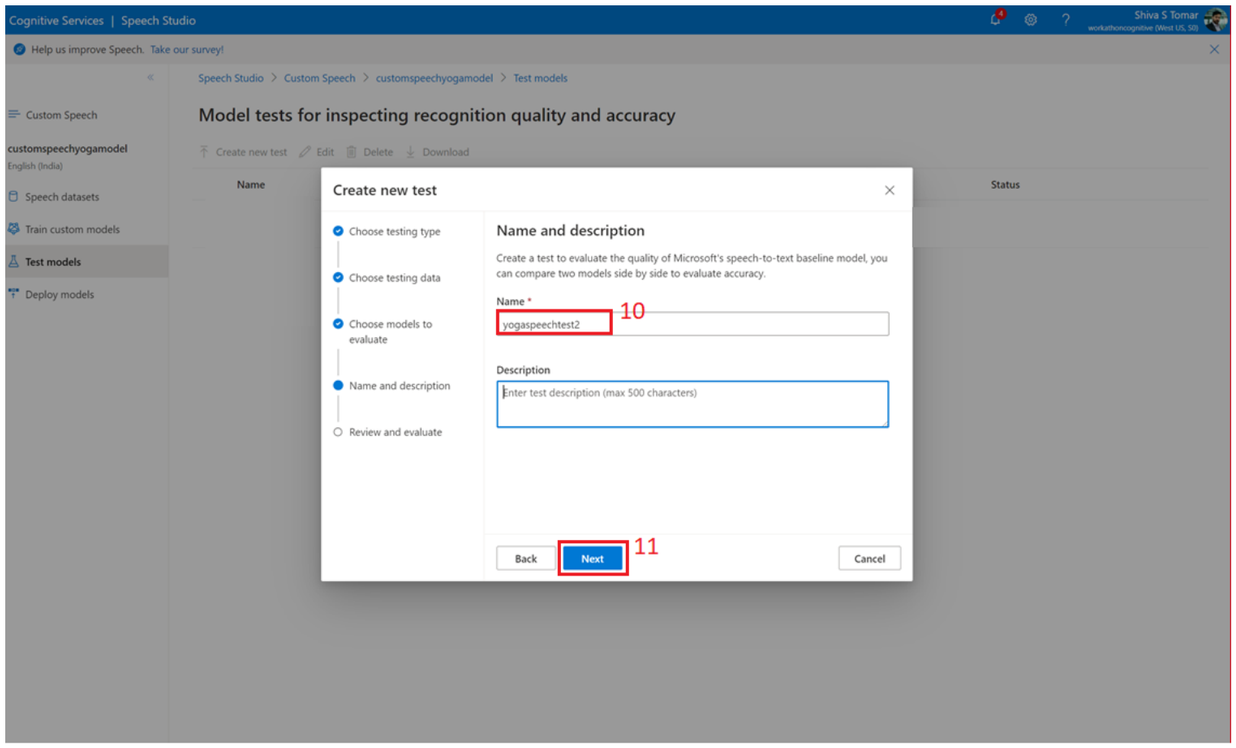




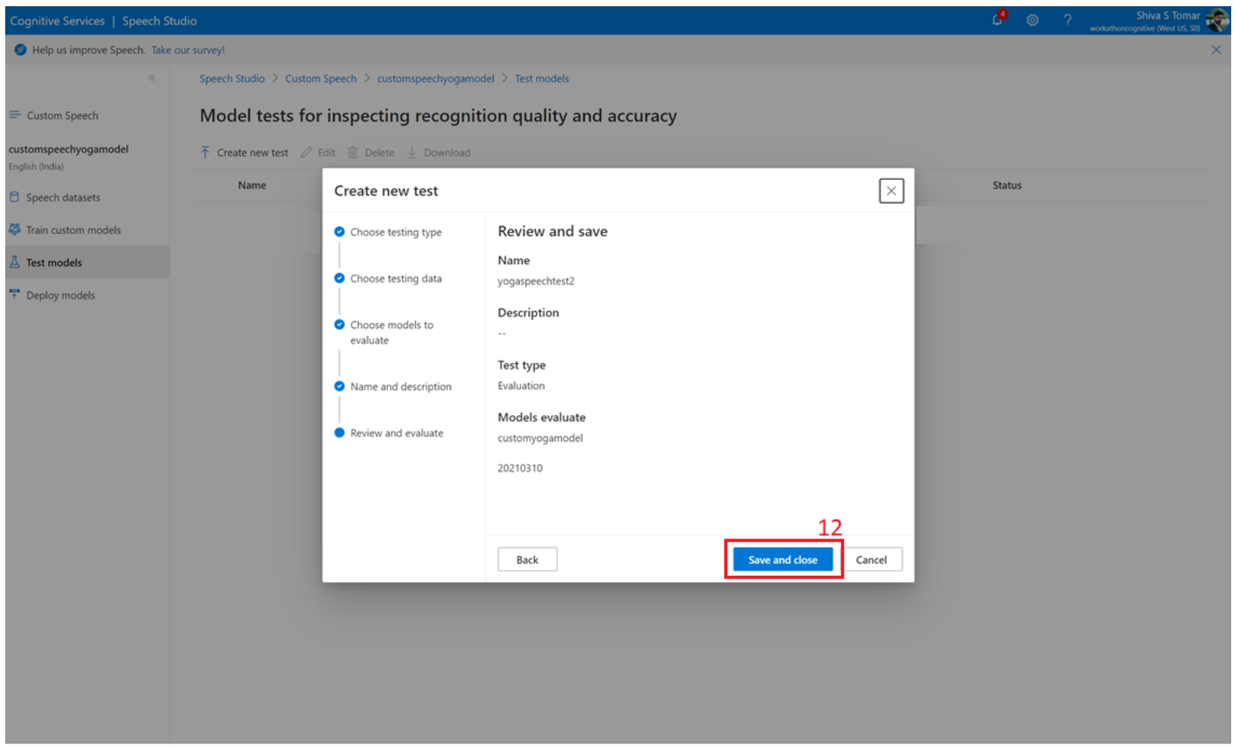


Create a new test

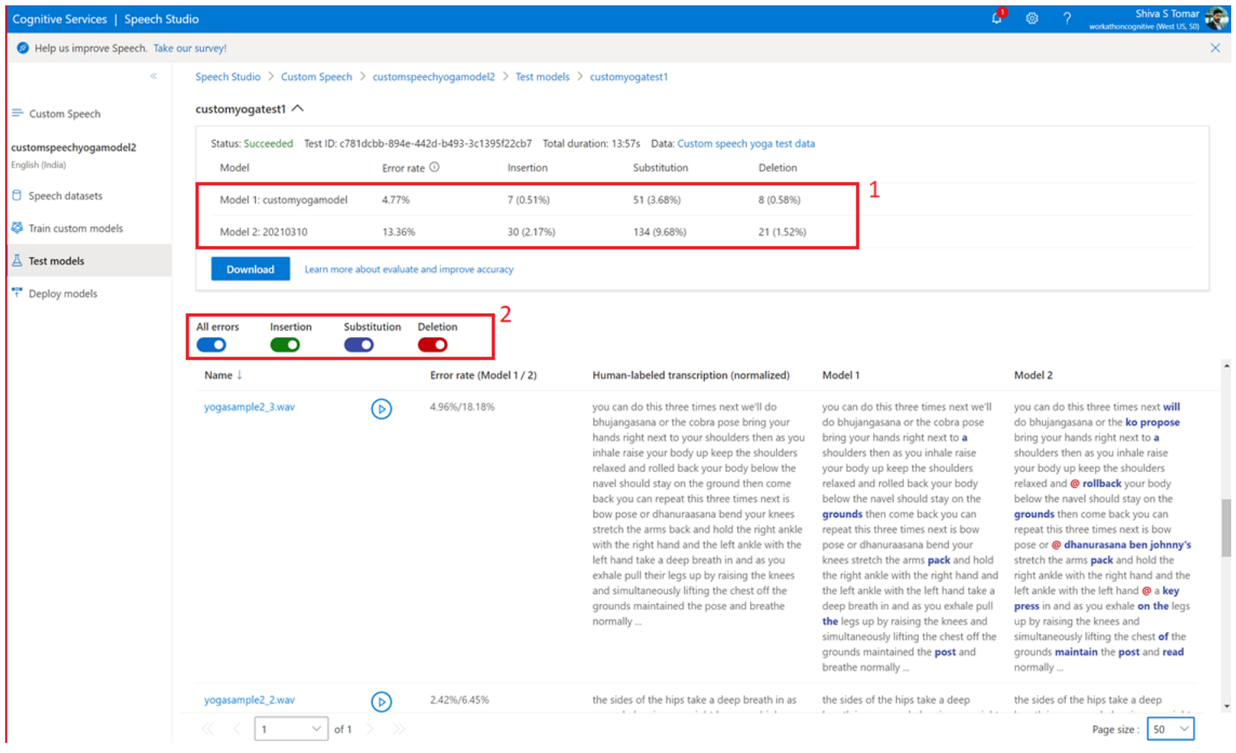
1. Select ‘Test Models’ tab
2. Click ‘Create new test’
3. In choose testing type, select Evaluate accuracy type. (If you don’t have transcripts available for the testing data and are planning to upload a zipped file containing only Audio data, select Inspect quality option. However, if you test the model this way, you will have to manually go through the transcript generated by the model and evaluate manually. This way you can just obtain a qualitative idea of how the model performed, not a quantitative idea.
   Selecting the Evaluate accuracy option provides a detailed quantitative analysis of how the model performed)
4. Click Next
5. Select the test dataset you had uploaded earlier
6. Click Next
7. Choose the custom model you want to evaulate (customyogamodel)
8. Choose a baseline model as Model 2(Baseline models are the pre-built Speech-to-text models available out of the box. These help in benchmarking as to how your custom model performs versus how the pre-built Azure model would have performed on your data. This gives you an idea whether the pre-built models are a good fit for you).
   In case you have trained more than 1 model, you can also choose custom models for both Model 1 and 2.
   If you choose baseline models for the Model 1 and 2, your test data will be evaluated against pre-built models only. This can act as a baseline on how well pre-built models work for your use case and then you can take a call whether you want to train a custom model or not.
9. Click Next



10. Give your test a name and optionally, give your test a description
11. Click Next



12. Select Save and Close



Observe Test Results

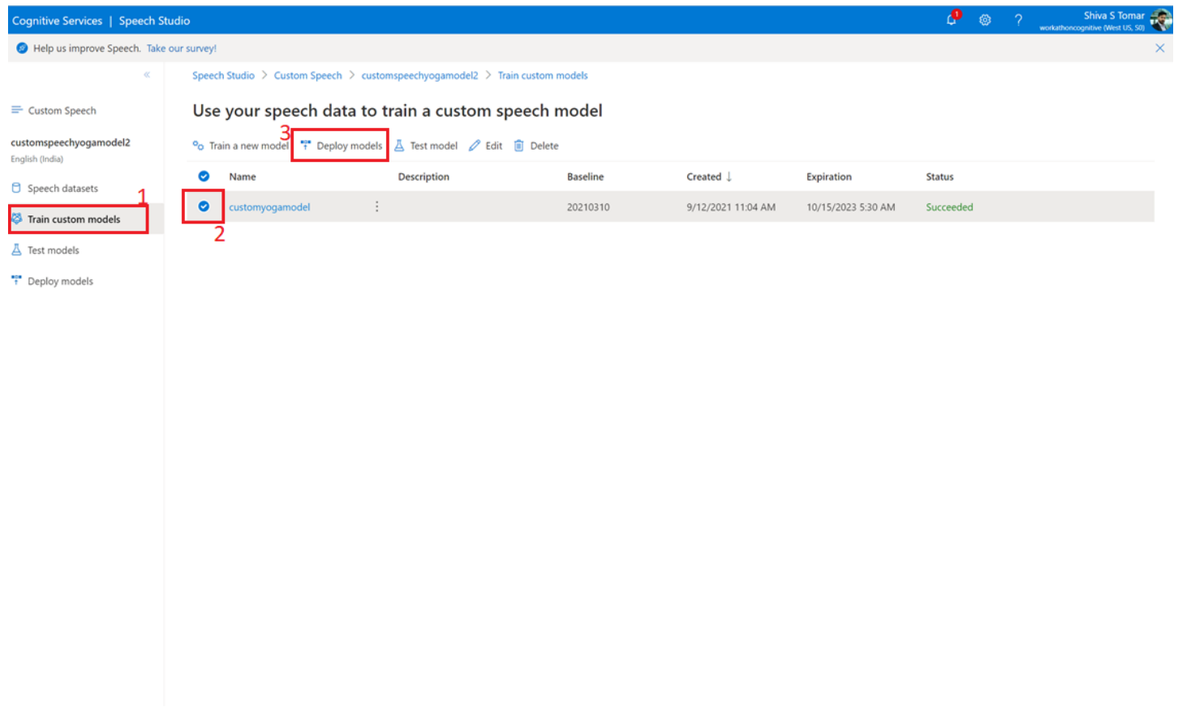
1. As shown in step 1, view the metrics (Overall Error rate, Insertion, Substitution & Deletion for Model 1 (custom Model) and Model 2 (Pre-built model).
   Observe how with just few minutes of audio, the Errors have gone down drastically. In a production scenario, when we train on a larger dataset, we will receiver an even lower error rate. This gives both absolute values and percentage errors.

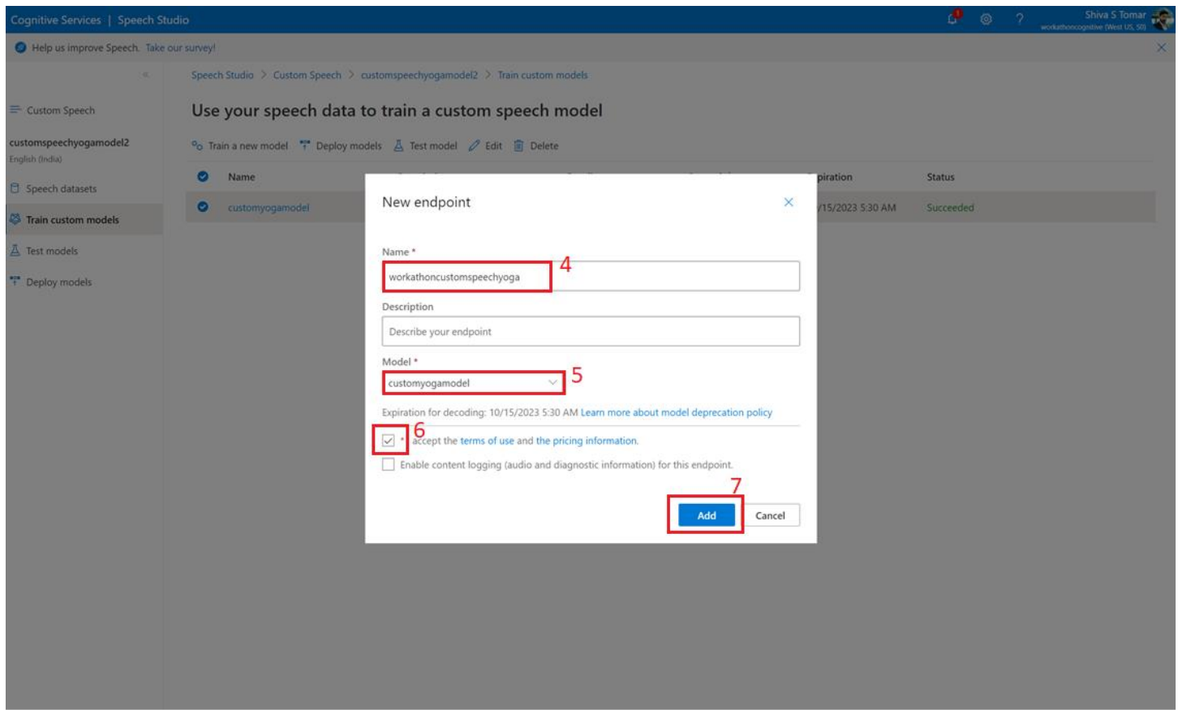
2. For each of the files we uploaded, we can view the results individually as well. This shows the name of the file, error rate for the 2 models at the file level, normalised Human-labelled transcript (the one we uploaded as part of test data, Transcript generated by Model 1 and Model 2.

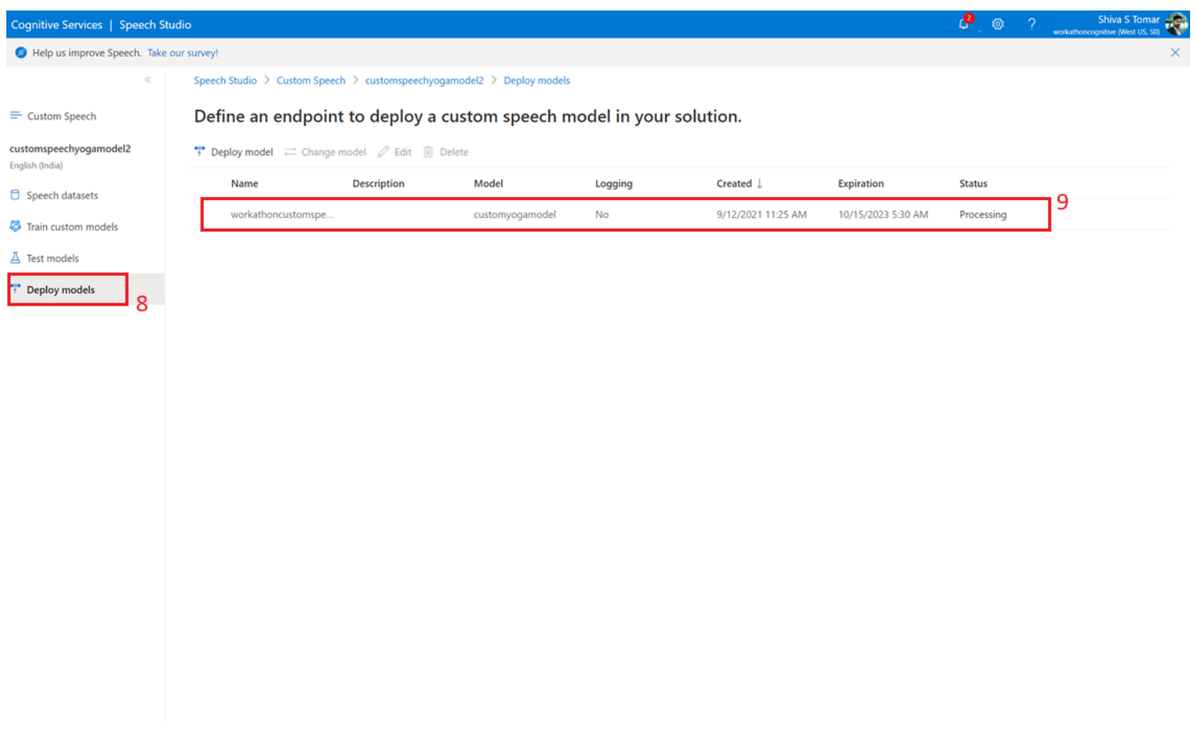
3. Basis the colour coding, you can also view the Insertion, Substitution & Deletion for each individual file in both the Models.

4. You can also toggle off 1 or more of the error types, to focus on each one individually. This analysis can help you decide what kind of data to upload, to further tune your model.

5. Observe how, for our data, the custom model (Model 1) has lesser errors than the pre-built model (Model 2) for yogasample2_3.wav file. Similarly explore for other files.







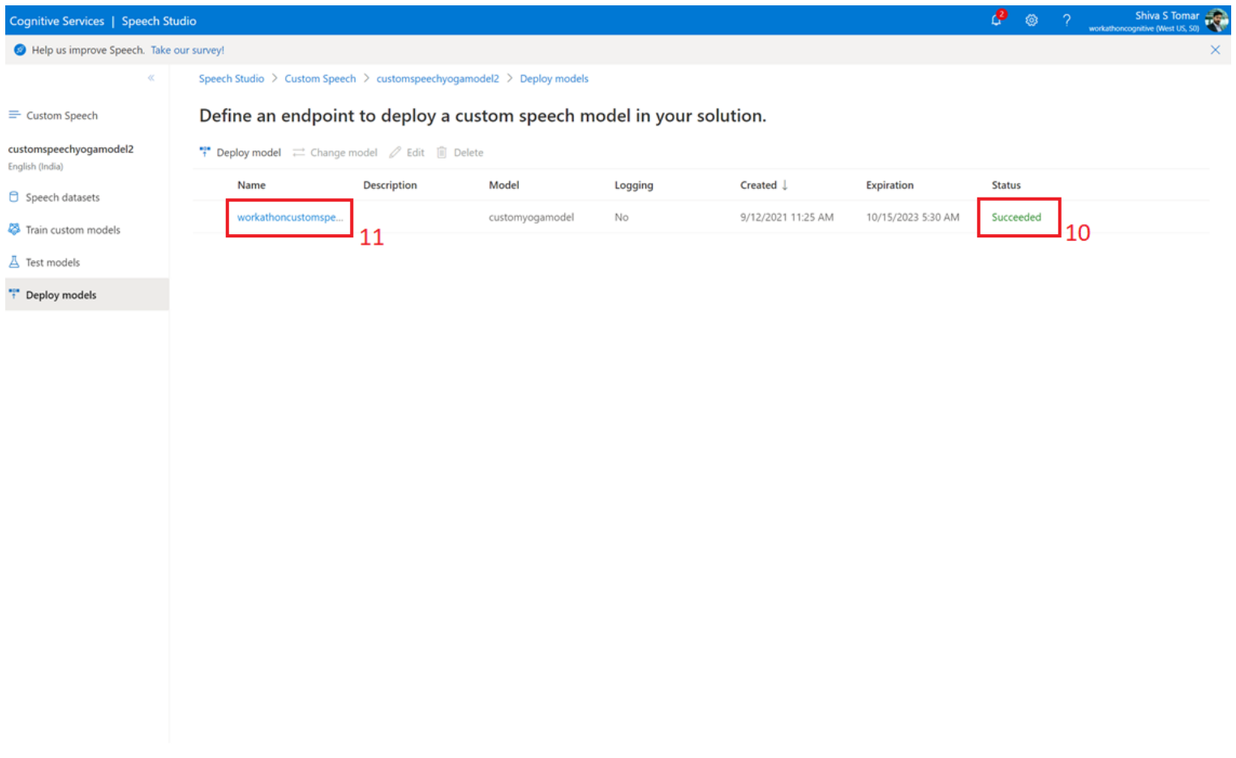
Deploy the model

Once you have trained and tuned the mode and are fine with the accuracy you're getting, you can deploy the model. Deploying the model will create a custom ID and endpoint for you, so that you can leverage this in a production scenario for processing real time or batch data.

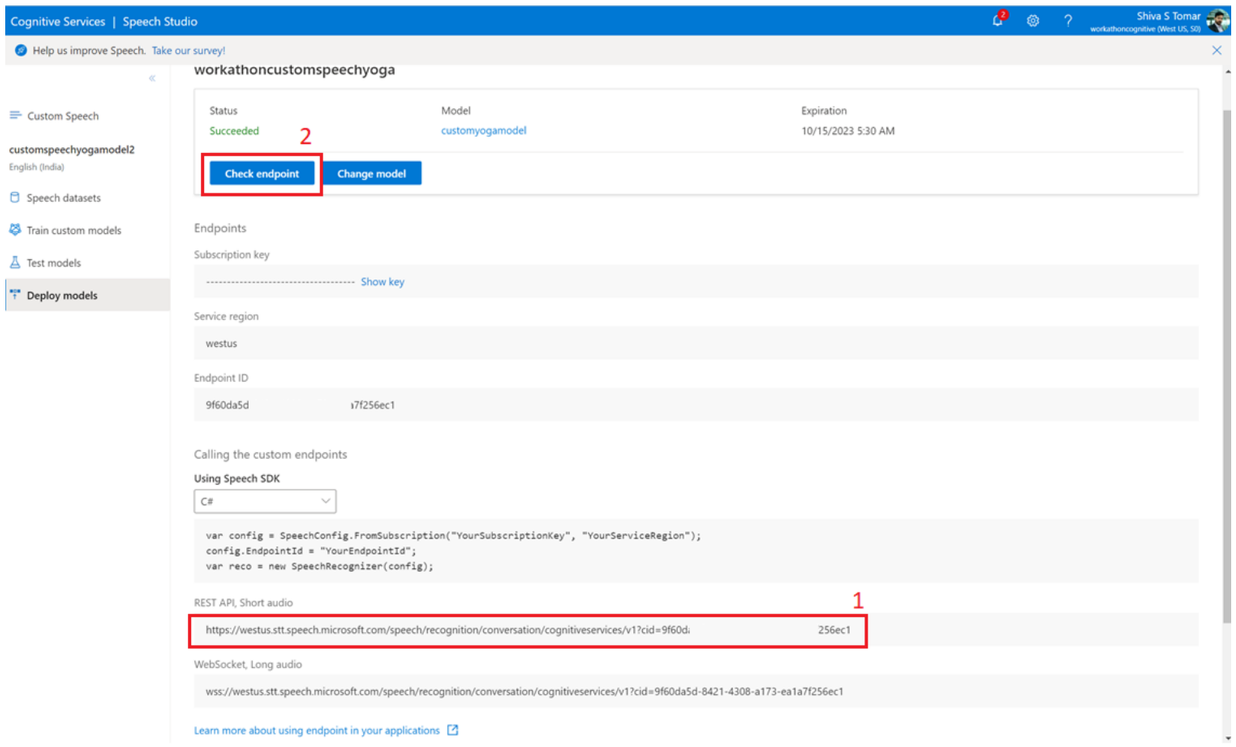
1. Select the Train custom models tab
2. Select the model that you trained
3. Click Deploy models

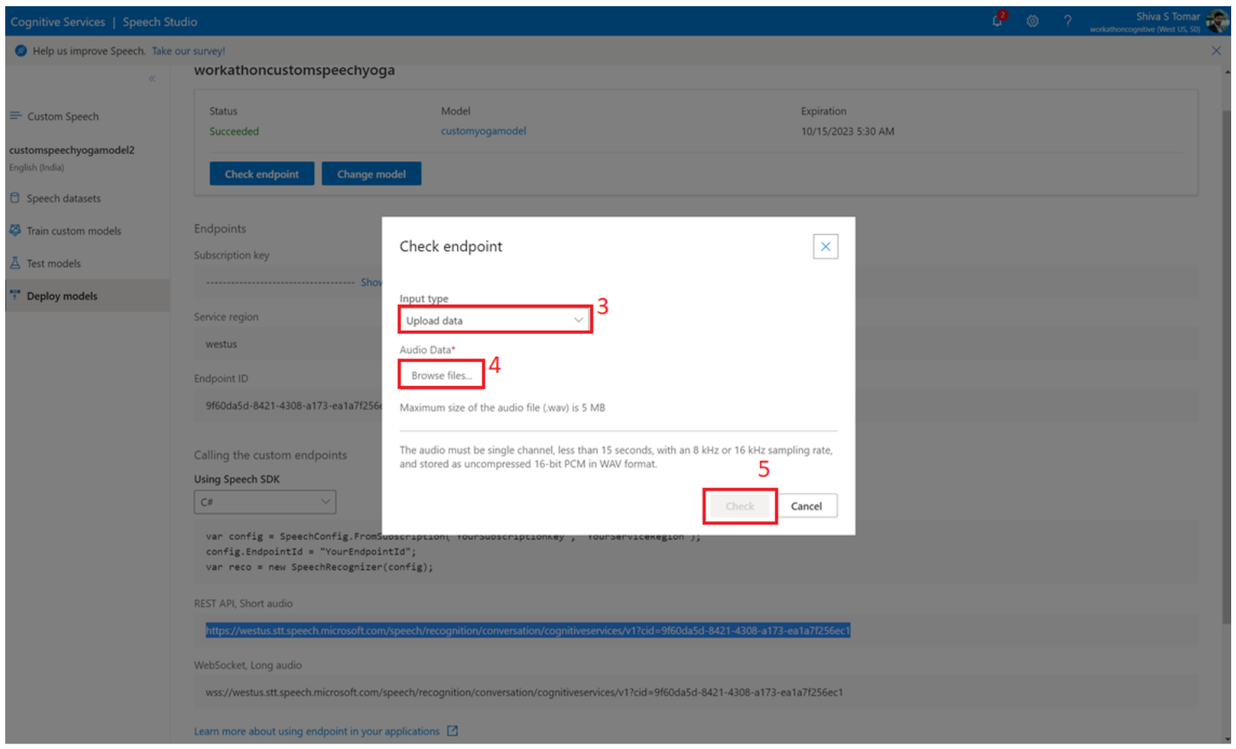
4. Provide the endpoint a name and optionally a description
5. Select the model you want to deploy if it's not auto populated
6. Accept the term of use. You can optionally enable logging and diagnostics.
7. Click Add

8. Switch to Deploy models tab
9. Wait while the deployment status is 'Processing'



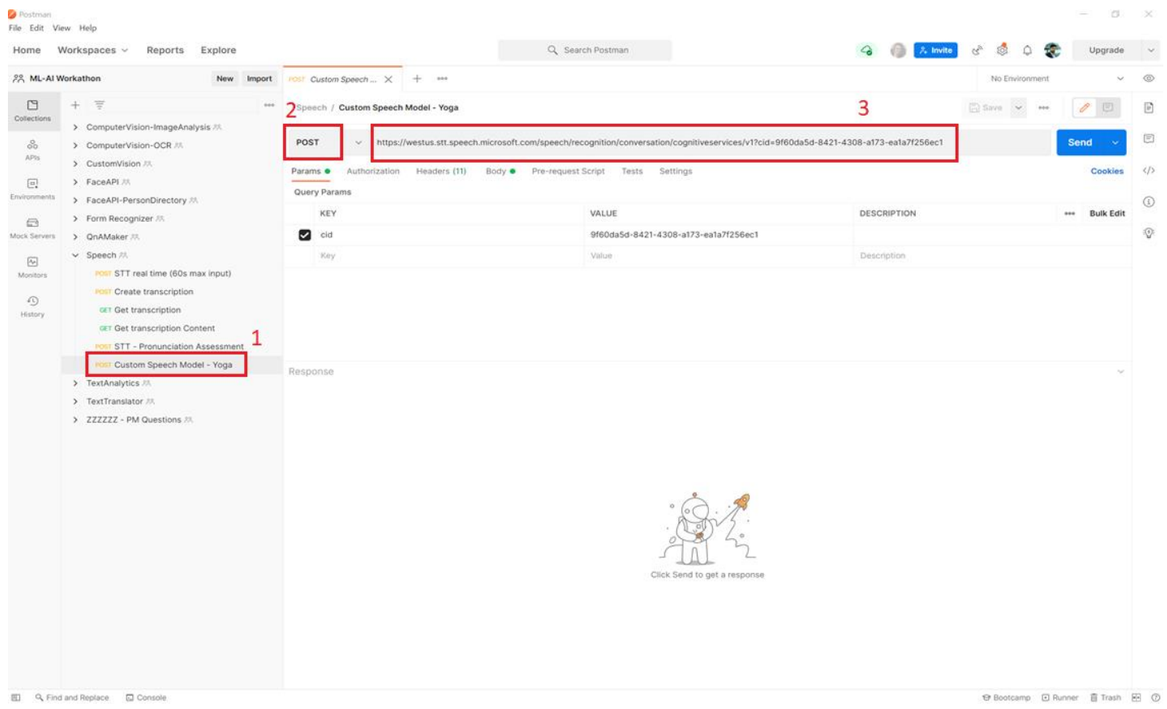
10. Once deployed, the status will change to successful
11. Select the endpoint name, to further explore the deployment details

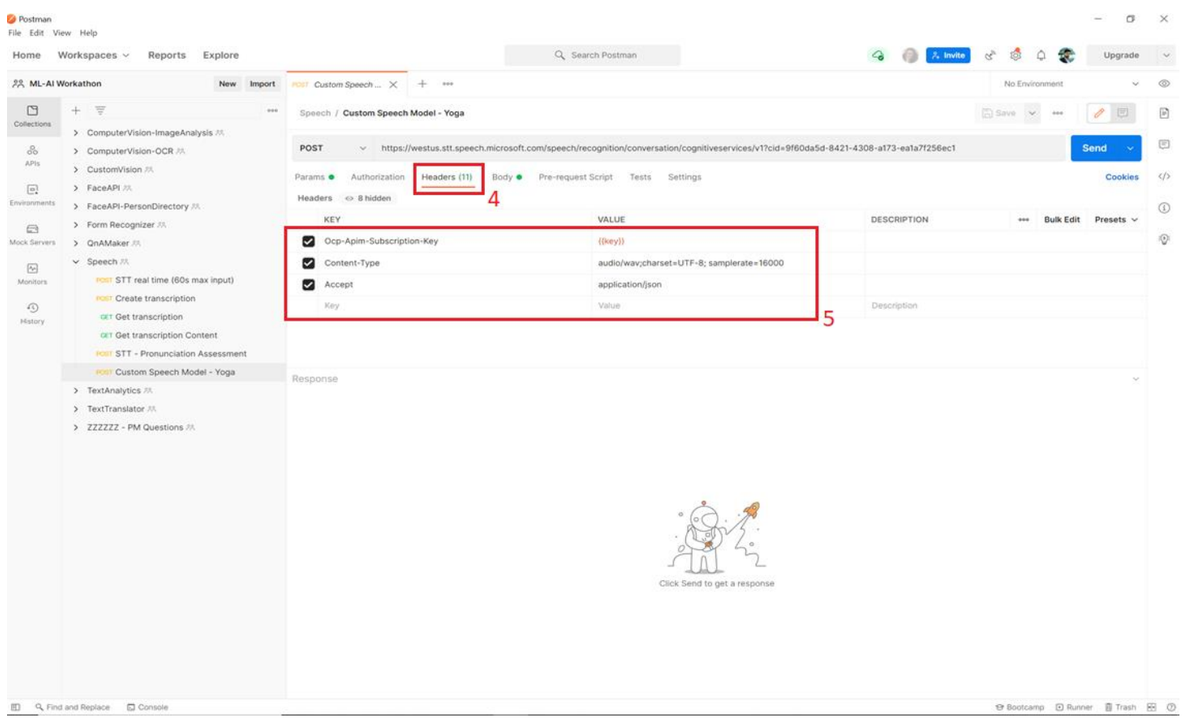


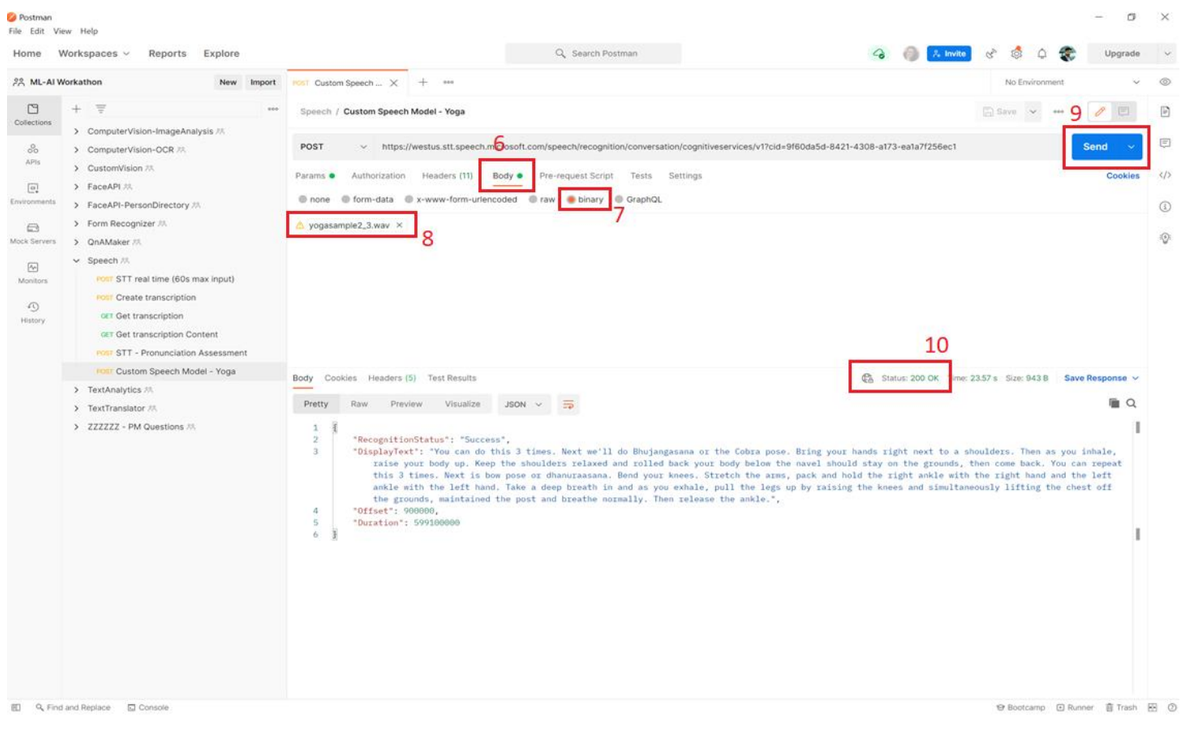


<u>Test the endpoint with new audio data via Speech Portal</u>

1. Copy the REST API as highlighted. We will be leveraging this to make an API call from Postman in the next step.
2. From within the portal, you can do a quick test as well. Click ‘Check endpoint’.
3. Select the input type as Upload Data
4. Browse the test file from your computer
5. Click Check
6. Observe the transcription generated







<u>Test the endpoint with new audio data via REST API</u>

Here, we will be making an API call like the STT real time call we made in '<u>Realtime STT API Call (sync)'</u> step. The difference being that this time we will be using a custom trained model instead of the pre-built STT model.

URL : *<<paste the URL copied from the portal in 'Test the endpoint with new audio data via Speech Portal' step>>*

Params :
cid will be auto populated from the URL. This is a unique ID for your custom model

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

Body :
Select the binary input format and choose a local file from your machine. Make sure the file is less than 60 seconds in length

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

1. {{key}} : Value being picked from global variables

2. Headers:
   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.
   Accept : This refers to the output type that you are expecting.

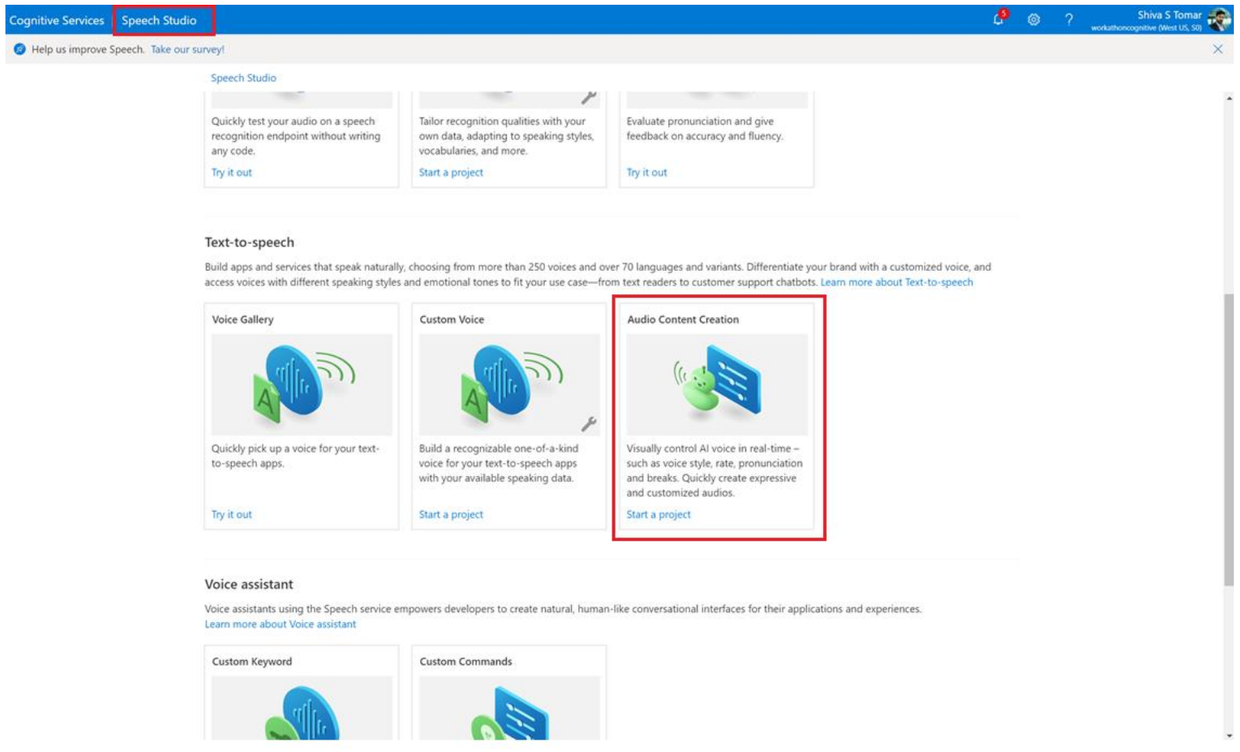
3. Params
   cid : This is a unique ID for your custom model.

4. After you execute the call, observe the status returned, as shown in step 10. This should reflect 200 OK.

5. Observe the JSON output and transcription returned.

Try generating the transcription for the same file using the pre-built model as well and notice the improvement in transcription using the custom model.

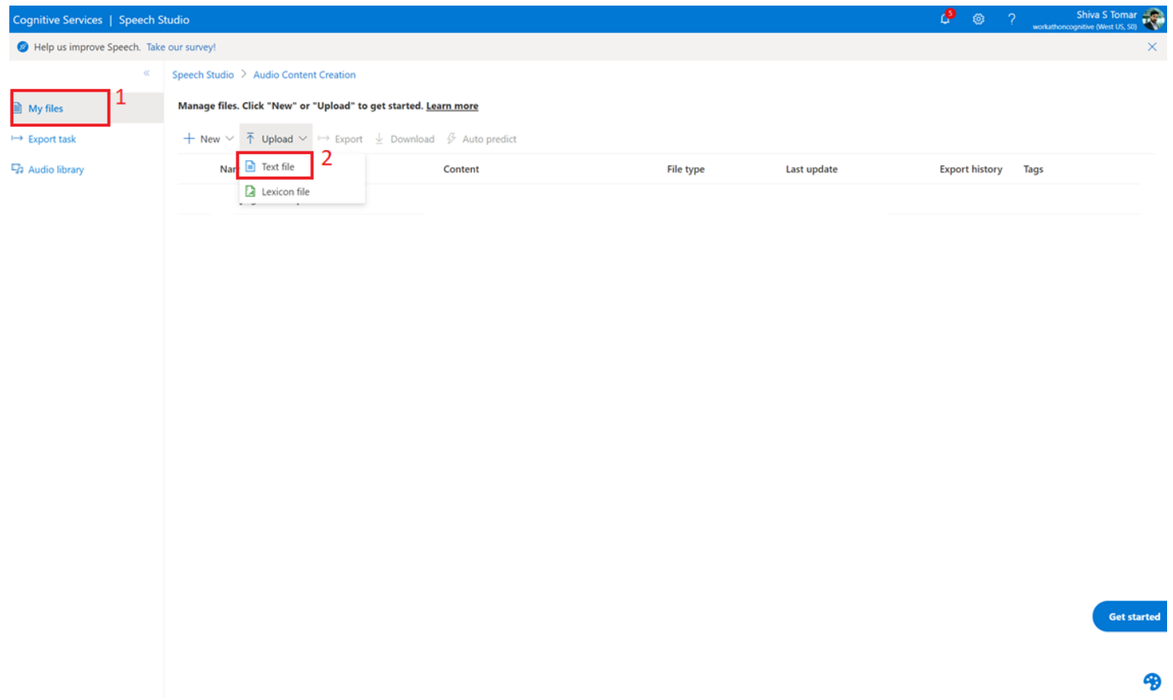
Custom Text to Speech Model

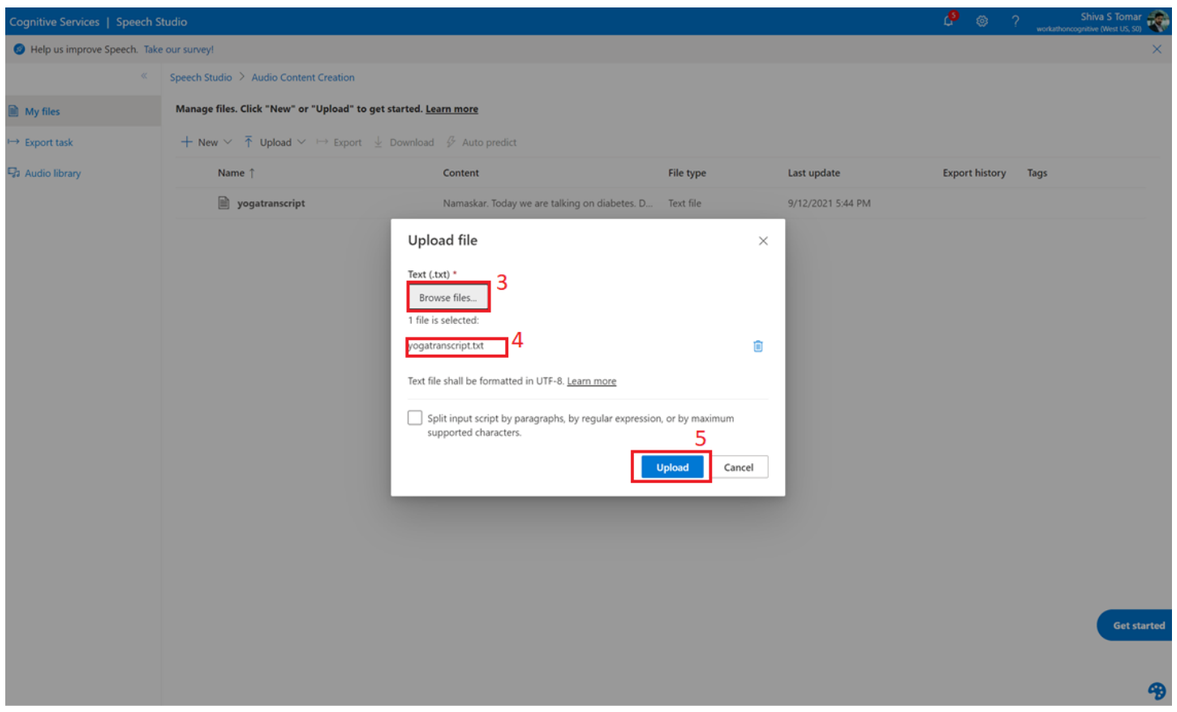


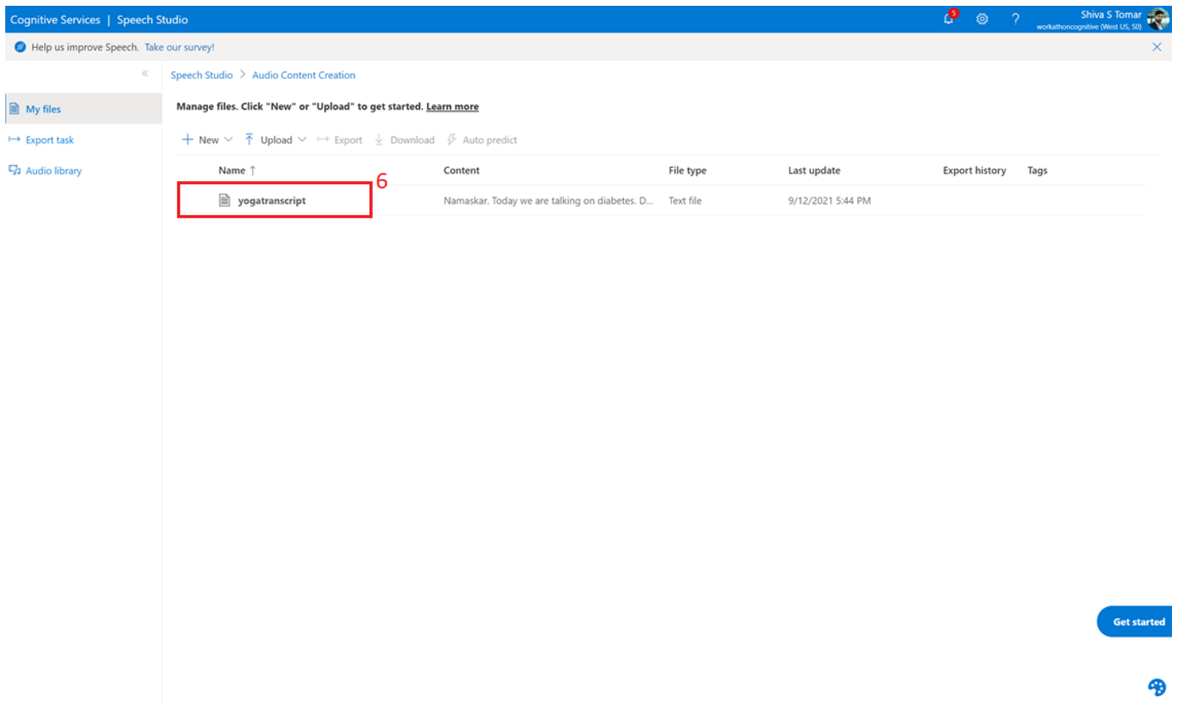
We will now create the 2nd Custom Model, using 'Audio Content Creation' functionality of the Speech Portal.

Exploring Azure Speech Portal

1. Go to speech portal
2. Ensure that you are connected to the right cognitive service account. If not, click the settings gear to select the cognitive service account we created in the beginning.
3. Select Audio Content Creation, as part of Text-to-speech functionality.







Create a new Project

For 'Audio Content Creation' functionality, a project refers to a collection of text files for which you want to create custom audio content.

1. Click My files

2. Select Text File in the Upload menu
   - As part of Text Files, you can upload plain text or SSML files. These are the files containing raw text, on which you can apply customisations like speaking styles, change volume, pitch etc.
   - You can also upload Lexicon Files. These are the files that contain pronunciation rules for individual words [graphemes]. These can be in terms of an alias, phonetic pronunciation etc.
   - Apart from uploading files, you can also create Text files or Lexicon files from scratch using the + New Option.
   - Do try opening a Lexicon File as part of + New menu, to get a better idea of what it is and how it's structured

3. Browse for a Text file on your computer

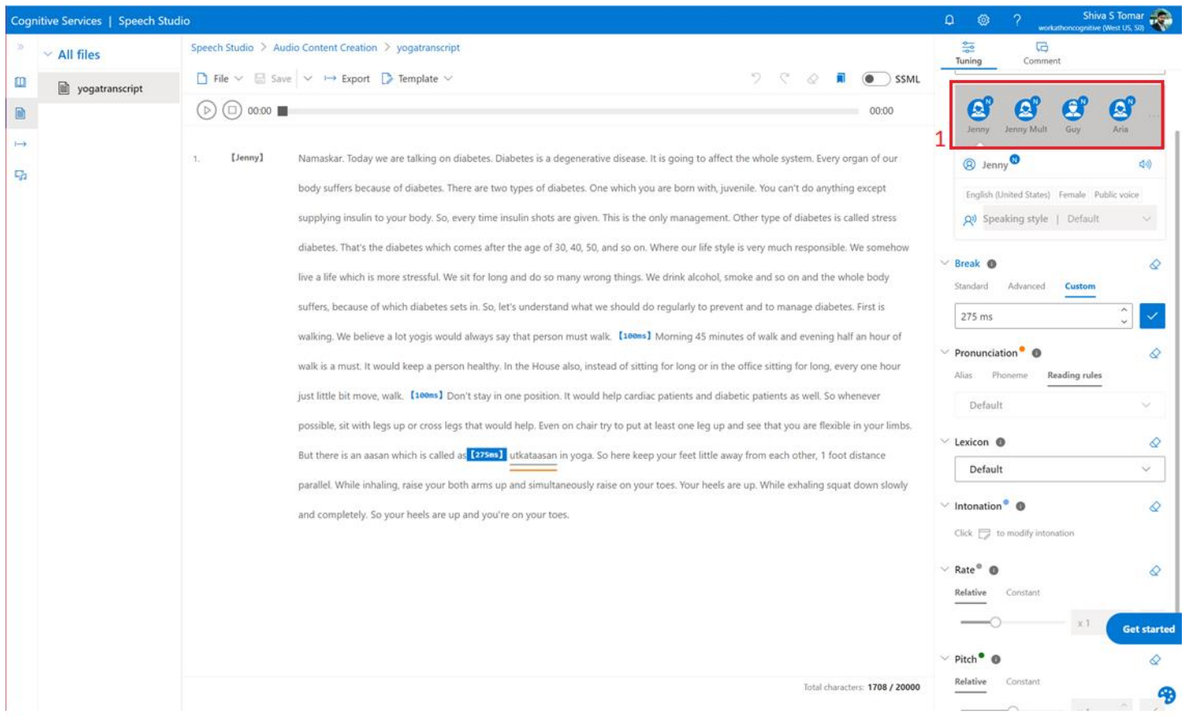
4. Wait for the file to be uploaded

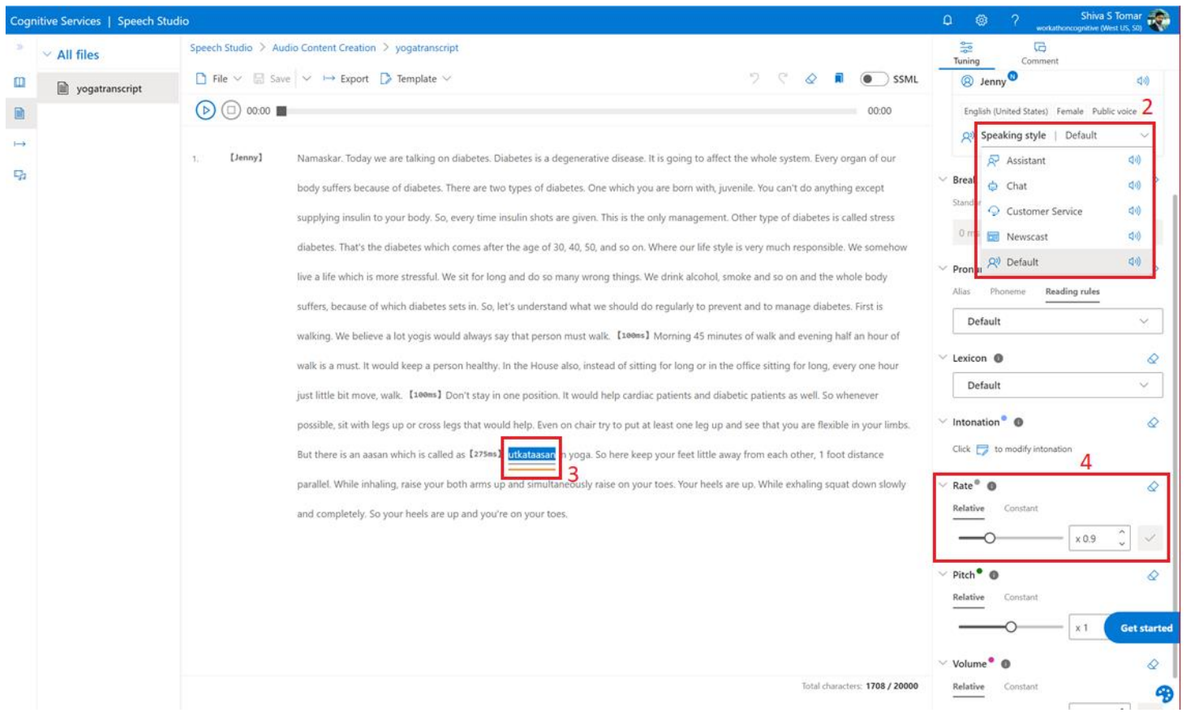
5. Click Upload

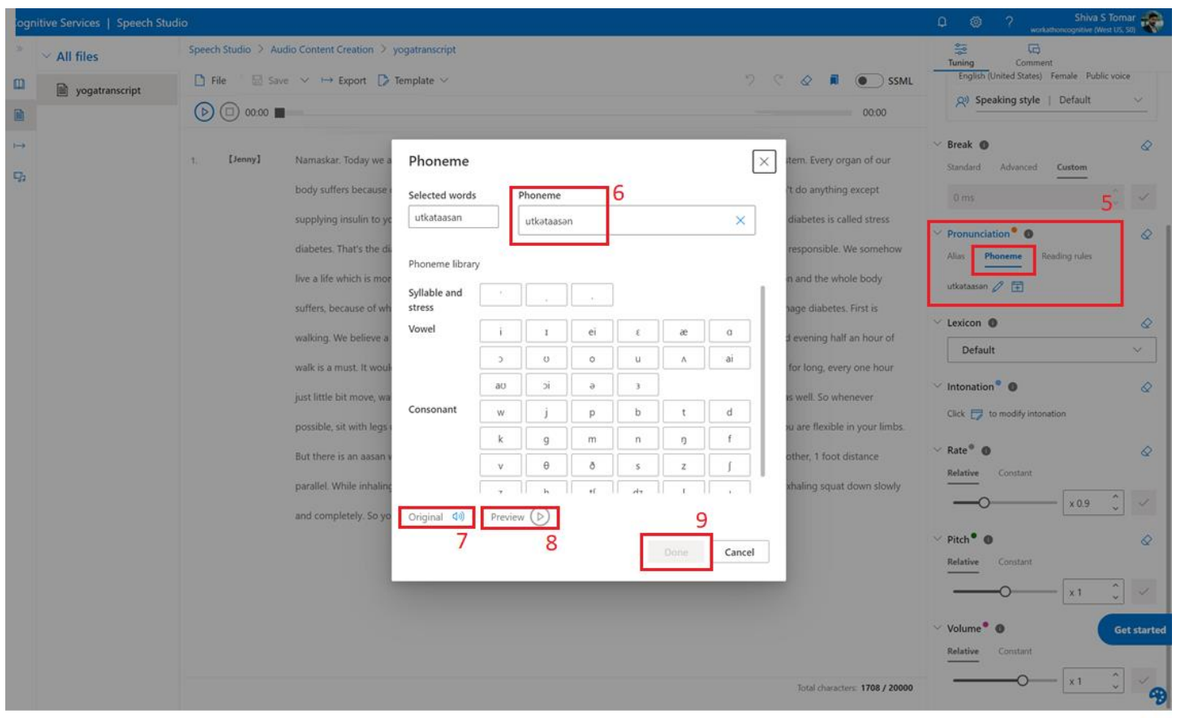
When uploading a new file, you can alternatively choose to break it down into separate smaller files by paragraphs, characters or by using regular expressions. This is useful in 2 ways :

a. The maximum limit for 1 file is 20000. So, you might want to break it into smaller files in case the file contains more than 20000 characters.
b. You might want to organize the file and extract the audio in pieces. For eg : by chapters.

6. You will see the file listed in the My files tab. Click on the file name to edit and get started.







Creating Audio Content

When you are creating your own Audio Content, you can customise the audio output in a variety of ways, including different voices, speaking styles, custom pronunciation, changing pitch, volume or rate of speech and so on.
These customisations can be applied at different granular levels – Complete document, a paragraph, a sentence or even at an individual word level.

1. We first start with choosing a voice for our audio content
   - Depending on the language of your content, you can choose from different locales available for the language.
   - You can also choose between a male and a female voice.
   - Each paragraph [Denoted by the numbering in the left index] can have a different voice. You can also have content in different languages within the same file and choose the voice accordingly.
   - Given we had got this text by converting speech in Indian English accent (in the above workshop task), we will now choose a voice in US English accent (Jenny) to create our content.
   - In case you have a custom voice built, you can also select that.

2. For some voices, you also have different speaking styles available.
   - Assistant, Chat, Customer Service, Cheerful, Empathetic etc. Some of these might or might not be available basis the voice you have chosen.
   - These speaking styles allow you to give a personality to you chosen voice.

3. To try out other customizations, select a particular word. We will be applying some changes to the work ‘utkataasan’, since this wasn’t being pronounced so well by the model by default.

4. You can reduce the rate of speech in case you want to slow down the speed at which a particular word is spoken. Alternatively, you can also try experimenting with the Pitch & Volume for individual words.

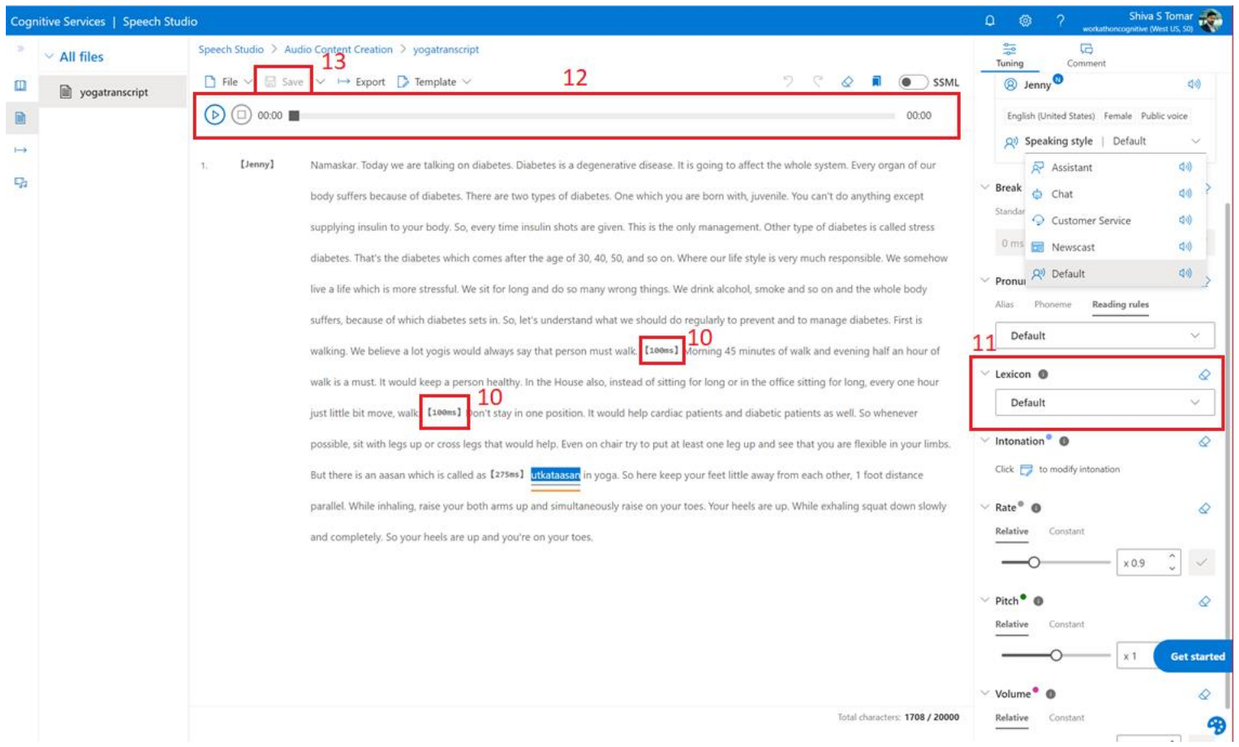
5. We will also be changing the pronunciation for the word ‘utkataasan’, since this being an Indian Origin Sanskrit word isn’t being pronounced accurately by the model. You have different ways to improve the pronunciation
   - You can provide an alias (Eg - if you want IaaS to be pronounced as ‘Infrastructure as a Service’, you can put that as an alias for IaaS)
   - You can provide the exact Phoneme for a word [Phoneme is the smallest unit of speech distinguishing one word (or word element) from another]
   - You can provide reading rules such as Cardinal, Ordinal, Currency, date, Email, Spell etc. This provides clarity to the model for otherwise ambiguous text. (Eg – ¾ can be read as ‘three fourths’ or ‘3rd April’ or ‘4th March’ depending on the context and culture. All this can be defined here)
   - In our case, we will be using Phoneme to improve the pronunciation.
   - Do try experimenting with customisations learnt above on different words

6. Using the Phoneme Library, set in the exact phoneme for your chosen word. You can refer the Phoneme reference library to learn more about this.

7. You can listen in to the original sound in case you want to compare it with how the custom voice is coming up.

8. You can also preview the custom sound as and when you build on the phoneme.

9. Click Done when you are done.

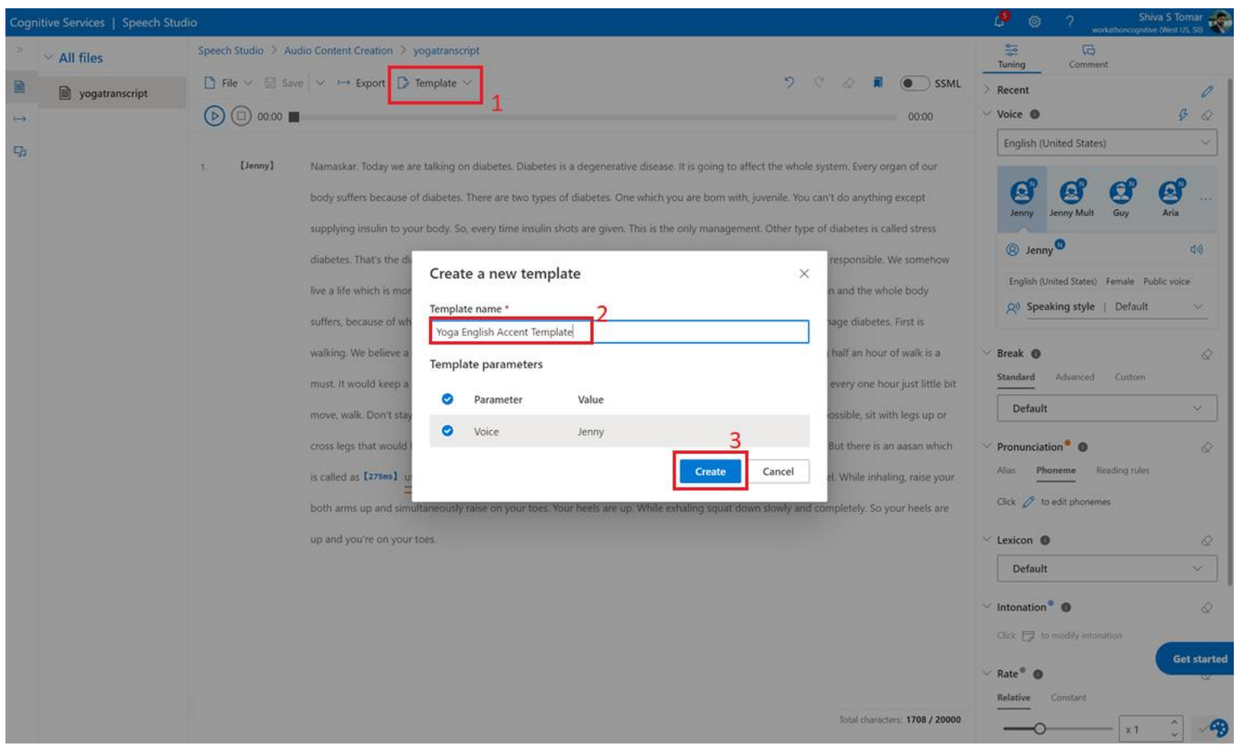


10. You can also customise the breaks between words. Click between the words where you want to increase or decrease the break time. In the break tab on right, try out different options.

11. In case you had created a Lexicon file, you can select the file as shown in step 11. This will apply the custom lexical pronunciations to all the corresponding words in the document, thus saving time and effort.

12. You can listen to the content created using the play button at the top. To listen to a particular snippet, select the text before hitting play. Stop and play again to allow changes to be reflected.

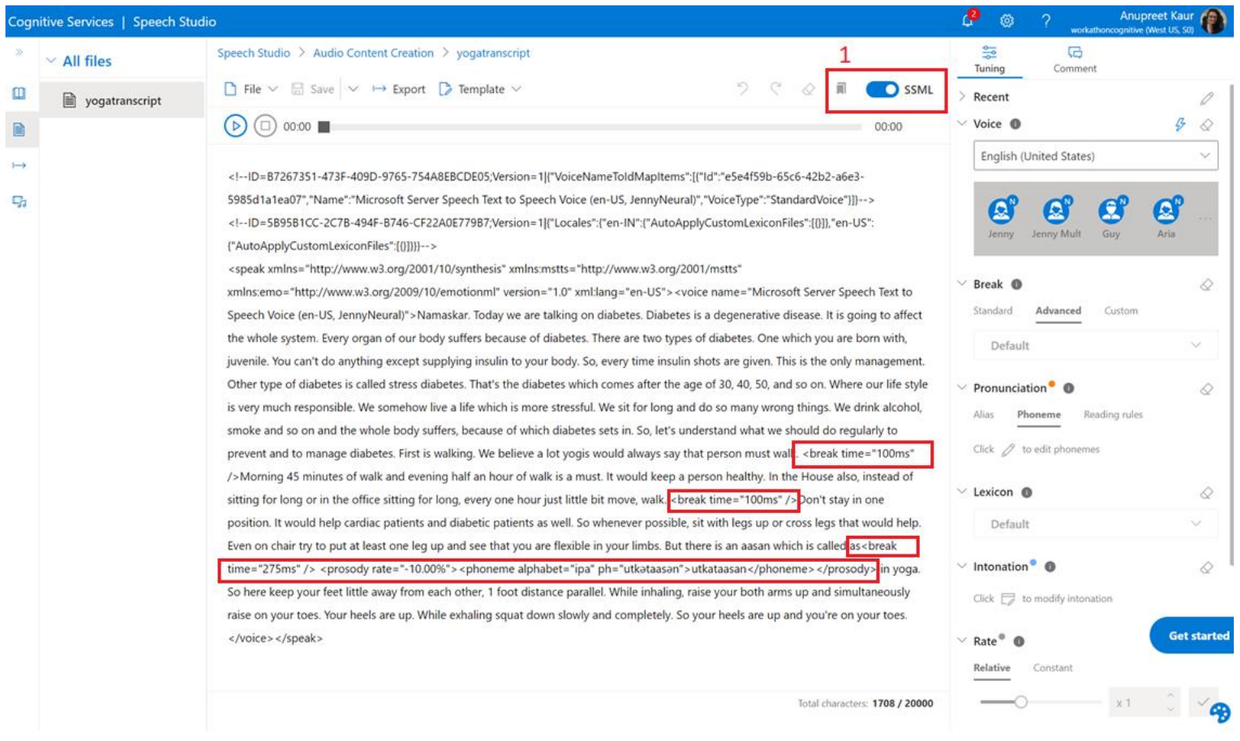
13. Make sure to Save the changes made



Creating Templates

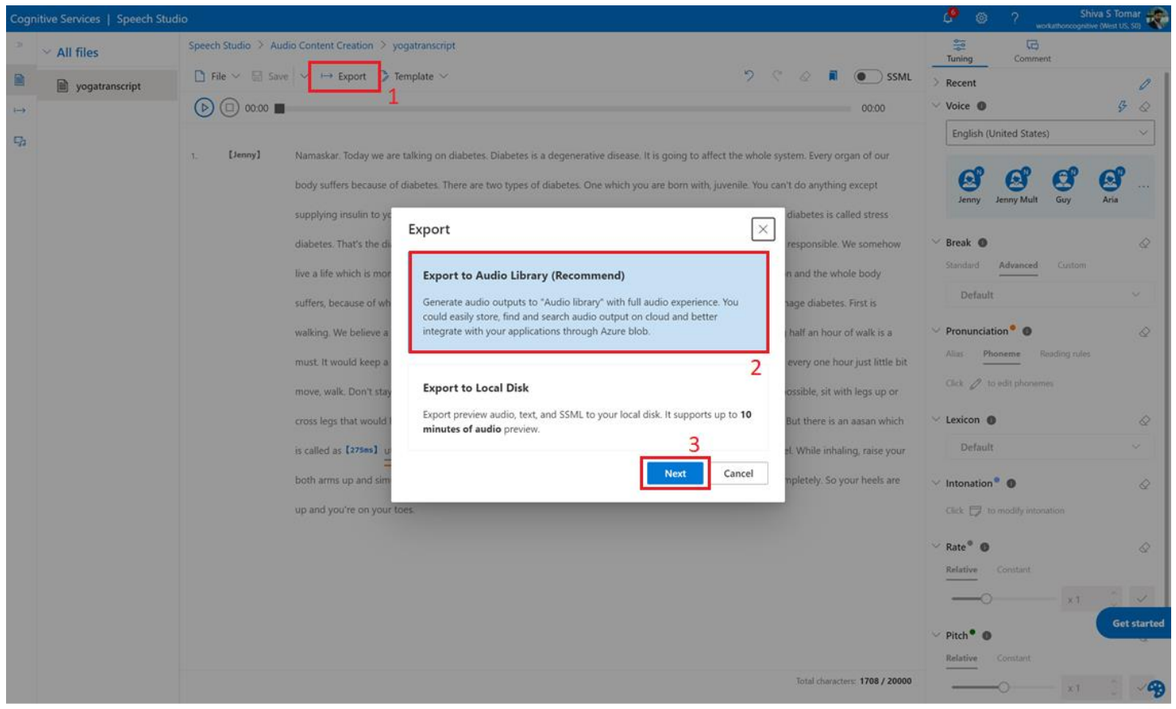
Using templates, you can also save the customisations to be applied to some content later.

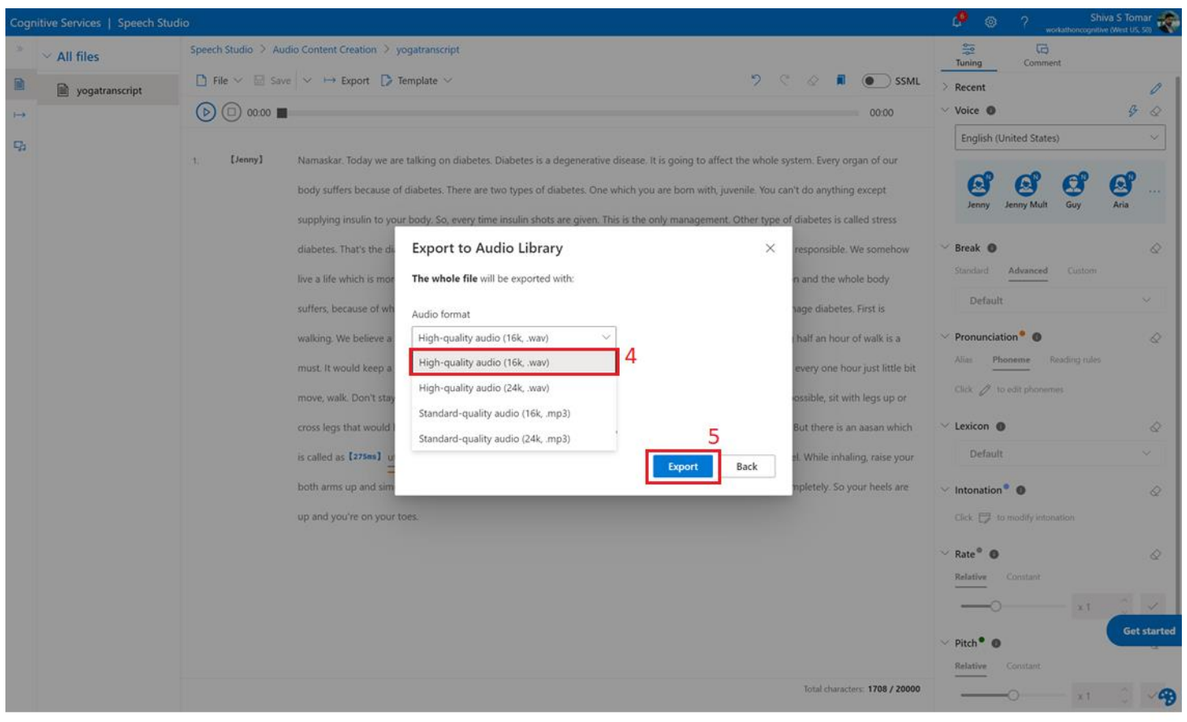
1. Select Template
2. Give a suitable name
3. Click Create

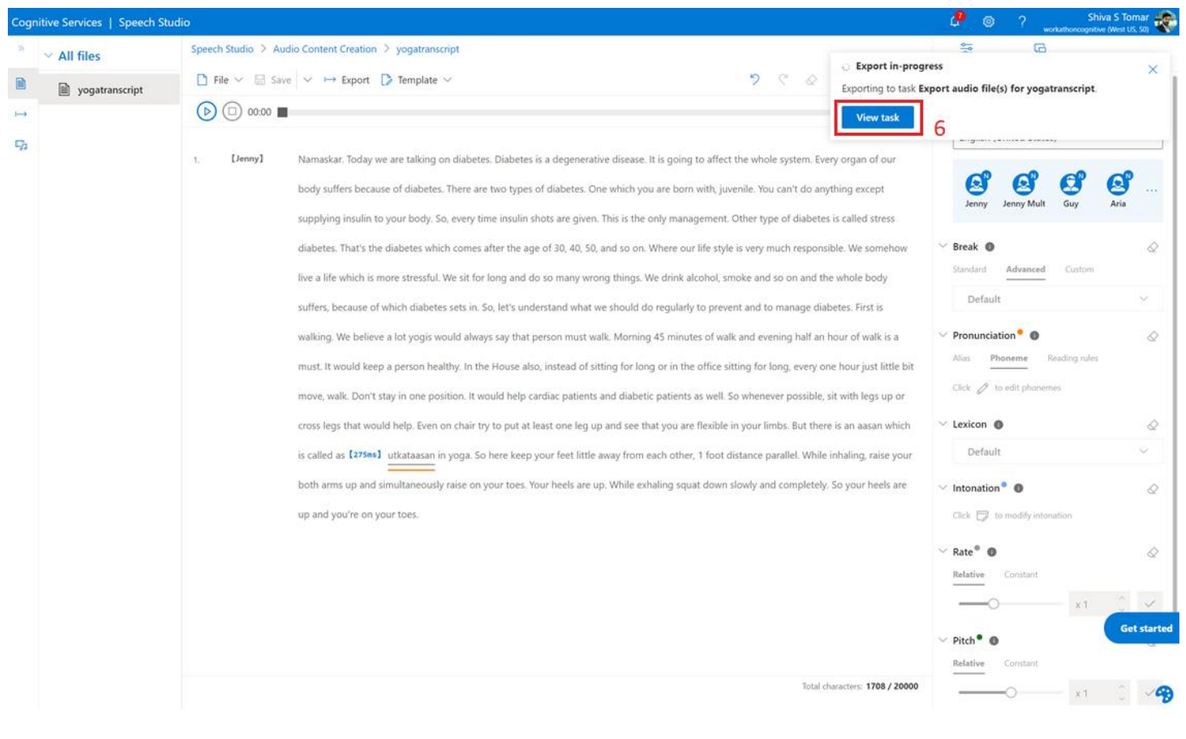


Extracting SSML

1. Once you are done with the customisations, toggle the key on top to SSML
2. Observe how all the changes you made have been applied as SSML tags (Breaks, phonemes etc)
3. You can copy or save the SSML from here to leverage it later. This way, you can also generate SSML tags and content to be used in a real time production scenario. You can dynamically change the main text and reuse the surrounding SSML tags basis the use-case.
4. We will make a real time API call using the generated SSML in a later step.







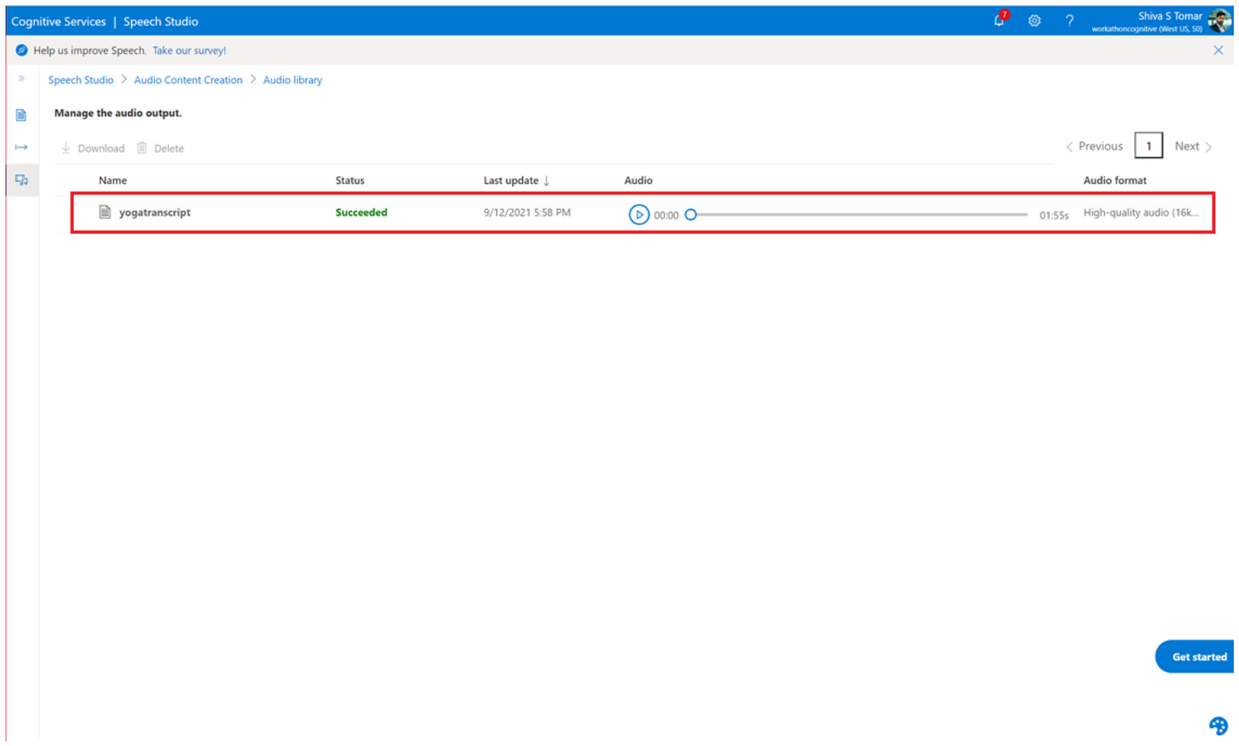
<u>Export Content</u>

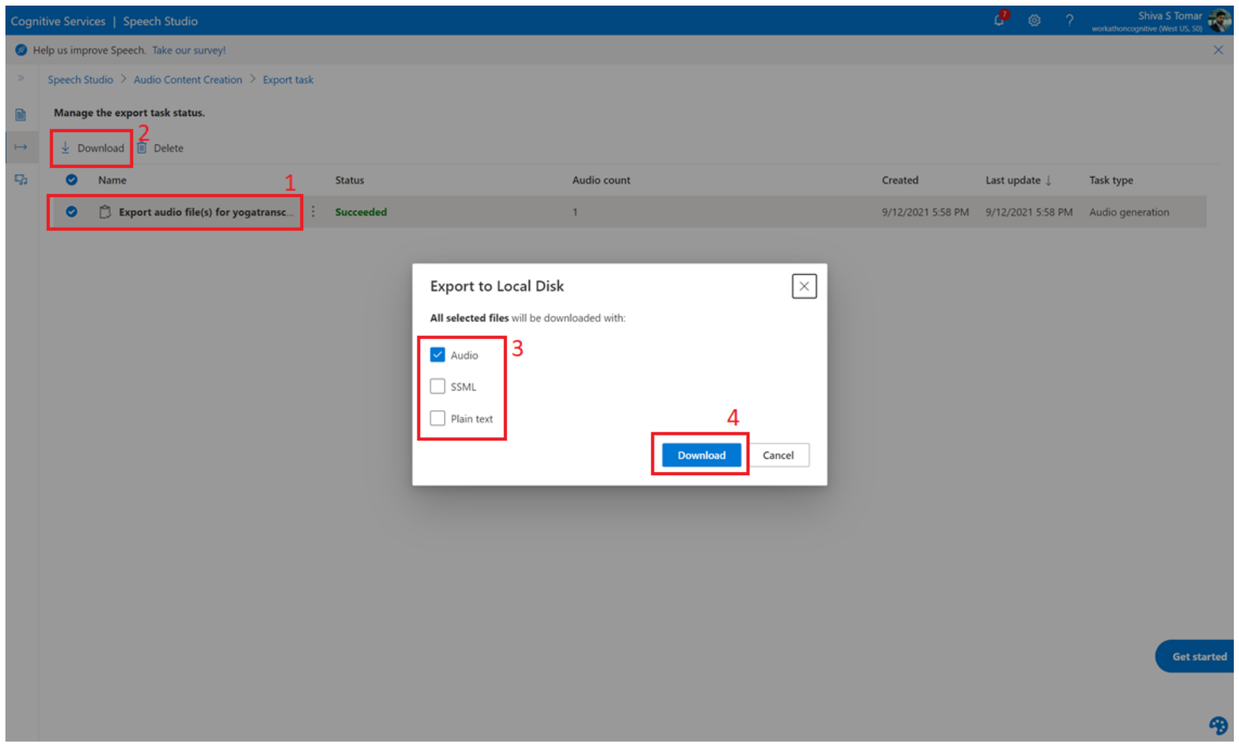
Once you are satisfied with the content that you have prepared and have saved the changes, you are good to export the audio.

1. Select Export
2. Select Export to Audio Library.
   - This will save the content to an inbuilt library in the Speech Studio and you can extract the audio content anytime you like.
   - You can also directly export to local disk, however, any audio greater that 10 minutes will be chunked
3. Click Next

4. Select the Audio Quality and File Media type basis your need
5. Click Export

6. Select View task to see the progress. This will take some time depending upon the size of the file.

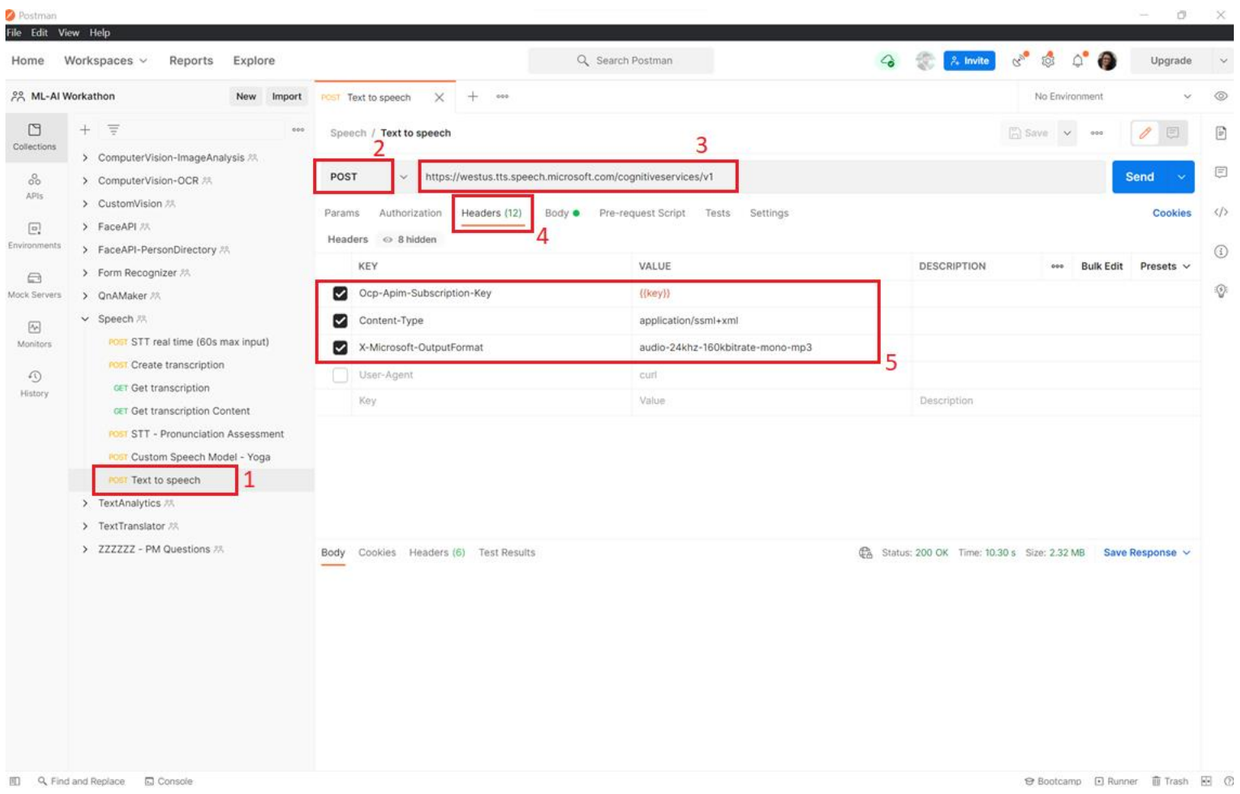


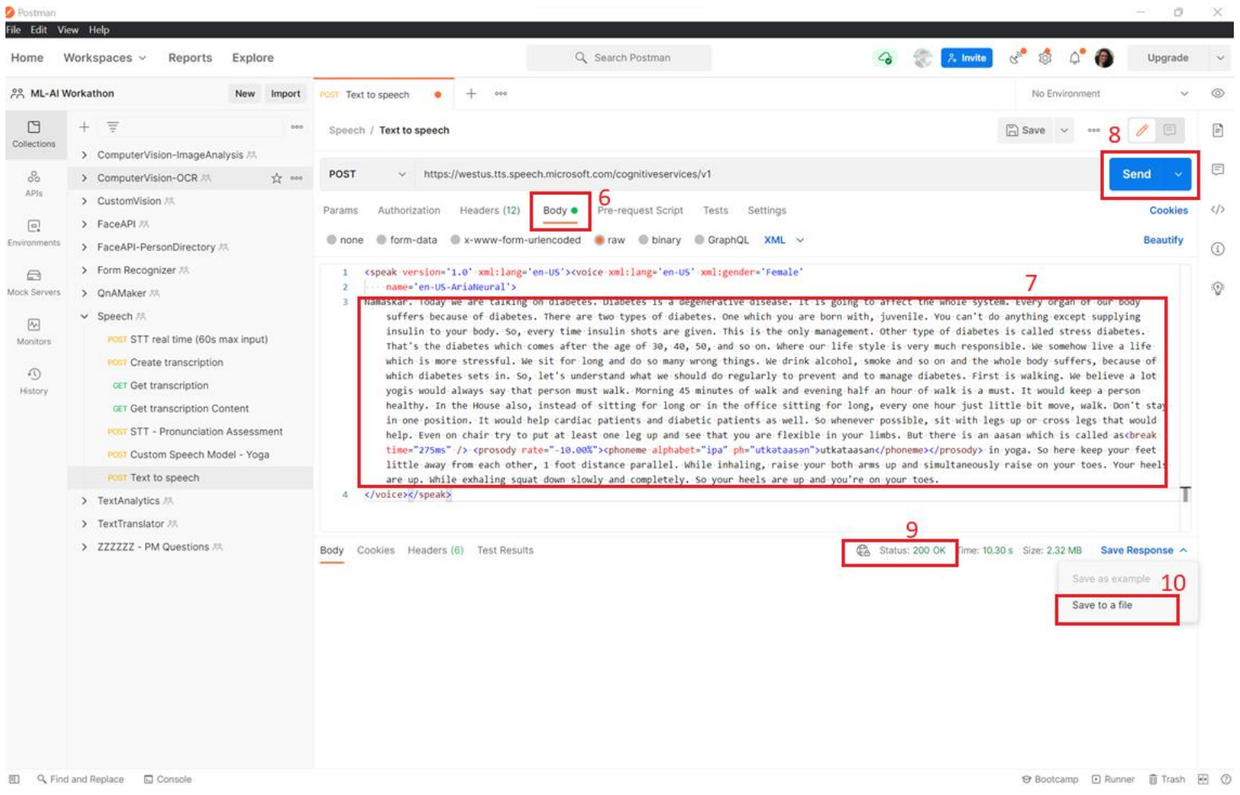


Download Content

Once the content is imported, you can view it in the Audio Library tab.

1. Select the Exported task
2. Click Download
3. In the pop up, select the content that you want to download. You can download the Audio content, corresponding SSML as seen above and the original Plain Text file.
4. Click Download.





Leveraging SSML to make a real time TTS API call

URL : https://westus.tts.speech.microsoft.com/cognitiveservices/v1

Headers :
Ocp-Apim-Subscription-Key : {{key}}
Content-Type : application/ssml+xml
X-Microsoft-OutputFormat : audio-24khz-160kbitrate-mono-mp3

Sample Body :

```
<speak version='1.0' xml:lang='en-US'><voice xml:lang='en-US' xml:gender='Female'
   name='en-US-AriaNeural'>
Place your custom content here
</voice></speak>
```

Replace the custom content with content from SSML tab in the Speech Portal. See step 7 in screenshot for reference.

Significance of input & output

1. {{key}} : Value being picked from global variables created early on in the workshop

2. Headers
   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.
   X-Microsoft-OutputFormat : This refers to the audio output format that you are want.

3. Body This provides an SSML input. SSML contains main tags like <Speak> </Speak>, <voice> </voice> and additional tags like <break time="275ms" /> <prosody rate="-10.00%"><phoneme alphabet="ipa" ph="utkətaasən">utkataasan</phoneme></prosody> depending on your content.

4. If the call is successful, you will receive status 200 OK as shown in step 9

5. Select Save to a file option in Save Response menu and save this to a location on your computer. Play the saved audio to observe the output. In a production scenario, you might use language specific SDKs to play the audio content in real time or automatically save it to a storage location, however, we are saving the output as a local file since Postman doesn't allow to play audio outputs in the client itself.

Homework

1. Try different languages and locales for both STT and TTS functionality
2. Try more customisations while creating custom Audio Content.
3. Once you are comfortable with the batch STT API call flow (3 call API flow for Async batch STT), try out the workshop for Text Analytics and then move on to End to End Meeting Summarization workshop, that leverages Logic Apps, Speech API, text Analytics APIs and Azure Data Factory to build an end-to-end use case on Meeting Summarization.

Additional recommended resources

Service Name	Category	Links
Speech Services	Programming Language	C++, C#, Java, JavaScript, Objective-C, Python, Go
	Tiers	Free, Standard | (Difference between Tiers)
	Pricing	https://azure.microsoft.com/en-us/pricing/details/cognitive-services/speech-services/
	Limits	https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/speech-services-quotas-and-limits
	Language Support	https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/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.

Speech service Documentation

Speech-to-text API references

Text-to-speech API references