**User Identification Testing & Speech-to-Text : Report**

Rocket Elevators wants to go the “extra mile” and to launch a new features for its elevators : a “Speaker Recognition” function that will enable the users to sign in with their profile ID with their voice. If not already registered in the system with a profile ID, the user can sign up (also vocally) with a sequence of steps that will be detailed below. A second objective of Rocket Elevators is to be able to transcript conversations in its elevators to actual text.

The present report will detail the inscription and recognition sequences a user must follow to use the system. It will also describe the tests and steps that had to be realized in order for the user profiles to be saved in the system and recognized. It will then detail the procedures required to render those audio conversations in the text form with Azure Speech-to-Text.

The Microsoft Cognitive Services were used for that purpose. The system has to be taught which voice profiles to recognize as well as the vocal signatures related to it. The profiles have to be created following the cognitive services documentation (detailed on <https://westus.dev.cognitive.microsoft.com/docs/services/563309b6778daf02acc0a508/operations/5645c068e597ed22ec38f42e>).The instructions that were followed were in the “Identification Profile” and “Speaker Recognition” sections of the documentation.

The first part of the report, “Speaker identification”, will detail 10 audio conversations from the elevators and try to identify one or many speakers in those conversations. The second part of the report, “Speech-to-text” will take those 10 conversations and try to render them in a text form. The final part of the report, “Methodology”, will detail the methods and steps that were needed for the user profile creation, its enrollment, the fetching of its profile, and finally the identification and confirmation of that user. All the tests of this report have been made on the Postman application to make sure all the calls are ready to be integrated to our Ruby on Rails Web application.

**Speaker identification : Azure Speaker Recognition**

This part of the report will analyze 10 different conversations in the form of an audio file and try to identify (or not) different registered “profile IDs” (see the “Methodology” part of the present report). Each conversation will be analyzed with the following criteria :

1. Main objective
2. Choice of implementation and parameters
3. Result for every file
4. Conclusions and recommendations if the experience was to be repeated

**Overview of the four criteria**

1. *Main objective* : see the result given by the Azure speaker recognition API when no registered profile ID can be heard and see the result when one, two or three registered profile ID(s) can be heard. We also wanted to push the limits of the speaker recognition API by handing it bilingual conversations to see what would come out.
2. *Choice of implementation and parameters* : we posted the conversation [x} audio file(s) to the following url in the Postman application : https://westus.api.cognitive.microsoft.com/spid/v1.0/identify?identificationProfileIds={identificationProfileIds}

Because we have no profile ID to recognize sometimes (conversation 1, for example) and because the documentation is clear on the fact that we absolutely need to enter a profile ID to be verified (even though the user is not in the audio file), we went with the user 1 profile by default for the unidentifiable conversations (1 & 2) (see the “Methodology”/”User profile creation” part of this report). The following key/value pairings were entered in the “Headers” section : Content-type / multipart/form-data, Ocp-Apim-Subscription-Key / a9e3e21c91124fc3bb63a9e8ec9813a4. The “{identificationProfileIds}” in the url is the “identificationProfileId” we want to identify and was/were entered in the “Params” section of the Postman request. Up to 10 different profile IDs can be verified on the same audio file. The audio file of conversation [x] was then attached to the “Body” section of the POST request as a binary file. When all the parameters were entered, we pressed the “Send” button to get the Operation ID for every conversation to then be able to GET the result with another operation (see the “User confirmation” part of the “Methodology” section of the present report).

1. *Result for every file* :

**Conversation #1, Audio file : Conversation-1.wav : a conversation between 3 people, none of them being a user with a profile ID**

**Operation-Location : https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/4097ef60-d0c6-493f-95d9-fb2f0a2ef391**

**GET result (user confirmation) :** {

"status": "succeeded",

"createdDateTime": "2019-05-02T12:28:58.8363939Z",

"lastActionDateTime": "2019-05-02T12:29:01.9335256Z",

"processingResult": {

"identifiedProfileId": "00000000-0000-0000-0000-000000000000",

"confidence": "Normal"

}

}

*As explained in the “User confirmation” part of the present report, the message we got from the GET [operation-location url] means no registered profile ID has been recognized by the API, which is in this case 100% normal.*

**Conversation #2, Audio file : Conversation-2.wav : a conversation between 5 people, none of them being a user with a profile ID**

**Operation-Location : https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/d7e6b85b-e375-4a08-abe3-9b57ab3ec81a**

**GET result (user confirmation) :**{

"status": "succeeded",

"createdDateTime": "2019-05-02T12:36:07.1332413Z",

"lastActionDateTime": "2019-05-02T12:36:09.4730124Z",

"processingResult": {

"identifiedProfileId": "00000000-0000-0000-0000-000000000000",

"confidence": "Normal"

}

}

*As explained in the “User confirmation” part of the present report, the message we got from the GET [operation-location url] means no registered profile ID has been recognized by the API, which is again completely normal.*

**Conversation #3, Audio file : Conversation-3.wav : a conversation between 2 people, 1 of them being a user with a profile ID (Maxime Patry, ID : cb5edf35-9396-46f3-b133-7e695ab1479d, en-us)**

**Operation-Location : https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/b0cedb56-0c5e-4036-8902-591668d7b2c8**

**GET result (user confirmation) :**{

"status": "succeeded",

"createdDateTime": "2019-05-02T12:41:03.615273Z",

"lastActionDateTime": "2019-05-02T12:41:05.8456756Z",

"processingResult": {

"identifiedProfileId": "cb5edf35-9396-46f3-b133-7e695ab1479d",

"confidence": "High"

}

}

*Here we can see the right profile ID (“identificationProfileId”), Maxime, has been recognized by the speaker recognition API with a “High” confidence level.*

**Conversation #4, Audio file : Conversation-4.wav : a conversation between 3 people, 1 of them being a user with a profile ID (Raphaël Vallée, ID : 1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58, en-us)**

**Operation-Location : https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/c33e620a-6f51-4012-950a-c1209f72f36e**

**GET result (user confirmation) :**{

"status": "succeeded",

"createdDateTime": "2019-05-02T12:44:34.1867454Z",

"lastActionDateTime": "2019-05-02T12:44:36.7787664Z",

"processingResult": {

"identifiedProfileId": "1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58",

"confidence": "Normal"

}

}

*Here we can see the right profile ID (“identificationProfileId”), Raphaël, has been recognized by the speaker recognition API with a “Normal” confidence level. A question seems to raise : will the confidence level of the API change depending on the number of speakers ?*

**Conversation #5, Audio file : Conversation-5.wav : a conversation between 5 people, 1 of them being a user with a profile ID (Rémi Lemay-Dupont, ID :** **df0fc84f-ed6a-42ae-a4f1-3e3ba39a5e4e, fr-fr)**

**Operation-Location : https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/69130e50-7a0f-475d-a14a-88f3f7bfac15**

**GET result (user confirmation) :**{

"status": "succeeded",

"createdDateTime": "2019-05-02T12:49:12.0817945Z",

"lastActionDateTime": "2019-05-02T12:49:15.169842Z",

"processingResult": {

"identifiedProfileId": "df0fc84f-ed6a-42ae-a4f1-3e3ba39a5e4e",

"confidence": "Normal"

}

}

*Here we can see the right profile ID (“identificationProfileId”), Rémi, has been recognized by the speaker recognition API with a “Normal” confidence level. This conversation was in french, this may be useful for the conclusion of this section of the report.*

**Conversation #6, Audio file : Conversation-6.wav : a conversation between 2 people, 2 of them being users with a profile ID (Raphaël Vallée, ID : 1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58, en-us & Maxime Patry, ID : cb5edf35-9396-46f3-b133-7e695ab1479d, en-us)  
Operation-Location : https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/46846ab8-f462-47fe-9d55-e1c75490d0e5**

**GET result (user confirmation) :**{

"status": "succeeded",

"createdDateTime": "2019-05-02T12:54:49.3568711Z",

"lastActionDateTime": "2019-05-02T12:54:52.4001728Z",

"processingResult": {

"identifiedProfileId": "1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58",

"confidence": "High"

}

}

*Here we can see only one profile ID (“identificationProfileId”), Raphaël, has been recognized by the speaker recognition API with a “High” confidence level. After numerous tries, Maxime cannot be identified until we take Raphaël’s ID off the POST request. If we do POST only Maxime’s ID, we get a ‘’High’’ confidence level.*

**Conversation #7, Audio file : Conversation-7.wav : a conversation between 3 people, 2 of them being users with a profile ID (Raphaël Vallée, ID : 1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58, en-us & Maxime Patry, ID : cb5edf35-9396-46f3-b133-7e695ab1479d, en-us)  
Operation-Location : https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/9d1e1fe7-1e6a-4022-8f63-aaca65a2ebce**

**GET result (user confirmation) :**{

"status": "succeeded",

"createdDateTime": "2019-05-02T13:01:19.1975139Z",

"lastActionDateTime": "2019-05-02T13:01:21.599529Z",

"processingResult": {

"identifiedProfileId": "1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58",

"confidence": "Normal"

}

}

*Here we can see only one profile ID (“identificationProfileId”), Raphaël, has again been recognized by the speaker recognition API with a “Normal” confidence level. After numerous tries, Maxime cannot be identified until we take Raphaël’s ID off the POST request.When we tried posting the same request with Maxime’s ID first, no profile ID could be identified with a ‘’Normal’’ confidence level. If we do POST only Maxime’s ID, we get a ‘’Normal’’ confidence level too.*

**Conversation #8, Audio file : Conversation-8.wav : a conversation between 4 people, 2 of them being users with a profile ID (Maxime Patry, ID : cb5edf35-9396-46f3-b133-7e695ab1479d, en-us, Rémi Lemay-Dupont, ID :** **df0fc84f-ed6a-42ae-a4f1-3e3ba39a5e4e, fr-fr)  
Operation-Location :** https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/a54ca3c1-c11e-4871-a489-1a188dd26a44

**GET result (user confirmation) :**{

"status": "succeeded",

"createdDateTime": "2019-05-02T13:07:03.752193Z",

"lastActionDateTime": "2019-05-02T13:07:06.4075913Z",

"processingResult": {

"identifiedProfileId": "cb5edf35-9396-46f3-b133-7e695ab1479d",

"confidence": "High"

}

}

*Here we can see only one profile ID (“identificationProfileId”), Maxime, has again been recognized by the speaker recognition API with a “High” confidence level. After numerous tries, Rémi cannot be identified until we take Maxime’s ID off the POST request. If we do POST only Rémi’s ID, we get a right profile ID recognition with a ‘’Normal’’ confidence level this time. \*\*PLEASE NOTE that the conversation was bilingual.\*\**

**Conversation #9, Audio file : Conversation-9.wav : a conversation between 5 people, 2 of them being users with a profile ID (Raphaël Vallée, ID : 1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58, en-us, Rémi Lemay-Dupont, ID :** **df0fc84f-ed6a-42ae-a4f1-3e3ba39a5e4e, fr-fr)  
Operation-Location :** https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/f014d3f5-3a80-4f31-b517-b095855a5d64

**GET result (user confirmation) :**{

"status": "succeeded",

"createdDateTime": "2019-05-02T13:14:54.7127666Z",

"lastActionDateTime": "2019-05-02T13:14:58.0278925Z",

"processingResult": {

"identifiedProfileId": "1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58",

"confidence": "High"

}

}

*Here we can see only one profile ID (“identificationProfileId”), Raphaël, has again been recognized by the speaker recognition API with a “High” confidence level. After numerous tries, Rémi cannot be identified until we take Raphaël’s ID off the POST request. If we do POST only Rémi’s ID, we get a right profile ID recognition with a ‘’Normal’’ confidence level this time. It is interesting to note that even though Rémi had a French-speaking profile, his English speech was still recognized by the API.*

**Conversation #10, Audio file : Conversation-10.wav : a conversation between 5 people, 3 of them being users with a profile ID (Raphaël Vallée, ID : 1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58, en-us, Maxime Patry, ID : cb5edf35-9396-46f3-b133-7e695ab1479d, en-us, Rémi Lemay-Dupont, ID :** **df0fc84f-ed6a-42ae-a4f1-3e3ba39a5e4e, fr-fr)  
Operation-Location :** https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/824da1d4-d53f-4a63-b6db-086e960856b7

**GET result (user confirmation) :**{

"status": "succeeded",

"createdDateTime": "2019-05-02T13:19:15.3756127Z",

"lastActionDateTime": "2019-05-02T13:19:18.2308534Z",

"processingResult": {

"identifiedProfileId": "1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58",

"confidence": "Normal"

}

}

*Here we can see only one profile ID (“identificationProfileId”), Raphaël, has again been recognized by the speaker recognition API with a “Normal” confidence level. After numerous tries, Maxime nor Rémi can be identified until we take Raphaël’s ID off the POST request. If we do POST only Maxime and Rémi’s ID, we get only one right profile ID recognition with a ‘’Normal’’ confidence level this time (Maxime). We need to take Maxime’s ID off again to be able to recognize Rémi. The conversation was bilingual (Maxime and Raphaël spoke in English, Rémi in French).*

1. *Conclusions and recommendations if the experience was to be repeated* :   
   -The Azure Speaker Recognition API works fairly well, it can always recognize a profile (if present) in a conversation when asked to.  
   -It can also recognize a user speaking in another language than the one provided in his enrollment profile (see Conversation 9 example).  
   -It has its limitations : when asked to recognize more than one user, the API always returns the first profile ID asked as recognized with various confidence levels (either ‘’High’’ or ‘’Normal’’).   
   -The API NEVER returned multiple profile ID recognitions, IDs had to be taken off for another profile to be recognized  
   -The documentation is detailed on what a result means (for example when receiving the 0000-0000-00000000-00000000000 profile ID).   
   -If we had to do it again, we would try with very different voice profiles (children, women) to see if the API can ever recognize more than one speaker at the same time.

-Even if the documentation mentions being able to recognize up to 10 profile IDs, if asked, in an audio file, the experiment could not demonstrate this at all.

**Conversation transcription : Azure Speech-to-Text**

This part of the report will analyze 10 different conversations in the form of an audio file and try to transcript the conversations from the same 10 audio files analyzed in the “Speaker identification” part of this report. The tests will be realized with the Postman application and the methods to do so are detailed in the “Speech-to-Text” part of this report located in the “Methodology” section. The results will be analyzed with the following criteria :

1. Main objective
2. Choice of implementation and parameters
3. Result for every file
4. Conclusions and recommendations if the experience was to be repeated

**Overview of the four criteria**

1. *Main objective* : see the result given by the Azure Speech-to-Text API when sent an audio file with English-speaking, French-speaking and bilingual recorded conversations. We also wanted to push the limits of the speech-to-text API by handing it bilingual conversations to see what would come out. The result of every API call will be extracted in .txt files with the following name format : Conv-[x]-json.txt.
2. *Choice of implementation and parameters* : we sent a POST request with the following url in the Postman application : <https://westus.cris.ai/api/speechtotext/v2.0/transcriptions>

The following key/value pairings were entered in the “Headers” section : Content-type / application/json, Ocp-Apim-Subscription-Key / 759435af27c74cf1b2500edbb9eb51d6. The following text was entered in the ‘’Body’’ part of the request :

{

"recordingsUrl": "https://respeechtotextstorage.blob.core.windows.net/rocketconversations/Conversation-1.wav",

"models": [],

"locale": "en-US",

"name": "Transcription using locale en-US",

"description": "An optional description of the transcription.",

"properties": {

"ProfanityFilterMode": "Masked",

"PunctuationMode": "DictatedAndAutomatic"

}

}

Where only the ‘’recordingsUrl’’ and ‘’locale’’ parameters have to be changed from conversation to conversation (see the ‘’Speech-to-Text’’ part of the ‘’Methodology’’ section of the present report). Then, the ‘’Send’’ button was pressed, which returned a result where only the ‘’Location’’ url is what we need to make another Postman request, this time to GET the result of our blob analysis (for example, we GET the result of a similar url ; <https://westus.cris.ai/api/speechtotext/v2.0/transcriptions/4ebbca96-eca8-48c1-91a4-7186356a6ca9).> When the ‘’Send’’ button of the GET request is pressed, the result rendered gives a url in the body of the response called ‘’channel\_0’’. We click on the link to download the text result of our speech-to-text API request. For this example, we open that file in the form of a simple .txt file. For the purpose of a clean report, we will only put the ‘’Display’’ part of the file in the ‘’Result’’ section of every conversation, because the returned file is very repetitive. The confidence rate of the AI will also be displayed after every quote of a conversation.

1. *Result for every file* :

**Conversation #1, Audio file : Conversation-1.wav : a conversation between 3 people in English**

**Result :** *"Display": "Can you close the door please? Yes, sorry where wet floor going to? I'm going to floor 5? I'm going to floor number 8? How is the weather outside is not too bad? Yeah, it's pretty cool. It's a beautiful there? It is sunny outside. It's cloudy cloudy that's too bad. What's the weather for tomorrow. I don't know probably rainy? Yeah, I think it's raining, yeah, we do." (confidence level : 0.892014) "i have nice shoes oh thanks where did you get them no it's not OK" (confidence level : 0.82800000).*

*Final thoughts* *:* Overall, the result is fairly good, with only a few words that were misinterpreted. Some sentences are not well punctuated and are put into one. The conversation was split in two sections.

**Conversation #2, Audio file : Conversation-2.wav : a conversation between 5 people, in French**

**Result :** *"Display": "Bonjour monsieur quel état je trouve neuvième s' il vous plaît parfait moi 12 e s' il te plaît je voudrais aller au travail." (Confidence level : 0.865355). "Moi j'aimerais aller au 22 e étage s' il vous plaît excellent." (Confidence level : 0.941123). "Vous avez une belle voiture monsieur Jacky lequel modèle c'est une Ferrari rouge il ne 97 une Ferrari ouais sont très rare cette chanson qui a beaucoup d'argent non je voulais voler à ma grand-mère votre grand-mère oui il a eu des problèmes de sérénité mais pourquoi le volume ouais parce que c'est tombé sur le portefeuille puisse aller dans ma poche ?" (Confidence level : 0.88402). "C'est une voiture qu'on parle." (Confidence level : 0.908355). "Non c'est l'argent qui a payé la voiture qui a été volé ?" (Confidence level : 0.888106).*

*Final thoughts* : The French-spoken conversation had a result that was very close to the actual conversation. Only a few words and punctuation were not well interpreted. The conversation has been split into multiple parts, but not necessarily to split the different speakers (some sentences include two speakers).

**Conversation #3, Audio file : Conversation-3.wav : a conversation between 2 people, in English**

**Result :** *"Display": "At the web level do you want to go please? Can I go to level? Yes, I click on the button 5 right now click an? How do you like your noodle? Oh my model is pretty good. I'm another logic. I think you are noodle at pretty sure like too much novel with cheese and tomato salsa." (Confidence level : 0.833242). "it's pretty good yes it's pretty good indeed" (Confidence level : 0.929785). "Me, I prefer pizza." (Confidence level : 0.889929).*

*Final thoughts* : For this one, many words were not rightly interpreted. “Web” was meant to be “what”, “model” was supposed to be “noodle”, “I’m another logic” was supposed to be “I’m a noodle addict” and “novel” was supposed to be “noodle”. However, one of the speakers had a thick accent that changed the usual pronunciation of the words. I would not blame the Azure Speech-to-Text API for these misinterpretations.

**Conversation #4, Audio file : Conversation-4.wav : a conversation between 3 people, in English**

**Result :** *"Display": "Hey Martin, what have you eaten today? I've eaten some potatoes and some mistake. That's about it? Was it good man? At that much. What did you eat?" (Confidence level : 0.903165). "I ate some spaghetti with some margarines. It was awesome. I love pudding margarines on my spaghetti. Even though it may seem a little weird at first, it is actually pretty weird and I ate a pizza with pineapple and pepperoni's that was really good. That's so bad eating pineapples on no don't be disrespectful." (Confidence level : 0.915592).*

*Final thoughts* : This test gave a very faithful result compared to the original. Only a few words like “putting” were interpreted as “pudding”, “steak” mistaken for “mistake”, but overall the result was very close to the actual conversation. Again, the API did not do a great job splitting the conversation between the three speakers, but the speech-to-text was well rendered.

**Conversation #5, Audio file : Conversation-5.wav : a conversation between 5 people, in French**

**Result :** *"Display": "Bonjour monsieur semi chauve Comment allez-vous bien et toi super si vous avez des animaux de compagnie oui j'ai un chat noir j'ai un chat noir aussi obèses moi j'ai acheté mon chat à la petite animalerie qui est en bas là est-ce que vous avez vu le nom c'est quoi le nom le animalerie c'est Miss maf moi en passant j'aime pas les chats donc j'en ai pas acheté je suis allergique bah c'est pas un chien est-ce que t'as d'autres vous avez parlé cher monsieur non j'aime pas les animaux tout court déjà son vidéo je vous demanderai de sortir de l'ascenseur s' il vous plaît ?" (Confidence level : 0.888342). "En décembre ouvre la porte ouvre la porte s' il vous plaît parfait monsieur veuillez sortir." (Confidence level : 0.855118).*

*Final thoughts* : The result was again very close to the original conversation, the API even understood some French contractions like “t’as” which usually is “tu as”. Proper names like “Miss maf” were again well understood by the Speech-to-Text API. A few words like “décembre’’ were not in the original audio conversation, but the users spoke a lot on top of one another, so the result is still pretty impressive. Again, the text was not split between the speakers.

**Conversation #6, Audio file : Conversation-6.wav : a conversation between 2 people, in English**

**Result :** *"Display": "Hey Max, how are you Hey I'm fine Hey? Did you watch the soccer game last night it was awful? Yeah, like every soccer game ever that's right soccer? Sucks so much. I hate soccer, so much that I my friend was playing soccer and after it after he played his game. He wasn't my friend anymore. Yeah, I know the feeling an when I listen." (Confidence level : 0.911308). "My ears cry, Bloods wow that's heavy man I know." (Confidence level : 0.872938).*

*Final thoughts* : This transcription perfectly rendered what was said in the elevator’s conversation 6. Even though the speakers made some grammatical and syntax errors in their speech, the Azure Speech-to-Text API rendered the exact same discourse. Even though the conversation was still not split between the two speakers, the results were beyond expectations.

**Conversation #7, Audio file : Conversation-7.wav : a conversation between 3 people, in English (including one with a southern American accent)**

**Result :** *"Display": "Display": "Damn boy have you seen the rodeo last night. I no I actually didn't go to the radio where was it to be gas man." (Confidence level : 0.864941). "Have you ever been to Vegas? Yeah. I'm from Vegas. Anna I've been to the rodeo. Stadio twice so they, I rarely hear that weird accent from Vegas. Are you sure you are from Vegas now. My mom from an my dad from Texas. But what are you doing? Here nothing absolutely nothing mail so you travel from Las Vegas to Charlottesville?" (Confidence level : 0.912098). "For nothing, what's your name Sir?" (Confidence level : 0.922509).*

*Final thoughts* : Here, we tried pushing the API by using a southern accent to be rendered to text. The result was very faithful to the original conversation, with only a few words “be gas” instead of “Vegas”, “mail” instead of “man”, “stadio” instead of “stadium”. We must say that the speaker had somewhat of a special accent even for a southern American. Even if the conversation was not split, it stood really well the “accent” test.

**Conversation #8, Audio file : Conversation-8.wav : a conversation between 4 people, in English and French (including one English-speaker with a southern American accent)**

**Result :** *"Display": "I'm back at it again boy." (Confidence level : 0.92267). "Back up what?" (Confidence level : 0.852982). "At this elevator, oh OK that's great. I guess have you seen those 2 guys that are standing next to us. You have nice blue eyes? Let's get our tickets kids is literally past speakers. Compara procas quicksort patterns sequence sequence are they talking to me already talking to me, it does." (Confidence level : 0.798521). "I think Chewbacca talking to you." (Confidence level : 0.887748).*

*Final thoughts* : Here again, we pushed the boundaries of our tests. We tested if the API could switch between English and French if recording a conversation between two English-speakers and two French-speakers. The English part of it was very close to the original (even with the southern accent man), but some of the words were still misinterpreted (“up” should’ve been “at”, some words like “is” are missing in the last sentence, for example). The worst confidence level we’ve had yet (0.798521) has been on the half-English, half-French sequence where the French discourse was loosely interpreted with English words that make no sense together, and then the interpreter just stopped to “speech-to-text” until English words could be heard again.

**Conversation #9, Audio file : Conversation-9.wav : a conversation between 5 people, in English**

**Result :** *"Display": "Hey, what's up guys you again. I'm fine. I'm fine. I went to the zoo yesterday man. I never thought I love panda, so much prefer black Panther black Panther is really good and lion are Savage for life. Have you seen the black Panther the movie you mean Yeah, Marvel, German exactly yeah, he died?" (Confidence level : 0.848678). "No, I don't know, don't get it. Are you spoiling Avengers and game I think so man." (Confidence level : 0.799053).*

*Final thoughts* : Even though one of the confidence levels is pretty low, the result rendered by the Azure Speech-to-Text API was very faithful to the original conversation, with only two or three misinterpreted or missing words. We pushed the test by speaking on top of one another at the end and the API still managed to render almost exactly what we said. We still need to mention that a whole sentence, “How are you ?”, was ignored by the API at the beginning of the sequence.

**Conversation #10, Audio file : Conversation-10.wav : a conversation between 5 people, in French and English**

**Result :** *"Display": "Cher monsieur vous êtes pour moi très important et vous êtes une source d'inspiration vous êtes." (Confidence level : 0.938737). "Comme la série sur samedi vous êtes le jaune dans le pissenlit pour ce genre dans un jaune d'oeuf mais quelle belle part elle vous dites monsieur ça me touche vraiment droit au coeur merci le soleil que le soleil plombant vos yeux cher copain laussucq Freddy spirale qui me hum dessine normal de St Barth after des starlettes Cannes bat sunflowers and some collez-y last Word Word chacun boules ?" (Confidence level : 0.784988). "Bien mytho." (Confidence level : 0.666933).*

*Final thoughts* : Here, we did the opposite of the test we realized on “Conversation #8” and sent an “fr-FR” tag in the “Body” of our request to analyze a bilingual conversation. Again, the results were very close to the actual conversation for the French part, but the English discourse was interpreted into French words that make no sense together. Many parts of the English speech were ignored. The confidence levels of the sentences where English could be heard are very low compared to usual results (0.784988 and 0.666933 respectively).

1. *Conclusions and recommendations if the experience was to be repeated* :  
   -The Microsoft Azure Speech-to-Text API was surprisingly efficient in transcribing recorded audio conversations in English and in French to a text format.   
   -A lot of confidence levels were under 90%, but when looking at the text rendered, were surprisingly accurate  
   -The whole accuracy of the English and French speech-to-text was pretty impressive on its own knowing the speakers had a variety of accents (French, French-Canadian, American, Southern American, Latino American).  
   -The real weakness of Microsoft Azure’s Speech-to-Text lies in the fact that it is still unable to recognize different speakers and split the recorded speech every time the discourse switches from speaker to speaker  
   -We tried asking for two different languages in the “Body” of our request, which always returned us a “400 Bad Request”.

-We recommend pushing the API as far as possible with different accents, tones and volumes of voice, because it is surprisingly accurate in rendering the correct speech into text. This kind of experimentation can also inform the developer on what are the limitations of the API

**Methodology**

The main objectives of this project are : creating three user profiles and be able to recognize them in various contexts (mainly conversations). We also then want to be able to identify a selected user profile and be able to put conversational speeches to text with Azure Speech-to-text. The user profiles were created using the Microsoft Cognitive Services documentation methodology with the help of the Postman app allowing us to make the required API calls to create the user profiles and then verify their actual integrity (for example, are the right users correctly identified ?). All the vocal samples used for the profile creations were English or French discourses where the subjects recited song lyrics on a conversational tone for a duration of about 40 to 60 seconds. The ten conversations used for the speaker recognition part of this project were “elevator-related business conversations” in French and in English. Every test detailed below will be analyzed with the following criteria :

1. Main objective
2. Choice of implementation and parameters
3. Result for every file
4. Conclusions and recommendations if the experience was to be repeated

Every step was completed using the Microsoft Cognitive Services documentation that can be found here : <https://westus.dev.cognitive.microsoft.com/docs/services/563309b6778daf02acc0a508/operations/5645c068e597ed22ec38f42e>

**User profile creation**

1. *Main objective* : create three different user profiles, two English and one French. Raphaël Vallée et Maxime Patry are the English-speaking users. Rémi Lemay-Dupont is the French-speaking user.
2. *Choice of implementation and parameters* : we created the user profiles using the Postman app to call the following GET url with a POST function : **https://westus.api.cognitive.microsoft.com/spid/v1.0/identificationProfiles**

The following parameters had to be entered in the Postman application :  
-in the “Headers” section, the following key/values pairings : Content-type / application/json, Accept / application/json, Ocp-Apim-Subscription-Key / a9e3e21c91124fc3bb63a9e8ec9813a4 (related to the Azure Cloud account)  
-in the “Body” section, enter the following text : { “locale” : “en-us” }, select “raw” and “JSON” as a type (the “fr-fr” parameters were entered for user 3, a french-speaking user).

1. *Result for every file* : after pressing the “Send” button, the response gave back an “identificationProfileId” in the following form : { “identificationProfileId” : “49a36324-fc4b-4387-aa06-090cfbf0064f” } which will be needed for the subsequent steps.
2. *Conclusions and recommendations* : this being the first step (and a very simple one), and the documentation being crystal clear on this, it wasn’t hard to complete.

**Enrollment creation**

1. *Main objective* : associate each created profile with a voice do the speaker can later be identified. A 30 to 45 seconds voice sample was recorded to be sent and processed in the instructions below.
2. *Choice of implementation and parameters* : we enrolled the different user profile IDs using the following url with a POST function : **https://westus.api.cognitive.microsoft.com/spid/v1.0/identificationProfiles/1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58/enroll/**

Here, the “1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58” in the url was a profile ID created in the earlier step. The following parameters had to be entered in the Postman application :   
-in the “Headers” section, the following key/values pairings : Content-type / multipart/form-data, Ocp-Apim-Subscription-Key / a9e3e21c91124fc3bb63a9e8ec9813a4.  
-in the “Body” section, the recorded voice sample that we wanted to associate the profile ID to was sent as a binary file that we had to send along with the POST function. The voice samples were recorded using a Blue Yeti microphone as well as the Audacity software. The samples were song lyrics recited by the users on a conversational tone.

1. *Result for every file* : we ran into a few issues while transmitting the audio files. We firstly tried to send a too short (28 seconds) sample to the url, with an 404 error message returned to us telling us the sample was too short. Another try gave us another error message because the file had to be in mono. Another try returned us a 404 message saying the sample had to be in 16K format. We retried the B) steps with a 16K, mono and longer audio sample, which returned a 202 OK response. Therefore, we only ran into issues for user 1’s enrollment, because we recorded longer and appropriately formatted audio samples for the others (about 40 seconds long).
2. *Conclusions and recommendations* : audio files should be at least 35 seconds long to avoid recording it twice. The enrollment result will be verified with in the subsequent section : “Profile fetching”.

**Profile fetching**

1. *Main objective* : see if the profile creation and enrollment procedures now allow us to “GET” a profile and see its properties.
2. *Choice of implementation and parameters* : we fetched the user profiles using the following url with a GET function : **https://westus.api.cognitive.microsoft.com/spid/v1.0/identificationProfiles/1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58**

Where the “1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58” in the url was the profile ID we wanted to fetch, to “GET”. The following key/value parameters had to be entered in the “Headers” section of the Postman request : Ocp-Apim-Subscription-Key / a9e3e21c91124fc3bb63a9e8ec9813a4. The GET request could now be sent to the url.

1. *Result for every file* : All our profiles were rightly created, so our query returned the following in the “Body” section of Postman :

{

"identificationProfileId": "1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58",

"locale": "en-us",

"enrollmentSpeechTime": 56.19,

"remainingEnrollmentSpeechTime": 0,

"createdDateTime": "2019-04-29T16:50:43.513Z",

"lastActionDateTime": "2019-04-29T18:16:34.600Z",

"enrollmentStatus": "Enrolled"

}

The results varied from user to user (because they have different “identificationProfileIds”, because they were not created at the same time and because user 3 has “fr-fr” instead of “en-us” in the “locale” section), but all returned 200 OK responses. They were all “Enrolled”, which means the enrollment did work.

1. *Conclusions and recommendations* : all profiles returned “Enrolled” the first time we tried to “GET” them, so no further recommendation will be suggested here. Very few and very simple steps have to be realized and are all crystal clear in the Microsoft Cognitive Services documentation.

**User identification**

1. *Main objective* : identify if a chosen user (with its profile ID) is present or not in a specific audio sample sent to an identification url. The Microsoft API will then analyze if the profile ID X can actually be heard in the sample and other data that we will describe to you below.
2. *Choice of implementation and parameters* : we sent the user profiles to be identified using the following url with a POST function : **https://westus.api.cognitive.microsoft.com/spid/v1.0/identify?identificationProfileIds={identificationProfileIds}**

Where the “{identificationProfileIds}” in the url was the profile ID we wanted to identify, which has to be entered as a key/value pair in the “Params” section of the Postman application. The following key/value parameters had to be entered in the Postman request :   
-in the “Params” section, identificationProfileIds/1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58 where “1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58” is the user 1’s profile ID (for the purpose of this example). It should be noted that up to 10 profiles can be identified at the same time in a single request.   
-in the “Headers” section, the following key/values pairings : Content-type / multipart/form-data, Ocp-Apim-Subscription-Key / a9e3e21c91124fc3bb63a9e8ec9813a4.  
-in the “Body” section, a new recorded voice sample that we wanted to identify the profile ID with was sent as a binary file that we had to send along with the POST function. The voice samples were recorded using a Blue Yeti microphone as well as the Audacity software. The “ID-verification” samples were different samples from those used in the “Enrollment creation” section and were improvised speeches or other song lyrics recited on a conversational tone. The samples were all 35 seconds or more.

1. *Result for every file* : When we pressed “Send”, a 202 accepted response appeared in the Postman dashboard and new headers were rendered in response :

**Cache-Control →**no-cache

**Pragma →**no-cache

**Content-Length →**0

**Expires →**-1

**Operation-Location →**https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/5dee28ac-bf86-4b7c-9584-cd99ef201f31

**X-AspNet-Version →**4.0.30319

**X-Powered-By →**ASP.NET

**apim-request-id →**3dadcfed-d12c-406c-b466-554a596e0273

**Strict-Transport-Security →**max-age=31536000; includeSubDomains; preload

**x-content-type-options →**nosniff

**Date →**Wed, 01 May 2019 19:49:44 GMT

Here, the only thing that we need and that varies from user to user is the “Operation-Location” url which will then be used in the subsequent “User confirmation” step to check the result of the user identification.

1. *Conclusions and recommendations* : As the Microsoft Cognitive Services documentation puts it, each recorded speech that we want to verify should be at least 30 seconds. The conclusion of this step will then be clearer in the subsequent step where we get to check the status of our identification.

**User confirmation**

1. *Main objective* : Check the status of the precedent step’s request to see if the user(s) has rightly been identified.
2. *Choice of implementation and parameters* : we sent the “operationId” used to identify a user using the following url with a GET function :

<https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/5dee28ac-bf86-4b7c-9584-cd99ef201f31>

Where the whole url that we want to GET is the result of the”Operation-Location” header rendered in the precedent step. The only key/value pairing that needs to be entered here is the Ocp-Apim-Subscription-Key / a9e3e21c91124fc3bb63a9e8ec9813a4. An “operationId” key/value pairing then appears by default in the “Params” section of the Postman request.

1. *Result for every file* : When the “Send” button was pressed, the response returned in the “Body” part of every request was similar to the following :

{

"status": "succeeded",

"createdDateTime": "2019-04-29T18:58:21.4717298Z",

"lastActionDateTime": "2019-04-29T18:58:23.627657Z",

"processingResult": {

"identifiedProfileId": "1bb58f2d-7db2-4b5d-8d58-7a5ea7aece58",

"confidence": "High"

}

}

Where only the time stamps and “identifiedProfileId” changed from user to user. The confidence level for every file was “High”. We tried sending a sample where another user was speaking, but we asked for a different user to be identified. This test gave the following result :

{

"status": "succeeded",

"createdDateTime": "2019-04-29T19:08:21.4717298Z",

"lastActionDateTime": "2019-04-29T19:08:23.627657Z",

"processingResult": {

"identifiedProfileId": "00000000-0000-0000-0000-000000000000",

"confidence": "Normal"

}

}

The Microsoft Cognitive Services documentation is clear on this kind of result (<https://westus.dev.cognitive.microsoft.com/docs/services/563309b6778daf02acc0a508/operations/5645c725ca73070ee8845bd6>) : the API cannot recognize the provided profile in the provided audio sample.

1. *Conclusions and recommendations* : This step being the final one of the “Speaker Recognition” part of our project, a few conclusions and recommendations can be provided :   
   -CLOSELY read the Microsoft Cognitive Services documentation and follow all the steps and you should have no problem. Every issue that we ran into was us going too fast and passing over clear instructions like “samples should be AT LEAST 30 seconds long, in mono and in the 16K format”.   
   -All the expected responses and scenarios are provided in the documentation. You are having a 404 response or an unsatisfying response in the body of a 202 response ? Chances are the exact result can be found in the documentation to tell you what may have been wrong with your request.

-There can be a delay to “GET” the identified user confirmation. Simply sending the operation ID url another time did the trick.

-Many operations are possible with the Speaker Recognition API, make sure you are using the right url to get the wanted request (for example, do not mix-up “Identification Profile” and “Verification Profile”, which are two completely different methods).

**Speech-to-Text**

1. *Main objective* : render audio files (conversations recorded in elevators) into a readable text form.
2. *Choice of implementation and parameters* : we sent the conversation [x] audio files to the following url :

[https://westus.cris.ai/api/speechtotext/v2.0/transcriptions](https://westus.api.cognitive.microsoft.com/spid/v1.0/operations/5dee28ac-bf86-4b7c-9584-cd99ef201f31)

The url and file will be sent as a POST request. The following key/value parameters had to be entered in the Postman request :   
-in the “Headers” section, the following key/values pairings : Content-type / application/json , Ocp-Apim-Subscription-Key / 759435af27c74cf1b2500edbb9eb51d6.  
-in the “Body” section, the following text :

{

"recordingsUrl": "https://respeechtotextstorage.blob.core.windows.net/rocketconversations/Conversation-1.wav",

"models": [],

"locale": "en-US",

"name": "Transcription using locale en-US",

"description": "An optional description of the transcription.",

"properties": {

"ProfanityFilterMode": "Masked",

"PunctuationMode": "DictatedAndAutomatic"

}

}

Where the “recordingsUrl” is the url of the Azure blob storage location containing the conversation [x] files. It is the only part of the request that must be changed from conversation to conversation, except from the “Locale” parameter that has to be changed to “fr-fr” if the conversation we want transcripted was in French.

1. *Result for every file* : the only part of the response we need is the “Location” url (for example : “https://westus.cris.ai/api/speechtotext/v2.0/transcriptions/d89d0420-2fcd-4ebc-85b0-b77cdc7a3313” that will be needed in the subsequent step to GET the speech-to-text result. The GET will only require the Azure Ocp-Apim-Subscription-Key and the url from the upper POST method.
2. *Conclusions and recommendations* : The upper operations are all it takes to analyze a file. Please note that other operations (like storing audio files in an Azure Blob Storage) have been taken for granted in the present example. Recommendation : do not try to analyze a French-speaking conversation with the “en-us” “Locale” parameters entered in the request.