Speech-to-Text API: Qwik Start

Task 1. Create an API key

Since you'll be using `curl` to send a request to the Speech-to-Text API, you need to generate an API key to pass in our request URL.

1. To create an API key, click **Navigation menu** > **APIs & services** > **Credentials**.

2. Then click **Create credentials**.

3. In the drop down menu, select **API key**.

4. Copy the key you just generated and click **Close**.

Now that you have an API key, save it as an environment variable to avoid having to insert the value of your API key in each request.

To perform the next steps, connect using SSH to the instance provisioned for you.

1. In the **Navigation menu**, select **Compute Engine**. You should see a `linux-instance` listed in the **VM instances** window.

2. Click on the **SSH** button in line with the `linux-instance`. You will be brought to an interactive shell.

3. In the command line, enter in the following, replacing `<YOUR_API_KEY>` with the API key you copied from previously generated:

```
export API_KEY=<YOUR_API_KEY>
```

Copied!

content_copy
You remain in this SSH session for the rest of the lab.

Task 2. Create your Speech-to-Text API request

Note: You will use a pre-recorded file that's available on Cloud Storage: `gs://cloud-samples-tests/speech/brooklyn.flac`. Listen to the audio file before sending it to the Speech-to-Text API.

1. Create `request.json` in the SSH command line. You'll use this to build your request to the Speech-to-Text API:

```
touch request.json
```

Copied!

content_copy

2. Open the `request.json`:

```
nano request.json
```

Copied!

content_copy

Note: You can use your preferred command line editor (`nano`, `vim`, `emacs`) or `gcloud`. This lab will provide instructions for `nano`.

3. Add the following to your `request.json` file, using the `uri` value of the sample raw audio file:

```
{
  "config": {
      "encoding":"FLAC",
      "languageCode": "en-US"
  },
  "audio": {
      "uri":"gs://cloud-samples-tests/speech/brooklyn.flac"
  }
}
```

Copied!

content_copy

4. Press `control` + `x` and then `y` to save and click `Enter` to close the `request.json` file.

The request body has a `config` and `audio` object.

In `config`, you tell the Speech-to-Text API how to process the request.
The `encoding` parameter tells the API which type of audio encoding you're using while the file is being sent to the API. `FLAC` is the encoding type for .raw files. Learn more about encoding types in the RecognitionConfig Guide.
There are other parameters you can add to your `config` object, but `encoding` is the only required one.

In the `audio` object, you pass the API the uri of the audio file in Cloud Storage

Task 3. Call the Speech-to-Text API

1. Pass your request body, along with the API key environment variable, to the Speech-to-Text API with the following `curl` command (all in one single command line):

```
curl -s -X POST -H "Content-Type: application/json" --data-binary @request.json \
"https://speech.googleapis.com/v1/speech:recognize?key=${API_KEY}"
```

Copied!

content_copy

Your response should look something like this:

```
{
  "results": [
    {
      "alternatives": [
        {
          "transcript": "how old is the Brooklyn Bridge",
          "confidence": 0.98267895
        }
      ]
    }
  ]
}
```

The `transcript` value will return the Speech-to-Text API's text transcription of your audio file, and the `confidence` value indicates how sure the API is that it has accurately transcribed your audio.

You'll notice that you called the `syncrecognize` method in the request above. The Speech-to-Text API supports both synchronous and asynchronous speech to text

transcription. In this example you sent it a complete audio file, but you can also use the `syncrecognize` method to perform streaming speech to text transcription while the user is still speaking.

You created a Speech-to-Text API request then called the Speech-to-Text API.

2. Run the following command to save the response in a `result.json` file:

```
curl -s -X POST -H "Content-Type: application/json" --data-binary
@request.json \
"https://speech.googleapis.com/v1/speech:recognize?key=${API_KEY}" >
result.json
```