VoiceBase speech analytics API.

Release 3.0

VoiceBase

May 26, 2021

Overview

VoiceBase provides an API for speech recognition and analytics for developers looking to get the most from their audio and video data.

If you are getting started, we recommend the Hello World Developer guide as a practical primer, and the API Overview as a high-level introduction.

More information and guidance can be found in the:

- *How-To Guides*
- *API Reference*
- VoiceBase account

CHAPTER 1

API Overview

VoiceBase provides APIs for speech recognition and speech analytics, offering customers a wide variety of insights about the data in their audio files.

1.1 Authorization

The VoiceBase /v3 REST API is secured using OAuth Bearer tokens. Bearer tokens are issued and managed through your VoiceBase account.

For details on creating your first token, see the How to Get Your Bearer Token section of the Hello, World How-To Guide.

1.2 Core Workflow

The core workflow of the API is to generate transcriptions and analytics from voice recordings. This workflow is asynchronous, and a typical usage is to:

1. Upload a voice recording, starting the transcription and analysis process
2. Wait for completion, using periodic polling for status or callbacks
3. Process or retrieve results, including the transcript, keywords, topics and predictions

To achieve scalability, this workflow runs for multiple recordings in parallel.

1.3 REST Call Flow

A typical pattern of REST API calls to accomplish the workflow is to:

```
POST /media
```

The body of POST request is MIME multipart, with three parts:

One of:

- *media*: the voice recording attachment or,
- *mediaUrl*: URL where the API can retrieve the voice recording

and optionally:

- *configuration*: (optional) a JSON object with customized processing instructions
- *metadata*: (optional) a JSON object with metadata

The API will return a unique identifier for the new object, called a **mediaId**.

```
GET /media/{mediaId}/progress
```

This call retrieves status and progress information. When the processing is finished, the transcript and analytics can be retrieved.

```
GET /media/{mediaId}
```

The API supports Callbacks instead of polling for status, and this pattern is recommended for production integrations.

1.4 Getting Started

The Hello, World How-To Guide provides a practical introduction to getting started with the VoiceBase V3 REST API.

CHAPTER 2

Hello, World

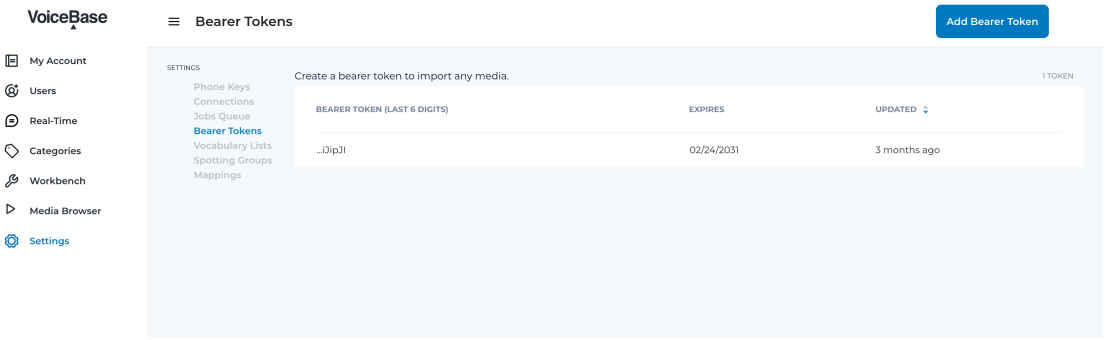
This Quickstart tutorial gets you up and running with your first transcription and speech analytics.

The examples assume you have these prerequisites:

- *Your Bearer token*
- *jq* for working with JSON (see: *A Quick Note on Tools* for details)
- *curl* (comes with most Linux systems, see: *A Quick Note on Tools* for details)

2.1 How to Get Your Bearer Token

Sign into your VoiceBase account. Go to Settings/Bearer Tokens and click *Add Bearer Token*.



Save your token in a secure location; it won't be visible in your account after its initial creation.

2.2 Verify your Bearer Token and tools are working

Start by verifying your Bearer token is working by running the following from a command prompt. Please remember to insert your Bearer token after *TOKEN*= in the first line of the example. You can also find more detailed instructions

on *Understanding Your First Request* below.

```
export TOKEN= # Insert your VoiceBase Bearer token after TOKEN=

curl https://apis.voicebase.com/v3/media \
  --header "Authorization: Bearer TOKEN" \
  | jq
```

You should see a response like this:

```
{
  "_links": {
    "self": {
      "href": "/v3/media"
    }
  },
  "media": []
}
```

2.3 Upload a media file for transcription and analysis

To upload a recording for transcription and analysis, POST to /media with the recording as an attachment named media (you can also provide a URL to your recording instead using the form field 'mediaUrl').

Using local media:

```
curl https://apis.voicebase.com/v3/media \
  --form 'media=@hello-world.mp3' \
  --header "Authorization: Bearer ${TOKEN}" \
  | tee media-post.json \
  | jq .
```

Using a remote media URL:

```
curl https://apis.voicebase.com/v3/media \
  --form 'mediaUrl=http://myServer.com/mediaFile.mp3' \
  --header "Authorization: Bearer ${TOKEN}" \
  | tee media-post.json \
  | jq .
```

The response includes a *mediaId* (assigned by the API) and a status of *accepted*.

```
{
  "_links": {
    "self": {
      "href": "/v3/media/10827f19-7574-4b54-bf9d-9387999eb5ec"
    },
    "progress": {
      "href": "/v3/media/10827f19-7574-4b54-bf9d-9387999eb5ec/progress"
    },
    "metadata": {
      "href": "/v3/media/10827f19-7574-4b54-bf9d-9387999eb5ec/metadata"
    }
  },
  "mediaId": "10827f19-7574-4b54-bf9d-9387999eb5ec",
  "status": "accepted",
```

(continues on next page)

(continued from previous page)

```
  "dateCreated": "2021-04-22T18:23:02Z",
  "dateFinished": "2021-04-22T18:23:58Z",
  "mediaContentType": "audio/mp3",
  "length": 10031,
  "metadata": {}
}
```

You can poll for status until the processing is done (for production, we recommend using Callbacks).

```
export MEDIA_ID=$( cat media-post.json | jq --raw-output .mediaId )
export STATUS=$( cat media-post.json | jq --raw-output .status )

while [[ ${STATUS} != 'finished' && ${STATUS} != 'failed' ]]; do
  sleep 1
  STATUS=$(
    curl https://apis.voicebase.com/v3/media/${MEDIA_ID}/progress \
      --header "Authorization: Bearer ${TOKEN}" \
      | jq --raw-output .progress.status
  )
  echo "Got status: ${STATUS} for mediaId: ${MEDIA_ID} on $( date )"
done
```

2.4 Get your transcript and analytics

You can retrieve the JSON version of the transcript and all analytics with a simple API call.

```
curl https://apis.voicebase.com/v3/media/${MEDIA_ID}/transcript \
  --header "Authorization: Bearer ${TOKEN}" \
  | jq .
```

You can also retrieve a plain-text version using *transcript/text* and the *Accept* HTTP header.

```
curl https://apis.voicebase.com/v3/media/${MEDIA_ID}/transcript/text \
  --header 'Accept: text/plain' \
  --header "Authorization: Bearer ${TOKEN}"
```

2.5 Understanding Your First Request

The root URL of the VoiceBase V3 API is **https://apis.voicebase.com/v3**. Every recording you submit for analysis appears in the **/media** collection and is viewable in the 'Media Browser' tab. The first request is to GET the **/media** collection (which will be empty when you first sign up). Pagination default limit is set to 100.

```
export TOKEN= # Insert your VoiceBase Bearer token after TOKEN=

curl https://apis.voicebase.com/v3/media?limit=10 \
  --header "Authorization: Bearer ${TOKEN:?'(hint: insert your token after export␣
↪TOKEN=)'}" \
  | jq
```

If you're running this for the first time, the API returns:

```
{
  "_links": {
    "self": {
      "href": "/v3/media"
    }
  },
  "media": []
}
```

All successful responses from the API will include an *_links* section with HAL metadata that helps navigate the API.

The *media* section the list of media in your account. If you have previously uploaded media, it will appear in the list.

```
{
  "media": []
}
```

2.6 Understanding Your First Upload

The next step is to upload a recording to the API for transcription and analysis, but making a POST to /media, with the recording as an attachment named media.

```
curl https://apis.voicebase.com/v3/media \
  --form media=@hello-world.mp3 \
  --header "Authorization: Bearer ${TOKEN}" \
  | jq
```

When you add the *–form media=@filename.mp3* parameters, *curl* automatically sets the HTTP method to *POST* and the *Content-Type* to *multipart/form-data*. This is equivalent to the more explicit:

```
curl https://apis.voicebase.com/v3/media \
  --form media=@hello-world.mp3 \
  --header "Authorization: Bearer ${TOKEN}" \
  --request POST \
  --header "Content-Type: multipart/form-data" \
  | jq
```

Finally, many operations will rely on providing a configuration JSON attachment with additional processing instructions. Omitting the attachment is equivalent to including the following default configuration:

```
curl https://apis.voicebase.com/v3/media \
  --form media=@hello-world.mp3 \
  --form configuration='{}' \
  --header "Authorization: Bearer ${TOKEN}" \
  | jq
```

The 'How-to' guides in this documentation show configurations for each feature of the VoiceBase platform, including an overall sample configuration.

2.7 A Quick Note on Tools

- **curl**: The examples in this documentation make use of curl for making HTTP requests to the API.
- **jq**: The jq tool helps parse JSON responses and work with JSON data.

CHAPTER 3

Transcripts

Once processing is complete, transcripts can be retrieved in several formats.

3.1 JSON Transcript

Retrieve a JSON-formatted transcript with metadata using a `GET` against the `transcript` resource under the `media` item.

Make a GET on the /media/$MEDIA_ID/transcript resource.

```
curl https://apis.voicebase.com/v3/media/$MEDIA_ID/transcript
  --header "Authorization: Bearer ${TOKEN}"
```

3.1.1 Example Response

In this example, the agent spoke first, and said "Hello" and the caller spoke second and said "Hi". Speaker identification is enabled by multi-channel audio, where each channel is associated with a specific speaker. For more information on how to use multi-speaker audio, see the stereo section.

How to read it

- "p" = Position (word # in the transcript)
- "c" = Confidence (A value between 0-1 that relates to the confidence percent ex: 0.88 = 88%. When the metadata flag is present confidence contains an arbitrary value.)
- "s" = Start time (milliseconds)
- "e" = End time (milliseconds)
- "w" = The word itself (When the metadata flag is present "w" refers to the speaker)
- "m" = metadata (In this case when "m": "turn" it is detecting a change in speaker)

```
{
  "mediaId": "bc14632d-e81b-4673-992d-5c5fb6573fb8",
  "status": "finished",
  "dateCreated": "2017-06-22T19:18:49Z",
  "dateFinished": "2017-06-22T19:19:27Z",
  "mediaContentType": "audio/x-wav",
  "length": 10031,
  "transcript": {
     "words":[
        {
           "p":1,
           "s":2200,
           "c":1.0,
           "e":2350,
           "w":"agent",
           "m":"turn"
        },
        {
           "p":2,
           "s":2200,
           "c":0.537,
           "e":2300,
           "w":"Hello"
        },
        {
          "p":3,
          "s":2300,
          "c":1.0,
          "e":2300,
          "w":".",
          "m":"punc"
        },
        {
           "p":4,
           "s":2400,
           "c":0.1,
           "e":2550,
           "w":"caller",
           "m":"turn"
        },
        {
           "p":5,
           "s":2400,
           "c":0.975,
           "e":2500,
           "w":"Hi"
        }
     ]
  }
}
```

3.2 Plain Text Transcript

3.2.1 Example Response

The plaintext response will contain only the words from the transcript, without formatting.

```
To find the source of success we started at work we asked people to identify who they␣
↪thought
were their most effective colleagues in fact over the past twenty five years we have␣
↪asked
over twenty thousand people to identify the individuals in their organizations who␣
↪could
really get things done
```

3.2.2 Example cURL

To download a transcript as plain text, make a GET on the /media/$MEDIA_ID/transcript/text resource specifying an `Accept` HTTP header with the value `text/plain`.

```
curl https://apis.voicebase.com/v3/media/$MEDIA_ID/transcript/text \
    --header "Authorization: Bearer ${TOKEN}" \
    --header "Accept: text/plain"
```

Alternatively, the text transcript can be retrieved with the JSON transcript by adding the `includeAlternateFormat` query parameter in the request set to the value 'text'

```
curl https://apis.voicebase.com/v3/media/$MEDIA_ID/transcript?
↪includeAlternateFormat=text \
    --header "Authorization: Bearer ${TOKEN}" \
    --header "Accept: application/json"
```

In this case, the transcript will contain the additional section 'alternateFormats', the 'data' attribute contains the text transcript encoded in Base64.

```
{
  "alternateFormats": [
    {
      "format": "text",
      "contentType": "text/plain",
      "contentEncoding": "Base64",
      "charset": "utf-8",
      "data":
↪"VG8gZmluZCB0aGUgc291cmNlIG9mIHN1Y2Nlc3Mgd2Ugc3RhcnRlZCBhdCB3b3JrIHdlIGFza2VkIHBlb3BsZSB0byBpZGVud
↪"
    }
  ]
}
```

3.3 SRT transcript

To retrieve a transcript as a SRT file which is useful for closed captioning or timing the transcript with the audio, make a GET on the /media/$MEDIA_ID/transcript/srt resource specifying an `Accept` HTTP header with the value `text/srt`.

The closed captioning section has a detailed discussion of the SRT transcript format.

3.3.1 Example Response

```
1
00:00:00,05 --> 00:00:05,81
To find the source of success we started
at work we asked people to identify who

2
00:00:05,82 --> 00:00:10,90
they thought were their most effective
colleagues in fact over the past twenty five

3
00:00:10,91 --> 00:00:16,13
years we have asked over twenty thousand
people to identify the individuals in their

4
00:00:16,14 --> 00:00:20,93
organizations who could really get things
done we wanted to find those who were not
```

3.3.2 Example cURL

```
curl https://apis.voicebase.com/v3/media/$MEDIA_ID/transcript/srt \
    --header "Authorization: Bearer ${TOKEN}" \
    --header "Accept: text/srt"
```

Alternatively, the SRT transcript can be retrieved with the JSON transcript by adding the `includeAlternateFormat` query parameter in the request.

```
curl https://apis.voicebase.com/v3/media/$MEDIA_ID/transcript?
↪includeAlternateFormat=srt \
    --header "Authorization: Bearer ${TOKEN}" \
    --header "Accept: application/json"
```

In this case, the SRT transcript is returned encoded with Base64 within the JSON

```
{
  "alternateFormats": [
    {
      "format": "srt",
      "contentType": "text/srt",
      "contentEncoding": "Base64",
      "charset": "utf-8",
      "data":
↪"MQ0KMDA6MDA6MDAsMDUgLS0+IDAwOjAwOjA1LDgxDQpUbyBmaW5kIHRoZSBzb3VyY2Ugb2Ygc3VjY2VzcyB3ZSBzdGFydGVkD(
↪"
    },
  ]
}
```

3.4 WebVTT transcript

WebVTT is a W3C standard for displaying timed text in HTML 5 utilizing the element. To retrieve a transcript as a WebVTT file which is useful for closed captioning or timing the transcript with the audio, make a GET on the /media/$MEDIA_ID/transcript/webvtt resource specifying an `Accept` HTTP header with the value `text/vtt`.

The closed captioning section has a detailed discussion of the WebVTT and SRT transcript formats.

3.4.1 Example Response

```
WEBVTT

1
00:00:00.05 --> 00:00:05.81
To find the source of success we started
at work we asked people to identify who

2
00:00:05.82 --> 00:00:10.90
they thought were their most effective
colleagues in fact over the past twenty five

3
00:00:10.91 --> 00:00:16.13
years we have asked over twenty thousand
people to identify the individuals in their

4
00:00:16.14 --> 00:00:20.93
organizations who could really get things
done we wanted to find those who were not
```

3.4.2 Example cURL

```
curl https://apis.voicebase.com/v3/media/$MEDIA_ID/transcript/webvtt \
    --header "Authorization: Bearer ${TOKEN}" \
    --header "Accept: text/vtt"
```

Alternatively, the SRT transcript can be retrieved with the JSON transcript by adding the `includeAlternateFormat` query parameter in the request.

```
curl https://apis.voicebase.com/v3/media/$MEDIA_ID/transcript?
↪includeAlternateFormat=webvtt \
    --header "Authorization: Bearer ${TOKEN}" \
    --header "Accept: application/json"
```

In this case, the WebVTT transcript is returned encoded with Base64 within the JSON

```
{
  "alternateFormats": [
    {
      "format": "webvtt",
      "contentType": "text/vtt",
```

(continues on next page)

(continued from previous page)

```
      "contentEncoding": "Base64",
      "charset": "utf-8",
      "data":
↪"MQ0KMDA6MDA6MDAsMDUgLS0+IDAwOjAwOjA1LDgxDQpUbyBmaW5kIHRoZSBzb3VyY2Ugb2Ygc3VjY2VzcyB3ZSBzdGFydGVkD(
↪"
    },
  ]
}
```

3.5 Retrieving transcript in several formats in a single request

You may specify several formats to be returned in the same request, just add 'includeAlternateFormat' in the query string as many times as needed:

```
export FIELDS='?includeAlternateFormat=srt&includeAlternateFormat=webvtt'

curl https://apis.voicebase.com/v3/media/$MEDIA_ID/transcript${FIELDS} \
    --header "Authorization: Bearer ${TOKEN}" \
    --header "Accept: application/json"
```

Valid formats are: text, srt, dfxp, webvtt

CHAPTER 4

Aligner

VoiceBase allows you to align a human edited or new machine transcript with a previously run machine-generated transcript.

4.1 Examples

Note: Export your api token prior to running any of the following examples.

```
export TOKEN='Your Api Token'
```

4.1.1 Correcting a machine transcript and re-processing analytics and callbacks.

First, make a POST request to the /media resource.

```
curl -v -s https://apis.voicebase.com/v3/media \
  --header "Authorization: Bearer ${TOKEN}" \
  --form media=@musicVoiceTone.wav \
  --form configuration='{}'
```

The response contains the mediaId you will use when aligning (e.g., 7eb7964b-d324-49cb-b5b5-76a29ea739e1, as below):

```
  {
    "_links": {
      "self": {
        "href": "/v3/media/7eb7964b-d324-49cb-b5b5-76a29ea739e1"
      },
      "progress": {
        "href": "/v3/media/7eb7964b-d324-49cb-b5b5-76a29ea739e1/progress"
      },
      "metadata": {
```

(continues on next page)

(continued from previous page)

```
        "href": "/v3/media/7eb7964b-d324-49cb-b5b5-76a29ea739e1/metadata"
      }
    },
    "mediaId": "7eb7964b-d324-49cb-b5b5-76a29ea739e1",
    "status": "accepted",
    "dateCreated": "2017-06-22T18:23:02Z",
    "dateFinished": "2017-06-22T18:23:58Z",
    "mediaContentType": "audio/mp3",
    "length": 10031,
    "metadata": {}
  }
```

Export `MEDIA_ID`

```
export MEDIA_ID='7eb7964b-d324-49cb-b5b5-76a29ea739e1'
```

Make a request to the `/media/$MEDIA_ID/transcript/text` resource, including the `Accept: text/plain` header to retrieve the text transcript.

```
curl -v -s https://apis.voicebase.com/v3/media/$MEDIA_ID/transcript/text \
  --header "Authorization: Bearer ${TOKEN}" --header "Accept: text/plain"
```

You may receive a 404 response indicating that the alignment of the new transcript with the original transcript and the recalculation of analytics and predictions is not complete.

```
{
    "status": 404,
    "warnings": {
        "message": "Transcripts only become available when a media item has status␣
↪finished."},
    "reference":"e072fda3-d66e-48a6-9f9e-643937165e39"
}
```

When processing is complete on the media, you will receive the plain text transcript transcribed by VoiceBase. Save it to an ascii text file named `transcript.txt`.

```
Old transcript in file.
```

You notice that the names are garbled, so you edit the plain text transcript in the file with your corrections.

```
New text transcript in file.
```

Now make a POST request to the `/media/${MEDIA_ID}` including a `configuration` and a `transcript` attachment.

```
curl -v -s https://apis.voicebase.com/v3/media/$MEDIA_ID \
  --header "Authorization: Bearer ${TOKEN}" \
  -X POST \
  --form configuration='{}' \
  --form transcript=@transcript.text
```

Finally, make a GET request on the `/media/${MEDIA_ID}` resource to download the latest aligned transcripts and configured analytics and predictions.

```
curl -v -s https://apis.voicebase.com/v3/media/$MEDIA_ID \
  --header "Authorization: Bearer ${TOKEN}"
```

Note that the simple act of including a transcript with the POST triggers the alignment configuration.

CHAPTER 5

Callbacks

VoiceBase can optionally make a callback request to a specific url when media upload processing is complete.

5.1 Uploading Media with Callbacks Enabled

To request a processing-completed callback from VoiceBase, include a JSON configuration attachment with your media POST. The configuration attachment should contain the publish key, for example as below:

Notes:

- Each callback in the set below will result in a unique call to the specified server upon media completion.
- If "type" is not specified: "analytics" is assumed and all sections will be included (transcript, knowledge, metadata, prediction, streams, spotting)

```
{
  "publish": {
    "callbacks": [
      {
        "url" : "https://example.org/callback"
      },
      {
        "url" : "https://example.org/callback",
        "method" : "POST",
        "include" : [ "knowledge", "metadata", "prediction", "spotting", "streams",
↪"transcript", "metrics", "speakerSentiments", "conversation", "categories",
↪"messages" ]
      },
      {
        "url" : "https://example.org/callback/vtt",
        "method" : "PUT",
        "type" : "transcript",
        "format" : "webvtt"
```

(continues on next page)

(continued from previous page)

```
      },
      {
        "url" : "https://example.org/callback/srt",
        "method" : "PUT",
        "type" : "transcript",
        "format" : "srt"
      },
      {
        "url" : "https://example.org/callback/media",
        "method" : "PUT",
        "type" : "media",
        "stream": "original"
      }
    ]
  }
}
```

5.1.1 Configuration Description

- `publish` : object for publish-specific configuration
  - `callbacks` : array of callbacks, with one object per callback desired
    - `[n]` : callback array element
      - `url` : the https url for delivering a callback notification
      - `method` : the HTTPS method for callback delivery, with the following supported values:
      - `POST`: deliver callbacks as an HTTP POST (default)
      - `PUT`: deliver callbacks as an HTTP PUT
      - `type` : Type of results to callback
      - `analytics`: the result analytics in json format (default)
      - `transcript`: the transcript in specific format, see `format` attribute
      - `media`: the media, see `stream` attribute
      - `format` : the format of the callback of type 'transcript'
      - `json`: transcript in json format (application/json)
      - `text`: transcript in text format (text/plain)
      - `webvtt`: transcript in webvvt format (text/vtt)
      - `srt`: transcript in srt format (text/srt)
      - `dfxp`: transcript in dfxp format (application/ttaf+xml)
      - `include` : array of data to include with the callback of type 'analytics', with the following supported values. If "include" is omitted the callback will return all sections:
      - `metadata` : include supplied metadata, often useful for correlated to records in a different system
      - `transcript`: include transcripts for the media on all formats.
      - `knowledge` : include topics and keywords for the media

- `prediction` : include prediction information for the media based on supplied predictors
- `spotting` : include spotted groups information for the media based on supplied keyword groups
- `streams` : include links for the media
- `stream` : the media stream of type 'media'
- `original`: the original media
- `redacted-audio`: the redacted media

The following table shows the Content-Type of the requests when we call you back with the results. This header is important if you are providing pre-signed URLs generated for a service like Amazon AWS S3

Type	Format	Content-Type header
analytics	•	application/json
transcript	text	text/plain
transcript	json	application/json
transcript	srt	text/srt
transcript	dfxp	application/ttaf+xml
transcript	webvtt	text/vtt
media	•	application/octet-stream

If the media fails to be processed, you will only receive the callbacks of type "analytics" describing the error that prevented the media from being processed. Callbacks of type "transcript" and "media" will be ignored.

5.2 Example cURL Request with Callback

For example, to upload media from a local file called recording.mp3 and receive a callback at https://example.org/callback, make the following POST request using curl, or an equivalent request using a tool of your choice:

```
curl https://apis.voicebase.com/v3/media \
    --header "Authorization: Bearer $TOKEN" \
    --form media=@recording.mp3 \
    --form configuration='{
          "publish": {
            "callbacks": [
              {
                "url" : "https://example.org/callback",
                "method" : "POST",
                "include" : [ "transcript", "knowledge", "metadata", "prediction",
↪"streams" ]
              }
            ]
          }
      }'
```

When using callbacks, you can still query the status of the media processing using a GET request to /v3/media/{mediaId}.

5.2.1 Callback Retry Logic

If a success response is not achieved on the first attempt, VoiceBase will retry the callback URL provided according to the following schedule until a success response or the schedule ends. Each of these 7 scheduled attempts will include 3 attempts one immediately after each other for a maximum 21 attempts to the endpoint over 8 hours.

Retry number	Time since last retry	Time since initial try
1	Immediate	0
2	15 min	15 min
3	30 min	45 min
4	1 hour	1 hour 45 min (105 min)
5	2 hours	3 hours 45 min (223 min)
6	4 hours	7 hours 45 min (465 min)
7	8 hours	15 hours 45 min (945 min)

5.2.2 IP Whitelist

All egress traffic flows from VoiceBase servers through the following (currently) NAT gateways:

IPs	VoiceBase US Instance	VoiceBase EU Instance
1	52.6.244.43	34.248.80.158
2	52.6.208.178	52.210.18.246
3	52.2.171.140	54.72.141.175

5.3 Callback Data

When media processing is complete, VoiceBase will call back your specified endpoint by making an HTTPS POST request. The body is a JSON object with the following data:

```
{
 "_links": {},
 "formatVersion": "3.0.7",
 "mediaId": "efbb8c49-87f0-4f6c-9ce9-781599918f8c",
 "accountId": "710d1652-63a4-4355-8e9a-523ddacd3066",
 "accountName": "ACME Inc",
 "status": "finished",
 "dateCreated": "2017-04-28T21:12:55.563Z",
 "dateFinished": "2018-07-19T11:34:45.134Z",
 "timeToLiveInSeconds": 1200,
 "expiresOn": "2018-10-04T00:41:06.145Z",
 "metadata": {
   "title": "Inbound call 2018-07-01 from 15081231234",
   "description": "Inbound call to 1-800-599-1234, ACME Support Center",
   "externalId": "inbound-dd9e7002-4a5a-43b3-bd46-73a66362db29",
   "extended": {
     "anything": "goes here",
     "nested": {
       "is": 0,
       "also": 0,
       "accepted": 0
     }
```

(continues on next page)

(continued from previous page)

```
    }
  },
  "mediaContentType": "audio/mpeg",
  "length": 66900, // Duration in ms of the audio
  "knowledge": {
    "keywords": [
      {
        "keyword": "Microsoft",
        "relevance": 0.433,
        "mentions": [
          {
            "speakerName": "Speaker 1",
            "occurrences": [
              {
                "s": 34234,
                "e": 34234,
                "exact": "Microsoft"
              }
            ]
          }
        ]
      }
    ],
    "topics": [
      {
        "topicName": "Algorithms",
        "relevance": 0.4353,
        "subtopics": [],
        "keywords": []
      }
    ]
  },
  "spotting": {
    "groups": [
      {
        "groupName": "Competitors",
        "spotted": false,
        "score": "0.4439",
        "spottedKeywords": [
          {
            "keyword": "Microsoft",
            "relevance": 1,
            "mentions": [
              {
                "speakerName": "Speaker 1",
                "occurrences": [
                  {
                    "s": 34234,
                    "e": 34234,
                    "exact": "Microsoft"
                  }
                ]
              }
            ]
          }
        ]
      }
```

(continues on next page)

(continued from previous page)


```
    ]
  },
  "prediction": {
    "classifiers": [
      {
        "classifierId": "3e8dee45-ae2a-432f-b5ff-aa2513986f23",
        "classifierName": "SaleCompleted",
        "classifierVersion": "1.0",
        "classifierDisplayName": "Sales Completed",
        "classifierType": "binary",
        "predictedClassLabel": "completed",
        "predictionScore": 0.929,
        "predictedClass": 1
      }
    ],
    "detectors": [
      {
        "detectorId": "99179da8-2ef4-4478-92d0-f296399a90b7",
        "detectorName": "PCI",
        "detectorVersion": "1.0",
        "detectorDisplayName": "Detects credit card data",
        "detectorType": "binary",
        "detections": [
          {
            "detectorClass": 1,
            "detectorClassLabel": "pci",
            "detectedSegments": [
              {
                "speakerName": "Speaker 1",
                "occurrences": [
                  {
                    "s": 34322,
                    "e": 458375
                  }
                ]
              }
            ]
          }
        ]
      }
    ]
  },
  "metrics": [
    {
      "metricGroupName": "groupX",
      "metricValues": [
        {
          "metricName": "xyz",
          "metricValue": 200
        }
      ]
    }
  ],
  "categories": [
    {
      "categoryName": "abc",
      "categoryValue": 0
```


(continues on next page)

(continued from previous page)


```
  }, {
    "categoryName": "def",
    "categoryValue": 1,
    "categoryMatches": [
      {
        "speakerName": "caller",
        "occurrences": [
          {
            "s": 24251,
            "e": 24981,
            "exact": "hello"
          }, {
            "s": 26491,
            "e": 30571,
            "exact": "hi"
          }
        ]
      }, {
        "speakerName": "agent",
        "occurrences": [
          {
            "s": 24251,
            "e": 24981,
            "exact": "greetings"
          }, {
            "s": 26491,
            "e": 30571,
            "exact": "how are you"
          }
        ]
      }
    ]
  }
],
"conversation": {
  "speakerVerbNounPairs": [
    {
      "speakerName": "Agent",
      "verbNounPairs": [
        {
          "s": 6679,
          "e": 7260,
          "verb": "call",
          "noun": "team"
        }, {
          "s": 44279,
          "e": 45099,
          "verb": "have",
          "verbNeg": "not",
          "noun": "question"
        }, {
          "s": 807908,
          "e": 808918,
          "verb": "like",
          "noun": "service",
          "question": true
        }
```


(continues on next page)

(continued from previous page)


```
      ]
    }, {
      "speakerName": "Caller",
      "verbNounPairs": [
        {
          "s": 16250,
          "e": 17340,
          "verb": "need",
          "noun": "help"
        }, {
          "s": 901234,
          "e": 902786,
          "verb": "provide",
          "noun": "justification",
          "nounNeg": "no"
        }, {
          "s": 1002560,
          "e": 1003010,
          "verb": "work",
          "verbNeg": "not",
          "noun": "internet"
        }
      ]
    }
  ]
},
"speakerSentiments": [
  {
    "speakerName": "Caller",
    "sentimentValues": [
      {
        "s": 4558,
        "e": 7064,
        "v": -0.5434
      }, {
        "s": 9373,
        "e": 10345,
        "v": 0.7039
      }
    ]
  }, {
    "speakerName": "Agent",
    "sentimentValues": [
      {
        "s": 7464,
        "e": 9373,
        "v": 0.4328
      }, {
        "s": 12937,
        "e": 14627,
        "v": -0.3294
      }
    ]
  }
],
"transcript": {
  "confidence": 1.0,
```


(continues on next page)

(continued from previous page)


```
  "words": [
    {
      "p": 3,
      "c": 0.845,
      "s": 13466,
      "e": 15648,
      "m": "turn|punc",
      "v": 34,
      "w": "supercalifragilisticexpialidocious",
      "frq": [
        {
          "e": 1.344,
          "f": 234.0
        }, {
          "e": 2.344,
          "f": 340.0
        }
      ]
    }
  ],
  "voiceActivity": [
    {
      "speakerName": "Speaker 1",
      "occurrences": [
        {
          "s": 13000,
          "e": 150547
        }, {
          "s": 163746,
          "e": 258726
        }
      ]
    }
  ],
  "alternateFormats": [
    {
      "format": "srt",
      "contentType": "text/srt",
      "contentEncoding": "base64",
      "charset": "utf-8",
      "data": "A Base64 encoded transcript"
    }
  ]
},
"streams": [
  {
    "status": "HTTP Status of the stream. Are we using this?",
    "streamName": "original",
    "streamLocation": "https://somewhere.voicebase.com/xyzt&amp;expires=12344",
  }
],
"encryption": {
  "publishKeyId": "11e13265-e688-428b-b7bb-708c12a30a41",
  "publicKeyHash": "A SHA256"
}
}
```

5.3.1 Data Description

- `_links` : HAL metadata with a URL for the corresponding media item
  - `self` : section for the media item
    - `href` : URL for the media item
- `media` : the requested data for the media item
  - `mediaId` : the unique VoiceBase id for the media item
  - `status` : the status of processing for the media item
  - `mediaContentType` : the media item content type
  - `length` : the media item length
  - `metadata` : the metadata for the media item, typically for correlation to external systems (present if requested when media is uploaded)
  - `transcript` : the transcript(s) for the media (present if requested when media is uploaded)
  - `knowledge` : the topics and keywords for the media (present if requested when media is uploaded)
  - `predictions` : the predictions results for the media
  - `streams` : links for the results of the media

CHAPTER 6

Categories

VoiceBase enables you to apply speech analytics to a recording and categorize it according to business definitions. Each definition is called a category and it consists of a conditional statement that is applied to the transcription results and any corresponding metadata provided with the recording.

VoiceBase runs the categories when the recording is processed, and returns a boolean result where a 1 equals a match for the category definition and 0 when it does not match.

Categories are especially useful as a discovery tool when used in conjunction with the Analytic Workbench available in your VoiceBase account.

Categories are created using VoiceBase's proprietary query language, VBQL. For a step-by-step guide to syntax and best practices, the VBQL documentation is available for VoiceBase customers, and we are also happy to spend time in training customers on its usage.

6.1 Adding a category

To define or update a category, make a `PUT` request against the `/definition/categories/{categoryName}` resource.

The JSON body of the request contains the category definition. For example:

```
{
  "categoryDescription": "This category looks for participants saying hello",
  "query": "SELECT * FROM media MATCH 'hello'",
  "notes": "Notes about creating or maintaining the category can be added here",
  "tags": [
    "hello",
    "greeting",
    "example"
  ],
  "version": "0.0.1"
}
```

Categories may alternatively be added using the "Publish" option in the Analytic Workbench.

The body of the category definition contains the following fields:

- `categoryName`: The key name of the category. The name may not contain any spaces.
- `categoryDescription`: The optional description of the category. The description is a string that is limited to ? characters including spaces.
- `query`: A condition for evaluating a recording transcription and the accompanying metadata. The condition is written using the VBQL syntax.
- `notes` : Optional information that helps the users and managers of the category further understand the category. The notes is a string that is limited to 4,000 characters including spaces.
- `tags` : An optional array of terms that helps users and managers of the category further understand the category. Each tag is string that is limited to 64 characters with maximum of 32 tags. The tags are not used anywhere else in the VoiceBase platform.
- `version`: An optional string value that helps the users and managers of the category further understand the version of the category. A version is unique to the account and is not used anywhere else in the VoiceBase platform.

6.2 Listing categories

To list defined categories, make a `GET` request against the `/definition/categories` resource.

For example, if the account has `hello` and `goodbye` categories defined, the response would be (fields omitted for clarity):

```
{
  "categories": [
    {
      "categoryName": "hello",
      "categoryDescription": "This category looks for participants saying hello"
    },
    {
      "categoryName": "goodbye",
      "categoryDescription": "This category looks for participants saying goodbye"
    }
  ]
}
```

6.3 Computing all categories

To enable categorization (computing categories), add a `categories` section to your `configuration` when POSTing to /media. The `categories` configuration is an array. To request that all categories be computed, add a single element consisting of a JSON object with the `allCategories` flag set to `true`.

```
{
  "categories": [
    { "allCategories": true }
  ]
}
```

The `configuration` contains the following fields:

- `categories`: The configuration section for categorization
- `allCategories`: A flag that indicates all categories should be computed

6.4 Computing specific categories

To enable categorization (computing categories) for a specific subset of categories, add a `categories` section to your `configuration` when POSTing to /media. The `categories` configuration is an array. For each category to compute, add an element consisting of a JSON object with the `categoryName` key set to the name of the category.

For example:

```
{
  "categories": [
    { "categoryName": "hello" },
    { "categoryName": "goodbye" }
  ]
}
```

The `configuration` contains the following fields:

- `categories`: The configuration section for categorization
- `categoryName`: The name of a specific category to compute

6.5 Results for categories

Results of categorization (computing categories) are included with responses from `GET /v3/media/{mediaId}` API, and in callbacks.

For example:

```
{
  "categories": [
    { "categoryName": "hello", "categoryValue": 1 },
    { "categoryName": "goodbye", "categoryValue": 0 }
  ]
}
```

This section contains the following fields:

- `categories`: the response section for categories
- `categoryName`: the name of the category computed
- `categoryValue`: the result of computing the category (`0` = no match, `1` = match)

Note: Category values are represented as `0` and `1` rather than `false` and `true` to help in aggregation use cases common in reporting.

CHAPTER 7

Closed Captioning

VoiceBase can generate subtitles or closed captions for your video project, by allowing you to retrieve the transcript of your audio or video file using the WebVTT or SubRip Text (SRT) format.

No special configuration is required. All transcripts are always available in four formats: JSON word-by-word, plain text, WebVTT and SRT.

7.1 WebVTT format

WebVTT is a W3C standard which may be used for displaying timed text within the HTML5 element. A WebVTT file will begin with WEBVTT after an optional UTF-8 byte order mark. The timecode format used is hours:minutes:seconds.milliseconds with hours being optional and time units fixed to two zero-padded digits and fractions fixed to three zero-padded digits (00:00:00.000). The fractional separator used is the full-stop.

Example:

```
WEBVTT

1
00:00:01.44 --> 00:00:04.25
Customer: Hi this is C.S.V. Shipping
company Brian speaking how can I help you.

2
00:00:05.61 --> 00:00:08.48
Agent: This is Henry A
We spoke earlier I got a quote from you guys.
```

7.2 GET a WebVTT Transcript

Export `MEDIA_ID` and `TOKEN`

```
export MEDIA_ID='7eb7964b-d324-49cb-b5b5-76a29ea739e1'
export TOKEN='Your Api Token'
```

and provide the `Accept` HTTP header with the value `"text/vtt"` when requesting the transcript.

```
curl https://apis.voicebase.com/v3/media/${MEDIA_ID}/transcript/webvtt \
  --header "Accept: text/vtt" \
  --header "Authorization: Bearer ${TOKEN}"
```

7.3 SRT subtitle format

An SRT file contains formatted lines of plain text in groups separated by a blank line. Subtitles are numbered sequentially, starting at 1. The timecode format used is hours:minutes:seconds,milliseconds with time units fixed to two zero-padded digits and fractions fixed to three zero-padded digits (00:00:00,000). The fractional separator used is the comma.

1. A number indicating which subtitle it is in the sequence.
2. The time that the subtitle should appear on the screen, and then disappear.
3. The subtitle itself.
4. A blank line indicating the start of a new subtitle.

Example:

```
1
00:00:02,76 --> 00:00:05,08
Agent: Well this is Michael
thank you for calling A.B.C.

2
00:00:05,08 --> 00:00:07,03
Cable services. How may I help you today.

3
00:00:08,28 --> 00:00:11,93
Customer: Hi I'm calling because I'm
interested in buying new cable services.

4
00:00:12,64 --> 00:00:16,43
Agent: OK great let's get started.
```

7.4 GET an SRT format

Export `MEDIA_ID` and `TOKEN`

```
export MEDIA_ID='7eb7964b-d324-49cb-b5b5-76a29ea739e1'
export TOKEN='Your Api Token'
```

and provide the `Accept` HTTP header with the value `"text/srt"` when requesting the transcript.

```
curl https://apis.voicebase.com/v3/media/${MEDIA_ID}/transcript/srt \
  --header "Accept: text/srt" \
  --header "Authorization: Bearer ${TOKEN}"
```

7.5 Callbacks

Note that when posting a media file with a configuration including a callback, the results posted to the callback URL always contain the JSON (word-by-word), the plain text, WebVTT and SRT transcripts.

CHAPTER 8

Conversation Metrics

VoiceBase enables you to apply to a recording predefined algorithms that compute metrics useful for tracking the quality of the conversation in the recording. Each metric is applied to the transcription results, uses the word timing and words themselves and the numeric results for each metric are provided with the json output.

8.1 Using Conversation Metrics

Enable conversation metrics for the account by setting the "metricGroupName" for each group in the "metrics" section of the "configuration" when POSTing to /media. The configuration may have one, two, three or all four of the metric groups in the POST.

```
{
  "metrics": [
    { "metricGroupName": "overtalk" },
    { "metricGroupName": "sentiment" },
    { "metricGroupName": "talk-style-tone-and-volume" },
    { "metricGroupName": "talk-time-and-rate" }
  ]
}
```

The `metrics` section of the configuration contains the following fields:

- `metrics`: The metrics section
- `metricGroupName`: The name of the metric group (available groups are: `"overtalk"`, `"sentiment"`, `"talk-style-tone-and-volume"`, and `"talk-time-and-rate"`)

8.2 Results for Metrics

Results of metrics are included with response from the `GET /v3/media/{mediaId}` API, and in callbacks.

For example (data omitted for clarity):

```
{
  "metrics": [
    {
      "metricGroupName": "overtalk",
      "metricValues": [
        {
          "metricName": "overtalk-ratio",
          "metricValue": 0.04
        },
        {
          "metricName": "overtalk-incidents",
          "metricValue": 2
        }
      ]
    },
    {
      "metricGroupName": "sentiment",
      "metricValues": [
        {
          "metricName": "agent-sentiment",
          "metricValue": 0.56
        },
        {
          "metricName": "caller-sentiment",
          "metricValue": 0.67
        }
      ]
    }
  ]
}
```


This section contains the following fields:

- `metrics`: The response section for metrics
- `metricGroupName`: The name of the metric group
- `metricValues`: The array of metrics and their values for the call
- `metricName`: The name for the specific metric
- `metricValue`: The value of the specific metric for the call

8.3 Metric Definitions

The definition of each available metric is included below.

8.3.1 Overtalk Metrics (6 metrics)

- `overtalk-incidents`: The number of times that the agent and caller talked at the same time, expressed as an integer from zero to n.
- `agent-overtalk-incidents`: The number of times that the agent began talking while the caller was already talking, expressed as an integer from zero to n. This metric applies only to Stereo calls, and will return a zero for all other types of calls.

- `caller-overtalk-incidents`: The number of times that the caller began talking while the agent was already talking, expressed as an integer from zero to n. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `overtalk-ratio`: The percentage of the total call duration where the agent and caller talked at the same time, expressed as a number from 0 to 1 with .01 resolution, with 0.1 corresponding to overtalk for 10% of the call. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `agent-overtalk-ratio`: The percentage of the call where the agent and caller talked at the same time, and where the caller began talking first (e.g. the agent talked over the caller) expressed as a number from 0 to 1 with .01 resolution. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `caller-overtalk-ratio`: The percentage of the call where the agent and caller talked at the same time, and where the agent began talking first, expressed as a number from 0 to 1 with .01 resolution. This metric applies only to Stereo calls, and will return a zero for all other types of calls.

8.3.2 Talk Time and Rate Metrics (18 metrics)

- `agent-talk-ratio`: The percentage of non-silence time that the agent was talking, expressed as a number from 0 to 1 with .01 resolution. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `caller-talk-ratio`: The percentage of non-silence time that the caller was talking, expressed as a number from 0 to 1 with .01 resolution. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `silence-ratio`: The percentage of time where neither the caller nor the agent were talking, expressed as a number from 0 to 1 with .01 resolution.
- `silence-incidents`: The number of times that neither the agent nor the caller were talking, where the duration was >4 seconds, expressed as an integer from zero to n.
- `agent-talk-rate`: The average rate of speech for the agent over the entire call, with times when the other part is talking and significant pauses removed, expressed as word per minute (WPM) as an integer from zero to n. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `caller-talk-rate`: The average rate of speech for the agent over the entire call, with times when the other part is talking and significant pauses removed, expressed as word per minute (WPM) as an integer from zero to n. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `agent-to-caller-talk-rate-ratio`: A measure of agent talk rate compared to the caller talk rate, expressed as ratio. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `agent-intra-call-change-in-talk-rate`: A measure of the talk rate for the agent, as measured above, from the first 1/3 of the call compared to the rate from the last 1/3 of the call, expressed as a number from zero to positive n, where 1.0 represents no change on talk rate and .3 represents a 70% decrease in talk rate. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `caller-intra-call-change-in-talk-rate`: A measure of the talk rate for the caller, as measured above, from the first 1/3 of the call compared to the rate from the last 1/3 of the call, expressed as a number from zero to positive n, where 1.0 represents no change in talk rate and .3 represents a 70% decrease in talk rate. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `agent-average-streak`: The average time for all streaks of an agent talking, expressed in seconds. A streak starts with the first word spoken by the caller at the beginning of a call or the next word spoken after a pause of 3 seconds or more and ends with a pause of 3 seconds or more or the end of the call. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `agent-longest-streak`: The total time for the longest streak of an agent talking, expressed in seconds. A streak starts with the first word spoken by the agent at the beginning of a call or the next word spoken after

a pause of 3 seconds or more and ends with a pause of 3 seconds or more or the end of the call. This metric applies only to Stereo calls, and will return a zero for all other types of calls.

- `agent-median-streak`: The streak talking time value where half the agent talking streaks are below this value and half are above this value. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `average-streak`: The average time for all streaks of both an agent and a caller talking, expressed in seconds. A streak starts with the first word spoken by the caller at the beginning of a call or the next word spoken after a pause of 3 seconds or more and ends with a pause of 3 seconds or more or the end of the call.
- `caller-average-streak`: The average time for all streaks of a caller talking, expressed in seconds. A streak starts with the first word spoken by the caller at the beginning of a call or the next word spoken after a pause of 3 seconds or more and ends with a pause of 3 seconds or more or the end of the call. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `caller-longest-streak`: The total time for the longest streak of a caller talking, expressed in seconds. A streak starts with the first word spoken by the caller at the beginning of a call or the next word spoken after a pause of 3 seconds or more and ends with a pause of 3 seconds or more or the end of the call. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `caller-median-streak`: The streak talking time value where half the caller talking streaks are below this value and half are above this value. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `longest-streak`: The total time for the longest streak of an agent or caller talking, expressed in seconds. A streak starts with the first word spoken by the caller at the beginning of a call or the next word spoken after a pause of 3 seconds or more and ends with a pause of 3 seconds or more or the end of the call.
- `median-streak`: The streak talking time value where half of all talking streaks are below this value and half are above this value.

8.3.3 Talk Style Tone and Volume Metrics (8 metrics)

Note: These metrics require the "voice features" feature enabled in the configuration that goes with the processing request sent to the VoiceBase /v3 endpoint.

- `agent-intra-call-change-in-pitch`: A ratio for the intra call change in pitch of the agent where the ratio is calculated by taking the average pitch for the last third of the agent's words divided by the average pitch of the first third of the agent's words. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `agent-intra-call-change-in-relative-voice-volume-energy`: A ratio for the intra call change in volume of the agent where the ratio is calculated by taking the average volume for the last third of the agent's words divided by the average volume of the first third of the agent's words. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `agent-relative-voice-volume-energy`: The measure of how energetically an agent speaks. The metric is the average volume of the agent's words divided by a fixed average volume of a good agent (=4.33). This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `agent-voice-dynamism-std-dev-score`: The measure of the standard deviation for all the agent's words in a transcription. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `caller-intra-call-change-in-pitch`: A ratio for the intra call change in pitch of the caller where the ratio is calculated by taking the average pitch for the last third of the agent's words divided by the average pitch of the first third of the caller's words. This metric applies only to Stereo calls, and will return a zero for all other types of calls.

- `caller-intra-call-change-in-relative-voice-volume-energy`: A ratio for the intra call change in volume of the caller where the ratio is calculated by taking the average volume of the last third of the caller's words divided by the average volume of the first third of the caller's words. This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `caller-relative-voice-volume-energy`: The measure of how energetically an agent speaks. The metric is the average volume of the agent's words divided by a fixed average volume of a good caller (=7.57). This metric applies only to Stereo calls, and will return a zero for all other types of calls.
- `caller-voice-dynamism-std-dev-score`: The measure of the standard deviation for all the caller's words in a transcription. This metric applies only to Stereo calls, and will return a zero for all other types of calls.

8.3.4 Sentiment Metrics (12 metrics)

Note: Sentiment scores are calculated with a combination of machine classification and NLP for sentences in the transcript.

- `call-sentiment`: A score that compares all the sentence sentiment values for the transcript. The range is between -1 and 1 with -1 for very negative, 0 for neutral and 1 for very positive.
- `start-sentiment`: A score that compares all of the sentence sentiment values for the first third of the total sentences for the transcript. The range is between -1 and 1 with -1 for very negative, 0 for neutral and 1 for very positive.
- `end-sentiment`: A score that compares all of the sentence sentiment values for the last third of the total sentences for the transcript. The range is between -1 and 1 with -1 for very negative, 0 for neutral and 1 for very positive.
- `call-change-in-sentiment`: A score that subtracts the End-sentiment from the Start-sentiment. The range is -2 to 2 with a -2 for a call that started very positive but ended very negative, 0 for a call that did not change in sentiment and 2 for a call that started very negative but ended very positive.
- `agent-sentiment`: A score that compares all the sentence sentiment values of the agent for the transcript. The range is between -1 and 1 with -1 for very negative, 0 for neutral and 1 for very positive.
- `agent-start-sentiment`: A score that compares all of the agent sentence sentiment values for the first third of the total agent sentences for the transcript. The range is between -1 and 1 with -1 for very negative, 0 for neutral and 1 for very positive.
- `agent-end-sentiment`: A score that compares all of the agent sentence sentiment values for the last third of the total agent sentences for the transcript. The range is between -1 and 1 with -1 for very negative, 0 for neutral and 1 for very positive.
- `agent-intra-call-change-in-sentiment`: A score that subtracts the Agent-end-sentiment from the Agent-start-sentiment. The range is -2 to 2 with a -2 for a call that started very positive but ended very negative, 0 for a call that did not change in sentiment and 2 for a call that started very negative but ended very positive.
- `caller-sentiment`: A score that compares all the sentence sentiment values of the caller for the transcript. The range is between -1 and 1 with -1 for very negative, 0 for neutral and 1 for very positive.
- `caller-start-sentiment`: A score that compares all of the caller sentence sentiment values for the first third of the total caller sentences for the transcript. The range is between -1 and 1 with -1 for very negative, 0 for neutral and 1 for very positive.
- `caller-end-sentiment`: A score that compares all of the caller sentence sentiment values for the last third of the total caller sentences for the transcript. The range is between -1 and 1 with -1 for very negative, 0 for neutral and 1 for very positive.

- `caller-intra-call-change-in-sentiment`: A score that subtracts the Caller-end-sentiment from the Caller-start-sentiment. The range is -2 to 2 with a -2 for a call that started very positive but ended very negative, 0 for a call that did not change in sentiment and 2 for a call that started very negative but ended very positive.

8.4 Prerequisites for Use

Conversation Metrics are designed to be used for stereo conversations between a caller and an agent. There are some hard requirements to generate meaningful results, and some strongly recommended features of call processing.

Requirements for use are:

- Calls MUST be processed in stereo for agent and caller specific metrics (otherwise, these metrics always return 0)
- The `agent` SHOULD have a speaker name that is one of: `agent`, `service`, `representative`, `operator`, `salesperson`, `callcenter`, `contactcenter`
- The `caller` SHOULD have a speaker name that is one of: `caller`, `client`, `customer`, `prospect`
- If the speaker names are not specified as above, the first speaker is assumed to be the agent, and the second speaker is assumed to be the caller
- For the sentiment metrics, the language MUST be one of: `en-US`, `en-AU`, `en-UK`
- Voice Features SHOULD be enabled, and is required for some talk style, tone, and volume metrics
- Advanced Punctuation SHOULD be enabled

CHAPTER 9

Custom Vocabulary

9.1 Summary

VoiceBase allows you to customize the speech engine to correctly recognize words and phrases that you would not find in a standard dictionary. Using this, you can have the flexibility of a highly tuned, and even more accurate speech engine for specific verticals and vernacular. You may add custom terms to the speech engine vocabulary, create 'sounds like' spellings which more accurately reflect how an uncommon word may sound, and add a weighting factor which will fine-tune preferential recognition of custom terms. We have recently augmented the power of custom vocabulary by allowing you to specify a 'script' instead of a set of terms. Think of scripts as contextually-rich terms. Scripts, like terms, are made up of words, but the order and their context is relevant. Our speech engine becomes more accurate, as the scripts essentially train the speech engine on what to expect.

9.2 Use Case

The Custom Vocabulary feature is most useful for:

- Jargon
- Proper Nouns (names, products, companies, street and city names, etc.)
- Acronyms
- Non-dictionary words
- Hyphenated words
- Multi-word phrases

Let's consider a simple message:

> Hi this is Bryon from VoiceBase.

Let's look at how to use Custom Vocabulary terms to help ensure that Bryon (an uncommon spelling) and VoiceBase are correctly recognized. The name Bryon can be input from a CRM system, along with street names, product names, or company names.

Using the ad-hoc method of Custom Vocabulary we can add these terms as part of the request configuration file:

```
{
  "vocabularies": [
    {
      "terms" : [
        {
          "term": "Bryon"
        },
        {
          "term": "VoiceBase"
        }
      ]
    }
  ]
}
```

When processed with the Custom Vocabulary configuration, the returned transcript would read:

> "Hi, this is Bryon from VoiceBase."

9.3 Sounds Like and Weighting

Let's suppose a recording starts with an agent saying "Hi this is C.S.V. Shipping Comnpany" but the speech engine does not recognize "C.S.V." and mistranscribes the opening as "Hi this is just the shipping company." The best practice to correct this would be to employ the mistranscription as a "soundsLike" term, so since the error appears as "just the", this would be configured as:

```
{
  "soundsLike": [
    "just the"
  ],
  "term": "C.S.V."
}
```

Up to 100 Sounds Like terms separated by commas may be added for each custom term.

9.3.1 Weighting

Weighting may optionally be used to increase the likelihood that terms are recognized by the speech engine. Terms are weighted with an integer value from 0 to 5, with 0 being default. Each increase in the weighting value will increase the likelihood that the speech engine will select the Custom Vocabulary term, but also will increase the likelihood of false positives. With weighting added to the custom term "gnocchi", the custom vocabulary string becomes:

```
{
  "soundsLike": [
    "nyohki", "nokey"
  ],
  "weight": 2,
  "term": "gnocchi"
}
```

Weighting may be used with or without Sounds Like. Please note: Weighting is not supported with the Europa Speech Engine.

9.3.2 Phrases

Phrases may also be added using Custom Vocabulary as way to ensure that common or scripted phrases are properly recognized. We may wish to add common phrases for our recordings: "Thank you for calling VoiceBase" and "what's in your calls?". By adding 'Bryon from VoiceBase' and 'Bryon M.', we can ensure that Bryon's name is properly recognized, and allows recognition of the more common Brian in the same recording when the rest of the custom phrase is not present. Custom Phrases result in the speech engine evaluating the phrase as a unit and generally are tolerant of pronunciation variances in any one term. *Note:* Custom Phrases *should not* contain Sounds Like terms, and multi-term Custom Vocabulary entries will have Sounds Like ignored. An individual term may be added to the Custom Vocabulary list with its own Sounds Like if required.

9.4 Acronyms

Acronyms may be added to the Custom Vocabulary for processing your recording. The VoiceBase speech engine will recognize acronyms when periods are placed between the letters, such as: 'Q.E.D.' or 'C.E.O.'. **Remember:** For the purpose of recognizing acronyms, the VoiceBase speech engine follows The New York Times' style guide, which recommends following each segment with a period when when letters are pronounced individually, as in 'C.E.O.', but not when pronounced as a word, such as 'NATO' or 'NAFTA'. It is therefore important to remember that acronyms which are spelled **must** have a period following each letter. If you would like your transcription to read differently, you may use Sounds Like to accomplish this by adding the spelled acronym (including punctuation) as the Sounds Like term, as in:

```
{
  "soundsLike": [
    "C.E.O."
  ],
  "term": "CEO"
}
```

There are two ways to upload custom vocabulary terms to VoiceBase:

9.5 Ad-Hoc Scenario

If you have a recording to transcribe and want to add ad hoc custom terms specifically for that file, upload the file with the following configuration:

```
{
  "vocabularies": [
    {
      "terms" : [
        {
          "soundsLike": [
            "A.F.F.O."
          ],
          "term": "AFFO",
          "weight": 2
        },
        {
          "soundsLike": [
            "Aypack"
          ],
          "term": "APAC",
```

(continues on next page)

(continued from previous page)

```
            "weight": 2
          },
          {
            "term": "CapEx"
          }
        ]
      }
  ]
}
```

9.6 Pre-Defined List

You can add a re-usable custom vocabulary list to your VoiceBase account with a PUT request to /v3/definition/vocabularies/($VOCAB-LIST-NAME) with Content-Type application/json and the following body:

```
{
  "vocabularyName": "earningsCalls",
  "terms": [
    {
      "soundsLike": [
        "A.F.F.O."
      ],
      "term": "AFFO",
      "weight": 2
    },
    {
      "soundsLike": [
        "Aypack"
      ],
      "term": "APAC",
      "weight": 2
    },
    {
      "term": "CapEx"
    }
  ]
}
```

From this example, the Custom Vocabulary list is named "earningsCalls", and can be now be used in configurations attached with all media uploads. To attach a pre-defined custom vocabulary list to media files, use the configuration below: (**Note** This feature can also be combined with ad-hoc terms.)

```
{
  "vocabularies": [
    {
      "vocabularyName" :  "earningsCalls"
    }
  ]
}
```

9.7 Limitations

- Currently, Custom Vocabulary has a limit of 1000 terms per file processed. Greater than 1000 terms may result in noticeable degradation in turnaround time and accuracy.
- Custom Vocabulary terms and phrases are limited to 64 characters.
- Up to 100 Sounds Like spellings may be added to each Custom Vocabulary term. Only single words may be used with Sounds Like, though a phrase may be used to describe how a single word sounds. Acronyms which are spelled must contain periods between each letter (A.F.F.O.), or use Sounds Like. Acronyms which are said as words must be written as words (NATO).
- Sounds Like may not contain multi-term phrases.

Note: VoiceBase allows you to create vocabularies specific for each recording. In this way, you can submit names of people, streets, cities, products, and industry or company specific terms from a CRM system or other data source which are useful in this call, but will not affect others.

9.8 Example cURL Request

```
curl https://apis.voicebase.com/v3/definition/vocabularies/earningsCalls \
  --request PUT \
  --header "Content-Type: application/json" \
  --header "Authorization: Bearer ${TOKEN}" \
  --data '{
      "vocabularyName": "earningsCalls",
      "terms": [
        {
          "soundsLike": [
            "A.F.F.O."
          ],
          "term": "AFFO",
          "weight": 2
        },
        {
          "soundsLike": [
            "Aypack"
          ],
          "term": "APAC",
          "weight": 2
        },
        {
          "term": "CapEx"
        }
      ]
    }'
```

9.9 Example successful response to PUT request

```
{

  "_links": {
    "self": {
```

(continues on next page)

(continued from previous page)

```
      "href": "/v3/definition/vocabularies/earningsCalls"
    }
  },
  "vocabularyName": "earningsCalls",
  "terms": [
    {
      "term": "AFFO",
      "soundsLike": [
        "A.F.F.O."
      ],
      "weight": 2
    },
    {
      "term": "APAC",
      "soundsLike": [
        "Aypack"
      ],
      "weight": 2
    },
    {
      "term": "CapEx"
    }
  ]
}
```

CHAPTER 10

Entity Extraction

10.1 Overview

VoiceBase Entity Extraction locates named entities mentioned in an audio file, then classifies them into pre-defined categories such as person names, organizations, locations, time expressions, quantities, and monetary values.

Available entities include “location”, “person”, “organization”, “money”, “number”, “phone”, “time”, and “date”.

10.2 Configuration

To add entities to your transcript, include the following configuration when making a POST request to the /media resource:

```
{ "entity" : { "enableEntityExtraction" : true } }
```

The returned transcript will contain entities in the following format:

```
{
        "entities": [{
                        "text": "four one five eight zero nine nine",
                        "type": "phone",
                        "formattedText": "4158099",
                        "s": 32582,
                        "e": 36183
                },
                {
                        "text": "Aisha",
                        "type": "person",
                        "s": 2160,
                        "e": 2490
                },
                {
```

(continues on next page)

(continued from previous page)


```
                        "text": "San Francisco",
                        "type": "location",
                        "s": 2760,
                        "e": 3030
                },
                {
                        "text": "VoiceBase",
                        "type": "organization",
                        "s": 43075,
                        "e": 43705
                },
                {
                        "type": "money",
                        "formattedText": "$240.00",
                        "text": "two hundred forty",
                        "s": 55549,
                        "e": 57779,
                        "money": {
                                "whole": 240,
                                "centi": 0
                        }
                },
                {
                        "type": "date",
                        "text": "July",
                        "s": 165739,
                        "e": 166299
                },
                {
                        "type": "date",
                        "text": "second",
                        "s": 166309,
                        "e": 166799
                },
                {
                        "type": "time",
                        "text": "one",
                        "s": 168219,
                        "e": 168509
                },
                {
                        "type": "time",
                        "text": "o'clock",
                        "s": 168519,
                        "e": 168989
                }

        ]
}
```

CHAPTER 11

Formatting and Punctuation

Formatting for US phone numbers and digits is enabled by default, though you may optionally disable it.

Additional punctuation may be added to your transcript by adding the correct tag to your configuration.

11.1 Advanced Punctuation

By default VoiceBase transcripts provide minimal formatting, and you may optionally include the following advanced formatting feature: Advanced Punctuation may be added to your transcripts by including `"advancedPunctuation"` in the `features` array of your config file.

Please note that Advanced Punctuation is not available in French, German, Italian, or Portuguese at this time.

```
{
  "speechModel" : {
        "language": "en-UK",
        "features" : [  "advancedPunctuation"  ]
  }
}
```

11.2 Advanced Punctuation Results

Results of advanced punctuation are included with responses from `GET /v3/media/{mediaId}` API, and in callbacks. The punctuation uses the same format, appears in the `transcript`'s `words` array with a value of `punc` in the `m` (metadata) field. Advanced Punctuation augments and improves the usage of `.`, `,`, and `?`.

For example:

```
{
  "mediaId": "bc14632d-e81b-4673-992d-5c5fb6573fb8",
  "status": "finished",
```

(continues on next page)

(continued from previous page)


```
"dateCreated": "2017-06-22T19:18:49Z",
"dateFinished": "2017-06-22T19:19:27Z",
"mediaContentType": "audio/x-wav",
"length": 10031,
"transcript": {
   "words":[
      {
        "p":0,
        "s":700,
        "c":0.1,
        "e":3000,
        "w":"agent",
        "m":"turn"
      },
      {
        "p":1,
        "s":700,
        "c":0.537,
        "e":870,
        "w":"Hello"
      },
      {
        "p":2,
        "s":900,
        "c":0,
        "e":900,
        "w":",",
        "m":"punc"
      },
      {
         "p":3,
         "s":950,
         "c":0.1,
         "e":990,
         "w":"How"
      },
      {
         "p":4,
         "s":1020,
         "c":0.975,
         "e":1060,
         "w":"are"
      },
      {
        "p":5,
        "s":1070,
        "c":0.975,
        "e":1120,
        "w":"you"
      },
      {
        "p":6,
        "s":1300,
        "c":0,
        "e":1300,
        "w":"?",
        "m":"punc"
```


(continues on next page)

(continued from previous page)

```
        }
      ]
  }
}
```

11.3 Number Formatting

The Number Formatting feature will transcribe numbers in digit form, that is, "1" rather than "one". Number formatting is enabled by default.

To explicitly disable number-formatting, include the following snippet in your configuration at upload:

```
{
   "transcript": {
      "formatting":{
         "enableNumberFormatting": false
      }
   }
}
```

Please note: our speech engine "Europa" does not allow Number Formatting to be disabled.

Number formatting allows options for transcription preference around phone numbers, currency, addresses, and more. The current version transcribes number words to symbols and US phone number formatting.

For example, with number formatting disabled, a transcript might read:

> "**Agent:** The total of your bill comes to one hundred thirty eight dollars and sixty five cents"

When number formatting is enabled it will read:

> "**Agent:** The total of your bill comes to $138.65"

Additionally, VoiceBase can detect phone numbers and format them into a US phone number format within the transcript.

With number formatting disabled, a plain-text transcript will look like:

> "Hi this is Brian from VoiceBase please give me a call back at six five zero eight nine seven five one seven zero thank you."

A plain-text transcript with number formatting enabled:

> "Hi this is Brian from VoiceBase please give me a call back at 650-897-5170 thank you."

And an excerpt of the number-formatted JSON response:

For simplicity the following keys & respective values have been omitted - start time ('s'), end time ('e'), confidence ('c').

```
          {
            "p": 12,
            "w": "call"
          },
          {
            "p": 13,
            "w": "back"
          },
```

(continues on next page)

(continued from previous page)

```
        {
          "p": 14,
          "w": "at"
        },
        {
          "p": 15,
          "w": "650-897-5170"
        },
```

As you can see in the JSON response, the phone number will be returned in one word value with the time-stamp beginning at the first digit said and ending at the last digit.

CHAPTER 12

Keywords and Topics

VoiceBase can discover the keywords, key phrases, and topics in your recording using a processing known as semantic indexing. The results are known as "semantic knowledge discovery" and it includes keywords and topics.

VoiceBase also supports keyword and key phrase spotting. You can define groups of keywords or key phrases, which are flagged when they are spotted in the recording. For more details, see Keyword Spotting.

12.1 Semantic Keywords and Topics

Semantic keywords and topics are discovered automatically (for languages where the feature is supported) and returned with the analytics under the section "knowledge" For example, the `knowledge.keywords` section may contain an entry like the following:

```
{
  "keyword": "subscription",
  "relevance": 0.17,
  "mentions" : [
     {
       "speakerName" : "caller",
       "occurrences" : [
          { "s": 34480, "exact": "subscription" },
          { "s": 57340, "exact": "membership" }
       ]
     },
     {
       "speakerName" : "agent",
       "occurrences" : [
          { "s": 40010, "exact": "subscription" }
       ]
     }

  ]
}
```

In this example, the keyword `subscription` was spoken by the caller around 34 and 57 seconds into the recording, and by the agent around 40 seconds into the recording.

Topics are reported by grouping a set of keywords that relate to each other. For example, the section `knowledge.topics` may contain an entry like this:


```
{
  "topicName": "Solar energy",
  "subTopics": [ ],
  "relevance": 0.004,
  "keywords": [
    {
      "keyword": "solar power",
      "relevance": 0.17,
      "mentions" : [
          {
            "speakerName" : "caller",
            "occurrences" : [
               { "s": 34480, "exact": "solar power" },
               { "s": 57340, "exact": "solar energy" }
            ]
          },
          {
            "speakerName" : "agent",
            "occurrences" : [
               { "s": 40010, "exact": "solar power" }
            ]
          }

      ]
    },
    {
      "keyword": "solar water heating",
      "relevance": 0.17,
      "mentions" : [
          {
            "speakerName" : "caller",
            "occurrences" : [
               { "s": 134480, "exact": "solar water heater" },
               { "s": 157340, "exact": "solar thermal collector" }
            ]
          }
      ]
    }
  ]
}
```


12.1.1 Relevance

Semantic keywords and topics are scored for relevance. The Keyword `relevance` score ranges from 0.0 to 1.0, with higher scores indicating higher relevance. The Topic `relevance` score is an average of the scores of its keywords, and may be any real number.

Keyword relevance is computed by matching detected keywords and key phrases to an extensive taxonomy of potential keywords (VoiceBase's semantic index). The relevance algorithm generally produces higher scores for more frequent, closely matching, and distinctive keywords, while producing lower scores for very common keywords found in the recording.

12.2 Enabling Semantic Keywords and Topics

Enable semantic keywords and topic extraction by adding keywords and topics to your media POST configuration.

The configuration attachment should contain the key:

- `knowledge` : Settings for Knowledge Discovery
  - `enableDiscovery` : Switch for enabling/disabling knowledge discovery. The default is false.
  - `enableExternalDataSources` : Switch for allowing the search on sources external to VoiceBase. Users concerned about data privacy or PCI requirements can turn this off. Default is true.

For example:

```
{
  "knowledge": {
    "enableDiscovery": true,
    "enableExternalDataSources" : true
  }
}
```

12.3 Examples

12.3.1 Example: Enabling semantic knowledge discovery

The following is an example of posting a media document with semantic `keywords` and `topics` extraction enabled.

```
curl https://apis.voicebase.com/v3/media \
  --header "Authorization: Bearer $TOKEN" \
  --form media=@recording.mp3 \
  --form configuration='{
    "knowledge": {
      "enableDiscovery": true,
      "enableExternalDataSources" : false
     }
  }'
```

CHAPTER 13

Keyword and Phrase Spotting

VoiceBase supports keyword and phrase spotting within your transcript. You can define groups of keywords (or key phrases), which are flagged when they are spotted in the recording.

VoiceBase can also discover the keywords, key phrases, and topics in your recording using a processing known as semantic indexing. For more details, view Keywords and Topics.

13.1 Managing Keyword Spotting Groups

VoiceBase allows you to specify pre-defined groups of keywords (or key phrases), which can be used to flag recordings of interest using keyword spotting.

To define new keyword group, or update an existing keyword group, simply PUT the group under '/definition/spotting/groups'. The body of the PUT request is a JSON object (`Content-Type: application/json`) that contains two keys:

- `groupName` : the name of the keyword group
- `keywords` : an array of the included keywords

For example, to create group `big-data` that includes the keywords `data science`, `big data`, and `data mining`, make the following PUT request using curl, or an equivalent request using a tool of your choice:

```
curl https://apis.voicebase.com/v3/definition/spotting/groups/data-science \
  --request PUT \
  --header "Content-Type: application/json" \
  --data '{ "groupName" : "data-science", "keywords" : [ "data science", "machine␣
↪learning", "data mining", "classification" ] }' \
  --header "Authorization: Bearer ${TOKEN}"
```

13.2 Displaying Keyword Spotting Groups

To display keyword groups that you have created make the following GET request using curl, or an equivalent request using a tool of your choice:

```
curl https://apis.voicebase.com/v3/definition/spotting/groups/data-science \
  --request GET \
  --header "Authorization: Bearer ${TOKEN}"
```

The response will look like the following:

```
{
        "groupName": "data-science",
        "keywords": [ "data science", "machine learning", "data mining",
↪"classification" ]
}
```

Revision is the uniqueID that VoiceBase uses applies to each keyword group that can be used to monitor the group.

13.3 Enabling Keyword Spotting

To upload media with keyword spotting enabled, include a JSON configuration attachment with your media POST. The configuration attachment should contain the key:

```
- `spotting` : object for keyword-spotting configuration
    - `groups` : array of keyword-spotting groups
```

For example:

```
{
    "spotting": {
      "groups": [ { "groupName": "data-science" } ]
    }
}
```

When keyword spotting is enabled, this adds extra analytics to the response. For example:

```
{
  "spotting": {
    "groups": [
          {
            "groupName" : "data-science",
            "score" : 0,
            "spotted" : true,
            "spottedKeywords": [
              {
                "keyword" : "data science",
                "mentions" : [
                  {
                    "speakerName" : "Alice",
                    "occurrences" : [
                      { "s" : 4620 , "exact": "data science" }
                    ]
                  }
                ]
```

(continues on next page)

(continued from previous page)

```
            },
            {
              "keyword" : "data science",
              "mentions" : [
                {
                  "speakerName" : "Bob",
                  "occurrences" : [
                    { "s" : 5730 , "exact": "machine learning" }
                  ]
                }
              ]
            }
          ]
        }
      ]
  }
}
```

13.4 Examples

13.4.1 Example: Defining and enabling a keyword spotting group

Define a `keyword group` by making a PUT request to the `/definition/spotting/groups/data-science` resource.

```
curl https://apis.voicebase.com/v3/definition/spotting/groups/data-science \
  --request PUT \
  --header "Content-Type: application/json" \
  --data '{ "groupName" : "data-science", "keywords" : [  "data science", "machine␣
↪learning", "data mining", "classification" ] }' \
  --header "Authorization: Bearer ${TOKEN}"
```

Upload media from a local file called recording.mp3 and spot keywords using the `data-science` group, make the following POST request to `/media`

```
curl https://apis.voicebase.com/v3/media \
  --header "Authorization: Bearer ${TOKEN}" \
  --form media=@recording.mp3 \
  --form configuration='{ "spotting": { "groups": [ { "groupName" : "data-science"  }␣
↪] } }'
```

CHAPTER 14

Languages

VoiceBase supports the following languages and dialects.

Feature Support by Language:

Language	Code	Transcription	Callbacks	NumFormatting	Keywords&Topics	Predictions	PCI	VoiceFeatures	ConversationMet	SentimentByTurn	VerbNoun
English US	en-US										
English UK	en-UK										
English AU	en-AU										
French	fr-FR										
German	de-DE										
Italian	it-IT										
Portuguese Brazil	pt-BR										
Spanish LatinAm	es-LA										
Spanish Spain	es-ES										
Spanish US	es-US										
Spanish Mexico	es-MX										

Note: en-UK and en-AU use the en-US functions for Keywords, Topics, Number Formatting and PCI.

14.1 Configuring Language Support

Use the language configuration parameter to set the language. Omitting the parameter defaults the language to U.S. English (`en-US`).

For example, to transcribe a recording in Australian English:

```
{
  "speechModel" : {
    "language" : "en-AU"
  }
}
```

- `language` : The language code. Refer to the table above.

14.2 Disabling Semantic Keywords and Topics

Semantic keywords and topics are not currently supported with German, Italian, or Portuguese, please disable the feature in your configuration for these languages.

```
{
  "knowledge": {
    "enableDiscovery" : false
}
```

14.3 Examples

** Note: Export your api `TOKEN` prior to running any of the following examples.

```
export TOKEN='Your Api Token'
```

14.3.1 U.S. English for Voicemail

```
curl https://apis.voicebase.com/v3/media \
    --form media=@recording.mp3 \
    --form configuration='{
      "speechModel" : {
        "language" : "en-US"
        "extensions" : [ "voicemail" ]
      }
    }' \
    --header "Authorization: Bearer ${TOKEN}"
```

14.3.2 U.K. English

```
curl https://apis.voicebase.com/v3/media \
    --form media=@recording.mp3 \
    --form configuration='{
     "speechModel" : {
        "language" : "en-UK"
      }
    }' \
    --header "Authorization: Bearer ${TOKEN}"
```

14.4 Europa speech engine

You may want to try our premium speech engine, Europa for variants of English such as Singapore, Indian, UK, or Australian English. Europa also supports European Spanish and other variants of Spanish.

14.5 More Language Options

VoiceBase has the capability to add languages per customer requests. Please talk to sales if you have a use case requiring any of the following languages:

Arabic * Bulgarian * Catalan * Croatian * Czech * Danish * Dutch * Finnish * Greek * Hindi * Hungarian * Japanese * Korean * Latvian * Lithuanian * Malay * Mandarin * Norwegian * Polish * Romanian * Russian * Slovak * Slovenian * Swedish * Turkish.

CHAPTER 15

Metadata Guide

Transcriptions may be organized by providing additional information about a recording submitted by a POST request to /v3/media.

Media can be submitted with additional information that helps you organize and search your transcriptions based on it. The additional information is supplied by providing a form attribute named "metadata" with a JSON document.

Here is a sample metadata:

```
{
  "title" : "Call with John Smith from Acme Inc",
  "description" : "Contacting customer for offering additional services",
  "externalId" : "CRM-9473-2393470-3",
  "extended" : {
        "call" : {
          "date" : "2017-05-01",
          "phone" : "+1-307-388-2123"
        },
        "upselling" : true
  }
}
```

Metadata is composed of the following items, all of them are optional:

- `title` A title for your recording.
- `description` A description of the recording.
- `externalId` A tag value. It may refer to an identifier on another system. Voicebase does not enforce uniqueness on it, you may have several recordings submitted with the same value for this attribute.
- `extended` Free JSON document. This must be a JSON object, not a primitive nor an array.

15.1 Associating a request with Metadata

In addition to the configuration, specify the "metadata" form attribute in your request.

```
curl https://apis.voicebase.com/v3/media \
  --header "Authorization: Bearer ${TOKEN}"  \
  --form media=@recording.mp3  \
  --form configuration='{ }' \
  --form metadata='{
        "title" : "Gathering Leaves by Robert Frost",
        "externalId" : "A12345"
    }'
```

Metadata will be returned as part of your calls to GET /v3/media/{mediaId} and GET /v3/media.

Metadata will be included in callbacks of type "analytics" that do not specify the "include" attribute and those that explicitly request the "metadata" section through the "include" attribute.

Later on, you can search for media recordings based on the values on the submitted metadata. For example, the following request search for recordings tagged with "A12345" as the "externalId":

```
curl https://apis.voicebase.com/v3/media?externalId=A12345 \
  --header "Authorization: Bearer ${TOKEN}"
```

You may also use the "query" attribute

```
curl https://apis.voicebase.com/v3/media?query=Frost  \
  --header "Authorization: Bearer ${TOKEN}"
```

The search above would match recordings where the word "Frost" occurred on the transcript, title or description. If you want to restrict the matches to only the title, then you must submit the query as

```
 title:Frost
```

which encoded would look like:

```
curl https://apis.voicebase.com/v3/media?query=title%3AFrost  \
  --header "Authorization: Bearer ${TOKEN}"
```

15.2 Pass through Metadata

You may like to attach some metadata to a VoiceBase job and have VoiceBase remember that metadata throughout the life of the job to eventually pass back to you in the results. The metadata can be passed back in a callback or through polling.

Using the 'extended' section of the metadata will allow you to attach any arbitrary JSON that you like. For example, you may like to have metadata from your phone system attached as you pass a new call recording through VoiceBase. That metadata would then be delivered to a destination with transcription and analytic results.

You could achieve that like this,

```
curl https://apis.voicebase.com/v3/media \
  --header "Authorization: Bearer ${TOKEN}"  \
  --form media=@recording.mp3  \
  --form configuration='{ }' \
  --form metadata='{
  "extended" : {
       "agent" : "${AGENT}",
           "date" : "${DATE}",
```

(continues on next page)

(continued from previous page)

```
            "time" : "${TIME}",
            "ani" : "${ANI}",
            "dispcode" : "${DISPCODE}",
            "campaign" : "${CAMPAIGN}"
    }
}'
```

This would add an additional section to your JSON results (both polling and callback) that looks like this:

```
{
    "metadata": {
                "extended": {
                        "agent" : "Your-Agent",
                        "date" : "Your-Date",
                        "time" : "Your-Time",
                        "ani" : "Your-ANI",
                        "dispcode" : "Your-Dispcode",
                        "campaignId" : "Your-Campaign"
                    }
          }
}
```

15.3 ExternalId

You are able to add your own ID to media you sent into VoiceBase. That ID will remain with the job as long as it is in VoiceBase. This most commonly acts as an identifier on another system.

In addition to having the reference permanently attached to the job, some VoiceBase API calls allow filtering by externalId.

An externalId can be added as a value to the key externalId in the metadata JSON

```
curl https://apis.voicebase.com/v3/media \
  --header "Authorization: Bearer ${TOKEN}"  \
  --form media=@recording.mp3  \
  --form configuration='{ }' \
  --form metadata='{
        "externalId" : "A12345"
    }'
```

You can GET jobs by externalId,

```
curl https://apis.voicebase.com/v3/media?externalId=A12345 \
  --header "Authorization: Bearer ${TOKEN}"
    }'
```

Your externalId can be unique, but VoiceBase does not force uniqueness on externalId. If you have multiple jobs with the same externalId, then multiple jobs will be returned when you make the GET above.

15.4 Metadata Filter

You are able to index your own metadata which will allow you to filter GET results based on metadata.

For example, you might like to collect your media, but only where your campaign ID is 1234.

First, you must pre-emptively index campaignId,

Use the PUT /definition/media/search api endpoint to do so.

```
curl https://apis.voicebase.com/v3/definition/media/search \
  --header "Authorization: Bearer ${TOKEN}" \
  --header "Content-Type: application/json" \
  --request PUT \
  --data '{ "fields" : [ "extended.campaignId" ] }'
```

You can then do a metadata-scoped GET

```
curl --header "Authorization: Bearer ${TOKEN}" \
https://apis.voicebase.com/v3/media?extendedFilter=campaignId%3A1234
```

Keep in mind that only media processed AFTER your metadata was indexed will be returned.

For all filtering options, pagination, etc see our Search page.

CHAPTER 16

PCI, SSN, PII Detection

VoiceBase allows you to detect and/or redact sensitive data in your recordings, transcripts, and analytics.

PCI and related detectors are based on machine learned models of real calls where both a caller and an agent are audible. This gives higher reliability and adaptability to real world situations than deterministic models, but also means that for accurate results the audio being processed for PCI, SSN, or PII detection must reflect a real transaction, not a test audio. An actual phone order may reflect some amount of conversation, followed by a product and quantity, the agent giving a total, asking for the card type and number, expiration date and possibly CVV code, and this is the type of data the PCI and related detectors have been trained on.

In contrast, the Number Detector is rule-based and will detect any portion of the conversation containing numbers.

The API offers the following three detectors for sensitive data:

- Payment Card Information (PCI) Detector
  - Detects PCI sensitive numbers, including:
    - Credit Card, Debit Card, and Payment Card numbers
    - Card expiration dates
    - CVV validation codes
- Social Security Number (SSN) Detector
  - Detects Social security numbers
- Number Detector
  - Detects numbers, to be used for Personally Identifiable Information (PII) numbers that do not fall into above categories

VoiceBase offers three detectionLevel parameters within the PCI model for detection or redaction: “entireRegion”, “probableNumbers”, and “numbersOnly”. The “entireRegion” parameter is the default if no parameters are indicated in the configuration.

The “entireRegion” parameter detects sensitive portions of the conversation and typically will mark some buffer before and after, so some portion of the pre and post PCI interval may also show as detected or redacted.

The "probableNumbers" parameter detects the segments of the conversation containing digits within the PCI portion of the conversation, and results contain minimal buffering. While this approach is more precise than the default, it relies on recognition of the speech as numbers or words that sound sufficiently like numbers. For example, an expiration date of: 08/2017 will be redacted, but *August* 2017 will result in only the year '2017' marked as PCI.

The "numbersOnly" parameter is similar to "probableNumbers" but there is no buffering, resulting in more of the non-PCI portions of a conversation remaining in the transcript.

16.1 Support for Spanish

When uploading audio files in Spanish, the detectorLevel parameters are as above, but the model names are "PCI-Spanish" and "SSN-Spanish". "Number" is the model name for "Number Detector" for both English and Spanish files.

16.2 Custom Models

VoiceBase offers custom PCI or SSN models for customers who would like specialized models trained on their own data, resulting in even greater accuracy than our standard model.

16.3 Detected Regions

When detection for sensitive data is enabled, the API returns detected regions as part of analytics for the recording. For example, in a recording with two regions detected as PCI and one region detected as SSN, the analytics would contain:

```
{
  "prediction": {
    "detectors": [
      {
        "detectorId": "abcdefg-1f10-11f2-a085-ec48ab4fbb59",
        "detections": [
          {
            "detectorClass": 1,
            "detectorClassLabel": "PCI",
            "detectedSegments": [
              {
                "occurrences": [
                  { "s": 362000, "e": 410055 },
                  { "s": 575390, "e": 629607 }
                ]
              }
            ]
          }
        ]
      },
      {
        "detectorId": "e79c540f-0d47-484e-859e-30d1ae6e4009",
        "detections": [
          {
            "detectorClass": 1,
            "detectorClassLabel": "ssn",
```

(continues on next page)

(continued from previous page)

```
            "detectedSegments": [
              {
                "occurrences": [
                  { "s": 202293, "e": 229835 }
                ]
              }
            ]
          }
        ]
      }
    ]
  }
}
```

For each detection, the API returns three data points:

- `detectorName` and/or `detectorId`: The type of sensitive data detected
- `detections`: array of the detected regions
- `s`: The start time of the detected region, in milliseconds
- `e`: The start end of the detected region, in milliseconds

16.4 PCI Detector

To enable it, add PCI detector and optional detectionLevel parameter to your configuration when you make a POST request to the /v3/media resource.

IMPORTANT NOTE: We recommend disabling Number Formatting on the PCI detector for best results.

```
{
  "transcript": {
    "formatting" : {
      "enableNumberFormatting" : false
    }
  },
  "prediction": {
      "detectors": [{
        "detectorName": "PCI",
        "parameters": [{
          "parameter": "detectionLevel",
          "value": "probableNumbers"
        }]
    }]
 }
}
```

16.5 SSN Detector

To enable it, add the SSN detector to your configuration when you make a POST request to the /v3/media resource.

IMPORTANT NOTE: We recommend disabling Number Formatting on the SSN detector for best results.

```
{
  "transcript": {
    "formatting" : {
      "enableNumberFormatting" : false
    }
  },
  "prediction": {
    "detectors": [
      { "detectorName": "SSN" }
    ]
  }
}
```

16.6 Number Detector

To enable it, add the Number detector to your configuration when you make a POST request to the /v3/media resource.

IMPORTANT NOTE: We recommend disabling Number Formatting on the Number detector for best results.

```
{
  "transcript": {
    "formatting" : {
      "enableNumberFormatting" : false
    }
  },
  "prediction": {
    "detectors": [
      { "detectorName": "Number" }
    ]
  }
}
```

16.7 Examples

** Note: Export your api `TOKEN` prior to running the following example.

```
export TOKEN='Your Api Token'
```

16.7.1 Enabling the detectors

```
curl https://apis.voicebase.com/v3/media \
  --header "Authorization: Bearer ${TOKEN}"  \
  --form media=@recording.mp3 \
  --form configuration='{
  "transcript": {
    "formatting" : {
      "enableNumberFormatting" : false
    }
  },
  "prediction": {
```

(continues on next page)

(continued from previous page)

```
    "detectors": [
      { "detectorName": "PCI" },
      { "detectorName": "SSN" },
      { "detectorName": "Number" }
    ]}
  }'
```

CHAPTER 17

PCI, SSN, PII Redaction

VoiceBase allows you to redact and/or detect sensitive data from your recordings, transcripts, and analytics.

17.1 Transcript redaction

To redact sensitive information from a media file, include a `redactor` section in the detector configuration when making a POST request to the /v3/media resource. In this example, any words in the detected region will be replaced with `[redacted]`.

For more on detectionLevel parameter options within the PCI model, please refer to the detect page.

IMPORTANT NOTE: We recommend disabling number formatting on the PCI detector for best results.

```
{
  "transcript": {
    "formatting" : {
      "enableNumberFormatting" : false
    }
  },
  "prediction": {
    "detectors": [ {
        "detectorName": "PCI",
         "redactor": {
            "transcript":{
                "replacement" : "[redacted]"
            }
         }
    } ]
  }
}
```

17.2 Analytics redaction

Sensitive data is automatically redacted from analytics (such as keywords and topics) when transcript redaction is enabled. No additional configuration is needed.

17.3 Audio redaction

To redact sensitive regions from your recording, include an `audio` section to the detector's redaction configuration when making a POST request to the /v3/media resource. In this example, sensitive audio regions will be replaced with a 170 Hz tone of moderate volume.

IMPORTANT NOTE: We recommend disabling Number Formatting on the PCI detector for best results.

```
{
  "transcript": {
    "formatting" : {
      "enableNumberFormatting" : false
    }
  },
  "prediction": {
    "detectors": [ {
        "detectorName": "PCI",
         "redactor": {
            "transcript":{
                "replacement" : "[redacted]"
            },
            "audio": {
                "tone": 170,
                "gain": 0.25
            }
         }
    }]
  }
}
```

To download the redacted audio, make a GET request to /v3/media/{mediaId}/streams. The response will be of the form:

```
{
  "streams": [{
    "streamName": "original",
    "streamLocation": "https://link.to.redacted.media"
  }]
}
```

An expiring link to download the audio file with redacted will appear in place of *https://link.to.redacted.media*. The link expires after 15 minutes. If the link has expired, perform a new GET request to /v3/media/{mediaId}/streams.

17.4 Examples

17.4.1 Transcript Redaction Request

```
curl https://apis.voicebase.com/v3/media \
  --form media=@recording.mp3 \
  --form configuration='{
      "transcript": {
        "formatting" : {
          "enableNumberFormatting" : false
        }
      },
      "prediction": {
        "detectors": [ {
            "detectorName": "PCI",
             "redactor": {
                "transcript":{
                    "replacement" : "[redacted]"
                }
             }
        }, {
           "detectorName": "SSN",
            "redactor": {
               "transcript":{
                   "replacement" : "[redacted]"
               }
            }
       }]
      }
  }' \
  --header "Authorization: Bearer ${TOKEN}"
```

17.4.2 Audio Redaction Request

```
curl https://apis.voicebase.com/v3/media \
  --form media=@recording.mp3 \
  --form configuration='{
      "transcript": {
        "formatting" : {
          "enableNumberFormatting" : false
        }
      },
      "prediction": {
        "detectors": [ {
            "detectorName": "PCI",
             "redactor": {
                "transcript": {
                    "replacement" : "[redacted]"
                },
                "audio": {
                    "tone": 170,
                    "gain": 0.25
                }
             }
```

(continues on next page)

(continued from previous page)

```
        }, {
            "detectorName": "SSN",
             "redactor": {
                "transcript": {
                    "replacement" : "[redacted]"
                },
                "audio": {
                     "tone": 170,
                     "gain": 0.25
                }
             }
       }]
      }
  }' \
  --header "Authorization: Bearer ${TOKEN}"
```

17.4.3 Redacted Audio Request

```
curl https://apis.voicebase.com/v3/media/${MEDIA_ID}/streams \
  --header "Authorization: Bearer ${TOKEN}"
```

CHAPTER 18

Player

The VoiceBasePlayer is available either as a React component designed to display and play transcripts and media, or as a Tableau Dashboard Extension available to customers as a manifest (.trex) file.

18.1 React Component

The full README and npm installation information are available here.

Once installed, populate the "/src/index.js" file in your React app with the JavaScript code below:

```
import React, { Component } from 'react';
import VoiceBasePlayer, { getVoicebaseMediaUrl, getVoicebaseAnalytics } from
↪'@voicebase/react-player';
import _ from 'lodash-core';
import ReactDOM from 'react-dom';


const centerText = {
  textAlign: 'center',
  fontSize: '1rem',
};

class PlayerWrapper extends Component {
  constructor(props) {
    super(props);
    this.state = {"token":"*VB_TOKEN*", "mediaId": ""};
  }

  static getDerivedStateFromProps(props) {
    return props;
  }

  componentDidMount() {
```

(continues on next page)

(continued from previous page)

```
    if (this.state.token && this.state.mediaId) {
      getVoicebaseMediaUrl(this.state.token, this.state.mediaId)
        .then(( url ) => {
          this.setState({ mediaUrl: url });
        })
        .catch((mediaUrlLoadError) => {
          this.setState({ mediaUrlLoadError });
        });

      getVoicebaseAnalytics(this.state.token, this.state.mediaId, this.state.
↪apiSuffix)
        .then((analytics) => {
          this.setState({ analytics });
        })
        .catch((analyticsLoadError) => {
          this.setState({ analyticsLoadError });
        });
    } else {
      console.log('no token/media id:', this.state);
    }
  }

  render() {
    if (this.state.mediaUrlLoadError || this.state.analyticsLoadError) {

      return (
        <VoiceBasePlayer error={true}>
          {this.state.mediaUrlLoadError && (
            <p style={centerText}>
              {_.get(this, 'state.mediaUrlLoadError.message', 'Error loading Media')}
            </p>
          )}
          {this.state.analyticsLoadError && (
            <p style={centerText}>
              {_.get(this, 'state.analyticsLoadError.message', 'Error loading␣
↪Analytics')}
            </p>
          )}
        </VoiceBasePlayer>
      );
    } else if (this.state.mediaUrl && this.state.analytics) {

      return <VoiceBasePlayer {...this.state} />;
    }


    return (
<h1>waiting</h1>
    );
  }
}


ReactDOM.render(
  <PlayerWrapper  />,
  document.getElementById('root'),
);
```

Please note that the line "this.state = {"token":"*VB_TOKEN*", "mediaId": ""};" contains the "token" and "mediaId" fields as an example only.

Once the Player is loaded, the "token" and "mediaId" fields may be added to the browser:

```
localhost:3000/#token=<jwt or vb token>#mediaId=<mediaId>
```

18.2 Tableau Dashboard Extension

Download the manifest file for the Tableau Dashboard Extension here.

CHAPTER 19

Predictions (Classifiers)

VoiceBase enables you to take advantage of machine learning to make business predictions from a recording.

VoiceBase runs the classifier when the recording is processed, and return a predicted class and confidence score. Common use cases of classifiers include:

- Identifying leads and prospects
- Automating quality control
- Predicting customer churn
- Detecting appointments, estimates, and orders
- Marking calls sent to voice mail

19.1 Adding a classifier

Please contact your VoiceBase Account Team if you are interested in adding a classifier to your account, and we can guide you through the process.

19.2 Listing classifiers

A list of classifiers available in your account can be obtained as follows by making a GET request on the `/definition/prediction/classifiers` resource.

```
curl https://apis.voicebase.com/v3/definition/prediction/classifiers \
     --header "Authorization: Bearer ${TOKEN}"
```

An example response:

```
{
  "_links" : {
    "self" : { "href": "/v3/definition/prediction/classifiers" }
  },
  "classifiers" : [
    {
      "classifierId": "c8b8c1f7-e663-4a05-a6b5-7e1109ee2947",
      "classifierName": "sales-lead",
      "classifierDisplayName": "Sales Lead",
      "classifierType": "binary",
      "categories": [
        { "categoryValue" : 1, "categoryLabel" : "sales-lead" },
        { "categoryValue" : 0, "categoryLabel" : "not-a-sales-lead" }
      ]
    }
  ]
}
```

- `classifiers` : array of classifier model objects.
  - `[n]` : a `model` object.
    - `classifierId` : The unique identifier of the classifier.
    - `classifierName` : the key name of the classifier.
    - `classifierDisplayName` : the display name of the classifier.
    - `classifierType` : The model type, one of:
      - `binary` : A binary classifier
      - `multiclass` : A classifier with multiple classes
    - `categories` : an array classes that the model detects.

19.3 Using a classifier

Enable a classifier by providing the classifier model revision identifier in the `predictions` section of the `configuration` when POSTing to /media. A classifier can be referred by its name or by its revision identifier. Classifiers referred by name are provided by VoiceBase.

```
{
  "prediction": {
    "classifiers": [
      { "classifierId" : "c8b8c1f7-e663-4a05-a6b5-7e1109ee2947" },
      { "classifierName": "sentiment" }
    ]
  }
}
```

- `prediction` : Prediction section
  - `classifiers` : List of classifiers to run
    - `[n]` : A classifier item.
      - `classifierId` : The unique identifier of a classifier model.
      - `classifierName` : The unique name of a classifier model.

When you GET the media results, the response will include the `prediction` section.

```
{
  "prediction": {
      "classifiers": [
        {
          "classifierDisplayName": "Sales Lead",
          "classifierId": "c8b8c1f7-e663-4a05-a6b5-7e1109ee2947",
          "classifierName": "sales-lead",
          "classifierType": "binary",
          "predictedCategory" : 0,
          "predictedCategoryLabel": "not-a-sales-lead",
          "predictionScore" : 1.729
        }
      ]
   }
}
```

- `prediction` : Prediction section
  - `classifiers` : List of results for each requested classifier
  - `[n]` : A classifier result.
    - `classifierId` : The unique identifier of the classifier model
    - `classifierName`: The classifier model name
    - `classifierType` : The type of the classifier.
    - `predictedCategory` : The class of the prediction expressed as an integer
    - `predictedCategoryLabel` : The name of the class
    - `predictionScore` : Probability score.

19.4 Examples

19.4.1 Example: Using a classifier

```
curl https://apis.voicebase.com/v3/media \
    --form media=@recording.mp3 \
    --form configuration='{
        "prediction":
          "classifiers" : [
             { "classifierId" : "630c31d0-f116-47a4-8b9f-3e7ac6313463" }
           ]
      }' \
    --header "Authorization: Bearer ${TOKEN}"
```

CHAPTER 20

Priority

With this feature you have the ability to define the processing priority of your POST request to /v3/media

Media can be submitted with different priority depending on the urgency of the request. This is helpful to obtain better pricing on archive processing, or to increase the speed with which transcripts are returned for time-sensitive items like voicemail.

VoiceBase offers low, medium, and high priority processing. Low priority is the default and least costly. If your use case requires medium or high priority processing, please contact Voicebase Sales to learn more about pricing.

CHAPTER 21

Reprocessing your files

VoiceBase allows you to re-run the post-processing with different options.

A common use-case for this is to align a previously generated machine transcript with a human edited or new machine transcript, then re-running knowledge extraction with the new more accurate transcript. This is documented in the aligner section.

You may also use the reprocessing feature to re-process different configuration options without requesting a new transcript.

For example, you may use the reprocessing feature to:

- Enable Knowledge Extraction, if you previously did not have this set
- Enable PCI Detection / Redaction if you decide you would like to remove PCI from your transcripts and / or audio files.
- Enable Number Formatting, if you previously set this to disabled.
- Run new machine learning models on existing media files.

Please Note: Custom Vocabulary modifies the speech engine with a custom set of terms for audio processing. Since the speech engine is not run during reprocessing, Custom Vocabulary is not updated.

21.1 Examples

Assume that MEDIA_ID='7eb7964b-d324-49cb-b5b5-76a29ea739e1' is a valid media ID of a previously uploaded file for transcription.

Make a POST request to `/media/${MEDIA_ID}` including a `configuration` attachment. Do *not* include a 'media' attachment or 'mediaUrl'

```
curl -v -s https://apis.voicebase.com/v3/media/$MEDIA_ID \
  --header "Authorization: Bearer ${TOKEN}" \
  -X POST \
  --form configuration='{}'
```

Then, make a GET request on the `/media/${MEDIA_ID}` resource to download the latest transcripts and configured analytics and predictions.

```
curl -v -s https://apis.voicebase.com/v3/media/$MEDIA_ID \
  --header "Authorization: Bearer ${TOKEN}"
```

CHAPTER 22

Search

VoiceBase can search across uploaded media documents **and** related results and metadata. This is intended for light use. We recommend building your own search if you have many end users.

To search media documents, one makes a GET request on the `/media` resource with any combination of the following acceptable query parameters:

- `after` : Document uploaded after this mediaId
- `externalId` : Documents with the specified external id in the metadata.
- `extendedFilter` : A special filter which is of the form 'extendedFilter=Name:Value' which allows you to filter by extended metadata. Please note that the value of this field should be url encoded.
- `onOrAfterDate` : Created on or after this date. Acceptable format: ISO8601 dates in either short form (YYYY-MM-DD) or including time (YYYY-MM-DDThh:mm:ssZ).
- `onOrBeforeDate` : Created on or before this date. Acceptable format: ISO8601 dates in either short form (YYYY-MM-DD) or including time (YYYY-MM-DDThh:mm:ssZ).
- `limit` : Limit of media documents returned.
- `sortOrder` : Sort order of media documents by date.
  - `asc` : Ascending by date. Oldest to Newest.
  - `desc` : Descending by date. Newest to Oldest.

22.1 Examples

** Note: Export your api `TOKEN` prior to running any of the following examples.

```
export TOKEN='Your Api Token'
```

22.1.1 Metadata-Scoped Search

To scope the search to records containing a specific metadata value, add a metadata filter parameter. For example, to search media with a "customerId" of "10101":

```
curl --header "Authorization: Bearer ${TOKEN}" \
https://apis.voicebase.com/v3/media?extendedFilter=customerId%3A10101
```

Searchable Field Definitions

To enable search over an extended metadata field, you must explicitly make the field searchable. Use the PUT /definition/media/search api endpoint to do so.

```
curl https://apis.voicebase.com/v3/definition/media/search \
  --header "Authorization: Bearer ${TOKEN}" \
  --header "Content-Type: application/json" \
  --request PUT \
  --data '{ "fields" : [ "extended.customerId" ] }'
```

22.1.2 Time-Range Restricted Search

To restrict a search to media for a specific time range, add filters for lower and/or upper bounds on the creation time and date of the media. For example, to find media only for the month of January 2016:

```
curl --header "Authorization: Bearer ${TOKEN}" \
https://apis.voicebase.com/v3/media?onOrAfterDate=2016-01-01&onOrBeforeDate=2016-02-01
```

VoiceBase supports ISO8601 dates in either short form (YYYY-MM-DD) or including time (YYYY-MM-DDThh:mm:ssZ).

22.1.3 Search Pagination

VoiceBase V2 uses cursor-based pagination to achieve pages that are stable even as media is concurrently added to the collection. Each page is governed by two parameters: "after", indicating the "mediaId" that immediately precedes the page, and "limit" which determines the size of the page (maximum implicit size is 1000). For example, if the first page ends with the "mediaId" of "f1ea0482-af9b-45d1-bd00-5eac31cd8259", the next page of 100 results is:

** Note: Export the mediaId for which you want more recent results.

```
export MEDIA_ID='The pivot mediaId'
```

Make a GET request on the `/media` resource with the `after` query parameter set.

```
curl --header "Authorization: Bearer ${TOKEN}" \
https://apis.voicebase.com/v3/media?after=${MEDIA_ID}
```

22.1.4 Search with compound expressions, metadata-scoping, time-range, and pagination

To achieve metadata-scoping, date range filters, compound expressions and pagination, all three expressions can be combined as shown in this GET request on the `/media` resource with the `onOrBeforeDate`, `onOrAfterDate`, `after` and `limit` parameters included.

```
curl --header "Authorization: Bearer ${TOKEN}" \
https://apis.voicebase.com/v3/media?extendedFilter=customerId%3A10101&
↪onOrAfterDate=2016-01-01&onOrBeforeDate=2016-02-01&limit=100&after=8d109ced-2627-
↪4427-8d8f-24a30f6b86b3
```

CHAPTER 23

Sentiment by Turn

23.1 Overview

Sentiment by Turn allows the user to track sentiment between the caller and agent on stereo calls. For more value, Sentiment by Turn may be configured with Verb-Noun Pairs so the user can understand the topical drivers of positive and negative sentiment.

23.2 Prerequisites for Use

- Calls MUST be processed in stereo for agent and caller specific metrics (otherwise, these metrics always return 0)
- The `agent` SHOULD have a speaker name that is one of: `agent`, `service`, `representative`, `operator`, `salesperson`, `callcenter`, `contactcenter`
- The `caller` SHOULD have a speaker name that is one of: `caller`, `client`, `customer`, `prospect`
- If the speaker names are not specified as above, the first speaker is assumed to be the agent, and the second speaker is assumed to be the caller
- Advanced Punctuation MUST be enabled.
- English and Spanish are the only supported languages for this feature.

23.3 Configuration

```
{
  "speechModel": {
    "language": "es-US",
    "features": [  "advancedPunctuation"  ]
  },
```

(continues on next page)

(continued from previous page)

```
  "ingest": {
    "stereo": {
       "left": { "speakerName": "Caller" },
      "right": { "speakerName": "Agent"    }
    }
  },

   "speakerSentiments":true

}
```

23.4 Output

```
{
  "speakerSentiments": [
    {
      "speakerName": "Caller",
      "sentimentValues": [
        {
          "s": 4558,
          "e": 7064,
          "v": -0.5434
        }, {
          "s": 9373,
          "e": 10345,
          "v": 0.7039
        }
      ]
    }, {
      "speakerName": "Agent",
      "sentimentValues": [
        {
          "s": 7464,
          "e": 9373,
          "v": 0.4328
        }, {
          "s": 12937,
          "e": 14627,
          "v": -0.3294
        }
      ]
    }
  ]
}
```

CHAPTER 24

Speech Engines

VoiceBase offers three distinct proprietary speech engines, each tuned to address different use cases. The table below shows feature support by engine.

Titan

The Titan engine is VoiceBase's default engine, offering the highest accuracy and best value for US English content. Trained on call center audio, Titan is a great choice for US based customers working with agent/customer calls. It supports US English only.

Europa

The Europa engine offers the highest accuracy for a wide range of regional accents for English and Spanish.

We recommend it for international use cases with combinations of English speakers from the UK, India, Singapore, and other regions.

No specific configuration is required when using its default language, US English. The default is the same for all regions and dialects of English.

Europa is trained mainly with European Spanish, as well as other variants. Europa Spanish may be configured as any one of the following: "language":"es-ES", "es-MX" or "es-US". The resulting transcript will be the same regardless of which variant is configured.

Please note Europa's price point is higher than our default engine, Titan.

Proteus

The Proteus engine is our general engine trained on a wide range of audio. All languages and features are supported with the Proteus engine.

Feature support by engine:

Feature	Europa	Proteus	Titan
categories			
custom vocabulary			
number formatting			
keywords-and-topics			
keyword-spotting			
languages	en, es		en-US
pci-ssn-pii-detection			
pci-ssn-pii-redaction			
stereo			
swear word filter			
verb-noun pairs			
voice features			
voicemail			

Please note that number formatting may not be disabled on the Europa speech engine. Custom Vocabulary is supported with Europa, but the "Weighting" feature is not.

24.1 Configuration

Enable a specific speech engine by including the `model` parameter under `speechModel` when making a call to `POST /media`. Omitting this optional parameter defaults to the Titan engine for US-English, and Proteus for other languages.

```
{
  "speechModel": {
    "model": "Titan"
  }
}
```

24.2 Custom Speech Models

VoiceBase also offers the unique ability to train a custom speech model on a per customer basis in order to optimize transcription accuracy for a specific use case, lexicon, accent, or codec. Please reach out to your account manager to learn more about the custom speech model training process.

24.3 Language Options

Information about VoiceBase language offerings may be found here.

CHAPTER 25

Stereo

Recording and processing conversations (such a phone calls) in stereo can significantly improve transcription accuracy and analytical insight. To realize the benefit, each speaker is recorded on a different channel (left or right), and the speaker metadata is provided to VoiceBase when uploading the recording.

25.1 Enabling stereo transcription

To enable one speaker per channel stereo transcription, add the "ingest" configuration when making a POST request to the /media resource and specify the label to use for each channel.

```
{
  "ingest": {
    "stereo": {
       "left": { "speakerName": "Customer" },
      "right": { "speakerName": "Agent"    }
    }
  }
}
```

```
- `ingest` : the ingest sub-section.
    - `stereo` : specification of the stereo channels.  Both child sections are␣
↪required.
        - `left` : specification of the left channel.
            - `speakerName` : the name of left channel speaker.
        - `right` : specification of the right channel.
            - `speakerName` : the name of right channel speaker.
```

25.2 Effects on Transcripts

When stereo processing is enabled, the word-by-word JSON transcript will return both a "words" array and a "turns" array, offering different options for parsing the data.

The "words" array contains nodes indicating the change of turn on the speaker. These nodes are identified by the attribute "m" set to "turn" and the attribute "w" set to the speaker's label provided in the configuration for one of the channels. The words following a turn node belong to the speaker identified by it until a new turn node appears in the transcript.

```
{
  "transcript" : {
    "confidence": 0.958,
        "words": [
            {
                "p": 0,
                "c": 1.0,
                "s": 1420,
                "e": 4260,
                "m": "turn",
                "w": "Agent"
            },
            {
                "p": 1,
                "c": 0.486,
                "s": 1420,
                "e": 1620,
                "w": "Hi"
            },
            {
                "p": 2,
                "c": 0.917,
                "s": 1630,
                "e": 1790,
                "w": "this"
            }
        ]
  }
}
```

The "turns" array contains the key "text", and its value is a block of text all pertaining to one speaker, as in the following example:

```
        "turns": [
            {
                "speaker": "Agent",
                "text": "Hi this is c.s.v. shipping company Brian speaking how can I␣
↪help you.",
                "s": 1420,
                "e": 4260
            }
           ]
```

The plain text version of the transcript will show each segment of the conversation prefixed with the speaker name (e.g. 'Agent:' or 'Customer')

```
curl https://apis.voicebase.com/v3/media/${MEDIA_ID}/transcript/text \
    --header "Accept: text/plain" \
    --header "Authorization: Bearer ${TOKEN}"
```

Agent: Hi this is c.s.v. shipping company Brian speaking how can I help you. Customer: This is Henry we spoke earlier I got a quote from you.

The SRT version of the transcript will also contain the speaker names provided in the configuration.

```
curl https://apis.voicebase.com/v3/media/${MEDIA_ID}/transcript/srt  \
    --header "Accept: text/srt" \
    --header "Authorization: Bearer ${TOKEN}"
```

```
1
00:00:02,76 --> 00:00:05,08
Agent: Well this is Taneisha
thank you for calling A.B.C.

2
00:00:05,08 --> 00:00:07,03
Cable services. How may I help you today.

3
00:00:08,28 --> 00:00:11,93
Customer: Hi I'm calling because I'm
interested in buying new cable services.

4
00:00:12,64 --> 00:00:16,43
Agent: OK great let's get started.
```

25.3 Effects on Keywords and Topics

When processing a file in stereo, you will get keywords and topics detected independently on each channel. The occurrences of the keywords are grouped by speaker.

```
{
  "knowledge": {
    "keywords": [
      {
        "name": "cable service",
        "relevance": "0.953",
        "mentions": [
          {
            "speakerName" : "Agent",
            "occurrences" : [
                { "s" : 5090, "e": 6070, "exact": "cable service" }
              ]
          },
          {
            "speakerName" : "Customer",
            "occurrences" : [
                { "s" : 234550, "e": 235700, "exact": "cable product" },
                { "s" : 347567, "e": 349000, "exact": "cable services" }
              ]
          }
        ]
      }
    ]
```

(continues on next page)

(continued from previous page)

```
  }
}
```

25.4 Effects on Audio Redaction

Audio redaction of PCI will redact the audio in both channels, irrespective of which channel contains the detected PCI data.

25.5 Examples

25.5.1 Processing in Stereo

```
curl https://apis.voicebase.com/v3/media  \
    --form media=@recording.mp3 \
    --form configuration='{
        "ingest": {
          "stereo": {
            "left" : { "speakerName": "Customer" },
            "right": { "speakerName": "Agent"    }
          }
        }
      }' \
    --header "Authorization: Bearer ${TOKEN}"
```

CHAPTER 26

Swagger Code Generation Tool

VoiceBase V3 API follows the OpenAPI 2.0 Specification (also known as Swagger RESTful API Documentation Specification).

By following the specification, the API is documented in a way that code generation tools can understand, saving you time and effort on generating code to build requests or access the response from the API.

The VoiceBase V3 OpenAPI specification resides at https://apis.voicebase.com/v3/defs/v3-api.yaml

This document is a summary of how to use the Swagger Code Generation tool to generate a VoiceBase V3 client. We assume that you already have installed Java 8 and Maven.

26.1 Download Swagger Codegen CLI tool

The following command uses Maven to download the Swagger Codegen tool.

```
mvn dependency:copy -Dartifact=io.swagger:swagger-codegen-cli:2.2.2 -
↪DoutputDirectory=. -Dmdep.stripVersion=true
```

26.2 Generating a client

Swagger Codegen allows you to generate a client in a wide variety of languages. You may check if your preferred programming language is available by executing

```
java -jar swagger-codegen-cli.jar langs
```

Here is a sample of the results:

```
Available languages: [android, aspnet5, aspnetcore, async-scala, bash, cwiki, csharp,
cpprest, dart, elixir, flash, python-flask, go, groovy, java, jaxrs, jaxrs-cxf-client,
jaxrs-cxf, jaxrs-resteasy, jaxrs-resteasy-eap, jaxrs-spec, jaxrs-cxf-cdi, inflector,
```

(continues on next page)

(continued from previous page)

```
javascript, javascript-closure-angular, jmeter, nancyfx, nodejs-server, objc, perl,
php, python, qt5cpp, ruby, scala, scalatra, finch, silex-PHP, sinatra, rails5, slim,
spring, dynamic-html, html, html2, swagger, swagger-yaml, swift, swift3, tizen,
typescript-angular2, typescript-angular, typescript-node, typescript-fetch, akka-
↪scala,
CsharpDotNet2, clojure, haskell, lumen, go-server, erlang-server, undertow, msf4j, ze-
↪ph]
```

The client code generation for a given programming language can be customized with several options. You can get a list of available options by executing

```
java -jar swagger-codegen-cli.jar config-help -l <language>
```

For example, if you want to know what options are available for the Scala language:

```
java -jar swagger-codegen-cli.jar config-help -l scala
```

26.2.1 Generating client code for Java

```
java -jar swagger-codegen-cli.jar generate  \
     -i https://apis.voicebase.com/v3/defs/v3-api.yaml \
     -l java \
     -c java-config.json \
     -o v3client
```

Where `java-config.json` is the file with the options for the generator:

```
{
  "modelPackage" : "com.example.v3client.model",
  "apiPackage"   : "com.example.v3client.api",
  "invokerPackage" : "com.example.v3client",
  "groupId" : "com.example.v3",
  "artifactId"  : "v3client",
  "fullJavaUtil" : true,
  "dateLibrary" : "java8",
  "library" :  "jersey2"
}
```

With the generated client, you may now submit a media for processing

```
  String baseApiUrl = "https://apis.voicebase.com/";
  String v3Token = "your-token-goes here";
  int connectTimeout = 60000;
  int readTimeout = 120000;

  // Create an ApiClient per thread
  ApiClient client = new ApiClient();

  client.setBasePath(baseApiUrl);
  client.setApiKey("Bearer " + v3Token);
  client.setConnectTimeout(connectTimeout);
  client.setDebugging(true);

  MediaApi mediaApi = new MediaApi(client);
```

(continues on next page)

(continued from previous page)

```
ObjectMapper objectMapper = new ObjectMapper();
objectMapper.configure(SerializationFeature.WRITE_EMPTY_JSON_ARRAYS, false);
objectMapper.configure(SerializationFeature.WRITE_NULL_MAP_VALUES, false);
objectMapper.configure(SerializationFeature.INDENT_OUTPUT, true);
objectMapper.setSerializationInclusion(Include.NON_NULL);

File mediaFile = new File("stereo-recording.wav");

// Create the configuration programmatically
VbConfiguration configuration = new VbConfiguration();
VbStereoConfiguration stereo = new VbStereoConfiguration();
VbChannelConfiguration leftChannel = new VbChannelConfiguration();
leftChannel.setSpeakerName("Agent");
stereo.setLeft(leftChannel);
VbChannelConfiguration rightChannel = new VbChannelConfiguration();
rightChannel.setSpeakerName("Caller");
stereo.setRight(rightChannel);
VbIngestConfiguration ingest = new VbIngestConfiguration();
ingest.setStereo(stereo);
configuration.setIngest(ingest);
String strConfiguration = objectMapper.writeValueAsString(configuration);

// And execute the request
VbMedia upload = mediaApi.postMedia(mediaFile, null, strConfiguration, null, null);
String mediaId = upload.getMediaId();
```

Alternatively, you could build the configuration yourself as a JSON string

```
  strConfiguration = "{'ingest': { 'stereo': { 'left': { 'speakerName' : 'Agent' },
↪'right':{ 'speakerName' : 'Caller' }}}}";
  VbMedia upload = mediaApi.postMedia(mediaFile, null, strConfiguration , null, null␣
↪);
```

Later on, you could retrieve the analytics and transcripts:

```
VbMedia analytics = mediaApi.getMediaById(mediaId, null);
switch (analytics.getStatus()) {
    case FINISHED:
        // Keywords
        if( analytics.getKnowledge() != null && analytics.getKnowledge().
↪getKeywords() != null) {
            System.out.println("**** Keywords: ");
            for( VbKeyword keyword : analytics.getKnowledge().getKeywords() ) {
                System.out.println( keyword.getKeyword() );
            }
        }
        // Topics
        if( analytics.getKnowledge() != null && analytics.getKnowledge().
↪getTopics() != null) {
            System.out.println("**** Topics: ");
            for(  VbTopic topic : analytics.getKnowledge().getTopics()) {
                System.out.println( topic.getTopicName() );
            }
        }

        // Retrieving the text transcript
```

(continues on next page)

(continued from previous page)

```
            String text = mediaApi.getTextById(mediaId);
            System.out.println(text);

            break;
        case FAILED:
            System.out.println("Transcription failed");
            break;
        default:
            System.out.println("Results not yet available, please wait...");
            break;
    }
```

CHAPTER 27

Swear Word Filter

You can filter out a preset list of offensive words (”swear words”) from English transcriptions. After the transcript is produced, the swear word filter redacts words on the disallowed list with “. . . ” . Download the list of disallowed words here.

27.1 How to Use It

To enable the swear word filtering, configure the media post configuration with the swearFilter parameter set to `true` in the `"transcript"` section:

```
{
    "transcript": {
      "contentFiltering": {
            "enableProfanityFiltering": true
      }
    }
}
```

27.2 Examples

** Note: Export your api `TOKEN` prior to running the following example.

```
export TOKEN='Your Api Token'
```

27.2.1 Enable the swear word filter

```
curl https://apis.voicebase.com/v3/media \
  --form media=@recording.mp3 \
  --form configuration='{
    "transcript": {
      "contentFiltering": {
            "enableProfanityFiltering": true
      }
    }
  }' \
  --header "Authorization: Bearer ${TOKEN}"
```

CHAPTER 28

TimeToLive (TTL)

TimeToLive refers to the default storage time limit set on uploaded data by VoiceBase.

Files uploaded to VoiceBase have a TTL of 121 days, at which point they are deleted. If users prefer a shorter TTL, they may contact support to request an adjustment.

CHAPTER 29

Transcoding

29.1 Overview

Transcoding is offered as an option to customers who wish to change the codec of a file uploaded to the VoiceBase platform. Reasons to use this option may include playing calls in browsers or BI tools that have codec limitations, or customer requirements to compress files due to concerns about file size.

The VoiceBase platform may need to transcode files if doing so is required to redact a file when that file's format is not supported. In that case, the platform will transcode the original file into a WAV file, perform the redaction and store this as the redacted file.

29.2 Configuration

To transcode a file, add the "publish" configuration when making a POST request to the /media resource:

```
{
   "publish" : {
      "audioTranscode" : {
         "onlyIfRedacted" : true,
         "fileFormat": "mp3|wav|flac|opus|ogg",
         "codec" : "mp3|pcm|vorbis..."
      }
   }
}
```

The "onlyIfRedacted" attribute indicates that the audio should be transcoded only if the audio is required to be redacted. Its default value is "false".

The "fileFormat" attribute is used for the original codec of the file. Its default value is "wav".

The "codec" attribute refers to the target codec. Its default value depends on the fileFormat, as does its set of supported values.

CHAPTER 30

Verb-Noun Pairs

30.1 Overview

Noun-Verb Pairs enables analysts and business users to organically discover topics of discussion within calls, email, chat, survey and social media without having to create categories. POS, specifically noun-verbs-pairs, pulls out the nouns and each noun's associated verb from each text sentence. The nouns are then paired with their associated verbs together from a call, set of calls, or text files to visualize the groupings of nouns (by count) and their associated verbs (by count) for each noun.

"Neg" and "Noun-neg" are optional fields. When present, they indicate a negative modifier for the verb or noun, respectively.

When the Verb-Noun Pairs feature is combined with Sentiment by Turn, nouns and verbs pairings will be able to carry a sentiment score and display sentiment by noun and verb cluster so the user can understand the topical drivers of positive and negative sentiment.

30.2 Configuration

```
{
   "speakerSentiments":true,
   "conversation":{
      "verbNounPairsEnabled":true
   }
}
```

30.3 Output

```
{
"noun-verb-pairs": [
{"timestamp": {"s": 6679, "e": 7260}, "V": "call", "N": "team", "speaker": "agent"},
{"timestamp": {"s": 16250, "e": 17340}, "V": "need", "N": "help", "speaker": "caller"}
↪,
{"timestamp": {"s": 44279, "e": 45099}, "V": "have", "N": "question", "neg": "not",
↪"speaker": "caller"},
{"timestamp": {"s": 807908, "e": 808918}, "V": "work", "N": "internet", "neg" : "not",
↪ "speaker": "caller"},
{"timestamp": {"s": 901234, "e": 902786}, "V": "work", "N": "internet", "neg" : "not",
↪ "speaker": "caller"},
{"timestamp": {"s": 1002560, "e": 1003010}, "V": "work", "N": "internet", "neg" :
↪"never", "speaker": "caller"},
{"timestamp": {"s": 1167845, "e": 1169252}, "V": "provide", "N": "justification",
↪"noun-neg" : "no", "speaker": "agent"},
{"timestamp": {"s": 1223475, "e": 1224005}, "V": "like", "N": "service", "question" :␣
↪true, "speaker": "agent"}
]
}
```


Corresponding sample text for each of the above pairs is given below:

- Calling team
- Need help with my account
- I do not have a question
- The stupid internet doesn”t work
- My internet is not working
- My internet never works
- They provide no justification
- Do they like our service

Please note that “N”, “V” and “neg” words are replaced by their lemma. For example, “working”, “works”, “worked” will all be denoted as “work” only and “doesn’t” will have “neg” value as “not”. This means that both “internet doesn’t work” and “internet is not working” will have the same values.

CHAPTER 31

Voice Features

This optional feature returns a set of per-word metrics, used in data science to serve as input attributes to a predictive model. Voice Features is supported for all regions of English as well as US and Latin American Spanish.

31.1 Uses for Voice Features

Voice Features metrics may be used in multiple ways. One example would be determining that a caller raised their voice and spoke at a higher pitch during a call, suggesting they may be upset. Making this determination requires that you process these per-word metrics to: 1) Create a function that transforms per-word metrics into utterance or time-leveled metrics 2) Gather baseline values for the caller 3) Track changes over time relative to the baseline 4) Set thresholds, and assign meaning to them.

31.2 Enabling Voice Features

Enable Voice Features by including `"voiceFeatures"` in the `features` array under `speechModel` when making a call to `POST /media`.

```
{
  "speechModel": {
    "features": [
        "voiceFeatures"
    ]
  }
}
```

The `configuration` contains the following fields:

- `speechModel`: speech model section
- `features`: speech model features to enable

31.3 Voice Features Results

When you GET the media results, the response will include the "frq" array, populated with "f"(frequency),"e"(energy), and "v"(volume) values for each word.

"f" is the frequency in Hertz,computed from the dominant and secondary peaks of each word. The maximum value is 8khz or 16khz, depending on submitted audio.

"e" is the relative energy (amplitude) of the frequency. The value will be between 0 and 1.

"v" is the relative volume. Volume is determined by a formula where v = A * M. A is the average between the words timing and the frequencies amplitude and M is the maximum amplitude value from any frequency and spectrum frames within the words. The value can be unlimited but it is typically between 0 and 2. Any value greater than 2 can be cropped to 2.

For example:


```
{
  "mediaId": "bc14632d-e81b-4673-992d-5c5fb6573fb8",
  "status": "finished",
  "dateCreated": "2017-06-22T19:18:49Z",
  "dateFinished": "2017-06-22T19:19:27Z",
  "mediaContentType": "audio/x-wav",
  "length": 10031,
  "transcript": {
     "words":[
        {
            "p": 0,
            "s": 1880,
            "c": 0.461,
            "frq": [
              {
                "e": 1,
                "f": 260.942
              },
              {
                "e": 0.347,
                "f": 521.807
              }
            ],
            "e": 2180,
            "v": 14.876,
            "w": "Because"
          }
     ]
  }
}
```

CHAPTER 32

Visual Voicemail

VoiceBase is able to return automatic transcripts for voicemail recordings which can then be delivered via email or SMS.

Below is an optimized configuration for fast and accurate voicemail transcription. This includes features that benefit shorter forms of audio, however they are all optional.

- **Disable Phrase Groups and Topic Extraction:** Disabling semantic knowledge discovery improves turnaround time and is generally not needed for short files.
- **Enable Number Formatting:** This will format numbers as digits instead of words and adds US phone number formatting which will enable click-to-call for users.
- **Use Callbacks:** By using callbacks, our servers will send the data as soon as processing is finished. You won't have to wait until the next polling interval.

32.1 Sample configuration for English Voicemail with a callback endpoint receiving JSON

```
{
   "speechModel" : {
         "language" : "en-US",
          "extensions" : [ "voicemail" ]
   },
   "knowledge": {
     "enableDiscovery" : false
   },
   "transcript": {
     "formatting" : {
       "enableNumberFormatting" : true
     }
  },
  "publish":{
```

(continues on next page)

(continued from previous page)

```
    "callbacks": [
        {
            "url": "https://example.com/transcription",
            "method": "POST",
            "type": "analytics",
            "include":[
                "transcript"
            ]
        }
    ]
  }
}
```

32.2 Sample configuration for Voicemail with a callback endpoint receiving plain text

You may want a callback just with the text transcript. In this case, you should use a PUT operation to a unique URL that is associated with the recording on your system and the endpoint must accept content type "text/plain"

```
{
    "speechModel":{
        "language":"en-US",
        "extensions": [ "voicemail" ]
    },
    "knowledge":{
        "enableDiscovery":false
    },
    "publish":{
        "callbacks":[
            {
                "url":"https://example.com/transcription/recording-7d66f194786d",
                "method":"PUT",
                "type":"transcript",
                "format":"text"
            }
        ]
    }
}
```

Now your customers can quickly scan and read through the important parts of their voicemail messages without having to listen to every single recording in their inbox! Here's a quick how-to video using Twilio recording.

CHAPTER 33

Client Supplied Encryption

PKI (Public Key Infrastructure) encryption technology is leveraged to provide additional security for customers using the VoiceBase platform. The customer provides us with their PGP "public key", which can be used to encrypt the data, but not to decrypt it. The customer keeps their "private key" and never sends it to VoiceBase. When the media is processed, we encrypt the results using the customer's "public key" for storage and delivery to the customer. The customer receives the encrypted data, and uses the "private key" to decrypt and use the results.

Users may create a PGP key using PGP or GPG tools. We recommend to use an "RSA and RSA" key with 2048 bits, with one-year crypto-period.

33.1 Upload the public key to VoiceBase

```
curl -s "https://apis.voicebase.com/v3/security/public-keys"
--header "Authorization: Bearer $TOKEN"
--header "Content-Type: application/octet-stream"
--data-binary @./my-public-key.asc
```

The upload will return a "publicKeyId". Use this ID when uploading the media. At this point the key is ready to be used.

33.2 Using POST /v3/media with a public key

Add the publicKeyId in your v3 API Configuration:

```
{
    "encryption" : {
        "publicKeyId" : "the unique identifier"
     },
    "searchability" : {
        "searchable" : false
```

(continues on next page)

(continued from previous page)

```
    }
}
```

33.2.1 Behavior of GETs, POSTs and DELETE when Client Supplied Encryption has been applied:

`GET /v3/media?query={query}` Because the media is encrypted, you will be unable to search by words in the transcript or metadata.

`GET /v3/media/{mediaId}` will display processing progress, but will not show the transcript contents. mediaQuery — will not index or show encrypted media. mediaGetById — Will work as normal until the media has finished processing.

`GET /v3/media/{mediaId}/transcript` Will return a 404 if the media is encrypted.

`GET /v3/media/{mediaId}/streams` Will list the streams available. It will only return original-encrypted or redacted-encrypted as streams, rather than original or redacted, if encryption was applied.

`GET /v3/media/{mediaId}/streams/{streamName}` Will return the audio or video stream specified. *Note:* names of available streams are given by the previous API endpoint.

`GET /v3/media/{mediaId}/progress` Will return processing progress.

`GET /v3/media/{mediaId}/metadata` Will return 404 as the media metadata is encrypted.

`DELETE /v3/media/{mediaId}` Will delete media.

`POST /v3/media/` Will allow you to post media to the API.

`POST /v3/media/{mediaId}` Alignment and reprocessing will fail as the original media is encrypted.

33.3 Encryption Key Management

`GET /v3/security/public-keys` Returns a list of the (public) encryption keys available. This only returns href to the URLs with the keys

`POST /v3/security/public-keys` Stores the (public) encryption key. The body must be the PEM format of the public key. You will get back a publicKeyId, which is a UUID.

`GET /v3/security/public-keys/{publicKeyId}` Returns the current (public) encryption key as it was originally provided through the POST operation

`DELETE /v3/security/public-keys/{publicKeyId}` Deletes the (public) encryption key associated with the publicKeyId.

CHAPTER 34

HTTPS Request Security

VoiceBase uses an advanced Web Application Firewall (WAF) to detect and reject requests that are likely to pose a security risk.

In order to validate and filter out security threats as early as possible, VoiceBase has additional requirements for HTTPS requests. Requiring clients to follow these requirements allows us to monitor, mitigate and response to security threats, keeping the system secure for all legitimate users.

Sending requests to VoiceBase that do not follow these requirements can result in suspension of your credentials and/or blocking of your traffic.

This page uses MUST, MUST NOT, SHOULD, and SHOULD NOT as defined in RFC 2119.

34.1 All requests

The following requirements apply to all requests, regardless of type:

- All API requests to VoiceBase MUST contain a valid `Authorization` header, with a valid Scheme. The only supported scheme is `Bearer` (OAuth Bearer tokens).

34.2 GET requests

The following requirements apply to `GET` requests:

- `GET` requests MUST NOT contain a body, expect when the body is a query string.
- `GET` requests SHOULD include an `Accept` header that includes the type returned by the API as acceptable (most APIs return `application/json`). Omitting the Accept header is interpreted as `Accept: */*`, but this is not recommended.

34.3 PUT requests

The following requirements apply to `PUT` requests:

- `PUT` requests MUST specify the `Content-Type` of the body
- `PUT` requests MUST specify the `Content-Length` of the body
- `PUT` request type size MUST match the specified `Content-Type`
- `PUT` request body size MUST match the specified `Content-Length`
- `PUT` requests over 1MB in size SHOULD use Chunked-Transfer encoding and 100-Continue

34.3.1 API-Specific Details

All currently available `PUT` APIs support `application/json` as the required `Content-Type`.

34.4 POST requests

The following requirements apply to `POST` requests:

- `POST` requests MUST specify the `Content-Type` of the body
- `POST` requests MUST specify the `Content-Length` of the body
- `POST` request type size MUST match the specified `Content-Type`
- `POST` request body size MUST match the specified `Content-Length`, unless Chunked-Transfer is used
- `POST` requests with Chunked-Transfer encoding MUST specify the `Content-Length` of each chunk
- `POST` requests over 1MB in size SHOULD use Chunked-Transfer encoding and 100-Continue

34.4.1 API-Specific Details

Except for multi-part `POST` APIs specified below, all currently available `POST` APIs support `application/json` as the required `Content-Type`.

34.4.2 Multi-part POST requests

The following additional requirements apply to multi-part POST requests:

- Attachment names MUST be unique
- Attachments MUST specify a `Content-Type` and `Content-Length`
- Attachment length MUST match the specified `Content-Length`
- Attachment type MUST match the specified `Content-Type`

Currently available Multi-part POST APIs are: `/media` and `/media/{mediaId}`. These APIs support `multipart/form-data` as the required `Content-Type`.

34.5 DELETE requests

The following requirements apply to `DELETE` requests:

- `DELETE` requests MUST NOT contain a body, expect when the body is a query string.
- `DELETE` requests SHOULD include an `Accept` header that includes the type returned by the API as acceptable (most APIs return `application/json`). Omitting the `Accept` header is interpreted as `Accept: */*`, but this is not recommended.

34.6 TLS Security

Requests must use TLS v1.2 protocol. We continually update our TLS policies to maintain a grade of A- of better from SSL Labs. Requests that specify weaker TLS security are rejected.

CHAPTER 35

Working with Media (/media)

The */media* resource contains the media files you have uploaded to VoiceBase for analysis, as well as transcripts and analytics generated from processing these files.

You can find more details around the */media* resource in the API Console for /media in the Developer Portal.

CHAPTER 36

Customizing VoiceBase (/definitions)

The */definitions* resource contains reference data that can be used in processing, including keyword or phrase spotting groups, custom vocabularies, or predictive models.

You can find more details around the */definitions* resource in the API Console for /definitions in the Developer Portal

36.1 Keyword Spotting Groups

Keyword spotting groups are specified using the */definitions/keywords/groups* resource. See API Console for /definitions in the Developer Portal for more details.

36.2 Custom Vocabularies

Custom vocabularies are specified using the */definitions/transcripts/vocabularies* resource. See API Console for /definitions in the Developer Portal for more details.

..include:: definitions/predictions-models.rst

CHAPTER 37

Managing Access (/profile)

The */profile* resource contains user profile information, and get can be used to manage your Bearer tokens.

You can find more details around the */profile* resource in the API Console for /profile in the Developer Portal.

CHAPTER 38

Sample configuration

Please see notes below for details about specific lines.

```
{
  "speechModel": {
    "language": "es-US",
    "features": [  "advancedPunctuation"  ]
  },
  "ingest": {
    "stereo": {
      "left": {
        "speakerName": "Agent"
      },
      "right": {
        "speakerName": "Caller"
      }
    }
  },
  "prediction": {
    "detectors": [
      {
        "detectorName": "PCI",
        "redactor": {
          "transcript": {
            "replacement": "[redacted]"
          },
          "audio": {
            "tone": 170,
            "gain": 0.5
          }
        }
      }
    ]
  },
  "spotting": {
```

(continues on next page)

(continued from previous page)

```
33      "groups": [
34        {
35            "groupName": "finance"
36        }
37      ]
38    },
39    "knowledge": {
40        "enableDiscovery": true
41
42    },
43    "transcript": {
44        "formatting": {
45          "enableNumberFormatting": false
46        },
47        "contentFiltering": {
48          "enableProfanityFiltering": true
49        }
50    },
51    "vocabularies": [
52      {
53        "vocabularyName": "earningsCalls"
54      },
55      {
56        "terms": [
57          {
58            "term": "VoiceBase",
59            "soundsLike": [ "voice base" ],
60            "weight": 1
61          },
62          {
63            "term": "Malala Yousafzai"
64          }
65        ]
66      }
67    ],
68    "publish": {}
69  }
```

Notes:

- line 4 (`"features": [ "advancedPunctuation" ]`): Advanced Punctuation is supported in English and Spanish only.
- line 6 (`"ingest": {}`): speakerName and stereo are mutually exclusive
- line 7 (`"stereo": {}`): for mono scenarios, specify `"speakerName": "Speaker"` instead of `stereo` - all channels will be mixed into a single audio channel prior to speech recognition processing
- line 35 (`"groupName": "finance"`): this spotting group must be created before it is used
- line 40 (`"enableDiscovery": true`): Default is false - knowledge discovery is disabled by default
- line 45 (`"enableNumberFormatting": false`): Default is true
- line 48 (`"enableProfanityFiltering": true`): Default is false
- line 53 (`"vocabularyName": "earningsCalls"`): this vocabulary must be created before it is used
- line 60 (`"weight": 1`): weights range from 0 to 5, 0 being default weight

CHAPTER 39

media

Below is a sample of the media response.

```
{
  "_links": {},
  "formatVersion": "3.0.7",
  "mediaId": "efbb8c49-87f0-4f6c-9ce9-781599918f8c",
  "accountId": "710d1652-63a4-4355-8e9a-523ddacd3066",
  "accountName": "ACME Inc",
  "status": "finished",
  "dateCreated": "2017-04-28T21:12:55.563Z",
  "dateFinished": "2018-07-19T11:34:45.134Z",
  "timeToLiveInSeconds": 1200,
  "expiresOn": "2018-10-04T00:41:06.145Z",
  "metadata": {
  "title": "Inbound call 2018-07-01 from 15081231234",
  "description": "Inbound call to 1-800-599-1234, ACME Support Center",
  "externalId": "inbound-dd9e7002-4a5a-43b3-bd46-73a66362db29",
  "extended": {
   "anything": "goes here",
     "nested": {
       "is": 0,
       "also": 0,
       "accepted": 0
      }
    }
  },
  "mediaContentType": "audio/mpeg",
  "length": 66900, // Duration in ms of the audio
  "knowledge": {
    "keywords": [
     {
       "keyword": "Microsoft",
       "relevance": 0.433,
       "mentions": [
```

(continues on next page)

(continued from previous page)


```
        {
          "speakerName": "Speaker 1",
          "occurrences": [
            {
              "s": 34234,
              "e": 34234,
              "exact": "Microsoft"
            }
           ]
         }
       ]
     }
   ],
   "topics": [
    {
      "topicName": "Algorithms",
      "relevance": 0.4353,
      "subtopics": [],
      "keywords": []
     }
   ]
 },
  "spotting": {
    "groups": [
      {
       "groupName": "Competitors",
       "spotted": false,
       "score": "0.4439",
       "spottedKeywords": [
        {
          "keyword": "Microsoft",
          "relevance": 1,
          "mentions": [
            {
              "speakerName": "Speaker 1",
              "occurrences": [
                {
                  "s": 34234,
                  "e": 34234,
                  "exact": "Microsoft"
                }
               ]
             }
            ]
          }
        ]
      }
    ]
  },
  "prediction": {
    "classifiers": [
      {
      "classifierId": "3e8dee45-ae2a-432f-b5ff-aa2513986f23",
      "classifierName": "SaleCompleted",
      "classifierVersion": "1.0",
      "classifierDisplayName": "Sales Completed",
      "classifierType": "binary",
```


(continues on next page)

(continued from previous page)

```
      "predictedClassLabel": "completed",
      "predictionScore": 0.929,
      "predictedClass": 1
      }
    ],
      "detectors": [
        {
          "detectorId": "99179da8-2ef4-4478-92d0-f296399a90b7",
          "detectorName": "PCI",
          "detectorVersion": "1.0",
          "detectorDisplayName": "Detects credit card data",
          "detectorType": "binary",
          "detections": [
            {
            "detectorClass": 1,
            "detectorClassLabel": "pci",
            "detectedSegments": [
              {
                "speakerName": "Speaker 1",
                "occurrences": [
                   {
                    "s": 34322,
                    "e": 458375
                   }
                 ]
              }
            ]
          }
        ]
     },
    "metrics": [
      {
      "metricGroupName": "groupX",
      "metricValues": [
        {
          "metricName": "xyz",
          "metricValue": 200
        }
      ]
     }
    ],
    "categories": [
      {
      "categoryName": "abc",
      "categoryValue": 0
    }, {
      "categoryName": "def",
      "categoryValue": 1,
      "categoryMatches": [
        {
          "speakerName": "caller",
          "occurrences": [
            {
              "s": 24251,
              "e": 24981,
```

(continues on next page)

(continued from previous page)


```
              "exact": "hello"
            }, {
              "s": 26491,
              "e": 30571,
              "exact": "hi"
            }
          ]
        }, {
          "speakerName": "agent",
          "occurrences": [
            {
              "s": 24251,
              "e": 24981,
              "exact": "greetings"
            }, {
              "s": 26491,
              "e": 30571,
              "exact": "how are you"
            }
          ]
        }
      ]
    }
  ],
  "conversation": {
    "speakerVerbNounPairs": [
      {
        "speakerName": "Agent",
        "verbNounPairs": [
          {
            "s": 6679,
            "e": 7260,
            "verb": "call",
            "noun": "team"
          }, {
            "s": 44279,
            "e": 45099,
            "verb": "have",
            "verbNeg": "not",
            "noun": "question"
          }, {
            "s": 807908,
            "e": 808918,
            "verb": "like",
            "noun": "service",
            "question": true
          }
        ]
      }, {
        "speakerName": "Caller",
        "verbNounPairs": [
          {
            "s": 16250,
            "e": 17340,
            "verb": "need",
            "noun": "help"
          }, {
```


(continues on next page)

(continued from previous page)

```
            "s": 901234,
            "e": 902786,
            "verb": "provide",
            "noun": "justification",
            "nounNeg": "no"
          }, {
            "s": 1002560,
            "e": 1003010,
            "verb": "work",
            "verbNeg": "not",
            "noun": "internet"
          }
        ]
      }
    ]
  },
  "speakerSentiments": [
    {
      "speakerName": "Caller",
      "sentimentValues": [
        {
          "s": 4558,
          "e": 7064,
          "v": -0.5434
        }, {
          "s": 9373,
          "e": 10345,
          "v": 0.7039
        }
      ]
    }, {
      "speakerName": "Agent",
      "sentimentValues": [
        {
          "s": 7464,
          "e": 9373,
          "v": 0.4328
        }, {
          "s": 12937,
          "e": 14627,
          "v": -0.3294
        }
      ]
    }
  ],
  "transcript": {
    "confidence": 1.0,
    "words": [
      {
        "p": 3,
        "c": 0.845,
        "s": 13466,
        "e": 15648,
        "m": "turn|punc",
        "v": 34,
        "w": "supercalifragilisticexpialidocious",
        "frq": [
```

(continues on next page)

(continued from previous page)

```
            {
              "e": 1.344,
              "f": 234.0
            }, {
              "e": 2.344,
              "f": 340.0
            }
          ]
        }
      ],
      "voiceActivity": [
        {
          "speakerName": "Speaker 1",
          "occurrences": [
            {
              "s": 13000,
              "e": 150547
            }, {
              "s": 163746,
              "e": 258726
            }
          ]
        }
      ],
      "alternateFormats": [
        {
          "format": "srt",
          "contentType": "text/srt",
          "contentEncoding": "base64",
          "charset": "utf-8",
          "data": "A Base64 encoded transcript"
        }
      ]
    },
    "streams": [
      {
        "status": "HTTP Status of the stream. Are we using this?",
        "streamName": "original",
        "streamLocation": "https://somewhere.voicebase.com/xyzt&amp;expires=12344",
      }
    ],
    "encryption": {
      "publishKeyId": "11e13265-e688-428b-b7bb-708c12a30a41",
      "publicKeyHash": "A SHA256"
    }
}
```

CHAPTER 40

Voicebase API Error Codes and Messages

Code	Message
10000	The API you requested was not found. The available endpoints are defined in the
—	VoiceBase API documentation.
10001	The request submitted to VoiceBase contains an invalid http multipart request.
—	Please review the code calling the api to ensure proper use of the http
—	multipart request.
10002	Your request could not be authorized because it did not contain an
—	Authorization header. Please review the VoiceBase authorization documentation
—	for instructions on how to obtain and include an Authorization header in your
—	request.
10003	The Authorization header you provided is invalid and thus the request you
—	submitted is not permitted. Please provide a valid VoiceBase access token.
10004	The account associated with the request you made has been locked. To unlock
—	your account, please contact VoiceBase support.
10005	The account associated with the request is pending approval by VoiceBase. If
—	you wish to expedite this process, please contact VoiceBase support.
10006	The access token submitted with the request is not authorized to access the

Continued on next page

Table 1 – continued from previous page

Code	Message
—	resource specified. Please consult the VoiceBase documentation on access
—	tokens and authorization.
10007	We could not parse the configuration element associated with your request. May
—	we suggest using the open source tool 'jq' (https://stedolan.github.io/jq/) to
—	identify the issues with your configuration.
10008	The property (%s) is not part of the %s configuration. VoiceBase uses strict
—	configuration validation and extra properties are not allowed.
10009	We were unable to identify the priority specified in the configuration. Please
—	provide one of low|normal|high.
10010	The configuration contains a duplicated property: '%s'. Please remove one of
—	the duplicated properties.
10011	The VoiceBase api does not support non-url-encoded query variable values.
—	Please update your request to appropriate encode these values.
10012	You are attempting to access the profile keys api with a VoiceBase api access
—	token, but this is not allowed. Please resubmit your request with an VoiceBase
—	user access token.
10013	Please submit your request with an VoiceBase user access token.
10107	We could not parse the metadata element associated with your request. May we
—	suggest using the open source tool 'jq' (https://stedolan.github.io/jq/) to
—	identify the issues with your metadata.
10108	You provided an externalId property in your metadata, but it was empty. If you
—	include the externalId property, please supply a value of length less than %s.
10109	You provided an externalId property in your metadata, but it was too long. If
—	you include the externalId property, please supply a value of length less than
—	s.
10110	The field '%s' is included more than once in the metadata associated with the
—	request. Please include each metadata property only once.

Continued on next page

Table 1 – continued from previous page

Code	Message
10201	The media file submitted to be processed is malformed in some way and is un-
—	readable by VoiceBase.
10202	The request included neither a media attachment nor a mediaUrl. One of these
—	two is required for us to process your request.
10203	The media url you supplied (%s) is unreachable by VoiceBase. This may be
—	because the resource is behind a firewall.
10204	The media url you supplied (%s) is malformed. Please verify that the url is
—	conformant to internet standards.
10205	Requested resource not found
10206	The request submitted contains too many '%s' attachments. The maximum is 1.
—	Please review the request and ensure only one is included.
10207	We were not able to download a media file from the url supplied with the
—	request (%s). VoiceBase is receiving (%s) status code.
10301	You have requested a transcript from a media file that has not yet completed
—	processing. Transcripts only become available when a media item has status
—	finished. Please try again in a while.
10401	The language specified in your configuration %s is not supported by the speech
—	engine also specified in the configuration. Please consult the VoiceBase
—	language documentation for a compatibility map.
10402	The '%s' language requires a locale to be specified. Please re-submit your
—	request with a supported locale. Supported locales are: %s
10501	The property (testability) is not part of the configuration. Please remove the
—	property and try again.
10601	VoiceBase does not yet support the language specified in the configuration (
—	language: %s). Please contact support@voicebase.com regarding languages
—	support timelines.
10602	The speech model extension '%s' specified in the configuration is not supported
—	in language: %s. Please remove the speech model extension from the
—	configuration.
10603	The speech model feature '%s' specified in the configuration is not supported

Continued on next page

Table 1 – continued from previous page

Code	Message
—	in language: %s. Please remove the speech model feature from the configuration.
10604	Additional speech model are not yet supported
10800	The request payload is larger than %s bytes. Please contact VoiceBase support
15000	The configuration for channels is not supported. Please specify one of (
—	speakerName, stereo or channels)
15001	The configuration includes a channels parent property, but no channels were
—	found. Please provide list of channels.
15002	The configuration includes channels, but no speaker was found. Please provide
—	speaker for the channel configuration.
15003	The configuration includes channels, but only one was found. Please provide
—	both channels (Right and Left)
15004	The configuration includes %s channel(s) and the media contains %s channel(s)
—	Please provide configuration for all channels in the media.
15005	The configuration ignores all the available channels. Please specify one of
—	the channels to process.
20001	VoiceBase could not find the speech model specified in the configuration.
—	Please provide a correct model identifier.
20002	The configuraton specifies a speech model '%s', but the version is incorrect.
—	Please provide a valid speech model version.
20003	The configuration specifies an unsupported transcription engine. Please
—	provide one from the list of supported engines.
20004	The configuration enables alignment and also transcription configuration.
—	Please remove the transcription configuration.
20005	The configurations enables alignment and specifies the language '%s', but
—	alignment not available for the selected language. Please review the list of
—	supported languages for alignment in the documentation.
20006	The configuration contains vocabulary configuration, but does not specify a
—	named collection nor terms. Please provide one or the other.
20007	Vocabulary contains a vocabulary property with both a named collection (%s) AND
—	terms (%s). Only one or the other is acceptable. You may update the

Continued on next page

Table 1 – continued from previous page

Code	Message
—	configuration to include the named collection in one element and the terms in
—	another.
20008	The configuration specifies a vocabulary property with a named collection (%s)
—	but VoiceBase could not find it. Please consult the list of named vocabulary
—	collections and correct the configuration.
20009	The configuration specifies a vocabulary property with a named collection (%s)
—	but it is empty. Please either remove the empty collection or update it to
—	include terms.
20010	The configuration includes a vocabulary specification whose number of terms
—	exceeds the maximum of %s. Please reduce the vocabulary configuration.
20101	The configuration includes a vocabulary specification whose number of scripts
—	exceeds the maximum of %s. Please reduce the vocabulary configuration.
20012	s. Please correct the error.
20013	The configuration includes a vocabulary specification whose number of words in
—	the scripts exceeds the maximum of %s. Please reduce the vocabulary
—	configuration.
20014	The configuration includes a vocabulary specification with an empty term.
—	Please remove the empty term or include a value.
20015	The configuration specifies a vocabulary term (%s) with an invalid weight (%s)
—	Please update the configuration to specify a vocabulary weight that is an
—	integer between 0 to 5.
20016	s. Please correct the error.
20017	The following terms (%s) are duplicated in the vocabularies configuration
—	submitted with the request. Please update the configuration to ensure terms
—	are unique across the saved vocabularies and adhoc vocabularies.
20011	The configuration specifies diarization, but it is not available for the
—	selected language: %s. Please remove this property.

Continued on next page

Table 1 – continued from previous page

Code	Message
40001	The configuration specifies a semantic configuration with either topics or
—	keywords to be true. VoiceBase requires both to be true or both to be false.
—	Please update the semantic configuration to be either both true or both false.
40002	The configuration specifies a keywords semantic configuration and the language
—	s'. VoiceBase does not support keywords semantic search for the language.
—	Please remove the keywords semantic configuration for this media.
40003	The configuration specifies a topics semantic configuration and the language '%
—	s'. VoiceBase does not support topics semantic search for the language. Please
—	remove the topics semantic configuration for this media.
40004	The configuration specifies knowledge discovery and the language '%s'.
—	VoiceBase does not support knowledge discovery for the language. Please remove
—	the knowledge discovery configuration for this media.
40005	The configuration specifies a keyword spotting group (%s), but VoiceBase could
—	not find it. Please update the configuration to specify a keyword spotting
—	group in your definitions, or remove the property from the configuration for
—	this media.
40006	The configuration specifies a keyword spotting group (%s), but the definition
—	contains an empty collection. Please please add to the keyword spotting group,
—	or remove the specified group from the configuration.
50001	The configuration specifies a prediction model (%s), but it is not available
—	for the selected language (%s). Please please remove the model from the
—	configuration.
50002	The configuration specifies an prediction model that is missing an identifier
—	or name. Please please add an identifier or name to the model, or remove the
—	property from the configuration.
50003	The configuration specifies a prediction model with a classifierId (%s) that is
—	not not properly formatted. ClassifierIds must be UUIDs (https://en.wikipedia.
—	org/wiki/Universally_unique_identifier). Please correct the configuration.

Continued on next page

Table 1 – continued from previous page

Code	Message
50004	The configuration includes a detector element without a detectorId or a
—	detectorName. Please update the configuration to include one or the other.
50005	The configuration specifies a detection model with a dectorId (%s) that is not
—	not properly formatted. DetectorIds must be UUIDs (https://en.wikipedia.org/
—	wiki/Universally_unique_identifier). Please correct the configuration.
50006	The configuration specifies a detection model (%s), but it is not available
—	for the selected language (%s). Please please remove the model from the
—	configuration.
50007	The configuration includes a redaction element, but it does not specify audio
—	or transcript. Please update the configuration to specify one or the other (
—	audio or transcript)
50008	The configuration specifies a redaction element, but it is missing the property
—	s. Please update the configuration to specify the %s.
50009	The configuration specifies transcript redaction and the language %s'.
—	VoiceBase does not support transcript redaction in this language. Please
—	update the configuration to remove the transcript redaction element.
50010	The configuration specifies audio redaction and the language '%s'. VoiceBase
—	does not support audio redaction in this language. Please update the
—	configuration to remove the audio redaction element.
50011	VoiceBase does not support audio redaction for the content-type '%s'. Please
—	update the configuration to remove the audio redaction element.
55000	The configuration specifies content filtering and the language '%s' and region
—	s'. VoiceBase does not support content filtering in this language and region
—	combination. Please remove the content filtering element from the
—	configuration.
60001	The configuration specifies a callback element without a url. Without a url,

Continued on next page

Table 1 – continued from previous page

Code	Message
—	VoiceBase cannot make the callback. Please update the configuration provide
—	callback url.
60002	The configuration specifies a callback url (%s) with an invalid protocol. The
—	supported protocols are %s. Please update the configuration to correct the
—	callback url.
60003	The configuration specifies an unknown callback include (%s). Valid callback
—	includes are %s. Please limit the callback include configuration to items from
—	the supported list.
60004	Invalid attachment type: %s
60005	Please provide callback url for type: %s
60006	Invalid callback url: %s
60007	The configuration specifies a callback with an unknown callback type (%s).
—	Valid callback types are %s. Please limit the callback type configuration to
—	items in the supported list.
60008	The callback configuration contains an property '%s' is not required for
—	callback of type '%s'. Please remove it from the configuration.
60009	The callback configuration specifies a callback type (%s), but is missing the
—	property '%s', which is required for callback of type '%s'. Please specify a
—	value.
60010	The callback configuration specifies the callback type (%s) with an invalid
—	form (%s). Please specify a format from the list of formats for this type
—	supported by VoiceBase (%s)
60011	The callback configuration specifies the http method '%s', which is not
—	supported by VoiceBase. Please update the configuration to specify an http
—	method from one of 'POST' or 'PUT'
60012	The callback configuration includes an S3 pre-signed url, but the content type
—	has not been specified. Please update the configuration to specify '%s'
60013	The callback configuration includes an S3 pre-signed url that has the content
—	type set to '%s' and it should be '%s'. Please update the configuration
—	specify the correct content type.

Continued on next page

Table 1 – continued from previous page

Code	Message
70000	The request for an API key does not include any configuration elements. Please
—	update the key to include the configuration.
70001	The request for an API key specifies both expiration (expirationDate) and time
—	to live in milliseconds (ttlMillis). Please update the request to include
—	expirationDate or ttlMillis, not both.
70002	The request for an API key specifies an expiration (expirationDate) that is
—	before now. Please update the request to include an expirationDate value that
—	is in the future, but less than 2 hours from now.
70003	The request for an API key specifies an expiration (expirationDate) that is
—	after 2 hours now. Please update the request to include an expirationDate
—	value that is less than 2 hours from now.
70004	The request for an API key specifies a time to live in milliseconds (ttlMillis)
—	that is less than or equal to zero. Please update the request to include a
—	ttlMillis greater than zero.
70005	The request for an API key specifies a time to live in milliseconds (ttlMillis)
—	that is greater than 2 hours. Please update the request to include a ttlMillis
—	that less than or equals to 2 hours.
70006	The request for an API key specifies that the key should be ephemeral, but does
—	not include an expiration date (expirationDate) nor a time to live in
—	milliseconds (ttlMillis). Please update the request to include one or the
—	other.
70007	VoiceBase could not find the API key specified in the request. Please update
—	the request to provide a known API key identifier.
70008	VoiceBase could not create the api key because we found the following error in
—	the scope configuration: %s. Please update the requeset to correct the issues.
80000	The request specifies a definition (%s) in the collection (%s). Please review
—	your request to identify the definition.
80100	The request is missing or has an empty keyword group (keywordGroup) element.
—	Please review the request body and submit a keyword group object.

Continued on next page

Table 1 – continued from previous page

Code	Message
80101	The request specifies a keyword group, but it contains no keywords. Please
—	update the request to specify keywords for the group.
80102	Keyword group name must be equals to '%s'
80103	The keyword group submitted in the request contains an invalid keyword (%s)
—	because it %s. Please correct the error.
80200	The request is missing or has an empty vocabulary element. Please review the
—	request body and submit a vocabulary object.
80202	The request specifies a vocabulary name in the path (%s) that is not equal to
—	the vocabulary name in the body (%s). They must be the same. Please update
—	either the path variable or the body property.
80203	The vocabulary submitted in the request contains an invalid term '%s' with the
—	following errors, %s. Please correct the errors.
80204	The vocabulary submitted contains both terms and scripts, but only one or the
—	other is allowed. Please correct the errors.
80205	The Vocabulary submitted does not contain a vocabulary type field. Please add
—	vocabularyType: 'scripts' to the request.
80206	The request specifies a vocabulary collection, but it contains neither terms
—	nor scripts. Please update the request to specify terms or scripts for the
—	vocabulary.
80207	The vocabulary submitted in the request contains an invalid script '%s' with
—	the following errors, %s. Please correct the errors.
80208	The vocabulary submitted in the request contains too many terms (%s). The
—	maximum is %s. Please limit the vocabulary to the maximum.
80209	The vocabulary submitted contains duplicate terms (%s). Please remove the
—	duplicated terms.
80300	The request is missing or has an empty searchableFields element. Please review
—	the request body and submit a searchableFields object.
80301	The request specifies a searchable field name (%s) that is invalid because %s.
—	Please correct the field to address the issue.
80302	The request specifies a searchable field expression (%s) that is invalid
—	because %s. Please correct the expression to address the issue.

Continued on next page

Table 1 – continued from previous page

Code	Message
80303	The request contains a searchable field (%s) whose length exceeds %s the
—	maximum length. Please remove the field from the request.
90000	We encountered an issue interacting with S3 (%s)
90001	The data in the collection (%s) for the organization (%s) was not found.
90002	Mongo failed to process request
90003	Internal server I/O error
90004	Message broker failed to process request
90005	Data version transformation process failed

Note: The 's%' in the messages are substitution variables that are filled when an actual error message is generated.

CHAPTER 41

Apache License, Version 2.0

This documentation is provided under an Apache License, Version 2.0.

Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

   "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

   "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

   "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

   "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attri-

bution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "{}" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright 2017 VoiceBase, Inc.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.