
Amazon Transcribe

Developer Guide



Amazon Transcribe: Developer Guide

Copyright © Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

What is Amazon Transcribe?	1
Amazon Transcribe features	1
Cross-service applications	2
How it works	3
Speech input	5
Containers and formats for batch transcription	5
Audio containers and formats for streaming transcription	6
Transcribing digits	6
Rules for transcribing numbers in English	6
Rules for transcribing numbers in German	8
Custom vocabularies	8
Acronyms	9
Create a custom vocabulary using a table	9
Create a custom vocabulary using a list	11
Character sets for custom vocabularies	11
Alternative transcriptions	45
Job queuing	47
IAM policies for job queuing	48
Tagging resources	50
Tag-based access control	51
Tagging your Amazon Transcribe resources	51
Guidelines and quotas	54
Supported Regions	54
Guidelines	54
Quotas	54
Getting started	57
Step 1: Set up an account	57
Sign up for AWS	57
Create an IAM user	57
Next step	58
Step 2: Set up the AWS CLI	58
Next step	59
Step 3: Getting started using the console	59
Create a transcription job	59
View a transcription job	60
Step 4: Getting started using the API	62
Getting started (AWS CLI)	62
Getting started (SDK for Python)	64
Step 5: Getting started with streaming audio	66
Streaming transcription	70
Event stream encoding	74
Using WebSocket streaming	76
Adding a policy for WebSocket requests to your IAM role	76
Creating a pre-signed URL	77
Handling the WebSocket upgrade response	81
Making a WebSocket streaming request	81
Handling a WebSocket streaming response	82
Handling WebSocket streaming errors	82
Using HTTP/2 streaming	83
Streaming request	83
Streaming response	86
Example request and response	87
HTTP/2 retry client	89
Using the HTTP/2 retry client	94

Partial result stabilization	95
Using partial result stabilization in an HTTP/2 request	96
Using partial result stabilization with WebSocket streams	97
Streaming transcription output	97
Redaction	100
Redacting or identifying PII in an audio file	101
Redacting or identifying PII in a real-time stream	104
Example PII redaction and identification output	106
Example redacted batch output	106
Example redacted streaming output	107
Example PII identification output	108
Custom language models	110
Step 1: Preparing the data	111
Step 2: Providing Amazon Transcribe with data permissions	112
Step 3: Creating a custom language model	114
Using S3 Prefixes to Retrieve Your Data	114
Step 4: Transcribing with a custom language model	116
Step 5: Viewing and updating your custom language models	118
Call analytics	119
Using categories and rules	120
Setting up call analytics transcription jobs	122
Sample call analytics output	127
Language identification	132
Getting notifications using Automatic language identification	134
Vocabulary filtering	136
Step 1: Creating a list of unwanted words	136
Step 2: Creating a vocabulary filter	137
Step 3: Filtering transcriptions	138
Filtering batch transcriptions	139
Filtering streaming transcriptions	143
Speaker diarization	147
Identifying speakers in audio files	147
Identifying speakers in real-time streams	150
HTTP/2 streaming	151
WebSocket streaming	151
Streaming transcription output	152
Channel identification	155
Transcribing multi-channel audio files	155
Transcribing multi-channel audio streams	159
Transcribing multi-channel audio in an HTTP/2 stream	159
Transcribing multi-channel audio in a WebSocket stream	160
Multi-channel streaming output	160
Security	162
Data protection	162
Encryption at rest	163
Encryption in transit	163
Key management	163
Opting out of using your data for service improvement	165
VPC endpoints (AWS PrivateLink)	166
Identity and access management	168
Audience	168
Authenticating with identities	168
Managing access using policies	170
How Amazon Transcribe works with IAM	172
Identity-based policy examples	175
Troubleshooting	179
Monitoring Amazon Transcribe	181

Monitoring Amazon Transcribe with CloudTrail	181
Using Amazon EventBridge with Amazon Transcribe	183
CloudWatch metrics	186
Compliance validation	186
Resilience	187
Infrastructure security	187
Amazon Transcribe Medical	188
What is Amazon Transcribe Medical?	188
Streaming audio transcription	189
Batch transcription	189
How it works	189
Speech input	189
Streaming transcription overview	190
Batch transcription overview	190
Transcribing numbers	192
Transcribing medical terms and measurements	193
Getting started	194
Set up an account	194
Getting started with console streaming	196
Getting started with batch transcription	196
Streaming transcription	196
Event stream encoding	201
Using HTTP/2 streaming	203
Using WebSocket streaming	215
Transcribing a medical conversation	222
Transcribing an audio file	223
Transcribing a real-time stream	226
Identifying speakers	228
Transcribing multi-channel audio	235
Transcribing a medical dictation	241
Transcribing an audio file	241
Transcribing a streaming medical dictation	245
Creating and using medical custom vocabularies	247
Creating a text file for your medical custom vocabulary	247
Using a text file to create a medical custom vocabulary	250
Transcribing an audio file using a medical custom vocabulary	252
Transcribing a real-time stream using a medical custom vocabulary	253
Character sets for Amazon Transcribe Medical	255
Identifying PHI in a transcript	256
Identifying PHI in an audio file	257
Identifying PHI in a real-time stream	260
Generating alternative transcriptions	262
Guidelines and quotas	265
Supported Regions	54
Guidelines	265
Quotas	265
VPC endpoints (AWS PrivateLink)	266
Considerations for Amazon Transcribe Medical VPC endpoints	267
Creating an interface VPC endpoint for Amazon Transcribe Medical	267
Creating a VPC endpoint policy for Amazon Transcribe Medical streaming	267
Document history	269
API Reference	275
Actions	275
Amazon Transcribe Service	276
Amazon Transcribe Streaming Service	401
Data Types	414
Amazon Transcribe Service	415

Amazon Transcribe Streaming Service	468
Common Errors	491
Common Parameters	492
AWS glossary	495

What is Amazon Transcribe?

Amazon Transcribe uses machine learning to recognize speech in audio and video files and transcribe that speech into text. Practical use cases for Amazon Transcribe include transcriptions of customer-agent calls and closed captions for videos.

Amazon Transcribe features

The following list highlights the Amazon Transcribe features that are available with **all supported languages**. There are several features that are only supported with specific languages; refer to [Supported languages and language-specific features \(p. 3\)](#) for more information.

- **Custom vocabularies (p. 8):** Use a list of specific words you want Amazon Transcribe to recognize in your audio input. Custom vocabularies are often used for domain-specific terms or proper nouns that Amazon Transcribe isn't rendering correctly in your transcription output.

Use custom vocabularies to:

- Recognize industry-specific terms
- Display acronyms correctly
- Improve the accuracy of your transcription output

See also: [Custom language models \(p. 110\)](#)

- **Language identification (p. 132):** Amazon Transcribe can automatically identify the predominant language in a media file without you having to specify a language code. You can also select several language codes to help Amazon Transcribe narrow down the predominant language and improve its accuracy. Automatic language identification can be used for any **batch transcription** with the API or the [Amazon Transcribe console](#).

Amazon Transcribe also has the ability to transcribe accented speech of individuals who are non-native speakers of a language. For example, you can transcribe US English (en-US) audio spoken with a German (de-DE) accent.

- **Vocabulary filtering (p. 136):** Mask, remove, or tag words you don't want to appear in your transcription. Vocabulary filtering helps you filter for any word you consider profane, obscene, offensive, or otherwise unsuitable for display in your transcripts.

Use vocabulary filtering to:

- Generate family-friendly captions of a TV show
- Remove proprietary terms from transcripts of conference proceedings
- **Speaker diarization (p. 147):** Identify individual speakers in an audio clip—a technique known as speaker diarization. When you activate speaker diarization, Amazon Transcribe includes an attribute that identifies each speaker in the audio clip.

Use speaker diarization to:

- Identify the customer and the support representative in a recorded customer support call
- Identify characters for closed captions
- Identify the speaker and questioners in a recorded press conference or lecture
- **Channel identification (p. 155):** Create a transcript for each audio channel or single stream of recorded sound in an audio file. For example, a phone conversation between two people consists of two separate audio channels. With channel identification, Amazon Transcribe returns two or more

transcriptions: a combined transcription of all of the audio channels, and a transcription of each audio channel.

Important

Not all Amazon Transcribe features are available in all languages; please review the [Supported languages and language-specific features \(p. 3\)](#) table before getting started.

Cross-service applications

You can use Amazon Transcribe with other AWS services to create applications. For example, you can:

- Translate your audio into another language. Use Amazon Transcribe to convert voice to text, Amazon Translate to translate your text into another language, and Amazon Polly to generate audio from the translated text.
- Use Amazon Transcribe to transcribe recordings of customer service calls for analysis. After transcribing a recording, send the transcription to Amazon Comprehend to identify keywords, topics, or sentiments.
- Use Amazon Transcribe to transcribe live broadcasts, such as television, to provide real-time subtitles. Amazon Transcribe might require additional customization—such as using a custom language model or manual transcript correction—for broadcast-grade applications.

How Amazon Transcribe works

Amazon Transcribe analyzes audio files that contain speech and uses advanced machine learning techniques to transcribe the voice data into text. You can then use the transcription as you would any text document.

Each language has its own *language code* that you can use to specify the language of your audio or video file.

You can transcribe an uploaded file or live-stream audio. Processing an uploaded file is referred to as a **batch transcription job**, while using live audio is referred to as a **streaming transcription**. For information on supported file containers and formats for both batch and streaming options, see [Speech input \(p. 5\)](#).

To transcribe an audio file, Amazon Transcribe uses three APIs:

- [StartTranscriptionJob \(p. 376\)](#) – Starts a batch job to transcribe the speech in an audio file to text.
- [ListTranscriptionJobs \(p. 356\)](#) – Returns a list of transcription jobs that have been started. You can specify the status of the jobs that you want the API to return. For example, you can get a list of all pending jobs, or a list of completed jobs.
- [GetTranscriptionJob \(p. 329\)](#) – Returns the result of a transcription job. The response contains a link to a JSON file containing the results.

To transcribe streaming audio to text, Amazon Transcribe uses one API:

- [StartStreamTranscription \(p. 408\)](#) – Starts a bi-directional HTTP/2 stream where audio is streamed to Amazon Transcribe and the transcription results are streamed to your application.

You can also start a WebSocket protocol stream to send audio to the Amazon Transcribe. For more information, see [Using Amazon Transcribe streaming with WebSockets \(p. 76\)](#).

You can use Amazon Transcribe to create and manage custom vocabularies for your solution. A custom vocabulary gives Amazon Transcribe more information about how to process speech in an audio clip.

- [CreateVocabulary \(p. 291\)](#) – Creates a custom vocabulary that you can use in your transcription jobs.
- [DeleteVocabulary \(p. 311\)](#) – Deletes a custom vocabulary from your account.
- [GetVocabulary \(p. 332\)](#) – Gets information about a custom vocabulary and a URL that you can use to download the contents of a vocabulary.
- [ListVocabularies \(p. 359\)](#) – Gets a list of custom vocabularies in your account.
- [UpdateVocabulary \(p. 395\)](#) – Updates an existing vocabulary.

Supported languages and language-specific features

Language	Language Code	Speech input (p. 5)	Transcribing digits (p. 6)	Acronyms (p. 12)	Custom language models (p. 12)	Redaction (p. 12)	Call analytics (p. 119)
Afrikaans (p. 12)	af-ZA	batch only	no	yes	no	no	no
Arabic (p. 13)	ar-AE Gulf	batch only	no	no	no	no	yes

Language	Language Code	Speech input (p. 5)	Transcribing digits (p. 6)	Acronyms (p. 6)	Custom language models (p. 11)	Redaction (p. 11)	Call analytics (p. 119)
Arabic (p. 13) Modern Standard	ar-SA	batch only	no	no	no	no	no
Chinese, Simplified (p. 15)	zh-CN	batch, streaming	no	no	no	no	yes
Chinese, Traditional (p. 16)	zh-TW	batch only	no	no	no	no	no
Danish (p. 16)	da-DK	batch only	no	yes	no	no	no
Dutch (p. 17)	nl-NL	batch only	no	yes	no	no	no
English (p. 19) Australian	en-AU	batch, streaming	yes	yes	yes	no	yes
English (p. 19) British	en-GB	batch, streaming	yes	yes	yes	no	yes
English (p. 19) Indian	en-IN	batch only	yes	yes	no	no	yes
English (p. 19) Irish	en-IE	batch only	yes	yes	no	no	yes
English (p. 19) New Zealand	en-NZ	batch only	yes	yes	no	no	no
English (p. 19) Scottish	en-AB	batch only	yes	yes	no	no	yes
English (p. 19) South African	en-ZA	batch only	yes	yes	no	no	no
English (p. 19) US	en-US	batch, streaming	yes	yes	yes	yes	yes
English (p. 19) Welsh	en-WL	batch only	yes	yes	no	no	yes
French (p. 21)	fr-FR	batch, streaming	no	yes	no	no	yes
French (p. 21) Canadian	fr-CA	batch, streaming	no	yes	no	no	yes
Farsi (p. 20)	fa-IR	batch only	no	no	no	no	no
German (p. 21)	de-DE	batch, streaming	yes	yes	no	no	yes
German (p. 21) Swiss	de-CH	batch only	yes	yes	no	no	yes

Language	Language Code	Speech input (p. 5)	Transcribing digits (p. 6)	Acronyms (p. 1)	Custom language models (p. 1)	Redaction (p. 1)	Call analytics (p. 119)
Hebrew (p. 24)	he-IL	batch only	no	no	no	no	no
Hindi (p. 25)	hi-IN	batch only	no	yes	yes	no	yes
Indonesian (p. 26)	id-ID	batch only	no	yes	no	no	no
Italian (p. 29)	it-IT	batch, streaming	no	yes	no	no	yes
Japanese (p. 30)	ja-JP	batch, streaming	no	no	no	no	yes
Korean (p. 31)	ko-KR	batch, streaming	no	no	no	no	yes
Malay (p. 32)	ms-MY	batch only	no	yes	no	no	no
Portuguese (p. 33)	pt-BR	batch only	no	yes	no	no	yes
Portuguese (p. 34)	pt-BR	batch, streaming	no	yes	no	no	yes
Russian (p. 35)	ru-RU	batch only	no	no	no	no	no
Spanish (p. 36)	es-ES	batch only	no	yes	no	no	yes
Spanish (p. 37)	es-US	batch, streaming	no	yes	yes	no	yes
Tamil (p. 37)	ta-IN	batch only	no	no	no	no	no
Telugu (p. 39)	te-IN	batch only	no	no	no	no	no
Thai (p. 41)	th-TH	batch only	no	yes	no	no	no
Turkish (p. 44)	tr-TR	batch only	no	yes	no	no	no

Speech input

Amazon Transcribe can transcribe speech as either a media file or a real-time stream. Your input audio must use the encodings and formats described in the following sections. For a list of supported languages, see [Supported languages and language-specific features \(p. 3\)](#).

Containers and formats for batch transcription

When you transcribe an audio file or video file using the [StartTranscriptionJob \(p. 376\)](#) API or the [Amazon Transcribe console](#), make sure that the file is:

- In FLAC, MP3, MP4, Ogg, WebM, AMR, or WAV file format
- Less than 4 hours in length and less than 2 GB in size (500 MB for call analytics jobs)

Note

For AMR, Amazon Transcribe supports both Adaptive Multi-Rate Wideband (AMR-WB) and Adaptive Multi-Rate Narrowband (AMR-NB) codecs.

For the Ogg and WebM file formats, Amazon Transcribe supports the Opus codec.

For best results:

- Use a lossless format. You can choose either FLAC, or WAV with PCM 16-bit encoding.
- Use a sample rate of 8,000 Hz for telephone audio.

Audio containers and formats for streaming transcription

When you transcribe a real-time stream using the [StartStreamTranscription \(p. 408\)](#) API or a WebSocket request, make sure that your stream is encoded in:

- PCM 16-bit signed little endian
- FLAC
- OPUS encoded audio in the Ogg container

For best results:

- Use a lossless format, such as FLAC or PCM encoding.
- Use a sample rate of 8,000 Hz for telephone audio.

For more information on using a WebSocket request to transcribe your streaming audio, see [Using Amazon Transcribe streaming with WebSockets \(p. 76\)](#).

For a list of supported languages, see [Supported languages and language-specific features \(p. 3\)](#).

Transcribing numbers and punctuation

Amazon Transcribe automatically adds punctuation to all the languages that it supports. It also capitalizes words appropriately for languages that use case distinction in their writing systems. For example, the spoken number "one thousand two hundred forty-two" is transcribed as "1242" in supported languages. For all other languages, numbers are transcribed into their word forms.

For a list of supported languages, see [Supported languages and language-specific features \(p. 3\)](#).

Rules for transcribing numbers in English

For all English languages, such British English (en-GB) or US English (en-US), numbers are transcribed according to the following rules.

Rule	Example
Convert cardinal numbers greater than 10 to numbers.	<ul style="list-style-type: none">• "Fifty five" > 55• "a hundred" > 100• "One thousand and thirty one" > 1031

Rule	Example
	<ul style="list-style-type: none"> "One hundred twenty-three million four hundred fifty six thousand seven hundred eight nine" > 123,456,789
Convert cardinal numbers followed by "million" or "billion" to numerals followed by a word when "million" or "billion" is not followed by a number.	<ul style="list-style-type: none"> "one hundred million" > 100 million "one billion" > 1 billion "two point three million" > 2.3 million
Convert ordinal numbers greater than 10 to numbers.	<ul style="list-style-type: none"> "Forty third" > 43rd "twenty sixth avenue" > 26th avenue
Convert fractions to their numeric format.	<ul style="list-style-type: none"> "a quarter" > 1/4 "three sixteenths" > 3/16 "a half" > 1/2 "a hundredth" > 1/100
Convert numbers less than 10 to digits if there are more than one in a row.	<ul style="list-style-type: none"> "three four five" > 345 "My phone number is four two five five five five one two one two" > 4255551212
Decimals are indicated by "dot" or "point."	<ul style="list-style-type: none"> "three hundred and three dot five" > 303.5 "three point twenty three" > 3.23 "zero point four" > 0.4 "point three" > 0.3
Convert the word "percent" after a number to a percent symbol (%).	<ul style="list-style-type: none"> "twenty three percent" > 23% "twenty three point four five percent" > 23.45%
Convert the words "dollar," "US dollar," "Australian dollar," "AUD," or "USD" after a number to a dollar sign (\$) before the number.	<ul style="list-style-type: none"> "one dollar and fifteen cents" > \$1.15 "twenty three USD" > \$23 "twenty three Australian dollars" > \$23
Convert the words "pounds," "British pounds," or "GDB" after a number to pound sign (£) before the number.	<ul style="list-style-type: none"> "twenty three pounds" > £23 "I have two thousand pounds" > I have £2,000 "five pounds thirty three pence" > £5.33
Convert the words "rupees," "Indian rupees," or "INR" after a number to rupee sign (₹) before the number.	<ul style="list-style-type: none"> "twenty three rupees" > ₹23 "fifty rupees thirty paise" > ₹50.30
Convert times to numbers.	<ul style="list-style-type: none"> "seven a m eastern standard time" > 7 a.m. eastern standard time "twelve thirty p m" > 12:30 p.m.
Combine years expressed as two digits into four. Only valid for the 20th, 21st, and 22nd centuries.	<ul style="list-style-type: none"> "nineteen sixty two" > 1962 "the year is twenty twelve" > the year is 2012 "twenty nineteen" > 2019 "twenty one thirty" > 2130
Convert dates to numbers.	<ul style="list-style-type: none"> "May fifth twenty twelve" > May 5th 2012 "May five twenty twelve" > May 5 2012 "five May twenty twelve" > 5 May 2012

Rule	Example
Separate spans of numbers by the word "to."	<ul style="list-style-type: none"> "twenty three to thirty seven" > 23 to 37

Rules for transcribing numbers in German

For German (de-DE) and Swiss German (de-CH), numbers are transcribed according to the following rules.

Rule	Example
Convert cardinal numbers greater than 10 to numbers.	<ul style="list-style-type: none"> "fünfundfünfzig" > 55 "vier tausend sechs hundert einundachtzig" > 4681 "eine Sache" > "eine Sache"
Convert cardinal numbers followed by "million" or "billion" to numerals followed by a word when "million" or "billion" is not followed by a number.	<ul style="list-style-type: none"> "zehn Millionen Menschen" > 10 Millionen Menschen "zehn Millionen fünf hundert tausend" > 10.500.000
Convert ordinal numbers greater than 10 to numbers.	<ul style="list-style-type: none"> "dreiundzwanzigste" > 23. "vierzigster" > 40. "ich war Erster" > "ich war Erster"
Fractions are not converted into a numeric format.	<ul style="list-style-type: none"> "ein Drittel" > "ein Drittel"
Convert numbers less than 10 to digits if there are more than one in a row.	<ul style="list-style-type: none"> "eins zwei drei" > 123 "plus vier neun zwei vier eins" > +49241
Decimals are indicated by ",".	<ul style="list-style-type: none"> "zweiundzwanzig komma drei" > 22,3
Convert the word "percent" after a number to a percent symbol (%).	<ul style="list-style-type: none"> "fünf Prozent Hürde" > 5 % Hürde "dreiundzwanzig komma vier Prozent" > 23,4%
Convert the words "Euro" to a euro sign.	<ul style="list-style-type: none"> "ein euro" > 1 € "ein Euro vierzig" > 1,40 € "ein Euro vierzig Cent" > 1,40 €
Convert times to numbers.	<ul style="list-style-type: none"> "vierzehn Uhr fünfzehn" > 14:15 Uhr
Convert dates to numbers.	<ul style="list-style-type: none"> "dritter Dezember neunzehn hundert sechundfünfzig" > 3. Dezember 1956
Display slashes and dashes.	<ul style="list-style-type: none"> "55 Schrägstrich 13" > 55/13 "55 Strich 13" > 55-13
Display numbered paragraphs	<ul style="list-style-type: none"> "Paragraf 17" > § 17

Custom vocabularies

You can give Amazon Transcribe more information about how to process speech in your input file by creating a custom vocabulary in text file format. A *custom vocabulary* is a list of specific words that you

want Amazon Transcribe to recognize in your audio input. These are generally domain-specific words and phrases, words that Amazon Transcribe isn't recognizing, or proper nouns.

Important

You are responsible for the integrity of your own data when you use Amazon Transcribe. Do not enter confidential information, personal information (PII), or protected health information (PHI) into a custom vocabulary.

Custom vocabularies work best when used to target specific words or phrases. We recommend you create separate small vocabularies tailored to specific audio recordings instead of creating a single, large vocabulary for use with all of your recordings.

Custom vocabulary basics:

- You can have up to 100 vocabularies (as separate text files) in your account.
- The size limit for each custom vocabulary file is 50 Kb.
- Each vocabulary file can be in either table or list format; table format is strongly recommended.
- Your vocabulary files must be stored in an S3 bucket if using a table format. If using a list, you can upload a text file using the console or include your list of words within an API/CLI call.
- Each entry must contain fewer than 256 characters, including hyphens.
- Only use characters from the allowed character set for your language (see [Character sets for custom vocabularies \(p. 11\)](#)).
- If an entry in the list is a phrase, you must separate the words of the phrase with a hyphen. For example, if the phrase is **Los Angeles**, you would enter it in the file as **Los-Angeles**.

Why use a table instead of a list for your custom vocabulary?

The table format gives you more options for—and more control over—the input and output of words within your custom vocabulary. The console, API, and CLI all use custom vocabulary tables in the same way; lists are used differently for each method and may require additional formatting-related steps for successful use between the console, API, and CLI.

To create a custom vocabulary, use the [CreateVocabulary \(p. 291\)](#) API or the [Amazon Transcribe console](#). After you submit the [CreateVocabulary \(p. 291\)](#) request, Amazon Transcribe processes the vocabulary. To see the processing status of the vocabulary, use the console or the [GetVocabulary \(p. 332\)](#) API.

To use the custom vocabulary, set the `VocabularyName` field of the `Settings` field when you call the [StartTranscriptionJob \(p. 376\)](#) API or choose the vocabulary in the console when you create the transcription job.

Using acronyms in your custom vocabulary

Enter acronyms, or other words whose letters should be pronounced individually, as single letters separated by periods; for example: **A.B.C.**, **F.B.I.**, **A.W.S.**. To enter the plural form of an acronym, such as "ABCs", separate the "s" from the acronym with a hyphen: **A.B.C.-s**. You can use upper or lower case letters to define an acronym. Acronyms are not supported in all languages; refer to [Supported languages and language-specific features \(p. 3\)](#).

Create a custom vocabulary using a table

You can create a custom vocabulary by creating a four-column table in a text file with the following headers:

- **Phrase** – The word or phrase that should be recognized.

If the entry is a phrase, separate the words with a hyphen (-). For example, you type **Los Angeles** as **Los-Angeles**.

Enter acronyms or other words whose letters should be pronounced individually as single letters followed by dots, such **A.B.C.** or **F.B.I.**. To enter the plural form of an acronym, such as "ABCs," separate the "s" from the acronym with a hyphen: "**A.B.C.-s**". You can use either upper- or lower-case letters to enter an acronym. For a list of languages that support acronyms, see [Supported languages and language-specific features \(p. 3\)](#).

The **Phrase** field is required. You can use any of the allowed characters for the input language. For the list of allowed characters, see the individual languages. If you do not specify the **DisplayAs** field, Amazon Transcribe uses the contents of the **Phrase** field in the output file.

- **IPA** – The pronunciation of your word or phrase using IPA characters.

You can include characters in the [International Phonetic Alphabet \(IPA\)](#) in this field. The **IPA** field can't contain leading or trailing spaces, and you must use a single space to separate each phoneme in the input. For example, in English, you would enter the phrase **Los-Angeles** as **l # s æ n # # l # s** and **F.B.I.** as **# f b i a#**.

For a list of allowed IPA characters for a specific language, refer to [Character sets for custom vocabularies \(p. 11\)](#).

- **SoundsLike** – The pronunciation of your word or phrase using the standard orthography of the language to mimic the way that the word sounds.

You can break a word or phrase down into smaller pieces (typically based on syllables) and provide a pronunciation for each piece based on how that piece sounds. For example, in English, you can provide pronunciation hints for the phrase **Los-Angeles** as **loss-ann-gel-es**. The hint for the word **Etienne** would look like this: **eh-tee-en**. You separate each part of the hint with a hyphen (-).

- **DisplayAs** – Defines the how the word or phrase looks when it's output. For example, if the word or phrase is **Los-Angeles**, you can specify the display form as "Los Angeles" so that the hyphen is not present in the output.

If you don't specify the **DisplayAs** field, Amazon Transcribe uses the **Phrase** field from the input file in the output.

You can use any UTF-8 character in the **DisplayAs** field.

Place each word or phrase in your text file on a separate row, separating each field with a TAB character. Only use spaces *within* the **IPA** and **DisplayAs** columns.

A basic custom vocabulary may look something like the following (where **[TAB]** represents a TAB character):

```
Phrase[TAB]IPA[TAB]SoundsLike[TAB]DisplayAs
Los-Angeles[TAB][TAB][TAB]Los Angeles
F.B.I.[TAB]# f b i a#[TAB][TAB]FBI
Etienne[TAB][TAB]eh-tee-en[TAB]
```

Important

In a given row, you cannot have content for both **IPA** and **SoundsLike** fields. You must choose one or the other, or leave both fields blank; **DisplayAs** is the only required field.

Columns can be entered in any order, as shown in the following examples. Note that these examples use spaces to align the columns for visual clarity; however, your input files **must only** use TAB characters to separate column entries. If you copy these examples, remove the extra spaces between columns and replace **[TAB]** with a TAB character.

```
Phrase      [TAB]SoundsLike [TAB]IPA          [TAB]DisplayAs
Los-Angeles[TAB]           [TAB]                  [TAB]Los Angeles
F.B.I       [TAB]           [TAB]# f b i a# [TAB]FBI
Etienne    [TAB]eh-tee-en [TAB]                 [TAB]
```

```
DisplayAs  [TAB]SoundsLike [TAB]IPA          [TAB]Phrase
Los Angeles[TAB]           [TAB]                  [TAB]Los-Angeles
FBI        [TAB]           [TAB]# f b i a# [TAB]F.B.I.
[TAB]eh-tee-en [TAB]                 [TAB]Etienne
```

Save your custom vocabulary file with the extension .txt in an S3 bucket in the same region where you are calling the API.

Tip

Make sure your text file is in LF format. If you use any other format, such as CRLF, your custom vocabulary is not accepted by Amazon Transcribe.

Create a custom vocabulary using a list

You can create a custom vocabulary using a list of words or phrases in a text file. You can place each word on its own line, or you can put multiple words on a single line, separating the words or phrases from each other with a comma.

The list format can be uploaded as a text file when using the [Amazon Transcribe console](#). If using a list with the API or CLI, you must include the list of words within the API/CLI call.

Character sets for custom vocabularies

Amazon Transcribe limits the characters that you can use to create custom vocabularies. You can use the following character sets for each language.

Important

Be sure to check that your custom vocabulary file uses only the supported Unicode code points and code point sequences listed within the following character sets.

Many Unicode characters can appear identical in popular fonts, even if they use different code points. **Only the code points listed in this guide are supported.** For example, the French word **déjà** can be rendered using *precomposed* characters (where one Unicode value represents an accented character) or *decomposed* characters (where two Unicode values represent an accented character, one value for the base character and another for the accent).

- **Precomposed version:** 0064 00E9 006A 00E0 (renders as **déjà**)
- **Decomposed version:** 0064 0065 0301 006A 0061 0300 (renders as **déjà**)

Topics

- [Afrikaans character set \(p. 12\)](#)
- [Arabic character set \(p. 13\)](#)
- [Chinese, Mandarin \(Mainland China\), Simplified character set \(p. 15\)](#)
- [Chinese, Mandarin \(Taiwan\), Traditional character set \(p. 16\)](#)
- [Danish character set \(p. 16\)](#)
- [Dutch character set \(p. 17\)](#)
- [English character set \(p. 19\)](#)
- [Farsi character set \(p. 20\)](#)
- [French character set \(p. 21\)](#)

- [German character set \(p. 23\)](#)
- [Hebrew character set \(p. 24\)](#)
- [Hindi character set \(p. 25\)](#)
- [Indonesian character set \(p. 28\)](#)
- [Italian character set \(p. 29\)](#)
- [Japanese character set \(p. 30\)](#)
- [Korean character set \(p. 31\)](#)
- [Malay character set \(p. 32\)](#)
- [Portuguese character set \(p. 33\)](#)
- [Russian character set \(p. 35\)](#)
- [Spanish character set \(p. 36\)](#)
- [Tamil character set \(p. 37\)](#)
- [Telugu character set \(p. 39\)](#)
- [Thai character set \(p. 41\)](#)
- [Turkish character set \(p. 44\)](#)

Afrikaans character set

For Afrikaans custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- - (hyphen)
- . (period)

You can also use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
á	00E1	í	00EF
è	00E8	ó	00F3
é	00E9	ô	00F4
ê	00EA	ö	00F6
ë	00EB	ú	00FA
í	00ED	û	00FB
î	00EE	ü	00FC

You can use the following International Phonetic Alphabet characters in the `IPA` field of the vocabulary input file:

Character	Code	Character	Code
a	0061	z	007A
b	0062	æ	00E6

Character	Code	Character	Code
c	0063	ð	00F0
d	0064	ø	00F8
e	0065	ŋ	014B
e~	0065 0303	œ	0153
f	0066	æ	0250
g	0067	ɑ	0251
h	0068	ã	0251 0303
i	0069	ɒ	0252
j	006A	ɔ	0254
k	006B	ɔ̄	0254 0303
l	006C	ə	0259
m	006D	ɛ	025B
n	006E	Ξ	025B 0303
o	006F	ɜ	025C
o~	006F 0303	ɪ	026A
p	0070	‡	026C
r	0072	ɹ	0279
s	0073	ݕ	0281
t	0074	݂	0283
u	0075	ݔ	028A
v	0076	݈	028C
w	0077	ݖ	0292
x	0078	݇	03B8
y	0079		

Arabic character set

For Arabic custom vocabularies, you can use the following Unicode characters in the `Phrase` and `SoundsLike` fields. You can also use the hyphen (-) character to separate words.

Character	Code	Character	Code
݂	0621	ݔ	0633
݇	0622	ݖ	0634

Character	Code	Character	Code
أ	0623	ص	0635
ڦ	0624	ڙ	0636
ؠ	0625	ٻ	0637
ڻ	0626	ڻ	0638
ؠ	0627	ڦ	0639
ٻ	0628	ڦ	063A
ڮ	0629	ڦ	0641
ڻ	062A	ڦ	0642
ڻ	062B	ڦ	0643
ڻ	062C	ڦ	0644
ڻ	062D	ڦ	0645
ڻ	062E	ڦ	0646
ڻ	062F	ڦ	0647
ڻ	0630	ڦ	0648
ڻ	0631	ڦ	0649
ڻ	0632	ڦ	064A

You can use the following International Phonetic Alphabet characters in the `IPA` field of the vocabulary input file:

Character	Code	Character	Code
a	0061	tʃ	0074 02E4
a:	0061 02D0	u	0075
b	0062	u:	0075 02D0
d	0064	v	0076
dʒ	0064 02E4	w	0077
f	0066	x	0078
h	0068	z	007A
i	0069	zʃ	007A 02E4
i:	0069 02D0	ð	00F0
j	006A	ðʒ	00F0 02E4
k	006B	ħ	0127

Character	Code	Character	Code
l	006C	ȝ	0263
m	006D	ȝ	026A
n	006E	ȝ	026B
p	0070	ȝ	0283
q	0071	ȝ	0292
r	0072	ȝ	0294
s	0073	ȝ	0295
s̄	0073 02E4	ȝ	03B8
t	0074	ȝ	03C7

Chinese, Mandarin (Mainland China), Simplified character set

For Chinese (Simplified) custom vocabularies, the `Phrase` field can use any of the characters listed in the following file on GitHub.

- [zh-cn-character-set.txt](#)

The `SoundsLike` field can contain the pinyin syllables listed in the following file on GitHub.

- [pinyin-set.txt](#)

When you use pinyin syllables in the `SoundsLike` field, separate the syllables with a hyphen (-).

Amazon Transcribe represents the four tones in Chinese (Simplified) using numbers. The following table shows how tone marks are mapped for the word 'ma'.

Tone	Tone mark	Tone number
Tone 1	mā	ma1
Tone 2	má	ma2
Tone 3	mǎ	ma3
Tone 4	mà	ma4

Note

For the 5th (neutral) tone, you can use Tone 1, with the exception of 'er', which must be mapped to Tone 2. For example, 打转儿 would be represented as 'da3-zhuan4-er2'.

Chinese (Simplified) custom vocabularies don't use the `IPA` field, but you must still include the `IPA` header in the vocabulary table.

The following example is an input file in text format. The example uses spaces to align the columns. Your input files should use TAB characters to separate the columns. Include spaces only in the `DisplayAs` column.

Phrase	SoundsLike	IPA	DisplayAs
##	kang1-jian4		
##	qian3-ze2		
####	guo2-fang2-da4-chen2		
#####	shi4-jie4-bo2-lan3-hui4	###	

Chinese, Mandarin (Taiwan), Traditional character set

For Chinese (Traditional) custom vocabularies, the `Phrase` field can use any of the characters listed in the following file on GitHub.

- [zh-tw-character-set.txt](#)

The `SoundsLike` field can contain the zhuyin syllables listed in the following file on GitHub.

- [zhuyin-set.txt](#)

When you use zhuyin syllables in the `SoundsLike` field, separate the syllables with a hyphen (-).

Amazon Transcribe represents the four tones in Chinese (Traditional) using numbers. The following table shows how tone marks are mapped for the word □ㄚ.

Tone	Tone mark
Tone 1	ㄇㄚ
Tone 2	ㄇㄚˊ
Tone 3	ㄇㄚˇ
Tone 4	ㄇㄚˋ

Chinese (Traditional) custom vocabularies don't use the `IPA` field, but you must still include the `IPA` header in the vocabulary table.

The following example is an input file in text format. The example uses spaces to align the columns. Your input files should use TAB characters to separate the columns. Include spaces only in the `DisplayAs` column.

Phrase	SoundsLike	IPA	DisplayAs
##	####-##		
##	###`-##		
####	####-###-###-##`		
#####	##-####-###-##`-###	##	

Danish character set

For Danish custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- - (hyphen)
- . (period)

You can also use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
Å	00C5	æ	00E6
Æ	00C6	é	00E9
Ø	00D8	ø	00F8
å	00E5		

You can use the following International Phonetic Alphabet characters in the `IPA` field of the vocabulary input file:

Character	Code	Character	Code
a	0061	æ	00E6
b	0062	ð	00F0
d	0064	ø	00F8
e	0065	ŋ	014B
f	0066	œ	0153
h	0068	ə	0250
i	0069	ɑ	0251
j	006A	ɒ	0252
k ^h	006B 02B0	ɔ	0254
l	006C	ç	0255
m	006D	ə	0259
n	006E	ɛ	025B
o	006F	g	0261
p ^h	0070 02B0	ɪ	026A
s	0073	ㅂ	0281
t ^s	0074 02E2	ㅂ	028A
u	0075	ઉ	028B
w	0077	ા	028C
y	0079		

Dutch character set

For Dutch custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- ' (apostrophe)
- - (hyphen)
- . (period)

You can also use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
à	00E0	î	00EE
á	00E1	ї	00EF
â	00E2	ñ	00F1
ä	00E4	ò	00F2
ç	00E7	ó	00F3
è	00E8	ô	00F4
é	00E9	ö	00F6
ê	00EA	ù	00F9
ë	00EB	ú	00FA
ì	00EC	û	00FB
í	00ED	ü	00FC

You can use the following International Phonetic Alphabet characters in the `IPA` field of the vocabulary input file:

Character	Code	Character	Code
a:	0061 003A	z	007A
b:	0062 02D0	ø:	00F8 003A
b	0062	ŋ	014B
d	0064	œy	0153 0079
e:	0065 02D0	œ:	0153 02D0
f	0066	ɑ	0251
g	0067	ɔ	0254
i	0069	ɔu	0254 0075
j	006A	ɔ:	0254 02D0
k	006B	ə	0259
l	006C	ɛ	025B

Character	Code	Character	Code
m	006D	ɛ:	025B 003A
n	006E	ɛi	025B 0069
o:	006F 02D0	ɦ	0266
p	0070	ɪ	026A
s	0073	ɲ	0272
t	0074	r	027E
u	0075	ʃ	0283
v	0076	ʏ	028F
w	0077	ʒ	0292
y	0079	χ	03C7

English character set

For English custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- ' (apostrophe)
- - (hyphen)
- . (period)

You can use the following International Phonetic Alphabet characters in the `IPA` field of the vocabulary input file:

Character	Code	Character	Code
aʊ	0061 028A	w	0077
aɪ	0061 026A	z	007A
b	0062	æ	00E6
d	0064	ð	00F0
eɪ	0065 026A	ŋ	014B
f	0066	ə	0251
g	0067	ɔ	0254
h	0068	ɔɪ	0254 026A
i	0069	ə	0259
j	006A	ɛ	025B

Character	Code	Character	Code
k	006B	ȝ	025D
l	006C	g	0261
;l	006C 0329	ı	026A
m	006D	ɹ	0279
n	006E	ʃ	0283
ɳ	006E 0329	ʊ	028A
ou	006F 028A	ʌ	028C
p	0070	ʍ	028D
s	0073	ʒ	0292
t	0074	dʒ	02A4
u	0075	tʃ	02A7
v	0076	θ	03B8

Farsi character set

For Farsi custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields.

Character	Code	Character	Code
ؚ	0621	ؠ	0638
܍	0622	ܕ	0639
܊	0623	ܖ	063A
܋	0624	ܔ	0641
܏	0626	ܔ	0642
ܐ	0627	ܔ	0644
ܑ	0628	ܔ	0645
ܒ	062A	ܔ	0646
ܓ	062B	ܔ	0647
ܔ	062C	ܔ	0648
ܕ	062D	ܔ	064E
ܖ	062E	ܔ	064F
ܗ	062F	ܔ	0650
ܙ	0630	ܔ	0651

Character	Code	Character	Code
ر	0631	ڦ	067E
ڙ	0632	ڻ	0686
ڦ	0633	ڢ	0698
ڦ	0634	ڪ	06A9
ڻ	0635	ڳ	06AF
ڻ	0636	ڻ	06CC
ٻ	0637		

You can use the following International Phonetic Alphabet in the `IPA` field of your vocabulary file:

Character	Code	Character	Code
b	0062	u	0075
d	0064	v	0076
f	0066	z	007A
g	0067	æ	00E6
h	0068	ڻ	0252
i	0069	ڦ	025B
j	006A	r	027E
k	006B	ڦ	0281
l	006C	ڻ	0283
m	006D	ڙ	0292
n	006E	ڙ	0294
o	006F	ڙ	0294
p	0070	dڙ	02A4
s	0073	tڙ	02A7
t	0074	څ	03C7

French character set

For French custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- ' (apostrophe)

- - (hyphen)
- . (period)

You can also use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
À	00C0	à	00E0
Â	00C2	â	00E2
Ç	00C7	ç	00E7
È	00C8	è	00E8
É	00C9	é	00E9
Ê	00CA	ê	00EA
Ë	00CB	ë	00EB
Î	00CE	î	00EE
Ï	00CF	ï	00EF
Ô	00D4	ô	00F4
Ö	00D6	ö	00F6
Ù	00D9	ù	00F9
Û	00DB	û	00FB
Ü	00DC	ü	00FC

You can use the following International Phonetic Alphabet in the `IPA` field of your vocabulary file:

Character	Code	Character	Code
a	0061	z	007A
b	0062	ã	00E3
d	0064	õ	00F5
e	0065	ø	00F8
f	0066	ŋ	014B
i	0069	œ	0153
j	006A	œ̃	0153 0303
k	006B	ə	0250
l	006C	ɔ	0254
m	006D	ə	0259

Character	Code	Character	Code
n	006E	ɛ	025B
o	006F	g	0261
p	0070	ɥ	0265
s	0073	ɲ	0272
t	0074	ㅂ	0281
u	0075	ʃ	0283
v	0076	ڙ	0292
w	0077	ڦ	1EBD
y	0079		

German character set

For German custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- ' (apostrophe)
- - (hyphen)
- . (period)

You can also use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
ä	00E4	Ä	00C4
ö	00F6	Ö	00D6
ü	00FC	Ü	00DC
ß	00DF		

You can use the following International Phonetic Alphabet characters in the `IPA` field of the vocabulary input file:

Character	Code	Character	Code
a	0061	ts	0074 0073
ai	0061 026A	u:	0075 02D0
au	0061 028A	v	0076
a:	0061 02D0	x	0078

Character	Code	Character	Code
b	0062	z	007A
d	0064	y:	0079 02D0
e:	0065 02D0	ã	00E3
f	0066	ç	00E7
g	0067	ø:	00F8 02D0
h	0068	ŋ	014B
i:	0069 02D0	œ	0153
j	006A	æ	0250 032F
k	006B	ɔ	0254
l	006C	ɔʏ	0254 028F
l	006C 0329	ə	0259
m	006D	ɛ	025B
ṁ	006D 0329	ɛ:	025B 02D0
n	006E	ɪ	026A
ṇ	006E 0329	ʊ	0281
o:	006F 02D0	ʃ	0283
p	0070	ʊ	028A
pf	0070 0066	ʏ	028F
s	0073	ť	02A7
t	0074		

Hebrew character set

For Hebrew custom vocabularies, you can use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
-	002D	□	05DD
׮	05D0	׫	05DE
ׯ	05D1	ױ	05DF
ײ	05D2	׳	05E0
״	05D3	׵	05E1
׶	05D4	׷	05E2

Character	Code	Character	Code
ଠ	05D5	ଙ	05E3
ର	05D6	ଙ	05E4
ନ	05D7	ୟ	05E5
ୱ	05D8	୪	05E6
୲	05D9	ଡ	05E7
ଞ	05DA	ଞ	05E8
ଙ	05DB	୬	05E9
ଖ	05DC	ଙ	05EA

You can use the following International Phonetic Alphabet characters in the `IPA` field of the vocabulary input file:

Character	Code	Character	Code
a	0061	p	0070
b	0062	s	0073
d	0064	t	0074
e	0065	u	0075
f	0066	v	0076
g	0067	w	0077
h	0068	z	007A
i	0069	ର୍ଜ	014B
j	006A	ୟ	0263
k	006B	ୱ	0283
l	006C	୩	0292
m	006D	୭	0294
n	006E	୯	03C7
o	006F		

Hindi character set

For Hindi custom vocabularies, you can use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
-	002D	ଥ	0925
.	002E	ଦ	0926
୨	0901	ଧ	0927
୯	0902	ନ	0928
:	0903	ପ	092A
ଓ	0905	ଫ	092B
ଆ	0906	ବ	092C
ଙ	0907	ଭ	092D
ଖ	0908	ମ	092E
ତ	0909	ଯ	092F
ଊ	090A	ର	0930
କୁ	090B	ଲ	0932
ଏ	090F	ବ	0935
ୟେ	0910	ଶ	0936
ଆଁ	0911	ଷ	0937
ଆଁ	0913	ସ	0938
ଆଁ	0914	ହ	0939
କ	0915	ଠ	093E
ଘ	0916	ଫି	093F
ଗ	0917	ଟି	0940
ଘ	0918	ଚ	0941
ଡ଼	0919	ର	0942
ଚ	091A	ର୍ର	0943
ଳ	091B	ରୁ	0945
ଜ	091C	ରୁ	0947
ଝ	091D	ରୂ	0948
ଝ	091E	ଙ୍କ	0949
ଟ	091F	ଙ୍କି	094B
ଠ	0920	ଙ୍କୁ	094C
ଡ	0921	ଙ୍କୁ	094D
ଢ	0922	ଙ୍କା	095B

Character	Code	Character	Code
ଣ	0923	ଡ୍	095C
ତ	0924	ଢ୍	095D

Amazon Transcribe maps the following characters:

Character	Mapped to
ନ (0929)	ନ (0928)
ର (0931)	ର (0930)
କ୍ର (0958)	କ୍ର (0915)
ଖ୍ର (0959)	ଖ୍ର (0916)
ଗ (095A)	ଗ (0917)
ଫ୍ର (095E)	ଫ୍ର (092B)
ଶ୍ର (095F)	ଶ୍ର (092F)

You can use the following International Phonetic Alphabet characters in the IPA field of your input file:

Character	Code	Character	Code
a:	0097 0720	ଏ	0331
b	0098	ବ୍	0598
b ^h	0098 0689	ବ୍ୟ	0596 0720
d	0100	ଦ୍	0598 0689
d ^h	0100 0689	ଦ୍ୟ	0601
e:	0101 0720	ୟେ	0603 0720
f	0102	ଫ୍	0609
i:	0105 0720	ି	0609 0689
j	0106	ଜ୍	0614
k	0107	କ୍	0618
k ^h	0107 0688	କ୍ୟ	0626
l	0108	ଲ୍	0627
m	0109	ମ୍	0638
n	0110	ନ୍	0642
o:	0111 0720	ୟୋ	0643
p	0112	ପ୍	0648

Character	Code	Character	Code
p ^h	0112 0688	t ^h	0648 0688
r	0114	ʊ	0650
s	0115	ʊ	0651
t	0116	dʒ	0676
t ^h	0116 0688	dʒ ^h	0676 0689
u:	0117 0720	tʃ	0679
z	0122	tʃ ^h	0679 0688

Indonesian character set

For Indonesian custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- ' (apostrophe)
- - (hyphen)
- . (period)

You can use the following International Phonetic Alphabet characters in the `IPA` field of your input file:

Character	Code	Character	Code
a	0061	r	0072
ai	0061 0069	s	0073
au	0061 0075	t	0074
b	0062	tʃ	0074 0283
d	0064	u	0075
dʒ	0064 0292	v	0076
e	0065	w	0077
f	0066	x	0078
h	0068	y	0079
i	0069	ŋ	014B
j	006A	ɔ	0254
k	006B	ə	0259
l	006C	ɛ	025B
m	006D	g	0261

Character	Code	Character	Code
n	006E	ȝ	0263
o	006F	ȝ	026A
ɔ̄	006F 0069 032F	ɲ	0272
p	0070	ʃ	0283
q	0071	ʊ	028A

Italian character set

For Italian custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- ' (apostrophe)
- - (hyphen)
- . (period)

You can also use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
À	00C0	à	00E0
Ä	00C4	ä	00E4
Ҫ	00C7	ҫ	00E7
È	00C8	è	00E8
É	00C9	é	00E9
Ê	00CA	ê	00EA
Ӭ	00CB	ӗ	00EB
ѝ	00CC	ѝ	00EC
Ӯ	00D2	Ӯ	00F2
Ӱ	00D9	Ӱ	00F9
Ӯ	00DC	Ӯ	00FC

You can use the following International Phonetic Alphabet characters in the `IPA` field of your input file:

Character	Code	Character	Code
a	0061	ss	0073 0073

Character	Code	Character	Code
b	0062	t	0074
bb	0062 0062	tt	0074 0074
d	0064	u	0075
dd	0064 0064	v	0076
e	0065	vv	0076 0076
f	0066	w	0077
ff	0066 0066	z	007A
gg	0067 0067	ɔ	0254
i	0069	ɛ	025B
j	006A	g	0261
k	006B	ɳ	0272
kk	006B 006B	ɳɳ	0272 0272
l	006C	ʃ	0283
ll	006C 006C	ʃʃ	0283 0283
m	006D	ʌ	028E
mm	006D 006D	ʌʌ	028E 028E
n	006E	ð	02A3
nn	006E 006E	ðð	02A3 02A3
o	006F	ðʒ	02A4
p	0070	ðʒðʒ	02A4 02A4
pp	0070 0070	ts	02A6
r	0072	ts-ts	02A6 02A6
rr	0072 0072	tʃ	02A7
s	0073	tʃtʃ	02A7 02A7

Japanese character set

For Japanese custom vocabularies, the `Phrase` and `SoundsLike` fields can use any of the characters listed in the following file on GitHub.

- [ja-jp-character-set.txt](#)

You can use the following International Phonetic Alphabet characters in the `IPA` field of your input file:

Character	Code	Character	Code
a	0061	p	0070
a:	0061 02D0	s	0073
b	0062	t	0074
d	0064	ts	0074 0073
dz	0064 007A	tç	0074 0255
dzš	0064 0291	w	0077
e	0065	z	007A
e:	0065 02D0	ç	00E7
g	0067	ŋ	014B
h	0068	ç	0255
i	0069	ɯ	026F
i:	0069 02D0	ɯ:	026F 02D0
j	006A	ɳ	0274
k	006B	ɸ	0278
m	006D	r	027E
n	006E	z	0291
o	006F	?	0294
o:	006F 02D0		

Korean character set

For Korean custom vocabularies, you can use any of the Hangul syllables in the `Phrase` and `SoundsLike` fields. For more information, see [Hangul Syllables](#) on Wikipedia.

You can use the following International Phonetic Alphabet characters in the `IPA` field of your input file:

Character	Code	Character	Code
a	00061	s#	0073 0348
e	00065	t	0074
h	00068	tç	0074 0255
i	00069	tçʰ	0074 0255 02B0
je	006A 0065	tʰ	0074 02B0
jo	006A 006F	t#	0074 0348
ju	006A 0075	t#ç	0074 0348 0255

Character	Code	Character	Code
jɛ	006A 025B	u	0075
jʌ	006A 028C	we	0077 0065
ja	006A 0061	wi	0077 0069
k	006B	wɛ	0077 025B
k ^h	006B 02B0	wʌ	0077 028C
k#	006B 0348	wa	0077 0061
l	006C	ø	00F8
m	006D	ŋ	0014B
n	006E	ɛ	0025B
o	006F	ɯ	026F
p	0070	wi	006F 0069
p ^h	0070 02B0	r	027E
p#	0070 0348	ʌ	028C
s	0073		

Malay character set

For Malay custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- ' (apostrophe)
- - (hyphen)
- . (period)

You can use the following International Phonetic Alphabet characters in the `IPA` field of your input file:

Character	Code	Character	Code
F	0046	r	0072
a	0061	s	0073
ai	0061 0069	t	0074
au	0061 0075	tʃ	0074 0283
b	0062	v	0076
d	0064	w	0077
dʒ	0064 0292	x	0078

Character	Code	Character	Code
e	0065	y	0079
h	0068	ñ	014B
i	0069	ɔ	0254
j	006A	ə	0259
k	006B	ɛ	025B
l	006C	g	0261
m	006D	ɣ	0263
n	006E	ɪ	026A
o	006F	ɲ	0272
ó	006F 0069 32F	ʃ	0283
p	0070	ʊ	028A
q	0071	ʊi	028A 0069

Portuguese character set

For Portuguese custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- ' (apostrophe)
- - (hyphen)
- . (period)

You can also use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
À	00C0	à	00E0
Á	00C1	á	00E1
Â	00C2	â	00E2
Ã	00C3	ã	00E3
Ä	00C4	ä	00E4
Ҫ	00C7	ҫ	00E7
È	00C8	è	00E8
É	00C9	é	00E9
Ê	00CA	ê	00EA

Character	Code	Character	Code
Ё	00CB	ë	00EB
Í	00CD	í	00ED
Ñ	00D1	ñ	00F1
Ó	00D3	ó	00F3
Ô	00D4	ô	00F4
Õ	00D5	õ	00F5
Ö	00D6	ö	00F6
Ú	00DA	ú	00FA
Ü	00DC	ü	00FC

You can use the following International Phonetic Alphabet characters in the **IPA** field of your input file:

Character	Code	Character	Code
a	0061	v	0076
b	0062	w	0077
d	0064	w~	0077 0303
e	0065	z	007A
f	0066	ð	00F5
g	0067	˜	00129
i	0069	˜	00169
j	006A	˜	0250 0303
k	006B	ɔ	0254
l	006C	ɛ	025B
m	006D	ŋ	0272
n	006E	r	027E
o	006F	ụ	0281
p	0070	ʃ	0283
s	0073	ʌ	028E
t	0074	ʒ	0292
ť	0074 0283	dʒ	02A4
u	0075	ẽ	1EBD

Russian character set

For Russian custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
'	0027	п	043F
-	002D	р	0440
.	002E	с	0441
а	0430	т	0442
б	0431	у	0443
в	0432	ф	0444
г	0433	х	0445
д	0434	ц	0446
е	0435	ч	0447
ж	0436	ш	0448
з	0437	щ	0449
и	0438	ъ	044A
й	0439	ы	044B
к	043A	ь	044C
л	043B	э	044D
м	043C	ю	044E
н	043D	я	044F
о	043E	ё	0451

You can use the following International Phonetic Alphabet characters in the `IPA` field of your input file:

Character	Code	Character	Code
b	0062	t	0074
b'	0062 02B2	tʃ	0074 0283
d	0064	t̪	0074 02B2
d'	0064 02B2	u	0075
f	0066	v	0076
f'	0066 02B2	v̪	0076 02B2
g	0067	x	0078

Character	Code	Character	Code
g ^j	067 02B2	x ^j	0078 02B2
i	0069	z	007A
j	006A	z ^j	007A 02B2
k	006B	æ	00E6
k ^j	006B 02B2	ə	0259
l	006C	ɛ	025B
l ^j	006C 02B2	ѣ	0268
m	006D	ſ	0283
m ^j	006D 02B2	ſ̥	0283 02B2
n	006E	υ	028A
n ^j	006E 02B2	ʌ	028C
p	0070	ȝ	0292
p ^j	0070 02B2	'i	02C8 0069
r	0072	'o	02C8 006F
r ^j	0072 02B2	'v	02C8 0075
s	0073	'ɛ	02C8 025B
s ^j	0073 02B2	'ѣ	02C8 0268
ts	0074 0073	'a	02C8 0061

Spanish character set

For Spanish custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- ' (apostrophe)
- - (hyphen)
- . (period)

You can also use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
Á	00C1	á	00E1
É	00C9	é	00E9
Í	00CD	ë	00ED

Character	Code	Character	Code
Ó	00D3	ó	0XF3
Ú	00DA	ú	00FA
Ñ	00D1	ñ	0XF1
ü	00FC		

You can use the following International Phonetic Alphabet characters in the `IPA` field of your input file:

Character	Code	Character	Code
a	0061	r	0072
b	0062	s	0073
d	0064	t	0074
e	0065	u	0075
f	0066	v	0076
g	0067	w	0077
h	0068	x	0078
i	0069	z	007A
j	006A	ŋ	014B
k	006B	ɳ	0272
l	006C	r̪	027E
m	006D	ʃ	0283
n	006E	ʒ	029D
o	006F	t̪ʃ	02A7
p	0070	θ	03B8

Tamil character set

For Tamil custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
ஃ	0B85	ஃ	0BB0
ஃ	0B86	வ	0BB2
ஃ	0B87	ஏ	0BB5
ஃ	0B88	ஃ	0BB4

Character	Code	Character	Code
ଙ	0B89	ଞ	0BB3
ଡୋର	0B8A	ନ୍ଦ	0BB1
ଶ୍ର	0B8E	ନେ	0BA9
ଶ୍ର	0B8F	ଜ୍ବ	0B9C
ଜ୍ବ	0B90	#	0BB6
ଫ୍ର	0B92	ଏଟ୍	0BB7
ଫ୍ର	0B93	ସ୍ଲ	0BB8
ଓଳ	0B94	ମ୍ବ	0BB9
ଙ୍ଗ	0B83	.	0BCD
କ୍ଷ	0B95	ର୍ମ	0BBE
ଙ୍କ	0B99	ଗ୍ର	0BBF
ଖ୍ର	0B9A	ମୁଁ	0BC0
ଙ୍ଗ୍ର	0B9E	ମୁଁ	0BC1
ଲ୍ର	0B9F	ମୁଁ	0BC2
ମୋର	0BA3	ରେ	0BC6
ତ୍ର	0BA4	ରେ	0BC7
ର୍ତ୍ତ	0BA8	ରେ	0BC8
ଲ୍ର	0BAA	ରେର୍	0BCA
ମ୍ର	0BAE	ରେର୍	0BCB
ଶ୍ର	0BAF	ରେର୍	0BCC

You can use the following International Phonetic Alphabet characters in the IPA field of your input file:

Character	Code	Character	Code
a	0061	v	0076
a:	0061 02D0	w	0077
b	0062	z	007A
d	0064	æ	00E6
dʒ	0064 0292	ð	00F0
e	0065	ŋ	014B
f	0066	ə	0251
g	0067	ɔ	0254

Character	Code	Character	Code
h	0068	ə	0259
i	0069	ɛ	025B
i:	0069 02D0	g	0261
j	006A	ɪ	026A
k	006B	l	026D
l	006C	ɳ	0272
m	006D	ɳ̥	0273
n	006E	ʐ	0279
ɳ	006E 032A	ʐ̥	0279
o	006F	ɹ	0279 0329
o:	006F 02D0	r̥	027E
p	0070	s̥	0282
r	0072	ʃ̥	0283
s	0073	t̥	0288
t	0074	ʊ̥	028A
ʈ	0074 032A	ʊ̥̥	028B
ʈʂ	0074 0283	ʌ̥	028C
u	0075	ɜ̥	0292
u:	0075 02D0	θ̥	03B8

Telugu character set

For Telugu custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
-	002D	ఁ	0C24
ం	0C01	ఁ	0C25
ం	0C02	ఁ	0C26
ః	0C03	ఁ	0C27
ఁ	0C05	ఁ	0C28
ఁ	0C06	ఁ	0C2A
ఁ	0C07	ఁ	0C2B

Character	Code	Character	Code
ଙ୍ଗ	0C08	ବ	0C2C
ଖ୍ରୀ	0C09	ଫ୍ରୀ	0C2D
ମ୍ହୀ	0C0A	ମୁ	0C2E
ମୁ୰ୁ	0C0B	ଯୁ	0C2F
ନ୍ତୁ	0C0C	ରୁ	0C30
ନ୍ତି	0C0E	ଣୁ	0C31
ନ୍ତିରୁ	0C0F	ଳୁ	0C32
ନ୍ତିରୁ	0C10	ଛୁ	0C33
ନ୍ତିରୁ	0C12	ଶୁ	0C35
ନ୍ତିରୁ	0C13	ଷୁ	0C36
ନ୍ତିରୁ	0C14	ସୁ	0C37
ନ୍ତିରୁ	0C15	ସୁ	0C38
ନ୍ତିରୁ	0C16	ପୁ	0C39
ନ୍ତିରୁ	0C17	ରୁ	0C3E
ନ୍ତିରୁ	0C18	ଋୁ	0C3F
ନ୍ତିରୁ	0C19	ଣୁ	0C40
ନ୍ତିରୁ	0C1A	ଜୁ	0C41
ନ୍ତିରୁ	0C1B	ଝୁ	0C42
ନ୍ତିରୁ	0C1C	ଙୁ	0C43
ନ୍ତିରୁ	0C1D	ଙ୍ଗୁ	0C44
ନ୍ତିରୁ	0C1E	ଦୁ	0C47
ନ୍ତିରୁ	0C1F	ଲୁ	0C48
ନ୍ତିରୁ	0C20	ଇୟୁ	0C4A
ନ୍ତିରୁ	0C21	ଈୟୁ	0C4B
ନ୍ତିରୁ	0C22	ଏୟୁ	0C4C
ନ୍ତିରୁ	0C23	ୟୁ	0C4D

You can use the following International Phonetic Alphabet characters in the `IPA` field of your input file:

Character	Code	Character	Code
ଡ	0064 032A	ଢ	00F0
ଡ଼	0064 032A 0324	ଣ	014B

Character	Code	Character	Code
dʒ	0064 0292	ə	0251
dʒ	0064 0292 0324	ɔ	0254
e	0065	ɖ	0256
e:	0065 02D0	ڏ	0256 0324
f	0066	ə	0259
h	0068	ɛ	025B
i	0069	g	0261
iz	0069 0290	ڳ	0261 0324
j	006A	ڀ	026A
k	006B	l	026D
kʰ	006B 02B0	ڻ	0272
l	006C	ڻ	0273
m	006D	ڙ	0279
n	006E	ڙ	0279 0329
o	006F	ڦ	027D
o:	006F 02D0	ڦ	0282
p	0070	ڢ	0283
pʰ	0070 02B0	ڦ	0288
r	0072	ڦ	0288 02B0
s	0073	ڻ	028A
t	0074	ڻ	028B
t̪	0074 032A	ڦ	028C
t̪ʰ	0074 032A 02B0	ڙ	0292
t̪	0074	ڦ	03B8
t̪ʰ	0074 0283 02B0		

Thai character set

For Thai custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- - (hyphen)
- . (period)

You can also use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
କ	OE01	ଲ	OE25
ୟ	OE02	ଗ	OE26
ୟ	OE03	ର	OE27
ମ	OE04	ଶ	OE28
ମ	OE05	ଷ	OE29
ୟ	OE06	ସ	OE2A
୳	OE07	ହ	OE2B
ଜ	OE08	ପ	OE2C
ଙ	OE09	ବ	OE2D
୯	OE0A	ଶ	OE2E
୯	OE0B	୧	OE2F
ଚ	OE0C	୩	OE30
ଘ	OE0D	୤	OE31
ଙ	OE0E	୦	OE32
ଙ	OE0F	୨	OE34
୭	OE10	୪	OE35
୭	OE11	୫	OE36
ଣ	OE12	୬	OE37
ଣ	OE13	୦	OE38
ଢ	OE14	୧	OE39
ଢ	OE15	୨	OE3A
ଠ	OE16	୩	OE40
ଠ	OE17	୫	OE41
ଥ	OE18	୬	OE42
ଥ	OE19	୦	OE43
ଥ	OE1A	୧	OE44
ଥ	OE1B	୨	OE45
ଥ	OE1C	୩	OE46
ଥ	OE1D	୪	OE47
ଥ	OE1E	୫	OE48

Character	Code	Character	Code
₪	0E1F	׵	0E49
׷	0E20	׸	0E4A
׹	0E21	׺	0E4B
׻	0E22	׼	0E4C
״	0E23	׽	0E4D
׶	0E24		

You can use the following International Phonetic Alphabet characters in the **IPA** field of your input file:

Character	Code	Character	Code
a	0061	p ^h	0070 02B0
a:	0061 02D0	r	0072
b	0062	s	0073
d	0064	t	0074
e	0065	t ^h	0074 02B0
e:	0065 02D0	t̪	0074 0361 0255
f	0066	t̪ ^h	0074 0361 0255 02B0
h	0068	u	0075
i	0069	u:	0075 02D0
i:	0069 02D0	u:a	0075 02D0 0061
i:a	0069 02D0 0061	w	0077
j	006A	ŋ	014B
k	006B	ɔ	0254
kw	006B 0077	ɔ:	0254 02D0
k ^h	006B 02B0	ɛ	025B
k ^{hw}	006B 02B0 0077	ɛ:	025B 02D0
l	006C	ɣ	0264
m	006D	ɣ:	0264 02D0
n	006E	ɯ	026F
o	006F	ɯ:	026F 02D0
o:	006F 02D0	ɯ:a	026F 02D0 0061
p	0070	?	0294

Turkish character set

For Turkish custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` fields:

- a - z
- A - Z
- ' (apostrophe)
- - (hyphen)
- . (period)

You can also use the following Unicode characters in the `Phrase` and `SoundsLike` fields:

Character	Code	Character	Code
ç	00C7	ö	00F6
Ö	00D6	û	00FB
Ü	00DC	ü	00FC
â	00E2	Ğ	011E
ä	00E4	ğ	011F
ç	00E7	ı	0130
è	00E8	ı	0131
é	00E9	§	015E
ê	00EA	§	015F
í	00ED	š	0161
î	00EE	ž	017E
ó	00F3		

You can use the following International Phonetic Alphabet characters in the `IPA` field of your input file:

Character	Code	Character	Code
a	0061	u	0075
a:	0061 02D0	u:	0075 02D0
b	0062	v	0076
c	0063	w	0077
d	0064	y	0079
e	0065	y:	0079 02D0
e:	0065 02D0	z	007A

Character	Code	Character	Code
f	0066	ø	00F8
g	0067	ø:	00F8 02D0
h	0068	ŋ	014B
i	0069	ʃ	025F
i:	0069 02D0	χ	0263
j	006A	†	026B
k	006B	ɯ	026F
l	006C	ɯ:	026F 02D0
m	006D	ɾ	027E
n	006E	ʃ	0283
o	006F	ʒ	0292
o:	006F 02D0	?	0294
p	0070	ðʒ	02A4
s	0073	ɸ	02A7
t	0074		

Alternative transcriptions

When Amazon Transcribe transcribes an audio file, it returns the transcription with the highest confidence level. You can specify that Amazon Transcribe return additional transcriptions with lower confidence levels. Use alternative transcriptions to see different interpretations of the transcribed audio. For example, in an application that enables a person to review the transcription, you can present the alternative transcriptions for the person to choose from. Alternative transcriptions are only available for the [StartTranscriptionJob \(p. 376\)](#) API.

You can configure Amazon Transcribe to return alternative transcription using the console or by using the Amazon Transcribe API. To get alternative transcriptions using the API, set the `ShowAlternatives` field to `true` and set the `MaxAlternatives` field to the number of alternatives to return when you call the [StartTranscriptionJob \(p. 376\)](#) API. You can specify that Amazon Transcribe return up to 10 alternative transcriptions.

You can combine alternative transcriptions with speaker identification and channel identification. Alternative transcriptions are available in all supported languages.

Alternatives are presented at the segment level of the transcription. Segments are defined by natural pauses in speech, such as a change in speaker or pause in the audio. For example, the spoken phrase "It is raining today in Seattle, but not in Portland" is separated into two segments: "It is raining today in Seattle" and "but not in Portland."

Amazon Transcribe returns an overall transcription of your audio file in the response. When you have configured Amazon Transcribe to return alternatives, the overall transcription is built from the segment alternatives with the highest confidence level. Alternative transcriptions are returned in the segments

structure in the output JSON. If Amazon Transcribe doesn't find alternatives, it returns fewer than the number of alternatives specified in the `MaxAlternatives` field.

The following is the JSON output from Amazon Transcribe. It is the transcription output for this input:
Uh, you can just call this number if I don't pick up, just leave a voicemail and I'll get back to you. Okay. And that's the number. The 1166 number, you mean?

The following is the JSON output with `ShowAlternatives` set to `false`.

```
{
  "results": {
    "transcripts": [
      "Uh, you can just call this number if I don't pick up and leave a voicemail and
      I'll get back to you. Okay. And that's the number. The 1166 number, you mean"
    ],
    "items": [
      {
        "start_time": 12.35,
        "end_time": 12.57,
        "alternatives": [
          {
            "confidence": 0.9989,
            "content": "Uh"
          }
        ],
        "type": "pronunciation"
      },
      Items removed for brevity.
    ]
  }
}
```

The following is the JSON output for the same input with `ShowAlternatives` set to `true` and `MaxAlternatives` set to `2`.

```
{
  "results": {
    "transcripts": [
      "Uh, you can just call this number if I don't pick up and leave a voicemail and
      I'll get back to you. Okay. And that's the number. The 1166 number, you mean"
    ],
    "items": [
      {
        "start_time": 12.35,
        "end_time": 12.57,
        "alternatives": [
          {
            "confidence": 0.9989,
            "content": "Uh"
          }
        ],
        "type": "pronunciation"
      },
      Items removed for brevity..
    ],
    "segments": [
      {
        "start_time": 11.84,
        "end_time": 19.665,
        "alternatives": [
          {

```

```

    "transcript": "Uh, you can just call this number if I don't pick up
and leave a voicemail and I'll get back to you.",
    "items": [
        {
            "start_time": 12.35,
            "end_time": 12.57,
            "confidence": 0.9989,
            "content": "Uh",
            "type": "pronunciation"
        },
        Items removed for brevity.
        {
            "start_time": 16.42,
            "end_time": 16.52,
            "confidence": 0.7572,
            "content": "and",
            "type": "pronunciation"
        },
        Items removed for brevity.
    ]
},
{
    "transcript": "Uh, you can just call this number if I don't pick
up, just leave a voicemail and I'll get back to you.",
    "items": [
        {
            "start_time": 12.35,
            "end_time": 12.57,
            "confidence": 0.9989,
            "content": "Uh",
            "type": "pronunciation"
        },
        Items removed for brevity..
        {
            "start_time": 16.42,
            "end_time": 16.52,
            "content": ",",
            "type": "punctuation"
        },
        {
            "start_time": 16.42,
            "end_time": 16.52,
            "confidence": 0.8934,
            "content": "just",
            "type": "punctuation"
        },
        Items removed for brevity..
    ],
    Alternatives removed for brevity.
],
Segments removed for brevity..
]
}
}

```

Job queuing

When you send transcription jobs to Amazon Transcribe, there is a limit to the total number of jobs that can run at one time. By default, there are 250 slots for jobs. When the limit is reached, you must wait until one or more jobs have finished and freed up a slot before you can send your next job.

To queue jobs so that they run as soon as a slot becomes available, you can use *job queuing*. Job queuing creates a queue on your behalf that contains your jobs. When a slot is available, Amazon Transcribe takes the next job from the queue and immediately starts processing it. To allow resources for new jobs to be submitted and processed, Amazon Transcribe uses at most 90 percent of your slots to process jobs in the queue.

You can turn on job queuing with the console, or you can set the `AllowDeferredExecution` field of the `JobExecutionSettings` parameter to `true` when you call the [StartTranscriptionJob \(p. 376\)](#) API.

When you submit a job with job queuing turned on, one of the following things happens.

- If slots are available, the job is processed immediately.
- If no slots are available, the job is sent into a queue. When slots become available, jobs are removed from the queue in FIFO order (first in, first out).

You can see the progress of a queued job using the console or by using the [GetTranscriptionJob \(p. 329\)](#) API. When a job is queued, the `Status` field of the `TranscriptionJob` object returned by the [StartTranscriptionJob \(p. 376\)](#) API is set to `QUEUED`. The status changes to `IN_PROGRESS` when Amazon Transcribe starts processing the audio, and then changes to either `COMPLETED` or `FAILED` when processing is finished. You can use the `TranscriptionJobName` field with the [GetTranscriptionJob \(p. 329\)](#) API to monitor the status of a job.

You can submit up to 10,000 jobs to the queue. If you exceed 10,000 jobs, you get a `LimitExceededConcurrentJobException` exception.

IAM policies for job queuing

To use job queuing, you must provide Amazon Transcribe with a data access role that permits access to your audio file to be transcribed. You can choose the data access role using the console, or you use the `DataAccessRoleArn` field of the `JobExecutionSettings` parameter of the [StartTranscriptionJob \(p. 376\)](#) API to specify the role to use.

The role policies that you use depend on where you are storing your input files, where you are storing your output files, and whether you are encrypting the output with an AWS KMS customer master key (CMK). The IAM policies in this section are required for the role. The policies enable Amazon Transcribe to work on your behalf, allow access to the input and output locations for your jobs, and enable Amazon Transcribe to use a AWS KMS CMK to encrypt your transcriptions.

Trust policy

The data access role that you use for transcription must have a trust policy that enables Amazon Transcribe to assume the role. Use the following trust policy.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "",  
            "Effect": "Allow",  
            "Principal": {  
                "Service": [  
                    "transcribe.amazonaws.com",  
                ]  
            },  
            "Action": "sts:AssumeRole"  
        }  
    ]  
}
```

```
        ],
    }
```

Input bucket policy

The following IAM policy gives the data access role permission to read files from your input bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3>ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::input-bucket-name",
                "arn:aws:s3:::input-bucket-name/*"
            ]
        }
    ]
}
```

Output bucket policy

The following IAM policy gives the data access role permission to write files to your output bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::output-bucket-name/*"
            ]
        }
    ]
}
```

AWS KMS key policy for input buckets

If you have encrypted your input files, the data access role needs permission to use the AWS KMS key to decrypt the files. The following policy provides that permission.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:::input-bucket-cmk-name"
            ]
        }
    ]
}
```

AWS KMS key policy for output buckets

To encrypt your output transcriptions, the data access role needs permission to use the AWS KMS key. The following policy provides that permission.

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": [  
            "kms:GenerateDataKey*"  
        ],  
        "Resource": [  
            "arn:aws:kms:::output-bucket-cmk-name"  
        ]  
    }  
}
```

Tagging resources

A *tag* is a custom metadata label that you can add to a resource in order to make it easier to identify, organize, and find in a search. Tags are comprised of two individual parts: A tag key and a tag value. This is referred to as a key:value pair.

A *tag key* typically represents a larger category, while a *tag value* represents a subset of that category. For example you could have *tag key=Color* and *tag value=Blue*, which would produce the key:value pair *Color:Blue*. Note that you can set the value of a tag to an empty string, but you can't set the value of a tag to null. Omitting the tag value is the same as using an empty string.

Tip

AWS Billing can use tags to separate your bills into dynamic categories. For example, if you add tags to represent different departments within your company, such as *Department:Sales* or *Department:Legal*, AWS can provide you with your cost distribution per department.

In Amazon Transcribe, you can tag the following resources:

- Transcription job
- Medical transcription job
- Vocabulary
- Medical vocabulary
- Vocabulary filter
- Custom language model

Tag keys can be up to 128 characters in length and tag values can be up to 256 characters in length; both are case sensitive. For more information, see [TagResource \(p. 383\)](#) in the [Amazon Transcribe API Reference \(p. 275\)](#).

Amazon Transcribe supports up to 50 tags per resource. For a given resource, each tag key must be unique with only one value.

Note

Your tags cannot begin with *aws:* because AWS reserves this prefix for system-generated tags. You cannot add, modify, or delete *aws:** tags, and they don't count against your tags-per-resource limit.

To learn more about tagging, including best practices, see [Tagging AWS resources](#).

Tag-based access control

You can use tags to control access within your AWS accounts. For tag-based access control, you provide tag information in the condition element of an IAM policy. You can then use tags and their associated tag condition key to control access to:

- **Resources:** Control access to your Amazon Transcribe resources based on the tags you've assigned to those resources.
 - Use the `aws:ResourceTag/key-name` condition key to specify which tag key:value pair must be attached to the resource.
- **Requests:** Control which tags can be passed in a request.
 - Use the `aws:RequestTag/key-name` condition key to specify which tags can be added, modified, or removed from an IAM user or role.
- **Authorization processes:** Control tag-based access for any part of your authorization process.
 - Use the `aws:TagKeys/` condition key to control whether specific tag keys can be used on a resource, in a request, or by a principal. In this case, the key value doesn't matter.

For more detailed information on tag-based access control, see [Controlling access to AWS resources using tags](#).

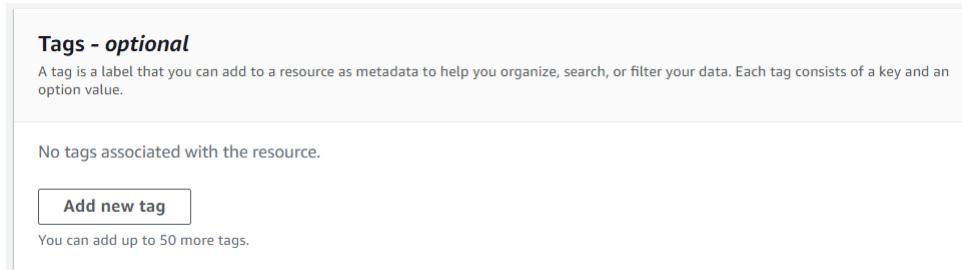
Tagging your Amazon Transcribe resources

You can add tags before or after you run your Amazon Transcribe job. The existing **Create*** and **Start*** APIs allow you to add tags with your transcription job.

You can add, modify or delete tags using the **Amazon Transcribe Console**, **AWS CLI**, or **AWS SDK**; see below for instructions:

Console

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, choose **Transcription jobs**, then select the **Create job** button (top right). This opens the **Specify job details** page.
3. Scroll to the bottom of the **Specify job details** page to find the **Tags - optional** box and click the **Add new tag** button.



4. Enter information for the **Key** field and, optionally, the **Value** field.

Tags - optional

A tag is a label that you can add to a resource as metadata to help you organize, search, or filter your data. Each tag consists of a key and an option value.

Key	Value - optional	
<input type="text" value="color"/> X	<input type="text" value="blue"/> X	Remove
Add new tag		
You can add up to 49 more tags.		

5. Fill in any other fields you wish to include on the **Specify job details** page, then click the **Next** button. This takes you to the **Configure job - optional** page.
- Click the **Create job** button to run your transcription job.
6. You can view the tags associated with a transcription job by navigating to the **Transcription jobs** page, selecting a transcription job, and scrolling to the bottom of that job's information page. If you wish to edit your tags, you can do so by clicking the **Manage tags** button.

Tags (2)		Manage Tags
Key	Value	
color	blue	

AWS SDK for Python (Boto3)

The following example uses the AWS SDK for Python (Boto3) to add a tag by using the `Tags` argument for the `start_transcription_job` method:

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "your-job-name"
job_uri = "s3://your-S3-bucket/S3-prefix/your-filename.file-extension"
transcribe.start_transcription_job(
    TranscriptionJobName=job_name,
    Media={'MediaFileUri': job_uri},
    MediaFormat='wav',
    LanguageCode='en-US',
    Tags=[{'Key': 'string', 'Value':'string'}]
)
while True:
    status = transcribe.get_transcription_job(TranscriptionJobName=job_name)
    if status['TranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED', 'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)
```

AWS CLI

This example uses the `start-transcription-job` command and `Tags` parameter.

```
aws transcribe start-transcription-job \
--transcription-job-name your-job-name
```

```
--media MediaFileUri=s3://your-S3-bucket/S3-prefix/your-filename.file-extension \
--language-code en-US \
--tags Key=color,Value=blue
```

Here's another example using the [start-transcription-job](#) command, and a request body that adds tags to that job.

```
aws transcribe start-transcription-job \
--cli-input-json file://filepath/example-start-command.json
```

The file *example-start-command.json* contains the following request body.

```
{
    "TranscriptionJobName": "your-job-name",
    "LanguageCode": "en-US",
    "Media": {
        "MediaFileUri": "s3://your-S3-bucket/S3-prefix/your-filename.file-extension"
    },
    "Tags": [ {"Key": "string","Value": "string"} ]
}
```

Guidelines and quotas

Amazon Transcribe has several guidelines to follow in order to achieve optimal results. There are also quotas that may impact your transcriptions; some of these can be increased. Refer to the below sections for details.

Supported Regions

For a list of AWS Regions where Amazon Transcribe is available, see [Amazon Transcribe Endpoints and Quotas](#) in the *Amazon Web Services General Reference*.

Guidelines

For best results:

- Use a lossless format, such as FLAC or WAV, with PCM 16-bit encoding.
- Use a sample rate of 8,000 Hz for low-fidelity audio and 16,000 Hz for high-fidelity audio.

Note

Amazon Transcribe may temporarily store your content to continuously improve the quality of its analysis models. See the [Amazon Transcribe FAQ](#) to learn more. To request that we delete content that may have been stored by Amazon Transcribe, open a case with [AWS Support](#).

If you don't need to process all of your transcription jobs concurrently, use [Job queuing \(p. 47\)](#). This enables Amazon Transcribe to keep track of your transcription jobs and process them when slots are available. You can request an increase to the job queue bandwidth ratio to run more transcription jobs. The quota for the transcription jobs in your job queue is the product of the number of transcription jobs you can run concurrently and the bandwidth ratio. For example, if you have a bandwidth ratio of 5 and a quota of 100 for the number of transcription jobs you can run concurrently then you can have 500 transcription jobs in your job queue.

Quotas

You can request a quota increase for the following resources:

Resource	Default
Number of concurrent batch transcription jobs	250
Number of concurrent batch transcription jobs (call analytics)	100
Job queue bandwidth ratio	0.9

Resource	Default
Number of concurrent HTTP/2 streams for streaming transcription	25
Number of StartStreamTranscription (p. 408) WebSocket requests	25
Total number of vocabularies per account	100
Number of pending vocabularies	10
Number of concurrently training custom language models	3
Total number of custom language models per account	10
Number of channels for channel identification	2
Number of categories per account (call analytics)	200
Number of rules per category (call analytics)	20

The below operations limits can also be increased upon request:

Operation	Transactions per second
StartTranscriptionJob (p. 376)	25
StartStreamTranscription (p. 408)	25
GetTranscriptionJob (p. 329)	30
DeleteTranscriptionJob (p. 309)	5
ListTranscriptionJobs (p. 356)	5
CreateVocabulary (p. 291)	10
UpdateVocabulary (p. 395)	10
DeleteVocabulary (p. 311)	5
GetVocabulary (p. 332)	20
ListVocabularies (p. 359)	5
StartCallAnalyticsJob (p. 365)	10

Operation	Transactions per second
GetCallAnalyticsJob (p. 320)	20
ListCallAnalyticsJobs (p. 342)	5
DeleteCallAnalyticsJob (p. 301)	5
CreateCallAnalyticsCategory (p. 278)	10
UpdateCallAnalyticsCategory (p. 387)	10
DeleteCallAnalyticsCategory (p. 299)	5
GetCallAnalyticsCategory (p. 317)	20
ListCallAnalyticsCategories (p. 338)	5

Note

For information about requesting a quota increase, see [AWS Service Quotas](#) in the *Amazon Web Services General Reference*.

The following quotas **cannot** be changed:

Description	Quota
Audio file length	4:00:00 (four) hours (14,400 seconds)
Audio file size	2 GB
Audio file size (call analytics)	500 MB
Size of a custom vocabulary	50 KB
Length of a custom vocabulary phrase	256 characters
Size of a vocabulary filter	50 KB
Number of vocabulary filters	100
Number of days job records are retained	90
Minimum audio file duration	500 milliseconds (ms)

Getting started with Amazon Transcribe

To get started using Amazon Transcribe, set up an AWS account and create an AWS Identity and Access Management (IAM) user. To use the AWS Command Line Interface (AWS CLI), download and configure it.

Topics

- [Step 1: Set up an AWS account and create an administrator user \(p. 57\)](#)
- [Step 2: Set up the AWS Command Line Interface \(AWS CLI\) \(p. 58\)](#)
- [Step 3: Getting started using the console \(p. 59\)](#)
- [Step 4: Getting started using the API \(p. 62\)](#)
- [Step 5: Getting started with streaming audio \(p. 66\)](#)

Step 1: Set up an AWS account and create an administrator user

Before you use Amazon Transcribe for the first time, complete the following tasks:

1. [Sign up for AWS \(p. 57\)](#)
2. [Create an IAM user \(p. 57\)](#)

Sign up for AWS

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all AWS services, including Amazon Transcribe. You are charged only for the services that you use.

With Amazon Transcribe, you pay only for the resources that you use. If you are a new AWS customer, you can get started with Amazon Transcribe for free. For more information, see [AWS Free Usage Tier](#).

If you already have an AWS account, skip to the next section.

To create an AWS account

1. Open <https://portal.aws.amazon.com/billing/signup>.
2. Follow the online instructions.

Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

Record your AWS account ID because you'll need it for the next task.

Create an IAM user

Services in AWS, such as Amazon Transcribe, require that you provide credentials when you access them. This allows the service to determine whether you have permissions to access the service's resources.

We strongly recommend that you access AWS using AWS Identity and Access Management (IAM), not the credentials for your AWS account. To use IAM to access AWS, create an IAM user, add the user to an IAM

group with administrative permissions, and then grant administrative permissions to the IAM user. You can then access AWS using a special URL and the IAM user's credentials.

The Getting Started exercises in this guide assume that you have a user with administrator privileges, `adminuser`.

To create an administrator user and sign in to the console

1. Create an administrator user called `adminuser` in your AWS account. For instructions, see [Creating Your First IAM User and Administrators Group](#) in the *IAM User Guide*.
2. Sign in to the AWS Management Console using a special URL. For more information, see [How Users Sign In to Your Account](#) in the *IAM User Guide*.

For more information about IAM, see the following:

- [AWS Identity and Access Management \(IAM\)](#)
- [Getting started](#)
- [IAM User Guide](#)

Next step

[Step 2: Set up the AWS Command Line Interface \(AWS CLI\) \(p. 58\)](#)

Step 2: Set up the AWS Command Line Interface (AWS CLI)

You don't need the AWS CLI to perform the steps in the Getting Started exercises. However, some of the other exercises in this guide do require it. If you prefer, you can skip this step and set up the AWS CLI later.

To set up the AWS CLI

1. Download and configure the AWS CLI. For instructions, see the following topics in the *AWS Command Line Interface User Guide*:
 - [Getting set up with the AWS Command Line Interface](#)
 - [Configuring the AWS Command Line Interface](#)
2. In the AWS CLI config file, add a named profile for the administrator user:

```
[profile adminuser]
aws_access_key_id = adminuser access key ID
aws_secret_access_key = adminuser secret access key
region = aws-region
```

You use this profile when executing the AWS CLI commands. For more information about named profiles, see [Named Profiles](#) in the *AWS Command Line Interface User Guide*. For a list of AWS Regions, see [Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

3. Verify the setup by typing the following help command at the command prompt:

```
aws help
```

Next step

[Step 3: Getting started using the console \(p. 59\)](#)

Step 3: Getting started using the console

The easiest way to get started with Amazon Transcribe is to submit a job using the console to transcribe an audio file. If you haven't reviewed the concepts and terminology in [How Amazon Transcribe works \(p. 3\)](#), we recommend that you do that before proceeding.

Topics

- [Create a transcription job \(p. 59\)](#)
- [View a transcription job \(p. 60\)](#)

Create a transcription job

Use the [Amazon Transcribe console](#) to create a transcription job for your audio files.

1. Provide the following information:

- **Transcription job name**—A name for the job. The name must be unique within your AWS account.
- **Amazon S3 input URL**—The Amazon S3 location of your input audio file. The location must be in the same region as the endpoint that you are calling.
- **Language**—Choose the language of your input file.
- **Format**—The format of the audio file. For best results you should use a lossless format such as FLAC or WAV with PCM 16-bit encoding.
- **Media sampling rate (Hz)**—Optional. The bit sampling rate of the audio file. Amazon Transcribe accepts sample rates between 8,000 Hz and 48,000 Hz. For best results, you should use 8,000 Hz for low-fidelity audio and 16,000 for high-fidelity audio.

The following shows the **Create Transcription Job** filled out for a sample job.

Create transcription job

Input Info

Name

 The name can be up to 200 characters long. Valid characters are a-z, A-Z, 0-9 and – (hyphen).

S3 input URL

 Type or paste the URL of your input audio file in S3.
 Valid formats for audio files are mp3, mp4, wav, and flac.

Language
 Choose the language of the input audio.
▼

Format
 Choose the format of your audio file.
▼
 Valid formats for the audio are mp3, mp4, wav and flac.

Audio sampling rate (Hz)
 Type the sampling rate of the input audio file.

 Must be an integer between 8000 and 48000

Apply custom vocabulary Info
 A custom vocabulary improves the accuracy of recognizing words, phrases, and commands.

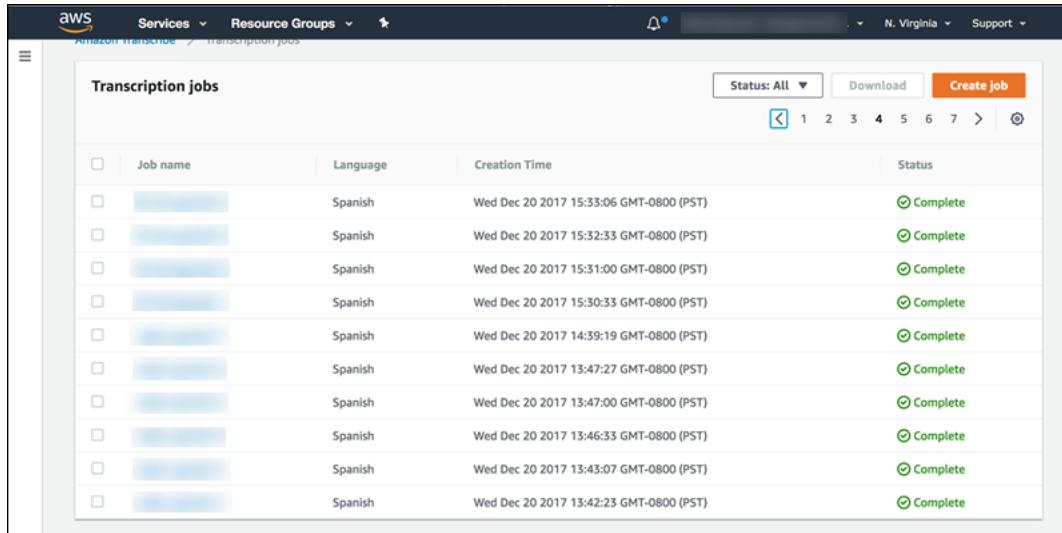
Speaker identification Info
 Identifies speakers in the input audio file.
 Disabled
 Enabled

Cancel Create

- Choose **Create** to submit the job for processing.

View a transcription job

Completed transcription jobs are displayed in a list that contains a brief description of the job. The **Availability** column shows the remaining time that the job results are kept on the server. Jobs are kept for 90 days and then deleted from the system.



The screenshot shows the AWS Lambda service dashboard with the 'Transcription jobs' section selected. The table lists ten completed transcription jobs, each with a blue placeholder icon for the job name, 'Spanish' as the language, and a creation timestamp. All jobs are marked as 'Complete' with a green checkmark icon.

<input type="checkbox"/>	Job name	Language	Creation Time	Status
<input type="checkbox"/>	[REDACTED]	Spanish	Wed Dec 20 2017 15:33:06 GMT-0800 (PST)	✓ Complete
<input type="checkbox"/>	[REDACTED]	Spanish	Wed Dec 20 2017 15:32:33 GMT-0800 (PST)	✓ Complete
<input type="checkbox"/>	[REDACTED]	Spanish	Wed Dec 20 2017 15:31:00 GMT-0800 (PST)	✓ Complete
<input type="checkbox"/>	[REDACTED]	Spanish	Wed Dec 20 2017 15:30:33 GMT-0800 (PST)	✓ Complete
<input type="checkbox"/>	[REDACTED]	Spanish	Wed Dec 20 2017 14:39:19 GMT-0800 (PST)	✓ Complete
<input type="checkbox"/>	[REDACTED]	Spanish	Wed Dec 20 2017 13:47:27 GMT-0800 (PST)	✓ Complete
<input type="checkbox"/>	[REDACTED]	Spanish	Wed Dec 20 2017 13:47:00 GMT-0800 (PST)	✓ Complete
<input type="checkbox"/>	[REDACTED]	Spanish	Wed Dec 20 2017 13:46:33 GMT-0800 (PST)	✓ Complete
<input type="checkbox"/>	[REDACTED]	Spanish	Wed Dec 20 2017 13:43:07 GMT-0800 (PST)	✓ Complete
<input type="checkbox"/>	[REDACTED]	Spanish	Wed Dec 20 2017 13:42:23 GMT-0800 (PST)	✓ Complete

Choose a job in the list to see information about the job.

The information page about the transcription job has three sections. The **Detail** section provides details about the transcription job, including the name, information about when the job is deleted from the server, and the input and output URLs. Use the output URL to download the output from your transcription job.

The **Output** section contains the transcription of the audio submitted to Amazon Transcribe. You can download the transcription by choosing the **Download transcription** button.

Amazon Transcribe > Transcription jobs > ES-US-spanish-4

The **Code samples** section contains the JSON input for the [StartTranscriptionJob](#) (p. 376) API and the output from the [GetTranscriptionJob](#) (p. 329) API.

Next step

[Step 4: Getting started using the API \(p. 62\)](#)

Step 4: Getting started using the API

This section contains examples the demonstrate using the Amazon Transcribe API. You can use these samples to learn about the API or as building blocks in your own applications.

Topics

- [AWS Command Line Interface \(p. 62\)](#)
- [AWS SDK for Python \(Boto\) \(p. 64\)](#)

AWS Command Line Interface

In the following exercise, you use the AWS Command Line Interface (AWS CLI) to transcribe speech into text. To complete this exercise, you need to:

- Have a text editor.
- Be familiar with the AWS CLI. For more information, see [Step 2: Set up the AWS Command Line Interface \(AWS CLI\) \(p. 58\)](#).
- Have a speech file in WAV or MP4 format that is stored in an S3 bucket that has the proper permissions. For more information about the permissions needed for Amazon Transcribe, see [Permissions required for IAM user roles \(p. 177\)](#).

To transcribe text, you have to provide the input parameters in a JSON file.

To transcribe text

1. Copy your input speech to an S3 bucket. The location must be in the same region as the endpoint that you are calling.
2. Create a JSON file named `test-start-command.json` that contains the input parameters for the [StartTranscriptionJob \(p. 376\)](#) API. Enter a unique name for your transcription job under "TranscriptionJobName".

```
{  
    "TranscriptionJobName": "unique job name",  
    "LanguageCode": "en-US",  
    "MediaFormat": "wav",  
    "Media": {  
        "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file"  
    }  
}
```

3. In the AWS CLI, run the following command. The example is formatted for Unix, Linux, and macOS. For Windows, replace the backslash (\) Unix continuation character at the end of each line with a caret (^).

```
aws transcribe start-transcription-job \  
    --region region \  
    --cli-input-json file://filepath/example-start-command.json
```

Amazon Transcribe responds with the following:

```
{  
    "TranscriptionJob": {  
        "TranscriptionJobName": "unique job name",  
        "LanguageCode": "en-US",  
        "TranscriptionJobStatus": "IN_PROGRESS",  
        "Media": {  
            "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file"  
        },  
        "CreationTime": timestamp,  
        "MediaFormat": "wav"  
    }  
}
```

To list transcription jobs

- Run the following command:

```
aws transcribe list-transcription-jobs \  
    --region region \  
    --status IN_PROGRESS
```

Amazon Transcribe responds with the following:

```
{  
    "Status": "IN_PROGRESS",  
    "TranscriptionJobSummaries": [  
        {  
            "TranscriptionJobName": "unique job name",  
            "LanguageCode": "en-US",  
            "CreationTime": timestamp,  
            "TranscriptionJobStatus": "IN_PROGRESS"  
        }  
    ]  
}
```

To get the results of a transcription job

- When the job has the status COMPLETED, get the results of the job. Type the following command:

```
aws transcribe get-transcription-job \  
    --region region \  
    --transcription-job-name "unique job name"
```

Amazon Transcribe responds with the following:

```
{  
    "TranscriptionJob": {  
        "TranscriptionJobName": "unique job name",  
        "LanguageCode": "en-US",  
        "TranscriptionJobStatus": "COMPLETED",  
        "Media": {  
            "MediaFileUri": "input URI"  
        },  
        "CreationTime": timestamp,  
        "CompletionTime": timestamp,  
        "Transcript": {  
            "TranscriptText": "The quick brown fox jumps over the lazy dog.",  
            "LanguageCode": "en-US",  
            "TranscriptType": "TRANSCRIPT",  
            "Offset": 0.0, "Length": 1.0  
        }  
    }  
}
```

```
        "TranscriptFileUri": "output URI"  
    }  
}  
}
```

2. Use the output URI to get the transcribed text from the audio file. The following is the output from transcribing a short audio clip:

```
{  
    "jobName": "job ID",  
    "accountId": "account ID",  
    "results": {  
        "transcripts": [  
            {  
                "transcript": "that's no answer"  
            }  
        ],  
        "items": [  
            {  
                "start_time": "0.180",  
                "end_time": "0.470",  
                "alternatives": [  
                    {  
                        "confidence": 0.84,  
                        "content": "that's"  
                    },  
                    {  
                        "confidence": 0.99,  
                        "content": "no"  
                    },  
                    {  
                        "confidence": 0.874,  
                        "content": "answer"  
                    }  
                ],  
                "type": "pronunciation"  
            },  
            {  
                "start_time": "0.470",  
                "end_time": "0.710",  
                "alternatives": [  
                    {  
                        "confidence": 0.99,  
                        "content": "no"  
                    },  
                    {  
                        "confidence": 0.874,  
                        "content": "answer"  
                    }  
                ],  
                "type": "pronunciation"  
            },  
            {  
                "start_time": "0.710",  
                "end_time": "1.080",  
                "alternatives": [  
                    {  
                        "confidence": 0.874,  
                        "content": "answer"  
                    }  
                ],  
                "type": "pronunciation"  
            }  
        ]  
    },  
    "status": "COMPLETED"  
}
```

AWS SDK for Python (Boto)

In this exercise you create script that uses the SDK for Python to transcribe speech into text. To complete this exercise, you need to:

- Install the AWS CLI. For more information, see [Step 2: Set up the AWS Command Line Interface \(AWS CLI\) \(p. 58\)](#). This installs the AWS SDK for Python (Boto).
- Have a speech file in .WAV or .MP4 format that is stored in an S3 bucket that has the proper permissions. For more information about the permissions needed for Amazon Transcribe, see [Permissions required for IAM user roles \(p. 177\)](#). The location must be in the same region as the endpoint that you are calling. This example assumes that the file is in an Amazon S3 bucket named test-transcribe and that the file name is answer2.wav.

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "job name"
job_uri = "s3://DOC-EXAMPLE-BUCKET1/key-prefix/file.file-extension"
transcribe.start_transcription_job(
    TranscriptionJobName=job_name,
    Media={'MediaFileUri': job_uri},
    MediaFormat='wav',
    LanguageCode='en-US'
)
while True:
    status = transcribe.get_transcription_job(TranscriptionJobName=job_name)
    if status['TranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED', 'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)
```

When the transcription job is complete, the result links to an Amazon S3 presigned URL that contains the transcription in JSON format:

```
{
  "jobName": "job ID",
  "accountId": "account ID",
  "results": {
    "transcripts": [
      {
        "transcript": " that's no answer"
      }
    ],
    "items": [
      {
        "start_time": "0.180",
        "end_time": "0.470",
        "alternatives": [
          {
            "confidence": 0.84,
            "word": "that's"
          }
        ]
      },
      {
        "start_time": "0.470",
        "end_time": "0.710",
        "alternatives": [
          {
            "confidence": 0.99,
            "word": "no"
          }
        ]
      },
    ],
  }
}
```

```
{
    "start_time": "0.710",
    "end_time": "1.080",
    "alternatives": [
        {
            "confidence": 0.87,
            "word": "answer"
        }
    ]
},
"status": "COMPLETED"
}
```

Step 5: Getting started with streaming audio

The following example is a Java program that transcribes streaming audio. The input comes from your computer's microphone or a file upload and the output is presented on your computer's standard output.

To run this example, you need the following:

- You must use the [AWS SDK for Java 2.x](#)
- Clients must use Java 1.8 to be compatible with the AWS SDK for Java 2.x.

```
public class TranscribeStreamingDemoApp {
    private static final Region REGION = Region.US_EAST_1;
    private static TranscribeStreamingAsyncClient client;

    public static void main(String args[]) throws URISyntaxException, ExecutionException,
    InterruptedException, LineUnavailableException {

        client = TranscribeStreamingAsyncClient.builder()
            .credentialsProvider(getCredentials())
            .region(REGION)
            .build();

        CompletableFuture<Void> result =
        client.startStreamTranscription(getRequest(16_000),
            new AudioStreamPublisher(getStreamFromMic()),
            getResponseHandler());

        result.get();
        client.close();
    }

    private static InputStream getStreamFromMic() throws LineUnavailableException {

        // Signed PCM AudioFormat with 16kHz, 16 bit sample size, mono
        int sampleRate = 16000;
        AudioFormat format = new AudioFormat(sampleRate, 16, 1, true, false);
        DataLine.Info info = new DataLine.Info(TargetDataLine.class, format);

        if (!AudioSystem.isLineSupported(info)) {
            System.out.println("Line not supported");
            System.exit(0);
        }

        TargetDataLine line = (TargetDataLine) AudioSystem.getLine(info);
        line.open(format);
    }
}
```

```

        line.start();

        InputStream audioStream = new AudioInputStream(line);
        return audioStream;
    }

    private static AwsCredentialsProvider getCredentials() {
        return DefaultCredentialsProvider.create();
    }

    private static StartStreamTranscriptionRequest getRequest(Integer mediaSampleRateHertz)
    {
        return StartStreamTranscriptionRequest.builder()
            .languageCode(LanguageCode.EN_US.toString())
            .mediaEncoding(MediaEncoding.PCM)
            .mediaSampleRateHertz(mediaSampleRateHertz)
            .build();
    }

    private static StartStreamTranscriptionResponseHandler getResponseHandler() {
        return StartStreamTranscriptionResponseHandler.builder()
            .onResponse(r -> {
                System.out.println("Received Initial response");
            })
            .onError(e -> {
                System.out.println(e.getMessage());
                StringWriter sw = new StringWriter();
                e.printStackTrace(new PrintWriter(sw));
                System.out.println("Error Occurred: " + sw.toString());
            })
            .onComplete(() -> {
                System.out.println("== All records stream successfully ==");
            })
            .subscriber(event -> {
                List<Result> results = ((TranscriptEvent)
event).transcript().results();
                if (results.size() > 0) {
                    if (!results.get(0).alternatives().get(0).transcript().isEmpty()) {

System.out.println(results.get(0).alternatives().get(0).transcript());
                }
            }
        })
        .build();
    }

    private InputStream getStreamFromFile(String audioFileName) {
        try {
            File inputFile = new
File(getClass().getClassLoader().getResource(audioFileName).getFile());
            InputStream audioStream = new FileInputStream(inputFile);
            return audioStream;
        } catch (FileNotFoundException e) {
            throw new RuntimeException(e);
        }
    }

    private static class AudioStreamPublisher implements Publisher<AudioStream> {
        private final InputStream inputStream;
        private static Subscription currentSubscription;

        private AudioStreamPublisher(InputStream inputStream) {
            this.inputStream = inputStream;
        }
    }
}

```

```

@Override
public void subscribe(Subscriber<? super AudioStream> s) {

    if (this.currentSubscription == null) {
        this.currentSubscription = new SubscriptionImpl(s, inputStream);
    } else {
        this.currentSubscription.cancel();
        this.currentSubscription = new SubscriptionImpl(s, inputStream);
    }
    s.onSubscribe(currentSubscription);
}

public static class SubscriptionImpl implements Subscription {
    private static final int CHUNK_SIZE_IN_BYTES = 1024 * 1;
    private final Subscriber<? super AudioStream> subscriber;
    private final InputStream inputStream;
    private final ExecutorService executor = Executors.newFixedThreadPool(1);
    private AtomicLong demand = new AtomicLong(0);

    SubscriptionImpl(Subscriber<? super AudioStream> s, InputStream inputStream) {
        this.subscriber = s;
        this.inputStream = inputStream;
    }

    @Override
    public void request(long n) {
        if (n <= 0) {
            subscriber.onError(new IllegalArgumentException("Demand must be
positive"));
        }
        demand.getAndAdd(n);

        executor.submit(() -> {
            try {
                do {
                    ByteBuffer audioBuffer = getNextEvent();
                    if (audioBuffer.remaining() > 0) {
                        AudioEvent audioEvent = audioEventFromBuffer(audioBuffer);
                        subscriber.onNext(audioEvent);
                    } else {
                        subscriber.onComplete();
                        break;
                    }
                } while (demand.decrementAndGet() > 0);
            } catch (Exception e) {
                subscriber.onError(e);
            }
        });
    }

    @Override
    public void cancel() {
        executor.shutdown();
    }

    private ByteBuffer getNextEvent() {
        ByteBuffer audioBuffer = null;
        byte[] audioBytes = new byte[CHUNK_SIZE_IN_BYTES];

        int len = 0;
        try {
            len = inputStream.read(audioBytes);

            if (len <= 0) {

```

```
        audioBuffer = ByteBuffer.allocate(0);
    } else {
        audioBuffer = ByteBuffer.wrap(audioBytes, 0, len);
    }
} catch (IOException e) {
    throw new UncheckedIOException(e);
}

return audioBuffer;
}

private AudioEvent audioEventFromBuffer(ByteBuffer bb) {
    return AudioEvent.builder()
        .audioChunk(SdkBytes.fromByteBuffer(bb))
        .build();
}
}
```

Transcribing streaming audio

Use Amazon Transcribe streaming transcription to send an audio stream and receive a stream of text in real time. You can use this text stream to add real-time speech-to-text capability to your applications.

Streaming transcription takes a stream of your audio data and transcribes it in real time. Streaming uses HTTP/2 or WebSocket streams so that the results of the transcription are returned to your application while you send more audio to Amazon Transcribe. Use streaming transcription when you want to make the results of live audio transcription available immediately, or when you have an audio file that you want to process as it is transcribed.

To see which languages support streaming audio transcription, see [Supported languages and language-specific features \(p. 3\)](#).

For a streaming transcription using:

- HTTP/2 – Use the [StartStreamTranscription \(p. 408\)](#) API to start a stream. When you use this API, the client handles retrying the connection when there are transient problems on the network.
- WebSocket – Create your own client.
- [Amazon Transcribe console](#) – Speak directly into a computer microphone. You can use the console as a preview to see how your transcription results are returned to your application when using the API.

Note

Streaming transcription is not supported in all languages. See [Supported languages and language-specific features \(p. 3\)](#) for details.

You can use Amazon Transcribe streaming transcription for a variety of purposes. For example:

- Streaming transcriptions can generate real-time subtitles for live broadcast media.
- Lawyers can make real-time annotations on top of streaming transcriptions during depositions.
- Video game chat can be transcribed in real time so that hosts can moderate content or run real-time analysis.
- Streaming transcriptions can provide assistance to the hearing impaired.

Streaming transcription takes a stream of your audio data and transcribes it in real time. The transcription is returned to your application in a stream of transcription events.

Amazon Transcribe breaks up the incoming audio stream based on natural speech segments, such as a change in speaker or a pause in the audio. The transcription is returned progressively to your application, with each response containing more transcribed speech until the entire segment is transcribed.

In the following example, each line is a partial result transcription output of an audio segment that is being streamed.

```
The  
The ad.  
The and  
The Amazon.  
The Amazon is
```

```
The Amazon is the
The Amazon is the law.
The Amazon is the lar.
The Amazon is the large
The Amazon is the largest
The Amazon is the largest ray
The Amazon is the largest rain
The Amazon is the largest rain for
The Amazon is the largest rainforest.
The Amazon is the largest rainforest on the planet
```

Each **Result** object in the response contains a field called `IsPartial`. The value indicates whether the response is a partial response that contains the transcription results so far, or whether it is a complete transcription of the audio segment.

Each **Result** object also contains the start and end time of the transcribed audio segment from the stream. You can use these values to, for example, synchronize the transcription with the video.

The following example is a partial transcription response.

```
{
    "TranscriptResultStream": {
        "TranscriptEvent": {
            "Transcript": {
                "Results": [
                    {
                        "Alternatives": [
                            {
                                "Items": [
                                    {
                                        "Content": "The",
                                        "EndTime": 0.3799375,
                                        "StartTime": 0.0299375,
                                        "Type": "pronunciation",
                                        "VocabularyFilterMatch": false
                                    },
                                    {
                                        "Content": "Amazon",
                                        "EndTime": 0.5899375,
                                        "StartTime": 0.3899375,
                                        "Type": "pronunciation",
                                        "VocabularyFilterMatch": false
                                    },
                                    {
                                        "Content": "is",
                                        "EndTime": 0.7899375,
                                        "StartTime": 0.5999375,
                                        "Type": "pronunciation",
                                        "VocabularyFilterMatch": false
                                    },
                                    {
                                        "Content": "the",
                                        "EndTime": 0.9199375,
                                        "StartTime": 0.7999375,
                                        "Type": "pronunciation",
                                        "VocabularyFilterMatch": false
                                    },
                                    {
                                        "Content": "largest",
                                        "EndTime": 1.0199375,
                                        "StartTime": 0.9299375,
                                        "Type": "pronunciation",
                                        "VocabularyFilterMatch": false
                                    }
                                ]
                            }
                        ]
                    }
                ]
            }
        }
    }
}
```

```

        ],
        "Transcript": "The Amazon is the largest"
    }
],
"EndTime": 1.02,
"IsPartial": true,
"ResultId": "2db76dc8-d728-11e8-9f8b-f2801f1b9fd1",
"StartTime": 0.0199375
}
]
}
}
}
}

```

The following example shows the transcription results for a fully transcribed speech segment.

```
{
    "Transcript": {
        "Results": [
            {
                "Alternatives": [
                    {
                        "Items": [
                            {
                                "Confidence": 1,
                                "Content": "The",
                                "EndTime": 2.58,
                                "StartTime": 2.33,
                                "Type": "pronunciation",
                                "VocabularyFilterMatch": false
                            },
                            {
                                "Confidence": 1,
                                "Content": "Amazon",
                                "EndTime": 3.12,
                                "StartTime": 2.59,
                                "Type": "pronunciation",
                                "VocabularyFilterMatch": false
                            },
                            {
                                "Confidence": 1,
                                "Content": "is",
                                "EndTime": 3.25,
                                "StartTime": 3.13,
                                "Type": "pronunciation",
                                "VocabularyFilterMatch": false
                            },
                            {
                                "Confidence": 1,
                                "Content": "the",
                                "EndTime": 3.39,
                                "StartTime": 3.26,
                                "Type": "pronunciation",
                                "VocabularyFilterMatch": false
                            },
                            {
                                "Confidence": 1,
                                "Content": "largest",
                                "EndTime": 3.93,
                                "StartTime": 3.4,
                                "Type": "pronunciation",
                                "VocabularyFilterMatch": false
                            },
                            {

```

```
        "Confidence": 0.99,
        "Content": "rainforest",
        "EndTime": 4.8,
        "StartTime": 3.95,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Confidence": 1,
        "Content": "on",
        "EndTime": 5.01,
        "StartTime": 4.82,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Confidence": 1,
        "Content": "the",
        "EndTime": 5.11,
        "StartTime": 5.02,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Confidence": 1,
        "Content": "planet",
        "EndTime": 5.64,
        "StartTime": 5.12,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Confidence": 0.89,
        "Content": "covering",
        "EndTime": 6.24,
        "StartTime": 5.72,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Confidence": 1,
        "Content": "over",
        "EndTime": 6.46,
        "StartTime": 6.25,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Confidence": 1,
        "Content": "seven",
        "EndTime": 6.82,
        "StartTime": 6.47,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Confidence": 1,
        "Content": "million",
        "EndTime": 7.15,
        "StartTime": 6.83,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Confidence": 1,
        "Content": "kilometers",
```

```
        "EndTime": 7.92,
        "StartTime": 7.16,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Content": ".",
        "EndTime": 7.92,
        "StartTime": 7.92,
        "Type": "punctuation",
        "VocabularyFilterMatch": false
    }
],
"Transcript": "The Amazon is the largest rainforest on the planet
covering over seven million kilometers."
}
],
"EndTime": 7.92,
"IsPartial": false,
"ResultId": "a37114fa-5288-438d-afdf-833cf1ea55c",
"StartTime": 2.33
}
]
}
```

Each word, phrase, or punctuation mark in the transcription output is an *item*. Each word or phrase has a *confidence* score. The confidence score is a value between 0 and 1 that indicates how confident Amazon Transcribe is that it correctly transcribed the item. A confidence score with a larger value indicates that Amazon Transcribe is more confident that it transcribed the item correctly.

Topics

- [Event stream encoding \(p. 74\)](#)
- [Using Amazon Transcribe streaming with WebSockets \(p. 76\)](#)
- [Using Amazon Transcribe streaming With HTTP/2 \(p. 83\)](#)
- [Partial result stabilization \(p. 95\)](#)

Event stream encoding

Event stream encoding provides bidirectional communication using messages between a client and a server. Data frames sent to the Amazon Transcribe streaming service are encoded in this format. The response from Amazon Transcribe also uses this encoding.

Each message consists of two sections: the prelude and the data. The prelude consists of:

1. The total byte length of the message
2. The combined byte length of all of the headers

The data section consists of:

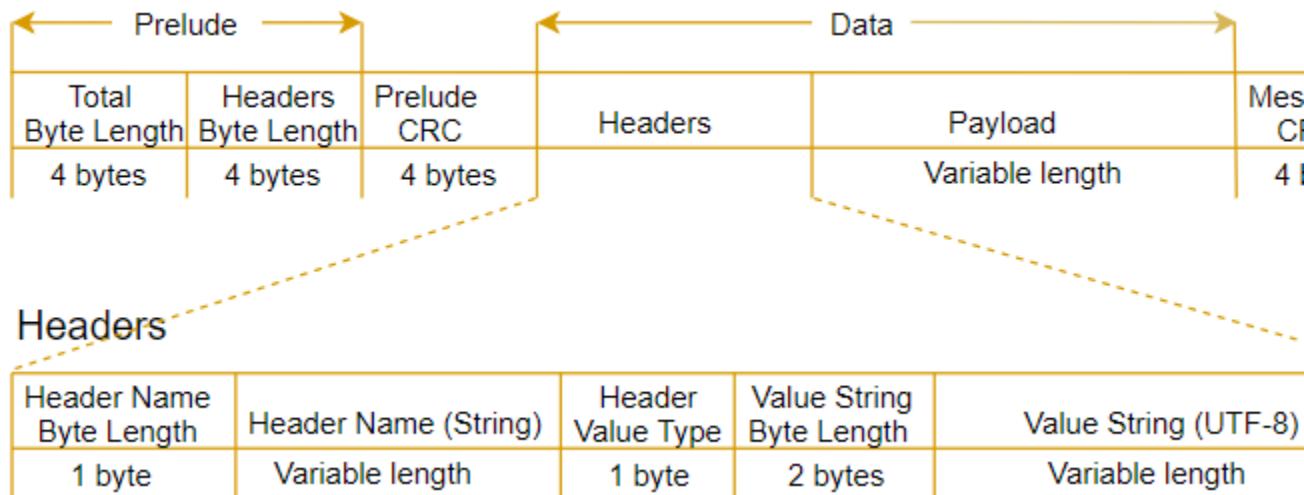
1. The headers
2. A payload

Each section ends with a 4-byte big-endian integer CRC checksum. The message CRC checksum is for both the prelude section and the data section. Amazon Transcribe uses CRC32 (often referred to as GZIP

CRC32) to calculate both CRCs. For more information about CRC32, see [GZIP file format specification version 4.3](#).

Total message overhead, including the prelude and both checksums, is 16 bytes.

The following diagram shows the components that make up a message and a header. There are multiple headers per message.



Each message contains the following components:

- **Prelude:** Always a fixed size of 8 bytes, two fields of 4 bytes each.
 - *First 4 bytes:* The total byte-length. This is the big-endian integer byte-length of the entire message, including the 4-byte length field itself.
 - *Second 4 bytes:* The headers byte-length. This is the big-endian integer byte-length of the headers portion of the message, excluding the headers length field itself.
- **Prelude CRC:** The 4-byte CRC checksum for the prelude portion of the message, excluding the CRC itself. The prelude has a separate CRC from the message CRC to ensure that Amazon Transcribe can detect corrupted byte-length information immediately without causing errors such as buffer overruns.
- **Headers:** Metadata annotating the message, such as the message type, content type, and so on. Messages have multiple headers. Headers are key-value pairs where the key is a UTF-8 string. Headers can appear in any order in the headers portion of the message and any given header can appear only once. For the required header types, see the following sections.
- **Payload:** The audio content to be transcribed.
- **Message CRC:** The 4-byte CRC checksum from the start of the message to the start of the checksum. That is, everything in the message except the CRC itself.

Each header contains the following components. There are multiple headers per frame.

- **Header name byte-length:** The byte-length of the header name.
- **Header name:** The name of the header indicating the header type. For valid values, see the following frame descriptions.
- **Header value type:** An enumeration indicating the header value.

The following shows the possible values for the header and what they indicate.

- 0 – TRUE
- 1 – FALSE
- 2 – BYTE

- 3 – SHORT
 - 4 – INTEGER
 - 5 – LONG
 - 6 – BYTE ARRAY
 - 7 – STRING
 - 8 – TIMESTAMP
 - 9 – UUID
- **Value string byte length:** The byte-length of the header value string.
 - **Header value:** The value of the header string. Valid values for this field depend on the type of header. For valid values, see the following frame descriptions.

Using Amazon Transcribe streaming with WebSockets

When you use the [WebSocket protocol](#) to stream audio, Amazon Transcribe transcribes the stream in real time. You encode the audio with event stream encoding, Amazon Transcribe responds with a JSON structure that is also encoded using event stream encoding. For more information, see [Event stream encoding \(p. 74\)](#). You can use the information in this section to create applications using the WebSocket library of your choice.

Topics

- [Adding a policy for WebSocket requests to your IAM role \(p. 76\)](#)
- [Creating a pre-signed URL \(p. 77\)](#)
- [Handling the WebSocket upgrade response \(p. 81\)](#)
- [Making a WebSocket streaming request \(p. 81\)](#)
- [Handling a WebSocket streaming response \(p. 82\)](#)
- [Handling WebSocket streaming errors \(p. 82\)](#)

Adding a policy for WebSocket requests to your IAM role

To use the WebSocket protocol to call Amazon Transcribe, you need to attach the following policy to the AWS Identity and Access Management (IAM) role that makes the request. See [Adding IAM policies](#) for more information on how to do this.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "transcribestreaming",  
            "Effect": "Allow",  
            "Action": "transcribe:StartStreamTranscriptionWebSocket",  
            "Resource": "*"  
        }  
    ]  
}
```

Creating a pre-signed URL

Construct a URL for your WebSocket request that contains the information needed to set up communication between your application and Amazon Transcribe. WebSocket streaming uses the Amazon Signature Version 4 process for signing requests. Signing the request helps to verify the identity of the requester and to protect your audio data in transit. It also protects against potential replay attacks. For more information about Signature Version 4, see [Signing AWS API Requests](#) in the *Amazon Web Services General Reference*.

The URL has the following format. Line breaks have been added for readability.

```
GET wss://transcribestreaming.region.amazonaws.com:8443/stream-transcription-websocket
?language-code=languageCode
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=Signature Version 4 credential scope
&X-Amz-Date=date
&X-Amz-Expires=time in seconds until expiration
&X-Amz-Security-Token=security-token
&X-Amz-Signature=Signature Version 4 signature
&X-Amz-SignedHeaders=host
&media-encoding=mediaEncoding
&sample-rate=mediaSampleRateHertz
&session-id=sessionId
&vocabulary-name=vocabularyName
```

Use the following values for the URL parameters:

- **language-code** – The language code for the input audio. Valid values are en-AU, en-GB, en-US, es-US, fr-CA, fr-FR, de-DE, ja-JP, ko-KR, pt-BR, zh-CN and it-IT.
- **media-encoding** – The encoding used for the input audio. Valid values are pcm, ogg-opus, and flac.
- **sample-rate** – The sample rate of the input audio in Hertz. We suggest that you use 8,000 Hz for low-quality audio and 16,000 Hz for high-quality audio. The sample rate must match the sample rate in the audio file.
- **sessionId** – Optional. An identifier for the transcription session. If you don't provide a session ID, Amazon Transcribe generates one for you and returns it in the response.
- **vocabulary-name** – Optional. The name of the vocabulary to use when processing the transcription job, if any.

The remaining parameters are Signature Version 4 parameters:

- **X-Amz-Algorithm** – The algorithm you're using in the signing process. The only valid value is AWS4-HMAC-SHA256.
- **X-Amz-Credential** – A string separated by slashes ("/") that is formed by concatenating your access key ID and your credential scope components. Credential scope includes the date in YYYYMMDD format, the AWS Region, the service name, and a special termination string (aws4_request).
- **X-Amz-Date** – The date and time that the signature was created. Generate the date and time by following the instructions in [Handling Dates in Signature Version 4](#) in the *Amazon Web Services General Reference*.
- **X-Amz-Expires** – The length of time in seconds until the credentials expire. The maximum value is 300 seconds (5 minutes).
- **X-Amz-Security-Token** – Optional. A Signature Version 4 token for temporary credentials. If you specify this parameter, include it in the canonical request. For more information, see [Requesting Temporary Security Credentials](#) in the *AWS Identity and Access Management User Guide*.
- **X-Amz-Signature** – The Signature Version 4 signature that you generated for the request.

- **X-Amz-SignedHeaders** – The headers that are signed when creating the signature for the request. The only valid value is host.

To construct the URL for the request and create the Signature Version 4 signature, use the following steps. The examples are in pseudocode.

Task 1: Create a canonical request

Create a string that includes information from your request in a standardized format. This ensures that when AWS receives the request, it can calculate the same signature that you calculate in Task 3. For more information, see [Create a Canonical Request for Signature Version 4](#) in the *Amazon Web Services General Reference*.

1. Define variables for the request in your application.

```
# HTTP verb
method = "GET"
# Service name
service = "transcribe"
# AWS Region
region = "AWS Region"
# Amazon Transcribe streaming endpoint
endpoint = "wss://transcribestreaming.region.amazonaws.com:8443"
# Host
host = "transcribestreaming.region.amazonaws.com:8443"
# Date and time of request
amz-date = YYYYMMDD'T'HHMMSS'Z'
# Date without time for credential scope
datestamp = YYYYMMDD
```

2. Create a canonical URI. The canonical URI is the part of the URI between the domain and the query string.

```
canonical_uri = "/stream-transcription-websocket"
```

3. Create the canonical headers and signed headers. Note the trailing \n in the canonical headers.

- Append the lowercase header name followed by a colon.
- Append a comma-separated list of values for that header. Do not sort the values in headers that have multiple values.
- Append a new line (\n).

```
canonical_headers = "host:" + host + "\n"
signed_headers = "host"
```

4. Match the algorithm to the hashing algorithm. You must use SHA-256.

```
algorithm = "AWS4-HMAC-SHA256"
```

5. Create the credential scope, which scopes the derived key to the date, Region, and service to which the request is made.

```
credential_scope = datestamp + "/" + region + "/" + service + "/" + "aws4_request"
```

6. Create the canonical query string. Query string values must be URI-encoded and sorted by name.

- Sort the parameter names by character code point in ascending order. Parameters with duplicate names should be sorted by value. For example, a parameter name that begins with the uppercase letter F precedes a parameter name that begins with a lowercase letter b.
- Do not URI-encode any of the unreserved characters that [RFC 3986](#) defines: A-Z, a-z, 0-9, hyphen (-), underscore (_), period (.), and tilde (~).
- Percent-encode all other characters with %XY, where X and Y are hexadecimal characters (0-9 and uppercase A-F). For example, the space character must be encoded as %20 (not using '+', as some encoding schemes do) and extended UTF-8 characters must be in the form %XY%ZA%BC.
- Double-encode any equals (=) characters in parameter values.

```
canonical_querystring = "X-Amz-Algorithm=" + algorithm
canonical_querystring += "&X-Amz-Credential=" + URI-encode(access_key + "/" +
    credential_scope)
canonical_querystring += "&X-Amz-Date=" + amz_date
canonical_querystring += "&X-Amz-Expires=300"
canonical_querystring += "&X-Amz-Security-Token=" + token
canonical_querystring += "&X-Amz-SignedHeaders=" + signed_headers
canonical_querystring += "&language-code=en-US&media-encoding=pcm&sample-rate=16000"
```

7. Create a hash of the payload. For a GET request, the payload is an empty string.

```
payload_hash = HashSHA256("").Encode("utf-8").HexDigest()
```

8. Combine all of the elements to create the canonical request.

```
canonical_request = method + '\n'
    + canonical_uri + '\n'
    + canonical_querystring + '\n'
    + canonical_headers + '\n'
    + signed_headers + '\n'
    + payload_hash
```

Task 2: Create the string to sign

The string to sign contains meta information about your request. You use the string to sign in the next step when you calculate the request signature. For more information, see [Create a String to Sign for Signature Version 4](#) in the *Amazon Web Services General Reference*.

- Create the string.

```
string_to_sign=algorithm + "\n"
    + amz_date + "\n"
    + credential_scope + "\n"
    + HashSHA256(canonical_request.Encode("utf-8")).HexDigest()
```

Task 3: Calculate the signature

You derive a signing key from your AWS secret access key. For a greater degree of protection, the derived key is specific to the date, service, and AWS Region. You use the derived key to sign the request. For more information, see [Calculate the Signature for AWS Signature Version 4](#) in the *Amazon Web Services General Reference*.

The code assumes that you have implemented the `GetSignatureKey` function to derive a signing key. For more information and example functions, see [Examples of How to Derive a Signing Key for Signature Version 4](#) in the *Amazon Web Services General Reference*.

The function `HMAC(key, data)` represents an HMAC-SHA256 function that returns the results in binary format.

- Create the signing key and sign the string to sign.

```
#Create the signing key
signing_key = GetSignatureKey(secret_key, datestamp, region, service)

# Sign the string_to_sign using the signing key
signature = HMAC.new(signing_key, (string_to_sign).Encode("utf-8"), Sha256()).HexDigest
```

Task 4: Add signing information to the request and create the request URL

After you calculate the signature, add it to the query string. For more information, see [Add the Signature to the Request](#) in the *Amazon Web Services General Reference*.

1. Add the authentication information to the query string.

```
canonical_querystring += "&X-Amz-Signature=" + signature
```

2. Create the URL for the request.

```
request_url = endpoint + canonical_uri + "?" + canonical_querystring
```

You use the request URL with your WebSocket library to make the request to the Amazon Transcribe service.

Including WebSocket request headers

The request to Amazon Transcribe must include the following headers. Typically these headers are managed by your WebSocket client library.

```
Host: transcribestreaming.region.amazonaws.com:8443
Connection: Upgrade
Upgrade: websocket
Origin: request source
Sec-WebSocket-Version: 13
Sec-WebSocket-Key: random key
```

Use the following values for the headers:

- **Connection** – Always Upgrade.
- **Upgrade** – Always websocket.
- **Origin** – The URI of the WebSocket client.
- **Sec-WebSocket-Version** – The version of the WebSocket protocol to use.
- **Sec-WebSocket-Key** – A base-64 encoded randomly generated string that identifies the request.

Handling the WebSocket upgrade response

When Amazon Transcribe receives your WebSocket request, it responds with a WebSocket upgrade response. Typically your WebSocket library manages this response and sets up a socket for communications with Amazon Transcribe.

The following is the response from Amazon Transcribe. Line breaks have been added to the `websocket-location` header for readability.

```
HTTP/1.1 101 WebSocket Protocol Handshake

Connection: upgrade
Upgrade: websocket
websocket-origin: wss://transcribestreaming.region.amazonaws.com:8443
websocket-location: transcribestreaming.region.amazonaws.com:8443/stream-transcription-
websocket?
    X-Amz-Algorithm=AWS4-HMAC-SHA256
    &X-Amz-Credential=AKIDEXAMPLE%2F20190117%2Fregion%2Ftranscribe
%2Faws%2Frequest
    &X-Amz-Date=date and time
    &X-Amz-Expires=expiration length
    &X-Amz-SignedHeaders=host
    &language-code=language code
    &media-encoding=media encoding
    &sample-rate=media sample rate
    &X-Amz-Signature=signature
x-amzn-RequestId: RequestId
x-amzn-SessionId: SessionId
Strict-Transport-Security: max-age=31536000
sec-websocket-accept: token
```

The response has the following values:

- **Connection** – Always Upgrade.
- **Upgrade** – Always websocket.
- **websocket-origin** – The URI of the WebSocket server that responded to the request.
- **websocket-location** – The contents of the request URI that was sent to the server. For a description of the contents, see [Creating a pre-signed URL \(p. 77\)](#).
- **x-amzn-RequestId** – An identifier for the request.
- **x-amzn-SessionId** – An identifier for a transcription session.
- **Strict-Transport-Security** – A header that informs browsers to access the endpoint using only HTTPS.
- **sec-websocket-accept** – The hash of the Sec-WebSocket-Key header sent in the request.

Making a WebSocket streaming request

After the WebSocket connection is established, the client can start sending a sequence of audio frames. Each frame contains one data frame that is encoded in event stream encoding. For more information, see [Event stream encoding \(p. 74\)](#).

Each data frame contains three headers combined with a chunk of raw audio bytes. The following table lists and describes the headers.

Header name byte length	Header name (string)	Header value type	Value string byte length	Value string (UTF-8)
13	:content-type	7	24	application/octet-stream
11	:event-type	7	10	AudioEvent
13	:message-type	7	5	event

To end the audio data stream, send an empty audio chunk in an event-stream-encoded message.

Handling a WebSocket streaming response

The response contains event-stream-encoded raw bytes in the payload. It contains the standard prelude and the following headers.

Header name byte length	Header name (string)	Header value type	Value string byte length	Value string (UTF-8)
13	:content-type	7	24	application/octet-stream
11	:event-type	7	15	TranscriptEvent
13	:message-type	7	5	event

When you decode the binary response, you end up with a JSON structure with the results of the transcription. For an example of the JSON response, see [Transcribing streaming audio \(p. 70\)](#).

Handling WebSocket streaming errors

If an exception occurs while processing your request, Amazon Transcribe responds with a terminal WebSocket frame containing an event-stream-encoded response. The response has the headers described in the following table, and the body of the response contains a descriptive error message. After sending the exception response, Amazon Transcribe sends a close frame.

Header name byte length	Header name (string)	Header value type	Value string byte length	Value string (UTF-8)
13	:content-type	7	24	application/octet-stream
15	:exception-type	7	varies	varies, see below
13	:message-type	7	9	exception

The exception-type header contains one of the following values.

- **BadRequestException** – There was a client error when the stream was created, or an error occurred while streaming data. Make sure that your client is ready to accept data and try your request again.

- **InternalFailureException** – Amazon Transcribe had a problem during the handshake with the client. Try your request again.
- **LimitExceededException** – The client exceeded the concurrent stream limit. For more information, see [Amazon Transcribe Limits](#) in the *Amazon Web Services General Reference*. Reduce the number of streams that you are transcribing.
- **UnrecognizedClientException** – The WebSocket upgrade request was signed with an incorrect access key or secret key. Make sure that you are correctly creating the access key and try your request again.

In addition, Amazon Transcribe can return any of the common service errors. For a list, see [Common Errors](#).

Using Amazon Transcribe streaming With HTTP/2

Amazon Transcribe uses a format called *event stream encoding* for streaming transcription. This format encoded binary data with header information that describes the contents of each event. For more information, see [Event stream encoding \(p. 74\)](#). You can use this information for applications that call the Amazon Transcribe endpoint without using the Amazon Transcribe SDK.

When Amazon Transcribe uses the [HTTP/2 protocol](#) for streaming transcriptions, the key components for a streaming request are:

- A header frame. This contains the HTTP/2 headers for the request, and a signature in the authorization header that Amazon Transcribe uses as a seed signature to sign the following data frames.
- One or more message frames in event stream encoding. The frame contains metadata and the raw audio bytes.
- An end frame. This is a signed message in event stream encoding with an empty body.

Note

Amazon Transcribe only supports one stream per HTTP/2 session. If you attempt to use multiple streams, your transcription request will fail.

Streaming request

To make a streaming request, use the [StartStreamTranscription \(p. 408\)](#) API.

Header frame

The header frame is the authorization frame for the streaming transcription. Amazon Transcribe uses the value of the authorization header as the seed for generating a chain of authorization headers for the data frames in the request.

Required headers

The header frame of a request to Amazon Transcribe requires the following HTTP/2 headers:

```
POST /stream-transcription HTTP/2.0
host: transcribestreaming.region.amazonaws.com
authorization: Generated value
```

```
content-type: application/vnd.amazon.eventstream
x-amz-target: com.amazonaws.transcribe.Transcribe.StartStreamTranscription
x-amz-content-sha256: STREAMING-AWS4-HMAC-SHA256-EVENTS
x-amz-date: Date
x-amzn-transcribe-language-code: en-US
x-amzn-transcribe-media-encoding: media encoding
x-amzn-transcribe-sample-rate: Sample rate
transfer-encoding: chunked
```

In the request, use the following values for the host, authorization, and `x-amz-date` headers:

- **host**: Use the AWS Region where you are calling Amazon Transcribe. For a list of valid regions, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.
- **authorization**: The Signature Version 4 signature for the request. For more information about creating a signature, see [Signing AWS Requests with Signature Version 4](#) in the *Amazon Web Services General Reference*.
- **x-amz-date**: Generate a date and time for the request following the instructions in [Handling Dates in Signature Version 4](#) in the *Amazon Web Services General Reference*.

For more information about the headers specific to Amazon Transcribe, see the [StartStreamTranscription \(p. 408\)](#) API.

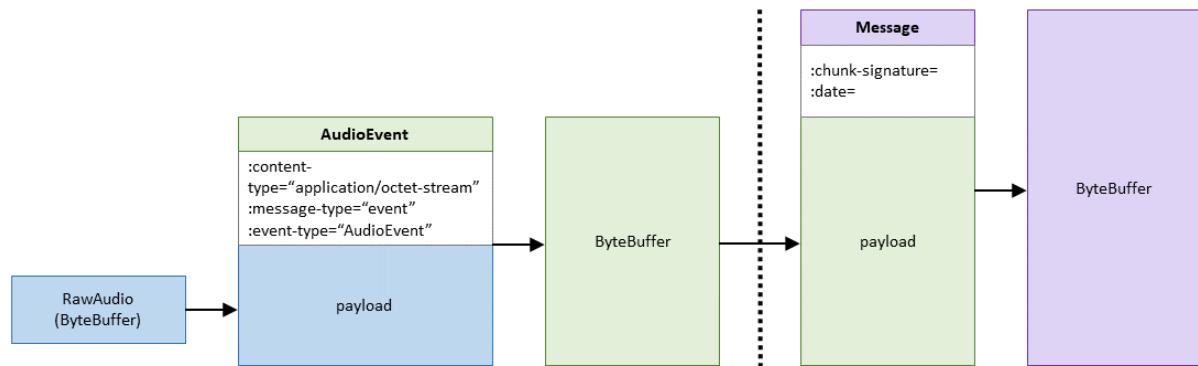
Data frames

Each request contains one or more data frames. The data frames use event stream encoding. The encoding supports bidirectional data transmission between a client and a server.

There are two steps to creating a data frame:

1. Combine the raw audio data with metadata to create the payload of the request.
2. Combine the payload with a signature to form the event message that is sent to Amazon Transcribe.

The following diagram shows how this works.



Create the audio event

To create the message to send to Amazon Transcribe, create the audio event. Combine the headers described in the following table with a chunk of audio bytes into an event-encoded message.

Header Name Byte Length	Header Name (string)	Header Value Type	Value String Byte Length	Value String (UTF-8)
13	:content-type	7	24	application/octet-stream
11	:event-type	7	10	AudioEvent
13	:message-type	7	5	event

To create the payload for the event message, use a buffer in raw-byte format.

Create the message

Create a data frame using the audio event payload to send to Amazon Transcribe. The data frame contains event-encoding headers that include the current date and a signature for the audio chunk and the audio event. To indicate to Amazon Transcribe that the audio stream is complete, send an empty data frame that contains only the date and signature.

Header Name Byte Length	Header Name (String)	Header Value Type	Value String Byte Length	Value
16	:chunk-signature	6	varies	Generated signature
5	:date	8	8	Timestamp

To create the signature for the data frame, first create a string to sign, and then calculate the signature for the event. Construct the string to sign as follows.

```
String stringToSign =
    "AWS4-HMAC-SHA256-PAYOUT" +
    "\n" +
    DATE +
    "\n" +
    KEYPATH +
    "\n" +
    Hex(priorSignature) +
    "\n" +
    HexHash(nonSignatureHeaders) +
    "\n" +
    HexHash(payload);
```

- **DATE:** The current date and time in Universal Time Coordinated (UTC) and using the [ISO 8601 format](#). Don't include milliseconds in the date. For example, 20190127T223754Z is 22:37:54 on 1/27/2019.
- **KEYPATH:** The signature scope in the format `date/region/service/aws4_request`. For example, 20190127/us-east-1/transcribe/aws4_request.
- **priorSignature:** The signature for the previous frame. For the first data frame, use the signature of the header frame.
- **nonSignatureHeaders:** The DATE header encoded as a string.
- **payload:** The byte buffer containing the audio event data.
- **Hex:** A function that encodes its input into a hexadecimal representation.
- **HexHash:** A function that first creates a SHA-256 hash of its input and then uses the `Hex` function to encode the hash.

After you have constructed the string to sign, sign it using the key that you derived for Signature Version 4, as follows. For details, see [Examples of How to Derive a Signing Key for Signature Version 4](#) in the [Amazon Web Services General Reference](#).

```
String signature = HMACSHA256(derivedSigningKey, stringToSign);
```

- **HMACSHA256**: A function that creates a signature using the SHA-256 hash function.
- **derivedSigningKey**: The Signature Version 4 signing key.
- **stringToSign**: The string that you calculated for the data frame.

After you have calculated the signature for the data frame, construct a byte buffer containing the date, the signature, and the audio event payload. Send the byte array to Amazon Transcribe for transcription.

End frame

To indicate that the audio stream is complete, send an end frame to Amazon Transcribe. The *end frame* is a data frame with an empty payload. You construct the end frame the same way that you construct a data frame.

Streaming response

Responses from Amazon Transcribe are also sent using event stream encoding. Use this information to decode a response from the [StartStreamTranscription \(p. 408\)](#) API.

Transcription response

A transcription response is event stream encoded. It contains the standard prelude and the following headers.

Header Name Byte Length	Header Name (String)	Header Value Type	Value String Byte Length	Value String (UTF-8)
13	:content-type	7	24	application/octet-stream
11	:event-type	7	10	AudioEvent
13	:message-type	7	5	event

For details, see [Event stream encoding \(p. 74\)](#).

When the response is decoded, it contains the following information.

```
:content-type: "application/json"
:event-type: "TranscriptEvent"
:message-type: "event"

JSON transcription information
```

For an example of the JSON structure returned by Amazon Transcribe, see [Using Amazon Transcribe streaming With HTTP/2 \(p. 83\)](#).

Exception response

If there is an error in processing your transcription stream, Amazon Transcribe sends an exception response. The response is event stream encoded. For details, see [Event stream encoding \(p. 74\)](#).

The response contains the standard prelude and the following headers.

Header Name Byte Length	Header Name (String)	Header Value Type	Value String Byte Length	Value String (UTF-8)
13	:content-type	7	16	application/json
11	:event-type	7	19	BadRequestException
13	:message-type	7	9	exception

When the exception response is decoded, it contains the following information.

```
:content-type: "application/json"
:event-type: "BadRequestException"
:message-type: "exception"
```

Exception message

Example request and response

The following is an end-to-end example of a streaming transcription request. In this example, binary data is represented as base64-encoded strings. In an actual response, the data are raw bytes.

Step 1: Start the session with Amazon Transcribe

To start the session, send an HTTP/2 request to Amazon Transcribe.

```
POST /stream-transcription HTTP/2.0
host: transcribestreaming.region.amazonaws.com
authorization: Generated value
content-type: application/vnd.amazon.eventstream
x-amz-content-sha256: STREAMING-AWS4-HMAC-SHA256-EVENTS
x-amz-date: Date
x-amzn-transcribe-language-code: en-US
x-amzn-transcribe-media-encoding: pcm
x-amzn-transcribe-sample-rate: Sample rate
transfer-encoding: chunked
```

Step 2: Send authentication information to Amazon Transcribe

Amazon Transcribe sends the following response.

```
HTTP/2.0 200
x-amzn-transcribe-language-code: en-US
x-amzn-transcribe-sample-rate: Sample rate
x-amzn-request-id: 8a08df7d-5998-48bf-a303-484355b4ab4e
x-amzn-transcribe-session-id: b4526fcf-5eee-4361-8192-d1cb9e9d6887
x-amzn-transcribe-media-encoding: pcm
x-amzn-RequestId: 8a08df7d-5998-48bf-a303-484355b4ab4e
content-type: application/vnd.amazon.eventstream
```

Step 3: Create an audio event

Create an audio event containing the audio data to send. For details, see [Event stream encoding \(p. 74\)](#). The binary data in this request is base64-encoded. In an actual request, the data is raw bytes.

```
:content-type: "application/octet-stream"
:event-type: "AudioEvent"
:message-type: "event"

Uk1GRjzxPQBXQVZFZm10IBAAAAABAAEAgD4AAAB9AACABAAGF0YVTwPQAAAAAAAAAAAAAAD//wIA/f8EAA==
```

Step 4: Create an audio event message

Create an audio message that contains the audio data to send to Amazon Transcribe. For details, see [Event stream encoding \(p. 74\)](#). The audio event data in this example is base64-encoded. In an actual request, the data is raw bytes.

```
:date: 2019-01-29T01:56:17.291Z
:chunk-signature: signature

AAAA0gAAAIKV0RFcTTcjb250ZW50LXR5cGUHABhhcHBsaWNhdGlvbi9vY3RldC1zdHJ1YW0LOmV2ZW50LXR5
cGUHAApBdWRpb0V2ZW50DTptZXNzYwd1LXR5cGUHAAVldmVudAxDb256ZW50LVR5cGUHABphcHBsaWNhdGlv
b194LWFtei1qc29uLTExMVJJRkY8T0AV0FWRWZtdCAQAAAAQABAIA+AAAAfQAAAQAgAQAGRhdGFU8D0AAAAA
AAAAAAAAAA//8CAP3/BAC7QLFF
```

Step 5: Use the response from Amazon Transcribe

Amazon Transcribe creates a stream of transcription events that it sends to your application. The events are sent in raw-byte format. In this example, the bytes are base64-encoded.

Amazon Transcribe sends the following response.

```
AAAAUwAAAEP1RHpyBTpkYXRlCAAAAWiXUkMLEDpjAHVuay1zaWduYXR1cmUGACt6Zy+uymwEK2SrLp/zVBI
5eGn83jdBwCaRUBJA+eaDafqjqI=
```

To see the transcription results, decode the raw bytes using event-stream encoding.

```
:event-type: "TranscriptEvent"
:content-type: "application/json"
:message-type: "event"

{"Transcript": {"Results": [results]}}
```

For an example of the JSON structure returned by Amazon Transcribe, see [Event stream encoding \(p. 74\)](#).

Step 6: End the transcription stream

Finally, send an empty audio event to Amazon Transcribe to end the transcription stream. Create the audio event exactly like any other, except with an empty payload. Sign the event and include the signature in the `:chunk-signature` header, as follows.

```
:date: 2019-01-29T01:56:17.291Z
:chunk-signature: signature
```

HTTP/2 streaming retry client

You can use the following code in your applications to handle retry logic for Amazon Transcribe streaming transcription. The code provides tolerance for intermittent failures in the connection to Amazon Transcribe. There are two parts of the client: an interface that you implement for your application, and the retry client itself.

Streaming retry client code

This code implements a streaming retry client. It manages the connection to Amazon Transcribe and retries sending data when there are errors on the connection. For example, if there is a transient error on the network, this client resends the request that failed.

The retry client has two properties that control the behavior of the client. You can set:

- The maximum number of times that the client should attempt before failing. Reduce this value to make your application stop retrying sooner when there are network issues. The default is 10.
- The time in milliseconds that the client should wait between retries. Longer times raise the risk of losing data, shorter times raise the risk of your application being throttled. The default is 100 milliseconds.

The following is the client. You can copy this code to your application or use it as a starting point for your own client.

```
public class TranscribeStreamingRetryClient {  
  
    private static final int DEFAULT_MAX_RETRIES = 10;  
    private static final int DEFAULT_MAX_SLEEP_TIME_MILLS = 100;  
    private static final Logger log =  
        LoggerFactory.getLogger(TranscribeStreamingRetryClient.class);  
    private final TranscribeStreamingAsyncClient client;  
    List<Class<?>> nonRetriableExceptions = Arrays.asList(BadRequestException.class);  
    private int maxRetries = DEFAULT_MAX_RETRIES;  
    private int sleepTime = DEFAULT_MAX_SLEEP_TIME_MILLS;  
  
    /**  
     * Create a TranscribeStreamingRetryClient with given credential and configuration  
     */  
    public TranscribeStreamingRetryClient(AwsCredentialsProvider creds,  
                                         String endpoint, Region region) throws  
        URISyntaxException {  
        this(TranscribeStreamingAsyncClient.builder()  
            .overrideConfiguration(  
                c -> c.putAdvancedOption(  
                    SdkAdvancedClientOption.SIGNER,  
                    EventStreamAws4Signer.create()))  
            .credentialsProvider(creds)  
            .endpointOverride(new URI(endpoint))  
            .region(region)  
            .build());  
    }  
  
    /**  
     * Initiate TranscribeStreamingRetryClient with TranscribeStreamingAsyncClient
```

```

        */

    public TranscribeStreamingRetryClient(TranscribeStreamingAsyncClient client) {
        this.client = client;
    }

    /**
     * Get Max retries
     */
    public int getMaxRetries() {
        return maxRetries;
    }

    /**
     * Set Max retries
     */
    public void setMaxRetries(int maxRetries) {
        this.maxRetries = maxRetries;
    }

    /**
     * Get sleep time
     */
    public int getSleepTime() {
        return sleepTime;
    }

    /**
     * Set sleep time between retries
     */
    public void setSleepTime(int sleepTime) {
        this.sleepTime = sleepTime;
    }

    /**
     * Initiate a Stream Transcription with retry.
     */
    public CompletableFuture<Void> startStreamTranscription(final
StartStreamTranscriptionRequest request,
                                                final Publisher<AudioStream>
publisher,
                                                final
StreamTranscriptionBehavior responseHandler) {

        CompletableFuture<Void> finalFuture = new CompletableFuture<>();

        recursiveStartStream(rebuildRequestWithSession(request), publisher,
responseHandler, finalFuture, 0);

        return finalFuture;
    }

    /**
     * Recursively call startStreamTranscription() until the request is completed or we run
out of retries.
     */
    private void recursiveStartStream(final StartStreamTranscriptionRequest request,
final Publisher<AudioStream> publisher,
final StreamTranscriptionBehavior responseHandler,
final CompletableFuture<Void> finalFuture,
final int retryAttempt) {
        CompletableFuture<Void> result = client.startStreamTranscription(request,
publisher,
getResponseHandler(responseHandler));
}

```

```

        result.whenComplete((r, e) -> {
            if (e != null) {
                log.debug("Error occurred:", e);

                if (retryAttempt <= maxRetries && isExceptionRetriable(e)) {
                    log.debug("Retriable error occurred and will be retried.");
                    log.debug("Sleeping for sometime before retrying...");
                    try {
                        Thread.sleep(sleepTime);
                    } catch (InterruptedException e1) {
                        log.debug("Unable to sleep. Failed with exception: ", e);
                        e1.printStackTrace();
                    }
                    log.debug("Making retry attempt: " + (retryAttempt + 1));
                    recursiveStartStream(request, publisher, responseHandler, finalFuture,
                            retryAttempt + 1);
                } else {
                    log.error("Encountered unretrievable exception or ran out of retries. ");
                    responseHandler.onError(e);
                    finalFuture.completeExceptionally(e);
                }
            } else {
                responseHandler.onComplete();
                finalFuture.complete(null);
            }
        });
    }

    private StartStreamTranscriptionRequest
rebuildRequestWithSession(StartStreamTranscriptionRequest request) {
    return StartStreamTranscriptionRequest.builder()
        .languageCode(request.languageCode())
        .mediaEncoding(request.mediaEncoding())
        .mediaSampleRateHertz(request.mediaSampleRateHertz())
        .sessionId(UUID.randomUUID().toString())
        .build();
}

/**
 * StartStreamTranscriptionResponseHandler implements subscriber of transcript stream
 * Output is printed to standard output
 */
private StartStreamTranscriptionResponseHandler getResponseHandler(
    StreamTranscriptionBehavior transcriptionBehavior) {
    final StartStreamTranscriptionResponseHandler build =
StartStreamTranscriptionResponseHandler.builder()
    .onResponse(r -> {
        transcriptionBehavior.onResponse(r);
    })
    .onError(e -> {
        //Do nothing here. Don't close any streams that shouldn't be cleaned up
yet.
    })
    .onComplete(() -> {
        //Do nothing here. Don't close any streams that shouldn't be cleaned up
yet.
    })

    .subscriber(event -> transcriptionBehavior.onStream(event))
    .build();
    return build;
}

/**
 * Check if the exception can be retried.
 *

```

```

        */
    private boolean isExceptionRetriable(Throwable e) {
        e.printStackTrace();

        return nonRetriableExceptions.contains(e.getClass());
    }

    public void close() {
        this.client.close();
    }
}

```

Streaming retry client interface code

This interface is similar to the response handler used in the getting started example. It implements the same event handlers. Implement this interface to use the streaming retry client.

```

package com.amazonaws.transcribestreaming;

import
    software.amazon.awssdk.services.transcribestreaming.model.StartStreamTranscriptionResponse;
import software.amazon.awssdk.services.transcribestreaming.model.TranscriptResultStream;

/**
 * Defines how a stream response should be handled.
 * You should build a class implementing this interface to define the behavior.
 */
public interface StreamTranscriptionBehavior {
    /**
     * Defines how to respond when encountering an error on the stream transcription.
     */
    void onError(Throwable e);

    /**
     * Defines how to respond to the Transcript result stream.
     */
    void onStream(TranscriptResultStream e);

    /**
     * Defines what to do on initiating a stream connection with the service.
     */
    void onResponse(StartStreamTranscriptionResponse r);

    /**
     * Defines what to do on stream completion
     */
    void onComplete();
}

```

The following is an example implementation of the `StreamTranscriptionBehavior` interface. You can use this implementation or use it as a starting point for your own implementation.

```

package com.amazonaws.transcribestreaming.retryclient;

import com.amazonaws.transcribestreaming.retryclient.StreamTranscriptionBehavior;

```

```

import software.amazon.awssdk.services.transcribestreaming.model.Result;
import software.amazon.awssdk.services.transcribestreaming.model.StartStreamTranscriptionResponse;
import software.amazon.awssdk.services.transcribestreaming.model.TranscriptEvent;
import software.amazon.awssdk.services.transcribestreaming.model.TranscriptResultStream;

import java.util.List;

/**
 * Implementation of StreamTranscriptionBehavior to define how a stream response should be
handled.
*
* COPYRIGHT:
*
* Copyright 2018 Amazon.com, Inc. or its affiliates. All Rights Reserved.
*
* Licensed under the Apache License, Version 2.0 (the "License").
* You may not use this file except in compliance with the License.
* A copy of the License is located at
*
*      http://www.apache.org/licenses/LICENSE-2.0
*
* or in the "license" file accompanying this file. This file 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.
*/
public class StreamTranscriptionBehaviorImpl implements StreamTranscriptionBehavior {

    @Override
    public void onError(Throwable e) {
        System.out.println("== Failure Encountered ==");
        e.printStackTrace();
    }

    @Override
    public void onStream(TranscriptResultStream e) {
        // EventResultStream has other fields related to the timestamp of the
transcripts in it.
        // Please refer to the javadoc of TranscriptResultStream for more details
        List<Result> results = ((TranscriptEvent) e).transcript().results();
        if (results.size() > 0) {
            if (results.get(0).alternatives().size() > 0)
                if (!results.get(0).alternatives().get(0).transcript().isEmpty())
                    System.out.println(results.get(0).alternatives().get(0).transcript());
        }
    }

    @Override
    public void onResponse(StartStreamTranscriptionResponse r) {
        System.out.println(String.format("== Received Initial response. Request Id: %s",
===", r.requestId()));
    }

    @Override
    public void onComplete() {
        System.out.println("== All records stream successfully ==");
    }
}

```

Next step

[Using the HTTP/2 retry client \(p. 94\)](#)

Using the HTTP/2 retry client

The following is a sample application that uses the retry client to transcribe audio from either a file or your microphone. You can use this application to test the client, or you can use it as a starting point for your own applications.

To run the sample, do the following:

- Copy the retry client to your workspace. See [Streaming retry client code \(p. 89\)](#).
- Copy the retry client interface to your workspace. Implement the interface, or you can use the sample implementation. See [Streaming retry client interface code \(p. 92\)](#).
- Copy the sample application to your workspace. Build and run the application.

```
public class StreamingRetryApp {  
    private static final String endpoint = "endpoint";  
    private static final Region region = Region.US_EAST_1;  
    private static final int sample_rate = 28800;  
    private static final String encoding = " ";  
    private static final String language = LanguageCode.EN_US.toString();  
  
    public static void main(String args[]) throws URISyntaxException, ExecutionException,  
    InterruptedException, LineUnavailableException, FileNotFoundException {  
        /**  
         * Create Amazon Transcribe streaming retry client.  
         */  
  
        TranscribeStreamingRetryClient client = new  
        TranscribeStreamingRetryClient(EnvironmentVariableCredentialsProvider.create() ,endpoint,  
        region);  
  
        StartStreamTranscriptionRequest request = StartStreamTranscriptionRequest.builder()  
            .languageCode(language)  
            .mediaEncoding(encoding)  
            .mediaSampleRateHertz(sample_rate)  
            .build();  
        /**  
         * Start real-time speech recognition. The Amazon Transcribe streaming java client  
         uses the Reactive-streams  
         * interface. For reference on Reactive-streams:  
         * https://github.com/reactive-streams/reactive-streams-jvm  
         */  
        CompletableFuture<Void> result = client.startStreamTranscription(  
            /**  
             * Request parameters. Refer to API documentation for details.  
             */  
            request,  
            /**  
             * Provide an input audio stream.  
             * For input from a microphone, use getStreamFromMic().  
             * For input from a file, use getStreamFromFile().  
             */  
            new AudioStreamPublisher(  
                new FileInputStream(new File("FileName"))),  
            /**  
             * Object that defines the behavior on how to handle the stream  
             */  
            new StreamTranscriptionBehaviorImpl());  
  
        /**  
         * Synchronous wait for stream to close, and close client connection  
         */
```

```

        result.get();
        client.close();
    }

    private static class AudioStreamPublisher implements Publisher<AudioStream> {
        private final InputStream inputStream;
        private static Subscription currentSubscription;

        private AudioStreamPublisher(InputStream inputStream) {
            this.inputStream = inputStream;
        }

        @Override
        public void subscribe(Subscriber<? super AudioStream> s) {
            if (this.currentSubscription == null) {
                this.currentSubscription = new
TranscribeStreamingDemoApp.SubscriptionImpl(s, inputStream);
            } else {
                this.currentSubscription.cancel();
                this.currentSubscription = new
TranscribeStreamingDemoApp.SubscriptionImpl(s, inputStream);
            }
            s.onSubscribe(currentSubscription);
        }
    }
}

```

Partial result stabilization

Amazon Transcribe starts returning transcription results as soon as you start streaming your audio. It returns these *partial results* incrementally until it generates a finalized result at the level of a natural speech segment. A natural speech segment is continuous speech that is broken up by a pause in the audio, or a change in speaker. For each speech segment, any word or phrase in these partial results might change. Amazon Transcribe continues outputting partial results until it generates the final transcription result for a speech segment.

You can use partial result stabilization to generate partial transcription results that are less likely to change. If you activate partial result stabilization, only the last few words from the partial results can change.

Turning on partial result stabilization changes how Amazon Transcribe produces the final transcription results. Because only the last few words can change between the partial results, activating partial result stabilization might impact the transcription accuracy.

Partial result stabilization reduces the time it takes to present the transcription results. Partial result stabilization also gives you the option to change the amount of text that you want to display. You can use partial result stabilization in either your HTTP/2 or WebSocket streams.

When you start a stream with Amazon Transcribe, you have two options for each speech segment:

- Wait for the finalized results.
- Use the partial results in the transcription output

Using partial results reduces latency. This reduction in latency could be helpful for use cases such as subtitling videos or generating captions for live streams. You can use partial result stabilization to present the transcription results to your viewers more quickly.

With partial result stabilization, you can choose the part of the transcription output that won't change. You can also use it to present viewers with subtitles that are easier to read. By displaying partial results, you also limit the amount of text displayed at a given time.

The following text shows how the transcription output might change as Amazon Transcribe processes a stream when partial result stabilization isn't activated. The last line represents the finalized result. The lines before that one represent the partial results.

```
And if you held onto the ships
and if you held onto the shift
and if you hold onto the shift key
and if you hold onto the shift keys
```

The following text shows how the transcription output might change as Amazon Transcribe processes a stream when partial result stabilization is activated. The last line represents the finalized result. The lines before that one represent the partial results.

```
and if you hold onto the ships
and if you hold onto the shift
and if you hold onto the shift key
and if you hold onto the shift keys
```

In addition to turning on partial result stabilization, you can also change the stability level of the transcription results. The stability level determines how stable you want the transcription results to be. A higher stability level means that the transcription results that Amazon Transcribe returns are less likely to change. This setting comes with lower overall transcription accuracy than the accuracy of a lower stability level. A lower stability level results in more accurate transcription results, but those results are more likely to change.

Using partial result stabilization in an HTTP/2 request

The following is the syntax for the parameters of an HTTP/2 request with partial result stabilization activated.

```
POST /stream-transcription HTTP/2
x-amzn-transcribe-language-code: LanguageCode
x-amzn-transcribe-sample-rate: MediaSampleRateHertz
x-amzn-transcribe-media-encoding: MediaEncoding
x-amzn-transcribe-vocabulary-name: VocabularyName
x-amzn-transcribe-session-id: SessionId
x-amzn-transcribe-vocabulary-filter-name: VocabularyFilterName
x-amzn-transcribe-vocabulary-filter-method: VocabularyFilterMethod
x-amzn-transcribe-language-model-name: LanguageModelName
x-amzn-transcribe-enable-channel-identification: EnableChannelIdentification
x-amzn-transcribe-number-of-channels: NumberOfChannels
x-amzn-transcribe-show-speaker-label: ShowSpeakerLabel
x-amzn-transcribe-enable-partial-results-stabilization: EnablePartialResultsStabilization
x-amzn-transcribe-partial-results-stability: PartialResultsStability
Content-type: application/json
```

To activate partial result stabilization in an HTTP/2 stream, use the [StartStreamTranscription \(p. 408\)](#) API and specify the following:

- **LanguageCode** – The language code that corresponds to the language spoken in the stream. For a list of languages available in streaming transcription, see [What is Amazon Transcribe? \(p. 1\)](#).
- **MediaSampleHertz** – The sample rate of the audio.

- `EnablePartialResultsStabilization = true`.
- `PartialResultsStability` (optional) – The stability level of the transcription results. Valid values are high, medium, and low. Between the stability levels, the high stability level has partial results that are least likely to change as the stream progresses. The low stability level shows partial results that are the most likely to change as the stream progresses, but those results have the highest overall accuracy. If you don't specify a value, Amazon Transcribe uses high.

Using partial result stabilization with WebSocket streams

To use partial result stabilization in WebSocket streams, use the following format to create a pre-signed URL to start a WebSocket request and specify the following:

```
GET wss://transcribestreaming.region.amazonaws.com:8443/stream-transcription-websocket
?language-code=languageCode
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=Signature Version 4 credential scope
&X-Amz-Date=date
&X-Amz-Expires=time in seconds until expiration
&X-Amz-Security-Token=security-token
&X-Amz-Signature=Signature Version 4 signature
&X-Amz-SignedHeaders=host
&media-encoding=mediaEncoding
&sample-rate=mediaSampleRateHertz
&session-id=sessionId
&enable-partial-results-stabilization=true
&partial-results-stability=stability-level
```

A pre-signed URL contains the information to set up bi-directional communication between your application and Amazon Transcribe.

To activate partial result stabilization, you specify the following parameters:

- `enable-partial-results-stabilization = true`
- `partial-results-stability` (optional) – The stability level of the transcription results. Valid values are high, medium, and low. Between the stability levels, the high stability level has partial results that are least likely to change as the stream progresses. The low stability level shows partial results that are the most likely to change as the stream progresses, but those results have the highest overall accuracy. If you don't specify a value, Amazon Transcribe uses high.

For more information about completing WebSocket requests, see [Creating a pre-signed URL \(p. 77\)](#).

Streaming transcription output

The following is an example response for a streaming request with metadata removed for clarity.

```
"Transcript": {
    "Results": [
        {
            ...
            "Alternatives": [
                {
                    "Items": [
                        ...
                    ]
                }
            ]
        }
    ]
}
```

```
...
{
    "Content": "and",
    "EndTime": 1.02,
    "StartTime": 0.98,
    "Type": "pronunciation",
    "VocabularyFilterMatch": false,
    "Stable": true
},
{
    "Content": "if",
    "EndTime": 1.26,
    "StartTime": 1.03,
    "Type": "pronunciation",
    "VocabularyFilterMatch": false,
    "Stable": true
},
{
    "Content": "you",
    "EndTime": 1.41,
    "StartTime": 1.27,
    "Type": "pronunciation",
    "VocabularyFilterMatch": false,
    "Stable": true
},
{
    "Content": "hold",
    "EndTime": 1.81,
    "StartTime": 1.42,
    "Type": "pronunciation",
    "VocabularyFilterMatch": true,
    "Stable": true
},
{
    "Content": "onto",
    "EndTime": 2.11,
    "StartTime": 1.82,
    "Type": "pronunciation",
    "VocabularyFilterMatch": true,
    "Stable": true
},
{
    "Content": "the",
    "EndTime": 2.32,
    "StartTime": 2.12,
    "Type": "pronunciation",
    "VocabularyFilterMatch": true,
    "Stable": true
},
{
    "Content": "shift",
    "EndTime": 2.56,
    "StartTime": 2.33,
    "Type": "pronunciation",
    "VocabularyFilterMatch": true,
    "Stable": true
}
{
    "Content": "key",
    "EndTime": 2.81,
    "StartTime": 2.57,
    "Type": "pronunciation",
    "VocabularyFilterMatch": true,
    "Stable": false
}
...
```

```
        ]
      }
    ]
  }
}
```

When you activate partial result stabilization, Amazon Transcribe uses the `Stable` field to indicate whether the `item` is stable. An item is a transcribed word or phrase. You can use the `Stable` field to include or remove items that aren't stable between the stability levels, the `high` stability level has partial results are least likely to change as the stream progresses. The `low` stability level shows partial results that are the most likely to change as the stream progresses, but those results have the highest overall accuracy.

Redacting or identifying personally identifiable information

Use *content redaction* to omit sensitive personally identifiable information (PII) from your transcription results. Use *content identification* to flag PII in your transcript without redacting it.

Important

The redaction feature is designed to identify and remove sensitive data. However, due to the predictive nature of machine learning, Amazon Transcribe may not identify and remove all instances of sensitive data in your transcript. We strongly recommend that you review any redacted output to ensure it meets your needs.

The redaction feature does not meet the requirements for de-identification under medical privacy laws, such as the U.S. Health Insurance Portability and Accountability Act of 1996 (HIPAA).

PII redaction can be performed on batch transcription jobs and streaming transcriptions, whereas PII identification can only be performed on streaming transcriptions. With batch transcription jobs, you can choose to have both a redacted and an unredacted transcript.

Amazon Transcribe can identify the following types of PII:

PII entity	Definition
Bank Account Number	A number that uniquely identifies a bank account.
Bank Routing Number	A number that identifies the location of a bank account.
Credit Card Number or Debit Card Number	A value that uniquely defines a payment card issued by a bank.
Credit Card or Debit Card CVV Code	A 3-digit or 4-digit security code on each credit card.
Credit Card or Debit Card Expiration Date	The month and year a card expires.
Debit or credit card PIN	A security code issued by a bank or credit union. This number is used for bank accounts and payment cards.
Email address	The unique identifier of an email box where messages are delivered.
U.S. mailing address	The United States mailing address of an individual.
Name	The first and last name of a person.
U.S. phone number	A 10-digit phone number within the United States.
Social Security Number	A 9-digit number (or the last 4 digits of that number). Issued to U.S. citizens, permanent residents, and temporary residents with employment.

PII entity	Definition
All PII	Redact or identify all PII types listed in this table.

Topics

- [Redacting or identifying PII in an audio file \(p. 101\)](#)
- [Redacting or identifying PII in a real-time stream \(p. 104\)](#)
- [Example PII redaction and identification output \(p. 106\)](#)

Redacting or identifying PII in an audio file

Use a batch transcription job to transcribe audio files and redact the personally identifiable information (PII) within them. When you activate PII Redaction, Amazon Transcribe redacts the PII that it identified in the transcription results. For information about PII that Amazon Transcribe can identify, see the list of [redactable information \(p. 100\)](#).

Note

PII redaction for batch jobs is only supported in these AWS regions: Asia Pacific (Hong Kong), Asia Pacific (Mumbai), Asia Pacific (Seoul), Asia Pacific (Singapore), Asia Pacific (Sydney), Asia Pacific (Tokyo), AWS GovCloud (US-West), Canada (Central), EU (Frankfurt), EU (Ireland), EU (London), EU (Paris), Middle East (Bahrain), South America (Sao Paulo), US East (N. Virginia), US East (Ohio), US West (Oregon), and US West (N. California).

Amazon Transcribe replaces each identified instance of PII with a [PII] tag in the transcript. You can use this feature to protect privacy and comply with local laws and regulations. PII redaction gives you the ability to easily review and share transcripts to improve the customer service experience, coach agents, and discover new business opportunities while protecting sensitive personal information. You can use this feature for source audio in US English (en-US).

When redaction is enabled, Amazon Transcribe can generate the following files, depending on which options you select:

- A redacted text file. A redacted text file displays [PII] in place of the redacted text; see below for example excerpts from a redacted conversation.
- An unredacted (original) text file. This file is only generated if you choose to get both the redacted and the original transcripts.

Note

The original unredacted file is the only place where the complete conversation is stored. If you delete it, there is no record of the unredacted sensitive data.

Both redacted and unredacted transcripts are stored in the same output S3 bucket. Amazon Transcribe stores them in either a bucket you specify or in the default S3 bucket managed by the service.

A redacted transcript replaces sensitive PII with a [PII] tag, as shown in the first truncated JSON output on this page. Because Amazon Transcribe has redacted this transcript, the `isRedacted` field of this JSON output is `true`. Each JSON output of a transcription job has a `results` section that contains the transcription results. Every word, number, punctuation mark, or redaction has a confidence value.

Transcription jobs using automatic content redaction generate two types of confidence values. The Automatic Speech Recognition (ASR) confidence indicates the items that have the type of pronunciation or punctuation is a specific utterance. In the transcript output below, the word Good has a confidence of 1.0. This confidence value indicates that Amazon Transcribe is 100 percent

confident that the word uttered in this transcript is Good. The confidence value for a [PII] tag is the confidence that the speech it flagged for redaction is truly PII. In the following transcript output, the confidence of 0.9999 indicates that Amazon Transcribe is 99.99 percent confident that the entity it redacted in the transcript is PII.

You can start a batch transcription job using either the [StartTranscriptionJob \(p. 376\)](#) operation or the Amazon Transcribe console.

Console

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, choose **Transcription jobs**, then select the **Create job** button (top right). This will open the **Specify job details** page.
3. After filling in your desired fields on the **Specify job details** page, click the **Next** button to go to the **Configure job - optional** page. Here you'll find the **Content removal** panel with the options of **Automatic content redaction** and **Vocabulary filtering**.

The screenshot shows the 'Configure job - optional' page with the 'Content removal' panel expanded. The panel contains two sections: 'Audio settings' and 'Content removal'. Under 'Audio settings', there are two options: 'Audio identification' (which is selected) and 'Alternative results'. Under 'Content removal', there are three options: 'Automatic content redaction' (selected), 'Vocabulary filtering', and 'Include unredacted transcript in job output' (unchecked).

Configure job - optional Info

Audio settings

Audio identification Info
Choose to split multi-channel audio into separate channels for transcription, or identify speakers in the input audio.

Alternative results Info
Enable to view more transcription results

Content removal

Content removal conceals information in the resulting transcript from your source audio file. Amazon Transcribe changes items in the transcript and does not modify the source audio.

Automatic content redaction Info
Automatic content redaction removes personally identifiable information (PII) in your transcripts. Redactions in transcripts show up as [PII].

Vocabulary filtering Info
Vocabulary filtering can remove, mask or tag specified words in the final transcript.

Include unredacted transcript in job output
Returns unredacted version of the transcript in addition to the redacted version.

4. Once you select **Automatic content redaction**, which redacts all PII, you have the option to check the **Include unredacted transcript in job output** box.

The screenshot shows the 'Content removal' panel with the 'Include unredacted transcript in job output' checkbox checked. The panel also contains other options: 'Automatic content redaction' (selected) and 'Vocabulary filtering'.

Content removal

Content removal conceals information in the resulting transcript from your source audio file. Amazon Transcribe changes items in the transcript and does not modify the source audio.

Automatic content redaction Info
Automatic content redaction removes personally identifiable information (PII) in your transcripts. Redactions in transcripts show up as [PII].

Include unredacted transcript in job output
Returns unredacted version of the transcript in addition to the redacted version.

Vocabulary filtering Info
Vocabulary filtering can remove, mask or tag specified words in the final transcript.

5. Click the **Create job** button to run your transcription job.

AWS CLI

This example uses the [start-transcription-job](#) command and `content-redaction` parameter.

```
aws transcribe start-transcription-job \
--transcription-job-name my-job-name \
--media MediaFileUri=s3://your-S3-bucket/S3-prefix/your-filename.file-extension \
--language-code en-US \
--content-redaction RedactionType=PII,RedactionOutput=redacted
```

Here's another example using the [start-transcription-job](#) method, and the request body redacts PII for that job.

```
aws transcribe start-transcription-job \
--cli-input-json file://example-start-command.json
```

The file *example-start-command.json* contains the following request body.

```
{
    "TranscriptionJobName": "my-job-name",
    "LanguageCode": "en-US",
    "Media": {
        "MediaFileUri": "s3://your-S3-bucket/S3-prefix/your-filename.file-extension"
    },
    "ContentRedaction": {
        "RedactionOutput": "redacted",
        "RedactionType": "PII"
    }
}
```

AWS SDK for Python (Boto3)

The following example uses the AWS SDK for Python (Boto3) to redact content using the `ContentRedaction` argument for the [start_transcription_job](#) method:

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "my-job-name"
job_uri = "s3://your-S3-bucket/S3-prefix/your-filename.file-extension"
transcribe.start_transcription_job(
    TranscriptionJobName=job_name,
    Media={'MediaFileUri': job_uri},
    MediaFormat='wav',
    LanguageCode='en-US',
    ContentRedaction = { 'RedactionOutput':'redacted',
        'RedactionType':'PII'
    })
while True:
    status = transcribe.get_transcription_job(TranscriptionJobName=job_name)
    if status['TranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED', 'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
```

```
print(status)
```

Redacting or identifying PII in a real-time stream

You can redact Personally Identifiable Information (PII) in either HTTP/2 or WebSocket streams. An additional option available for streaming audio is *PII identification*. When you activate PII Identification, Amazon Transcribe labels the PII in your transcription results under an `Entities` object. For an output sample, see [Example redacted streaming output \(p. 107\)](#) and [Example PII identification output \(p. 108\)](#).

Note

PII redaction for streaming is only supported in these AWS regions: Asia Pacific (Seoul), Asia Pacific (Sydney), Asia Pacific (Tokyo), Canada (Central), EU (Frankfurt), EU (Ireland), EU (London), US East (N. Virginia), US East (Ohio), and US West (Oregon).

PII identification and redaction for streaming jobs is performed only upon complete transcription of the audio segments.

For information about the types of PII that Amazon Transcribe can identify, see [redactable information \(p. 100\)](#).

Console

To use the Amazon Transcribe console to transcribe and redact PII from live speech, start the stream and begin speaking into the microphone on your computer.

To identify PII in a dictation using the Amazon Transcribe console

1. Sign into the [Amazon Transcribe console](#).
2. In the navigation pane, choose **Real-time transcription**.
3. For **Additional settings**, choose **PII redaction**.
4. Choose **Start streaming** and speak into the microphone.
5. Choose **Stop streaming** to end the dictation.

HTTP/2 stream

The following is the syntax for the parameters of an HTTP/2 request with PII identification or PII redaction activated.

```
POST /stream-transcription HTTP/2
x-amzn-transcribe-language-code: en-US
x-amzn-transcribe-sample-rate: MediaSampleRateHertz
x-amzn-transcribe-media-encoding: MediaEncoding
x-amzn-transcribe-vocabulary-name: VocabularyName
x-amzn-transcribe-session-id: SessionId
x-amzn-transcribe-vocabulary-filter-name: VocabularyFilterName
x-amzn-transcribe-vocabulary-filter-method: VocabularyFilterMethod
x-amzn-transcribe-language-model-name: LanguageModelName
x-amzn-transcribe-enable-channel-identification: EnableChannelIdentification
x-amzn-transcribe-number-of-channels: NumberOfChannels
x-amzn-transcribe-show-speaker-label: ShowSpeakerLabel
x-amzn-transcribe-enable-partial-results-stabilization: EnablePartialResultsStabilization
x-amzn-transcribe-partial-results-stability: PartialResultsStability
x-amzn-transcribe-content-identification-type: ContentIdentificationType (or x-amzn-
transcribe-content-redaction-type: ContentRedactionType)
x-amzn-transcribe-pii-entity-types: PiiEntityType
```

```
Content-type: application/json
```

To start an HTTP/2 stream with PII identification or PII redaction activated, use the [StartStreamTranscription \(p. 408\)](#) operation and specify the following:

- For `LanguageCode`, specify the language code for the language spoken in the stream. For US English, specify `en-US`.
- For `MediaSampleHertz`, specify the sample rate of the audio.
- For `ContentIdentificationType`, specify `PII`. Specify this field if you're identifying PII in the transcription results. You can only use this parameter if you **are not** using `content-redaction-type`.
- For `ContentRedactionType`, specify `PII`. Specify this field if you're redacting PII in the transcription results. You can only use this parameter if you **are not** using `content-identification-type`.
- For `PiiEntityTypes`, specify the types of PII you want to identify. You can specify one or more values. `pii-entity-types` is an optional parameter with a default of `ALL`; if you don't specify which entities you want to redact (or identify), all PII entity types are redacted (or identified).

The available values for `PiiEntityTypes` are: `BANK_ACCOUNT_NUMBER`, `BANK_ROUTING`, `CREDIT_DEBIT_NUMBER`, `CREDIT_DEBIT_CVV`, `CREDIT_DEBIT_EXPIRY`, `PIN`, `EMAIL`, `ADDRESS`, `NAME`, `PHONE`, `SSN`, and `ALL`. `PiiEntityTypes` is an optional parameter with a default value of `ALL`.

WebSocket stream

To start a WebSocket stream with PII Identification or PII redaction activated, use the following format to create a pre-signed URL.

```
wss://transcribestreaming.region.amazonaws.com:8443/stream-transcription-websocket?  
language-code=en-US  
&X-Amz-Algorithm=AWS4-HMAC-SHA256  
&X-Amz-Credential=Signature Version 4 credential scope  
&X-Amz-Date=date  
&X-Amz-Expires=time in seconds until expiration  
&X-Amz-Security-Token=security-token  
&X-Amz-Signature=Signature Version 4 signature  
&X-Amz-SignedHeaders=host  
&media-encoding=mediaEncoding  
&sample-rate=mediaSampleRateHertz  
&session-id=sessionId  
&enable-channel-identification=boolean  
&pii-entity-types=NAME, ADDRESS  
&content-redaction-type=PII (or &content-identification-type=PII)
```

Important

In the above code example, you can only have one of `&content-redaction-type=PII` or `&content-identification-type=PII`, not both.

To start a stream with PII Identification or PII redaction turned on, you must provide the following values:

- For `LanguageCode`, specify the language code for the language spoken in the stream. For US English, specify `en-US`; language codes are listed on the [What is Amazon Transcribe? \(p. 1\)](#) page.
- For `MediaSampleHertz`, specify the sample rate of the audio.
- For `content-identification-type`, specify `PII`.
- For `pii-entity-types`, specify the types of PII that you want to identify. You can specify one or more values. Note that `pii-entity-types` is an optional parameter with a default of `ALL`; if you don't specify which entities you want to redact (or identify), all PII entity types are redacted (or identified).

- The available values for `pii-entity-types` are: `BANK_ACCOUNT_NUMBER`, `BANK_ROUTING`, `CREDIT_DEBIT_NUMBER`, `CREDIT_DEBIT_CVV`, `CREDIT_DEBIT_EXPIRY`, `PIN`, `EMAIL`, `ADDRESS`, `NAME`, `PHONE`, `SSN`, or `ALL`

Example PII redaction and identification output

The following examples show PII-redacted output from batch and streaming jobs, and PII identification from a streaming job.

Example redacted batch output

```
{
    "jobName": "job id",
    "accountId": "account id",
    "isRedacted": true,
    "results": [
        "transcripts": [
            {
                "transcript": "Good morning, everybody. My name is [PII], and today I feel
like
sharing a whole lot of personal information with you. Let's start with my
Social
Security number [PII]. My credit card number is [PII] and my C V V code is
[PII].
I hope that Amazon Transcribe is doing a good job at redacting that
personal
information away. Let's check."
            }
        ],
        "items": [
            {
                "start_time": "2.86",
                "end_time": "3.35",
                "alternatives": [
                    {
                        "confidence": "1.0",
                        "content": "Good"
                    }
                ],
                "type": "pronunciation"
            },
            Items removed for brevity
            {
                "start_time": "5.56",
                "end_time": "6.25",
                "alternatives": [
                    {
                        "content": "[PII]",
                        "redactions": [
                            {
                                "confidence": "0.9999"
                            }
                        ]
                    }
                ],
                "type": "pronunciation"
            },
            Items removed for brevity
        ],
        "status": "COMPLETED"
    ]
}
```

```
}
```

Here's the unredacted transcript for comparison:

```
{
    "jobName": "job id",
    "accountId": "account id",
    "isRedacted": true,
    "results": {
        "transcripts": [
            {
                "transcript": "Good morning, everybody. My name is Mike, and today I feel
like
sharing a whole lot of personal information with you. Let's start with my
Social
Security number 000000000. My credit card number is 5555555555555555
and my C V V code is 000. I hope that Amazon Transcribe is doing a good
job
at redacting that personal information away. Let's check."
            }
        ],
        "items": [
            {
                "start_time": "2.86",
                "end_time": "3.35",
                "alternatives": [
                    {
                        "confidence": "1.0",
                        "content": "Good"
                    }
                ],
                "type": "pronunciation"
            },
            Items removed for brevity
            {
                "start_time": "5.56",
                "end_time": "6.25",
                "alternatives": [
                    {
                        "content": "Mike",
                        "redactions": [
                            {
                                "confidence": "0.9999"
                            }
                        ]
                    }
                ],
                "type": "pronunciation"
            },
            Items removed for brevity
        ],
        "status": "COMPLETED"
    }
}
```

Example redacted streaming output

```
{
    "TranscriptResultStream": {
        "TranscriptEvent": {
            "Transcript": {
                "Results": [

```

```
{
    "Alternatives": [
        {
            "Transcript": "my name is [NAME]",
            "Items": [
                {
                    "Content": "my",
                    "EndTime": 0.3799375,
                    "StartTime": 0.0299375,
                    "Type": "pronunciation"
                },
                {
                    "Content": "name",
                    "EndTime": 0.5899375,
                    "StartTime": 0.3899375,
                    "Type": "pronunciation"
                },
                {
                    "Content": "is",
                    "EndTime": 0.7899375,
                    "StartTime": 0.5999375,
                    "Type": "pronunciation"
                },
                {
                    "Content": "[NAME]",
                    "EndTime": 1.0199375,
                    "StartTime": 0.7999375,
                    "Type": "pronunciation"
                }
            ],
            "Entities": [
                {
                    "Content": "[NAME]",
                    "Category": "PII",
                    "Type": "NAME",
                    "StartTime": 0.7999375,
                    "EndTime": 1.0199375,
                    "Confidence": 0.9989
                }
            ]
        }
    ],
    "EndTime": 1.02,
    "IsPartial": false,
    "ResultId": "2db76dc8-d728-11e8-9f8b-f2801f1b9fd1",
    "StartTime": 0.0199375
}
]
```

Example PII identification output

PII identification is an additional feature that you can use with your streaming transcription job.

```
{
    "TranscriptResultStream": {
        "TranscriptEvent": {
            "Transcript": {
                "Results": [
                    {

```

```
"Alternatives": [
    {
        "Transcript": "my name is janet smithy",
        "Items": [
            {
                "Content": "my",
                "EndTime": 0.3799375,
                "StartTime": 0.0299375,
                "Type": "pronunciation"
            },
            {
                "Content": "name",
                "EndTime": 0.5899375,
                "StartTime": 0.3899375,
                "Type": "pronunciation"
            },
            {
                "Content": "is",
                "EndTime": 0.7899375,
                "StartTime": 0.5999375,
                "Type": "pronunciation"
            },
            {
                "Content": "janet",
                "EndTime": 0.9199375,
                "StartTime": 0.7999375,
                "Type": "pronunciation"
            },
            {
                "Content": "smithy",
                "EndTime": 1.0199375,
                "StartTime": 0.9299375,
                "Type": "pronunciation"
            }
        ],
        "Entities": [
            {
                "Content": "janet smithy",
                "Category": "PII",
                "Type": "NAME",
                "StartTime": 0.7999375,
                "EndTime": 1.0199375,
                "Confidence": 0.9989
            }
        ]
    }
],
"EndTime": 1.02,
"IsPartial": false,
"ResultId": "2db76dc8-d728-11e8-9f8b-f2801f1b9fd1",
"StartTime": 0.0199375
}
]
}
}
```

Improving transcription accuracy with custom language models

Use *custom language models* to train and develop language models that are domain-specific. For example, you can use custom language models to improve transcription performance for domains such as legal, hospitality, finance, and insurance. Although the general model provided by Amazon Transcribe works well in most instances, custom language models might produce even more accurate results.

To train a custom language model, you must upload text data from your specific use case to Amazon Simple Storage Service (Amazon S3), provide Amazon Transcribe with permission to access the data, and choose a *base model*. A base model is a general speech recognition model, which you customize with your text data.

Custom language models use your text data to improve transcription accuracy for your use case. Your text data can include domain-specific text or audio transcripts. Domain-specific text data includes website content, instruction manuals, and technical documentation. Audio transcript data consists of ground-truth audio transcripts. Ground-truth audio transcripts have been processed with very high accuracy and are the ideal representation of the source audio.

Note

Custom language models are not available with all Amazon Transcribe-supported languages; see [Supported languages and language-specific features \(p. 3\)](#).

You must provide text data that represents the audio you want to transcribe. The domain-specific data you provide must be related to your use case, and the audio transcript data you provide should be similar to the audio you want to transcribe. Any potential improvement in transcription accuracy depends on how closely your text data represents your audio and how much text data you provide. The quality of your text data is much more important than its quantity in creating custom language models that generate accurate transcriptions.

You upload your text data to Amazon Simple Storage Service (Amazon S3) and then give Amazon Transcribe access to the S3 buckets that contain the data. After uploading your data to Amazon S3, you choose a *base model*. A base model is a general speech recognition model that you customize with your text data.

There are two ways you can upload your text data to create a custom language model:

1. Upload your text as *training data*. You use training data to train your custom language model for your specific use case.
2. Upload your domain-specific text as training data and your audio transcripts as *tuning data*. You use tuning data to optimize your custom language model and increase its transcription accuracy.

Use the following table to determine how to upload your data.

If you have	Upload this
A large amount of domain-specific text and a much smaller amount of audio transcript text data	Domain-specific text as training data. Upload your transcription text as tuning data.
A minimum of 10,000 words of audio transcript text	Audio transcript text as training data.

If you have	Upload this
At least 100,000 words of audio transcript text and a large amount of additional domain-specific text	Audio transcript text as training data. Typically, this method leads to the greatest possible increase in transcription accuracy. Follow the first method described in this table if this method doesn't produce the desired increase in transcription accuracy.
Domain-specific text only	Domain-specific text as training data. We recommend any of the preceding methods of uploading your data over this one.

You can provide up to 2 GB of training data and 200 MB of tuning data.

If you have enough text that represents the audio you want to transcribe, training custom language models can produce significant improvements in accuracy over using [Custom vocabularies \(p. 8\)](#). Custom vocabularies improve the ability of Amazon Transcribe to recognize terms without using the context in which they're spoken. Custom language models not only recognize individual terms, but additionally use each term's context to transcribe your audio.

Custom language models can also add words to their recognition vocabularies automatically, eliminating the need for you to manually input new words. For the highest possible transcription accuracy, you can use custom vocabularies with your custom language model.

You can't use AWS Key Management Service (AWS KMS) to encrypt your training data, but you can use AWS KMS condition keys to encrypt your transcription output. For information about condition keys, see [Key management \(p. 163\)](#).

To use custom language models in your transcription jobs, you do the following:

- Prepare and upload plain text data
- Provide Amazon Transcribe with permissions to access your data
- Create a custom language model
- Use a custom language model in a transcription job
- View and update your custom language models to take advantage of speech recognition improvements in Amazon Transcribe

Topics

- [Step 1: Preparing the data \(p. 111\)](#)
- [Step 2: Providing Amazon Transcribe with data permissions \(p. 112\)](#)
- [Step 3: Creating a custom language model \(p. 114\)](#)
- [Step 4: Transcribing with a custom language model \(p. 116\)](#)
- [Step 5: Viewing and updating your custom language models \(p. 118\)](#)

Step 1: Preparing the data

Create a custom language model by providing training data in plain text format and by choosing a base model. You can also provide additional tuning data, also in plain text format, for optimization.

To prepare your text data:

1. Properly format it and save it in one or more text files. Make sure each text file:

- Is in the same language as the model you want to create. For example, if you want to create a custom language model that transcribes audio in US English (en-US), your text data must also be in US English (en-US).
- Is in plain text (it's not a file such as a Microsoft Word document, comma-separated value file, or PDF).
- Is encoded in UTF-8.
- Doesn't contain any formatting characters, such as HTML tags.
- Is less than 2 GB in size if you intend to use the file as training data. You can provide a maximum of 2 GB of training data.
- Is less than 200 MB in size if you intend to use the file as tuning data. You can provide a maximum of 200 MB of optional tuning data.

Note

Although not a requirement, it is best practice to have only one sentence per line within your text file.

2. Upload those files to Amazon Simple Storage Service (Amazon S3). If you intend to tune the model, store your tuning data in a separate S3 bucket from the one you use for your training data.

Use your own data processing pipelines to prepare your plain text files. If you're extracting text from HTML, remove the HTML tags and unescape the entities.

The following example shows how to format sentences in a text file:

```
Ribosomes help translate RNA into protein.  
RISC is essential in RNA interference.  
Interferon type 1 signaling proteins help prevent viruses from replicating their RNA or  
DNA.  
...
```

It doesn't matter how many text files you use to upload your training or tuning data. For model training, you get the same improvements in transcription accuracy if you use one file with 100,000 words or 10 files with 10,000 words. Prepare your text data in a way that's most convenient for you.

Next step

[Step 2: Providing Amazon Transcribe with data permissions \(p. 112\)](#)

Step 2: Providing Amazon Transcribe with data permissions

To create a custom language model, you need to give Amazon Transcribe access to your text data. To do this, give Amazon Transcribe the `GetObject` and `ListBucket` permissions to your Amazon Simple Storage Service (Amazon S3) bucket.

To provide data permissions

1. Sign in to AWS Management Console and open the IAM console at [IAM console](#).
2. In the navigation pane, for **Access management**, choose **Roles** and choose one of the following options.
 - Choose the name of the IAM role to use to create your custom language model and run transcription jobs.

- To create a new IAM role, choose **Create role**.
 - a. Under **Common use cases**, choose **EC2**. You can select any use case, but EC2 is one of the most straightforward ones.
 - b. Choose **Next: Tags**.
 - c. Choose **Next: Review**.
 - d. Specify a name for the role under **Role name**. Remove the text under **Role description**.
 - e. Choose **Create role**.
 - f. Choose the role you've created.
- 3. Choose **Trust relationships**.
- 4. Choose **Edit trust relationship**.
- 5. If you're using an existing role, modify your trust policy with the following code. If you've created a new role using this procedure, replace the trust policy text with the following code.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "transcribe.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- 6. In the navigation pane, choose **Policies**.
- 7. • From the policy list, choose **AmazonTranscribeFullAccess**.
 - a. From **Policy actions**, choose **Attach policy**
 - b. Choose the IAM role you want to attach the policy to.
- Choose **Create policy**.
 - a. Choose **JSON**.
 - b. Enter the following code, which adds **GetObject** and **ListBucket** permissions:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::your-input-bucket/*"
      ],
      "Effect": "Allow"
    },
    {
      "Action": [
        "s3>ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::your-input-bucket"
      ]
    }
  ]
}
```

```
        "arn:aws:s3:::your-input-bucket"  
    ],  
    "Effect": "Allow"  
}  
]  
}
```

- c. Choose **Review policy**.
- d. Specify a name for the policy.
- e. Choose **Create policy**.
- f. Attach the policy to the IAM role.

Amazon Transcribe now has the necessary permissions to access the data for your custom language model.

Next step

[Step 3: Creating a custom language model \(p. 114\)](#)

Step 3: Creating a custom language model

To create a custom language model, you must provide your text data using an Amazon S3 prefix and specify a *base model*. There are two base models:

- **NarrowBand** - For audio with a sample rate of less than 16 kHz. You typically use this type of model for telephone conversations recorded at 8 kHz.
- **WideBand** - For audio with a sample rate of 16 kHz or greater. This includes audio from media sources.

You can create a custom language model using the base model and training data in the [Amazon Transcribe console](#), and the [CreateLanguageModel \(p. 283\)](#) API.

Using prefixes in Amazon Simple Storage Service to retrieve your data

To create a custom language model, you use prefixes in Amazon Simple Storage Service to specify the training and the optional tuning data. To understand how to use prefixes, you need to be familiar with the following Amazon S3 concepts:

- **Bucket** – A container to store your objects.
- **Object** – The entity stored in the S3 bucket. In this case, it's your training or tuning text files.
- **Key** – The unique identifier for an object within a bucket.
- **Prefix** – Any portion of the key up to the final delimiter. You use prefixes to organize your data and specify objects in the S3 bucket.

You store an object, your text file, in a bucket with a key that uniquely identifies the file.

For example, the key `s3://DOC-EXAMPLE-BUCKET1/myfiles/2020/may/file.txt` uniquely identifies a text file in an S3 bucket.

The prefix can be any part of the key that comes before its final delimiter.

For example, the key `myfiles/2020/may/file.txt` has the following prefixes:

- `s3://DOC-EXAMPLE-BUCKET1/myfiles/2020/may/`
- `s3://DOC-EXAMPLE-BUCKET1/myfiles/2020/`
- `s3://DOC-EXAMPLE-BUCKET1/myfiles/`

You can use any of the preceding prefixes to access `file.txt`. For more information on prefixes, see [Organizing objects using prefixes](#).

When you use the [CreateLanguageModel \(p. 283\)](#) API, you specify prefixes for your text data in the following fields:

- `S3Uri` for the training data
- Optional: `TuningDataS3Uri` for the tuning data

Amazon Transcribe uses any object whose key matches one of the prefixes you specify in a custom language model. Amazon Transcribe returns an error for any file that matches a prefix and is not a plain text file.

You create a custom language model by providing prefixes to train a base model using either the console or the API.

Console

To create a custom language model using the console

To use the console to create a custom language model with the console, you must have your training data stored in an Amazon S3 bucket.

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, choose **Custom language models**.
3. Choose **Train model**.
4. For **Name**, enter a name for your custom language model that is unique within your AWS account.
5. For **Language**, choose the language of your custom language model.
6. For **Base model**, choose **Narrow band** or **Wide band**, as appropriate for the sample rate of the audio you want to transcribe.
7. Under **Training data**, for **Training data location on S3**, specify the S3 prefix that accesses only your training data.
8. Optional: Under **Tuning data - optional**, for **Tuning data location on S3**, specify the S3 prefix for the bucket where your tuning data is stored.
9. For **Access permissions**, use or create an IAM data access role that provides Amazon Transcribe with the `ListBucket` and `GetObject` permissions.
10. Choose **Train model**.

AWS SDK

To create a custom language model using the AWS SDK

- In an [CreateLanguageModel \(p. 283\)](#) request, specify the following:
 - `BaseModelName` - The type of base model you want to use for your custom language model
 - `InputDataConfig` - Specify the Amazon S3 object location and the IAM data access role for your training data:

`DataAccessRoleARN` - The Amazon Resource Name (ARN) that identifies the permissions to your Amazon S3 bucket.

`S3Uri` - The prefix of the keys to your training data.

(Optional) `TuningDataS3URI` - The prefix of the keys to your tuning data.

- `LanguageCode` - The language code for the language of your training data.
- `ModelName` - The name of your custom language model.

The following is an example request using the AWS SDK for Python (Boto3).

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
model_name = "example-language-model"
transcribe.create_language_model(
    LanguageCode = 'language-code',
    BaseModelName = 'base-model-name',
    ModelName = model_name,
    InputDataConfig = {'S3Uri': 's3://DOC-EXAMPLE-BUCKET/clm-training/',
                      'TuningDataS3Uri': 's3://DOC-EXAMPLE-BUCKET/clm-tuning',
                      'DataAccessRoleArn': 'arn:aws:iam::AWS-account-number:role/IAM role'
                     }
)
```

AWS CLI

To create a custom language model using AWS CLI

- Run the following code.

```
aws transcribe create-language-model \
--language-code language-code \
--base-model-name base-model-name \
--model-name example-model-name \
--input-data-config S3Uri="s3://example-bucket",DataAccessRoleArn="arn:aws:iam::aws-
account-number:role/IAM role"
```

Next step

[Step 4: Transcribing with a custom language model \(p. 116\)](#)

Step 4: Transcribing with a custom language model

You can use a custom language model in transcription jobs with the [Amazon Transcribe console](#) or the [StartTranscriptionJob \(p. 376\)](#) API.

Transcribing with a custom language model (console)

To start a transcription job (console)

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, choose **Transcription jobs**.
3. For **Name**, enter a transcription job name that is unique within your AWS account.
4. For **Language**, choose the language of your media file.
5. For **Custom model selection**, choose your custom language model.
6. For **Input file location on S3**, enter the URI of the media file. If you can't remember the URI, choose **Browse S3** and choose it.
7. Choose **Next**.
8. Enable any of the available features you want to use for your transcription job.
9. Choose **Create**.

Transcribing with a custom language model (API)

To start a transcription job (API)

- For the [StartTranscriptionJob \(p. 376\)](#) API, specify the following.
 - a. For **TranscriptionJobName**, specify a name that is unique to your AWS account.
 - b. For **LanguageCode**, specify the language code that corresponds to the language spoken in your audio or video file.
 - c. For the **MediaFileUri** parameter of the **Media** object, specify Amazon S3 location of the file you want to transcribe.
 - d. For the **LanguageModelName** parameter of the **ModelSettings** object, specify the name of the custom language model.

The following is an example request using the AWS SDK for Python (Boto3).

```
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = 'transcription-job-name'
job_uri = 's3://path-to-your-media-file/media-file.file-extension'

transcribe.start_transcription_job(
    TranscriptionJobName = job_name,
    Media = {'MediaFileUri': job_uri},
    MediaFormat = 'media-format',
    LanguageCode = 'language-code',
    ModelSettings = {
        'LanguageModelName': 'language-model-name'
    }
)
while True:
    status = transcribe.get_transcription_job(TranscriptionJobName=job_name)
    if status['TranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED', 'FAILED']:
        break
    print('Not ready yet...')
    time.sleep(5)
print(status)
```

Transcribing with a custom language model (AWS CLI)

To transcribe with a custom language model (AWS CLI)

- Run the following commands.

```
aws transcribe start-transcription-job \
--transcription-job-name "example-job-name" \
--language-code "language-code" \
--media MediaFileUri="s3://example-bucket/example-audio.wav" \
--model-settings LanguageModelName="ExampleLanguageModel"
```

Next step

[Step 5: Viewing and updating your custom language models \(p. 118\)](#)

Step 5: Viewing and updating your custom language models

The speech recognition capabilities of Amazon Transcribe are constantly improving. Specifically, Amazon Transcribe continually updates the base models used in custom language models.

To see if you're running the latest base model in your custom language model:

1. Use the [DescribeLanguageModel \(p. 315\)](#) API and check the `UpgradeAvailability` field.
2. If `UpgradeAvailability` is `true`, you're not running the latest version of the base model in your custom language model.

To use the latest base model in a custom language model, you must create a new custom language model. That custom language model will have the updated base model.

Call analytics

Call analytics helps you gain insight into the content and sentiment of your recordings, which you can use to create a better customer experience and improve product development. Using call analytics, you can:

- **Identify the sentiment of each participant:** Sentiment analysis tells you how the customer and agent were feeling during the call (positive, neutral, or negative).
- **Perform automated call categorization:** Set flags for key words and phrases, sentiment, non-talk time, or interruptions.
- **Perform call issue detection:** Identify the reason for the call (for example, did the customer call about the wrong item being delivered or was their item damaged in transit?).
- **Redact sensitive data:** Replace personally identifiable information (PII) in both the text transcript and the audio file. A redacted transcript will replace PII with [PII]; a redacted audio file will replace PII with silence. To learn more about redaction with Amazon Transcribe, see [Redacting or identifying personally identifiable information \(p. 100\)](#).
- **Extract conversation characteristics:** Identify the amount of non-talk time, number of interruptions, talk speed, or sentiment trends.

For a list of languages that support call analytics, refer to [Supported languages and language-specific features \(p. 3\)](#).

Using the call analytics **categories** feature, you can create categories with up to 20 rules. Amazon Transcribe then uses the rules you've defined within a category to identify phrases or conversational characteristics within your transcript that match your rules. For example, you can create a category with the following rules:

- *Rule 1:* The agent didn't say the welcoming phrase "How are you today?" during the first two minutes of the call.
- *Rule 2:* There's silence (a period when neither the customer nor agent spoke) that lasted longer than one minute.

If you run a call analytics job and the characteristics of that job match the rules you've specified in a category, Amazon Transcribe automatically flags your job with that category.

You can create as many categories as you'd like to cover a range of different situations. Using a variety of categories can help you determine how your rules and/or categories may correlate. For example, you might hypothesize that an agent not using a greeting is correlated with negative customer sentiment. To test this hypothesis, you can create two separate categories: category 1 has a rule for an agent not using a specific greeting; category 2 has a rule that the customer sentiment during the call was negative for the last five minutes. When you run your analytics jobs, you can then see how these categories correlate, if at all.

Considerations when using call analytics:

- Job queuing is enabled by default.
- If content redaction is enabled, you can only use call analytics in U.S. English (en-US).
- Input files for call analytics cannot be greater than 500 MB and must be less than 4 hours.

- A call analytics job only uses the categories you've previously created. Any new categories cannot be applied to previous call analytics jobs.

For call analytics quotas, refer to [Guidelines and quotas \(p. 54\)](#).

To use call analytics in your transcription job, use [StartCallAnalyticsJob \(p. 365\)](#) or **call analytics job** in the console. Call analytics output is similar to that of a standard transcription job, but contains additional data based on the analytics categories you create. To learn how to set up a call analytics job, see [Setting up call analytics transcription jobs \(p. 122\)](#). For example output, see [Sample call analytics output \(p. 127\)](#).

Tip

To view sample call analytics output and features, check out our [GitHub demo](#).

Using categories and rules

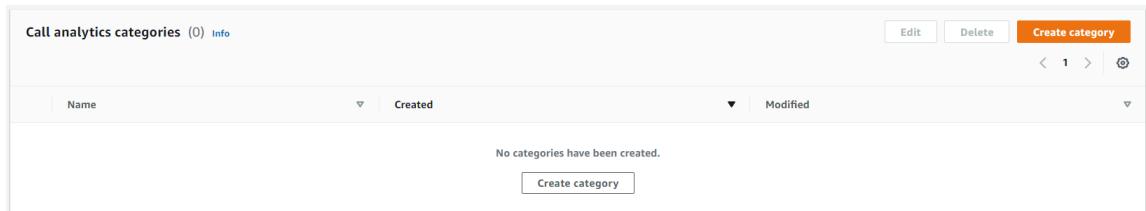
You can use call analytics categories to flag particular conversational characteristics in your call analytics job. For each category you create, you must create between 1 and 20 rules. Each rule contains one *filter* to identify criteria applicable to a category. You can create filters for the following:

- Non-talk time: When neither the customer nor the agent is talking.
- Interruptions: When either the customer or the agent is interrupting the other person.
- Customer or agent sentiment: How either the customer or the agent is feeling during a specified time period. If at least 50 percent of the conversation turns (the back-and-forth between two speakers) in a specified time period match the specified sentiment, Amazon Transcribe will consider the sentiment a match.
- Call categorization: Matches part of the transcription based on an exact phrase. For example, if you filter for the customer saying "I want to speak to the manager", Amazon Transcribe filters for that exact phrase.

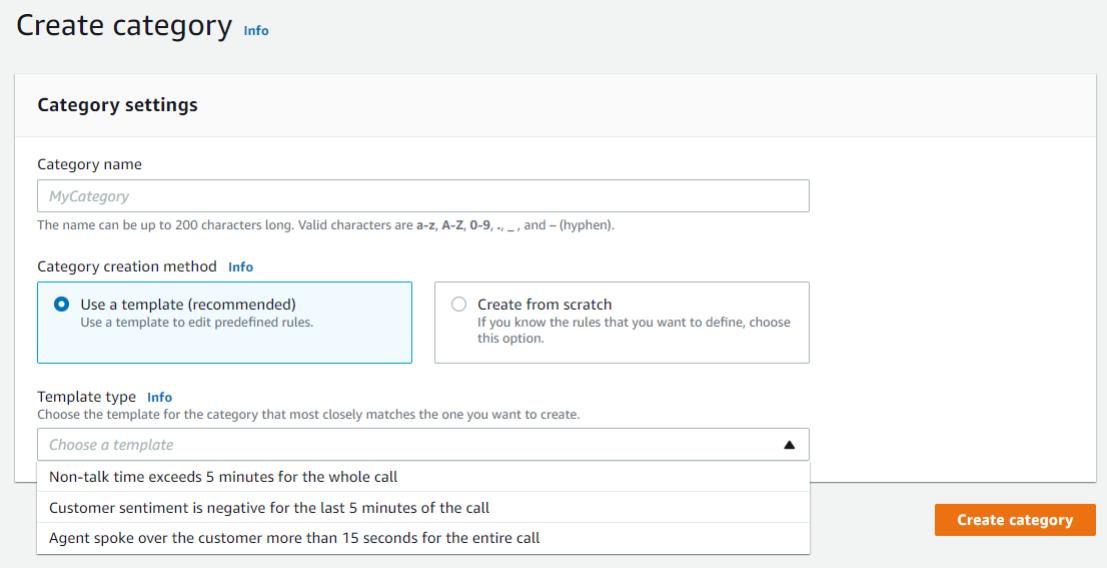
To create a category, you can use the **Amazon Transcribe Console**, **AWS CLI**, or **AWS SDK**; see below for instructions:

Console

1. In the navigation pane, under Amazon Transcribe, choose **Amazon Transcribe Call Analytics**.
2. Choose **Call analytics categories**, which takes you to the **Call analytics categories** page. Click the **Create category** button.



3. You're now on the **Create category page**. Enter a name for the category, then choose whether you're going to use a template to create a category.



4. Choose **Create category.**

AWS SDK for Python (Boto3)

The following example uses the AWS SDK for Python (Boto3) to create a category by using the `CategoryName` and `Rules` arguments for the [create_call_analytics_category](#) method:

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
category_name = "example-call-analytics-category"
transcribe.create_call_analytics_category(
    CategoryName=category_name,
    Rules=[{'SentimentFilter': {'ParticipantRole': 'AGENT',
                               'Sentiments': ['NEGATIVE']}}
)
result = transcribe.get_call_analytics_category(CategoryName=category_name)
print(result)
```

This example creates a category that filters for a negative agent sentiment.

AWS CLI

This example uses the `create-call-analytics-category` command.

This example creates a category that filters for a negative agent sentiment:

```
aws transcribe create-call-analytics-category \
--category-name your-category-name \
--rules "SentimentFilter={ParticipantRole=AGENT,Sentiments=[NEGATIVE]}"
```

Here's another example using the `create-call-analytics-category` command, and a request body that adds several rules to your category.

```
aws transcribe create-call-analytics-category \
--cli-input-json file://filepath/example-start-command.json
```

The file `example-start-command.json` contains the following request body.

```
{  
    "CategoryName": "example-call-analytics-category",  
    "Rules": [  
        {  
            "InterruptionFilter": {  
                "AbsoluteTimeRange": {  
                    "First": 60000  
                },  
                "Negate": true,  
                "ParticipantRole": "CUSTOMER",  
                "Threshold": 10000  
            }  
        },  
        {  
            "NonTalkTimeFilter": {  
                "Negate": false,  
                "RelativeTimeRange": {  
                    "EndPercentage": 80,  
                    "StartPercentage": 10  
                },  
                "Threshold": 20000  
            }  
        },  
        {  
            "SentimentFilter": {  
                "ParticipantRole": "AGENT",  
                "Sentiments": [  
                    "NEGATIVE"  
                ]  
            }  
        },  
        {  
            "TranscriptFilter": {  
                "AbsoluteTimeRange": {  
                    "First": 10000  
                },  
                "Targets": [  
                    "welcome",  
                    "hello"  
                ],  
                "TranscriptFilterType": "EXACT"  
            }  
        }  
    ]  
}
```

The above example creates a category with the rules:

- The customer was not interrupted in the first 60,000 milliseconds.
- No one was speaking between 10% into the call through 80% into the call.
- The agent had a negative sentiment.
- The words "welcome" or "hello" were used in the first 10,000 milliseconds of the call.

Setting up call analytics transcription jobs

To get analytics data, you must run call analytics jobs. Call analytics jobs not only transcribe the audio of the recordings between your agents and customers, but they provide meaningful information (in the

form of *analytics*) about these conversations. If you choose not to create any categories before running a call analytics job, the output only includes the following information:

- Detected customer issues
- Sentiment of the agent and the customer
- Non-talk time
- Interruptions
- Speaking volume

If you create categories, you can flag your analytics job based on the conversational characteristics you define. These conversational characteristics can include:

- Customer or agent sentiment during specific periods of time
- Matching a portion of the transcript to a phrase you've specified
- The customer or agent interrupting the other person
- Neither the customer nor the agent spoke for a defined period of time

Once you create categories, Amazon Transcribe flags your call analytics jobs with a label indicating that the call matches all the rules you've defined in a given category. For more information on creating categories, see [Using categories and rules \(p. 120\)](#).

Note

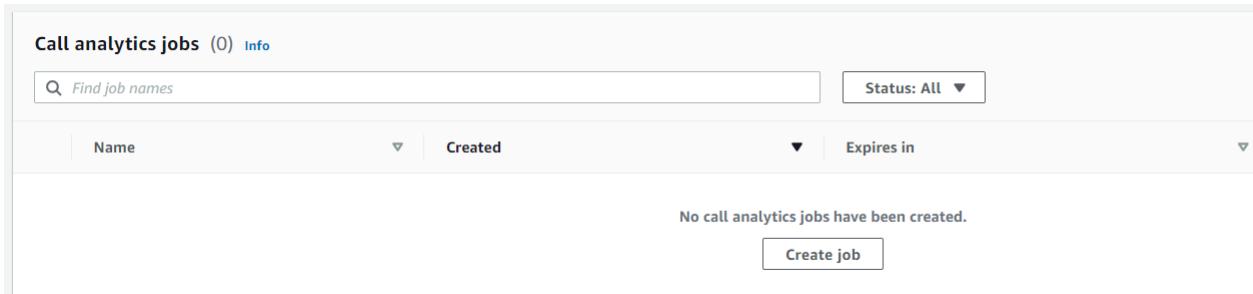
Call analytics jobs can't be classified retroactively with categories. Only the categories you created *before* running a call analytics job can be applied to that job.

To start a call analytics job, you can use the **Amazon Transcribe Console**, **AWS CLI**, or **AWS SDK**; see below for instructions:

Console

Use the following procedure to start a call analytics job. The calls that match all characteristics defined by a category are labeled with that category.

1. In the navigation pane, under Amazon Transcribe Call Analytics, choose **Call analytics jobs**.
2. Choose **Create job**.



3. On the **Specify job details** page, provide information about your call analytics job, including the location of your input data.

Specify job details [Info](#)

Job settings

Name
 [Info](#)
The name can be up to 200 characters long. Valid characters are a-z, A-Z, 0-9, . (period), _ (underscore), and – (hyphen).

Model type [Info](#)
Choose the type of model to use for the transcription job.

General model
To use a model that is not specialized for a particular use case, choose this option. Configuration options vary between languages.

Custom language model
To use a model that you trained for your specific use case, choose this option. This model has fewer configuration options than the general model.

Language settings
You can transcribe your audio file in a language that you specify or have Amazon Transcribe identify and transcribe it in the predominant language.

Specific language [Info](#)
If you know the language spoken in your source audio, choose this option to get the most accurate results. The options available for additional processing vary between languages.

Automatic language identification [Info](#)
If you don't know the language spoken in your audio files, choose this option. You have access to fewer options for additional processing than if you choose **Specific language**.

Language
Choose the language of the input audio.
 [▼](#)

Input data [Info](#)

Input file location on S3
Choose an input audio or video file in Amazon S3.
 [Browse S3](#)
Valid file formats: MP3, MP4, WAV, FLAC, AMR, OGG, and WebM.

Agent audio channel identification [Info](#)
Choose the channel that has the speech from the agent. The other channel is used for the customer's speech.
 [▼](#)

Specify the desired S3 location of your output data and which IAM role to use.

The screenshot shows two configuration steps in a wizard:

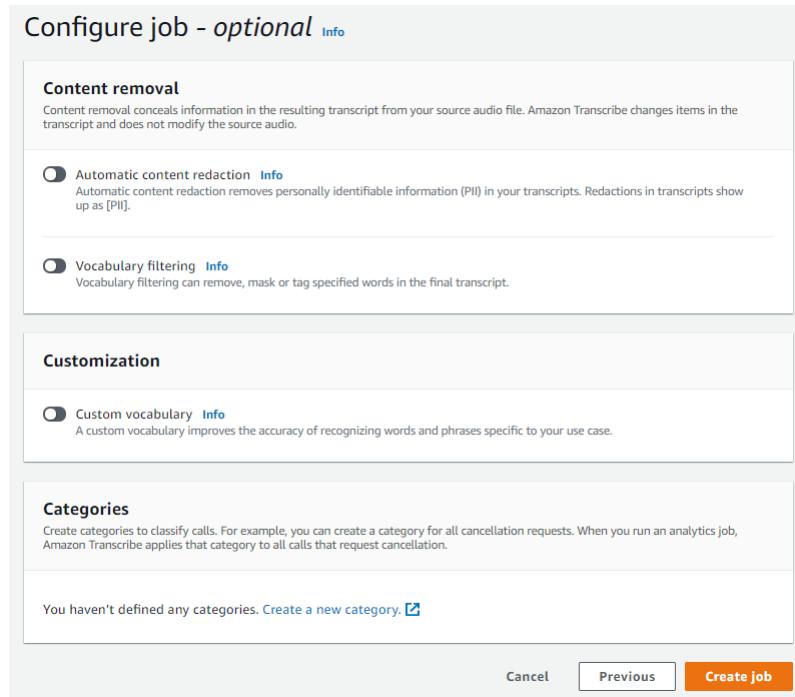
Output data (Step 1):
Output data location type info [Info](#)
 Service-managed S3 bucket
The output will be removed after 90 days when the job expires.
 Customer specified S3 bucket
The output will not be removed from bucket even after the job expires.

Access permissions (Step 2):
IAM role [Info](#)
 Use an existing IAM role
 Create an IAM role
By choosing **Create job** you are authorizing creation of this role.
Permissions to access
Your role has access to these resources. The KMS key permission is used only if your input bucket is encrypted
 Input S3 bucket and KMS decrypt permission to input bucket
 Any S3 bucket and any KMS keys
Role name
Roles are prefixed with "AmazonTranscribeServiceRoleFullAccess-". Your newly created role has full access to the S3 bucket and KMS key for your account.
MyTranscribeRole
The name can be up to 64 characters long
▼ Role permissions details
Your new role has these permissions to give Amazon Transcribe access to the resources that you've specified.

Service	Access level	Resource
S3	List, Read, Write	All resources
Key Management Service	GenerateDataKey, Decrypt	All resources

Cancel **Next**

4. Choose **Next**.
5. For **Configure job**, turn on any optional features you want to include with your call analytics job.



6. Choose **Create job**.

AWS SDK for Python (Boto3)

The following example uses the AWS SDK for Python (Boto3) to start a call analytics job using the `start_call_analytics_job` method:

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "your-call-analytics-job-name"
job_uri = "s3://your-S3-bucket/S3-prefix/your-filename.file-extension"
output_location = "s3://your-S3-bucket/S3-prefix/"
data_access_role = "arn:aws:iam::account-id:role/your-IAM-role"
transcribe.start_call_analytics_job(
    CallAnalyticsJobName=job_name,
    Media={'MediaFileUri': job_uri},
    DataAccessRoleArn=data_access_role,
    OutputLocation=output_location,
    ChannelDefinitions=[{'ChannelId': 0, 'ParticipantRole': 'AGENT'}, {'ChannelId': 1, 'ParticipantRole': 'CUSTOMER'}]
)
while True:
    status = transcribe.get_call_analytics_job(CallAnalyticsJobName=job_name)
    if status['CallAnalyticsJob']['CallAnalyticsJobStatus'] in ['COMPLETED', 'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)
```

AWS CLI

This example uses the `start-call-analytics-job` command and `channel-definitions` parameter.

```
aws transcribe start-call-analytics-job \
--call-analytics-job-name your-job-name \
--media MediaFileUri=s3://your-S3-bucket/S3-prefix/your-filename.file-extension \
--language-code en-US \
--data-access-role-arn "arn:aws:iam::accountID:role/your-IAM-role" \
--channel-definitions '[{"ChannelId": 0, "ParticipantRole": "AGENT"}, {"ChannelId": 1, "ParticipantRole": "CUSTOMER"}]'
```

Here's another example using the [start-call-analytics-job](#) command, and a request body that identifies audio channels for that job.

```
aws transcribe start-call-analytics-job \
--cli-input-json file://filepath/example-start-command.json
```

The file *example-start-command.json* contains the following request body.

```
{
    "CallAnalyticsJobName": "Example-Call-Analytics-Job",
    "OutputLocation": "s3://your-S3-bucket/",
    "DataAccessRoleArn": "arn:aws:iam::accountID:role/your-IAM-role",
    "Media": {
        "MediaFileUri": "s3://your-S3-bucket/S3-prefix/your-filename.file-extension"
    },
    "ChannelDefinitions": [
        {
            "ChannelId": 0,
            "ParticipantRole": "AGENT"
        },
        {
            "ChannelId": 1,
            "ParticipantRole": "CUSTOMER"
        }
    ]
}
```

Sample call analytics output

Below is an example of redacted JSON output from a call analytics job between a customer and an agent. Note that in addition to redacted content (shown as [PII]), there are loudness scores and sentiment values (negative, positive, or neutral) associated with each segment of the dialogue.

```
{
    "JobStatus": "COMPLETED",
    "LanguageCode": "en-US",
    "Transcript": [
        {
            "LoudnessScores": [
                87.87,
                85.25,
                83.46,
                81.37,
                80.37,
                75.91
            ],
            "Content": "Hey Mr [PII]? Um I think I lost my debit card or it was stolen. I don't know.",
            "IssuesDetected": [
                {
                    "IssueType": "FraudulentActivity"
                }
            ]
        }
    ]
}
```

```

        "CharacterOffsets": {
            "Begin": 27,
            "End": 63
        }
    },
    "Redaction": {
        "RedactedTimestamps": [
            [
                {
                    "BeginOffsetMillis": 6950,
                    "EndOffsetMillis": 7470
                }
            ]
        ],
    },
    Items removed for brevity
],
"Id": "cccccccc-cccc-cccc-cccc-cccccccccccc",
"BeginOffsetMillis": 6040,
"EndOffsetMillis": 11650,
"Sentiment": "NEUTRAL",
"ParticipantRole": "CUSTOMER"
},
{
    "LoudnessScores": [
        86.09,
        83.63,
        86.11,
        85.33,
        74.4,
        19.55,
        64.23,
        81.19,
        89.7
    ],
    "Content": "Uh I am very sorry to hear that. Um May I ask who I'm speaking with
[PII].",
    "Redaction": {
        "RedactedTimestamps": [
            [
                {
                    "BeginOffsetMillis": 19500,
                    "EndOffsetMillis": 20230
                }
            ]
        ],
        [
            "BeginOffsetMillis": 19500,
            "EndOffsetMillis": 20230
        ],
    },
    Items removed for brevity
],
"Id": "bbbbbbbb-bbbb-bbbb-bbbb-bbbbbbbbbb",
"BeginOffsetMillis": 12040,
"EndOffsetMillis": 20230,
"Sentiment": "NEGATIVE",
"ParticipantRole": "AGENT"
},
{
    "LoudnessScores": [

```

```

    78.49,
    84.47,
    81.35
],
"Content": "Yeah, this is [PII].",
"Redaction": {
    "RedactedTimestamps": [
        [
            [
                {
                    "BeginOffsetMillis": 17420,
                    "EndOffsetMillis": 18460
                }
            ]
        ],
    },
Items removed for brevity
],
"Id": "aaaaaaaa-aaaa-aaaa-aaa-aaaaaaaaaaaa",
"BeginOffsetMillis": 16840,
"EndOffsetMillis": 18460,
"Sentiment": "NEUTRAL",
"ParticipantRole": "CUSTOMER"
},
],
"AccountId": "123456789012",
"Categories": {
    "MatchedDetails": {
        "matchedCategoryA": {
            "PointsOfInterest": [
                {
                    "BeginOffsetMillis": 440,
                    "EndOffsetMillis": 4960
                }
            ]
        }
    },
    "MatchedCategories": ["matchedCategoryA"]
},
"Channel": "VOICE",
"JobName": "call-analytics-job-name",
"Participants": [
    {
        "ParticipantRole": "AGENT"
    },
    {
        "ParticipantRole": "CUSTOMER"
    }
],
"ConversationCharacteristics": {
    "NonTalkTime": {
        "Instances": [
            {
                "BeginOffsetMillis": 216450,
                "DurationMillis": 7690,
                "EndOffsetMillis": 224140
            }
        ],
        "TotalTimeMillis": 7690
    },
    "Interruptions": {
        "TotalCount": 4,
        "TotalTimeMillis": 12129,
        "InterruptionsByInterrupter": {
            "AGENT": [
                {
                    "BeginOffsetMillis": 30940,
                    "DurationMillis": 2800,
                    "EndOffsetMillis": 33740
                },
                {
                    "BeginOffsetMillis": 133440,
                    "DurationMillis": 4510,
                    "EndOffsetMillis": 137950
                }
            ]
        }
    }
}

```

```

        },
        {
            "BeginOffsetMillis": 163240,
            "DurationMillis": 2710,
            "EndOffsetMillis": 165950
        },
        {
            "BeginOffsetMillis": 169840,
            "DurationMillis": 2110,
            "EndOffsetMillis": 171950
        }
    ]
}
},
"TotalConversationDurationMillis": 279060,
"Sentiment": {
    "OverallSentiment": {
        "AGENT": 2.7,
        "CUSTOMER": 0.2
    },
    "SentimentByPeriod": {
        "QUARTER": [
            "AGENT": [
                {
                    "Score": 2.1,
                    "BeginOffsetMillis": 0,
                    "EndOffsetMillis": 69765
                },
                {
                    "Score": -0.7,
                    "BeginOffsetMillis": 69765,
                    "EndOffsetMillis": 139530
                },
                {
                    "Score": 5.0,
                    "BeginOffsetMillis": 139530,
                    "EndOffsetMillis": 209295
                },
                {
                    "Score": 3.0,
                    "BeginOffsetMillis": 209295,
                    "EndOffsetMillis": 279060
                }
            ],
            "CUSTOMER": [
                {
                    "Score": -2.0,
                    "BeginOffsetMillis": 0,
                    "EndOffsetMillis": 69660
                },
                {
                    "Score": 0.0,
                    "BeginOffsetMillis": 69660,
                    "EndOffsetMillis": 139320
                },
                {
                    "Score": 0.0,
                    "BeginOffsetMillis": 139320,
                    "EndOffsetMillis": 208980
                },
                {
                    "Score": 2.1,
                    "BeginOffsetMillis": 208980,
                    "EndOffsetMillis": 278640
                }
            ]
        }
    }
},
"TalkSpeed": {
    "DetailsByParticipant": {
        "AGENT": {
            "AverageWordsPerMinute": 195
        },
        "CUSTOMER": {
            "AverageWordsPerMinute": 116
        }
    }
},

```

```
"TalkTime": {  
    "DetailsByParticipant": {  
        "AGENT": {  
            "TotalTimeMillis": 158120  
        },  
        "CUSTOMER": {  
            "TotalTimeMillis": 108009  
        }  
    },  
    "TotalTimeMillis": 266129  
},  
"ContentMetadata": {  
    "Output": "Redacted",  
    "RedactionTypes": ["PII"]  
}  
}
```

Identifying the dominant language in your media files

Amazon Transcribe is able to automatically identify the predominant language in a media file without you having to specify a language code. Automatic language identification can be used for any batch transcription job, and all supported languages can be identified. Streaming transcriptions are not supported. For a complete list of languages available with Amazon Transcribe, see [Supported languages and language-specific features \(p. 3\)](#).

To identify the language with greater accuracy, you can specify a list of languages you think are present in your collection of media files. From that list, Amazon Transcribe chooses the language with the greatest confidence score to transcribe your audio. A score with a larger value indicates that Amazon Transcribe is more confident it identified the language correctly. For best results, if you are certain of the language spoken in each of the audio files, specify a language code.

Note

Media files are transcribed in a single language, even if they contain speech in two or more languages. Amazon Transcribe transcribes the audio according to the predominant language identified in the file.

Because some Amazon Transcribe features require you to specify a language code, automatic language identification is not supported with all features. You cannot use automatic language identification if the following features are enabled:

- Custom language models
- Custom vocabularies
- Vocabulary filtering
- Content redaction

To increase the chance of identifying the language successfully, media files should have at least 30 seconds of speech.

You can use automatic language identification in a batch transcription job using the [Amazon Transcribe Console](#), [AWS CLI](#), or [AWS SDK](#); see below for instructions:

Console

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, choose **Transcription jobs**, then select the **Create job** button (top right). This opens the **Specify job details** page.
3. In the **Job settings** panel, find the **Language settings** section and select **Automatic language identification**. You also have the option to select multiple languages (from the *Select languages* drop-down box) if you have an idea of which language your file contains. This option can improve accuracy, but is not required.

Specify job details Info

Job settings

Name

my-transcription-job

The name can be up to 200 characters long. Valid characters are a-z, A-Z, 0-9, . (period), _ (underscore), and – (hyphen).

Model type Info

Choose the type of model to use for the transcription job.

General model

To use a model that is not specialized for a particular use case, choose this option. Configuration options vary between languages.

Custom language model

To use a model that you trained for your specific use case, choose this option. This model has fewer configuration options than the general model.

Language settings

You can transcribe your audio file in a language that you specify or have Amazon Transcribe identify and transcribe it in the predominant language.

Specific language Info

If you know the language spoken in your source audio, choose this option to get the most accurate results. The options available for additional processing vary between languages.

Automatic language identification Info

If you don't know the language spoken in your audio files, choose this option. You have access to fewer options for additional processing than if you choose **Specific language**.

Language options for automatic language identification - *optional*

To improve accuracy, choose at least two languages spoken the most often in your audio library. Amazon Transcribe chooses from one of the languages you've specified to transcribe each audio file. Leave this field empty if you're unsure about which languages to select.

Select languages ▾

▼ Additional settings

Job queue - *optional* Info

Enables you to submit jobs beyond the limit for concurrent jobs (100). You must specify access permissions to the resources that job queuing uses.

Add to job queue

- Fill in any other fields you wish to include on the **Specify job details** page, then click the **Next** button. This takes you to the **Configure job - *optional*** page.

Configure job - *optional* Info

Audio settings

Audio identification Info

Choose to split multi-channel audio into separate channels for transcription, or identify speakers in the input audio.

Alternative results Info

Enable to view more transcription results

Cancel

Previous

Create job

- Click the **Create job** button to run your transcription job.

AWS SDK for Python (Boto3)

The following example uses the AWS SDK for Python (Boto3) to add a tag by using the `IdentifyLanguage` argument for the `start_transcription_job` method:

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "your-job-name"
job_uri = "s3://your-S3-bucket/S3-prefix/your-filename.file-extension"
transcribe.start_transcription_job(
    TranscriptionJobName=job_name,
    Media={'MediaFileUri': job_uri},
    MediaFormat='wav',
    IdentifyLanguage='True'
)
while True:
    status = transcribe.get_transcription_job(TranscriptionJobName=job_name)
    if status['TranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED', 'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)
```

AWS CLI

This example uses the `start-transcription-job` command and `IdentifyLanguage` parameter.

```
aws transcribe start-transcription-job \
--transcription-job-name your-job-name \
--media MediaFileUri=s3://your-S3-bucket/S3-prefix/your-filename.file-extension \
--identify-language
```

Here's another example using the `start-transcription-job` command, and a request body that identifies the language.

```
aws transcribe start-transcription-job \
--cli-input-json file://filepath/example-start-command.json
```

The file `example-start-command.json` contains the following request body.

```
{
    "TranscriptionJobName": "your-job-name",
    "Media": {
        "MediaFileUri": "s3://your-S3-bucket/S3-prefix/your-filename.file-extension"
    },
    "IdentifyLanguage": "true"
}
```

Getting notifications using Automatic language identification

Amazon Transcribe can notify you if it successfully identifies the language of your media file before the transcription job finishes. To receive a notification, enable a rule for an Amazon CloudWatch event.

CloudWatch is a service you can use to monitor your AWS resources and applications in real time. For more information see [What Is Amazon CloudWatch](#).

For more information about the CloudWatch events that show whether the language has been identified successfully in your audio, see [Language identification events \(p. 185\)](#). For general information about CloudWatch events in Amazon Transcribe, see [Using Amazon EventBridge with Amazon Transcribe \(p. 183\)](#).

Using vocabulary filtering to filter unwanted words

You can mask, remove, or tag words you don't want in your transcription results with *vocabulary filtering*. For example, you can use vocabulary filtering to prevent the display of offensive or profane terms.

You can use the following methods to filter unwanted words from your transcription results:

- Mask: replace unwanted words with three asterisks (***)
- Remove: remove the unwanted words
- Tag: mark words that are listed in your vocabulary filter in the transcription results

You can use tagging to manually remove the words from some transcripts and leave them in others. This enables you to generate transcripts for multiple audiences from a single stream.

To filter unwanted words, you:

1. Create a list of unwanted words.
2. Create a vocabulary filter.
3. Start your real-time stream or transcription job and specify your vocabulary filter and method. The method (mask, remove, or tag) indicates how you want to filter words from your transcript.

You can filter unwanted words with the [Amazon Transcribe console](#) or the API.

Topics

- [Step 1: Creating a list of unwanted words \(p. 136\)](#)
- [Step 2: Creating a vocabulary filter \(p. 137\)](#)
- [Step 3: Filtering transcriptions \(p. 138\)](#)

Step 1: Creating a list of unwanted words

To create a vocabulary filter, you can create a list of words to filter from your transcription results and save them in a text file. Or, you can use the [CreateVocabularyFilter \(p. 295\)](#) API and enter the words that you want to filter as an array of strings in the words parameter. Although listing unwanted words in the [CreateVocabularyFilter \(p. 295\)](#) API is more convenient, if you use a text file, you can edit your word list later and reuse it in another vocabulary filter.

The following guidelines apply to vocabulary filters:

- Words in a vocabulary filter aren't case sensitive. For example, "curse" and "CURSE" are considered the same word.
- Amazon Transcribe filters only words that exactly match words in the filter. For example, if your filter includes "swear," Amazon Transcribe filters "swear," but not "swears." You must provide every variation of a word that you want to filter.
- Amazon Transcribe doesn't filter words that are contained in other words. For example, if a vocabulary filter contained "marine," but not "submarine," "submarine" would appear in your transcription results.

To create a word list with the console, complete the following procedure. To use the [CreateVocabularyFilter \(p. 295\)](#) API, see [Step 2: Creating a vocabulary filter \(p. 137\)](#).

To create a list of unfiltered words (console)

1. In a text editor, create a new file and enter each word on a separate line followed by the newline (\n) as shown in the following example.

```
profanity
curse
swear
...
obscenity
```

2. Save the list as a plain text file locally or in Amazon Simple Storage Service (Amazon S3).

Next step

[Step 2: Creating a vocabulary filter \(p. 137\)](#)

Step 2: Creating a vocabulary filter

You can create a vocabulary filter with either the [CreateVocabularyFilter \(p. 295\)](#) API or the [Amazon Transcribe console](#).

If you use the [CreateVocabularyFilter \(p. 295\)](#) API, you can enter the words in your vocabulary filter as an array of strings into the `Words` parameter. Although this method is more convenient, if you create a text file, you can edit your word list later and reuse it in another vocabulary filter.

Console

To use the console to create a vocabulary filter, you must have a plain text file that contains the words that you want to filter, formatted as described in [Step 1: Creating a list of unwanted words \(p. 136\)](#). Your file can be saved locally or in Amazon Simple Storage Service (Amazon S3).

To create a vocabulary filter (console)

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, choose **Vocabulary filtering**.
3. Choose **Create vocabulary filter**.
4. For **Name**, enter a vocabulary filter name that is unique within your AWS account.
5. For **Language**, choose the language code for the language of your vocabulary filter.
6. For **Vocabulary input source**, choose one of the following:
 - If you saved the file that contains the word that you want to filter locally, choose **File upload**, then choose **Choose file** and choose the file.
 - If you saved the file in Amazon S3, for **S3 location**, enter the URI of the text file or choose **Browse S3** and browse to the file and choose it.
7. Choose **Create vocabulary filter**.

API

To create a vocabulary filter (API)

- For the [CreateVocabularyFilter \(p. 295\)](#) API, specify the following:

- a. A name for your vocabulary filter that is unique in your AWS account for the `VocabularyFilterName` parameter
- b. The language code for the language of your source audio in the `LanguageCode` parameter
- c. The words for your vocabulary filter using one of the following options:
 - Specify the Amazon Simple Storage Service (Amazon S3) location of the text file for the `VocabularyFilterFileUri` parameter using this format: `s3://DOC-EXAMPLE-BUCKET1/vocabulary-filter-example.txt`.
 - Enter the words as an array of strings in the `Words` parameter, for example `["word", "banana", "potato", "chair"]`.

To see all of the vocabulary filters that you've created, use the [ListVocabularyFilters \(p. 362\)](#) API. You can then use that information with the [GetVocabularyFilter \(p. 335\)](#) API to retrieve the download URI for your vocabulary filter and learn more about that filter.

AWS CLI

The following is an example AWS Command Line Interface (AWS CLI) request to create a vocabulary filter with a text file stored in an Amazon S3 bucket. The commands are followed by the response elements in JSON format.

```
aws transcribe create-vocabulary-filter \
    --vocabulary-filter-name your-filter-name \
    --language-code en-US \
    --vocabulary-filter-file-uri s3://DOC-EXAMPLE-BUCKET1/vocabulary-filter-example.txt

{
    "VocabularyFilterName": "your-filter-name",
    "LanguageCode": "en-US"
}
```

Next step

[Step 3: Filtering transcriptions \(p. 138\)](#)

Step 3: Filtering transcriptions

You can filter unwanted words from both batch and streaming transcriptions. When you create a real-time stream or batch transcription job, specify the vocabulary filter that you want to use and the vocabulary filter method. The method specifies how the words are filtered from your transcription results. There are two vocabulary filter methods available for batch transcription and three methods available for real-time streaming.

You can use the following filter methods for both batch and real-time streaming transcriptions:

- To replace the words caught by your vocabulary filter with three asterisks, ***, use the `mask` method. Use this method to hide unwanted words from your audience, but indicate that they were spoken.
- To remove the words from your transcripts, use the `remove` method. With this method, your audience won't know that unwanted words were spoken.

In real-time streaming transcriptions *only*, you can use the `tag` method to keep unwanted words in your transcription results with a tag that indicates that they were listed in your vocabulary filter. You can then

manually remove the words from some transcripts and leave them in others to generate transcripts for multiple audiences from a single stream.

For information about streaming transcriptions, see [Transcribing streaming audio \(p. 70\)](#). For information about batch transcriptions, see [How Amazon Transcribe works \(p. 3\)](#).

Topics

- [Filtering batch transcriptions \(p. 139\)](#)
- [Filtering streaming transcriptions \(p. 143\)](#)

Filtering batch transcriptions

Use a vocabulary filter to filter unwanted words from a batch transcription job with either the [Amazon Transcribe console](#) or the [StartTranscriptionJob \(p. 376\)](#) API.

The following code shows the parameters and data types.

```
{  
    "ContentRedaction": {  
        "RedactionOutput": "string",  
        "RedactionType": "string"  
    },  
    "JobExecutionSettings": {  
        "AllowDeferredExecution": boolean,  
        "DataAccessRoleArn": "string"  
    },  
    "LanguageCode": "string",  
    "Media": {  
        "MediaFileUri": "string"  
    },  
    "MediaFormat": "string",  
    "MediaSampleRateHertz": number,  
    "OutputBucketName": "string",  
    "OutputEncryptionKMSKeyId": "string",  
    "Settings": {  
        "ChannelIdentification": boolean,  
        "MaxAlternatives": number,  
        "MaxSpeakerLabels": number,  
        "ShowAlternatives": boolean,  
        "ShowSpeakerLabels": boolean,  
        "VocabularyFilterMethod": "string",  
        "VocabularyFilterName": "string",  
        "VocabularyName": "string"  
    },  
    "TranscriptionJobName": "string"  
}
```

Console

To use the console to start a batch transcription job with vocabulary filtering, you must have created a vocabulary filter, as described in [Step 2: Creating a vocabulary filter \(p. 137\)](#).

To filter unwanted words in a transcription job (console)

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, choose **Transcription jobs**.
3. Choose **Create job**.

4. For **Name**, specify a name that is unique within your AWS account for your batch transcription job.
5. For **Language**, choose the language that will be spoken in your transcription job.
6. Specify the location of your audio file or video file in Amazon S3:
 - For **Input file location on S3** under **Input data**, specify the Amazon S3 URI that identifies the media file that you will transcribe.
 - Choose **Browse S3** under **Input data** to browse for the media file and choose it.
7. Choose **Next**.
8. Enable **Vocabulary filtering** under **Content removal**.
9. Choose your vocabulary filter and vocabulary filtering method under **Filter selection**.
10. Choose **Create**.

API

To filter a batch transcription (API)

- For the [StartTranscriptionJob \(p. 376\)](#) API, specify the following:
 - a. For **TranscriptionJobName**, specify a name unique to your AWS account.
 - b. For **LanguageCode**, specify the language code that corresponds to the language spoken in your media file and the language of your vocabulary filter.
 - c. For the **MediaFileUri** parameter of the **Media** object, specify the Amazon S3 location of the audio file or video file that you want to transcribe.
 - d. For the **VocabularyFilterName** parameter, specify the name of your vocabulary filter.
 - e. For the **VocabularyFilterMethod** parameter, choose one of the following options:
 - To mask filtered words by replacing them with three asterisks *******, specify **mask**. Filtering the word "lazy" from the sentence: "The quick brown fox jumps over the lazy dog." with the **mask** method shows "The quick brown fox jumps over the ******* dog." in the transcription.
 - To remove the filtered words from the transcript, specify **remove**. Filtering the word "lazy" from the sentence "The quick brown fox jumps over the lazy dog." with the **remove** method shows "The quick brown fox jumps over the dog." in the transcript.
 - To mark words that match your vocabulary filter in the transcript, specify **tag**. Use this to produce transcripts that are tailored to different audiences. This enables you to:
 - Present the transcription results to one audience, without removing any of the marked words that match the vocabulary filter.
 - Remove any word that matched the vocabulary filter, and present those results to a different audience.
 - Remove some words that matched the vocabulary filter, and present those results to another audience. You can generate multiple transcriptions for many audiences from the same transcription output. For more information, see [Tailoring transcripts to different audiences with tagging \(p. 140\)](#).

Tailoring transcripts to different audiences with tagging

You can generate multiple transcriptions tailored to different audiences from a single audio file. For the [StartTranscriptionJob \(p. 376\)](#) API, use the **tag** method to mark the words in the transcription that match the words in your vocabulary filter. You can present the results of the transcription job to the audience that can see the complete transcription, including the words listed in your vocabulary filter. You can then copy your transcription results, remove the words tagged by your vocabulary filter, and show those results to the audience that shouldn't see the unwanted words.

With tagging, you aren't limited to generating transcriptions for two different audiences. You can generate multiple transcriptions for many audiences from the same audio file. You can choose to remove some words caught by the vocabulary filter in one transcript and leave them in other transcripts.

For example, if "lazy" were in the vocabulary filter, the sentence "The quick brown fox jumps over the lazy dog." would be unchanged in the transcription results. Instead of being masked in, or removed from, the transcription, the value for the `VocabularyFilterMatch` parameter would be `true` for "lazy."

To enable tagging in your batch transcription job, see [Filtering batch transcriptions \(p. 139\)](#).

The word "bloody" is tagged in the following truncated transcription output.

```
{  
    "jobName": "transcription-job-name",  
    "accountId": "account-id",  
    "results": [  
        "transcripts": [  
            {  
                "transcript": "...recording bloody well set up this stupid bloody meeting  
anyway...."  
                ...  
                "items": [  
                    ...  
                    {  
                        "start_time": "5.4",  
                        "end_time": "5.95",  
                        "alternatives": [  
                            {  
                                "confidence": "0.9966",  
                                "content": "recording"  
                            }  
                        ],  
                        "type": "pronunciation",  
                        "vocabularyFilterMatch": false  
                    },  
                    {  
                        "start_time": "7.84",  
                        "end_time": "8.54",  
                        "alternatives": [  
                            {  
                                "confidence": "1.0",  
                                "content": "bloody"  
                            }  
                        ],  
                        "type": "pronunciation",  
                        "vocabularyFilterMatch": true  
                    },  
                    {  
                        "start_time": "8.54",  
                        "end_time": "9.03",  
                        "alternatives": [  
                            {  
                                "confidence": "1.0",  
                                "content": "well"  
                            }  
                        ],  
                        "type": "pronunciation",  
                        "vocabularyFilterMatch": false  
                    },  
                    {  
                        "start_time": "9.04",  
                        "end_time": "9.31",  
                        "alternatives": [  
                            ...  
                        ]  
                    }  
                ]  
            }  
        ]  
    ]  
}
```

```
{  
    "confidence":"0.9905",  
    "content":"set"  
},  
],  
"type":"pronunciation",  
"vocabularyFilterMatch":false  
},  
{  
    "start_time":"9.31",  
    "end_time":"9.44",  
    "alternatives": [  
        {  
            "confidence":"0.9905",  
            "content":"up"  
        },  
        {  
            "type":"pronunciation",  
            "vocabularyFilterMatch":false  
        },  
{  
        "start_time":"9.44",  
        "end_time":"9.6",  
        "alternatives": [  
            {  
                "confidence":"0.8796",  
                "content":"this"  
            },  
            {  
                "type":"pronunciation",  
                "vocabularyFilterMatch":false  
            },  
{  
        "start_time":"9.6",  
        "end_time":"10.22",  
        "alternatives": [  
            {  
                "confidence":"1.0",  
                "content":"stupid"  
            },  
            {  
                "type":"pronunciation",  
                "vocabularyFilterMatch":false  
            },  
{  
        "start_time":"10.23",  
        "end_time":"10.66",  
        "alternatives": [  
            {  
                "confidence":"1.0",  
                "content":"bloody"  
            },  
            {  
                "type":"pronunciation",  
                "vocabularyFilterMatch":true  
            },  
{  
        "start_time":"10.66",  
        "end_time":"11.21",  
        "alternatives": [  
            {  
                "confidence":"0.9994",  
                "content":"meeting"  
            },  
            {  
                "type":"pronunciation",  
                "vocabularyFilterMatch":true  
            }  
        ]  
    ]  
}
```

```

        "vocabularyFilterMatch":false
    },
    {
        "start_time":"11.21",
        "end_time":"11.55",
        "alternatives":[
            {
                "confidence":"0.9978",
                "content":"anyway"
            }
        ],
        "type":"pronunciation",
        "vocabularyFilterMatch":false
    }
    ...
],
},
"status":"COMPLETED"
}

```

Filtering streaming transcriptions

Use a vocabulary filter to filter unwanted words in real-time streams with either the [Amazon Transcribe console](#) or the [StartStreamTranscription \(p. 408\)](#) API.

The following syntax shows the parameters and their data types.

```
{
    "LanguageCode" : "enum",
    "MediaSampleRateHertz" : "integer",
    "MediaEncoding" : "enum",
    "VocabularyName" : "string",
    "SessionId" : "string",
    "AudioStream" : "eventstream",
    "VocabularyFilterName" : "string",
    "VocabularyFilterMethod": "enum"
}
```

To filter a streaming transcription (API)

- For the [StartStreamTranscription \(p. 408\)](#) API, specify the following:
 - a. The language code of your audio in the `LanguageCode` field.
 - b. The sample rate of your audio in the `MediaSampleHertz` field.
 - c. The name of your vocabulary filter in the `VocabularyFilterName` field.
 - d. The filtering method in the `VocabularyFilterMethod` parameter:
 - To mask the filtered words by replacing them with three asterisks (***) specify `mask`. Filtering the word "lazy" from the sentence "The quick brown fox jumps over the lazy dog." with the `mask` method shows "The quick brown fox jumps over the *** dog." in the transcription.
 - To remove the words from the transcript, specify `remove`. Filtering the word "lazy" from the sentence "The quick brown fox jumps over the lazy dog." with the `remove` method shows "The quick brown fox jumps over the dog." in the transcription.
 - To tag words that match the vocabulary filter, specify `tag`. This enables you

to mark the words matching the vocabulary filter without masking or removing them.

To use the same stream to create one transcript with the content filtered and one transcript that is unfiltered, use the tagging method. For information, see [Tailoring transcripts to different audiences with tagging \(p. 144\)](#).

To filter a streaming transcription (console)

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, choose **Real-time transcription**.
3. In **Language**, choose the language of your real-time stream.
4. Choose the **Additional settings** tab and choose your vocabulary filter and vocabulary filtering method.
5. Choose **Start streaming** to begin your stream with vocabulary filtering enabled.

Tailoring transcripts to different audiences with tagging

You can use a single stream to generate a transcription that doesn't show unwanted words and one that does. For the [StartStreamTranscription \(p. 408\)](#) API, use the `tag` method to mark the words in the transcription that match the words in your vocabulary filter. You can present the results of the real-time stream to the audience that can see the complete transcription, including the words listed in your vocabulary filter. You can then copy your transcription results, remove the words tagged by your vocabulary filter, and show those results to the audience that shouldn't see the unwanted words.

With tagging, you aren't limited to generating transcriptions for two different audiences. You can generate multiple transcriptions for many audiences from the same stream. You can choose to remove some words caught by the vocabulary filter in one transcript and leave them in other transcripts.

To enable tagging in a real-time transcription

- For the [StartStreamTranscription \(p. 408\)](#) API, specify the following:
 - a. For `VocabularyFilterName`, the name of your vocabulary filter.
 - b. For `VocabularyFilterMethod`, specify `tag`.

For example, if "bloody" were in the vocabulary filter, the phrase "recording bloody well set up this stupid bloody meeting anyway..." would be unchanged in the transcription results. Instead of being masked in, or removed from, the transcription, the value for the `VocabularyFilterMatch` parameter would be `true` for "bloody."

The following example JSON output shows this.

```
{  
    "jobName": "your-transcription-job-name",  
    "accountId": "account-id",  
    "results": {  
        "transcripts": [  
            {  
                "transcript": "...recording bloody well set up this stupid bloody meeting  
anyway..."  
            }  
        ],  
        "items": [  
            {  
                "item": "recording bloody well set up this stupid bloody meeting  
anyway...",  
                "startOffset": 0,  
                "endOffset": 100,  
                "vocabularyFilterMatch": true  
            }  
        ]  
    }  
}
```

```
{
    "start_time": "5.4",
    "end_time": "5.95",
    "alternatives": [
        {
            "confidence": "0.9966",
            "content": "recording"
        }
    ],
    "type": "pronunciation",
    "vocabularyFilterMatch": false
},
{
    "start_time": "7.84",
    "end_time": "8.54",
    "alternatives": [
        {
            "confidence": "1.0",
            "content": "bloody"
        }
    ],
    "type": "pronunciation",
    "vocabularyFilterMatch": true
},
{
    "start_time": "8.54",
    "end_time": "9.03",
    "alternatives": [
        {
            "confidence": "1.0",
            "content": "well"
        }
    ],
    "type": "pronunciation",
    "vocabularyFilterMatch": false
},
{
    "start_time": "9.04",
    "end_time": "9.31",
    "alternatives": [
        {
            "confidence": "0.9905",
            "content": "set"
        }
    ],
    "type": "pronunciation",
    "vocabularyFilterMatch": false
},
{
    "start_time": "9.31",
    "end_time": "9.44",
    "alternatives": [
        {
            "confidence": "0.9905",
            "content": "up"
        }
    ],
    "type": "pronunciation",
    "vocabularyFilterMatch": false
},
{
    "start_time": "9.44",
    "end_time": "9.6",
    "alternatives": [
        {
            "confidence": "0.8796",
            "content": "the"
        }
    ],
    "type": "pronunciation",
    "vocabularyFilterMatch": false
}
}
```

```

        "content": "this"
    },
],
"type": "pronunciation",
"vocabularyFilterMatch": false
},
{
"start_time": "9.6",
"end_time": "10.22",
"alternatives": [
{
    "confidence": "1.0",
    "content": "stupid"
}
],
"type": "pronunciation",
"vocabularyFilterMatch": false
},
{
"start_time": "10.23",
"end_time": "10.66",
"alternatives": [
{
    "confidence": "1.0",
    "content": "bloody"
}
],
"type": "pronunciation",
"vocabularyFilterMatch": true
},
{
"start_time": "10.66",
"end_time": "11.21",
"alternatives": [
{
    "confidence": "0.9994",
    "content": "meeting"
}
],
"type": "pronunciation",
"vocabularyFilterMatch": false
},
{
"start_time": "11.21",
"end_time": "11.55",
"alternatives": [
{
    "confidence": "0.9978",
    "content": "anyway"
}
],
"type": "pronunciation",
"vocabularyFilterMatch": false
}
]
},
"status": "COMPLETED"
}

```

Identifying speakers (speaker diarization)

To identify different speakers in Amazon Transcribe, use *speaker diarization*. When you enable speaker diarization, Amazon Transcribe labels each speaker utterance. You enable speaker diarization by using the batch transcription or real-time streaming APIs, or the Amazon Transcribe console.

Note

Speaker diarization is supported for all languages with batch transcription jobs, but is only supported for US English (en-US) with streaming transcriptions.

Identifying speakers in audio files

You can enable speaker diarization in a batch transcription job using either the [StartTranscriptionJob \(p. 376\)](#) API or the [Amazon Transcribe console](#).

Console

To identify speakers in an audio file (console)

To use the console to enable speaker diarization in your transcription job, you enable audio identification and then speaker diarization.

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, under Amazon Transcribe, choose **Transcription jobs**.
3. Choose **Create job**.
4. On the **Specify job details** page, provide information about your transcription job.
5. Choose **Next**.
6. Enable **Audio identification**.
7. For **Audio identification type**, choose **Speaker identification**.
8. For **Maximum number of speakers**, specify the maximum number of speakers you think are speaking in your audio. For best results, match the number of speakers you ask Amazon Transcribe to identify to the number of speakers in the input audio. If you specify a value less than the number of speakers in your input audio, the transcription text of the most similar sounding speakers are attributed to a speaker label.
9. Choose **Create**.

API

To identify speakers in an audio file using a batch transcription job (API)

- For the [StartTranscriptionJob \(p. 376\)](#) API, specify the following.
 - a. For **TranscriptionJobName**, specify a name unique to your AWS account.
 - b. For **LanguageCode**, specify the language code that corresponds to the language spoken in your media file and the language of your vocabulary filter.
 - c. For the **MediaFileUri** parameter of the **Media** object, specify the name of the media file you want to transcribe.
 - d. For the **Settings** object, specify the following.

- i. `ShowSpeakerLabels - true`.
- ii. `MaxSpeakerLabels` - An integer between 2 and 10 that indicates the number of speakers you think are speaking in your audio. For best results, match the number of speakers you ask Amazon Transcribe to identify to the number of speakers in the input audio. If you specify a value less than the number of speakers in your input audio, the transcription text of the most similar sounding speakers are attributed to a speaker label.

The following syntax shows the request parameters to start a batch transcription job and their data types.

```
{
  "ContentRedaction": {
    "RedactionOutput": "string",
    "RedactionType": "string"
  },
  "JobExecutionSettings": {
    "AllowDeferredExecution": boolean,
    "DataAccessRoleArn": "string"
  },
  "LanguageCode": "string",
  "Media": {
    "MediaFileUri": "string"
  },
  "MediaFormat": "string",
  "MediaSampleRateHertz": number,
  "OutputBucketName": "string",
  "OutputEncryptionKMSKeyId": "string",
  "Settings": {
    "ChannelIdentification": boolean,
    "MaxAlternatives": number,
    "MaxSpeakerLabels": number,
    "ShowAlternatives": boolean,
    "ShowSpeakerLabels": boolean,
    "VocabularyFilterMethod": "string",
    "VocabularyFilterName": "string",
    "VocabularyName": "string"
  },
  "TranscriptionJobName": "string"
}
```

The following code shows an example output of a transcription job with speaker identification enabled.

```
{
  "jobName": "job ID",
  "accountId": "account ID",
  "results": {
    "transcripts": [
      {
        "transcript": "Professional answer."
      }
    ],
    "speaker_labels": {
      "speakers": 1,
      "segments": [
        {
          "start_time": "0.000000",
          "speaker_label": "spk_0",
          "end_time": "1.430",
        }
      ]
    }
  }
}
```

```
        "items": [
            {
                "start_time": "0.100",
                "speaker_label": "spk_0",
                "end_time": "0.690"
            },
            {
                "start_time": "0.690",
                "speaker_label": "spk_0",
                "end_time": "1.210"
            }
        ]
    ],
    "items": [
        {
            "start_time": "0.100",
            "end_time": "0.690",
            "alternatives": [
                {
                    "confidence": "0.8162",
                    "content": "Professional"
                }
            ],
            "type": "pronunciation"
        },
        {
            "start_time": "0.690",
            "end_time": "1.210",
            "alternatives": [
                {
                    "confidence": "0.9939",
                    "content": "answer"
                }
            ],
            "type": "pronunciation"
        },
        {
            "alternatives": [
                {
                    "content": "."
                }
            ],
            "type": "punctuation"
        }
    ],
    "status": "COMPLETED"
}
```

AWS CLI

To identify the speakers in an audio file using a batch transcription job (AWS CLI)

- Run the following code.

```
aws transcribe start-transcription-job \
--cli-input-json file://filepath/example-start-command.json
```

The following code shows the contents of `example-start-command.json`.

```
{  
    "TranscriptionJobName": "your-transcription-job-name",  
    "LanguageCode": "en-US",  
    "Media": {  
        "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.mp4"  
    },  
    "Settings": {  
        "MaxSpeakerLabels": 2,  
        "ShowSpeakerLabels": true  
    }  
}
```

The following is the response from running the preceding CLI command.

```
{  
    "TranscriptionJob": {  
        "TranscriptionJobName": "your-transcription-job-name",  
        "TranscriptionJobStatus": "IN_PROGRESS",  
        "LanguageCode": "en-US",  
        "Media": {  
            "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET1/your-audio-file"  
        },  
        "StartTime": "2020-07-29T17:45:09.826000+00:00",  
        "CreationTime": "2020-07-29T17:45:09.791000+00:00",  
        "Settings": {  
            "ShowSpeakerLabels": true,  
            "MaxSpeakerLabels": 2  
        }  
    }  
}
```

Identifying speakers in real-time streams

You can identify different speakers in either HTTP/2 or WebSocket streams. Speaker diarization works best for identifying between two and five speakers. Although Amazon Transcribe can identify more than five speakers in a stream, the accuracy of speaker diarization decreases if you exceed that number. To start an HTTP/2 stream, you specify the `ShowSpeakerLabel` request parameter of the [StartStreamTranscription \(p. 408\)](#) API. To start a Websocket request, you use a pre-signed URL, a URL that contains the information needed to start your stream. To use the console to transcribe speech spoken into your microphone, use the following procedure.

You can identify speakers in real-time streams that are in US English (en-US).

To identify speakers in audio that is spoken into your microphone (console)

You can use the [Amazon Transcribe console](#) to start a real-time stream and transcribe any speech picked up by your microphone.

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, choose **Real-time transcription**.
3. In **Language**, choose the language of your real-time stream.

4. Under **Additional settings**, enable **Speaker identification**.
5. Choose **Start streaming**.
6. Speak into the microphone.

HTTP/2 streaming

The following is the syntax for the parameters of an HTTP/2 request.

```
POST /stream-transcription HTTP/2
x-amzn-transcribe-language-code: LanguageCode
x-amzn-transcribe-sample-rate: MediaSampleRateHertz
x-amzn-transcribe-media-encoding: MediaEncoding
x-amzn-transcribe-session-id: SessionId
x-amzn-transcribe-vocabulary-filter-name: VocabularyFilterName
x-amzn-transcribe-vocabulary-filter-method: VocabularyFilterMethod
x-amzn-transcribe-show-speaker-label: ShowSpeakerLabel
Content-type: application/json

{
    "AudioStream": {
        "AudioEvent": {
            "AudioChunk": blob
        }
    }
}
```

To identify speakers in an HTTP/2 stream, use the [StartStreamTranscription \(p. 408\)](#) API and specify the following:

- **LanguageCode** – the language code that corresponds to the language spoken in the stream.
- **MediaSampleHertz** – the sample rate of the audio.
- **ShowSpeakerLabel** – true.

WebSocket streaming

To identify speakers in WebSocket streams, use the following format to create a pre-signed URL to start a WebSocket request and specify **show-speaker-label** as true. A pre-signed URL contains the information to set up bi-directional communication between your application and Amazon Transcribe.

```
GET wss://transcribestreaming.region.amazonaws.com:8443/stream-transcription-websocket
?language-code=languageCode
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=Signature Version 4 credential scope
&X-Amz-Date=date
&X-Amz-Expires=time in seconds until expiration
&X-Amz-Security-Token=security-token
&X-Amz-Signature=Signature Version 4 signature
&X-Amz-SignedHeaders=host
&media-encoding=mediaEncoding
&sample-rate=mediaSampleRateHertz
&session-id=sessionId
&show-speaker-label=true
```

For more information on completing WebSocket requests, see [Creating a pre-signed URL \(p. 77\)](#).

Streaming transcription output

The following code shows the truncated example response of a streaming request.

```
{  
    "Transcript": {  
        "Results": [  
            {  
                "Alternatives": [  
                    {  
                        "Items": [  
                            {  
                                "Confidence": 0.97,  
                                "Content": "From",  
                                "EndTime": 18.98,  
                                "Speaker": "0",  
                                "StartTime": 18.74,  
                                "Type": "pronunciation",  
                                "VocabularyFilterMatch": false  
                            },  
                            {  
                                "Confidence": 1,  
                                "Content": "the",  
                                "EndTime": 19.31,  
                                "Speaker": "0",  
                                "StartTime": 19,  
                                "Type": "pronunciation",  
                                "VocabularyFilterMatch": false  
                            },  
                            {  
                                "Confidence": 1,  
                                "Content": "last",  
                                "EndTime": 19.86,  
                                "Speaker": "0",  
                                "StartTime": 19.32,  
                                "Type": "pronunciation",  
                                "VocabularyFilterMatch": false  
                            },  
                            ...  
                            {  
                                "Confidence": 1,  
                                "Content": "chronic",  
                                "EndTime": 22.55,  
                                "Speaker": "0",  
                                "StartTime": 21.97,  
                                "Type": "pronunciation",  
                                "VocabularyFilterMatch": false  
                            },  
                            ...  
                            {  
                                "Confidence": 1,  
                                "Content": "fatigue",  
                                "EndTime": 24.42,  
                                "Speaker": "0",  
                                "StartTime": 23.95,  
                                "Type": "pronunciation",  
                                "VocabularyFilterMatch": false  
                            },  
                            {  
                                "EndTime": 25.22,  
                                "StartTime": 25.22,  
                                "Type": "speaker-change",  
                            }  
                        ]  
                    ]  
                ]  
            ]  
        ]  
    ]  
}
```

```
        "VocabularyFilterMatch": false
    },
    {
        "Confidence": 0.99,
        "Content": "True",
        "EndTime": 25.63,
        "Speaker": "1",
        "StartTime": 25.22,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Content": ".",
        "EndTime": 25.63,
        "StartTime": 25.63,
        "Type": "punctuation",
        "VocabularyFilterMatch": false
    }
],
"Transcript": "From the last note she still has mild sleep deprivation and
chronic fatigue True."
}
],
"EndTime": 25.63,
"IsPartial": false,
"ResultId": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXX",
"StartTime": 18.74
}
]
}
}
```

Amazon Transcribe breaks your incoming audio stream based on natural speech segments, such as a change in speaker or a pause in the audio. The transcription is returned progressively to your application, with each response containing more transcribed speech until the entire segment is transcribed. The above code is a truncated example of a fully-transcribed speech segment. Speaker labels only appear for entirely transcribed segments.

The following list shows the organization of the objects and parameters in a streaming transcription output.

Transcript

Each speech segment has its own **Transcript** object.

Results

Each **Transcript** object has its own **Results** object. This object contains the **isPartial** field. When its value is **false**, the results returned are for an entire speech segment.

Alternatives

Each **Results** object has an **Alternatives** object.

Items

Each **Alternatives** object has its own **Items** object that contains information about each word and punctuation mark in the transcription output. When you enable speaker identification, each word has a **Speaker** label for fully-transcribed speech segments. Amazon Transcribe uses this label to assign a unique integer to each speaker it identifies in the stream. The **Type** parameter having a value of **speaker-change** indicates that one person has stopped speaking and that another person is about to begin.

Transcript

Each `Items` object contains a transcribed speech segment as the value of the `Transcript` field.

Channel identification

If you have an audio file or stream that has multiple channels, you can use *channel identification* to transcribe the speech from each of those channels. Amazon Transcribe identifies the speech from each channel and transcribes that speech in separate transcriptions. It combines those transcriptions into a single transcription output.

You can enable channel identification for both batch processing and real-time streaming. The following list describes how to enable it for each method.

- Batch transcription - [Amazon Transcribe console](#) and [StartTranscriptionJob \(p. 376\)](#) API
- Streaming transcription - [WebSocket streaming](#) and [StartStreamTranscription \(p. 408\)](#) API

Transcribing multi-channel audio files

To transcribe multi-channel audio in a batch transcription job, use the [Amazon Transcribe console](#) or the [StartTranscriptionJob \(p. 376\)](#) API.

Console

To use the console to enable channel identification in your batch transcription job, you enable audio identification and then channel identification. Channel identification is a subset of audio identification in the console.

To transcribe a multi-channel audio file (console)

1. Sign in to the [Amazon Transcribe console](#).
2. In the navigation pane, under Amazon Transcribe, choose **Transcription jobs**.
3. Choose **Create job**.
4. On the **Specify job details** page, provide information about your transcription job.
5. Choose **Next**.
6. Enable **Audio identification**.
7. For **Audio identification type**, choose **Channel identification**.
8. Choose **Create**.

API

To transcribe a multi-channel audio file (API)

- For the [StartTranscriptionJob \(p. 376\)](#) API, specify the following.
 - a. For `TranscriptionJobName`, specify a name unique to your AWS account.
 - b. For `LanguageCode`, specify the language code that corresponds to the language spoken in your media file. For available languages and corresponding language codes, see [What is Amazon Transcribe? \(p. 1\)](#).
 - c. For the `MediaFileUri` parameter of the `Media` object, specify the name of the media file you want to transcribe.
 - d. For the `Settings` object, set `ChannelIdentification` to `true`.

The following is an example request using the AWS SDK for Python (Boto3).

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "your-transcription-job-name"
job_uri = "the-Amazon-S3-object-URL-of-your-media-file"
transcribe.start_transcription_job(
    TranscriptionJobName=job_name,
    Media= {'MediaFileUri': job_uri},
    MediaFormat= 'mp4',
    LanguageCode= 'en-US',
    Settings = {
        'ChannelIdentification': True,
    }
)
while True:
    status = transcribe.get_transcription_job(TranscriptionJobName=job_name)
    if status['TranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED', 'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)
```

AWS CLI

To transcribe a multi-channel audio file using a batch transcription job (AWS CLI)

- Run the following code.

```
aws transcribe start-transcription-job \
--cli-input-json file://filepath/example-start-command.json
```

The following is the code of `example-start-command.json`.

```
{
    "TranscriptionJobName": "your-transcription-job-name",
    "LanguageCode": "en-US",
    "Media": {
        "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.mp4"
    },
    "Settings": {
        "ChannelIdentification": true
    }
}
```

```
{
    "TranscriptionJobName": "your-transcription-job-name",
    "LanguageCode": "en-US",
    "Media": {
        "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file"
    },
    "Settings": {
```

```
    "ChannelIdentification":true
}
```

The following is the response.

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "your-transcription-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "en-US",
        "Media": {
            "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file"
        },
        "StartTime": "2020-09-12T23:20:28.239000+00:00",
        "CreationTime": "2020-09-12T23:20:28.203000+00:00",
        "Settings": {
            "ChannelIdentification": true
        }
    }
}
```

The following code shows the transcription output for an audio file that has a conversation on two channels.

```
{
    "jobName": "job id",
    "accountId": "account id",
    "results": {
        "transcripts": [
            {
                "transcript": "When you try ... It seems to ..."
            }
        ],
        "channel_labels": {
            "channels": [
                {
                    "channel_label": "ch_0",
                    "items": [
                        {
                            "start_time": "12.282",
                            "end_time": "12.592",
                            "alternatives": [
                                {
                                    "confidence": "1.0000",
                                    "content": "When"
                                }
                            ],
                            "type": "pronunciation"
                        },
                        {
                            "start_time": "12.592",
                            "end_time": "12.692",
                            "alternatives": [
                                {
                                    "confidence": "0.8787",
                                    "content": "It"
                                }
                            ],
                            "type": "pronunciation"
                        }
                    ]
                }
            ]
        }
    }
}
```

```

        "content": "you"
    }
],
"type": "pronunciation"
},
{
    "start_time": "12.702",
    "end_time": "13.252",
    "alternatives": [
        {
            "confidence": "0.8318",
            "content": "try"
        }
    ],
    "type": "pronunciation"
},
...
]
},
{
    "channel_label": "ch_1",
    "items": [
        {
            "start_time": "12.379",
            "end_time": "12.589",
            "alternatives": [
                {
                    "confidence": "0.5645",
                    "content": "It"
                }
            ],
            "type": "pronunciation"
        },
        {
            "start_time": "12.599",
            "end_time": "12.659",
            "alternatives": [
                {
                    "confidence": "0.2907",
                    "content": "seems"
                }
            ],
            "type": "pronunciation"
        },
        {
            "start_time": "12.669",
            "end_time": "13.029",
            "alternatives": [
                {
                    "confidence": "0.2497",
                    "content": "to"
                }
            ],
            "type": "pronunciation"
        },
        ...
    ]
}
}
```

Amazon Transcribe transcribes the audio from each channel separately and combines the transcribed text from each channel into a single transcription output.

For each channel in the transcription output, Amazon Transcribe returns a list of *items*. An item is a transcribed word, pause, or punctuation mark. Each item has a start time and an end time. If a person on

one channel speaks over a person on a separate channel, the start times and end times of the items for each channel overlap while the individuals are speaking over each other.

By default, you can transcribe audio files with two channels. You can request a quota increase if you need to transcribe files that have more than two channels. For information about requesting a quota increase, see [AWS service quotas](#).

Transcribing multi-channel audio streams

You can transcribe audio from separate channels in either HTTP/2 or WebSocket streams using the [StartStreamTranscription \(p. 408\)](#) API.

By default, you can transcribe streams with two channels. You can request a quota increase if you need to transcribe streams that have more than two channels. For information about requesting a quota increase, see [AWS service quotas](#).

Transcribing multi-channel audio in an HTTP/2 stream

To transcribe multi-channel audio in an HTTP/2 stream, use the [StartStreamTranscription \(p. 408\)](#) API and specify the following:

- `LanguageCode` - The language code of the audio.
- `MediaEncoding` - The encoding of the audio.
- `EnableChannelIdentification` - `true`
- `NumberOfChannels` - The number of channels in your streaming audio.

The following is the syntax for the parameters of an HTTP/2 request.

```
POST /stream-transcription HTTP/2
x-amzn-transcribe-language-code: LanguageCode
x-amzn-transcribe-sample-rate: MediaSampleRateHertz
x-amzn-transcribe-media-encoding: MediaEncoding
x-amzn-transcribe-session-id: SessionId
x-amzn-transcribe-vocabulary-filter-name: VocabularyFilterName
x-amzn-transcribe-vocabulary-filter-method: VocabularyFilterMethod
x-amzn-transcribe-enable-channel-identification: EnableChannelIdentification
x-amzn-transcribe-number-of-channels: NumberOfChannels
Content-type: application/json

{
    "AudioStream": {
        "AudioEvent": {
            "AudioChunk": blob
        }
    }
}
```

Transcribing multi-channel audio in a WebSocket stream

To identify speakers in WebSocket streams, use the following format to create a pre-signed URL and start a WebSocket request. Specify `enable-channel-id` as `true` and the number of channels in your stream in `number-of-channels`. A pre-signed URL contains the information needed to set up bi-directional communication between your application and Amazon Transcribe.

```
GET wss://transcribestreaming.region.amazonaws.com:8443/stream-transcription-websocket
?language-code=languageCode
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=Signature Version 4 credential scope
&X-Amz-Date=date
&X-Amz-Expires=time in seconds until expiration
&X-Amz-Security-Token=security-token
&X-Amz-Signature=Signature Version 4 signature
&X-Amz-SignedHeaders=host
&media-encoding=mediaEncoding
&sample-rate=mediaSampleRateHertz
&session-id=sessionId
&enable-channel-identification=true
&number-of-channels=number of channels in your audio stream
```

For more information about WebSocket requests, see [Creating a pre-signed URL \(p. 77\)](#).

Multi-channel streaming output

The output of a streaming transcription is the same for HTTP/2 and WebSocket requests. The following is an example output.

```
{
    "resultId": "XXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXX",
    "startTime": 0.11,
    "endTime": 0.66,
    "isPartial": false,
    "alternatives": [
        {
            "transcript": "Left.",
            "items": [
                {
                    "startTime": 0.11,
                    "endTime": 0.45,
                    "type": "pronunciation",
                    "content": "Left",
                    "vocabularyFilterMatch": false
                },
                {
                    "startTime": 0.45,
                    "endTime": 0.45,
                    "type": "punctuation",
                    "content": ".",
                    "vocabularyFilterMatch": false
                }
            ]
        }
    ],
    "channelId": "ch_0"
```

}

For each speech segment, there is a `channelId` flag that indicates which channel the speech belongs to.

Security in Amazon Transcribe

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The [shared responsibility model](#) describes this as security of the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the [AWS compliance programs](#). To learn about the compliance programs that apply to Amazon Transcribe, see [AWS Services in Scope by Compliance Program](#).
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This section helps you understand how to apply the shared responsibility model when using Amazon Transcribe. The following topics show you how to configure Amazon Transcribe to meet your security and compliance objectives. You also learn how to use other AWS services to monitor and secure your Amazon Transcribe resources.

Topics

- [Data protection in Amazon Transcribe \(p. 162\)](#)
- [Identity and access management for Amazon Transcribe \(p. 168\)](#)
- [Logging and monitoring in Amazon Transcribe \(p. 181\)](#)
- [Compliance validation for Amazon Transcribe \(p. 186\)](#)
- [Resilience in Amazon Transcribe \(p. 187\)](#)
- [Infrastructure security in Amazon Transcribe \(p. 187\)](#)

Data protection in Amazon Transcribe

Amazon Transcribe conforms to the AWS [shared responsibility model](#), which includes regulations and guidelines for data protection. AWS is responsible for protecting the global infrastructure that runs all of the AWS services. AWS maintains control over data hosted on this infrastructure, including the security configuration controls for handling customer content and personal data. AWS customers and APN partners, acting either as data controllers or data processors, are responsible for any personal data that they put in the AWS Cloud.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM), so that each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon Simple Storage Service (Amazon S3).

- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see [Federal Information Processing Standard \(FIPS\) 140-2](#).

We strongly recommend that you never put sensitive identifying information, such as your customers' account numbers, into free-form fields such as a **Name** field. This includes when you work with Amazon Transcribe or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into Amazon Transcribe or other services might get picked up for inclusion in diagnostic logs. When you provide a URL to an external server, don't include credentials information in the URL to validate your request to that server.

For more information about data protection, see the [AWS Shared Responsibility Model and GDPR](#) blog post on the [AWS Security Blog](#).

Topics

- [Encryption at rest \(p. 163\)](#)
- [Encryption in transit \(p. 163\)](#)
- [Key management \(p. 163\)](#)
- [Opting out of using your data for service improvement \(p. 165\)](#)
- [Amazon Transcribe and interface VPC endpoints \(AWS PrivateLink\) \(p. 166\)](#)

Encryption at rest

Amazon Transcribe uses the default Amazon S3 key (SSE-S3) for server-side encryption of transcripts placed in your S3 bucket.

When you use the [StartTranscriptionJob \(p. 376\)](#) API, you can specify your own AWS Key Management Service key to encrypt the output from a transcription job.

Amazon Transcribe uses an Amazon EBS volume encrypted with the default key.

Encryption in transit

Amazon Transcribe uses TLS 1.2 with AWS certificates to encrypt data in transit. This includes streaming transcription.

Key management

Amazon Transcribe works with AWS Key Management Service (KMS) to provide enhanced encryption for your data. Amazon S3 already enables you to encrypt your input audio when creating a transcription job. Integration with KMS enables you to encrypt the output of the [StartTranscriptionJob \(p. 376\)](#) API.

If you don't specify a customer master key (CMK), the output of the transcription job is encrypted with the default Amazon S3 key (SSE-S3).

For more information on AWS KMS, see the [AWS Key Management Service Developer Guide](#).

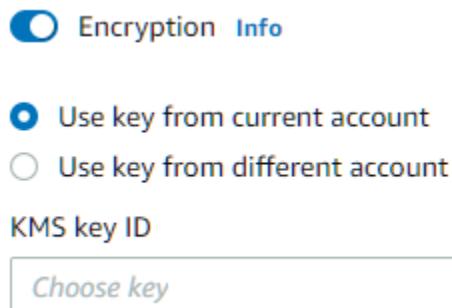
KMS encryption with the AWS Console

To encrypt the output of your transcription job, you can choose between using a customer managed CMK for the account that is making the request, or you can use a CMK from another account.

If you don't specify a CMK, the output of the transcription job is encrypted with the default Amazon S3 key (SSE-S3).

To enable output result encryption

- Under **Output data** choose **Encryption**.



- Choose whether the KMS customer managed CMK is from the account you're currently using or from a different account. If you want to use a key from the current account, choose the key from **KMS key ID**. If you're using a key from a different account, you need to enter the key ARN. To use a key from a different account, the caller must have `kms:Encrypt` permissions for the KMS key.

KMS encryption with the API

To use output encryption with the API, you set the `OutputEncryptionKMSKeyId` parameter of the [StartTranscriptionJob](#) (p. 376) API. You can use a customer managed CMK from the current account, or you can use a key from another account. The account that you are using to create the job must have `kms:Encrypt` permissions for the KMS key.

You can use either of the following to identify a KMS key in the current account:

- KMS Key ID: "1234abcd-12ab-34cd-56ef-1234567890ab"
- KMS Key Alias: "alias/ExampleAlias"

You can use either of the following to identify a AWS KMS key in the current account or another account:

- Amazon Resource Name (ARN) of a KMS Key: "arn:aws:kms:region:account ID:key/1234abcd-12ab-34cd-56ef-1234567890ab"
- ARN of a KMS Key Alias: "arn:aws:kms:region:account ID:alias/ExampleAlias"

KMS encryption context

AWS KMS encryption context is a map of plain text, non-secret key:value pairs. This map represents additional authenticated data, known as encryption context pairs, which provide an added layer of security for your data. Amazon Transcribe requires a symmetric key to encrypt transcription output into a customer-specified S3 bucket. To learn more, see [Using symmetric and asymmetric keys](#).

When creating your encryption context pairs, **do not** use sensitive information. Encryption context is not secret—it is visible in plain text within your CloudTrail logs (so you can use it to identify and categorize your cryptographic operations).

Your encryption context pair can include special characters, such as underscores (_), dashes (-), slashes (/), \ and colons (:).

Tip

It can be useful to relate the values in your encryption context pair to the data being encrypted.

Although not required, we recommend you use non-sensitive metadata related to your encrypted content, such as file names, header values, or unencrypted database fields.

To use output encryption with the API, set the `KMSEncryptionContext` parameter in the [StartTranscriptionJob \(p. 376\)](#) operation. In order to provide encryption context for the output encryption operation, the `OutputEncryptionKMSKeyId` parameter must reference a symmetric KMS key ID.

You can use [AWS KMS condition keys](#) with IAM policies to control access to a symmetric KMS key based on the encryption context that was used in the request for a [cryptographic operation](#). For example, the below policy grants the IAM role "ExampleRole" permission to use the KMS *Decrypt* and *Encrypt* operations for this particular KMS key. This policy works **only** for requests with at least one encryption context pair, in this case "color:indig0Blu3".

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "AWS": "arn:aws:iam::Account-number:role/ExampleRole"  
            },  
            "Action": [  
                "kms:Decrypt",  
                "kms:DescribeKey",  
                "kms:Encrypt",  
                "kms:GenerateDataKey*",  
                "kms:ReEncrypt"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {  
                    "kms:EncryptionContext:color": "indig0Blu3"  
                }  
            }  
        }  
    ]  
}
```

Using encryption context is optional, but recommended. For more information, see [Encryption context](#).

Opting out of using your data for service improvement

By default, Amazon Transcribe stores and uses voice inputs that it has processed to develop the service and continuously improve your experience. You can opt out of having your content used to develop and improve Amazon Transcribe by using an AWS Organizations opt-out policy. For information about how to opt out, see [AI services opt-out policies](#).

Opting out has the following effect:

- Amazon Transcribe deletes all of the transcripts stored in service-managed buckets that were generated before you opted out.
- When you use the [StartTranscriptionJob \(p. 376\)](#) API, you must specify where you want to store output with the `OutputBucketName` parameter. Otherwise, you get a `BadRequestException` error.
- If the transcripts were stored in a service-managed bucket, the [GetTranscriptionJob \(p. 329\)](#) API returns `null` as the value of the `TranscriptFileUri` or `RedactedTranscriptFileUri` parameters.

If you store transcripts in service-managed buckets, we highly recommend backing them up. To back up your transcripts, store them in an S3 bucket that you manage before you opt out. To see which of your

transcription jobs uses Amazon Transcribe to store its outputs, see the `OutputLocationType` response parameter of the [ListTranscriptionJobs \(p. 356\)](#) API.

To move transcripts to your own S3 buckets

1. In the `TranscriptionJobName` parameter of the [GetTranscriptionJob \(p. 329\)](#) API, specify the name of the transcription job whose output you want to back up.
2. Use the link provided in the `TranscriptFileUri` or `RedactedTranscriptFileUri` response parameters to download your transcript.
3. Sign in to the [Amazon S3 console](#).
4. In the **Bucket name** list, choose the name of the bucket that you want to upload your files to.
5. Choose **Upload**.
6. In the **Upload** dialog box, choose **Add files**.
7. Choose one or more files to upload, and then choose **Open**.
8. Choose **Upload**.

Amazon Transcribe and interface VPC endpoints (AWS PrivateLink)

You can establish a private connection between your VPC and Amazon Transcribe by creating an *interface VPC endpoint*. Interface endpoints are powered by [AWS PrivateLink](#), a technology that enables you to privately access Amazon Transcribe APIs without an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to communicate with Amazon Transcribe APIs. Traffic between your VPC and Amazon Transcribe does not leave the Amazon network.

Each interface endpoint is represented by one or more [Elastic Network Interfaces](#) in your subnets.

For more information, see [Interface VPC endpoints \(AWS PrivateLink\)](#) in the *Amazon VPC User Guide*.

Considerations for Amazon Transcribe VPC endpoints

Before you set up an interface VPC endpoint for Amazon Transcribe, ensure that you review [Interface endpoint properties and limitations](#) in the *Amazon VPC User Guide*.

Amazon Transcribe supports making calls to all of its API actions from your VPC.

Creating an interface VPC endpoint for Amazon Transcribe

You can create a VPC endpoint for the Amazon Transcribe service using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see [Creating an interface endpoint](#) in the *Amazon VPC User Guide*.

For batch transcription in Amazon Transcribe, create a VPC endpoint using the following service name:

- com.amazonaws.*region*.transcribe

For streaming transcription in Amazon Transcribe, create a VPC endpoint using the following service name:

- com.amazonaws.*region*.transcribestreaming

If you enable private DNS for the endpoint, you can make API requests to Amazon Transcribe using its default DNS name for the Region, for example, `transcribestreaming.us-east-2.amazonaws.com`.

For more information, see [Accessing a service through an interface endpoint](#) in the *Amazon VPC User Guide*.

Creating a VPC endpoint policy for Amazon Transcribe

You can attach an endpoint policy to your VPC endpoint that controls access to either the streaming service or the batch transcription service of Amazon Transcribe. The policy specifies the following information:

- The principal that can perform actions.
- The actions that can be performed.
- The resources on which actions can be performed.

For more information, see [Controlling access to services with VPC endpoints](#) in the *Amazon VPC User Guide*.

Example: VPC endpoint policy for Amazon Transcribe streaming transcription actions

The following is an example of an endpoint policy for streaming transcription in Amazon Transcribe. When attached to an endpoint, this policy grants access to the listed Amazon Transcribe actions for all principals on all resources.

```
{  
    "Statement": [  
        {  
            "Principal": "*",  
            "Effect": "Allow",  
            "Action": [  
                "transcribe:StartStreamTranscription",  
                "transcribe:StartStreamTranscriptionWebSocket"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

Example: VPC endpoint policy for Amazon Transcribe batch transcription actions

The following is an example of an endpoint policy for batch transcription in Amazon Transcribe. When attached to an endpoint, this policy grants access to the listed Amazon Transcribe actions for all principals on all resources.

```
{  
    "Statement": [  
        {  
            "Principal": "*",  
            "Effect": "Allow",  
            "Action": [  
                "transcribe:StartTranscriptionJob",  
                "transcribe>ListTranscriptionJobs"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

Identity and access management for Amazon Transcribe

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use Amazon Transcribe resources. IAM is an AWS service that you can use with no additional charge.

Topics

- [Audience \(p. 168\)](#)
- [Authenticating with identities \(p. 168\)](#)
- [Managing access using policies \(p. 170\)](#)
- [How Amazon Transcribe works with IAM \(p. 172\)](#)
- [Amazon Transcribe identity-based policy examples \(p. 175\)](#)
- [Troubleshooting Amazon Transcribe identity and access \(p. 179\)](#)

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in Amazon Transcribe.

Service user – If you use the Amazon Transcribe service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Amazon Transcribe features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in Amazon Transcribe, see [Troubleshooting Amazon Transcribe identity and access \(p. 179\)](#).

Service administrator – If you're in charge of Amazon Transcribe resources at your company, you probably have full access to Amazon Transcribe. It's your job to determine which Amazon Transcribe features and resources your employees should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with Amazon Transcribe, see [How Amazon Transcribe works with IAM \(p. 172\)](#).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Amazon Transcribe. To view example Amazon Transcribe identity-based policies that you can use in IAM, see [Amazon Transcribe identity-based policy examples \(p. 175\)](#).

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. For more information about signing in using the AWS Management Console, see [Signing in to the AWS Management Console as an IAM user or root user](#) in the *IAM User Guide*.

You must be *authenticated* (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role. You can also use your company's single sign-on authentication or even sign in using Google or Facebook. In these cases, your administrator previously set up identity federation using IAM roles. When you access AWS using credentials from another company, you are assuming a role indirectly.

To sign in directly to the [AWS Management Console](#), use your password with your root user email address or your IAM user name. You can access AWS programmatically using your root user or IAM

users access keys. AWS provides SDK and command line tools to cryptographically sign your request using your credentials. If you don't use AWS tools, you must sign the request yourself. Do this using *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see [Signature Version 4 signing process](#) in the *AWS General Reference*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *IAM User Guide*.

AWS account root user

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the *AWS account root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the [best practice of using the root user only to create your first IAM user](#). Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. An IAM user can have long-term credentials such as a user name and password or a set of access keys. To learn how to generate access keys, see [Managing access keys for IAM users](#) in the *IAM User Guide*. When you generate access keys for an IAM user, make sure you view and securely save the key pair. You cannot recover the secret access key in the future. Instead, you must generate a new access key pair.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see [When to create an IAM user \(instead of a role\)](#) in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by [switching roles](#). You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see [Using IAM roles](#) in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an [identity provider](#). For more information about federated users, see [Federated users and roles](#) in the *IAM User Guide*.

- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see [How IAM roles differ from resource-based policies](#) in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
- **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions. To see whether an action requires additional dependent actions in a policy, see [Actions, Resources, and Condition Keys for Amazon Transcribe](#) in the *Service Authorization Reference*.
- **Service role** – A service role is an [IAM role](#) that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see [Creating a role to delegate permissions to an AWS service](#) in the *IAM User Guide*.
- **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see [Using an IAM role to grant permissions to applications running on Amazon EC2 instances](#) in the *IAM User Guide*.

To learn whether to use IAM roles or IAM users, see [When to create an IAM role \(instead of a user\)](#) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. You can sign in as the root user or an IAM user, or you can assume an IAM role. When you then make a request, AWS evaluates the related identity-based or resource-based policies. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see [Overview of JSON policies](#) in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

Every IAM entity (user or role) starts with no permissions. In other words, by default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see [Creating IAM policies](#) in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see [Choosing between managed policies and inline policies](#) in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must [specify a principal](#) in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see [Access control list \(ACL\) overview](#) in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see [Permissions boundaries for IAM entities](#) in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see [How SCPs work](#) in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies.

Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see [Session policies](#) in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see [Policy evaluation logic](#) in the *IAM User Guide*.

For more information about identity and access management for Amazon Transcribe, continue to the following pages:

- [How Amazon Transcribe works with IAM \(p. 172\)](#)
- [Troubleshooting Amazon Transcribe identity and access \(p. 179\)](#)

How Amazon Transcribe works with IAM

Before you use IAM to manage access to Amazon Transcribe, you should understand which IAM features are available to use with Amazon Transcribe. To get a high-level view of how Amazon Transcribe and other AWS services work with IAM, see [AWS Services That Work with IAM](#) in the *IAM User Guide*.

Topics

- [Amazon Transcribe identity-based policies \(p. 172\)](#)
- [Amazon Transcribe IAM roles \(p. 175\)](#)

Amazon Transcribe identity-based policies

With IAM identity-based policies, you can specify allowed or denied actions and resources and the conditions under which actions are allowed or denied. Amazon Transcribe supports specific actions, resources, and condition keys. To learn about all of the elements that you use in a JSON policy, see [IAM JSON Policy Elements Reference](#) in the *IAM User Guide*.

Actions

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on **what resources**, and under what **conditions**.

The Action element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

Policy actions in Amazon Transcribe use the following prefix before the action: `transcribe:`. For example, to grant someone permission to run an Amazon EC2 instance with the Amazon Transcribe [StartTranscriptionJob \(p. 376\)](#) API, you include the `transcribe:StartTranscriptionJob` action in their policy. Policy statements must include either an Action or NotAction element. Amazon Transcribe defines actions that describe tasks that you can perform with this service.

To specify multiple actions in a single statement, separate them with commas as follows.

```
"Action": [  
    "transcribe:action1",
```

```
"transcribe:action2"
```

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word `List`, include the following action.

```
"Action": "transcribe>List*"
```

To see a list of Amazon Transcribe actions, see [Actions Defined by Amazon Transcribe](#) in the *IAM User Guide*.

Note

You can also control access to your resources using tags. For information on tag-based access control, see [Controlling access to AWS resources using tags](#).

Resources

Amazon Transcribe doesn't support specifying resource ARNs in a policy.

Condition keys

Use the `Condition` element (or a `Condition block`) to specify conditions in which a statement is in effect. The `Condition` element is optional. You can build conditional expressions that use [condition operators](#), such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical AND operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical OR operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see [IAM policy elements: variables and tags](#) in the *IAM user guide*.

Amazon Transcribe defines its own set of condition keys and also supports using some global condition keys. For a list of all AWS global condition keys, see [AWS global condition context keys](#) in the *IAM User Guide*.

The following table lists the Amazon Transcribe condition keys that apply to Amazon Transcribe resources. You can include these keys in the `Condition` element in an IAM permissions policy.

Amazon Transcribe condition key	Description	Value type	Action
<code>transcribe:OutputBucketName</code>	Filters access by the output bucket used to start a transcription job.	String	StartTranscriptionJob (p. 376)
<code>transcribe:OutputEncryptionKeyARN</code>	Filters access by the KMS key used to start a transcription job.	String	StartTranscriptionJob (p. 376)
<code>transcribe:Outputkey</code>	Filters access by the output key used to start a transcription job.	String	StartTranscriptionJob (p. 376)

For examples of how you can use condition keys to control access to the resources of Amazon Transcribe, see the following.

If you want your users to always use a specific output bucket when they use the [StartTranscriptionJob \(p. 376\)](#) API, you can use the following policy.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "VisualEditor0",  
            "Effect": "Allow",  
            "Action": [  
                "transcribe:StartTranscriptionJob",  
                ],  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {  
                    "transcribe:OutputBucketName": "DOC-EXAMPLE-BUCKET"  
                }  
            }  
        }  
    ]  
}
```

If you want users to always use a AWS Key Management Service (KMS) key when they use the [StartTranscriptionJob \(p. 376\)](#) API, you can use the following policy.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "VisualEditor0",  
            "Effect": "Allow",  
            "Action": [  
                "transcribe:StartTranscriptionJob",  
                ],  
            "Resource": "*",  
            "Condition": {  
                "Null": {  
                    "transcribe:OutputEncryptionKMSKeyId": "false"  
                }  
            }  
        }  
    ]  
}
```

For information on AWS KMS keys, see [Key management \(p. 163\)](#).

If you want your users to always use a specific output key when they use the [StartTranscriptionJob \(p. 376\)](#) API, you can use the following policy.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "VisualEditor0",  
            "Effect": "Allow",  
            "Action": [  
                "transcribe:StartTranscriptionJob",  
                ],  
            "Resource": "*"  
        }  
    ]  
}
```

```
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "transcribe:Outputkey": "DOC-EXAMPLE-BUCKET/prefix"
            }
        }
    ]
}
```

For more information, see the [OutputKey](#) parameter description of the [StartTranscriptionJob](#) (p. 376) API.

Examples

For examples of Amazon Transcribe identity-based policies, see [Amazon Transcribe identity-based policy examples](#) (p. 175).

Amazon Transcribe resource-based policies

Amazon Transcribe doesn't support resource-based policies.

Amazon Transcribe IAM roles

An [IAM role](#) is an entity within your AWS account that has specific permissions.

Using temporary credentials with Amazon Transcribe

You can use temporary credentials to sign in with federation, to assume an IAM role, or to assume a cross-account role. You obtain temporary security credentials by calling AWS Security Token Service (AWS STS) APIs, such as [AssumeRole](#) or [GetFederationToken](#).

Amazon Transcribe supports using temporary credentials.

Service-linked roles

[Service-linked roles](#) allow AWS services to access resources in other services to complete an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view, but can't edit, the permissions for service-linked roles.

Amazon Transcribe doesn't support service-linked roles.

Service roles

You can allow a service to assume a [service role](#) on your behalf. This role allows the service to access resources in other services to complete an action on your behalf. Service roles appear in your IAM account and are owned by the account. This means that an IAM administrator can change the permissions for this role. However, doing so might prevent the service from functioning as expected.

Amazon Transcribe doesn't support service roles.

Amazon Transcribe identity-based policy examples

By default, IAM users and roles don't have permission to create or modify Amazon Transcribe resources. They also can't perform tasks using the AWS Management Console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the resources that they need. The administrator must then attach those policies to the IAM users or groups that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see [Creating policies on the JSON tab](#) in the *IAM User Guide*.

Topics

- [Policy best practices \(p. 176\)](#)
- [Using the Amazon Transcribe console \(p. 176\)](#)
- [AWS managed \(predefined\) policies for Amazon Transcribe \(p. 177\)](#)
- [Permissions required for IAM user roles \(p. 177\)](#)
- [Permissions required for Amazon S3 encryption keys \(p. 178\)](#)
- [Allow users to view their own permissions \(p. 178\)](#)

Policy best practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete Amazon Transcribe resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started using AWS managed policies** – To start using Amazon Transcribe quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see [Get started using permissions with AWS managed policies](#) in the *IAM User Guide*.
- **Grant least privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later. For more information, see [Grant least privilege](#) in the *IAM User Guide*.
- **Enable MFA for sensitive operations** – For extra security, require IAM users to use multi-factor authentication (MFA) to access sensitive resources or API operations. For more information, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *IAM User Guide*.
- **Use policy conditions for extra security** – To the extent that it's practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions to allow requests only within a specified date or time range, or to require the use of SSL or MFA. For more information, see [IAM JSON policy elements: Condition](#) in the *IAM User Guide*.

Using the Amazon Transcribe console

To access the [Amazon Transcribe console](#), you must have a minimum set of permissions for the console. These permissions must allow you to list and view details about the Amazon Transcribe resources in your AWS account. If you create an identity-based policy that applies permissions that are more restrictive than the minimum required permissions, the console won't function as intended for entities (IAM users or roles) with that policy.

To ensure that those entities can use the [Amazon Transcribe console](#), attach the following AWS managed policy to them.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Action": [  
                "transcribe:*"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

```
        "Effect": "Allow"
    ]
}
```

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that you're trying to perform.

For more information, see [Adding Permissions to a User](#) in the *IAM User Guide*:

AWS managed (predefined) policies for Amazon Transcribe

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. These policies are called AWS managed policies. Managed policies make it easier for you to assign appropriate permissions to users, groups, and roles than if you had to write the policies yourself. For more information, see [AWS Managed Policies](#) in the *IAM User Guide*.

The following AWS managed policies, which you can attach to users, roles, and groups in your account, are specific to Amazon Transcribe:

- **AmazonTranscribeReadOnly** – Grants read-only access to Amazon Transcribe resources so that you can get and list transcription jobs and custom vocabularies.
- **AmazonTranscribeFullAccess** – Grants full access to create, read, update, delete, and run all Amazon Transcribe resources. It also allows access to Amazon Simple Storage Service (Amazon S3) buckets with `transcribe` in the bucket name.

Note

You can review the managed permission policies by signing in to the IAM console and searching by policy name. A search for "transcribe" returns both policies listed above (*AmazonTranscribeReadOnly* and *AmazonTranscribeFullAccess*).

You can also create your own custom IAM policies to allow permissions for Amazon Transcribe API actions. You can attach these custom policies to the IAM users, roles, or groups that require those permissions.

Permissions required for IAM user roles

When you create an IAM user to call Amazon Transcribe, the identity must have permission to access the S3 bucket and the AWS Key Management Service (AWS KMS) key used to encrypt the contents of the bucket, if you provided one.

The user must have the following IAM policy for decrypt permissions on the KMS Amazon Resource Name (ARN.)

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "KMS key ARN",
            "Effect": "Allow"
        }
    ]
}
```

The user's IAM policy must have Amazon S3 permissions to access the S3 bucket where audio files are stored and transcriptions are saved.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "S3 bucket location"
        }
    ]
}
```

Permissions required for Amazon S3 encryption keys

If you are using an AWS KMS key to encrypt an Amazon S3 bucket, include the following in the AWS KMS key policy. This gives Amazon Transcribe access to the contents of the bucket.

```
{
    "Sid": "Allow-Transcribe",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::account id:root",
    },
    "Action": [
        "kms:Decrypt"
    ],
    "Resource": "KMS key ARN"
}
```

For more information about allowing access to customer master keys, see [Allowing external AWS Accounts to access a CMK in the AWS KMS developer guide](#).

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam>ListGroupsForUser",
                "iam>ListAttachedUserPolicies",
                "iam>ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:ListGroupsForUser",
                "iam>ListAttachedUserPolicies",
                "iam>ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        }
    ]
}
```

```
        "Action": [
            "iam:GetGroupPolicy",
            "iam:GetPolicyVersion",
            "iam:GetPolicy",
            "iam>ListAttachedGroupPolicies",
            "iam>ListGroupPolicies",
            "iam>ListPolicyVersions",
            "iam>ListPolicies",
            "iam>ListUsers"
        ],
        "Resource": "*"
    }
]
```

Troubleshooting Amazon Transcribe identity and access

Use the following information to diagnose and fix common issues that you might encounter when working with Amazon Transcribe and AWS Identity and Access Management (IAM).

Topics

- [I am not authorized to perform an action in Amazon Transcribe \(p. 179\)](#)
- [I am not authorized to perform iam:PassRole \(p. 179\)](#)
- [I want to view my access keys \(p. 180\)](#)
- [I'm an administrator and want to allow others to access Amazon Transcribe \(p. 180\)](#)
- [I want to allow people outside of my AWS account to access my Amazon Transcribe resources \(p. 180\)](#)

I am not authorized to perform an action in Amazon Transcribe

If you are using the AWS Management Console, and you get a message that you're not authorized to perform an action, contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

For example, the following error occurs when an IAM user named `mateojackson` IAM tries to use the console to view details about a transcription job but doesn't have `transcribe:GetTranscriptionJob` permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
transcribe:GetTranscriptionJob
```

Mateo must ask his administrator to update his policies to allow him to access the [GetTranscriptionJob \(p. 329\)](#) API using the `transcribe:GetTranscriptionJob` action.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password. Ask that person to update your policies to allow you to pass a role to Amazon Transcribe.

Some AWS services allow you to pass an existing role to that service, instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in Amazon Transcribe. However, the action requires the service to have permissions granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform: iam:PassRole
```

In this case, Mary asks her administrator to update her policies to allow her to perform the `iam:PassRole` action.

I want to view my access keys

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair.

Access keys consist of two parts: an access key ID (for example, `AKIAIOSFODNN7EXAMPLE`) and a secret access key (for example, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Like a user name and password, you must use both the access key ID and secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

Important

Do not provide your access keys to a third party, even to help [find your canonical user ID](#). By doing this, you might give someone permanent access to your account.

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must add new access keys to your IAM user. You can have a maximum of two access keys. If you already have two, you must delete one key pair before creating a new one. To view instructions, see [Managing access keys](#) in the *IAM User Guide*.

I'm an administrator and want to allow others to access Amazon Transcribe

To allow others to access Amazon Transcribe, you must create an IAM entity (user or role) for the person or application that needs access. They will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in Amazon Transcribe.

To get started right away, see [Creating your first IAM delegated user and group](#) in the *IAM User Guide*.

I want to allow people outside of my AWS account to access my Amazon Transcribe resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether Amazon Transcribe supports these features, see [How Amazon Transcribe works with IAM](#) (p. 172).
- To learn how to provide access to your resources across AWS accounts that you own, see [Providing access to an IAM user in another AWS account that you own](#) in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see [Providing access to AWS accounts owned by third parties](#) in the *IAM User Guide*.
- To learn how to provide access through identity federation, see [Providing access to externally authenticated users \(identity federation\)](#) in the *IAM User Guide*.

- To learn the difference between using roles and resource-based policies for cross-account access, see [How IAM roles differ from resource-based policies](#) in the *IAM User Guide*.

Logging and monitoring in Amazon Transcribe

Monitoring is an important part of maintaining the reliability, availability, and performance of your Amazon Transcribe applications. To monitor Amazon Transcribe API calls, you can use AWS CloudTrail. To monitor the status of your jobs, use Amazon EventBridge.

Topics

- [Monitoring Amazon Transcribe API calls with AWS CloudTrail \(p. 181\)](#)
- [Using Amazon EventBridge with Amazon Transcribe \(p. 183\)](#)
- [Using Amazon CloudWatch metrics and dimensions with Amazon Transcribe \(p. 186\)](#)

Monitoring Amazon Transcribe API calls with AWS CloudTrail

Amazon Transcribe is integrated with AWS CloudTrail, a service that provides a record of actions taken in Amazon Transcribe by an AWS Identity and Access Management (IAM) user or role, or by an AWS service. CloudTrail captures all API calls for Amazon Transcribe, including calls from the Amazon Transcribe console and from code calls to the Amazon Transcribe APIs, as events. By creating a trail, you can enable continuous delivery of CloudTrail events, including events for Amazon Transcribe, to an Amazon Simple Storage Service (Amazon S3) bucket. If you don't create a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can see each request that was made to Amazon Transcribe, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the [AWS CloudTrail User Guide](#).

Amazon Transcribe information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Amazon Transcribe, that activity is recorded in a CloudTrail event along with other AWS service events in the CloudTrail **Event history**. You can view, search, and download recent events in your AWS account. For more information, see [Viewing Events with CloudTrail Event History](#).

For an ongoing record of events in your AWS account, including events for Amazon Transcribe, create a trail. A *trail* is a configuration that enables CloudTrail to deliver events as log files to a specified S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see:

- [Overview for Creating a Trail](#)
- [CloudTrail Supported Services and Integrations](#)
- [Configuring Amazon SNS Notifications for CloudTrail](#)
- [Receiving CloudTrail Log Files from Multiple Regions](#) and [Receiving CloudTrail Log Files from Multiple Accounts](#)

CloudTrail logs all Amazon Transcribe actions, which are documented in the [API Reference \(p. 275\)](#). For example, calls to the [CreateVocabulary \(p. 291\)](#), [GetTranscriptionJob \(p. 329\)](#), and [StartTranscriptionJob \(p. 376\)](#) APIs generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. This information helps you determine the following:

- Whether the request was made with root or IAM user credentials
- Whether the request was made with temporary security credentials for an IAM role or federated user
- Whether the request was made by another AWS service

For more information, see the [CloudTrail userIdentity Element](#).

You can also aggregate Amazon Transcribe log files from multiple AWS Regions and multiple AWS accounts into a single S3 bucket. For more information, see [Receiving CloudTrail Log Files from Multiple Regions](#) and [Receiving CloudTrail Log Files from Multiple Accounts](#).

Example: Amazon Transcribe log file entries

A *trail* is a configuration that enables delivery of events as log files to a specified S3 bucket. CloudTrail log files contain one or more log entries. An *event* represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

Calls to the [StartTranscriptionJob](#) (p. 376) and [GetTranscriptionJob](#) (p. 329) APIs create the following entry.

```
{  
    "Records": [  
        {  
            "eventVersion": "1.05",  
            "userIdentity": {  
                "type": "AssumedRole | FederatedUser | IAMUser | Root | SAMLUser | WebIdentityUser",  
                "principalId": "principal ID",  
                "arn": "ARN",  
                "accountId": "account ID",  
                "accessKeyId": "access key",  
                "userName": "user name"  
            },  
            "eventTime": "timestamp",  
            "eventSource": "transcribe.amazonaws.com",  
            "eventName": "StartTranscriptionJob",  
            "awsRegion": "region",  
            "sourceIPAddress": "source IP address",  
            "userAgent": "user agent",  
            "requestParameters": {  
                "mediaFormat": "flac | mp3 | mp4 | wav",  
                "languageCode": "en-US | es-US",  
                "transcriptionJobName": "unique job name",  
                "media": {  
                    "mediaFileUri": ""  
                }  
            },  
            "responseElements": {  
                "transcriptionJob": {  
                    "transcriptionJobStatus": "IN_PROGRESS",  
                    "mediaFormat": "flac | mp3 | mp4 | wav",  
                    "creationTime": "timestamp",  
                    "transcriptionJobName": "unique job name",  
                    "languageCode": "en-US | es-US",  
                    "media": {  
                        "mediaFileUri": ""  
                    }  
                }  
            }  
        }  
    ]  
}
```

```

        },
        "requestID": "request ID",
        "eventID": "event ID",
        "eventType": "AwsApiCall",
        "recipientAccountId": "account id"
    },
    {
        "eventVersion": "1.05",
        "userIdentity": {
            "type": "AssumedRole | FederatedUser | IAMUser | Root | SAMLUser | WebIdentityUser",
            "principalId": "principal ID",
            "arn": "ARN",
            "accountId": "account ID",
            "accessKeyId": "access key",
            "userName": "user name"
        },
        "eventTime": "timestamp",
        "eventSource": "transcribe.amazonaws.com",
        "eventName": "GetTranscriptionJob",
        "awsRegion": "region",
        "sourceIPAddress": "source IP address",
        "userAgent": "user agent",
        "requestParameters": {
            "transcriptionJobName": "unique job name"
        },
        "responseElements": {
            "transcriptionJob": {
                "settings": {

                },
                "transcriptionJobStatus": "COMPLETED | FAILED | IN_PROGRESS",
                "mediaFormat": "flac | mp3 | mp4 | wav",
                "creationTime": "timestamp",
                "transcriptionJobName": "unique job name",
                "languageCode": "en-US | es-US",
                "media": {
                    "mediaFileUri": ""
                },
                "transcript": {
                    "transcriptFileUri": ""
                }
            }
        },
        "requestID": "request ID",
        "eventID": "event ID",
        "eventType": "AwsApiCall",
        "recipientAccountId": "account id"
    }
}
]
}

```

Using Amazon EventBridge with Amazon Transcribe

With Amazon EventBridge, you can respond to state changes in your Amazon Transcribe jobs by triggering events in other AWS services. When a transcription job changes state, EventBridge automatically sends an event to an event stream. You create rules that define the events that you want to monitor in the event stream and the action that EventBridge should take when those events occur. For example, routing the event to another service (or target), which can then take an action. You could, for example, configure a rule to route an event to an AWS Lambda function when a transcription job has completed successfully.

Before using EventBridge, you should understand the following concepts:

- **Event** – An event indicates a change in the state of one of your transcription jobs. For example, when the `TranscriptionJobStatus` of a job changes from `IN_PROGRESS` to `COMPLETED`.
- **Target** – A target is another AWS service that processes an event. For example, AWS Lambda or Amazon Simple Notification Service (Amazon SNS). A target receives events in JSON format.
- **Rule** – A rule matches incoming events that you want EventBridge to watch for and routes them to a target or targets for processing. If a rule routes an event to multiple targets, all of the targets process the event in parallel. A rule can customize the JSON sent to the target.

EventBridge events are emitted on a best-effort basis. For more information about creating and managing events in EventBridge, see [What is Amazon CloudWatch Events](#) in the *Amazon CloudWatch User Guide*.

Defining EventBridge rules

To define EventBridge rules, use the [CloudWatch Events console](#). When you define a rule, use Amazon Transcribe as the service name. For an example of how to create an EventBridge rule, see [Creating a CloudWatch Events Rule That Triggers on an Event](#) in the *Amazon CloudWatch User Guide*.

The following is an example of an EventBridge rule for Amazon Transcribe. It's triggered when a transcription job's status changes to `COMPLETED` or `FAILED`.

```
{  
  "source": [  
    "aws.transcribe"  
,  
  "detail-type": [  
    "Transcribe Job State Change"  
,  
  "detail": {  
    "TranscriptionJobStatus": [  
      "COMPLETED",  
      "FAILED"  
    ]  
  }  
}
```

The rule contains the following fields:

- `source` – The source of the event. For Amazon Transcribe, this is always `aws.transcribe`.
- `detail-type` – An identifier for the details of the event. For Amazon Transcribe, this is always `Transcribe Job State Change`.
- `detail` – The new job status of the transcription job. In this example, the rule triggers an event when the job status changes to `COMPLETED` or `FAILED`. For a list of status values, see the `TranscriptionJobStatus` field of the [TranscriptionJob \(p. 459\)](#) API.

Amazon Transcribe events

Amazon CloudWatch logs three kinds of Amazon Transcribe events: transcription job events, automatic language identification events, and call analytics events.

Transcription job events

When a job's state changes from `IN_PROGRESS` to either `COMPLETED` or `FAILED`, Amazon Transcribe generates an event. To identify the job that changed state and triggered the event in your target, use the event's `TranscriptionJobName` field. An Amazon Transcribe event contains the following information.

```
{
```

```
{  
    "version": "0",  
    "id": "event ID",  
    "detail-type": "Transcribe Job State Change",  
    "source": "aws.transcribe",  
    "account": "account ID",  
    "time": "timestamp",  
    "region": "region",  
    "resources": [],  
    "detail": {  
        "TranscriptionJobName": "unique job name",  
        "TranscriptionJobStatus": "status"  
    }  
}
```

Language identification events

When you enable automatic language identification, Amazon Transcribe generates an event when the language identification state is either COMPLETED or FAILED. To identify the job that changed state and triggered the event in your target, use the event's JobName field. An Amazon Transcribe event contains the following information.

```
{  
    "version": "0",  
    "id": "event ID",  
    "detail-type": "Language Identification State Change",  
    "source": "aws.transcribe",  
    "account": "account ID",  
    "time": "timestamp",  
    "region": "region",  
    "resources": [],  
    "detail": {  
        "JobType": "TranscriptionJob",  
        "JobName": "unique job name",  
        "LanguageIdentificationStatus": "status"  
    }  
}
```

Call analytics events

When a call analytics job's state changes from IN_PROGRESS to either COMPLETED or FAILED, Amazon Transcribe generates an event. To identify the call analytics job that changed state and triggered the event in your target, use the event's JobName field. An Amazon Transcribe event contains the following information.

```
{  
    "version": "0",  
    "id": "event ID",  
    "detail-type": "Call Analytics Job State Change",  
    "source": "aws.transcribe",  
    "account": "account ID",  
    "time": "timestamp",  
    "region": "region",  
    "resources": [],  
    "detail": {  
        "JobName": "unique job name",  
        "JobStatus": "status"  
    }  
}
```

Descriptions for the fields listed above:

- **version** – The version of the event data. This value is always 0.
- **id** – A unique identifier generated by CloudWatch Events for the event.
- **detail-type** – An identifier for the details of the event. For Amazon Transcribe, this is either Transcribe Job State Change, Language Identification State Change, or Call Analytics Job State Change.
- **source** – The source of the event. For Amazon Transcribe this is always `aws.transcribe`.
- **account ID** – The AWS account ID of the account that generated the API call.
- **timestamp** – The date and time that the API call was made.
- **region** – The AWS Region where the API call was made.
- **resources** – The resources used by the API call. For Amazon Transcribe, this field is always empty.
- **detail** – Details about the event. The fields within the `detail` tag may vary depending on the type of event, but may contain the following fields:
 - **JobType** – Enables reusing the event for transcription jobs.
 - **JobName** – The unique name that you gave the transcription job.
 - **JobStatus** – The status of your call analytics transcription job. It can be either `COMPLETED` or `FAILED`.
 - **TranscriptionJobName** – The unique name that you gave the job.
 - **TranscriptionJobStatus** – The new status of the transcription job. For a list of status values, see the [TranscriptionJobStatus](#) field of the [TranscriptionJob \(p. 459\)](#) API.
 - **LanguageIdentificationStatus** – The status of language identification in a transcription job. It can be either `COMPLETED` or `FAILED`.

Using Amazon CloudWatch metrics and dimensions with Amazon Transcribe

Amazon Transcribe supports CloudWatch metrics and dimensions, which are data that can help you monitor performance; supported metrics categories include traffic, errors, data transfer, and latency associated with your transcription jobs. Supported metrics are located through CloudWatch in the **AWS/Transcribe** namespace.

Note

CloudWatch monitoring metrics are free of charge and don't count against CloudWatch service quotas.

For more information on CloudWatch metrics, see [Using Amazon CloudWatch metrics](#).

Compliance validation for Amazon Transcribe

Third-party auditors assess the security and compliance of Amazon Transcribe as part of multiple AWS compliance programs. These include PCI, FedRAMP, HIPAA, and others. You can download third-party audit reports using AWS Artifact. For more information, see [Downloading Reports in AWS Artifact](#).

Your compliance responsibility when using Amazon Transcribe is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- [Security and compliance quick start guides](#) – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.

- [Architecting for HIPAA Security and Compliance Whitepaper](#) – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- [AWS Compliance resources](#) – This collection of workbooks and guides might apply to your industry and location.
- [AWS Config](#) – This AWS service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- [AWS Security Hub](#) – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

For a list of AWS services in scope of specific compliance programs, see [AWS Services in Scope by Compliance Program](#). For general information, see [AWS Compliance Programs](#).

Resilience in Amazon Transcribe

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see [AWS Global Infrastructure](#).

Infrastructure security in Amazon Transcribe

As a managed service, Amazon Transcribe is protected by the AWS global network security procedures that are described in the [Amazon Web Services: Overview of Security Processes](#) whitepaper.

To access Amazon Transcribe through the network, you use AWS published API calls. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS), such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems, such as Java 7 and later, support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an AWS Identity and Access Management (IAM) principal. Or you can use the [AWS Security Token Service](#) (AWS STS) to generate temporary security credentials to sign requests.

Amazon Transcribe Medical

Contents

- [What is Amazon Transcribe Medical? \(p. 188\)](#)
- [How Amazon Transcribe Medical works \(p. 189\)](#)
- [Getting started with Amazon Transcribe Medical \(p. 194\)](#)
- [Establish a bi-directional connection using the WebSocket protocol \(p. 215\)](#)

What is Amazon Transcribe Medical?

Amazon Transcribe Medical is an automatic speech recognition (ASR) service designed for medical professionals who wish to transcribe medical-related speech, such as physician-dictated notes, drug safety monitoring, telemedicine appointments, or physician-patient conversations. Amazon Transcribe Medical is available through either real-time streaming (via microphone) or transcription of an uploaded file (batch).

Important

Amazon Transcribe Medical is not a substitute for professional medical advice, diagnosis, or treatment. Identify the right confidence threshold for your use case, and use high confidence thresholds in situations that require high accuracy. For certain use cases, results should be reviewed and verified by appropriately trained human reviewers. Amazon Transcribe Medical transcriptions should only be used in patient care scenarios after review for accuracy and sound medical judgment by trained medical professionals.

Amazon Transcribe Medical operates under a shared responsibility model, whereby AWS is responsible for protecting the infrastructure that runs Amazon Transcribe Medical and you are responsible for managing your data. For more information, see [Shared Responsibility Model](#).

Amazon Transcribe Medical is available in US English (en-US).

For analysis of your transcripts, you can use other AWS services, such as [Amazon Comprehend Medical](#).

Supported specialties

Specialty	Sub-specialty	Audio input
Cardiology	none	streaming only
Neurology	none	streaming only
Oncology	none	streaming only
Primary Care	Family Medicine	batch, streaming
Primary Care	Internal Medicine	batch, streaming
Primary Care	Obstetrics and Gynecology (OB-GYN)	batch, streaming
Primary Care	Pediatrics	batch, streaming
Radiology	none	streaming only
Urology	none	streaming only

Streaming audio transcription

Amazon Transcribe Medical supports the real-time transcription of medical speech, referred to as a streaming transcription. You can start a streaming transcription using the [Amazon Transcribe console](#) or a WebSocket protocol. When you send Amazon Transcribe Medical an audio stream, it returns a stream of JSON objects containing the audio transcription.

For information about processing audio streams, see [Establish a bi-directional connection using the WebSocket protocol \(p. 215\)](#).

Batch transcription

Amazon Transcribe Medical supports the transcription of individual audio files through asynchronous API batch calls. For more information about transcribing audio files, see [the section called "How it works" \(p. 189\)](#).

Batch transcription can improve your transcription results. You can also use channel identification to separate and transcribe channels in audio files that have multiple channels. Channel identification generates separate transcripts for each channel and merges them into a single output. To identify speakers in single-channel audio files, use speaker identification. To generate additional alternative transcription results for the same source audio, use alternative transcriptions.

How Amazon Transcribe Medical works

Amazon Transcribe Medical enables you to transcribe individual audio content with medical information into text. You can use either the streaming transcription API or the batch transcription API. Streaming transcription enables you to transcribe real-time streams of audio into text. To transcribe recorded audio files, use batch transcription. For more information on transcribing streaming audio, see [Streaming transcription overview \(p. 190\)](#) and [Establish a bi-directional connection using the WebSocket protocol \(p. 215\)](#). For more information on batch transcription, see [Batch transcription overview \(p. 190\)](#)

Speech input

Amazon Transcribe Medical can transcribe speech as either an audio file or a real-time stream. Your input audio must use the encodings and formats described in the following sections.

Topics

- [Containers and formats for batch transcription \(p. 189\)](#)
- [Audio containers and formats for streaming transcription \(p. 6\)](#)

Containers and formats for batch transcription

When you transcribe an audio file using the [StartMedicalTranscriptionJob \(p. 370\)](#) API or the Amazon Transcribe Medical console, make sure that the file is:

- In FLAC, MP3, MP4, Ogg, WebM, AMR, or WAV file format
- Less than 4 hours in length and less than 2 GB in size
- Encoded at a sample rate of 16,000 Hz or higher

Note

For AMR, Amazon Transcribe Medical supports both Adaptive Multi-Rate Wideband (AMR-WB) and Adaptive Multi-Rate Narrowband (AMR-NB) codecs.

For the Ogg and WebM file formats, Amazon Transcribe Medical supports the Opus codec.

For best results:

- Use a lossless format. You can choose either FLAC, or WAV with PCM 16-bit encoding.

Audio containers and formats for streaming transcription

When you transcribe a real-time stream using the [StartMedicalStreamTranscription \(p. 402\)](#) API or a WebSocket request, make sure that your stream is encoded in:

- PCM 16-bit signed little endian
- FLAC
- OPUS encoded audio in the Ogg container

Your stream must use a sample rate of 16,000 Hz or higher.

For best results:

- Use a lossless format, such as FLAC or PCM encoding.

For more information on using a WebSocket request to transcribe your streaming audio, see [Establish a bi-directional connection using the WebSocket protocol \(p. 215\)](#).

Streaming transcription overview

Streaming transcription takes a stream of your audio data and transcribes it in real time. It uses a bidirectional WebSocket connection so that the results of the transcription are returned to your application while you send more audio to Amazon Transcribe Medical. You can also use it when you have an audio file that you want to process as it is transcribed.

Streaming transcription is available in US English (en-US). It can produce transcriptions of accented English, spoken by non-native speakers. Streaming audio transcription comes with the following features:

- Transcribe 16 kHz audio in real time.
- Transcribe up to 4 hours of audio streams.
- Word-level timestamp in transcripts.
- Word-level confidence in transcripts.
- [Number normalization \(p. 192\)](#).
- Punctuation and true casing in transcripts.
- Supports both dictation and conversation speech types.

For more information, see [Establish a bi-directional connection using the WebSocket protocol \(p. 215\)](#).

Batch transcription overview

Amazon Transcribe Medical batch transcription is available in US English. It has the ability to transcribe accented English from non-native speakers. It supports the transcription of individual audio files. You start a transcription job with either the console or by direct API call.

You interact with Amazon Transcribe Medical using four main API resources. To start a medical transcription job, use the [StartMedicalTranscriptionJob \(p. 370\)](#) API. To retrieve information on a medical transcription job, use [GetMedicalTranscriptionJob \(p. 323\)](#). You list medical transcription jobs with [ListMedicalTranscriptionJobs \(p. 348\)](#). You delete a medical transcription job with [DeleteMedicalTranscriptionJob \(p. 305\)](#).

To transcribe an audio file, you use a transcription job. You store the file as an object in an Amazon Simple Storage Service (S3) bucket. The input file must:

- Be in FLAC, MP3, MP4, or WAV file format.
- Use 16-bit Linear PCM encoding.
- Be less than 4 hours in length and less than 2 GB in size
- Use a sample rate of at least 16,000 Hz.

For best results:

- Use a lossless format, such as FLAC or WAV.

When creating a medical transcription job, you specify the language, the medical specialty, and the audio type of the source file. You input US English (en-US) as the language and PRIMARYCARE as the medical specialty. Entering primary care as the value enables you to generate transcriptions from source audio in the following medical specialties:

- Family Medicine
- Internal Medicine
- Obstetrics and Gynecology (OB-GYN)
- Pediatrics

You have the choice between dictation and conversation for your audio type. Choose dictation for audio files where the physician is giving a report about a patient visit or procedure. Choose conversation for audio files that involve a conversation between a physician and a patient or a conversation between physicians.

To store the output of your transcription job, select an Amazon S3 bucket that you've already created. For more information on S3 buckets see [Getting Started with Amazon Simple Storage Service](#)

The following are the minimum number of request parameters to enter in the sample JSON.

```
{  
    "MedicalTranscriptionJobName": "job name",  
    "LanguageCode": "en-US",  
    "Media": {  
        "MediaFileUri": "s3://path to your audio file"  
    },  
    "OutputBucketName": "your output bucket name",  
    "Specialty": "PRIMARYCARE",  
    "Type": "CONVERSATION"  
}
```

Amazon Transcribe Medical enables you to generate alternative transcriptions. For more information, see [Generating alternative transcriptions \(p. 262\)](#).

You can also identify different speakers or channels in your audio. For more information, see [Identifying speakers and labeling their speech \(p. 228\)](#) and [Transcribing multi-channel audio \(p. 235\)](#).

Transcribing numbers

Amazon Transcribe Medical transcribes digits as numbers instead of words. For example, the spoken number "one thousand two hundred forty-two" is transcribed as 1242.

Numbers are transcribed according to the following rules.

Rule	Description
Convert cardinal numbers greater than 10 to numbers.	<ul style="list-style-type: none"> "Fifty five" > 55 "a hundred" > 100 "One thousand and thirty one" > 1031 "One hundred twenty-three million four hundred fifty six thousand seven hundred eight nine" > 123,456,789
Convert cardinal numbers followed by "million" or "billion" to numbers followed by a word when "million" or "billion" is not followed by a number.	<ul style="list-style-type: none"> "one hundred million" > 100 million "one billion" > 1 billion "two point three million" > 2.3 million
Convert ordinal numbers greater than 10 to numbers.	<ul style="list-style-type: none"> "Forty third" > 43rd "twenty sixth avenue" > 26th avenue
Convert fractions to their numeric format.	<ul style="list-style-type: none"> "a quarter" > 1/4 "three sixteenths" > 3/16 "a half" > 1/2 "a hundredth" > 1/100
Convert numbers less than 10 to digits if there are more than one in a row.	<ul style="list-style-type: none"> "three four five" > 345 "My phone number is four two five five five five one two one two" > 4255551212
Decimals are indicated by "dot" or "point."	<ul style="list-style-type: none"> "three hundred and three dot five" > 303.5 "three point twenty three" > 3.23 "zero point four" > 0.4 "point three" > 0.3
Convert the word "percent" after a number to a percent sign.	<ul style="list-style-type: none"> "twenty three percent" > 23% "twenty three point four five percent" > 23.45%
Convert the words "dollar," "US dollar," "Australian dollar," "AUD," or "USD" after a number to a dollar symbol before the number.	<ul style="list-style-type: none"> "one dollar and fifteen cents" > \$1.15 "twenty three USD" > \$23 "twenty three Australian dollars" > \$23
Convert the words "pounds," or "milligrams" to "lbs" or "mg".	<ul style="list-style-type: none"> "twenty three pounds" > 23 lbs "forty-five milligrams" > 45 mg
Convert the words "rupees," "Indian rupees," or "INR" after a number to rupee sign (₹) before the number.	<ul style="list-style-type: none"> "twenty three rupees" > ₹23 "fifty rupees thirty paise" > ₹50.30
Convert times to numbers.	<ul style="list-style-type: none"> "seven a m eastern standard time" > 7 a.m. eastern standard time "twelve thirty p m" > 12:30 p.m.

Rule	Description
Combine years expressed as two digits into four. Only valid for the 20th, 21st, and 22nd centuries.	<ul style="list-style-type: none"> "nineteen sixty two" > 1962 "the year is twenty twelve" > the year is 2012 "twenty nineteen" > 2019 "twenty one thirty" > 2130
Convert dates to numbers.	<ul style="list-style-type: none"> "May fifth twenty twelve" > May 5th 2012 "May five twenty twelve" > May 5 2012 "five May twenty twelve" > 5 May 2012
Separate spans of numbers by the word "to."	<ul style="list-style-type: none"> "twenty three to thirty seven" > 23 to 37

Transcribing medical terms and measurements

Amazon Transcribe Medical can transcribe medical terms and measurements. Amazon Transcribe Medical outputs abbreviations for spoken terms. For example, "blood pressure" is transcribed as BP. You can find a list of conventions that Amazon Transcribe Medical uses for medical terms and measurements in the table on this page. The *Spoken Term* column refers to the term spoken in the source audio. The *Output* column refers to the abbreviation you see in your transcription results.

You can see how the terms spoken in source audio correspond to the transcription output here.

Term spoken in source audio	Abbreviation used in output	Example output
Centigrade	C	The patient's temperature is 37.4 C.
Celsius	C	The patient's temperature is 37.4 C.
Fahrenheit	F	The patient's temperature is 101 F.
grams	g	A mass of 100 g was extracted from the patient.
meters	m	The patient is 1.8 m tall.
feet	ft	The patient is 6 ft tall.
kilos	kg	The patient weighs 80 kg.
kilograms	kg	The patient weighs 80 kg.
c c	cc	Patient received 100 cc of saline solution.
cubic centimeter	cc	Patient received 100 cc of saline solution.
milliliter	mL	Patient excreted 100 mL urine.
blood pressure	BP	Patient BP was elevated.
b p	BP	Patient BP was elevated.

Term spoken in source audio	Abbreviation used in output	Example output
X over Y	X/Y	Patient BP was 120/80.
beats per min	BPM	Patient had atrial fibrillation with heart rate of 160 BPM.
beats per minute	BPM	Patient had atrial fibrillation with heart rate of 160 BPM.
O 2	O2	Patient O2 saturation was 98%.
CO2	CO2	Patient required respiratory support for elevated CO2.
post operation	POSTOP	Patient came for POSTOP evaluation.
post op	POSTOP	Patient came for POSTOP evaluation.
cat scan	CT Scan	Patient indication of cerebral hemorrhage required use of CT Scan.
Pulse 80	P 80	Patient vitals were P 80, R 17,...
Respiration 17	R 17	Patient vitals were P 80, R 17,...
in and out	I/O	Patient was I/O sinus rhythm
L five	L5	Lumbar puncture was performed between L4 and L5

Getting started with Amazon Transcribe Medical

To get started using Amazon Transcribe Medical, set up an AWS account and create an AWS Identity and Access Management (IAM) user.

Topics

- [Set up an AWS account and create an administrator user \(p. 194\)](#)
- [Getting started with console streaming \(p. 196\)](#)
- [Getting started with batch transcription \(p. 196\)](#)

Set up an AWS account and create an administrator user

Before you use Amazon Transcribe Medical for the first time, complete the following tasks:

1. [Sign up for AWS \(p. 195\)](#)
2. [Create an IAM user \(p. 195\)](#)

Sign up for AWS

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all AWS services, including Amazon Transcribe Medical. You are charged only for the services that you use.

With Amazon Transcribe Medical, you pay only for the resources that you use. If you are a new AWS customer, you can get started with Amazon Transcribe Medical for free. For more information, see [AWS Free Usage Tier](#).

If you already have an AWS account, skip to the next section.

To create an AWS account

1. Open <https://portal.aws.amazon.com/billing/signup>.
2. Follow the online instructions.

Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

Record your AWS account ID because you'll need it for the next task.

Create an IAM user

Services in AWS, such as Amazon Transcribe Medical, require that you provide credentials when you access them. This allows the service to determine whether you have permissions to access the service's resources.

We strongly recommend that you access AWS using AWS Identity and Access Management (IAM), not the credentials for your AWS account. To use IAM to access AWS, create an IAM user, add the user to an IAM group with administrative permissions, and then grant administrative permissions to the IAM user. You can then access AWS using a special URL and the IAM user's credentials.

The Getting Started exercises in this guide assume that you have a user with administrator privileges, `adminuser`.

To create an administrator user and sign in to the console

1. Create an administrator user called `adminuser` in your AWS account. For instructions, see [Creating Your First IAM User and Administrators Group](#) in the *IAM User Guide*.
2. Sign in to the AWS Management Console using a special URL. For more information, see [How Users Sign In to Your Account](#) in the *IAM User Guide*.

For more information about IAM, see the following:

- [AWS Identity and Access Management \(IAM\)](#)
- [Getting started](#)
- [IAM User Guide](#)

Next step

To get started with medical transcription using the console, see [Getting Started \(Console\) \(p. 196\)](#).

Getting started with console streaming

Get started with Amazon Transcribe Medical streaming transcription by using the console to transcribe up to 15 minutes of medical speech into text. You use the microphone on your computer for the audio source.

To start a streaming medical transcription job

1. Open the [Amazon Transcribe console](#).
2. From the left menu, look under **Amazon Transcribe Medical** and choose **Real-time transcription**.
3. Choose **Start streaming** and then speak into your microphone. Amazon Transcribe Medical will transcribe the speech and display the results on the console.

Next step

To learn more about WebSocket streaming, go to [WebSocket Streaming \(p. 215\)](#).

Getting started with batch transcription

Get started with Amazon Transcribe Medical batch transcription in the console to transcribe your audio file. To start a transcription job, your audio file must be stored in Amazon S3 bucket and its file format must be either FLAC, WAV, MP3, or MP4.

To start a batch medical transcription job

1. Open the [Amazon Transcribe console](#).
2. From the left menu, look under **Amazon Transcribe Medical** and choose **Transcription jobs**.
3. Choose **Create job**
4. Specify the details of your job. Choose the **Info** links if you need help
5. When you finished specifying the details of your job, choose **Create** to start a medical transcription job.

[Batch transcription overview \(p. 190\)](#)

Streaming transcription

Amazon Transcribe Medical streaming transcription enables you to send an audio stream and receive a stream of text in real time. The API makes it easy for developers to add real-time speech-to-text capability to their applications.

The following table shows which languages are available for streaming transcription and how you can access them.

Language	Sample rate	Available in
US English (en-US)	16 kHz, 8 kHz	Amazon Transcribe Medical console , StartMedicalStreamTranscription (p. 402) API, and WebSocket request

If you are using HTTP/2, we provide an HTTP/2 streaming client that handles retrying the connection when there are transient problems on the network. You can use this client as a starting point for your own applications. To use Amazon Transcribe Medical streaming with the WebSocket protocol, you can create your own client.

Streaming transcription takes a stream of your audio data and transcribes it in real time. The transcription is returned to your application in a stream of transcription events.

Amazon Transcribe Medical breaks your incoming audio stream based on natural speech segments, such as a change in speaker or a pause in the audio. The transcription is returned progressively to your application, with each response containing more transcribed speech until the entire segment is transcribed.

In the following example, each line is a partial result transcription output of an audio segment being streamed.

```
The
The PE.
The pain.
The patient.
The patient was
The patient was in
The patient was entered.
The patient was entered, and, uh
The patient was I/O.
The patient was I/O of
The patient was I/O of some
The patient was I/O sign.
The patient was I/O of Sinus.
The patient was I/O of Sinus rhyth.
The patient was I/O of Sinus rhyth.
The patient was I/O of Sinus rhyth with
```

Each [Result](#) object in the response contains a field called `IsPartial` that indicates whether the response is a partial response containing the transcription results so far or if it is a complete transcription of the audio segment.

Each [Result](#) object also contains the start time and end time of the term from the audio stream so that you can, for example, synchronize the transcription with the video.

The following example is a partial transcription response.

```
{
    "Transcript": {
        "Results": [
            {
                "Alternatives": [
                    {
                        "Items": [
                            {
                                "Content": "The",
                                "EndTime": 1.07,
                                "StartTime": 1.04,
                                "Type": "pronunciation",
                                "VocabularyFilterMatch": false
                            },
                            {
                                "Content": "patient",
                                "EndTime": 1.5,
                                "StartTime": 1.08,
                                "Type": "pronunciation"
                            }
                        ]
                    }
                ]
            }
        ]
    }
}
```

```

        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Content": "was",
        "EndTime": 1.61,
        "StartTime": 1.51,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Content": "I/O",
        "EndTime": 2.25,
        "StartTime": 2.06,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Content": "of",
        "EndTime": 2.34,
        "StartTime": 2.26,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Content": "Sinus",
        "EndTime": 2.71,
        "StartTime": 2.35,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Content": "rhythm",
        "EndTime": 3.07,
        "StartTime": 2.72,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    },
    {
        "Content": "with",
        "EndTime": 3.68,
        "StartTime": 3.49,
        "Type": "pronunciation",
        "VocabularyFilterMatch": false
    }
],
"Transcript": "The patient was I/O of Sinus rhythm with"
}
],
"EndTime": 3.75,
"IsPartial": true,
"ResultId": "93b1df2b-8702-4c91-892a-ace3b65a6477",
"StartTime": 1.04
}
]
}
}

```

The following example shows the transcription results for a fully transcribed speech segment.

```
{
    "Transcript": {
        "Results": [
            {
                "Alternatives": [

```

```
{  
    "Items": [  
        {  
            "Confidence": 0.99,  
            "Content": "The",  
            "EndTime": 1.12,  
            "StartTime": 1.04,  
            "Type": "pronunciation",  
            "VocabularyFilterMatch": false  
        },  
        {  
            "Confidence": 0.99,  
            "Content": "patient",  
            "EndTime": 1.53,  
            "StartTime": 1.13,  
            "Type": "pronunciation",  
            "VocabularyFilterMatch": false  
        },  
        {  
            "Confidence": 1,  
            "Content": "was",  
            "EndTime": 1.73,  
            "StartTime": 1.54,  
            "Type": "pronunciation",  
            "VocabularyFilterMatch": false  
        },  
        {  
            "Confidence": 1,  
            "Content": "I/O",  
            "EndTime": 2.3,  
            "StartTime": 2.12,  
            "Type": "pronunciation",  
            "VocabularyFilterMatch": false  
        },  
        {  
            "Confidence": 1,  
            "Content": "of",  
            "EndTime": 2.39,  
            "StartTime": 2.31,  
            "Type": "pronunciation",  
            "VocabularyFilterMatch": false  
        },  
        {  
            "Confidence": 1,  
            "Content": "Sinus",  
            "EndTime": 2.82,  
            "StartTime": 2.4,  
            "Type": "pronunciation",  
            "VocabularyFilterMatch": false  
        },  
        {  
            "Confidence": 0.99,  
            "Content": "rhythm",  
            "EndTime": 3.32,  
            "StartTime": 2.83,  
            "Type": "pronunciation",  
            "VocabularyFilterMatch": false  
        },  
        {  
            "Confidence": 0.99,  
            "Content": "with",  
            "EndTime": 3.72,  
            "StartTime": 3.49,  
            "Type": "pronunciation",  
            "VocabularyFilterMatch": false  
        },  
    ]  
}
```

```
{
    "Confidence": 0.99,
    "Content": "a",
    "EndTime": 3.78,
    "StartTime": 3.73,
    "Type": "pronunciation",
    "VocabularyFilterMatch": false
},
{
    "Confidence": 1,
    "Content": "heart",
    "EndTime": 4.02,
    "StartTime": 3.79,
    "Type": "pronunciation",
    "VocabularyFilterMatch": false
},
{
    "Confidence": 1,
    "Content": "rate",
    "EndTime": 4.19,
    "StartTime": 4.03,
    "Type": "pronunciation",
    "VocabularyFilterMatch": false
},
{
    "Confidence": 0.99,
    "Content": "of",
    "EndTime": 4.26,
    "StartTime": 4.2,
    "Type": "pronunciation",
    "VocabularyFilterMatch": false
},
{
    "Confidence": 1,
    "Content": "75",
    "EndTime": 4.81,
    "StartTime": 4.27,
    "Type": "pronunciation",
    "VocabularyFilterMatch": false
},
{
    "Confidence": 0.97,
    "Content": "bpm",
    "EndTime": 5.47,
    "StartTime": 4.82,
    "Type": "pronunciation",
    "VocabularyFilterMatch": false
},
{
    "Content": ".",
    "EndTime": 5.47,
    "StartTime": 5.47,
    "Type": "punctuation",
    "VocabularyFilterMatch": false
}
],
"Transcript": "The patient was I/O of Sinus rhythm with a heart
rate of 75 bpm."
}
],
"EndTime": 5.53,
"IsPartial": false,
"ResultId": "93b1df2b-8702-4c91-892a-ace3b65a6477",
"StartTime": 1.04
}
]
```

```
}
```

Each word, phrase or punctuation mark in the transcription output is an *item*. Each word or phrase has a *confidence* score. The confidence score is a value between 0 and 1 that indicates how confident Amazon Transcribe Medical is that it correctly transcribed the item. A confidence score with a larger value indicates that Amazon Transcribe Medical is more confident that it transcribed the item correctly.

The preceding example shows "I/O" in the transcription output, which is an abbreviation for "in and out". For more information on how Amazon Transcribe Medical uses abbreviations in the transcription output, see [Transcribing medical terms and measurements \(p. 193\)](#).

Topics

- [Event stream encoding \(p. 201\)](#)
- [Using Amazon Transcribe Medical streaming with HTTP/2 \(p. 203\)](#)
- [Establish a bi-directional connection using the WebSocket protocol \(p. 215\)](#)

Event stream encoding

Event stream encoding provides bidirectional communication using messages between a client and a server. Data frames sent to the Amazon Transcribe Medical streaming service are encoded in this format. The response from Amazon Transcribe Medical also uses this encoding.

Each message consists of two sections: the prelude and the data. The prelude consists of:

1. The total byte length of the message
2. The combined byte length of all of the headers

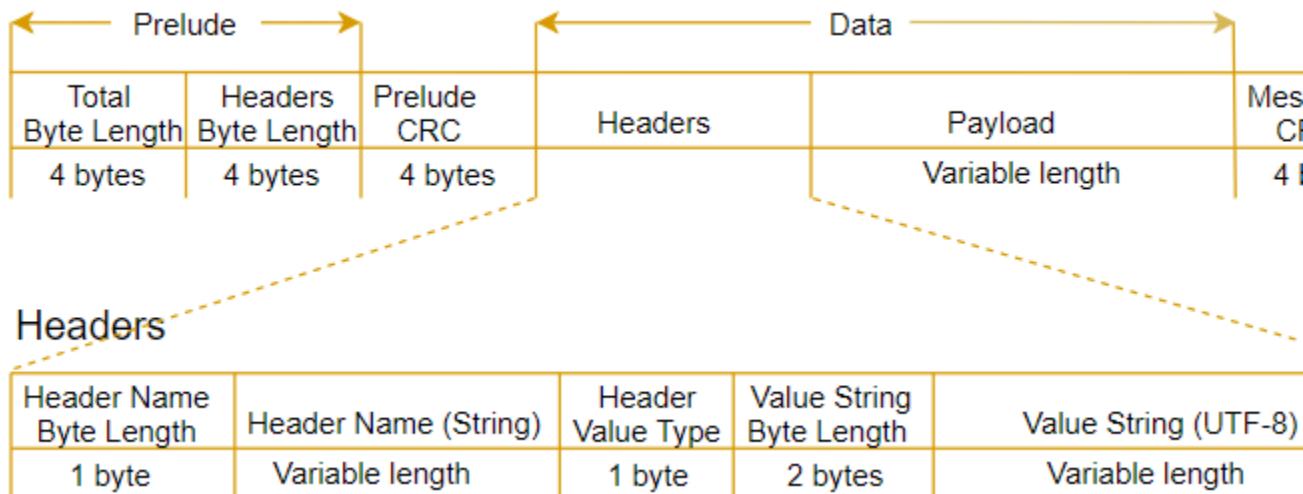
The data section consists of:

1. The headers
2. A payload

Each section ends with a 4-byte big-endian integer CRC checksum. The message CRC checksum is for both the prelude section and the data section. Amazon Transcribe Medical uses CRC32 (often referred to as GZIP CRC32) to calculate both CRCs. For more information about CRC32, see [GZIP file format specification version 4.3](#).

Total message overhead, including the prelude and both checksums, is 16 bytes.

The following diagram shows the components that make up a message and a header. There are multiple headers per message.



Each message contains the following components:

- **Prelude:** Always a fixed size of 8 bytes, two fields of 4 bytes each.
 - *First 4 bytes:* The total byte-length. This is the big-endian integer byte-length of the entire message, including the 4-byte length field itself.
 - *Second 4 bytes:* The headers byte-length. This is the big-endian integer byte-length of the headers portion of the message, excluding the headers length field itself.
- **Prelude CRC:** The 4-byte CRC checksum for the prelude portion of the message, excluding the CRC itself. The prelude has a separate CRC from the message CRC to ensure that Amazon Transcribe Medical can detect corrupted byte-length information immediately without causing errors such as buffer overruns.
- **Headers:** Metadata annotating the message, such as the message type, content type, and so on. Messages have multiple headers. Headers are key-value pairs where the key is a UTF-8 string. Headers can appear in any order in the headers portion of the message and any given header can appear only once. For the required header types, see the following sections.
- **Payload:** The audio content to be transcribed.
- **Message CRC:** The 4-byte CRC checksum from the start of the message to the start of the checksum. That is, everything in the message except the CRC itself.

Each header contains the following components. There are multiple headers per frame.

- **Header name byte-length:** The byte-length of the header name.
- **Header name:** The name of the header indicating the header type. For valid values, see the following frame descriptions.
- **Header value type:** An enumeration indicating the header value.

The following shows the possible values for the header and what they indicate.

- 0 – TRUE
- 1 – FALSE
- 2 – BYTE
- 3 – SHORT
- 4 – INTEGER
- 5 – LONG
- 6 – BYTE ARRAY

- 7 – STRING
- 8 – TIMESTAMP
- 9 – UUID
- **Value string byte length:** The byte-length of the header value string.
- **Header value:** The value of the header string. Valid values for this field depend on the type of header. For valid values, see the following frame descriptions.

Using Amazon Transcribe Medical streaming with HTTP/2

Amazon Transcribe Medical uses a format called *event stream encoding* for streaming-med transcription. This format encodes binary data with header information that describes the contents of each event. For more information, see [Event stream encoding \(p. 74\)](#). You can use this information for applications that call the Amazon Transcribe Medical endpoint without using the Amazon Transcribe Medical SDK.

When Amazon Transcribe Medical uses the [HTTP/2 protocol](#) for streaming medical transcriptions, the key components for a streaming medical request are:

- A header frame. This contains the HTTP/2 headers for the request, and a signature in the authorization header that Amazon Transcribe Medical uses as a seed signature to sign the following data frames.
- One or more message frames in event stream encoding. The frame contains metadata and the raw audio bytes.
- An end frame. This is a signed message in event stream encoding with an empty body.

Streaming request

To make a streaming request, you use the [StartMedicalStreamTranscription \(p. 402\)](#) API.

Header frame

The header frame is the authorization frame for the streaming transcription. Amazon Transcribe Medical uses the value of the authorization header as the seed for generating a chain of authorization headers for the data frames in the request.

Required headers

The header frame of a request to Amazon Transcribe Medical requires the following HTTP/2 headers.

```
POST /stream-transcription HTTP/2.0
host: transcribestreaming.region.amazonaws.com
authorization: Generated value
content-type: application/vnd.amazon.eventstream
x-amz-target: com.amazonaws.transcribe.Transcribe.StartStreamTranscription
x-amz-content-sha256: streaming-med-AWS4-HMAC-SHA256-EVENTS
x-amz-date: Date
x-amzn-transcribe-language-code: en-US
x-amzn-transcribe-media-encoding: media encoding
x-amzn-transcribe-sample-rate: Sample rate
transfer-encoding: chunked
```

In the request, use the following values for the host, authorization, and x-amz-date headers:

- **host:** Use the AWS Region where you are calling Amazon Transcribe Medical. For a list of valid regions, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.
- **authorization:** The Signature Version 4 signature for the request. For more information about creating a signature, see [Signing AWS Requests with Signature Version 4](#) in the *Amazon Web Services General Reference*.
- **x-amz-date:** Generate a date and time for the request following the instructions in [Handling Dates in Signature Version 4](#) in the *Amazon Web Services General Reference*.

For more information about the headers specific to Amazon Transcribe Medical, see the [StartMedicalStreamTranscription \(p. 402\)](#) API.

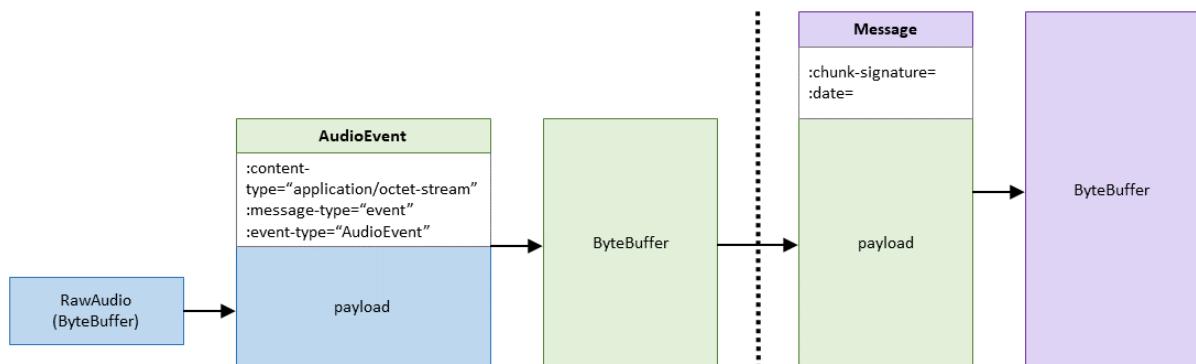
Data frames

Each request contains one or more data frames. The data frames use event stream encoding. The encoding supports bidirectional data transmission between a client and a server.

There are two steps to creating a data frame:

1. Combine the raw audio data with metadata to create the payload of the request.
2. Combine the payload with a signature to form the event message that is sent to Amazon Transcribe Medical.

The following diagram shows how this works.



Create the audio event

To create the message to send to Amazon Transcribe Medical, create the audio event. Combine the headers described in the following table with a chunk of audio bytes into an event-encoded message.

Header Name Byte Length	Header Name (string)	Header Value Type	Value String Byte Length	Value String (UTF-8)
13	:content-type	7	24	application/octet-stream
11	:event-type	7	10	AudioEvent
13	:message-type	7	5	event

To create the payload for the event message, use a buffer in raw-byte format.

Create the message

Create a data frame using the audio event payload to send to Amazon Transcribe Medical. The data frame contains event-encoding headers that include the current date and a signature for the audio chunk and the audio event. To indicate to Amazon Transcribe Medical that the audio stream is complete, send an empty data frame that contains only the date and signature.

Header Name Byte Length	Header Name (String)	Header Value Type	Value String Byte Length	Value
16	:chunk-signature	6	varies	Generated signature
5	:date	8	8	Timestamp

To create the signature for the data frame, first create a string to sign, and then calculate the signature for the event. Construct the string to sign as follows.

```
String stringToSign =
    "AWS4-HMAC-SHA256-PAYOUT" +
    "\n" +
    DATE +
    "\n" +
    KEYPATH +
    "\n" +
    Hex(priorSignature) +
    "\n" +
    HexHash(nonSignatureHeaders) +
    "\n" +
    HexHash(payload);
```

- **DATE:** The current date and time in Universal Time Coordinated (UTC) and using the [ISO 8601 format](#). Don't include milliseconds in the date. For example, 20190127T223754Z is 22:37:54 on 1/27/2019.
- **KEYPATH:** The signature scope in the format date/region/service/aws4_request. For example, 20190127/us-east-1/transcribe/aws4_request.
- **priorSignature:** The signature for the previous frame. For the first data frame, use the signature of the header frame.
- **nonSignatureHeaders:** The DATE header encoded as a string.
- **payload:** The byte buffer containing the audio event data.
- **Hex:** A function that encodes its input into a hexadecimal representation.
- **HexHash:** A function that first creates a SHA-256 hash of its input and then uses the Hex function to encode the hash.

After you have constructed the string to sign, sign it using the key that you derived for Signature Version 4, as follows. For details, see [Examples of How to Derive a Signing Key for Signature Version 4](#) in the [Amazon Web Services General Reference](#).

```
String signature = HMACSHA256(derivedSigningKey, stringToSign);
```

- **HMACSHA256:** A function that creates a signature using the SHA-256 hash function.
- **derivedSigningKey:** The Signature Version 4 signing key.
- **stringToSign:** The string that you calculated for the data frame.

After you have calculated the signature for the data frame, construct a byte buffer containing the date, the signature, and the audio event payload. Send the byte array to Amazon Transcribe Medical for transcription.

End frame

To indicate that the audio stream is complete, send an end frame to Amazon Transcribe Medical. The *end frame* is a data frame with an empty payload. You construct the end frame the same way that you construct a data frame.

Streaming response

Responses from Amazon Transcribe Medical are also sent using event stream encoding. Use this information to decode a response from the [StartMedicalStreamTranscription \(p. 402\)](#) API.

Transcription response

A transcription response is event stream encoded. It contains the standard prelude and the following headers.

Header Name Byte Length	Header Name (String)	Header Value Type	Value String Byte Length	Value String (UTF-8)
13	:content-type	7	24	application/octet-stream
11	:event-type	7	10	AudioEvent
13	:message-type	7	5	event

For details, see [Event stream encoding \(p. 74\)](#).

When the response is decoded, it contains the following information:

```
:content-type: "application/json"
:event-type: "TranscriptEvent"
:message-type: "event"

JSON transcription information
```

For an example of the JSON structure returned by Amazon Transcribe Medical, see [Using Amazon Transcribe Medical streaming with HTTP/2 \(p. 203\)](#).

Exception response

If there is an error in processing your transcription stream, Amazon Transcribe Medical sends an exception response. The response is event stream encoded. For details, see [Event stream encoding \(p. 74\)](#).

The response contains the standard prelude and the following headers.

Header Name Byte Length	Header Name (String)	Header Value Type	Value String Byte Length	Value String (UTF-8)
13	:content-type	7	16	application/json
11	:event-type	7	19	BadRequestException
13	:message-type	7	9	exception

When the exception response is decoded, it contains the following information.

```
:content-type: "application/json"  
:event-type: "BadRequestException"  
:message-type: "exception"
```

Exception message

Example request and response

The following is an end-to-end example of a streaming transcription request. In this example, binary data is represented as base64-encoded strings. In an actual response, the data are raw bytes.

Step 1: Start the session with Amazon Transcribe Medical

To start the session, send an HTTP/2 request to Amazon Transcribe Medical.

```
POST /stream-transcription HTTP/2.0  
host: transcribestreaming.region.amazonaws.com  
authorization: Generated value  
content-type: application/vnd.amazon.eventstream  
x-amz-content-sha256: streaming-med-AWS4-HMAC-SHA256-EVENTS  
x-amz-date: Date  
x-amzn-transcribe-language-code: en-US  
x-amzn-transcribe-media-encoding: Media encoding  
x-amzn-transcribe-sample-rate: Sample rate  
transfer-encoding: chunked
```

Step 2: Send authentication information to Amazon Transcribe Medical

Amazon Transcribe Medical sends the following response.

```
HTTP/2.0 200  
x-amzn-transcribe-language-code: en-US  
x-amzn-transcribe-sample-rate: Sample rate  
x-amzn-request-id: 8a08df7d-5998-48bf-a303-484355b4ab4e  
x-amzn-transcribe-session-id: b4526fcf-5eee-4361-8192-d1cb9e9d6887  
x-amzn-transcribe-media-encoding: pcm  
x-amzn-RequestId: 8a08df7d-5998-48bf-a303-484355b4ab4e  
content-type: application/vnd.amazon.eventstream
```

Step 3: Create an audio event

Create an audio event containing the audio data to send. For details, see [Event stream encoding \(p. 201\)](#). The binary data in this request is base64-encoded. In an actual request, the data is raw bytes.

```
:content-type: "application/octet-stream"  
:event-type: "AudioEvent"  
:message-type: "event"  
  
Uk1GRjzxPQBXQVZFZm10IBAAAAABAAEAgD4AAAB9AAACABAAGF0YVTwPQAAAAAAAAAAAAAAD//wIA/f8EAA==
```

Step 4: Create an audio event message

Create an audio message that contains the audio data to send to Amazon Transcribe Medical. For details, see [Event stream encoding \(p. 201\)](#). The audio event data in this example is base64-encoded. In an actual request, the data is raw bytes.

```
:date: 2019-01-29T01:56:17.291Z
:chunk-signature: signature

AAAA0gAAAIKV0RFcTTcjb250ZW50LXR5cGUHABhhcHBsaWNhdGlvbi9vY3RldC1zdHJ1YW0LOmV2ZW50LXR5
cGUHAApBdWRpb0V2ZW50DTptZXNzYwdllXR5cGUHAAVldmVudAxDb256ZW50LVR5cGUHABphcHBsaWNhdGlv
bi94LWFtei1qc29uLTEuMVJJRky88T0AV0FWRWZtdCAQAAAAQABAIA+AAAAfQAAAgaQAGRhdGFU8D0AAAAAA
AAAAAAAAAA//8CAP3/BAC7QLFF
```

Step 5: Use the response from Amazon Transcribe Medical

Amazon Transcribe Medical creates a stream of transcription events that it sends to your application. The events are sent in raw-byte format. In this example, the bytes are base64-encoded.

Amazon Transcribe Medical sends the following response.

```
AAAAAUwAAAEP1RHpyYBTpkYXR1CAAAAWixUkMLEDpjaHVuay1zaWduYXR1cmUGACct6Zy+uymwEK2SrLp/zVBI
5eGn83jdBwCaRUBJA+eaDafqjqI=
```

To see the transcription results, decode the raw bytes using event-stream encoding.

```
:event-type: "TranscriptEvent"
:content-type: "application/json"
:message-type: "event"

>{"Transcript": {"Results": [results]}}
```

For an example of the JSON structure returned by Amazon Transcribe Medical, see [Event stream encoding \(p. 201\)](#).

Step 6: End the transcription stream

Finally, send an empty audio event to Amazon Transcribe Medical to end the transcription stream. Create the audio event exactly like any other, except with an empty payload. Sign the event and include the signature in the :chunk-signature header, as follows.

```
:date: 2019-01-29T01:56:17.291Z
:chunk-signature: signature
```

HTTP/2 streaming retry client

You can use the following code in your applications to handle retry logic for Amazon Transcribe Medical streaming transcription. The code provides tolerance for intermittent failures in the connection to

Amazon Transcribe Medical. There are two parts of the client: an interface that you implement for your application, and the retry client itself.

Streaming retry client code

This code implements a streaming retry client. It manages the connection to Amazon Transcribe Medical and retries sending data when there are errors on the connection. For example, if there is a transient error on the network, this client resends the request that failed.

The retry client has two properties that control the behavior of the client. You can set:

- The maximum number of times that the client should attempt before failing. Reduce this value to make your application stop retrying sooner when there are network issues. The default is 10.
- The time in milliseconds that the client should wait between retries. Longer times raise the risk of losing data, shorter times raise the risk of your application being throttled. The default is 100 milliseconds.

The following is the client. You can copy this code to your application or use it as a starting point for your own client.

```
public class TranscribeStreamingRetryClient {  
  
    private static final int DEFAULT_MAX_RETRIES = 10;  
    private static final int DEFAULT_MAX_SLEEP_TIME_MILLS = 100;  
    private static final Logger log =  
        LoggerFactory.getLogger(TranscribeStreamingRetryClient.class);  
    private final TranscribeStreamingAsyncClient client;  
    List<Class<?>> nonRetriableExceptions = Arrays.asList(BadRequestException.class);  
    private int maxRetries = DEFAULT_MAX_RETRIES;  
    private int sleepTime = DEFAULT_MAX_SLEEP_TIME_MILLS;  
  
    /**  
     * Create a TranscribeStreamingRetryClient with given credential and configuration  
     */  
    public TranscribeStreamingRetryClient(AwsCredentialsProvider creds,  
                                         String endpoint, Region region) throws  
        URISyntaxException {  
        this(TranscribeStreamingAsyncClient.builder()  
            .overrideConfiguration(  
                c -> c.putAdvancedOption(  
                    SdkAdvancedClientOption.SIGNER,  
                    EventStreamAws4Signer.create()))  
            .credentialsProvider(creds)  
            .endpointOverride(new URI(endpoint))  
            .region(region)  
            .build());  
    }  
  
    /**  
     * Initiate TranscribeStreamingRetryClient with TranscribeStreamingAsyncClient  
     */  
    public TranscribeStreamingRetryClient(TranscribeStreamingAsyncClient client) {  
        this.client = client;  
    }  
  
    /**  
     * Get Max retries  
     */  
    public int getMaxRetries() {  
        return maxRetries;  
    }  
}
```

```

    /**
     * Set Max retries
     */
    public void setMaxRetries(int maxRetries) {
        this.maxRetries = maxRetries;
    }

    /**
     * Get sleep time
     */
    public int getSleepTime() {
        return sleepTime;
    }

    /**
     * Set sleep time between retries
     */
    public void setSleepTime(int sleepTime) {
        this.sleepTime = sleepTime;
    }

    /**
     * Initiate a Stream Transcription with retry.
     */
    public CompletableFuture<Void> startStreamTranscription(final
StartStreamTranscriptionRequest request,
                                                               final Publisher<AudioStream>
publisher,
                                                               final
StreamTranscriptionBehavior responseHandler) {

        CompletableFuture<Void> finalFuture = new CompletableFuture<>();

        recursiveStartStream(rebuildRequestWithSession(request), publisher,
responseHandler, finalFuture, 0);

        return finalFuture;
    }

    /**
     * Recursively call startStreamTranscription() until the request is completed or we run
out of retries.
     */
    private void recursiveStartStream(final StartStreamTranscriptionRequest request,
final Publisher<AudioStream> publisher,
final StreamTranscriptionBehavior responseHandler,
final CompletableFuture<Void> finalFuture,
final int retryAttempt) {
        CompletableFuture<Void> result = client.startStreamTranscription(request,
publisher,
            getResponseHandler(responseHandler));
        result.whenComplete((r, e) -> {
            if (e != null) {
                log.debug("Error occurred:", e);

                if (retryAttempt <= maxRetries && isExceptionRetriable(e)) {
                    log.debug("Retriable error occurred and will be retried.");
                    log.debug("Sleeping for sometime before retrying...");
                    try {
                        Thread.sleep(sleepTime);
                    } catch (InterruptedException e1) {
                        log.debug("Unable to sleep. Failed with exception: ", e);
                        e1.printStackTrace();
                    }
                }
            }
        });
    }
}

```

```

        }
        log.debug("Making retry attempt: " + (retryAttempt + 1));
        recursiveStartStream(request, publisher, responseHandler, finalFuture,
    retryAttempt + 1);
    } else {
        log.error("Encountered unretrievable exception or ran out of retries. ");
        responseHandler.onError(e);
        finalFuture.completeExceptionally(e);
    }
} else {
    responseHandler.onComplete();
    finalFuture.complete(null);
}
});
}

private StartStreamTranscriptionRequest
rebuildRequestWithSession(StartStreamTranscriptionRequest request) {
    return StartStreamTranscriptionRequest.builder()
        .languageCode(request.languageCode())
        .mediaEncoding(request.mediaEncoding())
        .mediaSampleRateHertz(request.mediaSampleRateHertz())
        .sessionId(UUID.randomUUID().toString())
        .build();
}

/**
 * StartStreamTranscriptionResponseHandler implements subscriber of transcript stream
 * Output is printed to standard output
 */
private StartStreamTranscriptionResponseHandler getResponseHandler(
    StreamTranscriptionBehavior transcriptionBehavior) {
    final StartStreamTranscriptionResponseHandler build =
StartStreamTranscriptionResponseHandler.builder()
    .onResponse(r -> {
        transcriptionBehavior.onResponse(r);
    })
    .onError(e -> {
        //Do nothing here. Don't close any streams that shouldn't be cleaned up
yet.
    })
    .onComplete(() -> {
        //Do nothing here. Don't close any streams that shouldn't be cleaned up
yet.
    })

    .subscriber(event -> transcriptionBehavior.onStream(event))
    .build();
    return build;
}

/**
 * Check if the exception can be retried.
 */
private boolean isExceptionRetriable(Throwable e) {
    e.printStackTrace();

    return nonRetriableExceptions.contains(e.getClass());
}

public void close() {
    this.client.close();
}
}

```

Streaming retry client interface code

This interface is similar to the response handler used in the getting started example. It implements the same event handlers. Implement this interface to use the streaming-med retry client.

```
package com.amazonaws.transcribestreaming;

import software.amazon.awssdk.services.transcribestreaming.model.StartStreamTranscriptionResponse;
import software.amazon.awssdk.services.transcribestreaming.model.TranscriptResultStream;

/**
 * Defines how a stream response should be handled.
 * You should build a class implementing this interface to define the behavior.
 */
public interface StreamTranscriptionBehavior {
    /**
     * Defines how to respond when encountering an error on the stream transcription.
     */
    void onError(Throwable e);

    /**
     * Defines how to respond to the Transcript result stream.
     */
    void onStream(TranscriptResultStream e);

    /**
     * Defines what to do on initiating a stream connection with the service.
     */
    void onResponse(StartStreamTranscriptionResponse r);

    /**
     * Defines what to do on stream completion
     */
    void onComplete();
}
```

The following is an example implementation of the `StreamTranscriptionBehavior` interface. You can use this implementation or use it as a starting point for your own implementation.

```
package com.amazonaws.transcribestreaming.retryclient;

import com.amazonaws.transcribestreaming.retryclient.StreamTranscriptionBehavior;
import software.amazon.awssdk.services.transcribestreaming.model.Result;
import software.amazon.awssdk.services.transcribestreaming.model.StartStreamTranscriptionResponse;
import software.amazon.awssdk.services.transcribestreaming.model.TranscriptEvent;
import software.amazon.awssdk.services.transcribestreaming.model.TranscriptResultStream;

import java.util.List;

/**
 * Implementation of StreamTranscriptionBehavior to define how a stream response should be
 * handled.
 *
```

```
* COPYRIGHT:  
*  
* Copyright 2018 Amazon.com, Inc. or its affiliates. All Rights Reserved.  
*  
* Licensed under the Apache License, Version 2.0 (the "License").  
* You may not use this file except in compliance with the License.  
* A copy of the License is located at  
*  
*     http://www.apache.org/licenses/LICENSE-2.0  
*  
* or in the "license" file accompanying this file. This file 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.  
*/  
public class StreamTranscriptionBehaviorImpl implements StreamTranscriptionBehavior {  
  
    @Override  
    public void onError(Throwable e) {  
        System.out.println("== Failure Encountered ==");  
        e.printStackTrace();  
    }  
  
    @Override  
    public void onStream(TranscriptResultStream e) {  
        // EventResultStream has other fields related to the timestamp of the  
        transcripts in it.  
        // Please refer to the javadoc of TranscriptResultStream for more details  
        List<Result> results = ((TranscriptEvent) e).transcript().results();  
        if (results.size() > 0) {  
            if (results.get(0).alternatives().size() > 0)  
                if (!results.get(0).alternatives().get(0).transcript().isEmpty()) {  
                    System.out.println(results.get(0).alternatives().get(0).transcript());  
                }  
        }  
    }  
  
    @Override  
    public void onResponse(StartStreamTranscriptionResponse r) {  
  
        System.out.println(String.format("== Received Initial response. Request Id: %s  
===", r.requestId()));  
    }  
  
    @Override  
    public void onComplete() {  
        System.out.println("== All records stream successfully ==");  
    }  
}
```

Next step

[Using the HTTP/2 retry client \(p. 213\)](#)

Using the HTTP/2 retry client

The following is a sample application that uses the retry client to transcribe audio from either a file or your microphone. You can use this application to test the client, or you can use it as a starting point for your own applications.

To run the sample, do the following:

- Copy the retry client to your workspace. See [Streaming retry client code \(p. 209\)](#).

- Copy the retry client interface to your workspace. Implement the interface, or you can use the sample implementation. See [Streaming retry client interface code \(p. 212\)](#).
- Copy the sample application to your workspace. Build and run the application.

```

public class StreamingRetryApp {
    private static final String endpoint = "endpoint";
    private static final Region region = Region.US_EAST_1;
    private static final int sample_rate = 28800;
    private static final String encoding = " ";
    private static final String language = LanguageCode.EN_US.toString();

    public static void main(String args[]) throws URISyntaxException, ExecutionException,
    InterruptedException, LineUnavailableException, FileNotFoundException {
        /**
         * Create Amazon Transcribe streaming retry client.
         */

        TranscribeStreamingRetryClient client = new
        TranscribeStreamingRetryClient(EnvironmentVariableCredentialsProvider.create() ,endpoint,
        region);

        StartStreamTranscriptionRequest request = StartStreamTranscriptionRequest.builder()
            .languageCode(language)
            .mediaEncoding(encoding)
            .mediaSampleRateHertz(sample_rate)
            .build();

        /**
         * Start real-time speech recognition. The Amazon Transcribe streaming java client
         uses the Reactive-streams
         * interface. For reference on Reactive-streams:
         * https://github.com/reactive-streams/reactive-streams-jvm
         */
        CompletableFuture<Void> result = client.startStreamTranscription(
            /**
             * Request parameters. Refer to API documentation for details.
             */
            request,
            /**
             * Provide an input audio stream.
             * For input from a microphone, use getStreamFromMic().
             * For input from a file, use getStreamFromFile().
             */
            new AudioStreamPublisher(
                new FileInputStream(new File("FileName"))),
            /**
             * Object that defines the behavior on how to handle the stream
             */
            new StreamTranscriptionBehaviorImpl());

        /**
         * Synchronous wait for stream to close, and close client connection
         */
        result.get();
        client.close();
    }

    private static class AudioStreamPublisher implements Publisher<AudioStream> {
        private final InputStream inputStream;
        private static Subscription currentSubscription;

        private AudioStreamPublisher(InputStream inputStream) {
            this.inputStream = inputStream;
        }
    }
}

```

```
    @Override
    public void subscribe(Subscriber<? super AudioStream> s) {
        if (this.currentSubscription == null) {
            this.currentSubscription = new
TranscribeStreamingDemoApp.SubscriptionImpl(s, inputStream);
        } else {
            this.currentSubscription.cancel();
            this.currentSubscription = new
TranscribeStreamingDemoApp.SubscriptionImpl(s, inputStream);
        }
        s.onSubscribe(currentSubscription);
    }
}
```

Establish a bi-directional connection using the WebSocket protocol

You can use the [WebSocket protocol](#) to open a bi-directional connection that is designed to be secure with Amazon Transcribe Medical. You encode the audio stream with event stream encoding, and Amazon Transcribe Medical returns a stream of text in JSON blobs that are also encoded using event stream encoding. For more information, see [Event stream encoding \(p. 201\)](#). You can use the information in this section to create applications using the WebSocket library of your choice.

Topics

- [Adding a policy for WebSocket requests to your IAM role \(p. 215\)](#)
- [Creating a pre-signed URL \(p. 215\)](#)
- [Handling the WebSocket upgrade response \(p. 220\)](#)
- [Making a WebSocket streaming request \(p. 220\)](#)
- [Handling a WebSocket streaming response \(p. 221\)](#)
- [Handling WebSocket streaming errors \(p. 221\)](#)

Adding a policy for WebSocket requests to your IAM role

To use the WebSocket protocol to call Amazon Transcribe Medical, you need to attach the following policy to the AWS Identity and Access Management (IAM) role that makes the request.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "transcribemedicalstreaming",
            "Effect": "Allow",
            "Action": "transcribe:StartMedicalStreamTranscription",
            "Resource": "*"
        }
    ]
}
```

Creating a pre-signed URL

You need to construct a URL for your WebSocket request that contains the information needed to set up communication between your application and Amazon Transcribe Medical. To open a bi-directional

connection, you must use port 8443. WebSocket streaming-med uses the Amazon Signature Version 4 process for signing requests. Signing the request helps to verify the identity of the requester and to protect your audio data in transit. It also protects against potential replay attacks. For more information about Signature Version 4, see [Signing AWS API Requests](#) in the *Amazon Web Services General Reference*.

The URL has the following format. Line breaks have been added for readability.

```
GET https://transcribestreaming.region.amazonaws.com:8443/medical-stream-transcription-
websocket
?language-code=languageCode
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=Signature Version 4 credential scope
&X-Amz-Date=date
&X-Amz-Expires=time in seconds until expiration
&X-Amz-Security-Token=security-token
&X-Amz-Signature=Signature Version 4 signature
&X-Amz-SignedHeaders=host
&media-encoding=mediaEncoding
&sample-rate=mediaSampleRateHertz
&session-id=sessionId
&specialty=specialty
&type=type
```

For TYPE, it is best to select CONVERSATION if your use case involves multiple speakers engaged in a discussion. An example would be a conversation between a clinician and a patient. Select DICTATION if your use case involves a single speaker where a person is dictating speech. An example would be if a physician is dictating medical notes for data entry purposes after a patient encounter.

Use the following values for the URL parameters:

- **language-code** – The language code for the input audio. The valid value is en-US.
- **media-encoding** – The encoding used for the input audio. The valid value is pcm.
- **sample-rate** – The sample rate of the input audio in hertz. 16,000 Hz or higher sample rates are accepted.
- **sessionId** – Optional. An identifier for the transcription session. If you don't provide a session ID, Amazon Transcribe Medical generates one for you and returns it in the response.
- **specialty** – The specialty of the medical domain. Valid values are PRIMARYCARE, CARDIOLOGY, NEUROLOGY, ONCOLOGY, RADIOLOGY, and UROLOGY.
- **type** – The type of audio. Must be DICTATION or CONVERSATION.

The remaining parameters are Signature Version 4 parameters:

- **X-Amz-Algorithm** – The algorithm you're using in the signing process. The only valid value is AWS4-HMAC-SHA256.
- **X-Amz-Credential** – A string separated by slashes ("/") that is formed by concatenating your access key ID and your credential scope components. Credential scope includes the date in YYYYMMDD format, the AWS Region, the service name, and a special termination string (aws4_request).
- **X-Amz-Date** – The date and time that the signature was created. Generate the date and time by following the instructions in [Handling Dates in Signature Version 4](#) in the *Amazon Web Services General Reference*.
- **X-Amz-Expires** – The length of time in seconds until the credentials expire. The maximum value is 300 seconds (5 minutes).
- **X-Amz-Security-Token** – Optional. A Signature Version 4 token for temporary credentials. If you specify this parameter, include it in the canonical request. For more information, see [Requesting Temporary Security Credentials](#) in the *AWS AWS Identity and Access Management User Guide*.

- **X-Amz-Signature** – The Signature Version 4 signature that you generated for the request.
- **X-Amz-SignedHeaders** – The headers that are signed when creating the signature for the request. The only valid value is host.

To construct the URL for the request and create the Signature Version 4 signature, use the following steps. The examples are in pseudocode.

Task 1: Create a canonical request

Create a string that includes information from your request in a standardized format. This ensures that when AWS receives the request, it can calculate the same signature that you calculate in Task 3. For more information, see [Create a Canonical Request for Signature Version 4](#) in the *Amazon Web Services General Reference*.

1. Define variables for the request in your application.

```
# HTTP verb
method = "GET"
# Service name
service = "transcribe"
# AWS Region
region = "AWS Region"
# Amazon Transcribe streaming-med endpoint
endpoint = "https://transcribestreaming.region.amazonaws.com:8443"
# Host
host = "transcribestreaming.region.amazonaws.com:8443"
# Date and time of request
amz-date = YYMMDD'T'HHMMSS'Z'
# Date without time for credential scope
datestamp = YYYYMMDD
```

2. Create a canonical URI. The canonical URI is the part of the URI between the domain and the query string.

```
canonical_uri = "/medical-stream-transcription-websocket"
```

3. Create the canonical headers and signed headers. Note the trailing "\n" in the canonical headers.

- Append the lowercase header name followed by a colon.
- Append a comma-separated list of values for that header. Do not sort the values in headers that have multiple values.
- Append a new line (\n).

```
canonical_headers = "host:" + host + "\n"
signed_headers = "host"
```

4. Match the algorithm to the hashing algorithm. You must use SHA-256.

```
algorithm = "AWS4-HMAC-SHA256"
```

5. Create the credential scope, which scopes the derived key to the date, AWS Region, and service to which the request is made.

```
credential_scope = datestamp + "/" + region + "/" + service + "/" + "aws4_request"
```

6. Create the canonical query string. Query string values must be URI-encoded and sorted by name.
 - Sort the parameter names by character code point in ascending order. Parameters with duplicate names should be sorted by value. For example, a parameter name that begins with the uppercase letter F precedes a parameter name that begins with a lowercase letter b.
 - Do not URI-encode any of the unreserved characters that [RFC 3986](#) defines: A-Z, a-z, 0-9, hyphen (-), underscore (_), period (.), and tilde (~).
 - Percent-encode all other characters with %XY, where X and Y are hexadecimal characters (0-9 and uppercase A-F). For example, the space character must be encoded as %20 (not using '+', as some encoding schemes do) and extended UTF-8 characters must be in the form %XY%ZA%BC.
 - Double-encode any equals (=) characters in parameter values.

```
canonical_querystring = "X-Amz-Algorithm=" + algorithm
canonical_querystring += "&X-Amz-Credential=" + URI-encode(access key + "/" +
    credential_scope)
canonical_querystring += "&X-Amz-Date=" + amz_date
canonical_querystring += "&X-Amz-Expires=300"
canonical_querystring += "&X-Amz-Security-Token=" + token
canonical_querystring += "&X-Amz-SignedHeaders=" + signed_headers
canonical_querystring += "&language-code=en-US&media-encoding=pcm&sample-rate=16000"
canonical_querystring += "&specialty=PRIMARYCARE"
canonical_querystring += "&type=DICTATION"
```

7. Create a hash of the payload. For a GET request, the payload is an empty string.

```
payload_hash = HashSHA256("").Encode("utf-8").HexDigest()
```

8. Combine all of the elements to create the canonical request.

```
canonical_request = method + '\n'
    + canonical_uri + '\n'
    + canonical_querystring + '\n'
    + canonical_headers + '\n'
    + signed_headers + '\n'
    + payload_hash
```

Task 2: Create the string to sign

The string to sign contains meta information about your request. You use the string to sign in the next step when you calculate the request signature. For more information, see [Create a String to Sign for Signature Version 4](#) in the *Amazon Web Services General Reference*.

- Create the string.

```
string_to_sign=algorithm + "\n"
    + amz_date + "\n"
    + credential_scope + "\n"
    + HashSHA256(canonical_request.Encode("utf-8").HexDigest())
```

Task 3: Calculate the signature

You derive a signing key from your AWS secret access key. For a greater degree of protection, the derived key is specific to the date, service, and AWS Region. You use the derived key to sign the request. For

more information, see [Calculate the Signature for AWS Signature Version 4](#) in the *Amazon Web Services General Reference*.

The code assumes that you have implemented the `GetSignatureKey` function to derive a signing key. For more information and example functions, see [Examples of How to Derive a Signing Key for Signature Version 4](#) in the *Amazon Web Services General Reference*.

The function `HMAC(key, data)` represents an HMAC-SHA256 function that returns the results in binary format.

- Create the signing key and sign the string to sign.

```
#Create the signing key
signing_key = GetSignatureKey(secret_key, datestamp, region, service)

# Sign the string_to_sign using the signing key
signature = HMAC.new(signing_key, (string_to_sign).Encode("utf-8"), Sha256()).HexDigest
```

Task 4: Add signing information to the request and create the request URL

After you calculate the signature, add it to the query string. For more information, see [Add the Signature to the HTTP Request](#) in the *Amazon Web Services General Reference*.

1. Add the authentication information to the query string.

```
canonical_querystring += "&X-Amz-Signature=" + signature
```

2. Create the URL for the request.

```
request_url = endpoint + canonical_uri + "?" + canonical_querystring
```

You use the request URL with your WebSocket library to make the request to the Amazon Transcribe Medical service.

Including WebSocket request headers

The request to Amazon Transcribe Medical must include the following headers. Typically these med headers are managed by your WebSocket client library.

```
Host: transcribestreaming.region.amazonaws.com:8443
Connection: Upgrade
Upgrade: websocket
Origin: request source
Sec-WebSocket-Version: 13
Sec-WebSocket-Key: random key
```

Use the following values for the headers:

- **Connection** – Always `Upgrade`.
- **Upgrade** – Always `websocket`.
- **Origin** – The URI of the WebSocket client.
- **Sec-WebSocket-Version** – The version of the WebSocket protocol to use.
- **Sec-WebSocket-Key** – A base-64 encoded randomly generated string that identifies the request.

Handling the WebSocket upgrade response

When Amazon Transcribe Medical receives your WebSocket request, it responds with a WebSocket upgrade response. Typically, your WebSocket library manages this response and sets up a socket for communications with Amazon Transcribe Medical.

The following is the response from Amazon Transcribe Medical. Line breaks have been added to the `websocket-location` header for readability.

```
HTTP/1.1 101 WebSocket Protocol Handshake

Connection: upgrade
Upgrade: websocket
websocket-origin: https://transcribestreaming.region.amazonaws.com:8443
websocket-location: transcribestreaming.region.amazonaws.com:8443/stream-transcription-
websocket?
    X-Amz-Algorithm=AWS4-HMAC-SHA256
    &X-Amz-Credential=AKIDEXAMPLE%2F20190117%2Fregion%2Ftranscribe
%2Faws4_request
    &X-Amz-Date=date and time
    &X-Amz-Expires=expiration length
    &X-Amz-SignedHeaders=host
    &language-code=language code
    &media-encoding=media encoding
    &sample-rate=media sample rate
    &type=dictation or conversation
    &specialty=medical specialty (must be Primary Care)
    &X-Amz-Signature=signature
x-amzn-RequestId: RequestId
x-amzn-SessionId: SessionId
Strict-Transport-Security: max-age=31536000
sec-websocket-accept: token
```

The response has the following values:

- **Connection** – Always Upgrade.
- **Upgrade** – Always websocket.
- **websocket-origin** – The URI of the WebSocket server that responded to the request.
- **websocket-location** – The contents of the request URI that was sent to the server. For a description of the contents, see [Creating a pre-signed URL \(p. 77\)](#).
- **x-amzn-RequestId** – An identifier for the request.
- **x-amzn-SessionId** – An identifier for a transcription session.
- **Strict-Transport-Security** – A header that informs browsers to access the endpoint using only HTTPS.
- **sec-websocket-accept** – The hash of the Sec-WebSocket-Key header sent in the request.

Making a WebSocket streaming request

After the WebSocket connection is established, the client can start sending a sequence of audio frames. Each frame contains one data frame that is encoded in event stream encoding. For more information, see [Event stream encoding \(p. 74\)](#).

Each data frame contains three headers combined with a chunk of raw audio bytes. The following table lists and describes the headers.

Header name Byte Length	Header name (string)	Header value type	Value string byte length	Value string (UTF-8)
13	:content-type	7	24	application/octet-stream
11	:event-type	7	10	AudioEvent
13	:message-type	7	5	event

To end the audio data stream, send an empty audio chunk in an event-stream-encoded message.

Handling a WebSocket streaming response

The response contains event-stream-encoded raw bytes in the payload. It contains the standard prelude and the following headers.

Header name byte length	Header name (string)	Header value type	Value string byte length	Value string (UTF-8)
13	:content-type	7	24	application/octet-stream
11	:event-type	7	15	TranscriptEvent
13	:message-type	7	5	event

When you decode the binary response, you end up with a JSON structure with the results of the transcription. For an example of the JSON response, see [Streaming transcription \(p. 196\)](#).

Handling WebSocket streaming errors

If an exception occurs while processing your request, Amazon Transcribe Medical responds with a terminal WebSocket frame containing an event-stream-encoded response. The response has the headers described in the following table, and the body of the response contains a descriptive error message. After sending the exception response, Amazon Transcribe Medical sends a close frame.

Header name byte length	Header name (string)	Header value type	Value string byte length	Value string (UTF-8)
13	:content-type	7	24	application/octet-stream
15	:exception-type	7	varies	varies, see below
13	:message-type	7	9	exception

The `:exception-type` header contains one of the following values:

- **BadRequestException** – There was a client error when the stream was created, or an error occurred while streaming-med data. Make sure that your client is ready to accept data and try your request again.

- **InternalFailureException** – Amazon Transcribe Medical had a problem during the handshake with the client. Try your request again.
- **LimitExceededException** – The client exceeded the concurrent stream limit. For more information, see [Amazon Transcribe Service Quotas](#) in the *Amazon Web Services General Reference*. Reduce the number of streams that you are transcribing.
- **UnrecognizedClientException** – The WebSocket upgrade request was signed with an incorrect access key or secret key. Make sure that you are correctly creating the access key and try your request again.

In addition, Transcribe Medical can return any of the common service errors. For a list, see [Common Errors](#).

Transcribing a medical conversation

You can use Amazon Transcribe Medical to transcribe a medical conversation between a clinician and a patient using either a batch transcription job or a real-time stream. Batch transcription jobs enable you to transcribe audio files. To ensure that Amazon Transcribe Medical produces transcription results with the highest possible accuracy, you must specify the medical specialty of the clinician in your transcription job or stream.

You can transcribe a clinician-patient visit in the following medical specialities:

- Cardiology – available in streaming transcription only
- Neurology – available in streaming transcription only
- Oncology – available in streaming transcription only
- Primary Care – includes the following types of medical practice:
 - Family medicine
 - Internal medicine
 - Obstetrics and Gynecology (OB-GYN)
 - Pediatrics
- Urology – available in streaming transcription only

You can improve transcription accuracy by using medical custom vocabularies. For information on how medical custom vocabularies work, see [Improving transcription accuracy with medical custom vocabularies \(p. 247\)](#).

By default, Amazon Transcribe Medical returns the transcription with the highest confidence level. If you'd like to configure it to return alternative transcriptions, see [Generating alternative transcriptions \(p. 262\)](#).

For information about how numbers and medical measurements appear in the transcription output, see [Transcribing numbers \(p. 192\)](#) and [Transcribing medical terms and measurements \(p. 193\)](#).

Topics

- [Transcribing an audio file of a medical conversation \(p. 223\)](#)
- [Transcribing a medical conversation in a real-time stream \(p. 226\)](#)
- [Identifying speakers and labeling their speech \(p. 228\)](#)
- [Transcribing multi-channel audio \(p. 235\)](#)

Transcribing an audio file of a medical conversation

Use a batch transcription job to transcribe audio files of medical conversations. You can use this to transcribe a clinician-patient dialogue. You can start a batch transcription job in either the [StartMedicalTranscriptionJob \(p. 370\)](#) API or the Amazon Transcribe Medical console.

When you start a medical transcription job with the [StartMedicalTranscriptionJob \(p. 370\)](#) API, you specify PRIMARYCARE as the value of the **Specialty** parameter.

Console

To transcribe a clinician-patient dialogue (console)

To use the console to transcribe a clinician-patient dialogue, create a transcription job and choose **Conversation** for **Audio input type**.

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, under Amazon Transcribe Medical, choose **Transcription jobs**.
3. Choose **Create job**.
4. On the **Specify job details** page, under **Job settings**, specify the following.
 - a. **Name** – the name of the transcription job.
 - b. **Audio input type – Conversation**
5. For the remaining fields, specify the Amazon Simple Storage Service (Amazon S3) location of your audio file and where you want to store the output of your transcription job.
6. Choose **Next**.
7. Choose **Create**.

API

To transcribe a medical conversation using a batch transcription job (API)

- For the [StartMedicalTranscriptionJob \(p. 370\)](#) API, specify the following.
 - a. For **MedicalTranscriptionJobName**, specify a name unique in your AWS account.
 - b. For **LanguageCode**, specify the language code that corresponds to the language spoken in your audio file and the language of your vocabulary filter.
 - c. For the **MediaFileUri** parameter of the **Media** object, specify the name of the audio file that you want to transcribe.
 - d. For **Specialty**, specify the medical specialty of the clinician speaking in the audio file as PRIMARYCARE.
 - e. For **Type**, specify CONVERSATION.
 - f. For **OutputBucketName**, specify the Amazon Simple Storage Service (Amazon S3) bucket to store the transcription results.

The following is an example request that uses the AWS SDK for Python (Boto3) to transcribe a medical conversation of a clinician in the PRIMARYCARE specialty and a patient.

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
```

```

job_name = "medical-conversation-transcription-job-name"
job_uri = "s3://DOC-EXAMPLE-BUCKET1/example-audio-file.extension"
transcribe.start_medical_transcription_job(
    MedicalTranscriptionJobName = job_name,
    Media = {'MediaFileUri': job_uri},
    LanguageCode = 'en-US',
    Specialty = 'PRIMARYCARE',
    Type = 'CONVERSATION',
    OutputBucketName = 'DOC-EXAMPLE-BUCKET2'
)
while True:
    status =
    transcribe.get_medical_transcription_job(MedicalTranscriptionJobName=job_name)
    if status['MedicalTranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED',
    'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)

```

The following example code shows the transcription results of a clinician-patient conversation.

```
{
    "jobName": "conversation-medical-transcription-job",
    "accountId": "453794026688",
    "results": {
        "transcripts": [
            {
                "transcript": "... come for a follow up visit today..."
            }
        ],
        "items": [
            {
                ...
                "start_time": "4.85",
                "end_time": "5.12",
                "alternatives": [
                    {
                        "confidence": "1.0",
                        "content": "come"
                    }
                ],
                "type": "pronunciation"
            },
            {
                "start_time": "5.12",
                "end_time": "5.29",
                "alternatives": [
                    {
                        "confidence": "1.0",
                        "content": "for"
                    }
                ],
                "type": "pronunciation"
            },
            {
                "start_time": "5.29",
                "end_time": "5.33",
                "alternatives": [
                    {
                        "confidence": "0.9955",

```

```
        "content": "a"
    },
],
"type": "pronunciation"
},
{
"start_time": "5.33",
"end_time": "5.66",
"alternatives": [
{
"confidence": "0.9754",
"content": "follow"
}
],
"type": "pronunciation"
},
{
"start_time": "5.66",
"end_time": "5.75",
"alternatives": [
{
"confidence": "0.9754",
"content": "up"
}
],
"type": "pronunciation"
},
{
"start_time": "5.75",
"end_time": "6.02",
"alternatives": [
{
"confidence": "1.0",
"content": "visit"
}
]
...
},
"status": "COMPLETED"
}
```

AWS CLI

To transcribe a medical conversation using a batch transcription job (AWS CLI)

- Run the following code.

```
aws transcribe start-medical-transcription-job \
--cli-input-json file://example-start-command.json
```

The following code shows the contents of `example-start-command.json`.

```
{
    "MedicalTranscriptionJobName": "conversation-medical-transcription-job",
    "LanguageCode": "en-US",
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION",
```

```
        "OutputBucketName": "the-S3-bucket-where-you-output-the-transcription-results",
        "Media": {
            "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET1/example-audio-file.extension"
        }
    }
```

The following is the response from running the preceding CLI command.

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "example-conversation-medical-transcription-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "en-US",
        "Media": {
            "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET1/example-audio-file.extension"
        },
        "StartTime": "2020-10-05T20:43:39.583000+00:00",
        "CreationTime": "2020-10-05T20:43:39.547000+00:00",
        "Specialty": "PRIMARYCARE",
        "Type": "CONVERSATION"
    }
}
```

Transcribing a medical conversation in a real-time stream

You can transcribe an audio stream of a medical conversation using either the HTTP/2 or [WebSocket](#) protocols. For information on how to start a stream using the WebSocket protocol, see [Establish a bi-directional connection using the WebSocket protocol \(p. 215\)](#). To start an HTTP/2 stream, use the [StartMedicalStreamTranscription \(p. 402\)](#) API.

You can transcribe streaming audio in the following medical specialties:

- Cardiology
- Neurology
- Oncology
- Primary Care
- Urology

Each medical specialty includes many types of procedures and appointments. Clinicians therefore dictate many different types of notes. Use the following examples as guidance to help you specify the value of the `specialty` URL parameter of the WebSocket request, or the `Specialty` parameter of the [StartMedicalStreamTranscription \(p. 402\)](#) API:

- For electrophysiology or echocardiography consultations, choose `CARDIOLOGY`.
- For medical oncology, surgical oncology, or radiation oncology consultations, choose `ONCOLOGY`.

- For a physician providing a consultation to a patient who had a stroke, either a transient ischemic attack or a cerebrovascular attack, choose **NEUROLOGY**.
- For a consultation around urinary incontinence, choose **UROLOGY**.
- For yearly checkup or urgent care visits, choose **PRIMARYCARE**.
- For inpatient hospitalist visits, choose **PRIMARYCARE**.
- For consultations regarding fertility, tubal ligation, IUD insertion, or abortion, choose **PRIMARYCARE**.

Console

To transcribe a streaming medical conversation (console)

To use the console to transcribe a clinician-patient dialogue in real-time stream, choose the option to transcribe a medical conversation, start the stream, and begin speaking into the microphone.

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, under Amazon Transcribe Medical, choose **Real-time transcription**.
3. Choose **Conversation**.
4. For **Medical specialty**, choose the clinician's specialty.
5. Choose **Start streaming**.
6. Speak into the microphone.

Transcribing a medical conversation in an HTTP/2 stream

The following is the syntax for the parameters of an HTTP/2 request.

To transcribe an HTTP/2 stream of a medical conversation, use the [StartMedicalStreamTranscription \(p. 402\)](#) API and specify the following:

- **LanguageCode** – The language code. The valid value is `en-US`
- **MediaEncoding** – The encoding used for the input audio. Valid values are `pcm`, `ogg-opus`, and `flac`.
- **Specialty** – The specialty of the medical professional.
- **Type** – `CONVERSATION`

To improve transcription accuracy of specific terms in a real-time stream, use a custom vocabulary. To enable a custom vocabulary, set the value of **VocabularyName** parameter to the name of the custom vocabulary that you want to use. For more information, see [Improving transcription accuracy with medical custom vocabularies \(p. 247\)](#).

To label the speech from different speakers, set the **ShowSpeakerLabel** parameter to `true`. For more information, see [Identifying speakers and labeling their speech \(p. 228\)](#).

For more information on setting up an HTTP/2 stream to transcribe a medical conversation, see [Streaming request \(p. 203\)](#).

Transcribing a medical conversation in a WebSocket stream

You can use a WebSocket request to transcribe a medical conversation. When you make a WebSocket request, you create a pre-signed URL. This URL contains the information needed to set up the audio stream between your application and Amazon Transcribe Medical. For more information on creating WebSocket requests, see [Establish a bi-directional connection using the WebSocket protocol \(p. 215\)](#).

Use the following template to create your pre-signed URL.

```
GET https://transcribestreaming.region.amazonaws.com:8443/medical-stream-transcription-  
websocket  
?language-code=languageCode  
&X-Amz-Algorithm=AWS4-HMAC-SHA256  
&X-Amz-Credential=Signature Version 4 credential scope  
&X-Amz-Date=date  
&X-Amz-Expires=time in seconds until expiration  
&X-Amz-Security-Token=security-token  
&X-Amz-Signature=Signature Version 4 signature  
&X-Amz-SignedHeaders=host  
&media-encoding=mediaEncoding  
&sample-rate=mediaSampleRateHertz  
&session-id=sessionId  
&specialty=medicalSpecialty  
&type=CONVERSATION  
&vocabulary-name=vocabularyName  
&show-speaker-label=boolean
```

To improve transcription accuracy of specific terms in a real-time stream, use a custom vocabulary. To enable a custom vocabulary, set the value of `vocabulary-name` to the name of the custom vocabulary that you want to use. For more information, see [Improving transcription accuracy with medical custom vocabularies \(p. 247\)](#).

To label the speech from different speakers, set the `show-speaker-label` parameter in to `true`. For more information, see [Identifying speakers and labeling their speech \(p. 228\)](#).

For more information on creating pre-signed URLs, see [Creating a pre-signed URL \(p. 215\)](#).

Identifying speakers and labeling their speech

To identify and label the speech from different speakers in Amazon Transcribe Medical, use *speaker diarization*. This enables you to see what the patient said and what the clinician said in the transcription output.

When you enable speaker diarization, Amazon Transcribe Medical labels each speaker *utterance* with a unique identifier for each speaker. An *utterance* is a unit of speech that is typically separated from other utterances by silence. In batch transcription, an utterance from the clinician could receive a label of `spk_0` and an utterance the patient could receive a label of `spk_1`.

If an utterance from one speaker overlaps with an utterance from another speaker, Amazon Transcribe Medical orders them in the transcription by their start times. Utterances that overlap in the input audio don't overlap in the transcription output.

You can enable speaker diarization when you transcribe an audio file using batch transcription job, or in a real-time stream.

Topics

- [Identifying speakers and labeling their speech in audio files \(p. 228\)](#)
- [Identifying speakers and labeling their speech in real-time streams \(p. 232\)](#)

Identifying speakers and labeling their speech in audio files

You can enable speaker identification in a batch transcription job using either the [StartMedicalTranscriptionJob \(p. 370\)](#) API or the Amazon Transcribe Medical console. This enables you to identify the speakers in a clinician-patient conversation and determine who said what in the transcription output.

Console

To enable speaker diarization in a transcription job (console)

To use the console to enable speaker diarization in your transcription job, you enable audio identification and then speaker identification.

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, under Amazon Transcribe Medical, choose **Transcription jobs**.
3. Choose **Create job**.
4. On the **Specify job details** page, provide information about your transcription job.
5. Choose **Next**.
6. Enable **Audio identification**.
7. For **Audio identification type**, choose **Speaker identification**.
8. For **Maximum number of speakers**, enter the maximum number of speakers that you think are speaking in your audio file. For best results, match the number of speakers that you ask to identify to the number of speakers in the input audio. If you specify a value less than the number of speakers in your input audio, the transcription text of the most similar sounding speakers are attributed to a speaker label.
9. Choose **Create**.

API

To identify speakers in an audio file using a batch transcription job (API)

- For the [StartMedicalTranscriptionJob \(p. 370\)](#) API, specify the following.
 - a. For `MedicalTranscriptionJobName`, specify a name that is unique in your AWS account.
 - b. For `LanguageCode`, specify the language code that corresponds to the language spoken in the audio file.
 - c. For the `MediaFileUri` parameter of the `Media` object, specify the name of the audio file that you want to transcribe.
 - d. For `Specialty`, specify the medical specialty of the clinician speaking in the audio file.
 - e. For `Type`, specify `CONVERSATION`.
 - f. For `OutputBucketName`, specify the Amazon Simple Storage Service (Amazon S3) bucket to store the transcription results.
 - g. For the `Settings` object, specify the following.
 - i. `ShowSpeakerLabels` – `true`.
 - ii. `MaxSpeakerLabels` – An integer between 2 and 10 to indicate the number of speakers that you think are speaking in your audio. For best results, match the number of speakers that you ask to identify to the number of speakers in the input audio. If you specify a value less than the number of speakers in your input audio, the transcription text of the most similar sounding speakers are attributed to the same speaker label.

The following request uses the AWS SDK for Python (Boto3) to start a batch transcription job of a primary care clinician patient dialogue with speaker identification enabled.

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
```

```

job_name = "example-diarization-transcription"
job_uri = "https://DOC-EXAMPLE-BUCKET1.s3-Region.amazonaws.com/example-file.mp4"
transcribe.start_medical_transcription_job(
    MedicalTranscriptionJobName=job_name,
    Media = {'MediaFileUri': job_uri},
    LanguageCode = 'en-US',
    Specialty = 'PRIMARYCARE',
    Type = 'CONVERSATION',
    OutputBucketName = 'DOC-EXAMPLE-BUCKET2',
    Settings = {'ShowSpeakerLabels': True,
                'MaxSpeakerLabels': 2
               }
)
while True:
    status = transcribe.get_medical_transcription_job(MedicalTranscriptionJobName=job_name)
    if status['MedicalTranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED',
    'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)

```

The following example code shows the transcription results of a transcription job with speaker identification enabled.

```

{
    "jobName": "job ID",
    "accountId": "account ID",
    "results": {
        "transcripts": [
            {
                "transcript": "Professional answer."
            }
        ],
        "speaker_labels": {
            "speakers": 1,
            "segments": [
                {
                    "start_time": "0.000000",
                    "speaker_label": "spk_0",
                    "end_time": "1.430",
                    "items": [
                        {
                            "start_time": "0.100",
                            "speaker_label": "spk_0",
                            "end_time": "0.690"
                        },
                        {
                            "start_time": "0.690",
                            "speaker_label": "spk_0",
                            "end_time": "1.210"
                        }
                    ]
                }
            ],
            "items": [
                {
                    "start_time": "0.100",
                    "end_time": "0.690",
                    "alternatives": [

```

```
{  
    "confidence": "0.8162",  
    "content": "Professional"  
}  
,  
    "type": "pronunciation"  
},  
{  
    "start_time": "0.690",  
    "end_time": "1.210",  
    "alternatives": [  
        {  
            "confidence": "0.9939",  
            "content": "answer"  
        }  
,  
        "type": "pronunciation"  
],  
{  
    "alternatives": [  
        {  
            "content": "."  
        }  
,  
        "type": "punctuation"  
    ]  
},  
"status": "COMPLETED"  
}
```

AWS CLI

To transcribe an audio file of a conversation between a clinician practicing primary care and a patient in an audio file and identify what each person said in the transcription output (AWS CLI)

- Run the following code.

```
aws transcribe start-transcription-job \  
--cli-input-json file:///example-start-command.json
```

The following code shows the contents of *example-start-command.json*.

```
{  
    "MedicalTranscriptionJobName": "speaker-id-conversation-medical-transcription-job",  
    "LanguageCode": "language-code",  
    "Specialty": "PRIMARYCARE",  
    "Type": "CONVERSATION",  
    "OutputBucketName": "DOC-EXAMPLE-BUCKET",  
    "Media": {  
        "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"  
    },  
    "Settings": {  
        "ShowSpeakerLabels": true,  
        "MaxSpeakerLabels": 2
```

```
    }  
}
```

The following is the response from running the preceding CLI command.

```
{  
    "MedicalTranscriptionJobName": "speaker-id-conversation-medical-transcription-job",  
    "LanguageCode": "en-US",  
    "Specialty": "PRIMARYCARE",  
    "Type": "CONVERSATION",  
    "OutputBucketName": "DOC-EXAMPLE-BUCKET1",  
    "Media": {  
        "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET1/example-file.extension"  
    },  
    "Settings": {  
        "ShowSpeakerLabels": true,  
        "MaxSpeakerLabels": 2  
    }  
}
```

Identifying speakers and labeling their speech in real-time streams

To identify speakers and label their speech in a real-time stream, use the Amazon Transcribe Medical console or a streaming request. Speaker identification works best for identifying between two and five speakers in a stream. Although Amazon Transcribe Medical can identify more than five speakers in a stream, the accuracy of speaker identification decreases if you exceed that number.

To start an HTTP/2 request, use the [StartMedicalStreamTranscription \(p. 402\)](#) API. To start a WebSocket request, use a pre-signed URL. The URL contains the information required to set up bi-directional communication between your application and Amazon Transcribe Medical.

Identifying speakers in audio that is spoken into your microphone (console)

To identify speakers in audio that is spoken into your microphone (console)

You can use the Amazon Transcribe Medical console to start a real-time stream of a clinician-patient conversation, or a dictation that is spoken into your microphone in real-time.

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, for Amazon Transcribe Medical choose **Real-time transcription**.
3. For **Audio input type**, choose the type of medical speech that you want to transcribe.
4. For **Additional settings**, choose **Speaker identification**.
5. Choose **Start streaming** to start transcribing your real-time audio.
6. Speak into the microphone.

Identifying speakers in an HTTP/2 stream

To identify speakers in an HTTP/2 stream of a medical conversation, use the [StartMedicalStreamTranscription \(p. 402\)](#) API and specify the following:

- For `LanguageCode`, specify the language code that corresponds to the language in the stream. The valid value is `en-US`.
- For `MediaSampleHertz`, specify the sample rate of the audio.
- For `Specialty`, specify the medical specialty of the provider.
- `ShowSpeakerLabel = true`

For more information on setting up an HTTP/2 stream to transcribe a medical conversation, see [Streaming request \(p. 203\)](#).

Identifying speakers in a WebSocket request

To identify speakers in WebSocket streams with the API, use the following format to create a pre-signed URL to start a WebSocket request and set `show-speaker-label` to `true`.

```
GET https://transcribestreaming.region.amazonaws.com:8443/medical-stream-transcription-  
websocket  
?language-code=languageCode  
&X-Amz-Algorithm=AWS4-HMAC-SHA256  
&X-Amz-Credential=Signature Version 4 credential scope  
&X-Amz-Date=date  
&X-Amz-Expires=time in seconds until expiration  
&X-Amz-Security-Token=security-token  
&X-Amz-Signature=Signature Version 4 signature  
&X-Amz-SignedHeaders=host  
&media-encoding=mediaEncoding  
&sample-rate=mediaSampleRateHertz  
&session-id=sessionId  
&specialty=medicalSpecialty  
&type=CONVERSATION  
&vocabulary-name=vocabularyName  
&show-speaker-label=boolean
```

The following code shows the truncated example response of a streaming request.

```
{  
    "Transcript": {  
        "Results": [  
            {  
                "Alternatives": [  
                    {  
                        "Items": [  
                            {  
                                "Confidence": 0.97,  
                                "Content": "From",  
                                "EndTime": 18.98,  
                                "Speaker": "0",  
                                "StartTime": 18.74,  
                                "Type": "pronunciation",  
                                "VocabularyFilterMatch": false  
                            },  
                            {  
                                "Confidence": 1,  
                                "Content": "the",  
                                "EndTime": 19.31,  
                                "Speaker": "0",  
                                "StartTime": 19,  
                                "Type": "pronunciation",  
                                "VocabularyFilterMatch": false  
                            }  
                        ]  
                    ]  
                ]  
            }  
        ]  
    }  
}
```

```
{  
    "Confidence": 1,  
    "Content": "last",  
    "EndTime": 19.86,  
    "Speaker": "0",  
    "StartTime": 19.32,  
    "Type": "pronunciation",  
    "VocabularyFilterMatch": false  
},  
...  
{  
    "Confidence": 1,  
    "Content": "chronic",  
    "EndTime": 22.55,  
    "Speaker": "0",  
    "StartTime": 21.97,  
    "Type": "pronunciation",  
    "VocabularyFilterMatch": false  
},  
...  
{  
    "Confidence": 1,  
    "Content": "fatigue",  
    "EndTime": 24.42,  
    "Speaker": "0",  
    "StartTime": 23.95,  
    "Type": "pronunciation",  
    "VocabularyFilterMatch": false  
},  
{  
    "EndTime": 25.22,  
    "StartTime": 25.22,  
    "Type": "speaker-change",  
    "VocabularyFilterMatch": false  
},  
{  
    "Confidence": 0.99,  
    "Content": "True",  
    "EndTime": 25.63,  
    "Speaker": "1",  
    "StartTime": 25.22,  
    "Type": "pronunciation",  
    "VocabularyFilterMatch": false  
},  
{  
    "Content": ".",  
    "EndTime": 25.63,  
    "StartTime": 25.63,  
    "Type": "punctuation",  
    "VocabularyFilterMatch": false  
}  
],  
    "Transcript": "From the last note she still has mild sleep deprivation and  
chronic fatigue True."  
}  
],  
    "EndTime": 25.63,  
    "IsPartial": false,  
    "ResultId": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXX",  
    "StartTime": 18.74  
}  
]  
}  
}
```

Amazon Transcribe Medical breaks your incoming audio stream based on natural speech segments, such as a change in speaker or a pause in the audio. The transcription is returned progressively to your application, with each response containing more transcribed speech until the entire segment is transcribed. The above code is a truncated example of a fully-transcribed speech segment. Speaker labels only appear for entirely transcribed segments.

The following list shows the organization of the objects and parameters in a streaming transcription output.

Transcript

Each speech segment has its own `Transcript` object.

Results

Each `Transcript` object has its own `Results` object. This object contains the `isPartial` field. When its value is `false`, the results returned are for an entire speech segment.

Alternatives

Each `Results` object has an `Alternatives` object.

Items

Each `Alternatives` object has its own `Items` object that contains information about each word and punctuation mark in the transcription output. When you enable speaker identification, each word has a `Speaker` label for fully-transcribed speech segments. Amazon Transcribe Medical uses this label to assign a unique integer to each speaker it identifies in the stream. The `Type` parameter having a value of `speaker-change` indicates that one person has stopped speaking and that another person is about to begin.

Transcript

Each `Items` object contains a transcribed speech segment as the value of the `Transcript` field.

For more information about WebSocket requests, see [Creating a pre-signed URL \(p. 215\)](#).

Transcribing multi-channel audio

If you have an audio file or stream that has multiple channels, you can use *channel identification* to transcribe the speech from each of those channels. Amazon Transcribe Medical transcribes the speech from each channel separately. It combines the separate transcriptions of each channel into a single transcription output.

Use channel identification to identify the separate channels in your audio and transcribe the speech from each of those channels. Enable this in situations such as a caller and agent scenario. Use this to distinguish a caller from an agent in recordings or streams from contact centers that perform drug safety monitoring.

You can enable channel identification for both batch processing and real-time streaming. The following list describes how to enable it for each method.

- Batch transcription – Console and [StartMedicalTranscriptionJob \(p. 370\)](#) API
- Streaming transcription – WebSocket streaming and [StartMedicalStreamTranscription \(p. 402\)](#) API

Transcribing multi-channel audio files

When you transcribe an audio file, Amazon Transcribe Medical returns a list of *items* for each channel. An item is a transcribed word or punctuation mark. Each word has a start time and an end time. If a person

on one channel speaks over a person on a separate channel, the start times and end times of the items for each channel overlap while the individuals are speaking over each other.

By default, you can transcribe audio files with two channels. You can request a quota increase if you need to transcribe files that have more than two channels. For information about requesting a quota increase, see [AWS service quotas](#).

To transcribe multi-channel audio in a batch transcription job, use the Amazon Transcribe Medical console or the [StartMedicalTranscriptionJob \(p. 370\)](#) API.

Console

To use the console to enable channel identification in your batch transcription job, you enable audio identification and then channel identification. Channel identification is a subset of audio identification in the console.

To transcribe a multi-channel audio file (console)

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, under Amazon Transcribe Medical, choose **Transcription jobs**.
3. Choose **Create job**.
4. On the **Specify job details** page, provide information about your transcription job.
5. Choose **Next**.
6. Enable **Audio identification**.
7. For **Audio identification type**, choose **Channel identification**.
8. Choose **Create**.

API

To transcribe a multi-channel audio file (API)

- For the [StartMedicalTranscriptionJob \(p. 370\)](#) API, specify the following.
 - a. For **TranscriptionJobName**, specify a name unique to your AWS account.
 - b. For **LanguageCode**, specify the language code that corresponds to the language spoken in the audio file. The valid value is `en-US`.
 - c. For the **MediaFileUri** parameter of the **Media** object, specify the name of the media file that you want to transcribe.
 - d. For the **Settings** object, set **ChannelIdentification** to `true`.

The following is an example request using the AWS SDK for Python (Boto3).

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "your-transcription-job-name"
job_uri = "the-Amazon-S3-object-URL-of-your-media-file"
transcribe.start_medical_transcription_job(
    MedicalTranscriptionJobName=job_name,
    Media = {'MediaFileUri': job_uri},
    MediaFormat = 'mp4',
    LanguageCode = 'en-US',
    OutputBucketName = 'DOC-EXAMPLE-BUCKET1',
    Specialty = 'PRIMARYCARE',
```

```
Type = 'CONVERSATION',
Settings = {
    'ChannelIdentification': True
}
)
while True:
    status = transcribe.get_transcription_job(MedicalTranscriptionJobName=job_name)
    if status['MedicalTranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED',
'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)
```

AWS CLI

To transcribe a multi-channel audio file using a batch transcription job (AWS CLI)

- Run the following code.

```
aws transcribe start-medical-transcription-job \
--cli-input-json file://example-start-command.json
```

The following is the code of `example-start-command.json`.

```
{
    "MedicalTranscriptionJobName": "multichannel-conversation-medical-transcription-
job",
    "LanguageCode": "language-code",
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION",
    "OutputBucketName": "DOC-EXAMPLE-BUCKET",
    "Media": {
        "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/example-audio-file.extension"
    },
    "Settings": {
        "ChannelIdentification": true
    }
}
```

The following is the response.

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "multichannel-conversation-medical-
transcription-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
        "Media": {
            "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/example-audio-file.extension"
        },
        "StartTime": "2020-09-20T23:46:44.081000+00:00",
        "CreationTime": "2020-09-20T23:46:44.053000+00:00",
        "Settings": {
```

```
        "ChannelIdentification": true
    },
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION"
}
}
```

The following code shows the transcription output for an audio file that has a conversation on two channels.

```
{
  "jobName": "job id",
  "accountId": "account id",
  "results": {
    "transcripts": [
      {
        "transcript": "When you try ... It seems to ..."
      }
    ],
    "channel_labels": {
      "channels": [
        {
          "channel_label": "ch_0",
          "items": [
            {
              "start_time": "12.282",
              "end_time": "12.592",
              "alternatives": [
                {
                  "confidence": "1.0000",
                  "content": "When"
                }
              ],
              "type": "pronunciation"
            },
            {
              "start_time": "12.592",
              "end_time": "12.692",
              "alternatives": [
                {
                  "confidence": "0.8787",
                  "content": "you"
                }
              ],
              "type": "pronunciation"
            },
            {
              "start_time": "12.702",
              "end_time": "13.252",
              "alternatives": [
                {
                  "confidence": "0.8318",
                  "content": "try"
                }
              ],
              "type": "pronunciation"
            },
            ...
          ]
        }
      ]
    }
}
```

```
        },
        {
            "channel_label": "ch_1",
            "items": [
                {
                    "start_time": "12.379",
                    "end_time": "12.589",
                    "alternatives": [
                        {
                            "confidence": "0.5645",
                            "content": "It"
                        }
                    ],
                    "type": "pronunciation"
                },
                {
                    "start_time": "12.599",
                    "end_time": "12.659",
                    "alternatives": [
                        {
                            "confidence": "0.2907",
                            "content": "seems"
                        }
                    ],
                    "type": "pronunciation"
                },
                {
                    "start_time": "12.669",
                    "end_time": "13.029",
                    "alternatives": [
                        {
                            "confidence": "0.2497",
                            "content": "to"
                        }
                    ],
                    "type": "pronunciation"
                },
                ...
            ]
        }
    }
```

Transcribing multi-channel audio streams

You can transcribe audio from separate channels in either HTTP/2 or WebSocket streams using the [StartMedicalStreamTranscription \(p. 402\)](#) API.

By default, you can transcribe streams with two channels. You can request a quota increase if you need to transcribe streams that have more than two channels. For information about requesting a quota increase, see [AWS service quotas](#).

Transcribing multi-channel audio in an HTTP/2 stream

To transcribe multi-channel audio in an HTTP/2 stream, use the [StartMedicalStreamTranscription \(p. 402\)](#) API and specify the following:

- `LanguageCode` – The language code of the audio. The valid value is `en-US`.
- `MediaEncoding` – The encoding of the audio. Valid values are `ogg-opus`, `flac`, and `pcm`.
- `EnableChannelIdentification` – `true`
- `NumberOfChannels` – the number of channels in your streaming audio.

For more information on setting up an HTTP/2 stream to transcribe a medical conversation, see [Streaming request \(p. 203\)](#).

Transcribing multi-channel audio in a WebSocket stream

To identify speakers in WebSocket streams, use the following format to create a pre-signed URL and start a WebSocket request. Specify `enable-channel-identification` as `true` and the number of channels in your stream in `number-of-channels`. A pre-signed URL contains the information needed to set up bi-directional communication between your application and Amazon Transcribe Medical.

```
GET wss://transcribestreaming.region.amazonaws.com:8443/stream-transcription-websocket
?language-code=languageCode
    &X-Amz-Algorithm=AWS4-HMAC-SHA256
    &X-Amz-Credential=Signature Version 4 credential scope
    &X-Amz-Date=date
    &X-Amz-Expires=time in seconds until expiration
    &X-Amz-Security-Token=security-token
    &X-Amz-Signature=Signature Version 4 signature
    &X-Amz-SignedHeaders=host
    &media-encoding=mediaEncoding
    &sample-rate=mediaSampleRateHertz
    &sessionId=sessionId
    &enable-channel-identification=true
    &number-of-channels=number of channels in your audio stream
```

For more information about WebSocket requests, see [Creating a pre-signed URL \(p. 77\)](#).

Multi-channel streaming output

The output of a streaming transcription is the same for HTTP/2 and WebSocket requests. The following is an example output.

```
{
    "resultId": "XXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXX",
    "startTime": 0.11,
    "endTime": 0.66,
    "isPartial": false,
    "alternatives": [
        {
            "transcript": "Left.",
            "items": [
                {
                    "startTime": 0.11,
                    "endTime": 0.45,
                    "type": "pronunciation",
                    "content": "Left",
                    "vocabularyFilterMatch": false
                },
                {
                    "startTime": 0.45,
                    "endTime": 0.45,
                    "type": "punctuation",
                    "content": ".",
                    "vocabularyFilterMatch": false
                }
            ]
        }
    ],
}
```

```
    "channelId": "ch_0"  
}
```

For each speech segment, there is a `channelId` flag that indicates which channel the speech belongs to.

Transcribing a medical dictation

You can use Amazon Transcribe Medical to transcribe clinician-dictated medical notes using either a batch transcription job or a real-time stream. Batch transcription jobs enable you to transcribe audio files. You specify the medical specialty of the clinician in your transcription job or stream to ensure that Amazon Transcribe Medical produces transcription results with the highest possible accuracy.

You can transcribe a medical dictation in the following specialties:

- Cardiology – available in streaming transcription only
- Neurology – available in streaming transcription only
- Oncology – available in streaming transcription only
- Primary Care – includes the following types of medical practice:
 - Family medicine
 - Internal medicine
 - Obstetrics and Gynecology (OB-GYN)
 - Pediatrics
- Radiology – available in streaming transcription only
- Urology – available in streaming transcription only

You can improve transcription accuracy by using custom vocabularies. For information on how medical custom vocabularies work, see [Improving transcription accuracy with medical custom vocabularies \(p. 247\)](#).

By default, Amazon Transcribe Medical returns the transcription with the highest confidence level. If you'd like to configure it to return alternative transcriptions, see [Generating alternative transcriptions \(p. 262\)](#).

For information about how numbers and medical measurements appear in the transcription output, see [Transcribing numbers \(p. 192\)](#) and [Transcribing medical terms and measurements \(p. 193\)](#).

Topics

- [Transcribing an audio file of a medical dictation \(p. 241\)](#)
- [Transcribing a medical dictation in a real-time stream \(p. 245\)](#)

Transcribing an audio file of a medical dictation

Use a batch transcription job to transcribe audio files of medical conversations. You can use this to transcribe a clinician-patient dialogue. You can start a batch transcription job in either the [StartMedicalTranscriptionJob \(p. 370\)](#) API or the Amazon Transcribe Medical console.

When you start a medical transcription job with the [StartMedicalTranscriptionJob \(p. 370\)](#) API, you specify `PRIMARYCARE` as the value of the `Specialty` parameter.

Console

To transcribe a clinician-patient dialogue (console)

To use the console to transcribe a clinician-patient dialogue, create a transcription job and choose **Conversation for Audio input type**.

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, under Amazon Transcribe Medical, choose **Transcription jobs**.
3. Choose **Create job**.
4. On the **Specify job details** page, under **Job settings**, specify the following.
 - a. **Name** – the name of the transcription job.
 - b. **Audio input type – Dictation**
5. For the remaining fields, specify the Amazon Simple Storage Service (Amazon S3) location of your audio file and where you want to store the output of your transcription job.
6. Choose **Next**.
7. Choose **Create**.

API

To transcribe a medical conversation using a batch transcription job (API)

- For the [StartMedicalTranscriptionJob \(p. 370\)](#) API, specify the following.
 - a. For `MedicalTranscriptionJobName`, specify a name unique in your AWS account.
 - b. For `LanguageCode`, specify the language code that corresponds to the language spoken in your audio file and the language of your vocabulary filter.
 - c. In the `MediaFileUri` parameter of the `Media` object, specify the name of the audio file that you want to transcribe.
 - d. For `Specialty`, specify the medical specialty of the clinician speaking in the audio file.
 - e. For `Type`, specify `Dictation`.
 - f. For `OutputBucketName`, specify the Amazon Simple Storage Service (Amazon S3) bucket to store the transcription results.

The following is an example request that uses the AWS SDK for Python (Boto3) to transcribe a medical dictation of a clinician in the `PRIMARYCARE` specialty.

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "your-medical-conversation-transcription-job-name"
job_uri = "s3://DOC-EXAMPLE-BUCKET1/example-audio-file.extension"
transcribe.start_medical_transcription_job(
    MedicalTranscriptionJobName = job_name,
    Media = {'MediaFileUri': job_uri},
    LanguageCode = 'en-US',
    Specialty = 'PRIMARYCARE',
    Type = 'Dictation',
    OutputBucketName = 'DOC-EXAMPLE-BUCKET1'
)
while True:
```

```
    status =
transcribe.get_medical_transcription_job(MedicalTranscriptionJobName=job_name)
    if status['MedicalTranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED',
'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)
```

The following example code shows the transcription results of a medical dictation.

```
{
    "jobName": "dictation-medical-transcription-job",
    "accountId": "account-id-number",
    "results": {
        "transcripts": [
            {
                "transcript": "... came for a follow up visit today..."
            }
        ],
        "items": [
            {
                ...
                "start_time": "4.85",
                "end_time": "5.12",
                "alternatives": [
                    {
                        "confidence": "1.0",
                        "content": "came"
                    }
                ],
                "type": "pronunciation"
            },
            {
                ...
                "start_time": "5.12",
                "end_time": "5.29",
                "alternatives": [
                    {
                        "confidence": "1.0",
                        "content": "for"
                    }
                ],
                "type": "pronunciation"
            },
            {
                ...
                "start_time": "5.29",
                "end_time": "5.33",
                "alternatives": [
                    {
                        "confidence": "0.9955",
                        "content": "a"
                    }
                ],
                "type": "pronunciation"
            },
            {
                ...
                "start_time": "5.33",
                "end_time": "5.66",
                "alternatives": [
                    {
                        "confidence": "0.9754",
                        ...
                    }
                ],
                "type": "pronunciation"
            }
        ]
    }
}
```

```
        "content": "follow"
    },
],
"type": "pronunciation"
},
{
"start_time": "5.66",
"end_time": "5.75",
"alternatives": [
{
"confidence": "0.9754",
"content": "up"
}
],
"type": "pronunciation"
},
{
"start_time": "5.75",
"end_time": "6.02",
"alternatives": [
{
"confidence": "1.0",
"content": "visit"
}
]
...
},
"status": "COMPLETED"
}
```

AWS CLI

To identify the speakers in an audio file using a batch transcription job (AWS CLI)

- Run the following code.

```
aws transcribe start-medical-transcription-job \
--cli-input-json file://filepath/example-start-command.json
```

The following code shows the contents of `example-start-command.json`.

```
{
    "MedicalTranscriptionJobName": "conversation-medical-transcription-job",
    "LanguageCode": "en-US",
    "Specialty": "PRIMARYCARE",
    "Type": "DICTATION",
    "OutputBucketName": "DOC-EXAMPLE-BUCKET",
    "Media": {
        "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET1/example-audio-file.extension"
    }
}
```

The following is the response from running the preceding CLI command.

```
{  
    "MedicalTranscriptionJob": {  
        "MedicalTranscriptionJobName": "example-dictation-medical-transcription-job",  
        "TranscriptionJobStatus": "IN_PROGRESS",  
        "LanguageCode": "en-US",  
        "Media": {  
            "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET1/example-audio-file.extension"  
        },  
        "StartTime": "2020-09-20T00:35:22.256000+00:00",  
        "CreationTime": "2020-09-20T00:35:22.218000+00:00",  
        "Specialty": "PRIMARYCARE",  
        "Type": "DICTATION"  
    }  
}
```

Transcribing a medical dictation in a real-time stream

Use a WebSocket stream to transcribe a medical dictation as an audio stream. You can also use the Amazon Transcribe Medical console to transcribe speech that you or others speak directly into a microphone.

For an HTTP/2 or a WebSocket stream, you can transcribe audio in the following medical specialties:

- Cardiology
- Oncology
- Neurology
- Primary Care
- Radiology
- Urology

Each medical specialty includes many types of procedures and appointments. Clinicians therefore dictate many different types of notes. Use the following examples as guidance to help you specify the value of the `specialty` URL parameter of the WebSocket request, or the `Specialty` parameter of the [StartMedicalStreamTranscription \(p. 402\)](#) API:

- For a dictation after electrophysiology or echocardiogram procedure, choose CARDIOLOGY.
- For a dictation after a surgical oncology or radiation oncology procedure, choose ONCOLOGY.
- For a physician dictating notes indicating a diagnosis of encephalitis, choose NEUROLOGY.
- For a dictation of procedure notes to break up a bladder stone, choose UROLOGY.
- For a dictation of clinician notes after an internal medicine consultation, choose PRIMARYCARE.
- For a dictation of a physician communicating the findings of a CT scan, PET scan, MRI, or radiograph, choose RADIOLOGY.
- For a dictation of physician notes after a gynecology consultation, choose PRIMARYCARE.

To improve transcription accuracy of specific terms in a real-time stream, use a custom vocabulary. To enable a custom vocabulary, set the value of `vocabulary-name` to the name of the custom vocabulary you want to use.

Transcribing a dictation spoken into your microphone with the console

To use the console to transcribe streaming audio of a medical dictation, choose the option to transcribe a medical dictation, start the stream, and begin speaking into the microphone.

To transcribe streaming audio of a medical dictation (console)

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, under Amazon Transcribe Medical, choose **Real-time transcription**.
3. Choose **Dictation**.
4. For **Medical specialty**, choose the medical specialty of the clinician speaking in the stream.
5. Choose **Start streaming**.
6. Speak into the microphone.

Transcribing a dictation in an HTTP/2 stream

To transcribe an HTTP/2 stream of a medical dictation, use the [StartMedicalStreamTranscription \(p. 402\)](#) API and specify the following:

- **LanguageCode** – The language code. The valid value is `en-US`
- **MediaEncoding** – The encoding used for the input audio. Valid values are `pcm`, `ogg-opus`, and `flac`.
- **Specialty** – The specialty of the medical professional.
- **Type** – `DICTATION`

For more information on setting up an HTTP/2 stream to transcribe a medical dictation, see [Streaming request \(p. 203\)](#).

Using a WebSocket streaming request to transcribe a medical dictation

To transcribe a medical dictation in a real-time stream using a WebSocket request, you create a pre-signed URL. This URL contains the information needed to set up the audio stream between your application and Amazon Transcribe Medical. For more information on creating WebSocket requests, see [Establish a bi-directional connection using the WebSocket protocol \(p. 215\)](#).

Use the following template to create your pre-signed URL.

```
GET https://transcribestreaming.region.amazonaws.com:8443/medical-stream-transcription-
websocket
?language-code=languageCode
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=Signature Version 4 credential scope
&X-Amz-Date=date
&X-Amz-Expires=time in seconds until expiration
&X-Amz-Security-Token=security-token
&X-Amz-Signature=Signature Version 4 signature
&X-Amz-SignedHeaders=host
&media-encoding=mediaEncoding
&sample-rate=mediaSampleRateHertz
&session-id=sessionId
&specialty=medicalSpecialty
&type=DICTATION
&vocabulary-name=vocabularyName
&show-speaker-label=boolean
```

For more information on creating pre-signed URLs, see [Creating a pre-signed URL \(p. 215\)](#).

Improving transcription accuracy with medical custom vocabularies

To improve transcription accuracy in Amazon Transcribe Medical, create and use one or more medical custom vocabularies. A *custom vocabulary* is a collection of words or phrases that are domain-specific. This collection helps improve the performance of Amazon Transcribe Medical in transcribing those words or phrases.

You are responsible for the integrity of your own data when you use Amazon Transcribe Medical. Do not enter confidential information, personal information (PII), or protected health information (PHI), into a custom vocabulary.

For best results, create separate small custom vocabularies that each help transcribe a specific audio recording. You receive greater improvements in transcription accuracy than if you created one large custom vocabulary to use with all of your recordings.

By default, you can have up to 100 custom vocabularies in your AWS account. A custom vocabulary can't exceed 50 KB in size. For information on requesting an increase to the number of custom vocabularies that you can have in your account, see [AWS service quotas](#).

Custom vocabularies are available in US English (en-US).

Topics

- [Creating a text file for your medical custom vocabulary \(p. 247\)](#)
- [Using a text file to create a medical custom vocabulary \(p. 250\)](#)
- [Transcribing an audio file using a medical custom vocabulary \(p. 252\)](#)
- [Transcribing a real-time stream using a medical custom vocabulary \(p. 253\)](#)
- [Character sets for Amazon Transcribe Medical \(p. 255\)](#)

Creating a text file for your medical custom vocabulary

To create a custom vocabulary, you create a text file that is in UTF-8 format. In this file, you create a four column table, with each column specifying a field. Each field tells Amazon Transcribe Medical either how the domain-specific terms are pronounced or how to display these terms in your transcriptions. You store the text file containing these fields in an Amazon Simple Storage Service (Amazon S3) bucket.

Understanding how to format your text file

To create a medical custom vocabulary, you enter the column names as a header row. You enter the values for each column beneath the header row.

The following are the names of the four columns of the table:

- **Phrase** – column required, values required
- **IPA** – column required, values can be optional
- **SoundsLike** – column required, values can be optional
- **DisplayAs** – column required, values can be optional

When you create a custom vocabulary, make sure that you:

- Separate each column with a single Tab character. Amazon Transcribe throws an error message if you try to separate the columns with spaces or multiple Tab characters.
- Make sure that there's no trailing spaces or white space after each value within a column.

Make sure that the values that you enter for each column:

- Have fewer than 256 characters, including hyphens
- Use only characters from the allowed character set, see [Character sets for Amazon Transcribe Medical \(p. 255\)](#).

Entering values for the columns of the table

The following information shows you how to specify values for the four columns of the table:

- **Phrase** – The word or phrase that should be recognized. You must enter values in this column.

If the entry is a phrase, separate the words with a hyphen (-). For example, enter **cerebral autosomal dominant arteriopathy with subcortical infarcts and leukoencephalopathy as cerebral-autosomal-dominant-arteriopathy-with-subcortical-infarcts-and-leukoencephalopathy**.

Enter acronyms or other words whose letters should be pronounced individually as single letters followed by dots, such as **D.N.A.** or **S.T.E.M.I.** To enter the plural form of an acronym, such as "STEMIs," separate the "s" from the acronym with a hyphen: "**S.T.E.M.I-s**" You can use either uppercase or lowercase letters for acronyms.

The **Phrase** column is required. You can use any of the allowed characters for the input language. For allowed characters, see [Character sets for Amazon Transcribe Medical \(p. 255\)](#). If you don't specify the **DisplayAs** column, Amazon Transcribe Medical uses the contents of the **Phrase** column in the output file.

- **IPA** (column required, values can be optional) – To specify the pronunciation of a word or phrase, you can include characters in the [International Phonetic Alphabet \(IPA\)](#) in this column. The IPA column can't contain leading or trailing spaces, and you must use a single space to separate each phoneme in the input. For example, in English you would enter the phrase **acute-respiratory-distress-syndrome** as # k j u t # s p # t # i d # s t # s s # n d # o# m. You would enter the phrase **A.L.L.** as e# # l # 1.

Even if you don't specify the contents of the **IPA** column, you must include a blank **IPA** column. If you include values in the **IPA** column, you can't provide values for the **SoundsLike** column.

For a list of allowed IPA characters for a specific language, see [Character sets for Amazon Transcribe Medical \(p. 255\)](#). US English is the only language available in Amazon Transcribe Medical.

- **SoundsLike** (column required, values can be optional) – You can break a word or phrase down into smaller segments and provide a pronunciation for each segment using the standard orthography of the language to mimic the way that the word sounds. For example, you can provide pronunciation hints for the phrase **cerebral-autosomal-dominant-arteriopathy-with-subcortical-infarcts-and-leukoencephalopathy** like this: **sir-e-brul-aut-o-som-ul-dah-mi-nant-ar-ter-ri-o-pa-thy-with-sub-cor-ti-cul-in-farcts-and-lewk-o-en-ce-phul-ah-pu-thy**. The hint for the phrase **atrioventricular-nodal-reentrant-tachycardia** would look like this: **ay-tree-o-ven-trick-u-lar-node-al-re-entr-ant-tack-ih-card-ia**. You separate each part of the hint with a hyphen (-).

Even if you don't provide values for the **SoundsLike** column, you must include a blank **SoundsLike** column. If you include values in the **SoundsLike** column, you can't provide values for the **IPA** column.

You can use any of the allowed characters for the input language. For the list of allowed characters, see [Character sets for Amazon Transcribe Medical \(p. 255\)](#).

- **DisplayAs** (column required, values can be optional) – Defines how the word or phrase looks when it's output. For example, if the word or phrase is **cerebral-autosomal-dominant-arteriopathy-with-subcortical-infarcts-and-leukoencephalopathy**, you can specify the display form as cerebral autosomal dominant arteriopathy with subcortical infarcts and leukoencephalopathy, so that the hyphen is not present. You can also specify DisplayAs as **CADASIL** if you'd like to show the acronym instead of the full term in the output.

If you don't specify the **DisplayAs** column, Amazon Transcribe Medical uses the **Phrase** column from the input file in the output.

You can use any UTF-8 character in the **DisplayAs** column.

You can include spaces only for the values in the **IPA** and **DisplayAs** columns.

To create the text file of your custom vocabulary, place each word or phrase in your text file on a separate line. Separate the columns with Tab characters. Include spaces only for values in the **IPA** and **DisplayAs** columns. Save the file with the extension **.txt** in an S3 bucket in the same AWS Region where you use Amazon Transcribe Medical to create your custom vocabulary.

If you edit your text file in Windows, make sure that your file is in **LF** format and not in **CRLF** format. Otherwise, you will be unable to create your custom vocabulary. Some text editors enable you to change the formatting with Find and Replace commands.

The following examples show text that you can use to create custom vocabularies. To create a custom vocabulary from these examples, copy an example into a text editor, replace **[TAB]** with a Tab character, and upload the saved text file to Amazon S3.

```
Phrase[TAB]IPA[TAB]SoundsLike[TAB]DisplayAs
acute-respiratory-distress-syndrome[TAB][TAB][TAB]acute respiratory distress syndrome
A.L.L.[TAB]e# # 1 # 1[TAB][TAB]ALL
atrioventricular-nodal-reentrant-tachycardia[TAB][TAB]ay-tree-o-ven-trick-u-lar-node-al-re-
entr-ant-tack-ih-card-ia[TAB]
```

You can enter columns in any order. The following examples show other valid structures for the custom vocabulary input file.

```
Phrase[TAB]SoundsLike[TAB]IPA[TAB]DisplayAs
acute-respiratory-distress-syndrome[TAB][TAB][TAB]acute respiratory distress syndrome
A.L.L.[TAB][TAB]e# # 1 # 1[TAB]ALL
atrioventricular-nodal-reentrant-tachycardia[TAB]ay-tree-o-ven-trick-u-lar-node-al-re-entr-
ant-tack-ih-card-ia[TAB][TAB]
```

```
DisplayAs[TAB]SoundsLike[TAB]IPA[TAB]Phrase
acute respiratory distress syndrome[TAB][TAB][TAB]acute-respiratory-distress-syndrome
ALL[TAB][TAB]e# # 1 # 1[TAB]A.L.L.
[TAB]ay-tree-o-ven-trick-u-lar-node-al-re-entr-ant-tack-ih-card-ia[TAB]
[TAB]atrioventricular-nodal-reentrant-tachycardia
```

For reading ease, the following tables show the preceding examples more clearly in html format. They are meant only to illustrate the examples above.

Phrase	IPA	SoundsLike	DisplayAs
acute-respiratory-distress-syndrome			acute respiratory distress syndrome
A.L.L.	eɪ ε l ε l		ALL
atrioventricular-nodal-reentrant-tachycardia		ay-tree-o-ven-trick-u-lar-node-al-re-entr-ant-tack-ih-card-ia	

Phrase	SoundsLike	IPA	DisplayAs
acute-respiratory-distress-syndrome			acute respiratory distress syndrome
atrioventricular-nodal-reentrant-tachycardia	ay-tree-o-ven-trick-u-lar-node-al-re-entr-ant-tack-ih-card-ia		
A.L.L.		eɪ ε l ε l	ALL

DisplayAs	SoundsLike	IPA	Phrase
acute respiratory distress syndrome			acute-respiratory-distress-syndrome
ALL		eɪ ε l ε l	A.L.L.
	ay-tree-o-ven-trick-u-lar-node-al-re-entr-ant-tack-ih-card-ia		atrioventricular-nodal-reentrant-tachycardia

Using a text file to create a medical custom vocabulary

To create a custom vocabulary, you must have prepared a text file that contains a collection of words or phrases. Amazon Transcribe Medical uses this text file to create a custom vocabulary that you can use to improve the transcription accuracy of those words or phrases. You can create a custom vocabulary using the [CreateMedicalVocabulary \(p. 287\)](#) API or the Amazon Transcribe Medical console.

Console

To create a custom vocabulary (console)

To use the console to create a custom vocabulary, you provide the Amazon S3 URI of the text file containing your words or phrases.

1. Sign in to AWS Management Console and open the Amazon Transcribe Medical console at [Amazon Transcribe Medical console](#).
2. In the navigation pane, under Amazon Transcribe Medical, choose **Custom vocabulary**.
3. For **Name**, under **Vocabulary settings**, choose a name for your custom vocabulary.

4. Specify the location of your audio file or video file in Amazon S3:
 - For **Vocabulary input file location on S3** under **Vocabulary settings**, specify the Amazon S3 URI that identifies the text file you will use to create your custom vocabulary.
 - For **Vocabulary input file location in Amazon S3**, choose **Browse S3** to browse for the text file and choose it.
5. Choose **Create vocabulary**.

You can see the processing status of your custom vocabulary in the console.

API

To create a medical custom vocabulary (API)

- For the [StartTranscriptionJob \(p. 376\)](#) API, specify the following.
 - a. For **LanguageCode**, specify **en-US**.
 - b. For **VocabularyFileUri**, specify the Amazon S3 location of the text file that you use to define your custom vocabulary.
 - c. For **VocabularyName**, specify a name for your custom vocabulary. The name you specify must be unique within your AWS account.

To see the processing status of your custom vocabulary, use the [GetMedicalVocabulary \(p. 326\)](#) API.

The following is an example request using the AWS SDK for Python (Boto3) to create a custom vocabulary.

```
from __future__ import print_function
import time
import boto3

transcribe = boto3.client('transcribe')
vocab_name = "example-med-custom-vocab"
text_file_uri = "https://DOC-EXAMPLE-
BUCKET1.s3-Region.amazonaws.com/example_custom_vocabulary.txt"
transcribe.create_medical_vocabulary(
    VocabularyName = vocab_name,
    VocabularyFileUri = text_file_uri,
    LanguageCode = 'en-US',
)
while True:
    status = transcribe.get_medical_vocabulary(VocabularyName=vocab_name)
    if status['VocabularyState'] in ['READY', 'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)
```

AWS CLI

To identify the speakers in an audio file using a batch transcription job (AWS CLI)

- Run the following code.

```
aws transcribe create-medical-vocabulary \
    --vocabulary-name your-custom-medical-vocabulary-name \
    --language-code en-US \
    --vocabulary-file-uri https://DOC-EXAMPLE-BUCKET1.AWS-Region.amazonaws.com/your-medical-custom-vocabulary
```

The following is the response from running the preceding CLI command.

```
{  
    "VocabularyName": "cli-medical-vocab-1",  
    "LanguageCode": "en-US",  
    "VocabularyState": "PENDING"  
}
```

Transcribing an audio file using a medical custom vocabulary

Use the [StartMedicalTranscriptionJob \(p. 370\)](#) or the Amazon Transcribe Medical console to start a transcription job that uses a custom vocabulary to improve transcription accuracy.

Console

To start a transcription job with a custom vocabulary (console)

To use the console to use your custom vocabulary in your transcription job, .

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, under Amazon Transcribe Medical, choose **Transcription jobs**.
3. Choose **Create job**.
4. On the **Specify job details** page, provide information about your transcription job.
5. Choose **Next**.
6. Under **Customization**, enable **Custom vocabulary**.
7. Under **Vocabulary selection**, choose a custom vocabulary.
8. Choose **Create**.

API

To identify speakers in an audio file using a batch transcription job (API)

- For the [StartMedicalTranscriptionJob \(p. 370\)](#) API, specify the following.
 - a. For `MedicalTranscriptionJobName`, specify a name that is unique in your AWS account.
 - b. For `LanguageCode`, specify the language code that corresponds to the language spoken in your audio file and the language of your vocabulary filter.
 - c. For the `MediaFileUri` parameter of the `Media` object, specify the name of the audio file that you want to transcribe.
 - d. For `Specialty`, specify the medical specialty of the clinician speaking in the audio file.

- e. For **Type**, specify whether the audio file is a conversation or a dictation.
- f. For **OutputBucketName**, specify the Amazon Simple Storage Service (Amazon S3) bucket to store the transcription results.
- g. For the **Settings** object, specify the following.
 - **VocabularyName** – the name of your custom vocabulary.

The following request uses the AWS SDK for Python (Boto3) to start a batch transcription job with a custom vocabulary.

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "example-med-vocab-transcription"
job_uri = "https://DOC-EXAMPLE-BUCKET1.s3-Region.amazonaws.com/example-audio-
file.extension"
transcribe.start_medical_transcription_job(
    MedicalTranscriptionJobName=job_name,
    Media = {'MediaFileUri': job_uri},
    LanguageCode = 'en-US',
    Specialty = 'PRIMARYCARE',
    Type = 'CONVERSATION',
    OutputBucketName = 'DOC-EXAMPLE-BUCKET2',
    Settings = {
        'VocabularyName': 'example-med-custom-vocab'
    }
)

while True:
    status = transcribe.get_medical_transcription_job(MedicalTranscriptionJobName=job_name)
    if status['MedicalTranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED',
    'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)
```

Transcribing a real-time stream using a medical custom vocabulary

To improve transcription accuracy in a real-time stream, you can use a custom vocabulary using either HTTP/2 or WebSocket streams. To start an HTTP/2 request, use the [StartMedicalStreamTranscription \(p. 402\)](#) API. You can use a custom vocabulary in real-time using either the Amazon Transcribe Medical console, the [StartMedicalStreamTranscription \(p. 402\)](#) API, or by using the WebSocket protocol.

Transcribing a dictation that is spoken into your Microphone (Console)

To use the console to transcribe streaming audio of a medical dictation, choose the option to transcribe a medical dictation, start the stream, and begin speaking into the microphone.

To transcribe streaming audio of a medical dictation (console)

1. Sign in to AWS Management Console and open the Amazon Transcribe Medical console at [Amazon Transcribe Medical console](#).

2. In the navigation pane, under Amazon Transcribe Medical, choose **Real-time transcription**.
3. For **Medical specialty**, choose the medical specialty of the clinician speaking in the stream.
4. For **Audio input type**, choose either **Conversation** or **Dictation**.
5. For **Additional settings**, choose **Custom vocabulary**.
 - For **Vocabulary selection**, choose the custom vocabulary.
6. Choose **Start streaming**.
7. Speak into the microphone.

Identifying speakers in an HTTP/2 stream

The following is the syntax for the parameters of an HTTP/2 request.

```
POST /medical-stream-transcription HTTP/2
x-amzn-transcribe-language-code: LanguageCode
x-amzn-transcribe-sample-rate: MediaSampleRateHertz
x-amzn-transcribe-media-encoding: MediaEncoding
x-amzn-transcribe-vocabulary-name: VocabularyName
x-amzn-transcribe-specialty: Specialty
x-amzn-transcribe-type: Type
x-amzn-transcribe-show-speaker-label: ShowSpeakerLabel
x-amzn-transcribe-session-id: SessionId
Content-type: application/json

{
    "AudioStream": {
        "AudioEvent": {
            "AudioChunk": blob
        }
    }
}
```

To use a custom vocabulary in an HTTP/2 stream, use the [StartMedicalStreamTranscription \(p. 402\)](#) API and specify the following:

- For **LanguageCode**, specify the language code that corresponds to the language in the stream. The valid value is `en-US`.
- For **MediaSampleHertz**, specify the sample rate of the audio.
- For **Specialty**, specify the medical specialty of the provider.
- For **VocabularyName**, specify the name of the custom vocabulary.

Identifying speakers in a WebSocket request

To identify speakers in WebSocket streams with the API, use the following format to create a pre-signed URL to start a WebSocket request and set `vocabulary-name` to the name of the custom vocabulary.

```
GET https://transcribestreaming.region.amazonaws.com:8443/medical-stream-transcription-
websocket
?language-code=languageCode
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=Signature Version 4 credential scope
&X-Amz-Date=date
&X-Amz-Expires=time in seconds until expiration
```

```
&X-Amz-Security-Token=security-token
&X-Amz-Signature=Signature Version 4 signature
&X-Amz-SignedHeaders=host
&media-encoding=mediaEncoding
&sample-rate=mediaSampleRateHertz
&session-id=sessionId
&specialty=medicalSpecialty
&type=CONVERSATION
&vocabulary-name=vocabularyName
&show-speaker-label=boolean
```

Character sets for Amazon Transcribe Medical

To use custom vocabularies in Amazon Transcribe Medical, use the following character sets.

English character set

For English custom vocabularies, you can use the following characters in the `Phrase` and `SoundsLike` columns:

- a - z
- A - Z
- ' (apostrophe)
- - (hyphen)
- . (period)

You can use the following International Phonetic Alphabet (IPA) characters in the `IPA` column of the vocabulary input file.

Character	Code	Character	Code
au	0061 028A	w	0077
ai	0061 026A	z	007A
b	0062	æ	00E6
d	0064	ð	00F0
er	0065 026A	ŋ	014B
f	0066	ɑ	0251
g	0067	ɔ	0254
h	0068	ɔɪ	0254 026A
i	0069	ə	0259
j	006A	ɛ	025B
k	006B	ɜ	025D
l	006C	g	0261
l	006C 0329	ɪ	026A

Character	Code	Character	Code
m	006D	ј	0279
n	006E	ſ	0283
ŋ	006E 0329	υ	028A
ou	006F 028A	ʌ	028C
p	0070	ʍ	028D
s	0073	ʒ	0292
t	0074	ðʒ	02A4
u	0075	ʃ	02A7
v	0076	θ	03B8

Identifying personal health information (PHI) in a transcription

Use *Personal Health Information Identification* to label personal health information (PHI) in your transcription results. By reviewing labels, you can find PHI that could be used to identify a patient.

You can identify PHI using either a real-time stream or batch transcription job.

You can use your own post-processing to redact the PHI identified in the transcription output.

Use Personal Health Information Identification to identify the following types of PHI:

- Personal PHI:
 - Names – Full name or last name and initial
 - Gender
 - Age
 - Phone numbers
 - Dates (not including the year) that directly relate to the patient
 - Email addresses
- Geographic PHI:
 - Physical address
 - Zip code
 - Name of medical center or practice
- Account PHI:
 - Fax numbers
 - Social security numbers (SSNs)
 - Health insurance beneficiary numbers
 - Account numbers
 - Certificate or license numbers
 - Biometric identifiers
 - Voice prints

- Vehicle PHI:
 - Vehicle identification number (VIN)
 - License plate number
- Other PHI:
 - Web Uniform Resource Location (URL)
 - Internet Protocol (IP) address numbers

Amazon Transcribe Medical is a Health Insurance Portability and Accountability Act of 1996 (HIPAA) compliant service. For more information, see [What is Amazon Transcribe Medical? \(p. 188\)](#). For information about identifying PHI in an audio file, see [Identifying PHI in an audio file \(p. 257\)](#). For information about identifying PHI in a stream, see [Identifying PHI in a real-time stream \(p. 260\)](#).

Topics

- [Identifying PHI in an audio file \(p. 257\)](#)
- [Identifying PHI in a real-time stream \(p. 260\)](#)

Identifying PHI in an audio file

Use a batch transcription job to transcribe audio files and identify the personal health information (PHI) within them. When you activate Personal Health Information (PHI) Identification, Amazon Transcribe Medical labels the PHI that it identified in the transcription results. For information about the PHI that Amazon Transcribe Medical can identify, see [Identifying personal health information \(PHI\) in a transcription \(p. 256\)](#).

You can start a batch transcription job using either the [StartMedicalTranscriptionJob \(p. 370\)](#) API or the Amazon Transcribe Medical console.

Console

To use the console to transcribe a clinician-patient dialogue, create a transcription job and choose **Conversation** for **Audio input type**.

To transcribe an audio file and identify its PHI (console)

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, under Amazon Transcribe Medical, choose **Transcription jobs**.
3. Choose **Create job**.
4. On the **Specify job details** page, under **Job settings**, specify the following.
 - a. **Name** – The name of the transcription job that is unique to your AWS account.
 - b. **Audio input type** – **Conversation** or **Dictation**.
5. For the remaining fields, specify the Amazon Simple Storage Service (Amazon S3) location of your audio file and where you want to store the output of your transcription job.
6. Choose **Next**.
7. Under **Audio settings**, choose **PHI Identification**.
8. Choose **Create**.

API

To transcribe an audio file and identify its PHI using a batch transcription job (API)

- For the [StartMedicalTranscriptionJob \(p. 370\)](#) API, specify the following.

- a. For `MedicalTranscriptionJobName`, specify a name that is unique to your AWS account.
- b. For `LanguageCode`, specify the language code that corresponds to the language spoken in your audio file.
- c. For the `MediaFileUri` parameter of the `Media` object, specify the name of the audio file that you want to transcribe.
- d. For `Specialty`, specify the medical specialty of the clinician speaking in the audio file as `PRIMARYCARE`.
- e. For `Type`, specify either `CONVERSATION` or `DICTATION`.
- f. For `OutputBucketName`, specify the Amazon Simple Storage Service (Amazon S3) bucket where you want to store the transcription results.

The following is an example request that uses the AWS SDK for Python (Boto3) to transcribe an audio file and identify the PHI of a patient.

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "medical-conversation-transcription-job-name"
job_uri = "s3://DOC-EXAMPLE-BUCKET1/example-audio-file.extension"
transcribe.start_medical_transcription_job(
    MedicalTranscriptionJobName = job_name,
    Media = {'MediaFileUri': job_uri},
    LanguageCode = 'en-US',
    ContentIdentificationType = 'PHI',
    Specialty = 'PRIMARYCARE',
    Type = 'type', # Specify 'CONVERSATION' for a medical conversation. Specify
'DICTATION' for a medical dictation.
    OutputBucketName = 'DOC-EXAMPLE-BUCKET2'
)
while True:
    status =
    transcribe.get_medical_transcription_job(MedicalTranscriptionJobName=job_name)
    if status['MedicalTranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED',
'FAILED']:
        break
    print("Not ready yet...")
    time.sleep(5)
print(status)
```

The following example code shows the transcription results with patient PHI identified.

```
{
    "jobName": "transcription-job-name",
    "accountId": "account-id",
    "results": {
        "transcripts": [
            {
                "transcript": "The patient's name is Bertrand."
            }
        ],
        "items": [
            {
                "start_time": "0.0",
                "end_time": "0.37",
                "alternatives": [
                    {
                        "confidence": "0.9993",
                        "content": "The"
                    }
                ]
            }
        ]
    }
}
```

```
        ],
        "type": "pronunciation"
    },
    "start_time": "0.37",
    "end_time": "0.44",
    "alternatives": [
        {
            "confidence": "0.9981",
            "content": "patient's"
        }
    ],
    "type": "pronunciation"
},
{
    "start_time": "0.44",
    "end_time": "0.52",
    "alternatives": [
        {
            "confidence": "1.0",
            "content": "name"
        }
    ],
    "type": "pronunciation"
},
{
    "start_time": "0.52",
    "end_time": "0.92",
    "alternatives": [
        {
            "confidence": "1.0",
            "content": "is"
        }
    ],
    "type": "pronunciation"
},
{
    "start_time": "0.92",
    "end_time": "0.9989",
    "alternatives": [
        {
            "confidence": "1.0",
            "content": "Bertrand"
        }
    ],
    "type": "pronunciation"
},
{
    "alternatives": [
        {
            "confidence": "0.0",
            "content": "."
        }
    ],
    "type": "punctuation"
},
"entities": [
    {
        "content": "Bertrand",
        "category": "PHI*-Personal*",
        "startTime": 0.92,
        "endTime": 1.2,
        "confidence": 0.9989
    }
],
"status": "COMPLETED"
}
```

AWS CLI

To transcribe an audio file and identify PHI using a batch transcription job (AWS CLI)

- Run the following code.

```
aws transcribe start-medical-transcription-job \
--medical-transcription-job-name job-name\
```

```
--language-code en-US \
--media MediaFileUri="s3://your-S3-bucket/S3-prefix/your-filename.file-extension" \
--output-bucket-name DOC-EXAMPLE-BUCKET2 \
--specialty PRIMARYCARE \
--type type \ # Choose CONVERSATION to transcribe a medical conversation.
Choose DICTATION to transcribe a medical dictation.
--content-identification-type PHI
```

The following is the response from running the preceding CLI command.

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "job-name",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "en-US",
        "Media": {
            "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET1/example-audio-file.extension"
        },
        "StartTime": "2021-04-27T22:21:52.505000+00:00",
        "CreationTime": "2021-04-27T22:21:52.459000+00:00",
        "ContentIdentificationType": "PHI",
        "Specialty": "PRIMARYCARE",
        "Type": "type"
    }
}
```

Identifying PHI in a real-time stream

You can identify Personally Identifiable Health Information (PHI) in either HTTP/2 or WebSocket streams. When you activate PHI Identification, Amazon Transcribe Medical labels the PHI that it identifies in the transcription results. For information about the PHI that Amazon Transcribe Medical can identify, see [Identifying personal health information \(PHI\) in a transcription \(p. 256\)](#).

Identifying PHI in a dictation that is spoken into your microphone

To use the Amazon Transcribe Medical console to transcribe the speech picked up by your microphone and identify any PHI, choose **Dictation** as the audio input type, start the stream, and begin speaking into the microphone on your computer.

To identify PHI in a dictation using the Amazon Transcribe Medical console

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, choose **Real-time transcription**.
3. For **Audio input type**, choose **Dictation**.
4. For **Additional settings**, choose **PHI identification**.
5. Choose **Start streaming** and speak into the microphone.
6. Choose **Stop streaming** to end the dictation.

Identifying PHI in an HTTP/2 stream

To start an HTTP/2 stream with PHI Identification activated, use the [StartMedicalStreamTranscription \(p. 402\)](#) API and specify the following:

- For `LanguageCode`, specify the language code for the language spoken in the stream. For US English, specify `en-US`.
- For `MediaSampleHertz`, specify the sample rate of the audio.
- For `content-identification-type`, specify `PHI`.

Identifying PHI in a WebSocket stream

To start a WebSocket stream with PHI Identification activated, use the following format to create a pre-signed URL.

```
GET https://transcribestreaming.region.amazonaws.com:8443/medical-stream-transcription-
websocket?
language-code=languageCode
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=Signature Version 4 credential scope
&X-Amz-Date=date
&X-Amz-Expires=time in seconds until expiration
&X-Amz-Security-Token=security-token
&X-Amz-Signature=Signature Version 4 signature
&X-Amz-SignedHeaders=host
&media-encoding=mediaEncoding
&sample-rate=mediaSampleRateHertz
&session-id=sessionId
&enable-channel-identification=boolean
&specialty=medical-specialty
&content-identification-type=PHI
```

To start a stream with PHI Identification turned on, you must provide the following values:

- For `LanguageCode`, specify the language code for the language spoken in the stream. For US English, specify `en-US`.
- For `MediaSampleHertz`, specify the sample rate of the audio.
- For `content-identification-type`, specify `PHI`.

The following example is an example response to a streaming request with PHI Identification activated. For brevity, metadata has been removed.

```
{
    "TranscriptResultStream": {
        "TranscriptEvent": {
            "Transcript": {
                "Results": [
                    {
                        "Alternatives": [
                            {
                                "Transcript": "my name is john doe",
                                "Items": [
                                    {
                                        "Content": "my",
                                        "EndTime": 0.3799375,
                                        "StartTime": 0.0299375,
                                        "Type": "pronunciation"
                                    }
                                ]
                            }
                        ]
                    }
                ]
            }
        }
    }
}
```

Generating alternative transcriptions

When you use Amazon Transcribe Medical, you get the transcription that has the highest confidence level. However, you can configure Amazon Transcribe Medical to return additional transcriptions with lower confidence levels.

Use alternative transcriptions to see different interpretations of the transcribed audio. For example, in an application that enables a person to review the transcription, you can present the alternative transcriptions for the person to choose from.

You can generate alternative transcriptions with the Amazon Transcribe Medical console or the [StartMedicalTranscriptionJob](#) (p. 370) API.

Console

To generate additional alternative transcriptions in a transcription job (console)

To use the console to generate alternative transcriptions, you enable alternative results when you configure your job.

1. Sign in to the [Amazon Transcribe Medical console](#).
2. In the navigation pane, under Amazon Transcribe Medical, choose **Transcription jobs**.
3. Choose **Create job**.
4. On the **Specify job details** page, provide information about your transcription job.
5. Choose **Next**.
6. Enable **Alternative results**.
7. For **Maximum alternatives**, enter an integer value between 2 and 10, for the maximum number of alternative transcriptions you want in the output.
8. Choose **Create**.

API

To identify speakers in an audio file using a batch transcription job (API)

- For the [StartMedicalTranscriptionJob \(p. 370\)](#) API, specify the following.
 - a. For `MedicalTranscriptionJobName`, specify a name that is unique in your AWS account.
 - b. For `LanguageCode`, specify the language code that corresponds to the language spoken in your audio file and the language of your vocabulary filter.
 - c. In the `MediaFileUri` parameter of the `Media` object, specify the name of the audio file you want to transcribe.
 - d. For `Specialty`, specify the medical specialty of the clinician speaking in the audio file.
 - e. For `Type`, specify whether you're transcribing a medical conversation or a dictation.
 - f. For `OutputBucketName`, specify the Amazon Simple Storage Service (Amazon S3) bucket to store the transcription results.
 - g. For the `Settings` object, specify the following.
 - i. `ShowAlternatives = true`.
 - ii. `MaxAlternatives` - An integer between 2 and 10 to indicate the number of alternative transcriptions you want in the transcription output.

The following request uses the AWS SDK for Python (Boto3) to start a transcription job that generates up to two alternative transcriptions.

```
from __future__ import print_function
import time
import boto3
transcribe = boto3.client('transcribe')
job_name = "your-transcription-job-name"
job_uri = s3://DOC-EXAMPLE-BUCKET1/example-audio-file.extension
transcribe.start_medical_transcription_job(
    MedicalTranscriptionJobName=job_name,
    Media = {'MediaFileUri': job_uri},
    LanguageCode = 'en-US',
    Specialty = 'PRIMARYCARE',
```

```
Type = 'type', # Specify 'CONVERSATION' for a medical conversation. Specify 'DICTATION' for a medical dictation.  
        OutputBucketName = 'Amazon-S3-bucket-name-storing-your-transcription-results'  
,  
Settings = {'ShowAlternatives': True,  
           'MaxAlternatives': 2  
         }  
while True:  
    status = transcribe.get_medical_transcription_job(MedicalTranscriptionJobName=job_name)  
    if status['MedicalTranscriptionJob']['TranscriptionJobStatus'] in ['COMPLETED',  
    'FAILED']:br/>        break  
    print("Not ready yet...")  
    time.sleep(5)  
print(status)
```

AWS CLI

To transcribe an audio file of a conversation between a primary care clinician and a patient in an audio file and identify what each person said in the transcription output (AWS CLI)

- Run the following code.

```
aws transcribe start-transcription-job \  
--cli-input-json file://filepath/example-start-command.json
```

The following code shows the contents of example-start-command.json.

```
{  
    "MedicalTranscriptionJobName": "alternatives-conversation-medical-transcription-job",  
    "LanguageCode": "en-US",  
    "Specialty": "PRIMARYCARE",  
    "Type": "CONVERSATION",  
    "OutputBucketName": "DOC-EXAMPLE-BUCKET",  
    "Media": {  
        "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"  
    },  
    "Settings": {  
        "ShowAlternatives": true,  
        "MaxAlternatives": 2  
    }  
}
```

The following is the response from running the preceding CLI command.

```
{  
    "MedicalTranscriptionJob": {  
        "MedicalTranscriptionJobName": "alternatives-medical-transcription-job",  
        "TranscriptionJobStatus": "IN_PROGRESS",  
        "LanguageCode": "en-US",  
        "Media": {  
            "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
```

```
        },
        "StartTime": "2020-09-21T19:09:18.199000+00:00",
        "CreationTime": "2020-09-21T19:09:18.171000+00:00",
        "Settings": {
            "ShowAlternatives": true,
            "MaxAlternatives": 2
        },
        "Specialty": "PRIMARYCARE",
        "Type": "CONVERSATION"
    }
}
```

Guidelines and quotas

Supported Regions

For a list of AWS Regions where Amazon Transcribe Medical is available, see [Amazon Transcribe Endpoints and Quotas](#) in the *Amazon Web Services General Reference*.

Guidelines

For best results:

- Use a lossless format, such as FLAC or WAV, with PCM 16-bit encoding.
- Use a sample rate of 16,000 Hz or greater.

Amazon Transcribe Medical may store your content to continuously improve the quality of its analysis models. See the [Amazon Transcribe Medical FAQ](#) to learn more. To request that we delete content that may have been stored by Amazon Transcribe Medical, open a case with AWS Support.

Quotas

You can request a quota increase for the following resources.

Resource	Default
Number of concurrent batch transcription jobs	250
Number of StartMedicalStreamTranscription (p. 402) or Websocket requests	25
Total number of medical vocabularies per account	100
Number of pending medical vocabularies	10

The below API limits can also be increased upon request:

API	Maximum transactions per second
StartMedicalTranscriptionJob (p. 370)	25
StartMedicalStreamTranscription (p. 402)	25
GetMedicalTranscriptionJob (p. 323)	30
DeleteMedicalTranscriptionJob (p. 305)	5
ListMedicalTranscriptionJobs (p. 348)	5
CreateMedicalVocabulary (p. 287)	10
UpdateMedicalVocabulary (p. 392)	10
DeleteMedicalVocabulary (p. 307)	5
GetMedicalVocabulary (p. 326)	20
ListMedicalVocabularies (p. 351)	5

Note

For information about requesting a quota increase, see [AWS Service Quotas](#) in the *Amazon Web Services General Reference*.

The following quotas **cannot** be increased:

Description	Quotas
Maximum audio file length	14,400 seconds
Maximum audio file size	2 GB
Number of days that job records are retained	90
Number of channels for channel identification	2
Minimum audio file duration, in milliseconds (ms)	500

Amazon Transcribe Medical and interface VPC endpoints (AWS PrivateLink)

You can establish a private connection between your VPC and Amazon Transcribe Medical by creating an *interface VPC endpoint*. Interface endpoints are powered by [AWS PrivateLink](#), a technology that enables you to privately access Amazon Transcribe Medical APIs without an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to communicate with Amazon Transcribe Medical APIs. Traffic between your VPC and Amazon Transcribe Medical does not leave the Amazon network.

Each interface endpoint is represented by one or more [Elastic Network Interfaces](#) in your subnets.

For more information, see [Interface VPC endpoints \(AWS PrivateLink\)](#) in the *Amazon VPC User Guide*.

Considerations for Amazon Transcribe Medical VPC endpoints

Before you set up an interface VPC endpoint for Amazon Transcribe Medical, ensure that you review [Interface endpoint properties and limitations](#) in the *Amazon VPC User Guide*.

Amazon Transcribe Medical supports making calls to all of its API actions from your VPC.

Creating an interface VPC endpoint for Amazon Transcribe Medical

You can create a VPC endpoint for the Amazon Transcribe Medical service using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see [Creating an interface endpoint](#) in the *Amazon VPC User Guide*.

For batch transcription in Amazon Transcribe Medical, create a VPC endpoint using the following service name:

- com.amazonaws.*region*.transcribe

For streaming transcription in Amazon Transcribe Medical, create a VPC endpoint using the following service name:

- com.amazonaws.*region*.transcribestreaming

If you enable private DNS for the endpoint, you can make API requests to Amazon Transcribe Medical using its default DNS name for the Region, for example, transcribestreaming.us-east-2.amazonaws.com.

For more information, see [Accessing a service through an interface endpoint](#) in the *Amazon VPC User Guide*.

Creating a VPC endpoint policy for Amazon Transcribe Medical streaming

You can attach an endpoint policy to your VPC endpoint that controls access to Amazon Transcribe Medical. The policy specifies the following information:

- The principal that can perform actions.
- The actions that can be performed.
- The resources on which actions can be performed.

For more information, see [Controlling access to services with VPC endpoints](#) in the *Amazon VPC User Guide*.

Example: VPC endpoint policy for Amazon Transcribe Medical streaming transcription actions

The following is an example of an endpoint policy for streaming transcription in Amazon Transcribe Medical. When attached to an endpoint, this policy grants access to the listed Amazon Transcribe Medical actions for all principals on all resources.

```
{
```

```
"Statement": [
    {
        "Principal": "*",
        "Effect": "Allow",
        "Action": [
            "transcribe:StartMedicalStreamTranscription",
        ],
        "Resource": "*"
    }
]
```

Example: VPC endpoint policy for Amazon Transcribe Medical batch transcription actions

The following is an example of an endpoint policy for batch transcription in Amazon Transcribe Medical. When attached to an endpoint, this policy grants access to the listed Amazon Transcribe Medical actions for all principals on all resources.

```
{
    "Statement": [
        {
            "Principal": "*",
            "Effect": "Allow",
            "Action": [
                "transcribe:StartMedicalTranscriptionJob"
            ],
            "Resource": "*"
        }
    ]
}
```

Document history for Amazon Transcribe and Amazon Transcribe Medical

- **Latest documentation update:** 04 August 2021

The following table describes important changes in each release of Amazon Transcribe. For notification about updates to this documentation, you can subscribe to an RSS feed.

update-history-change	update-history-description	update-history-date
New feature	Amazon Transcribe now supports PII redaction and identification for streaming.	August 31, 2021
New languages	Amazon Transcribe now supports Afrikaans, Danish, Mandarin Chinese (Traditional), Thai, New Zealand English, and South African English.	August 26, 2021
New feature	Amazon Transcribe now supports resource tagging.	August 24, 2021
New feature	Amazon Transcribe now supports call analytics for batch transcription jobs.	August 4, 2021
New feature	Amazon Transcribe Medical now supports personal health information (PHI) identification in batch transcription jobs.	May 14, 2021
New feature	Amazon Transcribe now supports using custom vocabularies with batch custom language models.	May 12, 2021
New feature	Amazon Transcribe now supports partial result stabilization for streaming transcription.	May 11, 2021
New feature	Amazon Transcribe now supports Australian English, British English, Hindi, and US Spanish for custom language models.	March 19, 2021
New feature	Amazon Transcribe now supports Australian English,	March 19, 2021

	British English, Hindi, and US Spanish for custom language models.	
New feature	Amazon Transcribe Medical now supports personal health information identification.	January 29, 2021
New feature	Amazon Transcribe Medical now supports channel identification.	December 21, 2020
New feature	Amazon Transcribe Medical now supports the HTTP/2 streaming protocol.	November 24, 2020
New feature	Amazon Transcribe Medical now supports the neurology medical specialty for streaming audio transcription.	November 24, 2020
New feature	Amazon Transcribe Medical now supports the urology medical specialty for streaming audio transcription.	November 24, 2020
New feature	Amazon Transcribe Medical now supports the radiology medical specialty for streaming audio transcription.	November 24, 2020
New feature	Amazon Transcribe Medical now supports the cardiology medical specialty for streaming audio transcription.	November 24, 2020
New feature	Amazon Transcribe Medical now supports the oncology medical specialty for streaming audio transcription.	November 24, 2020
New feature	Amazon Transcribe now supports OGG/OPUS and FLAC codecs for streaming audio transcription.	November 24, 2020
Region expansion	Amazon Transcribe is now available in the Frankfurt (eu-central-1) and London (eu-west-2) regions.	November 4, 2020
New languages	Amazon Transcribe adds support for Italian and German for streaming audio transcription.	November 4, 2020
New feature	Amazon Transcribe adds support for interface VPC endpoints in batch transcription. For more information, see Amazon Transcribe and interface VPC endpoints (AWS PrivateLink) .	October 9, 2020

New feature	Amazon Transcribe adds support for channel identification in streaming. For more information, see Transcribing Multi-Channel Audio .	September 17, 2020
New feature	Amazon Transcribe adds support for automatic language identification in batch transcription. For more information, see Identifying the Language .	September 15, 2020
New feature	Amazon Transcribe adds support for speaker identification in streaming. For more information, see Filtering Streaming Transcriptions .	August 19, 2020
New feature	Amazon Transcribe adds support for custom language models. For more information, see Improving Transcription Accuracy with Custom Models .	August 5, 2020
New feature	Amazon Transcribe adds support for interface VPC endpoints in streaming. For more information, see Amazon Transcribe and interface VPC endpoints (AWS PrivateLink) .	June 26, 2020
New feature	Amazon Transcribe adds support for vocabulary filtering in streaming. For more information, see Filtering Streaming Transcriptions .	May 20, 2020
New feature	Amazon Transcribe Medical adds support for custom vocabularies in both batch processing and streaming. For more information, see Medical Custom Vocabularies .	April 29, 2020
New feature	Amazon Transcribe Medical adds support for batch processing of audio files. For more information, see Batch Transcription Overview .	April 1, 2020
New feature	Amazon Transcribe adds support for automatically redacting personally identifiable information. For more information, see Automatic Content Redaction .	February 26, 2020

New feature	Amazon Transcribe adds support for creating a vocabulary of words to filter from a transcription. For more information, see Vocabulary Filtering .	December 20, 2019
New feature	Amazon Transcribe adds support for queuing transcription jobs. For more information, see Job Queuing .	December 19, 2019
Region expansion	Amazon Transcribe is now available in the Asia Pacific (Tokyo) (ap-northeast-1) region.	November 21, 2019
New languages	Amazon Transcribe adds support for Gulf Arabic, Hebrew, Japanese, Malay, Swiss German, Telugu, and Turkish.	November 21, 2019
New feature	Amazon Transcribe adds support for alternative transcriptions. For more information, see Alternative Transcriptions .	November 20, 2019
New languages	Amazon Transcribe adds support for Dutch, Farsi, Indonesian, Irish English, Portuguese, Scottish English, Tamil, and Welsh English.	November 12, 2019
New language	Amazon Transcribe now supports streaming transcription for Australian English (en-AU).	October 25, 2019
Region expansion	Amazon Transcribe is now available in the China (Beijing) (cn-north-1) and China (Ningxia) (cn-northwest-1) regions.	October 9, 2019
New feature	Amazon Transcribe enables you to provide your own AWS Key Management Service key to encrypt your transcription output files. For more information, see the OutputEncryptionKMSKeyId parameter of the StartStreamTranscription API.	September 24, 2019
New languages	Amazon Transcribe adds support for Chinese (Mandarin), Simplified, Mainland China and Russian.	August 23, 2019

New feature	Amazon Transcribe adds support for streaming audio transcription using the WebSocket protocol. For more information, see Streaming Transcription .	July 19, 2019
New feature	AWS CloudTrail now records events for the StartStreamTranscription API.	July 19, 2019
Region expansion	Amazon Transcribe is now available in the US West (N. California) (us-west-1) region.	June 27, 2019
New language	Amazon Transcribe adds support for Modern Standard Arabic.	May 28, 2019
New feature	Amazon Transcribe now transcribes numeric words into numbers for US English. For example, "forty-two" is transcribed as "42". For more information, see Transcribing Numbers .	May 23, 2019
New language	Amazon Transcribe adds support for Hindi and Indian English.	May 15, 2019
New SDK	The AWS SDK for C++ now supports Amazon Transcribe.	May 8, 2019
New language	Amazon Transcribe adds support for Spanish.	April 19, 2019
Region expansion	Amazon Transcribe is now available in the EU (Frankfurt) (eu-central-1) and Asia Pacific (Seoul) (ap-northeast-2) regions.	April 18, 2019
New language	Amazon Transcribe adds support for streaming transcription in British English, French, and Canadian French.	April 5, 2019
New feature	The AWS SDK for Ruby V3 now supports Amazon Transcribe	March 25, 2019
New feature	Amazon Transcribe now enables you to create custom vocabularies, lists of specific words that you want Amazon Transcribe to recognize in your audio input. For more information, see Custom Vocabularies .	March 25, 2019
New languages	Amazon Transcribe adds support for German and Korean.	March 22, 2019

Region expansion	Amazon Transcribe is now available in the South America (Sao Paulo) (sa-east-1) region.	February 7, 2019
New language	Amazon Transcribe now supports streaming transcription for US Spanish (es-US).	February 7, 2019
Region expansion	Amazon Transcribe is now available in the Asia Pacific (Mumbai) (ap-south-1), Asia Pacific (Singapore) (ap-southeast-1), EU (London) (eu-west-2), and EU (Paris) (eu-west3) regions.	January 24, 2019
New languages	Amazon Transcribe adds support for French, Italian, and Brazilian Portuguese.	December 20, 2018
New feature	Amazon Transcribe now supports transcription of audio streams. For more information, see Streaming Transcription .	November 19, 2018
New languages	Amazon Transcribe adds support for Australian English, British English, and Canadian French.	November 15, 2018
Region expansion	Amazon Transcribe is now available in Canada (Central) (ca-central-1) and Asia Pacific (Sydney) (ap-southeast-2).	July 17, 2018
New feature	You can now specify your own location to store the output from a transcription job. For more information, see the TranscriptionJobSummary data type.	July 11, 2018
New feature	Added AWS CloudTrail and Amazon CloudWatch Events integration. For more information, see Monitoring Amazon Transcribe .	June 28, 2018
New feature (p. 269)	Amazon Transcribe adds support for custom vocabularies. For more information, see Create a Custom Vocabulary .	April 4, 2018
New guide (p. 269)	This is the first release of the <i>Amazon Transcribe Developer Guide</i> .	November 29, 2017

API Reference

This section contains the API Reference documentation.

Topics

- [Actions \(p. 275\)](#)
- [Data Types \(p. 414\)](#)

Actions

The following actions are supported by Amazon Transcribe Service:

- [CreateCallAnalyticsCategory \(p. 278\)](#)
- [CreateLanguageModel \(p. 283\)](#)
- [CreateMedicalVocabulary \(p. 287\)](#)
- [CreateVocabulary \(p. 291\)](#)
- [CreateVocabularyFilter \(p. 295\)](#)
- [DeleteCallAnalyticsCategory \(p. 299\)](#)
- [DeleteCallAnalyticsJob \(p. 301\)](#)
- [DeleteLanguageModel \(p. 303\)](#)
- [DeleteMedicalTranscriptionJob \(p. 305\)](#)
- [DeleteMedicalVocabulary \(p. 307\)](#)
- [DeleteTranscriptionJob \(p. 309\)](#)
- [DeleteVocabulary \(p. 311\)](#)
- [DeleteVocabularyFilter \(p. 313\)](#)
- [DescribeLanguageModel \(p. 315\)](#)
- [GetCallAnalyticsCategory \(p. 317\)](#)
- [GetCallAnalyticsJob \(p. 320\)](#)
- [GetMedicalTranscriptionJob \(p. 323\)](#)
- [GetMedicalVocabulary \(p. 326\)](#)
- [GetTranscriptionJob \(p. 329\)](#)
- [GetVocabulary \(p. 332\)](#)
- [GetVocabularyFilter \(p. 335\)](#)
- [ListCallAnalyticsCategories \(p. 338\)](#)
- [ListCallAnalyticsJobs \(p. 342\)](#)
- [ListLanguageModels \(p. 345\)](#)
- [ListMedicalTranscriptionJobs \(p. 348\)](#)
- [ListMedicalVocabularies \(p. 351\)](#)
- [ListTagsForResource \(p. 354\)](#)
- [ListTranscriptionJobs \(p. 356\)](#)
- [ListVocabularies \(p. 359\)](#)
- [ListVocabularyFilters \(p. 362\)](#)
- [StartCallAnalyticsJob \(p. 365\)](#)

- [StartMedicalTranscriptionJob \(p. 370\)](#)
- [StartTranscriptionJob \(p. 376\)](#)
- [TagResource \(p. 383\)](#)
- [UntagResource \(p. 385\)](#)
- [UpdateCallAnalyticsCategory \(p. 387\)](#)
- [UpdateMedicalVocabulary \(p. 392\)](#)
- [UpdateVocabulary \(p. 395\)](#)
- [UpdateVocabularyFilter \(p. 399\)](#)

The following actions are supported by Amazon Transcribe Streaming Service:

- [StartMedicalStreamTranscription \(p. 402\)](#)
- [StartStreamTranscription \(p. 408\)](#)

Amazon Transcribe Service

The following actions are supported by Amazon Transcribe Service:

- [CreateCallAnalyticsCategory \(p. 278\)](#)
- [CreateLanguageModel \(p. 283\)](#)
- [CreateMedicalVocabulary \(p. 287\)](#)
- [CreateVocabulary \(p. 291\)](#)
- [CreateVocabularyFilter \(p. 295\)](#)
- [DeleteCallAnalyticsCategory \(p. 299\)](#)
- [DeleteCallAnalyticsJob \(p. 301\)](#)
- [DeleteLanguageModel \(p. 303\)](#)
- [DeleteMedicalTranscriptionJob \(p. 305\)](#)
- [DeleteMedicalVocabulary \(p. 307\)](#)
- [DeleteTranscriptionJob \(p. 309\)](#)
- [DeleteVocabulary \(p. 311\)](#)
- [DeleteVocabularyFilter \(p. 313\)](#)
- [DescribeLanguageModel \(p. 315\)](#)
- [GetCallAnalyticsCategory \(p. 317\)](#)
- [GetCallAnalyticsJob \(p. 320\)](#)
- [GetMedicalTranscriptionJob \(p. 323\)](#)
- [GetMedicalVocabulary \(p. 326\)](#)
- [GetTranscriptionJob \(p. 329\)](#)
- [GetVocabulary \(p. 332\)](#)
- [GetVocabularyFilter \(p. 335\)](#)
- [ListCallAnalyticsCategories \(p. 338\)](#)
- [ListCallAnalyticsJobs \(p. 342\)](#)
- [ListLanguageModels \(p. 345\)](#)
- [ListMedicalTranscriptionJobs \(p. 348\)](#)
- [ListMedicalVocabularies \(p. 351\)](#)
- [ListTagsForResource \(p. 354\)](#)
- [ListTranscriptionJobs \(p. 356\)](#)

- [ListVocabularies \(p. 359\)](#)
- [ListVocabularyFilters \(p. 362\)](#)
- [StartCallAnalyticsJob \(p. 365\)](#)
- [StartMedicalTranscriptionJob \(p. 370\)](#)
- [StartTranscriptionJob \(p. 376\)](#)
- [TagResource \(p. 383\)](#)
- [UntagResource \(p. 385\)](#)
- [UpdateCallAnalyticsCategory \(p. 387\)](#)
- [UpdateMedicalVocabulary \(p. 392\)](#)
- [UpdateVocabulary \(p. 395\)](#)
- [UpdateVocabularyFilter \(p. 399\)](#)

CreateCallAnalyticsCategory

Service: Amazon Transcribe Service

Creates an analytics category. Amazon Transcribe applies the conditions specified by your analytics categories to your call analytics jobs. For each analytics category, you specify one or more rules. For example, you can specify a rule that the customer sentiment was neutral or negative within that category. If you start a call analytics job, Amazon Transcribe applies the category to the analytics job that you've specified.

Request Syntax

```
{  
    "CategoryName": "string",  
    "Rules": [  
        {  
            "InterruptionFilter": {  
                "AbsoluteTimeRange": {  
                    "EndTime": number,  
                    "First": number,  
                    "Last": number,  
                    "StartTime": number  
                },  
                "Negate": boolean,  
                "ParticipantRole": "string",  
                "RelativeTimeRange": {  
                    "EndPercentage": number,  
                    "First": number,  
                    "Last": number,  
                    "StartPercentage": number  
                },  
                "Threshold": number  
            },  
            "NonTalkTimeFilter": {  
                "AbsoluteTimeRange": {  
                    "EndTime": number,  
                    "First": number,  
                    "Last": number,  
                    "StartTime": number  
                },  
                "Negate": boolean,  
                "RelativeTimeRange": {  
                    "EndPercentage": number,  
                    "First": number,  
                    "Last": number,  
                    "StartPercentage": number  
                },  
                "Threshold": number  
            },  
            "SentimentFilter": {  
                "AbsoluteTimeRange": {  
                    "EndTime": number,  
                    "First": number,  
                    "Last": number,  
                    "StartTime": number  
                },  
                "Negate": boolean,  
                "ParticipantRole": "string",  
                "RelativeTimeRange": {  
                    "EndPercentage": number,  
                    "First": number,  
                    "Last": number,  
                    "StartPercentage": number  
                },  
            },  
        }  
    ]  
}
```

```

        "Sentiments": [ "string" ]
    },
    "TranscriptFilter": {
        "AbsoluteTimeRange": {
            "EndTime": number,
            "First": number,
            "Last": number,
            "StartTime": number
        },
        "Negate": boolean,
        "ParticipantRole": "string",
        "RelativeTimeRange": {
            "EndPercentage": number,
            "First": number,
            "Last": number,
            "StartPercentage": number
        },
        "Targets": [ "string" ],
        "TranscriptFilterType": "string"
    }
}
]
}

```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

CategoryName (p. 278)

The name that you choose for your category when you create it.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Rules (p. 278)

To create a category, you must specify between 1 and 20 rules. For each rule, you specify a filter to be applied to the attributes of the call. For example, you can specify a sentiment filter to detect if the customer's sentiment was negative or neutral.

Type: Array of [Rule \(p. 450\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Required: Yes

Response Syntax

```
{
    "CategoryProperties": {
        "CategoryName": "string",
        "CreateTime": number,
        "LastUpdateTime": number,
    }
}
```

```
"Rules": [
  {
    "InterruptionFilter": {
      "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
      },
      "Negate": boolean,
      "ParticipantRole": "string",
      "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
      },
      "Threshold": number
    },
    "NonTalkTimeFilter": {
      "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
      },
      "Negate": boolean,
      "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
      },
      "Threshold": number
    },
    "SentimentFilter": {
      "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
      },
      "Negate": boolean,
      "ParticipantRole": "string",
      "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
      },
      "Sentiments": [ "string" ]
    },
    "TranscriptFilter": {
      "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
      },
      "Negate": boolean,
      "ParticipantRole": "string",
      "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
      }
    }
  }
]
```

```
        },
        "Targets": [ "string" ],
        "TranscriptFilterType": "string"
    }
}
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[CategoryProperties \(p. 279\)](#)

The rules and associated metadata used to create a category.

Type: [CategoryProperties \(p. 427\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)

- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateLanguageModel

Service: Amazon Transcribe Service

Creates a new custom language model. Use Amazon S3 prefixes to provide the location of your input files. The time it takes to create your model depends on the size of your training data.

Request Syntax

```
{  
    "BaseModelName": "string",  
    "InputDataConfig": {  
        "DataAccessRoleArn": "string",  
        "S3Uri": "string",  
        "TuningDataS3Uri": "string"  
    },  
    "LanguageCode": "string",  
    "ModelName": "string",  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ]  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

BaseModelName (p. 283)

The Amazon Transcribe standard language model, or base model used to create your custom language model.

If you want to use your custom language model to transcribe audio with a sample rate of 16,000 Hz or greater, choose `Wideband`.

If you want to use your custom language model to transcribe audio with a sample rate that is less than 16,000 Hz, choose `Narrowband`.

Type: String

Valid Values: `NarrowBand` | `WideBand`

Required: Yes

InputDataConfig (p. 283)

Contains the data access role and the Amazon S3 prefixes to read the required input files to create a custom language model.

Type: [InputDataConfig](#) (p. 430) object

Required: Yes

LanguageCode (p. 283)

The language of the input text you're using to train your custom language model.

Type: String

Valid Values: en-US | hi-IN | es-US | en-GB | en-AU

Required: Yes

[ModelName \(p. 283\)](#)

The name you choose for your custom language model when you create it.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

[Tags \(p. 283\)](#)

Adds one or more tags, each in the form of a key:value pair, to a new language model at the time you create this new model.

Type: Array of [Tag \(p. 455\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Required: No

Response Syntax

```
{  
    "BaseModelName": "string",  
    "InputDataConfig": {  
        "DataAccessRoleArn": "string",  
        "S3Uri": "string",  
        "TuningDataS3Uri": "string"  
    },  
    "LanguageCode": "string",  
    "ModelName": "string",  
    "ModelStatus": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[BaseModelName \(p. 284\)](#)

The Amazon Transcribe standard language model, or base model you've used to create a custom language model.

Type: String

Valid Values: NarrowBand | WideBand

[InputDataConfig \(p. 284\)](#)

The data access role and Amazon S3 prefixes you've chosen to create your custom language model.

Type: [InputDataConfig \(p. 430\)](#) object

[LanguageCode \(p. 284\)](#)

The language code of the text you've used to create a custom language model.

Type: String

Valid Values: en-US | hi-IN | es-US | en-GB | en-AU

[ModelName \(p. 284\)](#)

The name you've chosen for your custom language model.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

[ModelStatus \(p. 284\)](#)

The status of the custom language model. When the status is COMPLETED the model is ready to use.

Type: String

Valid Values: IN_PROGRESS | FAILED | COMPLETED

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception Message field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateMedicalVocabulary

Service: Amazon Transcribe Service

Creates a new custom vocabulary that you can use to modify how Amazon Transcribe Medical transcribes your audio file.

Request Syntax

```
{  
    "LanguageCode": "string",  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ],  
    "VocabularyFileUri": "string",  
    "VocabularyName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

[LanguageCode \(p. 287\)](#)

The language code for the language used for the entries in your custom vocabulary. The language code of your custom vocabulary must match the language code of your transcription job. US English (en-US) is the only language code available for Amazon Transcribe Medical.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: Yes

[Tags \(p. 287\)](#)

Adds one or more tags, each in the form of a key:value pair, to a new medical vocabulary at the time you create this new vocabulary.

Type: Array of [Tag \(p. 455\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Required: No

[VocabularyFileUri \(p. 287\)](#)

The location in Amazon S3 of the text file you use to define your custom vocabulary. The URI must be in the same AWS Region as the resource that you're calling. Enter information about your `VocabularyFileUri` in the following format:

`https://s3.<aws-region>.amazonaws.com/<bucket-name>/<keyprefix>/<objectkey>`

The following is an example URI for a vocabulary file that is stored in Amazon S3:

`https://s3.us-east-1.amazonaws.com/AWSDOC-EXAMPLE-BUCKET/vocab.txt`

For more information about Amazon S3 object names, see [Object Keys](#) in the *Amazon S3 Developer Guide*.

For more information about custom vocabularies, see [Medical Custom Vocabularies](#).

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (`s3://|http(s*)://`).+

Required: Yes

[VocabularyName \(p. 287\)](#)

The name of the custom vocabulary. This case-sensitive name must be unique within an AWS account. If you try to create a vocabulary with the same name as a previous vocabulary, you get a `ConflictException` error.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^[0-9a-zA-Z._-]+`

Required: Yes

Response Syntax

```
{  
    "FailureReason": "string",  
    "LanguageCode": "string",  
    "LastModifiedTime": number,  
    "VocabularyName": "string",  
    "VocabularyState": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[FailureReason \(p. 288\)](#)

If the `VocabularyState` field is `FAILED`, this field contains information about why the job failed.

Type: String

[LanguageCode \(p. 288\)](#)

The language code for the entries in your custom vocabulary. US English (en-US) is the only valid language code for Amazon Transcribe Medical.

Type: String

Valid Values: `af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-`

CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

[LastModifiedTime \(p. 288\)](#)

The date and time that you created the vocabulary.

Type: Timestamp

[VocabularyName \(p. 288\)](#)

The name of the vocabulary. The name must be unique within an AWS account and is case sensitive.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

[VocabularyState \(p. 288\)](#)

The processing state of your custom vocabulary in Amazon Transcribe Medical. If the state is READY, you can use the vocabulary in a `StartMedicalTranscriptionJob` request.

Type: String

Valid Values: PENDING | READY | FAILED

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateVocabulary

Service: Amazon Transcribe Service

Creates a new custom vocabulary that you can use to change the way Amazon Transcribe handles transcription of an audio file.

Request Syntax

```
{  
    "LanguageCode": "string",  
    "Phrases": [ "string" ],  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ],  
    "VocabularyFileUri": "string",  
    "VocabularyName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

LanguageCode ([p. 291](#))

The language code of the vocabulary entries. For a list of languages and their corresponding language codes, see [What is Amazon Transcribe? \(p. 1\)](#).

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: Yes

Phrases ([p. 291](#))

An array of strings that contains the vocabulary entries.

Type: Array of strings

Length Constraints: Minimum length of 0. Maximum length of 256.

Pattern: .+

Required: No

Tags ([p. 291](#))

Adds one or more tags, each in the form of a key:value pair, to a new Amazon Transcribe vocabulary at the time you create this new vocabulary.

Type: Array of [Tag \(p. 455\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Required: No

[VocabularyFileUri \(p. 291\)](#)

The S3 location of the text file that contains the definition of the custom vocabulary. The URI must be in the same region as the API endpoint that you are calling. The general form is:

```
https://s3.<AWS-region>.amazonaws.com/<AWSDOC-EXAMPLE-BUCKET>/<keyprefix>/<objectkey>
```

For example:

```
https://s3.us-east-1.amazonaws.com/AWSDOC-EXAMPLE-BUCKET/vocab.txt
```

For more information about S3 object names, see [Object Keys](#) in the *Amazon S3 Developer Guide*.

For more information about custom vocabularies, see [Custom vocabularies](#).

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (s3://|http(s*)://).+

Required: No

[VocabularyName \(p. 291\)](#)

The name of the vocabulary. The name must be unique within an AWS account. The name is case sensitive. If you try to create a vocabulary with the same name as a previous vocabulary you will receive a `ConflictException` error.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Syntax

```
{  
    "FailureReason": "string",  
    "LanguageCode": "string",  
    "LastModifiedTime": number,  
    "VocabularyName": "string",  
    "VocabularyState": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[FailureReason \(p. 292\)](#)

If the `VocabularyState` field is `FAILED`, this field contains information about why the job failed.

Type: String

[LanguageCode \(p. 292\)](#)

The language code of the vocabulary entries.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

[LastModifiedTime \(p. 292\)](#)

The date and time that the vocabulary was created.

Type: Timestamp

[VocabularyName \(p. 292\)](#)

The name of the vocabulary.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

[VocabularyState \(p. 292\)](#)

The processing state of the vocabulary. When the `VocabularyState` field contains `READY` the vocabulary is ready to be used in a `StartTranscriptionJob` request.

Type: String

Valid Values: PENDING | READY | FAILED

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

CreateVocabularyFilter

Service: Amazon Transcribe Service

Creates a new vocabulary filter that you can use to filter words, such as profane words, from the output of a transcription job.

Request Syntax

```
{  
    "LanguageCode": "string",  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ],  
    "VocabularyFilterFileUri": "string",  
    "VocabularyFilterName": "string",  
    "Words": [ "string" ]  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

LanguageCode (p. 295)

The language code of the words in the vocabulary filter. All words in the filter must be in the same language. The vocabulary filter can only be used with transcription jobs in the specified language.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: Yes

Tags (p. 295)

Adds one or more tags, each in the form of a key:value pair, to a new Amazon Transcribe vocabulary filter at the time you create this new vocabulary filter.

Type: Array of [Tag \(p. 455\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Required: No

VocabularyFilterFileUri (p. 295)

The Amazon S3 location of a text file used as input to create the vocabulary filter. Only use characters from the character set defined for custom vocabularies. For a list of character sets, see [Character Sets for Custom Vocabularies](#).

The specified file must be less than 50 KB of UTF-8 characters.

If you provide the location of a list of words in the `VocabularyFilterFileUri` parameter, you can't use the `Words` parameter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (s3://|http(s*)://).+

Required: No

[VocabularyFilterName \(p. 295\)](#)

The vocabulary filter name. The name must be unique within the account that contains it. If you try to create a vocabulary filter with the same name as another vocabulary filter, you get a `ConflictException` error.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

[Words \(p. 295\)](#)

The words to use in the vocabulary filter. Only use characters from the character set defined for custom vocabularies. For a list of character sets, see [Character Sets for Custom Vocabularies](#).

If you provide a list of words in the `Words` parameter, you can't use the `VocabularyFilterFileUri` parameter.

Type: Array of strings

Array Members: Minimum number of 1 item.

Length Constraints: Minimum length of 1. Maximum length of 256.

Required: No

Response Syntax

```
{  
    "LanguageCode": "string",  
    "LastModifiedTime": number,  
    "VocabularyFilterName": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[LanguageCode \(p. 296\)](#)

The language code of the words in the collection.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

[LastModifiedTime \(p. 296\)](#)

The date and time that the vocabulary filter was modified.

Type: Timestamp

[VocabularyFilterName \(p. 296\)](#)

The name of the vocabulary filter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)

- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteCallAnalyticsCategory

Service: Amazon Transcribe Service

Deletes a call analytics category using its name.

Request Syntax

```
{  
    "CategoryName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

CategoryName (p. 299)

The name of the call analytics category that you're choosing to delete. The value is case sensitive.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteCallAnalyticsJob

Service: Amazon Transcribe Service

Deletes a call analytics job using its name.

Request Syntax

```
{  
    "CallAnalyticsJobName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

CallAnalyticsJobName (p. 301)

The name of the call analytics job you want to delete.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteLanguageModel

Service: Amazon Transcribe Service

Deletes a custom language model using its name.

Request Syntax

```
{  
    "ModelName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

ModelName (p. 303)

The name of the model you're choosing to delete.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteMedicalTranscriptionJob

Service: Amazon Transcribe Service

Deletes a transcription job generated by Amazon Transcribe Medical and any related information.

Request Syntax

```
{  
    "MedicalTranscriptionJobName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

MedicalTranscriptionJobName ([p. 305](#))

The name you provide to the `DeleteMedicalTranscriptionJob` object to delete a transcription job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteMedicalVocabulary

Service: Amazon Transcribe Service

Deletes a vocabulary from Amazon Transcribe Medical.

Request Syntax

```
{  
    "VocabularyName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

VocabularyName ([p. 307](#))

The name of the vocabulary that you want to delete.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteTranscriptionJob

Service: Amazon Transcribe Service

Deletes a previously submitted transcription job along with any other generated results such as the transcription, models, and so on.

Request Syntax

```
{  
    "TranscriptionJobName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

TranscriptionJobName (p. 309)

The name of the transcription job to be deleted.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception Message field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteVocabulary

Service: Amazon Transcribe Service

Deletes a vocabulary from Amazon Transcribe.

Request Syntax

```
{  
    "VocabularyName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

VocabularyName ([p. 311](#))

The name of the vocabulary to delete.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DeleteVocabularyFilter

Service: Amazon Transcribe Service

Removes a vocabulary filter.

Request Syntax

```
{  
    "VocabularyFilterName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

VocabularyFilterName (p. 313)

The name of the vocabulary filter to remove.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

DescribeLanguageModel

Service: Amazon Transcribe Service

Gets information about a single custom language model. Use this information to see details about the language model in your AWS account. You can also see whether the base language model used to create your custom language model has been updated. If Amazon Transcribe has updated the base model, you can create a new custom language model using the updated base model. If the language model wasn't created, you can use this operation to understand why Amazon Transcribe couldn't create it.

Request Syntax

```
{  
    "ModelName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

ModelName (p. 315)

The name of the custom language model you submit to get more information.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Syntax

```
{  
    "LanguageModel": {  
        "BaseModelName": "string",  
        "CreateTime": number,  
        "FailureReason": "string",  
        "InputDataConfig": {  
            "DataAccessRoleArn": "string",  
            "S3Uri": "string",  
            "TuningDataS3Uri": "string"  
        },  
        "LanguageCode": "string",  
        "LastModifiedTime": number,  
        "ModelName": "string",  
        "ModelStatus": "string",  
        "UpgradeAvailability": boolean  
    }  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[LanguageModel \(p. 315\)](#)

The name of the custom language model you requested more information about.

Type: [LanguageModel \(p. 434\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

GetCallAnalyticsCategory

Service: Amazon Transcribe Service

Retrieves information about a call analytics category.

Request Syntax

```
{  
    "CategoryName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

CategoryName (p. 317)

The name of the category you want information about. This value is case sensitive.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Syntax

```
{  
    "CategoryProperties": {  
        "CategoryName": "string",  
        "CreateTime": number,  
        "LastUpdateTime": number,  
        "Rules": [  
            {  
                "InterruptionFilter": {  
                    "AbsoluteTimeRange": {  
                        "EndTime": number,  
                        "First": number,  
                        "Last": number,  
                        "StartTime": number  
                    },  
                    "Negate": boolean,  
                    "ParticipantRole": "string",  
                    "RelativeTimeRange": {  
                        "EndPercentage": number,  
                        "First": number,  
                        "Last": number,  
                        "StartPercentage": number  
                    },  
                    "Threshold": number  
                },  
                "NonTalkTimeFilter": {  
                    "AbsoluteTimeRange": {  
                        "EndTime": number,  
                        "First": number,  
                        "Last": number,  
                        "StartTime": number  
                    }  
                }  
            }  
        ]  
    }  
}
```

```
        "First": number,
        "Last": number,
        "StartTime": number
    },
    "Negate": boolean,
    "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
    },
    "Threshold": number
},
"SentimentFilter": {
    "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
    },
    "Negate": boolean,
    "ParticipantRole": "string",
    "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
    },
    "Sentiments": [ "string" ]
},
"TranscriptFilter": {
    "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
    },
    "Negate": boolean,
    "ParticipantRole": "string",
    "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
    },
    "Targets": [ "string" ],
    "TranscriptFilterType": "string"
}
}
]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[CategoryProperties \(p. 317\)](#)

The rules you've defined for a category.

Type: [CategoryProperties \(p. 427\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

GetCallAnalyticsJob

Service: Amazon Transcribe Service

Returns information about a call analytics job. To see the status of the job, check the `CallAnalyticsJobStatus` field. If the status is `COMPLETED`, the job is finished and you can find the results at the location specified in the `TranscriptFileUri` field. If you enable personally identifiable information (PII) redaction, the redacted transcript appears in the `RedactedTranscriptFileUri` field.

Request Syntax

```
{  
    "CallAnalyticsJobName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

`CallAnalyticsJobName` (p. 320)

The name of the analytics job you want information about. This value is case sensitive.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^ [0-9a-zA-Z._-] +`

Required: Yes

Response Syntax

```
{  
    "CallAnalyticsJob": {  
        "CallAnalyticsJobName": "string",  
        "CallAnalyticsJobStatus": "string",  
        "ChannelDefinitions": [  
            {  
                "ChannelId": number,  
                "ParticipantRole": "string"  
            }  
        ],  
        "CompletionTime": number,  
        "CreationTime": number,  
        "DataAccessRoleArn": "string",  
        "FailureReason": "string",  
        "IdentifiedLanguageScore": number,  
        "LanguageCode": "string",  
        "Media": {  
            "MediaFileUri": "string",  
            "RedactedMediaFileUri": "string"  
        },  
        "MediaFormat": "string",  
        "MediaSampleRateHertz": number,  
        "Settings": {  
            "ContentRedaction": {  
                "RedactionOutput": "string",  
            }  
        }  
    }  
}
```

```
        "RedactionType": "string"
    },
    "LanguageModelName": "string",
    "LanguageOptions": [ "string" ],
    "VocabularyFilterMethod": "string",
    "VocabularyFilterName": "string",
    "VocabularyName": "string"
},
"StartTime": number,
"Transcript": {
    "RedactedTranscriptFileUri": "string",
    "TranscriptFileUri": "string"
}
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[CallAnalyticsJob \(p. 320\)](#)

An object that contains the results of your call analytics job.

Type: [CallAnalyticsJob \(p. 419\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

GetMedicalTranscriptionJob

Service: Amazon Transcribe Service

Returns information about a transcription job from Amazon Transcribe Medical. To see the status of the job, check the `TranscriptionJobStatus` field. If the status is `COMPLETED`, the job is finished. You find the results of the completed job in the `TranscriptFileUri` field.

Request Syntax

```
{  
    "MedicalTranscriptionJobName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

MedicalTranscriptionJobName (p. 323)

The name of the medical transcription job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Syntax

```
{  
    "MedicalTranscriptionJob": {  
        "CompletionTime": number,  
        "ContentIdentificationType": "string",  
        "CreationTime": number,  
        "FailureReason": "string",  
        "LanguageCode": "string",  
        "Media": {  
            "MediaFileUri": "string",  
            "RedactedMediaFileUri": "string"  
        },  
        "MediaFormat": "string",  
        "MediaSampleRateHertz": number,  
        "MedicalTranscriptionJobName": "string",  
        "Settings": {  
            "ChannelIdentification": boolean,  
            "MaxAlternatives": number,  
            "MaxSpeakerLabels": number,  
            "ShowAlternatives": boolean,  
            "ShowSpeakerLabels": boolean,  
            "VocabularyName": "string"  
        },  
        "Specialty": "string",  
        "StartTime": number,  
        "Tags": [  
            {  
                "TagKey": "string",  
                "TagValue": "string"  
            }  
        ]  
    }  
}
```

```
        "Key": "string",
        "Value": "string"
    },
    "Transcript": {
        "TranscriptFileUri": "string"
    },
    "TranscriptionJobStatus": "string",
    "Type": "string"
}
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[MedicalTranscriptionJob \(p. 323\)](#)

An object that contains the results of the medical transcription job.

Type: [MedicalTranscriptionJob \(p. 438\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

GetMedicalVocabulary

Service: Amazon Transcribe Service

Retrieves information about a medical vocabulary.

Request Syntax

```
{  
    "VocabularyName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

VocabularyName (p. 326)

The name of the vocabulary that you want information about. The value is case sensitive.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Syntax

```
{  
    "DownloadUri": "string",  
    "FailureReason": "string",  
    "LanguageCode": "string",  
    "LastModifiedTime": number,  
    "VocabularyName": "string",  
    "VocabularyState": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DownloadUri (p. 326)

The location in Amazon S3 where the vocabulary is stored. Use this URI to get the contents of the vocabulary. You can download your vocabulary from the URI for a limited time.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (s3://|http(s*)://).+

[FailureReason \(p. 326\)](#)

If the `VocabularyState` is `FAILED`, this field contains information about why the job failed.

Type: String

[LanguageCode \(p. 326\)](#)

The valid language code for your vocabulary entries.

Type: String

Valid Values: `af-ZA` | `ar-AE` | `ar-SA` | `cy-GB` | `da-DK` | `de-CH` | `de-DE` | `en-AB` | `en-AU` | `en-GB` | `en-IE` | `en-IN` | `en-US` | `en-WL` | `es-ES` | `es-US` | `fa-IR` | `fr-CA` | `fr-FR` | `ga-IE` | `gd-GB` | `he-IL` | `hi-IN` | `id-ID` | `it-IT` | `ja-JP` | `ko-KR` | `ms-MY` | `nl-NL` | `pt-BR` | `pt-PT` | `ru-RU` | `ta-IN` | `te-IN` | `tr-TR` | `zh-CN` | `zh-TW` | `th-TH` | `en-ZA` | `en-NZ`

[LastModifiedTime \(p. 326\)](#)

The date and time that the vocabulary was last modified with a text file different from the one that was previously used.

Type: Timestamp

[VocabularyName \(p. 326\)](#)

The name of the vocabulary returned by Amazon Transcribe Medical.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^ [0-9a-zA-Z._-]+`

[VocabularyState \(p. 326\)](#)

The processing state of the vocabulary. If the `VocabularyState` is `READY` then you can use it in the `StartMedicalTranscriptionJob` operation.

Type: String

Valid Values: `PENDING` | `READY` | `FAILED`

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

GetTranscriptionJob

Service: Amazon Transcribe Service

Returns information about a transcription job. To see the status of the job, check the `TranscriptionJobStatus` field. If the status is `COMPLETED`, the job is finished and you can find the results at the location specified in the `TranscriptFileUri` field. If you enable content redaction, the redacted transcript appears in `RedactedTranscriptFileUri`.

Request Syntax

```
{  
    "TranscriptionJobName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

[TranscriptionJobName \(p. 329\)](#)

The name of the job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^[0-9a-zA-Z._-]+`

Required: Yes

Response Syntax

```
{  
    "TranscriptionJob": {  
        "CompletionTime": number,  
        "ContentRedaction": {  
            "RedactionOutput": "string",  
            "RedactionType": "string"  
        },  
        "CreationTime": number,  
        "FailureReason": "string",  
        "IdentifiedLanguageScore": number,  
        "IdentifyLanguage": boolean,  
        "JobExecutionSettings": {  
            "AllowDeferredExecution": boolean,  
            "DataAccessRoleArn": "string"  
        },  
        "LanguageCode": "string",  
        "LanguageOptions": [ "string" ],  
        "Media": {  
            "MediaFileUri": "string",  
            "RedactedMediaFileUri": "string"  
        },  
        "MediaFormat": "string",  
        "MediaSampleRateHertz": number,  
        "ModelSettings": {  
            "LanguageModelName": "string"  
        }  
    }  
}
```

```
        },
        "Settings": {
            "ChannelIdentification": boolean,
            "MaxAlternatives": number,
            "MaxSpeakerLabels": number,
            "ShowAlternatives": boolean,
            "ShowSpeakerLabels": boolean,
            "VocabularyFilterMethod": "string",
            "VocabularyFilterName": "string",
            "VocabularyName": "string"
        },
        "StartTime": number,
        "Tags": [
            {
                "Key": "string",
                "Value": "string"
            }
        ],
        "Transcript": {
            "RedactedTranscriptFileUri": "string",
            "TranscriptFileUri": "string"
        },
        "TranscriptionJobName": "string",
        "TranscriptionJobStatus": "string"
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[TranscriptionJob \(p. 329\)](#)

An object that contains the results of the transcription job.

Type: [TranscriptionJob \(p. 459\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

GetVocabulary

Service: Amazon Transcribe Service

Gets information about a vocabulary.

Request Syntax

```
{  
    "VocabularyName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

VocabularyName (p. 332)

The name of the vocabulary to return information about. The name is case sensitive.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Syntax

```
{  
    "DownloadUri": "string",  
    "FailureReason": "string",  
    "LanguageCode": "string",  
    "LastModifiedTime": number,  
    "VocabularyName": "string",  
    "VocabularyState": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DownloadUri (p. 332)

The S3 location where the vocabulary is stored. Use this URI to get the contents of the vocabulary. The URI is available for a limited time.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (s3://|http(s*)://).+

[FailureReason \(p. 332\)](#)

If the `VocabularyState` field is `FAILED`, this field contains information about why the job failed.

Type: String

[LanguageCode \(p. 332\)](#)

The language code of the vocabulary entries.

Type: String

Valid Values: `af-ZA` | `ar-AE` | `ar-SA` | `cy-GB` | `da-DK` | `de-CH` | `de-DE` | `en-AB` | `en-AU` | `en-GB` | `en-IE` | `en-IN` | `en-US` | `en-WL` | `es-ES` | `es-US` | `fa-IR` | `fr-CA` | `fr-FR` | `ga-IE` | `gd-GB` | `he-IL` | `hi-IN` | `id-ID` | `it-IT` | `ja-JP` | `ko-KR` | `ms-MY` | `nl-NL` | `pt-BR` | `pt-PT` | `ru-RU` | `ta-IN` | `te-IN` | `tr-TR` | `zh-CN` | `zh-TW` | `th-TH` | `en-ZA` | `en-NZ`

[LastModifiedTime \(p. 332\)](#)

The date and time that the vocabulary was last modified.

Type: Timestamp

[VocabularyName \(p. 332\)](#)

The name of the vocabulary to return.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^ [0-9a-zA-Z._-] +`

[VocabularyState \(p. 332\)](#)

The processing state of the vocabulary.

Type: String

Valid Values: `PENDING` | `READY` | `FAILED`

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

GetVocabularyFilter

Service: Amazon Transcribe Service

Returns information about a vocabulary filter.

Request Syntax

```
{  
    "VocabularyFilterName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

VocabularyFilterName ([p. 335](#))

The name of the vocabulary filter for which to return information.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Syntax

```
{  
    "DownloadUri": "string",  
    "LanguageCode": "string",  
    "LastModifiedTime": number,  
    "VocabularyFilterName": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DownloadUri ([p. 335](#))

The URI of the list of words in the vocabulary filter. You can use this URI to get the list of words.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (s3://|http(s*)://).+

LanguageCode ([p. 335](#))

The language code of the words in the vocabulary filter.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

[LastModifiedTime \(p. 335\)](#)

The date and time that the contents of the vocabulary filter were updated.

Type: Timestamp

[VocabularyFilterName \(p. 335\)](#)

The name of the vocabulary filter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)

- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListCallAnalyticsCategories

Service: Amazon Transcribe Service

Provides more information about the call analytics categories that you've created. You can use the information in this list to find a specific category. You can then use the [GetCallAnalyticsCategory \(p. 317\)](#) operation to get more information about it.

Request Syntax

```
{  
    "MaxResults": number,  
    "NextToken": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

[MaxResults \(p. 338\)](#)

The maximum number of categories to return in each page of results. If there are fewer results than the value you specify, only the actual results are returned. If you do not specify a value, the default of 5 is used.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

[NextToken \(p. 338\)](#)

When included, `NextToken` fetches the next set of categories if the result of the previous request was truncated.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: `.+`

Required: No

Response Syntax

```
{  
    "Categories": [  
        {  
            "CategoryName": "string",  
            "CreateTime": number,  
            "LastUpdateTime": number,  
            "Rules": [  
                {  
                    "InterruptionFilter": {  
                        "AbsoluteTimeRange": {  
                            "EndTime": number,  
                            "StartTime": number  
                        }  
                    }  
                }  
            ]  
        }  
    ]  
}
```

```

        "First": number,
        "Last": number,
        "StartTime": number
    },
    "Negate": boolean,
    "ParticipantRole": "string",
    "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
    },
    "Threshold": number
},
"NonTalkTimeFilter": {
    "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
    },
    "Negate": boolean,
    "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
    },
    "Threshold": number
},
"SentimentFilter": {
    "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
    },
    "Negate": boolean,
    "ParticipantRole": "string",
    "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
    },
    "Sentiments": [ "string" ]
},
"TranscriptFilter": {
    "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
    },
    "Negate": boolean,
    "ParticipantRole": "string",
    "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
    },
    "Targets": [ "string" ],
    "TranscriptFilterType": "string"
}
}

```

```
        ],
    },
    "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Categories (p. 338)

A list of objects containing information about analytics categories.

Type: Array of [CategoryProperties](#) (p. 427) objects

NextToken (p. 338)

The [ListCallAnalyticsCategories](#) (p. 338) operation returns a page of jobs at a time. The maximum size of the list is set by the `MaxResults` parameter. If there are more categories in the list than the page size, Amazon Transcribe returns the `NextPage` token. Include the token in the next request to the [ListCallAnalyticsCategories](#) (p. 338) operation to return the next page of analytics categories.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

Errors

For information about the errors that are common to all actions, see [Common Errors](#) (p. 491).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)

- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListCallAnalyticsJobs

Service: Amazon Transcribe Service

List call analytics jobs with a specified status or substring that matches their names.

Request Syntax

```
{  
    "JobNameContains": "string",  
    "MaxResults": number,  
    "NextToken": "string",  
    "Status": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

[JobNameContains \(p. 342\)](#)

When specified, the jobs returned in the list are limited to jobs whose name contains the specified string.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

[MaxResults \(p. 342\)](#)

The maximum number of call analytics jobs to return in each page of results. If there are fewer results than the value you specify, only the actual results are returned. If you do not specify a value, the default of 5 is used.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

[NextToken \(p. 342\)](#)

If you receive a truncated result in the previous request of [ListCallAnalyticsJobs \(p. 342\)](#), include NextToken to fetch the next set of jobs.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

Required: No

[Status \(p. 342\)](#)

When specified, returns only call analytics jobs with the specified status. Jobs are ordered by creation date, with the most recent jobs returned first. If you don't specify a status, Amazon Transcribe returns all analytics jobs ordered by creation date.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Required: No

Response Syntax

```
{  
    "CallAnalyticsJobSummaries": [  
        {  
            "CallAnalyticsJobName": "string",  
            "CallAnalyticsJobStatus": "string",  
            "CompletionTime": number,  
            "CreationTime": number,  
            "FailureReason": "string",  
            "LanguageCode": "string",  
            "StartTime": number  
        }  
    ],  
    "NextToken": "string",  
    "Status": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[CallAnalyticsJobSummaries \(p. 343\)](#)

A list of objects containing summary information for a transcription job.

Type: Array of [CallAnalyticsJobSummary \(p. 425\)](#) objects

[NextToken \(p. 343\)](#)

The [ListCallAnalyticsJobs \(p. 342\)](#) operation returns a page of jobs at a time. The maximum size of the page is set by the `MaxResults` parameter. If there are more jobs in the list than the page size, Amazon Transcribe returns the `NextPage` token. Include the token in your next request to the [ListCallAnalyticsJobs \(p. 342\)](#) operation to return next page of jobs.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

[Status \(p. 343\)](#)

When specified, returns only call analytics jobs with that status. Jobs are ordered by creation date, with the most recent jobs returned first. If you don't specify a status, Amazon Transcribe returns all transcription jobs ordered by creation date.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception Message field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListLanguageModels

Service: Amazon Transcribe Service

Provides more information about the custom language models you've created. You can use the information in this list to find a specific custom language model. You can then use the [DescribeLanguageModel \(p. 315\)](#) operation to get more information about it.

Request Syntax

```
{  
    "MaxResults": number,  
    "NameContains": "string",  
    "NextToken": "string",  
    "StatusEquals": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

MaxResults (p. 345)

The maximum number of language models to return in each page of results. If there are fewer results than the value you specify, only the actual results are returned. If you do not specify a value, the default of 5 is used.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NameContains (p. 345)

When specified, the custom language model names returned contain the substring you've specified.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

NextToken (p. 345)

When included, fetches the next set of jobs if the result of the previous request was truncated.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

Required: No

[StatusEquals \(p. 345\)](#)

When specified, returns only custom language models with the specified status. Language models are ordered by creation date, with the newest models first. If you don't specify a status, Amazon Transcribe returns all custom language models ordered by date.

Type: String

Valid Values: IN_PROGRESS | FAILED | COMPLETED

Required: No

Response Syntax

```
{  
  "Models": [  
    {  
      "BaseModelName": "string",  
      "CreateTime": number,  
      "FailureReason": "string",  
      "InputDataConfig": {  
        "DataAccessRoleArn": "string",  
        "S3Uri": "string",  
        "TuningDataS3Uri": "string"  
      },  
      "LanguageCode": "string",  
      "LastModifiedTime": number,  
      "ModelName": "string",  
      "ModelStatus": "string",  
      "UpgradeAvailability": boolean  
    }  
  ],  
  "NextToken": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[Models \(p. 346\)](#)

A list of objects containing information about custom language models.

Type: Array of [LanguageModel \(p. 434\)](#) objects

[NextToken \(p. 346\)](#)

The [ListLanguageModels \(p. 345\)](#) operation returns a page of jobs at a time. The maximum size of the list is set by the MaxResults parameter. If there are more language models in the list than the page size, Amazon Transcribe returns the NextPage token. Include the token in the next request to the [ListLanguageModels \(p. 345\)](#) operation to return the next page of language models.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListMedicalTranscriptionJobs

Service: Amazon Transcribe Service

Lists medical transcription jobs with a specified status or substring that matches their names.

Request Syntax

```
{  
    "JobNameContains": "string",  
    "MaxResults": number,  
    "NextToken": "string",  
    "Status": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

JobNameContains (p. 348)

When specified, the jobs returned in the list are limited to jobs whose name contains the specified string.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

MaxResults (p. 348)

The maximum number of medical transcription jobs to return in each page of results. If there are fewer results than the value you specify, only the actual results are returned. If you do not specify a value, the default of 5 is used.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken (p. 348)

If you receive a truncated result in the previous request of `ListMedicalTranscriptionJobs`, include `NextToken` to fetch the next set of jobs.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

Required: No

Status (p. 348)

When specified, returns only medical transcription jobs with the specified status. Jobs are ordered by creation date, with the newest jobs returned first. If you don't specify a status, Amazon Transcribe Medical returns all transcription jobs ordered by creation date.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Required: No

Response Syntax

```
{  
    "MedicalTranscriptionJobSummaries": [  
        {  
            "CompletionTime": number,  
            "ContentIdentificationType": "string",  
            "CreationTime": number,  
            "FailureReason": "string",  
            "LanguageCode": "string",  
            "MedicalTranscriptionJobName": "string",  
            "OutputLocationType": "string",  
            "Specialty": "string",  
            "StartTime": number,  
            "TranscriptionJobStatus": "string",  
            "Type": "string"  
        }  
    ],  
    "NextToken": "string",  
    "Status": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[MedicalTranscriptionJobSummaries \(p. 349\)](#)

A list of objects containing summary information for a transcription job.

Type: Array of [MedicalTranscriptionJobSummary \(p. 442\)](#) objects

[NextToken \(p. 349\)](#)

The `ListMedicalTranscriptionJobs` operation returns a page of jobs at a time. The maximum size of the page is set by the `MaxResults` parameter. If the number of jobs exceeds what can fit on a page, Amazon Transcribe Medical returns the `NextPage` token. Include the token in the next request to the `ListMedicalTranscriptionJobs` operation to return in the next page of jobs.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

[Status \(p. 349\)](#)

The requested status of the medical transcription jobs returned.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListMedicalVocabularies

Service: Amazon Transcribe Service

Returns a list of vocabularies that match the specified criteria. If you don't enter a value in any of the request parameters, returns the entire list of vocabularies.

Request Syntax

```
{  
    "MaxResults": number,  
    "NameContains": "string",  
    "NextToken": "string",  
    "StateEquals": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

MaxResults ([p. 351](#))

The maximum number of vocabularies to return in each page of results. If there are fewer results than the value you specify, only the actual results are returned. If you do not specify a value, the default of 5 is used.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NameContains ([p. 351](#))

Returns vocabularies whose names contain the specified string. The search is not case sensitive. `ListMedicalVocabularies` returns both "vocabularyname" and "VocabularyName".

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

NextToken ([p. 351](#))

If the result of your previous request to `ListMedicalVocabularies` was truncated, include the `NextToken` to fetch the next set of vocabularies.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

Required: No

[StateEquals \(p. 351\)](#)

When specified, returns only vocabularies with the `VocabularyState` equal to the specified vocabulary state. Use this field to see which vocabularies are ready for your medical transcription jobs.

Type: String

Valid Values: `PENDING` | `READY` | `FAILED`

Required: No

Response Syntax

```
{  
    "NextToken": "string",  
    "Status": "string",  
    "Vocabularies": [  
        {  
            "LanguageCode": "string",  
            "LastModifiedTime": number,  
            "VocabularyName": "string",  
            "VocabularyState": "string"  
        }  
    ]  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[NextToken \(p. 352\)](#)

The `ListMedicalVocabularies` operation returns a page of vocabularies at a time. You set the maximum number of vocabularies to return on a page with the `MaxResults` parameter. If there are more jobs in the list will fit on a page, Amazon Transcribe Medical returns the `NextPage` token. To return the next page of vocabularies, include the token in the next request to the `ListMedicalVocabularies` operation .

Type: String

Length Constraints: Maximum length of 8192.

Pattern: `.+`

[Status \(p. 352\)](#)

The requested vocabulary state.

Type: String

Valid Values: `PENDING` | `READY` | `FAILED`

[Vocabularies \(p. 352\)](#)

A list of objects that describe the vocabularies that match your search criteria.

Type: Array of [VocabularyInfo \(p. 467\)](#) objects

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListTagsForResource

Service: Amazon Transcribe Service

Lists all tags associated with a given transcription job, vocabulary, or resource.

Request Syntax

```
{  
    "ResourceArn": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

ResourceArn (p. 354)

Lists all tags associated with a given Amazon Resource Name (ARN).

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1011.

Pattern: arn:aws(-[^:]+)?:transcribe:[a-zA-Z0-9-]*:[0-9]{12}:[a-zA-Z-]*/[0-9a-zA-Z._-]+

Required: Yes

Response Syntax

```
{  
    "ResourceArn": "string",  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ]  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ResourceArn (p. 354)

Lists all tags associated with the given Amazon Resource Name (ARN).

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1011.

Pattern: arn:aws(-[^:]+)?:transcribe:[a-zA-Z0-9-]*:[0-9]{12}:[a-zA-Z-]*/[0-9a-zA-Z._-]+

[Tags \(p. 354\)](#)

Lists all tags associated with the given transcription job, vocabulary, or resource.

Type: Array of [Tag \(p. 455\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListTranscriptionJobs

Service: Amazon Transcribe Service

Lists transcription jobs with the specified status.

Request Syntax

```
{  
    "JobNameContains": "string",  
    "MaxResults": number,  
    "NextToken": "string",  
    "Status": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

JobNameContains (p. 356)

When specified, the jobs returned in the list are limited to jobs whose name contains the specified string.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

MaxResults (p. 356)

The maximum number of jobs to return in each page of results. If there are fewer results than the value you specify, only the actual results are returned. If you do not specify a value, the default of 5 is used.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken (p. 356)

If the result of the previous request to `ListTranscriptionJobs` is truncated, include the `NextToken` to fetch the next set of jobs.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

Required: No

Status (p. 356)

When specified, returns only transcription jobs with the specified status. Jobs are ordered by creation date, with the newest jobs returned first. If you don't specify a status, Amazon Transcribe returns all transcription jobs ordered by creation date.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Required: No

Response Syntax

```
{  
    "NextToken": "string",  
    "Status": "string",  
    "TranscriptionJobSummaries": [  
        {  
            "CompletionTime": number,  
            "ContentRedaction": {  
                "RedactionOutput": "string",  
                "RedactionType": "string"  
            },  
            "CreationTime": number,  
            "FailureReason": "string",  
            "IdentifiedLanguageScore": number,  
            "IdentifyLanguage": boolean,  
            "LanguageCode": "string",  
            "ModelSettings": {  
                "LanguageModelName": "string"  
            },  
            "OutputLocationType": "string",  
            "StartTime": number,  
            "TranscriptionJobName": "string",  
            "TranscriptionJobStatus": "string"  
        }  
    ]  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[NextToken \(p. 357\)](#)

The `ListTranscriptionJobs` operation returns a page of jobs at a time. The maximum size of the page is set by the `MaxResults` parameter. If there are more jobs in the list than the page size, Amazon Transcribe returns the `NextPage` token. Include the token in the next request to the `ListTranscriptionJobs` operation to return in the next page of jobs.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

[Status \(p. 357\)](#)

The requested status of the jobs returned.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

[TranscriptionJobSummaries \(p. 357\)](#)

A list of objects containing summary information for a transcription job.

Type: Array of [TranscriptionJobSummary \(p. 463\)](#) objects

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListVocabularies

Service: Amazon Transcribe Service

Returns a list of vocabularies that match the specified criteria. If no criteria are specified, returns the entire list of vocabularies.

Request Syntax

```
{  
    "MaxResults": number,  
    "NameContains": "string",  
    "NextToken": "string",  
    "StateEquals": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

[MaxResults \(p. 359\)](#)

The maximum number of vocabularies to return in each page of results. If there are fewer results than the value you specify, only the actual results are returned. If you do not specify a value, the default of 5 is used.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

[NameContains \(p. 359\)](#)

When specified, the vocabularies returned in the list are limited to vocabularies whose name contains the specified string. The search is not case sensitive. `ListVocabularies` returns both "vocabularyname" and "VocabularyName" in the response list.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

[NextToken \(p. 359\)](#)

If the result of the previous request to `ListVocabularies` was truncated, include the `NextToken` to fetch the next set of jobs.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

Required: No

[StateEquals \(p. 359\)](#)

When specified, only returns vocabularies with the `VocabularyState` field equal to the specified state.

Type: String

Valid Values: PENDING | READY | FAILED

Required: No

Response Syntax

```
{  
    "NextToken": "string",  
    "Status": "string",  
    "Vocabularies": [  
        {  
            "LanguageCode": "string",  
            "LastModifiedTime": number,  
            "VocabularyName": "string",  
            "VocabularyState": "string"  
        }  
    ]  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[NextToken \(p. 360\)](#)

The `ListVocabularies` operation returns a page of vocabularies at a time. The maximum size of the page is set in the `MaxResults` parameter. If there are more jobs in the list than will fit on the page, Amazon Transcribe returns the `NextPage` token. To return in the next page of jobs, include the token in the next request to the `ListVocabularies` operation.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

[Status \(p. 360\)](#)

The requested vocabulary state.

Type: String

Valid Values: PENDING | READY | FAILED

[Vocabularies \(p. 360\)](#)

A list of objects that describe the vocabularies that match the search criteria in the request.

Type: Array of [VocabularyInfo \(p. 467\)](#) objects

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception Message field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

ListVocabularyFilters

Service: Amazon Transcribe Service

Gets information about vocabulary filters.

Request Syntax

```
{  
    "MaxResults": number,  
    "NameContains": "string",  
    "NextToken": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

[MaxResults \(p. 362\)](#)

The maximum number of filters to return in each page of results. If there are fewer results than the value you specify, only the actual results are returned. If you do not specify a value, the default of 5 is used.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

[NameContains \(p. 362\)](#)

Filters the response so that it only contains vocabulary filters whose name contains the specified string.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

[NextToken \(p. 362\)](#)

If the result of the previous request to `ListVocabularyFilters` was truncated, include the `NextToken` to fetch the next set of collections.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

Required: No

Response Syntax

```
{
```

```
"NextToken": "string",
"VocabularyFilters": [
    {
        "LanguageCode": "string",
        "LastModifiedTime": number,
        "VocabularyFilterName": "string"
    }
]
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken (p. 362)

The `ListVocabularyFilters` operation returns a page of collections at a time. The maximum size of the page is set by the `MaxResults` parameter. If there are more jobs in the list than the page size, Amazon Transcribe returns the `NextPage` token. Include the token in the next request to the `ListVocabularyFilters` operation to return in the next page of jobs.

Type: String

Length Constraints: Maximum length of 8192.

Pattern: .+

VocabularyFilters (p. 362)

The list of vocabulary filters. It contains at most `MaxResults` number of filters. If there are more filters, call the `ListVocabularyFilters` operation again with the `NextToken` parameter in the request set to the value of the `NextToken` field in the response.

Type: Array of [VocabularyFilterInfo](#) (p. 466) objects

Errors

For information about the errors that are common to all actions, see [Common Errors](#) (p. 491).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

StartCallAnalyticsJob

Service: Amazon Transcribe Service

Starts an asynchronous analytics job that not only transcribes the audio recording of a caller and agent, but also returns additional insights. These insights include how quickly or loudly the caller or agent was speaking. To retrieve additional insights with your analytics jobs, create categories. A category is a way to classify analytics jobs based on attributes, such as a customer's sentiment or a particular phrase being used during the call. For more information, see the [CreateCallAnalyticsCategory \(p. 278\)](#) operation.

Request Syntax

```
{  
    "CallAnalyticsJobName": "string",  
    "ChannelDefinitions": [  
        {  
            "ChannelId": number,  
            "ParticipantRole": "string"  
        }  
    ],  
    "DataAccessRoleArn": "string",  
    "Media": {  
        "MediaFileUri": "string",  
        "RedactedMediaFileUri": "string"  
    },  
    "OutputEncryptionKMSKeyId": "string",  
    "OutputLocation": "string",  
    "Settings": {  
        "ContentRedaction": {  
            "RedactionOutput": "string",  
            "RedactionType": "string"  
        },  
        "LanguageModelName": "string",  
        "LanguageOptions": [ "string" ],  
        "VocabularyFilterMethod": "string",  
        "VocabularyFilterName": "string",  
        "VocabularyName": "string"  
    }  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

CallAnalyticsJobName (p. 365)

The name of the call analytics job. You can't use the string "." or ".." by themselves as the job name. The name must also be unique within an AWS account. If you try to create a call analytics job with the same name as a previous call analytics job, you get a `ConflictException` error.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

[ChannelDefinitions \(p. 365\)](#)

When you start a call analytics job, you must pass an array that maps the agent and the customer to specific audio channels. The values you can assign to a channel are 0 and 1. The agent and the customer must each have their own channel. You can't assign more than one channel to an agent or customer.

Type: Array of [ChannelDefinition \(p. 428\)](#) objects

Array Members: Fixed number of 2 items.

Required: No

[DataAccessRoleArn \(p. 365\)](#)

The Amazon Resource Name (ARN) of a role that has access to the S3 bucket that contains your input files. Amazon Transcribe assumes this role to read queued audio files. If you have specified an output S3 bucket for your transcription results, this role should have access to the output bucket as well.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: ^arn:(aws|aws-cn|aws-us-gov|aws-iso-{0,1}[a-z]{0,1}):iam::[0-9]{0,63}:role/[A-Za-z0-9:_/+=@.-]{0,1024}\\$

Required: Yes

[Media \(p. 365\)](#)

Describes the input media file in a transcription request.

Type: [Media \(p. 436\)](#) object

Required: Yes

[OutputEncryptionKMSKeyId \(p. 365\)](#)

The Amazon Resource Name (ARN) of the AWS Key Management Service key used to encrypt the output of the call analytics job. The user calling the [StartCallAnalyticsJob \(p. 365\)](#) operation must have permission to use the specified KMS key.

You use either of the following to identify an AWS KMS key in the current account:

- KMS Key ID: "1234abcd-12ab-34cd-56ef-1234567890ab"
- KMS Key Alias: "alias/ExampleAlias"

You can use either of the following to identify a KMS key in the current account or another account:

- Amazon Resource Name (ARN) of a KMS key in the current account or another account:
"arn:aws:kms:region:account ID:key/1234abcd-12ab-34cd-56ef1234567890ab"
- ARN of a KMS Key Alias: "arn:aws:kms:region:account ID:alias/ExampleAlias"

If you don't specify an encryption key, the output of the call analytics job is encrypted with the default Amazon S3 key (SSE-S3).

If you specify a KMS key to encrypt your output, you must also specify an output location in the `OutputLocation` parameter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2048.

Pattern: ^[A-Za-z0-9][A-Za-z0-9:_/+=@.-]{0,2048}\\$

Required: No

[OutputLocation \(p. 365\)](#)

The Amazon S3 location where the output of the call analytics job is stored. You can provide the following location types to store the output of call analytics job:

- s3://DOC-EXAMPLE-BUCKET1

If you specify a bucket, Amazon Transcribe saves the output of the analytics job as a JSON file at the root level of the bucket.

- s3://DOC-EXAMPLE-BUCKET1/folder/

If you specify a path, Amazon Transcribe saves the output of the analytics job as s3://DOC-EXAMPLE-BUCKET1/folder/your-transcription-job-name.json

If you specify a folder, you must provide a trailing slash.

- s3://DOC-EXAMPLE-BUCKET1/folder/filename.json

If you provide a path that has the filename specified, Amazon Transcribe saves the output of the analytics job as s3://DOC-EXAMPLEBUCKET1/folder/filename.json

You can specify an AWS Key Management Service (KMS) key to encrypt the output of our analytics job using the `OutputEncryptionKMSKeyId` parameter. If you don't specify a KMS key, Amazon Transcribe uses the default Amazon S3 key for server-side encryption of the analytics job output that is placed in your S3 bucket.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (s3://|http(s*)://).+

Required: No

[Settings \(p. 365\)](#)

A `Settings` object that provides optional settings for a call analytics job.

Type: [CallAnalyticsJobSettings \(p. 423\)](#) object

Required: No

Response Syntax

```
{  
  "CallAnalyticsJob": {  
    "CallAnalyticsJobName": "string",  
    "CallAnalyticsJobStatus": "string",  
    "ChannelDefinitions": [  
      {  
        "ChannelId": number,  
        "ParticipantRole": "string"  
      }  
    ],  
    "CompletionTime": number,  
    "CreationTime": number,  
    "DataAccessRoleArn": "string",  
    "FailureReason": "string",  
    "IdentifiedLanguageScore": number,  
    "LanguageCode": "string",  
    "Media": {  
      "Content": "string",  
      "Format": "string",  
      "Name": "string",  
      "Type": "string"  
    }  
  }  
}
```

```
        "MediaFileUri": "string",
        "RedactedMediaFileUri": "string"
    },
    "MediaFormat": "string",
    "MediaSampleRateHertz": number,
    "Settings": {
        "ContentRedaction": {
            "RedactionOutput": "string",
            "RedactionType": "string"
        },
        "LanguageModelName": "string",
        "LanguageOptions": [ "string" ],
        "VocabularyFilterMethod": "string",
        "VocabularyFilterName": "string",
        "VocabularyName": "string"
    },
    "StartTime": number,
    "Transcript": {
        "RedactedTranscriptFileUri": "string",
        "TranscriptFileUri": "string"
    }
}
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[CallAnalyticsJob \(p. 367\)](#)

An object containing the details of the asynchronous call analytics job.

Type: [CallAnalyticsJob \(p. 419\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

StartMedicalTranscriptionJob

Service: Amazon Transcribe Service

Starts a batch job to transcribe medical speech to text.

Request Syntax

```
{  
    "ContentIdentificationType": "string",  
    "KMSEncryptionContext": {  
        "string" : "string"  
    },  
    "LanguageCode": "string",  
    "Media": {  
        "MediaFileUri": "string",  
        "RedactedMediaFileUri": "string"  
    },  
    "MediaFormat": "string",  
    "MediaSampleRateHertz": number,  
    "MedicalTranscriptionJobName": "string",  
    "OutputBucketName": "string",  
    "OutputEncryptionKMSKeyId": "string",  
    "OutputKey": "string",  
    "Settings": {  
        "ChannelIdentification": boolean,  
        "MaxAlternatives": number,  
        "MaxSpeakerLabels": number,  
        "ShowAlternatives": boolean,  
        "ShowSpeakerLabels": boolean,  
        "VocabularyName": "string"  
    },  
    "Specialty": "string",  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ],  
    "Type": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

ContentIdentificationType (p. 370)

You can configure Amazon Transcribe Medical to label content in the transcription output. If you specify **PHI**, Amazon Transcribe Medical labels the personal health information (PHI) that it identifies in the transcription output.

Type: String

Valid Values: **PHI**

Required: No

[KMSEncryptionContext \(p. 370\)](#)

A map of plain text, non-secret key:value pairs, known as encryption context pairs, that provide an added layer of security for your data.

Type: String to string map

Map Entries: Maximum number of 10 items.

Key Length Constraints: Minimum length of 1. Maximum length of 2000.

Key Pattern: `.*\S.*`

Value Length Constraints: Minimum length of 1. Maximum length of 2000.

Value Pattern: `.*\S.*`

Required: No

[LanguageCode \(p. 370\)](#)

The language code for the language spoken in the input media file. US English (en-US) is the valid value for medical transcription jobs. Any other value you enter for language code results in a `BadRequestException` error.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: Yes

[Media \(p. 370\)](#)

Describes the input media file in a transcription request.

Type: [Media \(p. 436\)](#) object

Required: Yes

[MediaFormat \(p. 370\)](#)

The audio format of the input media file.

Type: String

Valid Values: mp3 | mp4 | wav | flac | ogg | amr | webm

Required: No

[MediaSampleRateHertz \(p. 370\)](#)

The sample rate, in Hertz, of the audio track in the input media file.

If you do not specify the media sample rate, Amazon Transcribe Medical determines the sample rate. If you specify the sample rate, it must match the rate detected by Amazon Transcribe Medical. In most cases, you should leave the `MediaSampleRateHertz` field blank and let Amazon Transcribe Medical determine the sample rate.

Type: Integer

Valid Range: Minimum value of 8000. Maximum value of 48000.

Required: No

[MedicalTranscriptionJobName \(p. 370\)](#)

The name of the medical transcription job. You can't use the strings ". ." or ". ." by themselves as the job name. The name must also be unique within an AWS account. If you try to create a medical transcription job with the same name as a previous medical transcription job, you get a `ConflictException` error.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

[OutputBucketName \(p. 370\)](#)

The Amazon S3 location where the transcription is stored.

You must set `OutputBucketName` for Amazon Transcribe Medical to store the transcription results. Your transcript appears in the S3 location you specify. When you call the [GetMedicalTranscriptionJob \(p. 323\)](#), the operation returns this location in the `TranscriptFileUri` field. The S3 bucket must have permissions that allow Amazon Transcribe Medical to put files in the bucket. For more information, see [Permissions Required for IAM User Roles](#).

You can specify an AWS Key Management Service (KMS) key to encrypt the output of your transcription using the `OutputEncryptionKMSKeyId` parameter. If you don't specify a KMS key, Amazon Transcribe Medical uses the default Amazon S3 key for server-side encryption of transcripts that are placed in your S3 bucket.

Type: String

Length Constraints: Maximum length of 64.

Pattern: [a-zA-Z0-9][\.-a-zA-Z0-9]{1,61}[a-zA-Z0-9]

Required: Yes

[OutputEncryptionKMSKeyId \(p. 370\)](#)

The Amazon Resource Name (ARN) of the AWS Key Management Service (KMS) key used to encrypt the output of the transcription job. The user calling the [StartMedicalTranscriptionJob \(p. 370\)](#) operation must have permission to use the specified KMS key.

You use either of the following to identify a KMS key in the current account:

- KMS Key ID: "1234abcd-12ab-34cd-56ef-1234567890ab"
- KMS Key Alias: "alias/ExampleAlias"

You can use either of the following to identify a KMS key in the current account or another account:

- Amazon Resource Name (ARN) of a KMS key in the current account or another account:
"arn:aws:kms:region:account ID:key/1234abcd-12ab-34cd-56ef-1234567890ab"
- ARN of a KMS Key Alias: "arn:aws:kms:region:account ID:alias/ExampleAlias"

If you don't specify an encryption key, the output of the medical transcription job is encrypted with the default Amazon S3 key (SSE-S3).

If you specify a KMS key to encrypt your output, you must also specify an output location in the `OutputBucketName` parameter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2048.

Pattern: ^[A-Za-z0-9][A-Za-z0-9:_/+,@.-]{0,2048}\$

Required: No

[OutputKey \(p. 370\)](#)

You can specify a location in an Amazon S3 bucket to store the output of your medical transcription job.

If you don't specify an output key, Amazon Transcribe Medical stores the output of your transcription job in the Amazon S3 bucket you specified. By default, the object key is "your-transcription-job-name.json".

You can use output keys to specify the Amazon S3 prefix and file name of the transcription output. For example, specifying the Amazon S3 prefix, "folder1/folder2/", as an output key would lead to the output being stored as "folder1/folder2/your-transcription-job-name.json". If you specify "my-other-job-name.json" as the output key, the object key is changed to "my-other-job-name.json". You can use an output key to change both the prefix and the file name, for example "folder/my-other-job-name.json".

If you specify an output key, you must also specify an S3 bucket in the `OutputBucketName` parameter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: [a-zA-Z0-9-_!*'()]{1,1024}\$

Required: No

[Settings \(p. 370\)](#)

Optional settings for the medical transcription job.

Type: [MedicalTranscriptionSetting \(p. 444\)](#) object

Required: No

[Specialty \(p. 370\)](#)

The medical specialty of any clinician speaking in the input media.

Type: String

Valid Values: PRIMARYCARE

Required: Yes

[Tags \(p. 370\)](#)

Add tags to an Amazon Transcribe medical transcription job.

Type: Array of [Tag \(p. 455\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Required: No

Type (p. 370)

The type of speech in the input audio. CONVERSATION refers to conversations between two or more speakers, e.g., a conversations between doctors and patients. DICTATION refers to single-speaker dictated speech, such as clinical notes.

Type: String

Valid Values: CONVERSATION | DICTATION

Required: Yes

Response Syntax

```
{  
    "MedicalTranscriptionJob": {  
        "CompletionTime": number,  
        "ContentIdentificationType": "string",  
        "CreationTime": number,  
        "FailureReason": "string",  
        "LanguageCode": "string",  
        "Media": {  
            "MediaFileUri": "string",  
            "RedactedMediaFileUri": "string"  
        },  
        "MediaFormat": "string",  
        "MediaSampleRateHertz": number,  
        "MedicalTranscriptionJobName": "string",  
        "Settings": {  
            "ChannelIdentification": boolean,  
            "MaxAlternatives": number,  
            "MaxSpeakerLabels": number,  
            "ShowAlternatives": boolean,  
            "ShowSpeakerLabels": boolean,  
            "VocabularyName": "string"  
        },  
        "Specialty": "string",  
        "StartTime": number,  
        "Tags": [  
            {  
                "Key": "string",  
                "Value": "string"  
            }  
        ],  
        "Transcript": {  
            "TranscriptFileUri": "string"  
        },  
        "TranscriptionJobStatus": "string",  
        "Type": "string"  
    }  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

MedicalTranscriptionJob (p. 374)

A batch job submitted to transcribe medical speech to text.

Type: [MedicalTranscriptionJob \(p. 438\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

StartTranscriptionJob

Service: Amazon Transcribe Service

Starts an asynchronous job to transcribe speech to text.

Request Syntax

```
{  
    "ContentRedaction": {  
        "RedactionOutput": "string",  
        "RedactionType": "string"  
    },  
    "IdentifyLanguage": boolean,  
    "JobExecutionSettings": {  
        "AllowDeferredExecution": boolean,  
        "DataAccessRoleArn": "string"  
    },  
    "KMSEncryptionContext": {  
        "string" : "string"  
    },  
    "LanguageCode": "string",  
    "LanguageOptions": [ "string" ],  
    "Media": {  
        "MediaFileUri": "string",  
        "RedactedMediaFileUri": "string"  
    },  
    "MediaFormat": "string",  
    "MediaSampleRateHertz": number,  
    "ModelSettings": {  
        "LanguageModelName": "string"  
    },  
    "OutputBucketName": "string",  
    "OutputEncryptionKMSKeyId": "string",  
    "OutputKey": "string",  
    "Settings": {  
        "ChannelIdentification": boolean,  
        "MaxAlternatives": number,  
        "MaxSpeakerLabels": number,  
        "ShowAlternatives": boolean,  
        "ShowSpeakerLabels": boolean,  
        "VocabularyFilterMethod": "string",  
        "VocabularyFilterName": "string",  
        "VocabularyName": "string"  
    },  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ],  
    "TranscriptionJobName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

[ContentRedaction \(p. 376\)](#)

An object that contains the request parameters for content redaction.

Type: [ContentRedaction \(p. 429\)](#) object

Required: No

[IdentifyLanguage \(p. 376\)](#)

Set this field to `true` to enable automatic language identification. Automatic language identification is disabled by default. You receive a `BadRequestException` error if you enter a value for a `LanguageCode`.

Type: Boolean

Required: No

[JobExecutionSettings \(p. 376\)](#)

Provides information about how a transcription job is executed. Use this field to indicate that the job can be queued for deferred execution if the concurrency limit is reached and there are no slots available to immediately run the job.

Type: [JobExecutionSettings \(p. 433\)](#) object

Required: No

[KMSEncryptionContext \(p. 376\)](#)

A map of plain text, non-secret key:value pairs, known as encryption context pairs, that provide an added layer of security for your data.

Type: String to string map

Map Entries: Maximum number of 10 items.

Key Length Constraints: Minimum length of 1. Maximum length of 2000.

Key Pattern: `.*\$.*`

Value Length Constraints: Minimum length of 1. Maximum length of 2000.

Value Pattern: `.*\$.*`

Required: No

[LanguageCode \(p. 376\)](#)

The language code for the language used in the input media file.

To transcribe speech in Modern Standard Arabic (ar-SA), your audio or video file must be encoded at a sample rate of 16,000 Hz or higher.

Type: String

Valid Values: `af-ZA` | `ar-AE` | `ar-SA` | `cy-GB` | `da-DK` | `de-CH` | `de-DE` | `en-AB` | `en-AU` | `en-GB` | `en-IE` | `en-IN` | `en-US` | `en-WL` | `es-ES` | `es-US` | `fa-IR` | `fr-CA` | `fr-FR` | `ga-IE` | `gd-GB` | `he-IL` | `hi-IN` | `id-ID` | `it-IT` | `ja-JP` | `ko-KR` | `ms-MY` | `nl-NL` | `pt-BR` | `pt-PT` | `ru-RU` | `ta-IN` | `te-IN` | `tr-TR` | `zh-CN` | `zh-TW` | `th-TH` | `en-ZA` | `en-NZ`

Required: No

[LanguageOptions \(p. 376\)](#)

An object containing a list of languages that might be present in your collection of audio files. Automatic language identification chooses a language that best matches the source audio from that list.

To transcribe speech in Modern Standard Arabic (ar-SA), your audio or video file must be encoded at a sample rate of 16,000 Hz or higher.

Type: Array of strings

Array Members: Minimum number of 1 item.

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: No

[Media \(p. 376\)](#)

An object that describes the input media for a transcription job.

Type: [Media \(p. 436\)](#) object

Required: Yes

[MediaFormat \(p. 376\)](#)

The format of the input media file.

Type: String

Valid Values: mp3 | mp4 | wav | flac | ogg | amr | webm

Required: No

[MediaSampleRateHertz \(p. 376\)](#)

The sample rate, in Hertz, of the audio track in the input media file.

If you do not specify the media sample rate, Amazon Transcribe determines the sample rate. If you specify the sample rate, it must match the sample rate detected by Amazon Transcribe. In most cases, you should leave the `MediaSampleRateHertz` field blank and let Amazon Transcribe determine the sample rate.

Type: Integer

Valid Range: Minimum value of 8000. Maximum value of 48000.

Required: No

[ModelSettings \(p. 376\)](#)

Choose the custom language model you use for your transcription job in this parameter.

Type: [ModelSettings \(p. 446\)](#) object

Required: No

[OutputBucketName \(p. 376\)](#)

The location where the transcription is stored.

If you set the `OutputBucketName`, Amazon Transcribe puts the transcript in the specified S3 bucket. When you call the [GetTranscriptionJob \(p. 329\)](#) operation, the operation returns this location in the `TranscriptFileUri` field. If you enable content redaction, the redacted transcript appears in `RedactedTranscriptFileUri`. If you enable content redaction and choose to output an

unredacted transcript, that transcript's location still appears in the `TranscriptFileUri`. The S3 bucket must have permissions that allow Amazon Transcribe to put files in the bucket. For more information, see [Permissions Required for IAM User Roles](#).

You can specify an AWS Key Management Service (KMS) key to encrypt the output of your transcription using the `OutputEncryptionKMSKeyId` parameter. If you don't specify a KMS key, Amazon Transcribe uses the default Amazon S3 key for server-side encryption of transcripts that are placed in your S3 bucket.

If you don't set the `OutputBucketName`, Amazon Transcribe generates a pre-signed URL, a shareable URL that provides secure access to your transcription, and returns it in the `TranscriptFileUri` field. Use this URL to download the transcription.

Type: String

Length Constraints: Maximum length of 64.

Pattern: [a-zA-Z0-9][\.-a-zA-Z0-9]{1,61}[a-zA-Z0-9]

Required: No

[OutputEncryptionKMSKeyId \(p. 376\)](#)

The Amazon Resource Name (ARN) of the AWS Key Management Service (KMS) key used to encrypt the output of the transcription job. The user calling the `StartTranscriptionJob` operation must have permission to use the specified KMS key.

You can use either of the following to identify a KMS key in the current account:

- KMS Key ID: "1234abcd-12ab-34cd-56ef-1234567890ab"
- KMS Key Alias: "alias/ExampleAlias"

You can use either of the following to identify a KMS key in the current account or another account:

- Amazon Resource Name (ARN) of a KMS Key: "arn:aws:kms:region:account ID:key/1234abcd-12ab-34cd-56ef-1234567890ab"
- ARN of a KMS Key Alias: "arn:aws:kms:region:account ID:alias/ExampleAlias"

If you don't specify an encryption key, the output of the transcription job is encrypted with the default Amazon S3 key (SSE-S3).

If you specify a KMS key to encrypt your output, you must also specify an output location in the `OutputBucketName` parameter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2048.

Pattern: ^[A-Za-z0-9][A-Za-z0-9:_/+@.-]{0,2048}\$

Required: No

[OutputKey \(p. 376\)](#)

You can specify a location in an Amazon S3 bucket to store the output of your transcription job.

If you don't specify an output key, Amazon Transcribe stores the output of your transcription job in the Amazon S3 bucket you specified. By default, the object key is "your-transcription-job-name.json".

You can use output keys to specify the Amazon S3 prefix and file name of the transcription output. For example, specifying the Amazon S3 prefix, "folder1/folder2/", as an output key would lead to the output being stored as "folder1/folder2/your-transcription-job-name.json". If you specify "my-other-job-name.json" as the output key, the object key is changed to "my-other-job-name.json". You

can use an output key to change both the prefix and the file name, for example "folder/my-other-job-name.json".

If you specify an output key, you must also specify an S3 bucket in the `OutputBucketName` parameter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: [a-zA-Z0-9-_!*'()/{1,1024}]\$

Required: No

[Settings \(p. 376\)](#)

A `Settings` object that provides optional settings for a transcription job.

Type: [Settings \(p. 453\)](#) object

Required: No

[Tags \(p. 376\)](#)

Add tags to an Amazon Transcribe transcription job.

Type: Array of [Tag \(p. 455\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Required: No

[TranscriptionJobName \(p. 376\)](#)

The name of the job. You can't use the strings "." or ".." by themselves as the job name. The name must also be unique within an AWS account. If you try to create a transcription job with the same name as a previous transcription job, you get a `ConflictException` error.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

[Response Syntax](#)

```
{  
    "TranscriptionJob": {  
        "CompletionTime": number,  
        "ContentRedaction": {  
            "RedactionOutput": "string",  
            "RedactionType": "string"  
        },  
        "CreationTime": number,  
        "FailureReason": "string",  
        "IdentifiedLanguageScore": number,  
        "IdentifyLanguage": boolean,  
        "JobExecutionSettings": {  
            "AllowDeferredExecution": boolean,  
            "DataAccessRoleArn": "string"  
        },  
        "LanguageCode": "string",  
        "LastModifiedTime": number,  
        "Status": "string",  
        "Transcript": {  
            "LanguageCode": "string",  
            "Text": "string"  
        }  
    },  
    "TranscriptionJobConfig": {  
        "EncryptionConfiguration": {  
            "EncryptionType": "string",  
            "KmsKey": "string"  
        },  
        "LanguageCode": "string",  
        "MaxSpeakerLabels": number,  
        "OutputBucketName": "string",  
        "OutputConfig": {  
            "FileFormat": "string",  
            "OutputFormat": "string"  
        },  
        "VocabularyName": "string"  
    }  
}
```

```
"LanguageOptions": [ "string" ],
"Media": {
    "MediaFileUri": "string",
    "RedactedMediaFileUri": "string"
},
"MediaFormat": "string",
"MediaSampleRateHertz": number,
"ModelSettings": {
    "LanguageModelName": "string"
},
"Settings": {
    "ChannelIdentification": boolean,
    "MaxAlternatives": number,
    "MaxSpeakerLabels": number,
    "ShowAlternatives": boolean,
    "ShowSpeakerLabels": boolean,
    "VocabularyFilterMethod": "string",
    "VocabularyFilterName": "string",
    "VocabularyName": "string"
},
"StartTime": number,
"Tags": [
    {
        "Key": "string",
        "Value": "string"
    }
],
"Transcript": {
    "RedactedTranscriptFileUri": "string",
    "TranscriptFileUri": "string"
},
"TranscriptionJobName": "string",
"TranscriptionJobStatus": "string"
}
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[TranscriptionJob \(p. 380\)](#)

An object containing details of the asynchronous transcription job.

Type: [TranscriptionJob \(p. 459\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception Message field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

TagResource

Service: Amazon Transcribe Service

Tags a Amazon Transcribe resource with the given list of tags.

Request Syntax

```
{  
    "ResourceArn": "string",  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ]  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

[ResourceArn \(p. 383\)](#)

The Amazon Resource Name (ARN) of the Amazon Transcribe resource you want to tag.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1011.

Pattern: arn:aws(-[^:]*)?:transcribe:[a-zA-Z0-9-]*:[0-9]{12}:[a-zA-Z-]*/[0-9a-zA-Z._-]+

Required: Yes

[Tags \(p. 383\)](#)

The tags you are assigning to a given Amazon Transcribe resource.

Type: Array of [Tag \(p. 455\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UntagResource

Service: Amazon Transcribe Service

Removes specified tags from a specified Amazon Transcribe resource.

Request Syntax

```
{  
    "ResourceArn": "string",  
    "TagKeys": [ "string" ]  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

[ResourceArn \(p. 385\)](#)

The Amazon Resource Name (ARN) of the Amazon Transcribe resource you want to remove tags from.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1011.

Pattern: arn:aws(-[^:]+)?:transcribe:[a-zA-Z0-9-]*:[0-9]{12}:[a-zA-Z-]*/[0-9a-zA-Z._-]+

Required: Yes

[TagKeys \(p. 385\)](#)

A list of tag keys you want to remove from a specified Amazon Transcribe resource.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Length Constraints: Minimum length of 1. Maximum length of 128.

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

[BadRequestException](#)

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception Message field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UpdateCallAnalyticsCategory

Service: Amazon Transcribe Service

Updates the call analytics category with new values. The `UpdateCallAnalyticsCategory` operation overwrites all of the existing information with the values that you provide in the request.

Request Syntax

```
{  
    "CategoryName": "string",  
    "Rules": [  
        {  
            "InterruptionFilter": {  
                "AbsoluteTimeRange": {  
                    "EndTime": number,  
                    "First": number,  
                    "Last": number,  
                    "StartTime": number  
                },  
                "Negate": boolean,  
                "ParticipantRole": "string",  
                "RelativeTimeRange": {  
                    "EndPercentage": number,  
                    "First": number,  
                    "Last": number,  
                    "StartPercentage": number  
                },  
                "Threshold": number  
            },  
            "NonTalkTimeFilter": {  
                "AbsoluteTimeRange": {  
                    "EndTime": number,  
                    "First": number,  
                    "Last": number,  
                    "StartTime": number  
                },  
                "Negate": boolean,  
                "RelativeTimeRange": {  
                    "EndPercentage": number,  
                    "First": number,  
                    "Last": number,  
                    "StartPercentage": number  
                },  
                "Threshold": number  
            },  
            "SentimentFilter": {  
                "AbsoluteTimeRange": {  
                    "EndTime": number,  
                    "First": number,  
                    "Last": number,  
                    "StartTime": number  
                },  
                "Negate": boolean,  
                "ParticipantRole": "string",  
                "RelativeTimeRange": {  
                    "EndPercentage": number,  
                    "First": number,  
                    "Last": number,  
                    "StartPercentage": number  
                },  
                "Sentiments": [ "string" ]  
            },  
            "TranscriptFilter": {  
                "AbsoluteTimeRange": {  
                    "EndTime": number,  
                    "First": number,  
                    "Last": number,  
                    "StartTime": number  
                }  
            }  
        }  
    ]  
}
```

```

        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
    },
    "Negate": boolean,
    "ParticipantRole": "string",
    "RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
    },
    "Targets": [ "string" ],
    "TranscriptFilterType": "string"
}
]
}

```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

CategoryName (p. 387)

The name of the analytics category to update. The name is case sensitive. If you try to update a call analytics category with the same name as a previous category you will receive a `ConflictException` error.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Rules (p. 387)

The rules used for the updated analytics category. The rules that you provide in this field replace the ones that are currently being used.

Type: Array of [Rule \(p. 450\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Required: Yes

Response Syntax

```
{
    "CategoryProperties": {
        "CategoryName": "string",
        "CreateTime": number,
        "LastUpdateTime": number,
        "Rules": [
            {
                "InterruptionFilter": {

```

```
"AbsoluteTimeRange": {
    "EndTime": number,
    "First": number,
    "Last": number,
    "StartTime": number
},
"Negate": boolean,
"ParticipantRole": "string",
"RelativeTimeRange": {
    "EndPercentage": number,
    "First": number,
    "Last": number,
    "StartPercentage": number
},
"Threshold": number
},
"NonTalkTimeFilter": {
    "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
},
"Negate": boolean,
"RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
},
"Threshold": number
},
"SentimentFilter": {
    "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
},
"Negate": boolean,
"ParticipantRole": "string",
"RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
},
"Sentiments": [ "string " ]
},
"TranscriptFilter": {
    "AbsoluteTimeRange": {
        "EndTime": number,
        "First": number,
        "Last": number,
        "StartTime": number
},
"Negate": boolean,
"ParticipantRole": "string",
"RelativeTimeRange": {
        "EndPercentage": number,
        "First": number,
        "Last": number,
        "StartPercentage": number
},
"Targets": [ "string " ],
"TranscriptFilterType": "string"
```

```
        }
    ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[CategoryProperties \(p. 388\)](#)

The attributes describing the analytics category. You can see information such as the rules that you've used to update the category and when the category was originally created.

Type: [CategoryProperties \(p. 427\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)

- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UpdateMedicalVocabulary

Service: Amazon Transcribe Service

Updates a vocabulary with new values that you provide in a different text file from the one you used to create the vocabulary. The `UpdateMedicalVocabulary` operation overwrites all of the existing information with the values that you provide in the request.

Request Syntax

```
{  
    "LanguageCode": "string",  
    "VocabularyFileUri": "string",  
    "VocabularyName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

[LanguageCode \(p. 392\)](#)

The language code of the language used for the entries in the updated vocabulary. US English (en-US) is the only valid language code in Amazon Transcribe Medical.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: Yes

[VocabularyFileUri \(p. 392\)](#)

The location in Amazon S3 of the text file that contains your custom vocabulary. The URI must be in the same AWS Region as the resource that you are calling. The following is the format for a URL:

`https://s3.<aws-region>.amazonaws.com/<bucket-name>/<keyprefix>/<objectkey>`

For example:

`https://s3.us-east-1.amazonaws.com/AWSDOC-EXAMPLE-BUCKET/vocab.txt`

For more information about Amazon S3 object names, see [Object Keys](#) in the *Amazon S3 Developer Guide*.

For more information about custom vocabularies in Amazon Transcribe Medical, see [Medical Custom Vocabularies](#).

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (`s3://|http(s*)://`).+

Required: No

[VocabularyName \(p. 392\)](#)

The name of the vocabulary to update. The name is case sensitive. If you try to update a vocabulary with the same name as a vocabulary you've already made, you get a `ConflictException` error.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^[0-9a-zA-Z._-]+`

Required: Yes

Response Syntax

```
{  
    "LanguageCode": "string",  
    "LastModifiedTime": number,  
    "VocabularyName": "string",  
    "VocabularyState": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[LanguageCode \(p. 393\)](#)

The language code for the language of the text file used to update the custom vocabulary. US English (en-US) is the only language supported in Amazon Transcribe Medical.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

[LastModifiedTime \(p. 393\)](#)

The date and time that the vocabulary was updated.

Type: Timestamp

[VocabularyName \(p. 393\)](#)

The name of the updated vocabulary.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^[0-9a-zA-Z._-]+`

[VocabularyState \(p. 393\)](#)

The processing state of the update to the vocabulary. When the `VocabularyState` field is `READY`, the vocabulary is ready to be used in a `StartMedicalTranscriptionJob` request.

Type: String

Valid Values: PENDING | READY | FAILED

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UpdateVocabulary

Service: Amazon Transcribe Service

Updates an existing vocabulary with new values. The `UpdateVocabulary` operation overwrites all of the existing information with the values that you provide in the request.

Request Syntax

```
{  
    "LanguageCode": "string",  
    "Phrases": [ "string" ],  
    "VocabularyFileUri": "string",  
    "VocabularyName": "string"  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

LanguageCode (p. 395)

The language code of the vocabulary entries. For a list of languages and their corresponding language codes, see [What is Amazon Transcribe? \(p. 1\)](#).

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: Yes

Phrases (p. 395)

An array of strings containing the vocabulary entries.

Type: Array of strings

Length Constraints: Minimum length of 0. Maximum length of 256.

Pattern: .+

Required: No

VocabularyFileUri (p. 395)

The S3 location of the text file that contains the definition of the custom vocabulary. The URI must be in the same region as the API endpoint that you are calling. The general form is

`https://s3.<aws-region>.amazonaws.com/<AWSDOC-EXAMPLE-BUCKET>/<keyprefix>/<objectkey>`

For example:

`https://s3.us-east-1.amazonaws.com/AWSDOC-EXAMPLE-BUCKET/vocab.txt`

For more information about S3 object names, see [Object Keys](#) in the *Amazon S3 Developer Guide*.

For more information about custom vocabularies, see [Custom Vocabularies](#).

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (s3://|http(s*)://).+

Required: No

[VocabularyName \(p. 395\)](#)

The name of the vocabulary to update. The name is case sensitive. If you try to update a vocabulary with the same name as a previous vocabulary you will receive a `ConflictException` error.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Response Syntax

```
{  
    "LanguageCode": "string",  
    "LastModifiedTime": number,  
    "VocabularyName": "string",  
    "VocabularyState": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[LanguageCode \(p. 396\)](#)

The language code of the vocabulary entries.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

[LastModifiedTime \(p. 396\)](#)

The date and time that the vocabulary was updated.

Type: Timestamp

[VocabularyName \(p. 396\)](#)

The name of the vocabulary that was updated.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

[VocabularyState \(p. 396\)](#)

The processing state of the vocabulary. When the `VocabularyState` field contains `READY` the vocabulary is ready to be used in a `StartTranscriptionJob` request.

Type: String

Valid Values: `PENDING` | `READY` | `FAILED`

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception `Message` field for more information.

HTTP Status Code: 400

ConflictException

There is already a resource with that name.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)

- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

UpdateVocabularyFilter

Service: Amazon Transcribe Service

Updates a vocabulary filter with a new list of filtered words.

Request Syntax

```
{  
    "VocabularyFilterFileUri": "string",  
    "VocabularyFilterName": "string",  
    "Words": [ "string" ]  
}
```

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters \(p. 492\)](#).

The request accepts the following data in JSON format.

VocabularyFilterFileUri (p. 399)

The Amazon S3 location of a text file used as input to create the vocabulary filter. Only use characters from the character set defined for custom vocabularies. For a list of character sets, see [Character Sets for Custom Vocabularies](#).

The specified file must be less than 50 KB of UTF-8 characters.

If you provide the location of a list of words in the `VocabularyFilterFileUri` parameter, you can't use the `Words` parameter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (s3://|http(s*)://).+

Required: No

VocabularyFilterName (p. 399)

The name of the vocabulary filter to update. If you try to update a vocabulary filter with the same name as another vocabulary filter, you get a `ConflictException` error.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: Yes

Words (p. 399)

The words to use in the vocabulary filter. Only use characters from the character set defined for custom vocabularies. For a list of character sets, see [Character Sets for Custom Vocabularies](#).

If you provide a list of words in the `Words` parameter, you can't use the `VocabularyFilterFileUri` parameter.

Type: Array of strings

Array Members: Minimum number of 1 item.

Length Constraints: Minimum length of 1. Maximum length of 256.

Required: No

Response Syntax

```
{  
    "LanguageCode": "string",  
    "LastModifiedTime": number,  
    "VocabularyFilterName": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

[LanguageCode \(p. 400\)](#)

The language code of the words in the vocabulary filter.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

[LastModifiedTime \(p. 400\)](#)

The date and time that the vocabulary filter was updated.

Type: Timestamp

[VocabularyFilterName \(p. 400\)](#)

The name of the updated vocabulary filter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

Your request didn't pass one or more validation tests. For example, if the entity that you're trying to delete doesn't exist or if it is in a non-terminal state (for example, it's "in progress"). See the exception Message field for more information.

HTTP Status Code: 400

InternalFailureException

There was an internal error. Check the error message and try your request again.

HTTP Status Code: 500

LimitExceededException

Either you have sent too many requests or your input file is too long. Wait before you resend your request, or use a smaller file and resend the request.

HTTP Status Code: 400

NotFoundException

We can't find the requested resource. Check the name and try your request again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

Amazon Transcribe Streaming Service

The following actions are supported by Amazon Transcribe Streaming Service:

- [StartMedicalStreamTranscription \(p. 402\)](#)
- [StartStreamTranscription \(p. 408\)](#)

StartMedicalStreamTranscription

Service: Amazon Transcribe Streaming Service

Starts a bidirectional HTTP/2 stream where audio is streamed to Amazon Transcribe Medical and the transcription results are streamed to your application.

Request Syntax

```
POST /medical-stream-transcription HTTP/2
x-amzn-transcribe-language-code: LanguageCode
x-amzn-transcribe-sample-rate: MediaSampleRateHertz
x-amzn-transcribe-media-encoding: MediaEncoding
x-amzn-transcribe-vocabulary-name: VocabularyName
x-amzn-transcribe-specialty: Specialty
x-amzn-transcribe-type: Type
x-amzn-transcribe-show-speaker-label: ShowSpeakerLabel
x-amzn-transcribe-session-id: SessionId
x-amzn-transcribe-enable-channel-identification: EnableChannelIdentification
x-amzn-transcribe-number-of-channels: NumberOfChannels
x-amzn-transcribe-content-identification-type: ContentIdentificationType
Content-type: application/json

{
    "AudioStream": {
        "AudioEvent": {
            "AudioChunk": blob
        }
    }
}
```

URI Request Parameters

The request uses the following URI parameters.

[ContentIdentificationType \(p. 402\)](#)

Set this field to `PHI` to identify personal health information in the transcription output.

Valid Values: `PHI`

[EnableChannelIdentification \(p. 402\)](#)

When `true`, instructs Amazon Transcribe Medical to process each audio channel separately and then merge the transcription output of each channel into a single transcription.

Amazon Transcribe Medical also produces a transcription of each item. An item includes the start time, end time, and any alternative transcriptions.

You can't set both `ShowSpeakerLabel` and `EnableChannelIdentification` in the same request. If you set both, your request returns a `BadRequestException`.

[LanguageCode \(p. 402\)](#)

Indicates the source language used in the input audio stream. For Amazon Transcribe Medical, this is US English (en-US).

Valid Values: `en-US` | `en-GB` | `es-US` | `fr-CA` | `fr-FR` | `en-AU` | `it-IT` | `de-DE` | `pt-BR` | `ja-JP` | `ko-KR` | `zh-CN`

Required: Yes

[MediaEncoding \(p. 402\)](#)

The encoding used for the input audio.

Valid Values: `pcm` | `ogg-opus` | `flac`

Required: Yes

[MediaSampleRateHertz \(p. 402\)](#)

The sample rate of the input audio in Hertz.

Valid Range: Minimum value of 8000. Maximum value of 48000.

Required: Yes

[NumberOfChannels \(p. 402\)](#)

The number of channels that are in your audio stream.

Valid Range: Minimum value of 2.

[SessionId \(p. 402\)](#)

Optional. An identifier for the transcription session. If you don't provide a session ID, Amazon Transcribe generates one for you and returns it in the response.

Length Constraints: Fixed length of 36.

Pattern: `[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}`

[ShowSpeakerLabel \(p. 402\)](#)

When `true`, enables speaker identification in your real-time stream.

[Specialty \(p. 402\)](#)

The medical specialty of the clinician or provider.

Valid Values: `PRIMARYCARE` | `CARDIOLOGY` | `NEUROLOGY` | `ONCOLOGY` | `RADIOLOGY` | `UROLOGY`

Required: Yes

[Type \(p. 402\)](#)

The type of input audio. Choose `DICTATION` for a provider dictating patient notes. Choose `CONVERSATION` for a dialogue between a patient and one or more medical professionals.

Valid Values: `CONVERSATION` | `DICTATION`

Required: Yes

[VocabularyName \(p. 402\)](#)

The name of the medical custom vocabulary to use when processing the real-time stream.

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^ [0-9a-zA-Z._-]+`

Request Body

The request accepts the following data in JSON format.

[AudioStream \(p. 402\)](#)

Represents the audio stream from your application to Amazon Transcribe.

Type: [AudioStream \(p. 471\)](#) object

Required: Yes

Response Syntax

```
HTTP/2 200
x-amzn-request-id: RequestId
x-amzn-transcribe-language-code: LanguageCode
x-amzn-transcribe-sample-rate: MediaSampleRateHertz
x-amzn-transcribe-media-encoding: MediaEncoding
x-amzn-transcribe-vocabulary-name: VocabularyName
x-amzn-transcribe-specialty: Specialty
x-amzn-transcribe-type: Type
x-amzn-transcribe-show-speaker-label: ShowSpeakerLabel
x-amzn-transcribe-session-id: SessionId
x-amzn-transcribe-enable-channel-identification: EnableChannelIdentification
x-amzn-transcribe-number-of-channels: NumberOfChannels
x-amzn-transcribe-content-identification-type: ContentIdentificationType
Content-type: application/json

{
    "TranscriptResultStream": {
        "BadRequestException": {
        },
        "ConflictException": {
        },
        "InternalFailureException": {
        },
        "LimitExceededException": {
        },
        "ServiceUnavailableException": {
        },
        "TranscriptEvent": {
            "Transcript": {
                "Results": [
                    {
                        "Alternatives": [
                            {
                                "Entities": [
                                    {
                                        "Category": "string",
                                        "Confidence": number,
                                        "Content": "string",
                                        "EndTime": number,
                                        "StartTime": number
                                    }
                                ],
                                "Items": [
                                    {
                                        "Confidence": number,
                                        "Content": "string",
                                        "EndTime": number,
                                        "Speaker": "string",
                                        "StartTime": number,
                                        "Type": "string"
                                    }
                                ],
                                "Transcript": "string"
                            }
                        ],
                        "ChannelId": "string",
                        "EndTime": number,
                        "IsPartial": boolean,
                        "ResultId": "string",
                        "StartTime": number
                    }
                ]
            }
        }
    }
}
```

```
        }
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The response returns the following HTTP headers.

[ContentIdentificationType \(p. 404\)](#)

If the value is `PHI`, indicates that you've configured your stream to identify personal health information.

Valid Values: `PHI`

[EnableChannelIdentification \(p. 404\)](#)

Shows whether channel identification has been enabled in the stream.

[LanguageCode \(p. 404\)](#)

The language code for the response transcript. For Amazon Transcribe Medical, this is US English (`en-US`).

Valid Values: `en-US` | `en-GB` | `es-US` | `fr-CA` | `fr-FR` | `en-AU` | `it-IT` | `de-DE` | `pt-BR` | `ja-JP` | `ko-KR` | `zh-CN`

[MediaEncoding \(p. 404\)](#)

The encoding used for the input audio stream.

Valid Values: `pcm` | `ogg-opus` | `flac`

[MediaSampleRateHertz \(p. 404\)](#)

The sample rate of the input audio in Hertz.

Valid Range: Minimum value of 8000. Maximum value of 48000.

[NumberOfChannels \(p. 404\)](#)

The number of channels identified in the stream.

Valid Range: Minimum value of 2.

[RequestId \(p. 404\)](#)

An identifier for the streaming transcription.

[SessionId \(p. 404\)](#)

Optional. An identifier for the transcription session. If you don't provide a session ID, Amazon Transcribe generates one for you and returns it in the response.

Length Constraints: Fixed length of 36.

Pattern: `[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}`

[ShowSpeakerLabel \(p. 404\)](#)

Shows whether speaker identification was enabled in the stream.

[Specialty \(p. 404\)](#)

The specialty in the medical domain.

Valid Values: PRIMARYCARE | CARDIOLOGY | NEUROLOGY | ONCOLOGY | RADIOLOGY | UROLOGY

[Type \(p. 404\)](#)

The type of audio that was transcribed.

Valid Values: CONVERSATION | DICTATION

[VocabularyName \(p. 404\)](#)

The name of the vocabulary used when processing the stream.

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

The following data is returned in JSON format by the service.

[TranscriptResultStream \(p. 404\)](#)

Represents the stream of transcription events from Amazon Transcribe Medical to your application.

Type: [MedicalTranscriptResultStream \(p. 484\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

One or more arguments to the StartStreamTranscription or StartMedicalStreamTranscription operation was invalid. For example, MediaEncoding was not set to a valid encoding, or LanguageCode was not set to a valid code. Check the parameters and try your request again.

HTTP Status Code: 400

ConflictException

A new stream started with the same session ID. The current stream has been terminated.

HTTP Status Code: 409

InternalFailureException

A problem occurred while processing the audio. Amazon Transcribe or Amazon Transcribe Medical terminated processing. Try your request again.

HTTP Status Code: 500

LimitExceededException

You have exceeded the maximum number of concurrent transcription streams, are starting transcription streams too quickly, or the maximum audio length of 4 hours. Wait until a stream has finished processing, or break your audio stream into smaller chunks and try your request again.

HTTP Status Code: 429

ServiceUnavailableException

Service is currently unavailable. Try your request later.

HTTP Status Code: 503

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS Command Line Interface](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK for Python](#)
- [AWS SDK for Ruby V3](#)

StartStreamTranscription

Service: Amazon Transcribe Streaming Service

Starts a bidirectional HTTP/2 stream where audio is streamed to Amazon Transcribe and the transcription results are streamed to your application.

The following are encoded as HTTP/2 headers:

- x-amzn-transcribe-language-code
- x-amzn-transcribe-media-encoding
- x-amzn-transcribe-sample-rate
- x-amzn-transcribe-session-id

See the [SDK for Go API Reference](#) for more detail.

Request Syntax

```
POST /stream-transcription HTTP/2
x-amzn-transcribe-language-code: LanguageCode
x-amzn-transcribe-sample-rate: MediaSampleRateHertz
x-amzn-transcribe-media-encoding: MediaEncoding
x-amzn-transcribe-vocabulary-name: VocabularyName
x-amzn-transcribe-session-id: SessionId
x-amzn-transcribe-vocabulary-filter-name: VocabularyFilterName
x-amzn-transcribe-vocabulary-filter-method: VocabularyFilterMethod
x-amzn-transcribe-show-speaker-label: ShowSpeakerLabel
x-amzn-transcribe-enable-channel-identification: EnableChannelIdentification
x-amzn-transcribe-number-of-channels: NumberOfChannels
x-amzn-transcribe-enable-partial-results-stabilization: EnablePartialResultsStabilization
x-amzn-transcribe-partial-results-stability: PartialResultsStability
x-amzn-transcribe-content-identification-type: ContentIdentificationType
x-amzn-transcribe-content-redaction-type: ContentRedactionType
x-amzn-transcribe-pii-entity-types: PiiEntityTypes
Content-type: application/json

{
    "AudioStream": {
        "AudioEvent": {
            "AudioChunk": blob
        }
    }
}
```

URI Request Parameters

The request uses the following URI parameters.

[ContentIdentificationType \(p. 408\)](#)

Set this field to PII to identify personally identifiable information (PII) in the transcription output. Content identification is performed only upon complete transcription of the audio segments.

You can't set both ContentIdentificationType and ContentRedactionType in the same request. If you set both, your request returns a BadRequestException.

Valid Values: PII

[ContentRedactionType \(p. 408\)](#)

Set this field to PII to redact personally identifiable information (PII) in the transcription output. Content redaction is performed only upon complete transcription of the audio segments.

You can't set both `ContentRedactionType` and `ContentIdentificationType` in the same request. If you set both, your request returns a `BadRequestException`.

Valid Values: PII

[EnableChannelIdentification \(p. 408\)](#)

When `true`, instructs Amazon Transcribe to process each audio channel separately and then merge the transcription output of each channel into a single transcription.

Amazon Transcribe also produces a transcription of each item. An item includes the start time, end time, and any alternative transcriptions.

You can't set both `ShowSpeakerLabel` and `EnableChannelIdentification` in the same request. If you set both, your request returns a `BadRequestException`.

[EnablePartialResultsStabilization \(p. 408\)](#)

When `true`, instructs Amazon Transcribe to present transcription results that have the partial results stabilized. Normally, any word or phrase from one partial result can change in a subsequent partial result. With partial results stabilization enabled, only the last few words of one partial result can change in another partial result.

[LanguageCode \(p. 408\)](#)

Indicates the source language used in the input audio stream.

Valid Values: en-US | en-GB | es-US | fr-CA | fr-FR | en-AU | it-IT | de-DE | pt-BR | ja-JP | ko-KR | zh-CN

Required: Yes

[MediaEncoding \(p. 408\)](#)

The encoding used for the input audio.

Valid Values: pcm | ogg-opus | flac

Required: Yes

[MediaSampleRateHertz \(p. 408\)](#)

The sample rate, in Hertz, of the input audio. We suggest that you use 8,000 Hz for low quality audio and 16,000 Hz for high quality audio.

Valid Range: Minimum value of 8000. Maximum value of 48000.

Required: Yes

[NumberOfChannels \(p. 408\)](#)

The number of channels that are in your audio stream.

Valid Range: Minimum value of 2.

[PartialResultsStability \(p. 408\)](#)

You can use this field to set the stability level of the transcription results. A higher stability level means that the transcription results are less likely to change. Higher stability levels can come with lower overall transcription accuracy.

Valid Values: high | medium | low

[PiiEntityTypes \(p. 408\)](#)

List the PII entity types you want to identify or redact. In order to specify entity types, you must have either `ContentIdentificationType` or `ContentRedactionType` enabled.

PiiEntityTypes must be comma-separated; the available values are: BANK_ACCOUNT_NUMBER, BANK_ROUTING, CREDIT_DEBIT_NUMBER, CREDIT_DEBIT_CVV, CREDIT_DEBIT_EXPIRY, PIN, EMAIL, ADDRESS, NAME, PHONE, SSN, and ALL.

PiiEntityTypes is an optional parameter with a default value of ALL.

Length Constraints: Minimum length of 1. Maximum length of 300.

Pattern: ^[A-Z_,]+

[SessionId \(p. 408\)](#)

A identifier for the transcription session. Use this parameter when you want to retry a session. If you don't provide a session ID, Amazon Transcribe will generate one for you and return it in the response.

Length Constraints: Fixed length of 36.

Pattern: [a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}

[ShowSpeakerLabel \(p. 408\)](#)

When true, enables speaker identification in your real-time stream.

[VocabularyFilterMethod \(p. 408\)](#)

The manner in which you use your vocabulary filter to filter words in your transcript. Remove removes filtered words from your transcription results. Mask masks filtered words with a *** in your transcription results. Tag keeps the filtered words in your transcription results and tags them. The tag appears as VocabularyFilterMatch equal to True

Valid Values: remove | mask | tag

[VocabularyFilterName \(p. 408\)](#)

The name of the vocabulary filter you've created that is unique to your account. Provide the name in this field to successfully use it in a stream.

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

[VocabularyName \(p. 408\)](#)

The name of the vocabulary to use when processing the transcription job.

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

[Request Body](#)

The request accepts the following data in JSON format.

[AudioStream \(p. 408\)](#)

PCM-encoded stream of audio blobs. The audio stream is encoded as an HTTP/2 data frame.

Type: [AudioStream \(p. 471\)](#) object

Required: Yes

Response Syntax

```

HTTP/2 200
x-amzn-request-id: RequestId
x-amzn-transcribe-language-code: LanguageCode
x-amzn-transcribe-sample-rate: MediaSampleRateHertz
x-amzn-transcribe-media-encoding: MediaEncoding
x-amzn-transcribe-vocabulary-name: VocabularyName
x-amzn-transcribe-session-id: SessionId
x-amzn-transcribe-vocabulary-filter-name: VocabularyFilterName
x-amzn-transcribe-vocabulary-filter-method: VocabularyFilterMethod
x-amzn-transcribe-show-speaker-label: ShowSpeakerLabel
x-amzn-transcribe-enable-channel-identification: EnableChannelIdentification
x-amzn-transcribe-number-of-channels: NumberOfChannels
x-amzn-transcribe-enable-partial-results-stabilization: EnablePartialResultsStabilization
x-amzn-transcribe-partial-results-stability: PartialResultsStability
x-amzn-transcribe-content-identification-type: ContentIdentificationType
x-amzn-transcribe-content-redaction-type: ContentRedactionType
x-amzn-transcribe-pii-entity-types: PiiEntityTypes
Content-type: application/json

{
    "TranscriptResultStream": {
        "BadRequestException": {
        },
        "ConflictException": {
        },
        "InternalFailureException": {
        },
        "LimitExceededException": {
        },
        "ServiceUnavailableException": {
        },
        "TranscriptEvent": {
            "Transcript": {
                "Results": [
                    {
                        "Alternatives": [
                            {
                                "Entities": [
                                    {
                                        "Category": "string",
                                        "Confidence": number,
                                        "Content": "string",
                                        "EndTime": number,
                                        "StartTime": number,
                                        "Type": "string"
                                    }
                                ],
                                "Items": [
                                    {
                                        "Confidence": number,
                                        "Content": "string",
                                        "EndTime": number,
                                        "Speaker": "string",
                                        "Stable": boolean,
                                        "StartTime": number,
                                        "Type": "string",
                                        "VocabularyFilterMatch": boolean
                                    }
                                ],
                                "Transcript": "string"
                            }
                        ],
                        "ChannelId": "string",
                    }
                ]
            }
        }
    }
}

```

```
        "EndTime": number,
        "IsPartial": boolean,
        "ResultId": "string",
        "StartTime": number
    }
}
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The response returns the following HTTP headers.

[ContentIdentificationType \(p. 411\)](#)

Shows whether content identification was enabled in this stream.

Valid Values: PII

[ContentRedactionType \(p. 411\)](#)

Shows whether content redaction was enabled in this stream.

Valid Values: PII

[EnableChannelIdentification \(p. 411\)](#)

Shows whether channel identification has been enabled in the stream.

[EnablePartialResultsStabilization \(p. 411\)](#)

Shows whether partial results stabilization has been enabled in the stream.

[LanguageCode \(p. 411\)](#)

The language code for the input audio stream.

Valid Values: en-US | en-GB | es-US | fr-CA | fr-FR | en-AU | it-IT | de-DE | pt-BR | ja-JP | ko-KR | zh-CN

[MediaEncoding \(p. 411\)](#)

The encoding used for the input audio stream.

Valid Values: pcm | ogg-opus | flac

[MediaSampleRateHertz \(p. 411\)](#)

The sample rate for the input audio stream. Use 8,000 Hz for low quality audio and 16,000 Hz for high quality audio.

Valid Range: Minimum value of 8000. Maximum value of 48000.

[NumberOfChannels \(p. 411\)](#)

The number of channels identified in the stream.

Valid Range: Minimum value of 2.

[PartialResultsStability \(p. 411\)](#)

If partial results stabilization has been enabled in the stream, shows the stability level.

Valid Values: `high` | `medium` | `low`

[PiiEntityTypes \(p. 411\)](#)

Lists the PII entity types you specified in your request.

Length Constraints: Minimum length of 1. Maximum length of 300.

Pattern: `^[A-Z_,]+`

[RequestId \(p. 411\)](#)

An identifier for the streaming transcription.

[SessionId \(p. 411\)](#)

An identifier for a specific transcription session.

Length Constraints: Fixed length of 36.

Pattern: `[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}`

[ShowSpeakerLabel \(p. 411\)](#)

Shows whether speaker identification was enabled in the stream.

[VocabularyFilterMethod \(p. 411\)](#)

The vocabulary filtering method used in the real-time stream.

Valid Values: `remove` | `mask` | `tag`

[VocabularyFilterName \(p. 411\)](#)

The name of the vocabulary filter used in your real-time stream.

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^[0-9a-zA-Z._-]+`

[VocabularyName \(p. 411\)](#)

The name of the vocabulary used when processing the stream.

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^[0-9a-zA-Z._-]+`

The following data is returned in JSON format by the service.

[TranscriptResultStream \(p. 411\)](#)

Represents the stream of transcription events from Amazon Transcribe to your application.

Type: [TranscriptResultStream \(p. 490\)](#) object

Errors

For information about the errors that are common to all actions, see [Common Errors \(p. 491\)](#).

BadRequestException

One or more arguments to the `StartStreamTranscription` or `StartMedicalStreamTranscription` operation was invalid. For example, `MediaEncoding` was

not set to a valid encoding, or `LanguageCode` was not set to a valid code. Check the parameters and try your request again.

HTTP Status Code: 400

ConflictException

A new stream started with the same session ID. The current stream has been terminated.

HTTP Status Code: 409

InternalFailureException

A problem occurred while processing the audio. Amazon Transcribe or Amazon Transcribe Medical terminated processing. Try your request again.

HTTP Status Code: 500

LimitExceededException

You have exceeded the maximum number of concurrent transcription streams, are starting transcription streams too quickly, or the maximum audio length of 4 hours. Wait until a stream has finished processing, or break your audio stream into smaller chunks and try your request again.

HTTP Status Code: 429

ServiceUnavailableException

Service is currently unavailable. Try your request later.

HTTP Status Code: 503

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Java](#)
- [AWS SDK for Ruby V3](#)

Data Types

The following data types are supported by Amazon Transcribe Service:

- [AbsoluteTimeRange \(p. 417\)](#)
- [CallAnalyticsJob \(p. 419\)](#)
- [CallAnalyticsJobSettings \(p. 423\)](#)
- [CallAnalyticsJobSummary \(p. 425\)](#)
- [CategoryProperties \(p. 427\)](#)
- [ChannelDefinition \(p. 428\)](#)
- [ContentRedaction \(p. 429\)](#)
- [InputDataConfig \(p. 430\)](#)
- [InterruptionFilter \(p. 431\)](#)
- [JobExecutionSettings \(p. 433\)](#)
- [LanguageModel \(p. 434\)](#)
- [Media \(p. 436\)](#)

- [MedicalTranscript \(p. 437\)](#)
- [MedicalTranscriptionJob \(p. 438\)](#)
- [MedicalTranscriptionJobSummary \(p. 442\)](#)
- [MedicalTranscriptionSetting \(p. 444\)](#)
- [ModelSettings \(p. 446\)](#)
- [NonTalkTimeFilter \(p. 447\)](#)
- [RelativeTimeRange \(p. 448\)](#)
- [Rule \(p. 450\)](#)
- [SentimentFilter \(p. 451\)](#)
- [Settings \(p. 453\)](#)
- [Tag \(p. 455\)](#)
- [Transcript \(p. 456\)](#)
- [TranscriptFilter \(p. 457\)](#)
- [TranscriptionJob \(p. 459\)](#)
- [TranscriptionJobSummary \(p. 463\)](#)
- [VocabularyFilterInfo \(p. 466\)](#)
- [VocabularyInfo \(p. 467\)](#)

The following data types are supported by Amazon Transcribe Streaming Service:

- [Alternative \(p. 469\)](#)
- [AudioEvent \(p. 470\)](#)
- [AudioStream \(p. 471\)](#)
- [Entity \(p. 472\)](#)
- [Item \(p. 474\)](#)
- [MedicalAlternative \(p. 476\)](#)
- [MedicalEntity \(p. 477\)](#)
- [MedicalItem \(p. 478\)](#)
- [MedicalResult \(p. 480\)](#)
- [MedicalTranscript \(p. 482\)](#)
- [MedicalTranscriptEvent \(p. 483\)](#)
- [MedicalTranscriptResultStream \(p. 484\)](#)
- [Result \(p. 486\)](#)
- [Transcript \(p. 488\)](#)
- [TranscriptEvent \(p. 489\)](#)
- [TranscriptResultStream \(p. 490\)](#)

Amazon Transcribe Service

The following data types are supported by Amazon Transcribe Service:

- [AbsoluteTimeRange \(p. 417\)](#)
- [CallAnalyticsJob \(p. 419\)](#)
- [CallAnalyticsJobSettings \(p. 423\)](#)
- [CallAnalyticsJobSummary \(p. 425\)](#)
- [CategoryProperties \(p. 427\)](#)

- [ChannelDefinition \(p. 428\)](#)
- [ContentRedaction \(p. 429\)](#)
- [InputDataConfig \(p. 430\)](#)
- [InterruptionFilter \(p. 431\)](#)
- [JobExecutionSettings \(p. 433\)](#)
- [LanguageModel \(p. 434\)](#)
- [Media \(p. 436\)](#)
- [MedicalTranscript \(p. 437\)](#)
- [MedicalTranscriptionJob \(p. 438\)](#)
- [MedicalTranscriptionJobSummary \(p. 442\)](#)
- [MedicalTranscriptionSetting \(p. 444\)](#)
- [ModelSettings \(p. 446\)](#)
- [NonTalkTimeFilter \(p. 447\)](#)
- [RelativeTimeRange \(p. 448\)](#)
- [Rule \(p. 450\)](#)
- [SentimentFilter \(p. 451\)](#)
- [Settings \(p. 453\)](#)
- [Tag \(p. 455\)](#)
- [Transcript \(p. 456\)](#)
- [TranscriptFilter \(p. 457\)](#)
- [TranscriptionJob \(p. 459\)](#)
- [TranscriptionJobSummary \(p. 463\)](#)
- [VocabularyFilterInfo \(p. 466\)](#)
- [VocabularyInfo \(p. 467\)](#)

AbsoluteTimeRange

Service: Amazon Transcribe Service

A time range, set in seconds, between two points in the call.

Contents

EndTime

A value that indicates the end of the time range in milliseconds. To set absolute time range, you must specify a start time and an end time. For example, if you specify the following values:

- StartTime - 10000
- Endtime - 50000

The time range is set between 10,000 milliseconds and 50,000 milliseconds into the call.

Type: Long

Valid Range: Minimum value of 0. Maximum value of 14400000.

Required: No

First

A time range from the beginning of the call to the value that you've specified. For example, if you specify 100000, the time range is set to the first 100,000 milliseconds of the call.

Type: Long

Valid Range: Minimum value of 0. Maximum value of 14400000.

Required: No

Last

A time range from the value that you've specified to the end of the call. For example, if you specify 100000, the time range is set to the last 100,000 milliseconds of the call.

Type: Long

Valid Range: Minimum value of 0. Maximum value of 14400000.

Required: No

StartTime

A value that indicates the beginning of the time range in seconds. To set absolute time range, you must specify a start time and an end time. For example, if you specify the following values:

- StartTime - 10000
- Endtime - 50000

The time range is set between 10,000 milliseconds and 50,000 milliseconds into the call.

Type: Long

Valid Range: Minimum value of 0. Maximum value of 14400000.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

CallAnalyticsJob

Service: Amazon Transcribe Service

Describes an asynchronous analytics job that was created with the `StartAnalyticsJob` operation.

Contents

CallAnalyticsJobName

The name of the call analytics job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

CallAnalyticsJobStatus

The status of the analytics job.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Required: No

ChannelDefinitions

Shows numeric values to indicate the channel assigned to the agent's audio and the channel assigned to the customer's audio.

Type: Array of [ChannelDefinition \(p. 428\)](#) objects

Array Members: Fixed number of 2 items.

Required: No

CompletionTime

A timestamp that shows when the analytics job was completed.

Type: Timestamp

Required: No

CreationTime

A timestamp that shows when the analytics job was created.

Type: Timestamp

Required: No

DataAccessRoleArn

The Amazon Resource Number (ARN) that you use to get access to the analytics job.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: ^arn:(aws|aws-cn|aws-us-gov|aws-iso-{0,1}[a-z]{0,1}):iam::[0-9]{0,63}:role/[A-Za-z0-9:_/+=@.-]{0,1024}\$

Required: No

FailureReason

If the `AnalyticsJobStatus` is `FAILED`, this field contains information about why the job failed.

The `FailureReason` field can contain one of the following values:

- **Unsupported media format:** The media format specified in the `MediaFormat` field of the request isn't valid. See the description of the `MediaFormat` field for a list of valid values.
- **The media format provided does not match the detected media format:** The media format of the audio file doesn't match the format specified in the `MediaFormat` field in the request. Check the media format of your media file and make sure the two values match.
- **Invalid sample rate for audio file:** The sample rate specified in the `MediaSampleRateHertz` of the request isn't valid. The sample rate must be between 8,000 and 48,000 Hertz.
- **The sample rate provided does not match the detected sample rate:** The sample rate in the audio file doesn't match the sample rate specified in the `MediaSampleRateHertz` field in the request. Check the sample rate of your media file and make sure that the two values match.
- **Invalid file size: file size too large:** The size of your audio file is larger than what Amazon Transcribe Medical can process. For more information, see *Guidelines and Quotas* in the Amazon Transcribe Medical Guide.
- **Invalid number of channels: number of channels too large:** Your audio contains more channels than Amazon Transcribe Medical is configured to process. To request additional channels, see Amazon Transcribe Medical Endpoints and Quotas in the [Amazon Web Services General Reference](#).

Type: String

Required: No

IdentifiedLanguageScore

A value between zero and one that Amazon Transcribe assigned to the language that it identified in the source audio. This value appears only when you don't provide a single language code. Larger values indicate that Amazon Transcribe has higher confidence in the language that it identified

Type: Float

Required: No

LanguageCode

If you know the language spoken between the customer and the agent, specify a language code for this field.

If you don't know the language, you can leave this field blank, and Amazon Transcribe will use machine learning to automatically identify the language. To improve the accuracy of language identification, you can provide an array containing the possible language codes for the language spoken in your audio. Refer to [Supported languages and language-specific features](#) for additional information.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CI | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR |

ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: No

Media

Describes the input media file in a transcription request.

Type: [Media \(p. 436\)](#) object

Required: No

MediaFormat

The format of the input audio file. Note: for call analytics jobs, only the following media formats are supported: MP3, MP4, WAV, FLAC, OGG, and WebM.

Type: String

Valid Values: mp3 | mp4 | wav | flac | ogg | amr | webm

Required: No

MediaSampleRateHertz

The sample rate, in Hertz, of the audio.

Type: Integer

Valid Range: Minimum value of 8000. Maximum value of 48000.

Required: No

Settings

Provides information about the settings used to run a transcription job.

Type: [CallAnalyticsJobSettings \(p. 423\)](#) object

Required: No

StartTime

A timestamp that shows when the analytics job started processing.

Type: Timestamp

Required: No

Transcript

Identifies the location of a transcription.

Type: [Transcript \(p. 456\)](#) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)

- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

CallAnalyticsJobSettings

Service: Amazon Transcribe Service

Provides optional settings for the `CallAnalyticsJob` operation.

Contents

ContentRedaction

Settings for content redaction within a transcription job.

Type: [ContentRedaction \(p. 429\)](#) object

Required: No

LanguageModelName

The structure used to describe a custom language model.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

LanguageOptions

When you run a call analytics job, you can specify the language spoken in the audio, or you can have Amazon Transcribe identify the language for you.

To specify a language, specify an array with one language code. If you don't know the language, you can leave this field blank and Amazon Transcribe will use machine learning to identify the language for you. To improve the ability of Amazon Transcribe to correctly identify the language, you can provide an array of the languages that can be present in the audio. Refer to [Supported languages and language-specific features](#) for additional information.

Type: Array of strings

Array Members: Minimum number of 1 item.

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: No

VocabularyFilterMethod

Set to `mask` to remove filtered text from the transcript and replace it with three asterisks ("***") as placeholder text. Set to `remove` to remove filtered text from the transcript without using placeholder text. Set to `tag` to mark the word in the transcription output that matches the vocabulary filter. When you set the filter method to `tag`, the words matching your vocabulary filter are not masked or removed.

Type: String

Valid Values: `remove` | `mask` | `tag`

Required: No

VocabularyFilterName

The name of the vocabulary filter to use when running a call analytics job. The filter that you specify must have the same language code as the analytics job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

VocabularyName

The name of a vocabulary to use when processing the call analytics job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

CallAnalyticsJobSummary

Service: Amazon Transcribe Service

Provides summary information about a call analytics job.

Contents

CallAnalyticsJobName

The name of the call analytics job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

CallAnalyticsJobStatus

The status of the call analytics job.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Required: No

CompletionTime

A timestamp that shows when the job was completed.

Type: Timestamp

Required: No

CreationTime

A timestamp that shows when the call analytics job was created.

Type: Timestamp

Required: No

FailureReason

If the CallAnalyticsJobStatus is FAILED, a description of the error.

Type: String

Required: No

LanguageCode

The language of the transcript in the source audio file.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: No

StartTime

A timestamp that shows when the job began processing.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

CategoryProperties

Service: Amazon Transcribe Service

An object that contains the rules and additional information about a call analytics category.

Contents

CategoryName

The name of the call analytics category.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

CreateTime

A timestamp that shows when the call analytics category was created.

Type: Timestamp

Required: No

LastUpdateTime

A timestamp that shows when the call analytics category was most recently updated.

Type: Timestamp

Required: No

Rules

The rules used to create a call analytics category.

Type: Array of [Rule \(p. 450\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ChannelDefinition

Service: Amazon Transcribe Service

For a call analytics job, an object that indicates the audio channel that belongs to the agent and the audio channel that belongs to the customer.

Contents

ChannelId

A value that indicates the audio channel.

Type: Integer

Valid Range: Minimum value of 0. Maximum value of 1.

Required: No

ParticipantRole

Indicates whether the person speaking on the audio channel is the agent or customer.

Type: String

Valid Values: AGENT | CUSTOMER

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ContentRedaction

Service: Amazon Transcribe Service

Settings for content redaction within a transcription job.

Contents

RedactionOutput

The output transcript file stored in either the default S3 bucket or in a bucket you specify.

When you choose `redacted` Amazon Transcribe outputs only the redacted transcript.

When you choose `redacted_and_unredacted` Amazon Transcribe outputs both the redacted and unredacted transcripts.

Type: String

Valid Values: `redacted` | `redacted_and_unredacted`

Required: Yes

RedactionType

Request parameter that defines the entities to be redacted. The only accepted value is `PII`.

Type: String

Valid Values: `PII`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

InputDataConfig

Service: Amazon Transcribe Service

The object that contains the Amazon S3 object location and access role required to train and tune your custom language model.

Contents

DataAccessRoleArn

The Amazon Resource Name (ARN) that uniquely identifies the permissions you've given Amazon Transcribe to access your Amazon S3 buckets containing your media files or text data.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: ^arn:(aws|aws-cn|aws-us-gov|aws-iso-{0,1}[a-z]{0,1}):iam::[0-9]{0,63}:role/[A-Za-z0-9:_/+=@.-]{0,1024}\$

Required: Yes

S3Uri

The Amazon S3 prefix you specify to access the plain text files that you use to train your custom language model.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (s3://|http(s*)://).+

Required: Yes

TuningDataS3Uri

The Amazon S3 prefix you specify to access the plain text files that you use to tune your custom language model.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (s3://|http(s*)://).+

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

InterruptionFilter

Service: Amazon Transcribe Service

An object that enables you to configure your category to be applied to call analytics jobs where either the customer or agent was interrupted.

Contents

AbsoluteTimeRange

An object you can use to specify a time range (in milliseconds) for when you'd want to find the interruption. For example, you could search for an interruption between the 30,000 millisecond mark and the 45,000 millisecond mark. You could also specify the time period as the first 15,000 milliseconds or the last 15,000 milliseconds.

Type: [AbsoluteTimeRange \(p. 417\)](#) object

Required: No

Negate

Set to `TRUE` to look for a time period where there was no interruption.

Type: Boolean

Required: No

ParticipantRole

Indicates whether the caller or customer was interrupting.

Type: String

Valid Values: `AGENT` | `CUSTOMER`

Required: No

RelativeTimeRange

An object that allows percentages to specify the proportion of the call where there was a interruption. For example, you can specify the first half of the call. You can also specify the period of time between halfway through to three-quarters of the way through the call. Because the length of conversation can vary between calls, you can apply relative time ranges across all calls.

Type: [RelativeTimeRange \(p. 448\)](#) object

Required: No

Threshold

The duration of the interruption.

Type: Long

Valid Range: Minimum value of 0. Maximum value of 14400000.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

JobExecutionSettings

Service: Amazon Transcribe Service

Provides information about when a transcription job should be executed.

Contents

AllowDeferredExecution

Indicates whether a job should be queued by Amazon Transcribe when the concurrent execution limit is exceeded. When the `AllowDeferredExecution` field is true, jobs are queued and executed when the number of executing jobs falls below the concurrent execution limit. If the field is false, Amazon Transcribe returns a `LimitExceeded` exception.

Note that job queuing is enabled by default for call analytics jobs.

If you specify the `AllowDeferredExecution` field, you must specify the `DataAccessRoleArn` field.

Type: Boolean

Required: No

DataAccessRoleArn

The Amazon Resource Name (ARN) of a role that has access to the S3 bucket that contains the input files. Amazon Transcribe assumes this role to read queued media files. If you have specified an output S3 bucket for the transcription results, this role should have access to the output bucket as well.

If you specify the `AllowDeferredExecution` field, you must specify the `DataAccessRoleArn` field.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `^arn:(aws|aws-cn|aws-us-gov|aws-iso-{0,1}[a-z]{0,1}):iam::[0-9]{0,63}:role/[A-Za-z0-9:_/+@.-]{0,1024}$`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

LanguageModel

Service: Amazon Transcribe Service

The structure used to describe a custom language model.

Contents

BaseModelName

The Amazon Transcribe standard language model, or base model used to create the custom language model.

Type: String

Valid Values: `NarrowBand` | `WideBand`

Required: No

CreateTime

The time the custom language model was created.

Type: Timestamp

Required: No

FailureReason

The reason why the custom language model couldn't be created.

Type: String

Required: No

InputDataConfig

The data access role and Amazon S3 prefixes for the input files used to train the custom language model.

Type: [InputDataConfig \(p. 430\)](#) object

Required: No

LanguageCode

The language code you used to create your custom language model.

Type: String

Valid Values: `en-US` | `hi-IN` | `es-US` | `en-GB` | `en-AU`

Required: No

LastModifiedTime

The most recent time the custom language model was modified.

Type: Timestamp

Required: No

ModelName

The name of the custom language model.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

ModelStatus

The creation status of a custom language model. When the status is COMPLETED the model is ready for use.

Type: String

Valid Values: IN_PROGRESS | FAILED | COMPLETED

Required: No

UpgradeAvailability

Whether the base model used for the custom language model is up to date. If this field is true then you are running the most up-to-date version of the base model in your custom language model.

Type: Boolean

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Media

Service: Amazon Transcribe Service

Describes the input media file in a transcription request.

Contents

MediaFileUri

The S3 object location of the input media file. The URI must be in the same region as the API endpoint that you are calling. The general form is:

`s3://<AWSDOC-EXAMPLE-BUCKET>/<keyprefix>/<objectkey>`

For example:

`s3://AWSDOC-EXAMPLE-BUCKET/example.mp4`

`s3://AWSDOC-EXAMPLE-BUCKET/mediadocs/example.mp4`

For more information about S3 object names, see [Object Keys](#) in the *Amazon S3 Developer Guide*.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: `(s3://|http(s*)://).+`

Required: No

RedactedMediaFileUri

The S3 object location for your redacted output media file. This is only supported for call analytics jobs.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: `(s3://|http(s*)://).+`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalTranscript

Service: Amazon Transcribe Service

Identifies the location of a medical transcript.

Contents

TranscriptFileUri

The S3 object location of the medical transcript.

Use this URI to access the medical transcript. This URI points to the S3 bucket you created to store the medical transcript.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (`s3://` | `http(s*)://`).+

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalTranscriptionJob

Service: Amazon Transcribe Service

The data structure that contains the information for a medical transcription job.

Contents

CompletionTime

A timestamp that shows when the job was completed.

Type: Timestamp

Required: No

ContentIdentificationType

Shows the type of content that you've configured Amazon Transcribe Medical to identify in a transcription job. If the value is `PHI`, you've configured the job to identify personal health information (PHI) in the transcription output.

Type: String

Valid Values: `PHI`

Required: No

CreationTime

A timestamp that shows when the job was created.

Type: Timestamp

Required: No

FailureReason

If the `TranscriptionJobStatus` field is `FAILED`, this field contains information about why the job failed.

The `FailureReason` field contains one of the following values:

- Unsupported media format- The media format specified in the `MediaFormat` field of the request isn't valid. See the description of the `MediaFormat` field for a list of valid values.
- The media format provided does not match the detected media format- The media format of the audio file doesn't match the format specified in the `MediaFormat` field in the request. Check the media format of your media file and make sure the two values match.
- Invalid sample rate for audio file- The sample rate specified in the `MediaSampleRateHertz` of the request isn't valid. The sample rate must be between 8,000 and 48,000 Hertz.
- The sample rate provided does not match the detected sample rate- The sample rate in the audio file doesn't match the sample rate specified in the `MediaSampleRateHertz` field in the request. Check the sample rate of your media file and make sure that the two values match.
- Invalid file size: file size too large- The size of your audio file is larger than what Amazon Transcribe Medical can process. For more information, see [Guidelines and Quotas](#) in the [Amazon Transcribe Medical Guide](#)
- Invalid number of channels: number of channels too large- Your audio contains more channels than Amazon Transcribe Medical is configured to process. To request additional channels, see [Amazon Transcribe Medical Endpoints and Quotas](#) in the [Amazon Web Services General Reference](#)

Type: String

Required: No

LanguageCode

The language code for the language spoken in the source audio file. US English (en-US) is the only supported language for medical transcriptions. Any other value you enter for language code results in a `BadRequestException` error.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: No

Media

Describes the input media file in a transcription request.

Type: [Media \(p. 436\)](#) object

Required: No

MediaFormat

The format of the input media file.

Type: String

Valid Values: mp3 | mp4 | wav | flac | ogg | amr | webm

Required: No

MediaSampleRateHertz

The sample rate, in Hertz, of the source audio containing medical information.

If you don't specify the sample rate, Amazon Transcribe Medical determines it for you. If you choose to specify the sample rate, it must match the rate detected by Amazon Transcribe Medical. In most cases, you should leave the `MedicalMediaSampleHertz` blank and let Amazon Transcribe Medical determine the sample rate.

Type: Integer

Valid Range: Minimum value of 8000. Maximum value of 48000.

Required: No

MedicalTranscriptionJobName

The name for a given medical transcription job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

Settings

Object that contains [MedicalTranscriptionSetting \(p. 444\)](#) object.

Type: [MedicalTranscriptionSetting \(p. 444\)](#) object

Required: No

Specialty

The medical specialty of any clinicians providing a dictation or having a conversation. Refer to [Transcribing a medical conversation](#)for a list of supported specialties.

Type: String

Valid Values: PRIMARYCARE

Required: No

StartTime

A timestamp that shows when the job started processing.

Type: Timestamp

Required: No

Tags

A key:value pair assigned to a given medical transcription job.

Type: Array of [Tag \(p. 455\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Required: No

Transcript

An object that contains the [MedicalTranscript](#). The [MedicalTranscript](#) contains the [TranscriptFileUri](#).

Type: [MedicalTranscript \(p. 437\)](#) object

Required: No

TranscriptionJobStatus

The completion status of a medical transcription job.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Required: No

Type

The type of speech in the transcription job. CONVERSATION is generally used for patient-physician dialogues. DICTATION is the setting for physicians speaking their notes after seeing a patient. For more information, see [What is Amazon Transcribe Medical?](#).

Type: String

Valid Values: CONVERSATION | DICTATION

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalTranscriptionJobSummary

Service: Amazon Transcribe Service

Provides summary information about a transcription job.

Contents

CompletionTime

A timestamp that shows when the job was completed.

Type: Timestamp

Required: No

ContentIdentificationType

Shows the type of information you've configured Amazon Transcribe Medical to identify in a transcription job. If the value is `PHI`, you've configured the transcription job to identify personal health information (PHI).

Type: String

Valid Values: `PHI`

Required: No

CreationTime

A timestamp that shows when the medical transcription job was created.

Type: Timestamp

Required: No

FailureReason

If the `TranscriptionJobStatus` field is `FAILED`, a description of the error.

Type: String

Required: No

LanguageCode

The language of the transcript in the source audio file.

Type: String

Valid Values: `af-ZA` | `ar-AE` | `ar-SA` | `cy-GB` | `da-DK` | `de-CH` | `de-DE` | `en-AB` | `en-AU` | `en-GB` | `en-IE` | `en-IN` | `en-US` | `en-WL` | `es-ES` | `es-US` | `fa-IR` | `fr-CA` | `fr-FR` | `ga-IE` | `gd-GB` | `he-IL` | `hi-IN` | `id-ID` | `it-IT` | `ja-JP` | `ko-KR` | `ms-MY` | `nl-NL` | `pt-BR` | `pt-PT` | `ru-RU` | `ta-IN` | `te-IN` | `tr-TR` | `zh-CN` | `zh-TW` | `th-TH` | `en-ZA` | `en-NZ`

Required: No

MedicalTranscriptionJobName

The name of a medical transcription job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

OutputLocationType

Indicates the location of the transcription job's output. This field must be the path of an S3 bucket; if you don't already have an S3 bucket, one is created based on the path you add.

Type: String

Valid Values: CUSTOMER_BUCKET | SERVICE_BUCKET

Required: No

Specialty

The medical specialty of the transcription job. Refer to [Transcribing a medical conversation](#)for a list of supported specialties.

Type: String

Valid Values: PRIMARYCARE

Required: No

StartTime

A timestamp that shows when the job began processing.

Type: Timestamp

Required: No

TranscriptionJobStatus

The status of the medical transcription job.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Required: No

Type

The speech of the clinician in the input audio.

Type: String

Valid Values: CONVERSATION | DICTATION

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalTranscriptionSetting

Service: Amazon Transcribe Service

Optional settings for the [StartMedicalTranscriptionJob \(p. 370\)](#) operation.

Contents

ChannelIdentification

Instructs Amazon Transcribe Medical to process each audio channel separately and then merge the transcription output of each channel into a single transcription.

Amazon Transcribe Medical also produces a transcription of each item detected on an audio channel, including the start time and end time of the item and alternative transcriptions of item. The alternative transcriptions also come with confidence scores provided by Amazon Transcribe Medical.

You can't set both `ShowSpeakerLabels` and `ChannelIdentification` in the same request. If you set both, your request returns a `BadRequestException`.

Type: Boolean

Required: No

MaxAlternatives

The maximum number of alternatives that you tell the service to return. If you specify the `MaxAlternatives` field, you must set the `ShowAlternatives` field to true.

Type: Integer

Valid Range: Minimum value of 2. Maximum value of 10.

Required: No

MaxSpeakerLabels

The maximum number of speakers to identify in the input audio. If there are more speakers in the audio than this number, multiple speakers are identified as a single speaker. If you specify the `MaxSpeakerLabels` field, you must set the `ShowSpeakerLabels` field to true.

Type: Integer

Valid Range: Minimum value of 2. Maximum value of 10.

Required: No

ShowAlternatives

Determines whether alternative transcripts are generated along with the transcript that has the highest confidence. If you set `ShowAlternatives` field to true, you must also set the maximum number of alternatives to return in the `MaxAlternatives` field.

Type: Boolean

Required: No

ShowSpeakerLabels

Determines whether the transcription job uses speaker recognition to identify different speakers in the input audio. Speaker recognition labels individual speakers in the audio file. If you set the `ShowSpeakerLabels` field to true, you must also set the maximum number of speaker labels in the `MaxSpeakerLabels` field.

You can't set both `ShowSpeakerLabels` and `ChannelIdentification` in the same request. If you set both, your request returns a `BadRequestException`.

Type: Boolean

Required: No

VocabularyName

The name of the vocabulary to use when processing a medical transcription job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

ModelSettings

Service: Amazon Transcribe Service

The object used to call your custom language model to your transcription job.

Contents

LanguageModelName

The name of your custom language model.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

NonTalkTimeFilter

Service: Amazon Transcribe Service

An object that enables you to configure your category to be applied to call analytics jobs where either the customer or agent was interrupted.

Contents

AbsoluteTimeRange

An object you can use to specify a time range (in milliseconds) for when no one is talking. For example, you could specify a time period between the 30,000 millisecond mark and the 45,000 millisecond mark. You could also specify the time period as the first 15,000 milliseconds or the last 15,000 milliseconds.

Type: [AbsoluteTimeRange \(p. 417\)](#) object

Required: No

Negate

Set to TRUE to look for a time period when people were talking.

Type: Boolean

Required: No

RelativeTimeRange

An object that allows percentages to specify the proportion of the call where there was silence. For example, you can specify the first half of the call. You can also specify the period of time between halfway through to three-quarters of the way through the call. Because the length of conversation can vary between calls, you can apply relative time ranges across all calls.

Type: [RelativeTimeRange \(p. 448\)](#) object

Required: No

Threshold

The duration of the period when neither the customer nor agent was talking.

Type: Long

Valid Range: Minimum value of 0. Maximum value of 14400000.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

RelativeTimeRange

Service: Amazon Transcribe Service

An object that allows percentages to specify the proportion of the call where you would like to apply a filter. For example, you can specify the first half of the call. You can also specify the period of time between halfway through to three-quarters of the way through the call. Because the length of conversation can vary between calls, you can apply relative time ranges across all calls.

Contents

EndPercentage

A value that indicates the percentage of the end of the time range. To set a relative time range, you must specify a start percentage and an end percentage. For example, if you specify the following values:

- StartPercentage - 10
- EndPercentage - 50

This looks at the time range starting from 10% of the way into the call to 50% of the way through the call. For a call that lasts 100,000 milliseconds, this example range would apply from the 10,000 millisecond mark to the 50,000 millisecond mark.

Type: Integer

Valid Range: Minimum value of 0. Maximum value of 100.

Required: No

First

A range that takes the portion of the call up to the time in milliseconds set by the value that you've specified. For example, if you specify 120000, the time range is set for the first 120,000 milliseconds of the call.

Type: Integer

Valid Range: Minimum value of 0. Maximum value of 100.

Required: No

Last

A range that takes the portion of the call from the time in milliseconds set by the value that you've specified to the end of the call. For example, if you specify 120000, the time range is set for the last 120,000 milliseconds of the call.

Type: Integer

Valid Range: Minimum value of 0. Maximum value of 100.

Required: No

StartPercentage

A value that indicates the percentage of the beginning of the time range. To set a relative time range, you must specify a start percentage and an end percentage. For example, if you specify the following values:

- StartPercentage - 10
- EndPercentage - 50

This looks at the time range starting from 10% of the way into the call to 50% of the way through the call. For a call that lasts 100,000 milliseconds, this example range would apply from the 10,000 millisecond mark to the 50,000 millisecond mark.

Type: Integer

Valid Range: Minimum value of 0. Maximum value of 100.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Rule

Service: Amazon Transcribe Service

A condition in the call between the customer and the agent that you want to filter for.

Contents

InterruptionFilter

A condition for a time period when either the customer or agent was interrupting the other person.

Type: [InterruptionFilter \(p. 431\)](#) object

Required: No

NonTalkTimeFilter

A condition for a time period when neither the customer nor the agent was talking.

Type: [NonTalkTimeFilter \(p. 447\)](#) object

Required: No

SentimentFilter

A condition that is applied to a particular customer sentiment.

Type: [SentimentFilter \(p. 451\)](#) object

Required: No

TranscriptFilter

A condition that catches particular words or phrases based on an exact match. For example, if you set the phrase "I want to speak to the manager", only that exact phrase will be returned.

Type: [TranscriptFilter \(p. 457\)](#) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

SentimentFilter

Service: Amazon Transcribe Service

An object that enables you to specify a particular customer or agent sentiment. If at least 50 percent of the conversation turns (the back-and-forth between two speakers) in a specified time period match the specified sentiment, Amazon Transcribe will consider the sentiment a match.

Contents

AbsoluteTimeRange

The time range, measured in seconds, of the sentiment.

Type: [AbsoluteTimeRange \(p. 417\)](#) object

Required: No

Negate

Set to TRUE to look for sentiments that weren't specified in the request.

Type: Boolean

Required: No

ParticipantRole

A value that determines whether the sentiment belongs to the customer or the agent.

Type: String

Valid Values: AGENT | CUSTOMER

Required: No

RelativeTimeRange

The time range, set in percentages, that correspond to proportion of the call.

Type: [RelativeTimeRange \(p. 448\)](#) object

Required: No

Sentiments

An array that enables you to specify sentiments for the customer or agent. You can specify one or more values.

Type: Array of strings

Array Members: Minimum number of 1 item.

Valid Values: POSITIVE | NEGATIVE | NEUTRAL | MIXED

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)

- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Settings

Service: Amazon Transcribe Service

Provides optional settings for the `StartTranscriptionJob` operation.

Contents

ChannelIdentification

Instructs Amazon Transcribe to process each audio channel separately and then merge the transcription output of each channel into a single transcription.

Amazon Transcribe also produces a transcription of each item detected on an audio channel, including the start time and end time of the item and alternative transcriptions of the item including the confidence that Amazon Transcribe has in the transcription.

You can't set both `ShowSpeakerLabels` and `ChannelIdentification` in the same request. If you set both, your request returns a `BadRequestException`.

Type: Boolean

Required: No

MaxAlternatives

The number of alternative transcriptions that the service should return. If you specify the `MaxAlternatives` field, you must set the `ShowAlternatives` field to true.

Type: Integer

Valid Range: Minimum value of 2. Maximum value of 10.

Required: No

MaxSpeakerLabels

The maximum number of speakers to identify in the input audio. If there are more speakers in the audio than this number, multiple speakers are identified as a single speaker. If you specify the `MaxSpeakerLabels` field, you must set the `ShowSpeakerLabels` field to true.

Type: Integer

Valid Range: Minimum value of 2. Maximum value of 10.

Required: No

ShowAlternatives

Determines whether the transcription contains alternative transcriptions. If you set the `ShowAlternatives` field to true, you must also set the maximum number of alternatives to return in the `MaxAlternatives` field.

Type: Boolean

Required: No

ShowSpeakerLabels

Determines whether the transcription job uses speaker recognition to identify different speakers in the input audio. Speaker recognition labels individual speakers in the audio file. If you set the `ShowSpeakerLabels` field to true, you must also set the maximum number of speaker labels `MaxSpeakerLabels` field.

You can't set both `ShowSpeakerLabels` and `ChannelIdentification` in the same request. If you set both, your request returns a `BadRequestException`.

Type: Boolean

Required: No

VocabularyFilterMethod

Set to `mask` to remove filtered text from the transcript and replace it with three asterisks ("***") as placeholder text. Set to `remove` to remove filtered text from the transcript without using placeholder text. Set to `tag` to mark the word in the transcription output that matches the vocabulary filter. When you set the filter method to `tag`, the words matching your vocabulary filter are not masked or removed.

Type: String

Valid Values: `remove` | `mask` | `tag`

Required: No

VocabularyFilterName

The name of the vocabulary filter to use when transcribing the audio. The filter that you specify must have the same language code as the transcription job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^[0-9a-zA-Z._-]+`

Required: No

VocabularyName

The name of a vocabulary to use when processing the transcription job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: `^[0-9a-zA-Z._-]+`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Tag

Service: Amazon Transcribe Service

A key:value pair that adds metadata to a resource used by Amazon Transcribe. For example, a tag with the key:value pair 'Department':'Sales' might be added to a resource to indicate its use by your organization's sales department.

Contents

Key

The first part of a key:value pair that forms a tag associated with a given resource. For example, in the tag 'Department':'Sales', the key is 'Department'.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Required: Yes

Value

The second part of a key:value pair that forms a tag associated with a given resource. For example, in the tag 'Department':'Sales', the value is 'Sales'.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 256.

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Transcript

Service: Amazon Transcribe Service

Identifies the location of a transcription.

Contents

RedactedTranscriptFileUri

The S3 object location of the redacted transcript.

Use this URI to access the redacted transcript. If you specified an S3 bucket in the `OutputBucketName` field when you created the job, this is the URI of that bucket. If you chose to store the transcript in Amazon Transcribe, this is a shareable URL that provides secure access to that location.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (`s3://` | `http(s*)://`).+

Required: No

TranscriptFileUri

The S3 object location of the transcript.

Use this URI to access the transcript. If you specified an S3 bucket in the `OutputBucketName` field when you created the job, this is the URI of that bucket. If you chose to store the transcript in Amazon Transcribe, this is a shareable URL that provides secure access to that location.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: (`s3://` | `http(s*)://`).+

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

TranscriptFilter

Service: Amazon Transcribe Service

Matches the output of the transcription to either the specific phrases that you specify, or the intent of the phrases that you specify.

Contents

AbsoluteTimeRange

A time range, set in seconds, between two points in the call.

Type: [AbsoluteTimeRange \(p. 417\)](#) object

Required: No

Negate

If TRUE, the rule that you specify is applied to everything except for the phrases that you specify.

Type: Boolean

Required: No

ParticipantRole

Determines whether the customer or the agent is speaking the phrases that you've specified.

Type: String

Valid Values: AGENT | CUSTOMER

Required: No

RelativeTimeRange

An object that allows percentages to specify the proportion of the call where you would like to apply a filter. For example, you can specify the first half of the call. You can also specify the period of time between halfway through to three-quarters of the way through the call. Because the length of conversation can vary between calls, you can apply relative time ranges across all calls.

Type: [RelativeTimeRange \(p. 448\)](#) object

Required: No

Targets

The phrases that you're specifying for the transcript filter to match.

Type: Array of strings

Array Members: Minimum number of 1 item.

Length Constraints: Minimum length of 1. Maximum length of 2000.

Pattern: .*\\$.*

Required: Yes

TranscriptFilterType

Matches the phrase to the transcription output in a word for word fashion. For example, if you specify the phrase "I want to speak to the manager." Amazon Transcribe attempts to match that specific phrase to the transcription.

Type: String

Valid Values: EXACT

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

TranscriptionJob

Service: Amazon Transcribe Service

Describes an asynchronous transcription job that was created with the `StartTranscriptionJob` operation.

Contents

CompletionTime

A timestamp that shows when the job completed.

Type: Timestamp

Required: No

ContentRedaction

An object that describes content redaction settings for the transcription job.

Type: [ContentRedaction \(p. 429\)](#) object

Required: No

CreationTime

A timestamp that shows when the job was created.

Type: Timestamp

Required: No

FailureReason

If the `TranscriptionJobStatus` field is `FAILED`, this field contains information about why the job failed.

The `FailureReason` field can contain one of the following values:

- `Unsupported media format` - The media format specified in the `MediaFormat` field of the request isn't valid. See the description of the `MediaFormat` field for a list of valid values.
- `The media format provided does not match the detected media format` - The media format of the audio file doesn't match the format specified in the `MediaFormat` field in the request. Check the media format of your media file and make sure that the two values match.
- `Invalid sample rate for audio file` - The sample rate specified in the `MediaSampleRateHertz` of the request isn't valid. The sample rate must be between 8,000 and 48,000 Hertz.
- `The sample rate provided does not match the detected sample rate` - The sample rate in the audio file doesn't match the sample rate specified in the `MediaSampleRateHertz` field in the request. Check the sample rate of your media file and make sure that the two values match.
- `Invalid file size: file size too large` - The size of your audio file is larger than Amazon Transcribe can process. For more information, see [Limits](#) in the *Amazon Transcribe Developer Guide*.
- `Invalid number of channels: number of channels too large` - Your audio contains more channels than Amazon Transcribe is configured to process. To request additional channels, see [Amazon Transcribe Limits](#) in the *Amazon Web Services General Reference*.

Type: String

Required: No

IdentifiedLanguageScore

A value between zero and one that Amazon Transcribe assigned to the language that it identified in the source audio. Larger values indicate that Amazon Transcribe has higher confidence in the language it identified.

Type: Float

Required: No

IdentifyLanguage

A value that shows if automatic language identification was enabled for a transcription job.

Type: Boolean

Required: No

JobExecutionSettings

Provides information about how a transcription job is executed.

Type: [JobExecutionSettings \(p. 433\)](#) object

Required: No

LanguageCode

The language code for the input speech.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: No

LanguageOptions

An object that shows the optional array of languages inputted for transcription jobs with automatic language identification enabled.

Type: Array of strings

Array Members: Minimum number of 1 item.

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: No

Media

An object that describes the input media for the transcription job.

Type: [Media \(p. 436\)](#) object

Required: No

MediaFormat

The format of the input media file.

Type: String

Valid Values: `mp3` | `mp4` | `wav` | `flac` | `ogg` | `amr` | `webm`

Required: No

MediaSampleRateHertz

The sample rate, in Hertz, of the audio track in the input media file.

Type: Integer

Valid Range: Minimum value of 8000. Maximum value of 48000.

Required: No

ModelSettings

An object containing the details of your custom language model.

Type: [ModelSettings \(p. 446\)](#) object

Required: No

Settings

Optional settings for the transcription job. Use these settings to turn on speaker recognition, to set the maximum number of speakers that should be identified and to specify a custom vocabulary to use when processing the transcription job.

Type: [Settings \(p. 453\)](#) object

Required: No

StartTime

A timestamp that shows when the job started processing.

Type: Timestamp

Required: No

Tags

A key:value pair assigned to a given transcription job.

Type: Array of [Tag \(p. 455\)](#) objects

Array Members: Minimum number of 1 item. Maximum number of 200 items.

Required: No

Transcript

An object that describes the output of the transcription job.

Type: [Transcript \(p. 456\)](#) object

Required: No

TranscriptionJobName

The name of the transcription job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

TranscriptionJobStatus

The status of the transcription job.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

TranscriptionJobSummary

Service: Amazon Transcribe Service

Provides a summary of information about a transcription job.

Contents

CompletionTime

A timestamp that shows when the job was completed.

Type: Timestamp

Required: No

ContentRedaction

The content redaction settings of the transcription job.

Type: [ContentRedaction \(p. 429\)](#) object

Required: No

CreationTime

A timestamp that shows when the job was created.

Type: Timestamp

Required: No

FailureReason

If the `TranscriptionJobStatus` field is `FAILED`, a description of the error.

Type: String

Required: No

IdentifiedLanguageScore

A value between zero and one that Amazon Transcribe assigned to the language it identified in the source audio. A higher score indicates that Amazon Transcribe is more confident in the language it identified.

Type: Float

Required: No

IdentifyLanguage

Whether automatic language identification was enabled for a transcription job.

Type: Boolean

Required: No

LanguageCode

The language code for the input speech.

Type: String

Valid Values: `af-ZA` | `ar-AE` | `ar-SA` | `cy-GB` | `da-DK` | `de-CH` | `de-DE` | `en-AB` | `en-AU` | `en-GB` | `en-IE` | `en-IN` | `en-US` | `en-WL` | `es-ES` | `es-US` | `fa-IR` | `fr-`

CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: No

ModelSettings

The object used to call your custom language model to your transcription job.

Type: [ModelSettings \(p. 446\)](#) object

Required: No

OutputLocationType

Indicates the location of the output of the transcription job.

If the value is CUSTOMER_BUCKET then the location is the S3 bucket specified in the outputBucketName field when the transcription job was started with the StartTranscriptionJob operation.

If the value is SERVICE_BUCKET then the output is stored by Amazon Transcribe and can be retrieved using the URI in the GetTranscriptionJob response's TranscriptFileUri field.

Type: String

Valid Values: CUSTOMER_BUCKET | SERVICE_BUCKET

Required: No

StartTime

A timestamp that shows when the job started processing.

Type: Timestamp

Required: No

TranscriptionJobName

The name of the transcription job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

TranscriptionJobStatus

The status of the transcription job. When the status is COMPLETED, use the GetTranscriptionJob operation to get the results of the transcription.

Type: String

Valid Values: QUEUED | IN_PROGRESS | FAILED | COMPLETED

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

VocabularyFilterInfo

Service: Amazon Transcribe Service

Provides information about a vocabulary filter.

Contents

LanguageCode

The language code of the words in the vocabulary filter.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: No

LastModifiedTime

The date and time that the vocabulary was last updated.

Type: Timestamp

Required: No

VocabularyFilterName

The name of the vocabulary filter. The name must be unique in the account that holds the filter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

VocabularyInfo

Service: Amazon Transcribe Service

Provides information about a custom vocabulary.

Contents

LanguageCode

The language code of the vocabulary entries.

Type: String

Valid Values: af-ZA | ar-AE | ar-SA | cy-GB | da-DK | de-CH | de-DE | en-AB | en-AU | en-GB | en-IE | en-IN | en-US | en-WL | es-ES | es-US | fa-IR | fr-CA | fr-FR | ga-IE | gd-GB | he-IL | hi-IN | id-ID | it-IT | ja-JP | ko-KR | ms-MY | nl-NL | pt-BR | pt-PT | ru-RU | ta-IN | te-IN | tr-TR | zh-CN | zh-TW | th-TH | en-ZA | en-NZ

Required: No

LastModifiedTime

The date and time that the vocabulary was last modified.

Type: Timestamp

Required: No

VocabularyName

The name of the vocabulary.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 200.

Pattern: ^[0-9a-zA-Z._-]+

Required: No

VocabularyState

The processing state of the vocabulary. If the state is READY you can use the vocabulary in a StartTranscriptionJob request.

Type: String

Valid Values: PENDING | READY | FAILED

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Amazon Transcribe Streaming Service

The following data types are supported by Amazon Transcribe Streaming Service:

- [Alternative \(p. 469\)](#)
- [AudioEvent \(p. 470\)](#)
- [AudioStream \(p. 471\)](#)
- [Entity \(p. 472\)](#)
- [Item \(p. 474\)](#)
- [MedicalAlternative \(p. 476\)](#)
- [MedicalEntity \(p. 477\)](#)
- [MedicalItem \(p. 478\)](#)
- [MedicalResult \(p. 480\)](#)
- [MedicalTranscript \(p. 482\)](#)
- [MedicalTranscriptEvent \(p. 483\)](#)
- [MedicalTranscriptResultStream \(p. 484\)](#)
- [Result \(p. 486\)](#)
- [Transcript \(p. 488\)](#)
- [TranscriptEvent \(p. 489\)](#)
- [TranscriptResultStream \(p. 490\)](#)

Alternative

Service: Amazon Transcribe Streaming Service

A list of possible transcriptions for the audio.

Contents

Entities

Contains the entities identified as personally identifiable information (PII) in the transcription output.

Type: Array of [Entity \(p. 472\)](#) objects

Required: No

Items

One or more alternative interpretations of the input audio.

Type: Array of [Item \(p. 474\)](#) objects

Required: No

Transcript

The text that was transcribed from the audio.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

AudioEvent

Service: Amazon Transcribe Streaming Service

Provides a wrapper for the audio chunks that you are sending.

For information on audio encoding in Amazon Transcribe, see [Speech input](#). For information on audio encoding formats in Amazon Transcribe Medical, see [Speech input](#).

Contents

AudioChunk

An audio blob that contains the next part of the audio that you want to transcribe. The maximum audio chunk size is 32 KB.

Type: Base64-encoded binary data object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

AudioStream

Service: Amazon Transcribe Streaming Service

Represents the audio stream from your application to Amazon Transcribe.

Contents

AudioEvent

A blob of audio from your application. Your audio stream consists of one or more audio events.

For information on audio encoding formats in Amazon Transcribe, see [Speech input](#). For information on audio encoding formats in Amazon Transcribe Medical, see [Speech input](#).

For more information on stream encoding in Amazon Transcribe, see [Event stream encoding](#). For information on stream encoding in Amazon Transcribe Medical, see [Event stream encoding](#).

Type: [AudioEvent \(p. 470\)](#) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Entity

Service: Amazon Transcribe Streaming Service

The entity identified as personally identifiable information (PII).

Contents

Category

The category of of information identified in this entity; for example, PII.

Type: String

Required: No

Confidence

A value between zero and one that Amazon Transcribe assigns to PII identified in the source audio.
Larger values indicate a higher confidence in PII identification.

Type: Double

Required: No

Content

The words in the transcription output that have been identified as a PII entity.

Type: String

Required: No

EndTime

The end time of speech that was identified as PII.

Type: Double

Required: No

StartTime

The start time of speech that was identified as PII.

Type: Double

Required: No

Type

The type of PII identified in this entity; for example, name or credit card number.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)

- [AWS SDK for Ruby V3](#)

Item

Service: Amazon Transcribe Streaming Service

A word, phrase, or punctuation mark that is transcribed from the input audio.

Contents

Confidence

A value between 0 and 1 for an item that is a confidence score that Amazon Transcribe assigns to each word or phrase that it transcribes.

Type: Double

Required: No

Content

The word or punctuation that was recognized in the input audio.

Type: String

Required: No

EndTime

The offset from the beginning of the audio stream to the end of the audio that resulted in the item.

Type: Double

Required: No

Speaker

If speaker identification is enabled, shows the speakers identified in the real-time stream.

Type: String

Required: No

Stable

If partial result stabilization has been enabled, indicates whether the word or phrase in the item is stable. If Stable is true, the result is stable.

Type: Boolean

Required: No

StartTime

The offset from the beginning of the audio stream to the beginning of the audio that resulted in the item.

Type: Double

Required: No

Type

The type of the item. PRONUNCIATION indicates that the item is a word that was recognized in the input audio. PUNCTUATION indicates that the item was interpreted as a pause in the input audio.

Type: String

Valid Values: pronunciation | punctuation

Required: No

VocabularyFilterMatch

Indicates whether a word in the item matches a word in the vocabulary filter you've chosen for your real-time stream. If true then a word in the item matches your vocabulary filter.

Type: Boolean

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalAlternative

Service: Amazon Transcribe Streaming Service

A list of possible transcriptions for the audio.

Contents

Entities

Contains the medical entities identified as personal health information in the transcription output.

Type: Array of [MedicalEntity \(p. 477\)](#) objects

Required: No

Items

A list of objects that contains words and punctuation marks that represents one or more interpretations of the input audio.

Type: Array of [MedicalItem \(p. 478\)](#) objects

Required: No

Transcript

The text that was transcribed from the audio.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalEntity

Service: Amazon Transcribe Streaming Service

The medical entity identified as personal health information.

Contents

Category

The type of personal health information of the medical entity.

Type: String

Required: No

Confidence

A value between zero and one that Amazon Transcribe Medical assigned to the personal health information that it identified in the source audio. Larger values indicate that Amazon Transcribe Medical has higher confidence in the personal health information that it identified.

Type: Double

Required: No

Content

The word or words in the transcription output that have been identified as a medical entity.

Type: String

Required: No

EndTime

The end time of the speech that was identified as a medical entity.

Type: Double

Required: No

StartTime

The start time of the speech that was identified as a medical entity.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalItem

Service: Amazon Transcribe Streaming Service

A word, phrase, or punctuation mark that is transcribed from the input audio.

Contents

Confidence

A value between 0 and 1 for an item that is a confidence score that Amazon Transcribe Medical assigns to each word that it transcribes.

Type: Double

Required: No

Content

The word or punctuation mark that was recognized in the input audio.

Type: String

Required: No

EndTime

The number of seconds into an audio stream that indicates the creation time of an item.

Type: Double

Required: No

Speaker

If speaker identification is enabled, shows the integer values that correspond to the different speakers identified in the stream. For example, if the value of Speaker in the stream is either a 0 or a 1, that indicates that Amazon Transcribe Medical has identified two speakers in the stream. The value of 0 corresponds to one speaker and the value of 1 corresponds to the other speaker.

Type: String

Required: No

StartTime

The number of seconds into an audio stream that indicates the creation time of an item.

Type: Double

Required: No

Type

The type of the item. PRONUNCIATION indicates that the item is a word that was recognized in the input audio. PUNCTUATION indicates that the item was interpreted as a pause in the input audio, such as a period to indicate the end of a sentence.

Type: String

Valid Values: pronunciation | punctuation

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalResult

Service: Amazon Transcribe Streaming Service

The results of transcribing a portion of the input audio stream.

Contents

Alternatives

A list of possible transcriptions of the audio. Each alternative typically contains one `Item` that contains the result of the transcription.

Type: Array of [MedicalAlternative \(p. 476\)](#) objects

Required: No

ChannelId

When channel identification is enabled, Amazon Transcribe Medical transcribes the speech from each audio channel separately.

You can use `ChannelId` to retrieve the transcription results for a single channel in your audio stream.

Type: String

Required: No

EndTime

The time, in seconds, from the beginning of the audio stream to the end of the result.

Type: Double

Required: No

IsPartial

Amazon Transcribe Medical divides the incoming audio stream into segments at natural points in the audio. Transcription results are returned based on these segments.

The `IsPartial` field is `true` to indicate that Amazon Transcribe Medical has additional transcription data to send. The `IsPartial` field is `false` to indicate that this is the last transcription result for the segment.

Type: Boolean

Required: No

ResultId

A unique identifier for the result.

Type: String

Required: No

StartTime

The time, in seconds, from the beginning of the audio stream to the beginning of the result.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalTranscript

Service: Amazon Transcribe Streaming Service

The medical transcript in a [MedicalTranscriptEvent \(p. 483\)](#).

Contents

Results

[MedicalResult \(p. 480\)](#) objects that contain the results of transcribing a portion of the input audio stream. The array can be empty.

Type: Array of [MedicalResult \(p. 480\)](#) objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalTranscriptEvent

Service: Amazon Transcribe Streaming Service

Represents a set of transcription results from the server to the client. It contains one or more segments of the transcription.

Contents

Transcript

The transcription of the audio stream. The transcription is composed of all of the items in the results list.

Type: [MedicalTranscript \(p. 482\)](#) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

MedicalTranscriptResultStream

Service: Amazon Transcribe Streaming Service

Represents the transcription result stream from Amazon Transcribe Medical to your application.

Contents

BadRequestException

One or more arguments to the `StartStreamTranscription` or `StartMedicalStreamTranscription` operation was invalid. For example, `MediaEncoding` was not set to a valid encoding, or `LanguageCode` was not set to a valid code. Check the parameters and try your request again.

Type: Exception

HTTP Status Code: 400

Required: No

ConflictException

A new stream started with the same session ID. The current stream has been terminated.

Type: Exception

HTTP Status Code: 409

Required: No

InternalFailureException

A problem occurred while processing the audio. Amazon Transcribe or Amazon Transcribe Medical terminated processing. Try your request again.

Type: Exception

HTTP Status Code: 500

Required: No

LimitExceededException

You have exceeded the maximum number of concurrent transcription streams, are starting transcription streams too quickly, or the maximum audio length of 4 hours. Wait until a stream has finished processing, or break your audio stream into smaller chunks and try your request again.

Type: Exception

HTTP Status Code: 429

Required: No

ServiceUnavailableException

Service is currently unavailable. Try your request later.

Type: Exception

HTTP Status Code: 503

Required: No

TranscriptEvent

A portion of the transcription of the audio stream. Events are sent periodically from Amazon Transcribe Medical to your application. The event can be a partial transcription of a section of the audio stream, or it can be the entire transcription of that portion of the audio stream.

Type: [MedicalTranscriptEvent \(p. 483\)](#) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Result

Service: Amazon Transcribe Streaming Service

The result of transcribing a portion of the input audio stream.

Contents

Alternatives

A list of possible transcriptions for the audio. Each alternative typically contains one item that contains the result of the transcription.

Type: Array of [Alternative \(p. 469\)](#) objects

Required: No

ChannelId

When channel identification is enabled, Amazon Transcribe transcribes the speech from each audio channel separately.

You can use ChannelId to retrieve the transcription results for a single channel in your audio stream.

Type: String

Required: No

EndTime

The offset in seconds from the beginning of the audio stream to the end of the result.

Type: Double

Required: No

IsPartial

Amazon Transcribe divides the incoming audio stream into segments at natural points in the audio. Transcription results are returned based on these segments.

The IsPartial field is true to indicate that Amazon Transcribe has additional transcription data to send, false to indicate that this is the last transcription result for the segment.

Type: Boolean

Required: No

ResultId

A unique identifier for the result.

Type: String

Required: No

StartTime

The offset in seconds from the beginning of the audio stream to the beginning of the result.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Transcript

Service: Amazon Transcribe Streaming Service

The transcription in a [TranscriptEvent \(p. 489\)](#).

Contents

Results

[Result \(p. 486\)](#) objects that contain the results of transcribing a portion of the input audio stream.
The array can be empty.

Type: Array of [Result \(p. 486\)](#) objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

TranscriptEvent

Service: Amazon Transcribe Streaming Service

Represents a set of transcription results from the server to the client. It contains one or more segments of the transcription.

Contents

Transcript

The transcription of the audio stream. The transcription is composed of all of the items in the results list.

Type: [Transcript \(p. 488\)](#) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

TranscriptResultStream

Service: Amazon Transcribe Streaming Service

Represents the transcription result stream from Amazon Transcribe to your application.

Contents

BadRequestException

A client error occurred when the stream was created. Check the parameters of the request and try your request again.

Type: Exception

HTTP Status Code: 400

Required: No

ConflictException

A new stream started with the same session ID. The current stream has been terminated.

Type: Exception

HTTP Status Code: 409

Required: No

InternalFailureException

A problem occurred while processing the audio. Amazon Transcribe terminated processing.

Type: Exception

HTTP Status Code: 500

Required: No

LimitExceededException

Your client has exceeded one of the Amazon Transcribe limits, typically the limit on audio length. Break your audio stream into smaller chunks and try your request again.

Type: Exception

HTTP Status Code: 429

Required: No

ServiceUnavailableException

Service is currently unavailable. Try your request later.

Type: Exception

HTTP Status Code: 503

Required: No

TranscriptEvent

A portion of the transcription of the audio stream. Events are sent periodically from Amazon Transcribe to your application. The event can be a partial transcription of a section of the audio stream, or it can be the entire transcription of that portion of the audio stream.

Type: [TranscriptEvent \(p. 489\)](#) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK for Ruby V3](#)

Common Errors

This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

IncompleteSignature

The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure

The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction

The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

InvalidClientTokenId

The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

InvalidParameterCombination

Parameters that must not be used together were used together.

HTTP Status Code: 400

InvalidParameterValue

An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

InvalidQueryParameter

The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

MalformedQueryString

The query string contains a syntax error.

HTTP Status Code: 404

MissingAction

The request is missing an action or a required parameter.

HTTP Status Code: 400

MissingAuthenticationToken

The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

MissingParameter

A required parameter for the specified action is not supplied.

HTTP Status Code: 400

NotAuthorized

You do not have permission to perform this action.

HTTP Status Code: 400

OptInRequired

The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired

The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable

The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400

Common Parameters

The following list contains the parameters that all actions use for signing Signature Version 4 requests with a query string. Any action-specific parameters are listed in the topic for that action. For more

information about Signature Version 4, see [Signature Version 4 Signing Process](#) in the *Amazon Web Services General Reference*.

Action

The action to be performed.

Type: string

Required: Yes

Version

The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

X-Amz-Algorithm

The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: AWS4-HMAC-SHA256

Required: Conditional

X-Amz-Credential

The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string ("aws4_request"). The value is expressed in the following format: *access_key/YYYYMMDD/region/service/aws4_request*.

For more information, see [Task 2: Create a String to Sign for Signature Version 4](#) in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-Date

The date that is used to create the signature. The format must be ISO 8601 basic format (YYYYMMDD'T'HHMMSS'Z'). For example, the following date time is a valid X-Amz-Date value: 20120325T120000Z.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see [Handling Dates in Signature Version 4](#) in the *Amazon Web Services General Reference*.

Type: string

Required: Conditional

X-Amz-Security-Token

The temporary security token that was obtained through a call to AWS Security Token Service (AWS STS). For a list of services that support temporary security credentials from AWS Security Token Service, go to [AWS Services That Work with IAM](#) in the *IAM User Guide*.

Condition: If you're using temporary security credentials from the AWS Security Token Service, you must include the security token.

Type: string

Required: Conditional

X-Amz-Signature

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-SignedHeaders

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see [Task 1: Create a Canonical Request For Signature Version 4](#) in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

AWS glossary

For the latest AWS terminology, see the [AWS glossary](#) in the *AWS General Reference*.