# TranscribeMe REST API

Jun 27, 2022

# Quickstarts

### Introduction

Once you , you will receive your own, unique API key and client secret. The key:

1. Uniquely identifies you.
2. Allows you to generate a token that will give you access to upload recordings, create transcription orders and retrieve output.
3. Should be kept private and should not be shared.
4. The API key should be passed in the API calls as a custom header called “X-Api-Key”.

### Authentication

The API key along with the client secret will be used to generate an initial token. API authentication is achieved via a bearer token which identifies a single user. Click *Authentication* to learn more about generating a token for authentication.

### Workflow

Click on our *Workflow Sample* for instructions on how to upload recordings, create orders, retrieve output and update settings. This will give a basic overview of how to use our API.

### Billing Information

Click *Billing Information* to view past orders and update your billing information.

CHAPTER 1

# Authentication

To use the TranscribeMe API, you must request an API key to use in all API requests. You can request an API key . The API key should be passed in the API calls as a custom header called "X-Api-Key", which is also called the "client_id".

API authentication is achieved via a bearer token which identifies a single user. Access token should be passed in the API calls as an authorization header parameter called "Bearer", which is typically used like 'Bearer {YOUR TOKEN}'.

Here are the values you will need to access your API:

1. **client_id** (X-Api-Key) - this is provided by TranscribeMe
2. **client_secret** - this is provided by TranscribeMe
3. **username** - Username (email) of the portal account
4. **password** or applicationtoken

---

When you first authenticate a particular account you should use your portal password and **grant_type=password** with the below method:

`POST https://rest-api.transcribeme.com/api/v1/token`

**HEADERS**:

```
Content-Type: application/x-www-form-urlencoded
X-Api-Key: {X-Api-Key}
```

**REQUEST**:

```
grant_type: password
client_id: {X-Api-Key}
client_secret: {CLIENT APP SECRET}
username: {USER NAME}
password: {PASSWORD}
```

**RESPONSE** *(Content-type:* **application/json**)*:

```
{
  "access_token":"{YOUR TOKEN}",
  "token_type":"bearer",
  "expires_in":35999,
  "refresh_token":"{YOUR REFRESH TOKEN}",
  "userName":"{USER EMAIL}",
  "userId":"{USER ID}",
  "firstName":"{USER FIRST NAME}",
  "lastName":"{USER LAST NAME}",
  "roles":"{USER ROLES LIST COMMA SEPARATED}",
  ".issued":"{ISSUE DATE}",
  ".expires":"{EXPIRES DATE}"
}
```

---

Then you will be able to generate {access_token} to be used in future calls like the below "regenerate" example.

You may then use **grant_type=applicationtoken** for authentication in the future. There are couple reasons why applicationtoken usage is more preferable than password for API Integration:

1. Password can be changed on UI and as a result all api calls authorization will fail
2. Passwords have policies and it is required to change password periodically

Application_token can be regenerated using regenerate method:

`POST https://rest-api.transcribeme.com/api/v1/applications/tokens/regenerate`

**HEADERS**:

```
Content-Type: application/x-www-form-urlencoded
Authorization: Bearer {YOUR TOKEN}
X-Api-Key: {X-Api-Key}
```

**REQUEST**:

```
client_id={X-Api-Key}
```

---

The json response will provide an application_token value, which can be used in the **grant_type=applicationtoken** token method below:

`POST https://rest-api.transcribeme.com/api/v1/token`

**HEADERS**:

```
Content-Type: application/x-www-form-urlencoded
X-Api-Key: {X-Api-Key}
```

**REQUEST**:

```
grant_type=applicationtoken
authtoken={application_token}
client_id={X-Api-Key}
client_secret={client_secret}
```

The access_token lifetime is 1 hour. You can also use **grant_type=refresh_token** for getting a new access token when the old one is expired. You just need to make the following POST request:

`POST https://rest-api.transcribeme.com/api/v1/token`

**HEADERS**:

```
Content-Type: application/x-www-form-urlencoded
X-Api-Key: {X-Api-Key}
```

**REQUEST**:

```
grant_type=refresh_token
refresh_token={refresh_token}
client_id={X-Api-Key}
client_secret={client_secret}
```

Our API also supports oAuth2. If you're going to obtain a bearer token using an external token the POST request is as follows:

`POST https://rest-api.transcribeme.com/api/v1/token`

**HEADERS**:

```
Content-Type: application/x-www-form-urlencoded
X-Api-Key: {X-Api-Key}
```

**REQUEST**:

```
grant_type=externaltoken
authtoken=[EXTERNAL TOKEN]
provider=[PROVIDER NAME]
role=[USER ROLE]
client_id={X-Api-Key}
client_secret={client_secret}
```

For now, the Facebook and Google are the only supported providers.

*Important: The external auth token should allow access to user profile information, including email.*

**Error Details**

The API uses two different formats to describe an error.

1. **Authentication error object** When the application makes requests to the API related to authentication or authorization (e.g. retrieving an access token or refreshing an access token) the error response follows RFC 6749 on The OAuth 2.0 Authorization Framework. Below is an example of a failing request to refresh an access token.

```
{
  "error": "invalid_client",
  "error_description": "Invalid client secret"
}
```

2. **Regular error object** Apart from the response code, unsuccessful responses return information about the error as an error JSON object containing the StatusCode and the array of error messages. Here is an example error response:

```
{
  StatusCode: 400,
  Messages: ["Some error message goes here", "Another error message goes here"]
}
```

# CHAPTER 2

# Workflow Sample

The most common use case contains 5 steps:

## 2.1 1. File Upload

You can choose your preferred way to deliver recordings:

**I.** The **preferred method** for uploading will require splitting the file into smaller "chunks". For optimal upload speed, each chunk should be exactly 5MB. Only the last chunk can be less than 5MB. If a total file size is less than 5 MBs, then it can still be uploaded with async approach, since the last chunk (a single chunk in this case) can be less than 5MB. Overall, this process requires 3 different endpoints:

**STEPS**

1. `GET https://rest-api.transcribeme.com/api/v1/uploads/url?fileName={filename.mp3}&isAsync=true`

- No body is required as parameters are passed in the url.
- Filename is whatever you choose to call the file, but the extension (i.e. mp3 or mp4) should match the file correctly.
- **This will return the URL endpoint (which includes uploadId and recordingId), to be used in steps 2 and 3. We will call it {URL1}.**

2. `POST {URL1}?uploadId={UPLOADID}&recordingId={RECORDINGID}&chunkNumber=0`

- Prior to this step, you will split an audio file into parts of equal duration. Each audio chunk should be exactly 5MB, except for the last one.
- You will use the URL provided in the response from step 1 above as the POST endpoint (**{URL1}**).
- **chunkNumber** in url above should be sequential to that chunk number (i.e. 0,1,2,etc).
- Body **REQUEST** includes file of the file "chunk" to upload. This field named should be called **file**.
- Repeat this step for each chunk, then move to step 3.

3. `POST https://rest-api.transcribeme.com/api/v1/uploads/async/commit?uploadId={uploadId}&recordingId={recordingId}`

- Once all chunks have been uploaded, this will commit them all and combine them into one recording.
- There is no body required here, just pass values in the url again.

---

- The simple method belows are being deprecated but can be used still:

**II.** `POST https://rest-api.transcribeme.com/api/v1/recordings/upload`

**REQUEST**:

```
Cache-Control: no-cache
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary1234567abcdefg
------WebKitFormBoundary1234567abcdefg
Content-Disposition: form-data; name="name"; filename="FILEPATH/MYFILE.mp3"
Content-Type: audio/mp3
------WebKitFormBoundary1234567abcdefg--
```

---

- By specifying publicly available url:

**III.** `POST https://rest-api.transcribeme.com/api/v1/recordings/remote`

**REQUEST**:

```
{
  "url": "https://www.MYWEBSITE.com/MYPATH"
}
```

*If you choose upload via publicly available url, you will need to add additional logic on your side to check the status of recording.*

*It is not possible to order a recording which is not uploaded to our system.*

**This will return a recordingId.**

## 2.2 2. Create Order

After an audio file has been successfully uploaded you are able to order a transcript. On this step you will send a list of recording id's that will be in the order.

*(Request object as Content-Type application/json)* `POST https://rest-api.transcribeme.com/api/v1/orders`

**REQUEST**:

```
{
  "id":"",
  "recordings":["{recordingID}"]
}
```

**This will return an Order json object.**

*You may also obtain the Order object using the following method:* `GET https://rest-api.transcribeme.com/api/v1/orders/{orderId}`

## 2.3 3. Update settings

Update settings within the recording object. It is most common to update type or output here. Use the endpoints below to obtain these expected values:

Type: `GET https://rest-api.transcribeme.com/api/v1/transcription/types`

Speakers: `GET https://rest-api.transcribeme.com/api/v1/transcription/speakers`

Output: `GET https://rest-api.transcribeme.com/api/v1/transcription/outputgroups`

Turnaround: `GET https://rest-api.transcribeme.com/api/v1/transcription/turnaround`

Language: `GET https://rest-api.transcribeme.com/api/v1/dictionaries/languages`

Accent: `GET https://rest-api.transcribeme.com/api/v1/dictionaries/languages/accents?languageId={languageId}`

Domain: `GET https://rest-api.transcribeme.com/api/v1/transcription/domains`

*(Request object as Content-Type application/json)* `POST https://rest-api.transcribeme.com/api/v1/orders/{orderId}/recordings/edit`

**REQUEST**:

```
[
     {
          "id": "{recordingID}",
          "settings": {
               "language": "{languageId}",
               "accent": "{accentID}",
               "type": {type},
               "domain": {domain},
               "output": {output},
               "turnaround": {turnaround},
               "speakers": {speakers},
               "isNoisyAudio": false,
               "isHeavyAccent": false
          }
     }
]
```

Also if you have a promo code to use, you may apply it here:

*(Request object as Content-Type application/json)* `POST https://rest-api.transcribeme.com/api/v1/orders/{orderID}/promocode`

**REQUEST**:

```
{
     "code": "YOUR_PROMO_CODE"
}
```

## 2.4 4. Place Order

***IMPORTANT!!!*** If you have been given a promo code to use, you MUST enter it before placing an order. Please see the above step for info about this.

Visit *Billing Information* to confirm that your billing information is setup correctly. You can also use a promo code created by the TranscribeMe Sales Team to bypass the credit card payment step and instead be billed by invoice.

*(Request object as Content-Type application/json)* `POST https://rest-api.transcribeme.com/api/v1/orders/{orderID}/place`

**Note the code for billingType below, as it should be passed as an array. REQUEST**:

```
[
     {
          "billingType": 0
     }
]
```

To query the status of the order, use the following method: `GET https://rest-api.transcribeme.com/api/v1/recordings/{recordingId}/status`

For list of available status values use: `GET https://rest-api.transcribeme.com/api/v1/dictionaries/recordingstatuses`

## 2.5 5. Get Results

You will receive transcription results within the agreed TAT. These are available in different formats.

To obtain the results as a json object use: `GET https://rest-api.transcribeme.com/api/v1/recordings/{recordingId}/transcription`

To download the file: `POST https://rest-api.transcribeme.com/api/v1/recordings/download`

**REQUEST**:

```
{
     "recordings": [
          {
               "id": "{recordingId}",
               "ownerId": "{userId}"``
          }
     ],
     "output": {output},
     "highlightedOnly": false,
     "removeStrikeout": false
}
```

# CHAPTER 3

# Billing Information

Here you may check your billing information.

**Address:** To check your billing address:

`GET https://rest-api.transcribeme.com/api/v1/billing/address`

To update your billing address:

`POST https://rest-api.transcribeme.com/api/v1/billing/address`

**REQUEST**:

```
{
    "firstName": "{firstName}",
    "lastName": "{lastName}",
    "email": "{email}",
    "address1": "{address1}",
    "address2": null,
    "country": {
        "id": "{country}",
        "name": "{countryName}"
    },
    "state": {
        "id": "{stateId}",
        "name": "{stateName}"
    },
    "city": "{city}",
    "zip": "{zip}"
}
```

**Payment information:** To check your credit card: `GET https://rest-api.transcribeme.com/api/v1/billing/card`

To update your credit card information or to add different credit cards for integration from your customers, BrainTree API/SDK's must be used to securely collect payment information. Please review the documentation here.

Using the BrainTree setup above, use the following to get a client token: `GET https://rest-api.transcribeme.com/api/v1/billing/gateway/client-token`

To send the payment method nonce to your server: `POST https://rest-api.transcribeme.com/api/v1/billing/card`

**REQUEST**:

```
{
  "token": "{gatewayToken}"
}
```

CHAPTER 4

# TranscribeMe REST API

- genindex
- modindex
- search