

# AppTone Get Questionnaires List

questionnaires is an HTTP GET endpoint to retrieve a list of all available questionnaires by filter.

A call to advisors endpoint requires basic authentication. Please refer to [Analyze Now Authentication](#)

Here is a sample Javascript code to fetch the questionnaires list

## Install required libraries

```
npm install axios
```

## And the actual code

```
const axios = require('axios');

const baseUrl = 'https://api.emlo.cloud/apigateway-service/apptone/v1';
const apiKey = '[YOUR API KEY]';
const apiPassword = '[YOUR API PASSWORD]';

async function fetchQuestionnaires() {
  try {
    const response = await axios.get(`${baseUrl}/questionnaires`, {
      headers: {
        'Authorization': 'Basic ' + Buffer.from(`${apiKey}:${apiPassword}`).toString('base64'),
      },
      params: {
        query: 'your query',
        tags: ['tag 1', 'tag 2'],
        languages: ['en', 'es'],
      },
    });

    console.log('Questionnaires:', response.data);
  } catch (error) {
    console.error('Error fetching questionnaires:', error);
  }
}

fetchQuestionnaires();
```

## Available filters for questionnaires endpoint

**query** - filter by the questionnaire name

**tags** - filter by a list of search tags assigned to the questionnaire

**languages** - filter by supported languages

## Response

The response is a list of questionnaires that matching the search criteria

```
[
  {
    "name": "Candidate Insight Questionnaire",
    "language": "en",
    "description": "A questionnaire for assessing the decision-making abilities, teamwork dynamics, work ethics, adaptability to change, and integrity and loyalty of potential candidates.",
    "apptoneQuestionnaireId": "7c834d22-78e9-4a83-9a7b-9bd3cd1cf21d"
  },
  {
    "name": "Candidate Insight Questionnaire - 16FP",
    "language": "en",
    "description": "A questionnaire for assessing the decision-making abilities, teamwork dynamics, work ethics, adaptability to change, and integrity and loyalty of potential candidates.",
    "apptoneQuestionnaireId": "95bb56e3-368a-41d8-a137-3492055d0bb2"
  }
]
```

## Fields

**name** - The questionnaire name

**language** - The questionnaire language

**description** - The questionnaire description

**apptoneQuestionnaireId** - The questionnaire id

## AppTone Send Questionnaire To Customer

sendToCustomer is an HTTP POST endpoint to start an asynchronous test interaction with a user.

The sendToCustomer process status reported though a webhook calls from AppTone service.

A call to sendToCustomer endpoint requires basic authentication. Please refer to [Analyze Now Authentication](#)

It is recommended to encrypt the callback payload data by passing an "encryptionKey" string value on the request. [Please read more](#)

## Sample NodeJS for sendToCustomer

### Install required libraries

```
npm install axios
```

## And the actual code

```
// Install the required libraries by running the following command:
// npm install axios

const axios = require('axios');

const baseUrl = 'https://api.emlo.cloud/apigateway-service/apptone/v1';
const apiUser = '[YOUR API KEY]';
const apiPassword = '[YOUR API PASSWORD]';
const questionnaireId = '[The questionnaire Id taht you want to execute]';
const webhookUrl = '[YOUR CALLBACK URL]';

async function sendToCustomer() {
  try {
    const response = await axios.post(
      `${baseUrl}/questionnaires/${questionnaireId}/sendToCustomer`,
      {
        phoneNumber: '[Phone Number]',
        name: '[Customer name]',
        statusCallbackUrl: webhookUrl
      },
      {
        headers: {
          'Authorization': 'Basic ' + Buffer.from(`${apiUser}:${apiPassword}`).toString('base64'),
          'Content-Type': 'application/json',
        }
      }
    );

    console.log('Response:', response.data);
  } catch (error) {
    console.error('Error sending to customer:', error);
  }
}

sendToCustomer();
```

## Response Structure

Upon request reception, AppTone validate the request parameters. For a valid request AppTone will return a "reportId" identifier to be used when recieving asynchronous status updates.

For invalid parameter the response AppTone will return an error code and message which indicates the invalid param.

### Sample response for a valid request

```
{
  "reportId": "573b1949-6f6a-4176-bfa5-25c50b116922"
}
```

### Sample response for a request with an invalid parameter

```
{
  "errorCode": "invalid_parameter",
  "error": "body.statusCallbackUrl must be a valid URL"
}
```

Once a valid request accepted on AppTone, it starts sending status update to the URL provided on "statusCallbackUrl" parameter.

### Sample status callback data

```
{
  "requestId": "f2de52a9-c1f2-4abb-b987-9453088b0157",
  "application": "apptone",
  "eventDate": "2024-06-19T13:40:46.079Z",
  "payload": {
    "reportId": "cdb1db8b-a7e8-4e0f-b2e6-7b9bdf38cb5d",
    "status": "pending"
  },
  "encrypted": false
}
```

### Params on status callback

application: always "apptone".

eventDate: Time of the event in GMT timezone

payload: contain the actual event data

payload/reportId: The reportId that was provided on the response for the sentToCustomer request

payload/status: The current analysis status

encrypted: true of "encryptionKey" parameter sent on the sentToCustomer request

### Available Statuses

**pending** - The test was sent to the customer

**running** - The customer is running the test. This status comes with "totalMessages" and "receivedMessages" params which indicates the running progress

**analyzing** - AppTone analyze the test

**completed** - The report is ready. the "analysis" data contains the analysis data

**In case an error happen during the test run, a relevant error status will be sent**

```
{
  "requestId": "76bded6e-3110-4f47-8230-040db611868c",
  "application": "apptone",
  "eventDate": "2024-06-19T13:56:09.973Z",
  ▼▼ "payload": {
    "reportId": "a4237ed4-5e46-47ad-b306-414c7d6d6db9",
    "error": "There is already a running test for this phone number.",
    "errorCode": "invalid_parameter"
  },
  "encrypted": false
}
```

## AppTone Cancel Test Run

**cancel** endpoint abort a test before its running completed

## Install the required libraries

```
npm install axios
```

## Actual code

```

// Install the required library by running the following command:
// npm install axios

const axios = require('axios');

// Define the necessary constants
const baseUrl = 'https://api.emlo.cloud/apigateway-service/apptone/v1';
const apiKey = '5492c434-8b3f-4e4b-8848-b66b94892460';
const apiPassword = 'HerecomestHesun449!';
const reportId = '7e1060a1-3279-4fa4-ac6b-16d7ec899ef9';

async function cancelReport() {
  try {
    const response = await axios.post(
      `${baseUrl}/reports/${reportId}/cancel`,
      {},
      {
        headers: {
          'Authorization': 'Basic ' + Buffer.from(`${apiKey}:${apiPassword}`).toString('base64'),
          'Content-Type': 'application/json',
        },
      },
    );

    // Log the response to the console
    console.log('Response:', response.data);
  } catch (error) {
    // Log any errors that occur
    console.error('Error cancelling report:', error);
    if (error.response) {
      console.error('Response data:', error.response.data);
    }
  }
}

// Call the function to execute the request
cancelReport();

```

**In case the reportId does not exists, or was already canceled, AppTone will respond with an HTTP 404 status**

# AppTone Download Report PDF

downloadPdf is an HTTP POST asynchronous endpoint to create and download the report in a PDF format.

The downloadPdf send process status report through a webhook calls from AppTone service.

A call to downloadPdf endpoint requires basic authentication. Please refer to [Analyze Now Authentication](#)

It is recommended to encrypt the callback payload data by passing an "encryptionKey" string value on the request. [Read more](#)

## Sample NodeJS code for downloadPdf

### Install required libraries

```
npm install axios fs
```

### And the actual code

```
// Install the required libraries by running the following commands:
// npm install axios

const axios = require('axios');
const fs = require('fs');

// Define the necessary constants
const baseUrl = 'https://api.emlo.cloud/apigateway-service/apptone/v1';
const apiKey = '[YOUR API KEY]';
const apiPassword = '[YOUR API PASSWORD]';
const reportId = '[REPORT ID]';
const webhookUrl = '[A URL ON YOUR SERVER TO RECEIVE THE STATUS CALLBACKS]';

async function downloadPdf() {
  try {
    const response = await axios.post(
      `${baseUrl}/reports/${reportId}/downloadPdf`,
      {
        statusCallbackUrl: webhookUrl,
      },
      {
        headers: {
          'Authorization': 'Basic ' + Buffer.from(`${apiKey}:${apiPassword}`).toString('base64'),
          'Content-Type': 'application/json',
        },
        responseType: 'stream',
      }
    );

    // Pipe the response data to a file
    response.data.pipe(fs.createWriteStream('report.pdf'));

    // Log the response to the console
    console.log('PDF downloaded successfully');
  } catch (error) {
    // Log any errors that occur
    console.error('Error downloading PDF:', error);
    if (error.response) {
      console.error('Response data:', error.response.data);
    }
  }
}

// Call the function to execute the request
downloadPdf();
```

### Response Structure

Upon request reception, AppTone validate the request parameters. For a valid request AppTone will return a "reportId" identifier to be used when receiving asynchronous status updates.

For invalid parameter the response AppTone will return an error code and message which indicates the invalid param.

## Sample response for a valid request

```
{
  "reportId": "573b1949-6f6a-4176-bfa5-25c50b116922"
}
```

## Sample response for a request with an invalid parameter

```
{
  "errorCode": "invalid_parameter",
  "error": "body.statusCallbackUrl must be a valid URL"
}
```

Once a valid request accepted on AppTone, it will send a status updates to the URL provided on "statusCallbackUrl" parameter.

## Sample status callback data with report PDF

```
{
  "requestId": "a9c3aa6c-d511-4370-8669-80ebf4ba8c06",
  "application": "apptone",
  "eventDate": "2024-06-20T08:47:59.973Z",
  "payload": {
    "reportId": "573b1949-6f6a-4176-bfa5-25c50b116922",
    "contentType": "application/pdf",
    "data": "[PDF DATA IN BASE64 FORMAT]"
  },
  "encrypted": false
}
```

## Params on status callback

**application:** always "apptone".

**eventDate:** Time of the event in GMT timezone

**payload:** contain the actual event data

**payload/reportId:** The reportId that was provided on the response for the sentToCustomer request

**payload/contentTyp:** always "application/pdf"

**payload/data:** The PDF file content in Base64 encoding

**encrypted:** true of "encryptionKey" parameter sent on the downloadPdf request

## Errors callback

In case an error happen during the test run, a relevant error status will be sent

```
{
  "requestId": "0d1267fc-9d63-419b-b48d-37d4464fe29f",
  "application": "apptone",
  "eventDate": "2024-06-20T08:42:13.582Z",
  "payload": {
    "reportId": "c98ba5a4-5464-477a-8e61-2667000facf6",
    "error": "report could not be found",
    "errorCode": "item_not_found"
  },
  "encrypted": false
}
```

---

Revision #1

Created 12 March 2025 21:25:23 by mauricio olarte

Updated 12 March 2025 21:25:38 by mauricio olarte