API Document

AIMS - Apprentice Internship Management System



# Version History

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Description** |
| 09/03/16 | Tiffany Trusty | Initial Draft |
| 11/22/16 | Tiffany Trusty | Updates after initial public testing |
| 12/13/16 | Tiffany Trusty | Updated token to contain user\_id  Tested logout  GET surveys/user\_id implemented and tested  GET survey/survey\_instance\_id partially implemented and tested  All red items reflect untested or incomplete items and are subject to change. |
| 12/15/16 | Tiffany Trusty | Completed GET survey/survey\_instance\_id  All red items reflect untested or incomplete items and are subject to change. |
| 12/17/16 | Tiffany Trusty | Completed POST survey call |
| 12/20/16 | Tiffany Trusty | Added new Register v2 call |

# Acronyms

|  |  |
| --- | --- |
| **Acronym** | **Definition** |
| SOW | Statement of Work |
| TDD | Technical Design Document |
| AIMS | Apprentice Internship Management System |
| API | Application Program Interface |
|  |  |
|  |  |

# References

1. AIMS SOW, EFC Salesforce / Tinderbox
2. AIMS Technical Design Document v2, Confluence
3. AIMS TDD Addendum v2, Confluence

# Table of Contents

Contents

[Version History 2](#_Toc469408791)

[Acronyms 2](#_Toc469408792)

[References 3](#_Toc469408793)

[Table of Contents 3](#_Toc469408794)

[Introduction 5](#_Toc469408795)

[Authentication 5](#_Toc469408796)

[HTTP Methods 6](#_Toc469408797)

[Response 6](#_Toc469408798)

[Date 6](#_Toc469408799)

[Errors 7](#_Toc469408800)

[Pagination 7](#_Toc469408801)

[URL 7](#_Toc469408802)

[Endpoints 8](#_Toc469408803)

[POST /register 8](#_Toc469408804)

[POST /login 9](#_Toc469408805)

[POST /logout 10](#_Toc469408806)

[GET /organizations 11](#_Toc469408807)

[GET /organization 12](#_Toc469408808)

[GET /groups/{organization\_id} 13](#_Toc469408809)

[GET /surveys/{userId} 14](#_Toc469408810)

[GET /survey/{survey\_id} Untested 15](#_Toc469408811)

[POST /survey Untested 17](#_Toc469408812)

[Appendix A – Developer Notes 18](#_Toc469408813)

# Introduction

This document defines the API for the AIMS project. AIMS API is built using REST principles which ensures predictable URLs that make writing apps easy. This API follows HTTP rules, enabling a wide range of HTTP clients can be used to interact with the API. Every resource is exposed as a URL. The URL of each resource can be obtained by accessing the API Root Endpoint.

# Authentication

Most AIMS APIs need to be authenticated using an authtoken. You can obtain an authtoken programmatically using the AIMS API (login).

# HTTP Methods

| **Method** | **Description** |
| --- | --- |
| GET | Used for retrieving resources. |
| POST | Used for creating resources and performing resource actions. |
| PUT | Used for updating resources. |
| DELETE | Used for deleting resources. |

# Response

Responses will be in the JSON format.

| **Node Name** | **Description** |
| --- | --- |
| code | AIMS error code. This will be zero for a success response and non-zero in case of an error. |
| message | Message for the invoked API. |
| data | Comprises the invoked API’s Data. |

# Date

All timestamps are returned in the ISO 8601 format - YYYY-MM-DDThh:mm:ssTZD.

Example: 2016-06-11T17:38:06-0700

# Errors

AIMS uses HTTP status codes to indicate success or failure of an API call. In general, status codes in the 2xx range means success, 4xx range means there was an error in the provided information, and those in the 5xx range indicates server side errors. Commonly used HTTP status codes are listed below.

| **Status Code** | **Description** |
| --- | --- |
| 200 | OK |
| 201 | Created |
| 400 | Bad request |
| 401 | Unauthorized (Invalid AuthToken) |
| 404 | URL Not Found |
| 405 | Method Not Allowed (Method you have called is not supported for the invoked API) |
| 429 | Rate Limit Exceeded (API usage limit exceeded) |
| 500 | Internal Error |

# Pagination

Post MVP Feature.

# URL

The base URL for all AIMS requests is in the following format:

http://aimswebapi.azurewebsites.net/api/<version>/<endpoint>

Current version = v2

Endpoints are defined below

# Endpoints

## POST /register/v1

**POST Method**

**Request JSON:**

{

"Email": "string",

"Password": "string",

"ConfirmPassword": "string"

}

**Response JSON:**

{

“code”: 200

}

## POST /register/v2

**POST Method**

**Request JSON:**

{

"FirstName": "string",

"LastName": "string",

"OrganizationId": 0,

"Email": "string",

"Password": "string",

"ConfirmPassword": "string"

}

**Response JSON:**

{

“code”: 200

}

## POST /login

**POST Method**

**Request JSON:**

**{**

**"Username": "string",**

**"Password": "string"**

**}**

Notes: string, username is email address

string, password, minlen = 6, requires a digit, requires an uppercase & lowercase, requires a special char ~`!@#$%^&\*()-\_+={}[]|\;:<>,./?"

**Response JSON:**

{

“code”: 200,

“data”:

{

"Token": "Bearer 065307ef-8036-46ea-bce4-bc6e8d2b9cb0-1",

"Expiration": "2016-12-02T17:28:35.5839502+00:00"

}

}

## POST /logout

**POST Method**

**Request JSON:**

Header: Authorization

Example:

/logout -H “Authorization: token ba4604e8e433g9c892e360d53463oec5-145“

**Response JSON:**

{

“code”: 200,

}

## GET /organizations

**GET Method:** Get the list of all AIMS organizations

**Request JSON:**

{}

http://aimswebapi.azurewebsites.net/api/v2/organizations

**Response JSON:**

Parameter: int, organization\_id

Parameter: string, name, maxlen = 256

Parameter: string, description, maxlen = 512

Example:

{

“code”: 200,

“message”: “Successfully retrieved orgs”,

“data”:

[

{

"organization\_id": 1,

"name": "Pending",

"description": "Pending"

},

{

"organization\_id": 16,

"name": "Trusty Apps",

"description": "Trusty Applications"

},

{

"organization\_id": 20,

"name": "Eleven Fifty Academy",

"description": "EFA"

}

]

}

## GET /organization

**GET Method:** Get the full details of a specific AIMS organization

**Request JSON:**

Parameter: int, organizationId

Example: http://aimswebapi.azurewebsites.net/api/v2/organization?organizationId=1

**Response JSON:**

Parameter: int, OrganizationId

Parameter: string, Name, maxlen = 256

Parameter: string, Description, maxlen = 512

Parameter: string, Address, maxlen = 100

Parameter: string, City, maxlen = 50

Parameter: string, State, maxlen = 50

Parameter: string, ZipCode, maxlen = 20

Parameter: string, PhoneNumber, maxlen = 20

Example:

{

“code”: 200,

“data”:

{

"OrganizationId": 16,

"Name": "Trusty Apps",

"Description": "Trusty Applications",

"Address": "5525 Shawnee Trail South Dr.",

"City": "Indianapolis",

"State": "IN",

"ZipCode": "46220",

"PhoneNumber": "3172597173"

}

}

## GET /groups/{organization\_id}

**GET Method:** Get the list of all groups within an organization

**Request JSON:**

Parameter: int, organization\_id to get the groups list for

http://aimswebapi.azurewebsites.net/api/v2/groups?organizationId=16

**Response JSON:**

Parameter: int, group\_id

Parameter: string, name, maxlen = 128

Example:

{

“code”: 200,

“message”: “Successfully retrieved groups in org”,

“data”:

[

{

"group\_id": 17,

"name": "Sales Team"

},

{

"group\_id": 18,

"name": "Testing Team"

}

]

}

## GET /surveys/{userId}

**GET Method:** Get the list of all INCOMPLETE surveys assigned to the user

**Request JSON:**

Parameter: int, userId to get the survey list for

The userId is the last number after the last “-“ in the Bearer Token:

"Token": "Bearer 065307ef-8036-46ea-bce4-bc6e8d2b9cb0-145",

Header: Authorization

http://aimswebapi.azurewebsites.net/api/v2/surveys?user\_id=145 -H “Authorization: token ba4604e8e433g9c892e360d53463oec5-145“

**Response JSON:**

Parameter: int, survey\_instance\_id, this is a survey instance in the user collection

Example:

{

“code”: 200,

“data”:

{

[

{

“survey\_id”: “14”,

}

{

“survey\_id”: “3”,

}

]

}

}

## GET /survey/{survey\_id}

**GET Method:** Get the details of a specific survey

**Request JSON:**

Parameter: int, survey\_instance\_id to get the details for

Header: Authorization

http://aimswebapi.azurewebsites.net/api/v2/survey?survey\_instance\_id=1 -H “Authorization: token ba4604e8e433g9c892e360d53463oec5-145“

**Response JSON:**

Parameter: string, survey\_name

Parameter: int, survey\_instance\_id

Parameter: Array of survey\_questions

Parameter: int, survey\_question\_id

Parameter: string, survey\_question\_text

Parameter: int, survey\_response\_type\*

Parameter: Array of survey\_responses

Parameter: int, survey\_response\_id

Parameter: string, maxLen 512, survey\_response\_text

Parameter: int, survey\_response\_value

Parameter: base64, survey\_response\_image

\*survey\_response\_type can be multiple choice (value = 0) or paragraph (value = 1). All multiple choice questions can have only 1 response, not many.

Example 1: *Reformatted nicely to match the spec above*

{

“code”: 200,

“data”:

{

"survey\_instance\_id": 1,

"survey\_name": "Employer Satisfaction",

"survey\_questions":

[

{

"survey\_question\_id": 1,

"survey\_question\_text": "My EFA resource has a stellar

attendance record",

"survey\_response\_type": 0,

"survey\_responses":

[

{

"survey\_response\_id": 1,

"survey\_response\_text": ""

“survey\_response\_value”: “5”

},

{

"survey\_response\_id": 2,

“survey\_response\_text”: “”

"survey\_response\_value": "4"

},

{

"survey\_response\_id": 3,

"survey\_response\_text": ""

"survey\_response\_value": "3"

},

{

"survey\_response\_id": 4,

"survey\_response\_text": ""

"survey\_response\_value": "2"

},

{

"survey\_response\_id": 5,

"survey\_response\_text": ""

"survey\_response\_value": "1"

}

]

},

{

"survey\_question\_id": 2,

"survey\_question\_text": "Please provide any additional

feedback to help us improve",

"survey\_response\_type": 1,

"survey\_responses": []

}

]

}

Example 2: *I didn’t reformat this one for you… focus on the levels, not the order.*

{

“code”: 200,

“data”:

{

"survey\_questions": [

{

"survey\_responses": [

{

"survey\_response\_id": 1007,

"survey\_response\_text": "Strongly Agree",

"survey\_response\_value": 5

},

{

"survey\_response\_id": 1008,

"survey\_response\_text": "Agree",

"survey\_response\_value": 4

},

{

"survey\_response\_id": 1009,

"survey\_response\_text": "Neutral",

"survey\_response\_value": 3

},

{

"survey\_response\_id": 1010,

"survey\_response\_text": "Disagree",

"survey\_response\_value": 2

},

{

"survey\_response\_id": 1011,

"survey\_response\_text": "Strongly Disagree",

"survey\_response\_value": 1

}

],

"survey\_question\_id": 1003,

"survey\_question\_text": "I'm happy with my EFA Employee(s)",

"survey\_response\_type": 0

},

{

"survey\_responses": [

{

"survey\_response\_id": 1012,

"survey\_response\_text": "True",

"survey\_response\_value": 100

},

{

"survey\_response\_id": 1013,

"survey\_response\_text": "False",

"survey\_response\_value": 101

}

],

"survey\_question\_id": 1004,

"survey\_question\_text": "I need to hire another entry level programmer now!",

"survey\_response\_type": 0

},

{

"survey\_responses": [],

"survey\_question\_id": 1005,

"survey\_question\_text": "Please supply added feedback.",

"survey\_response\_type": 1

}

],

"survey\_instance\_id": 3,

"survey\_name": "Employer Satisfaction 2016 Dec"

}

## POST /survey

**POST Method:** Post the details of a submitted survey.

**Request JSON:**

Parameter: int, survey\_instance\_id of completed survey

Header: Authorization

http://aimswebapi.azurewebsites.net/api/v2/survey?survey\_instance\_id=4 -H “Authorization: token ba4604e8e433g9c892e360d53463oec5-145“

**Response JSON:**

Parameter: string, survey\_name

Parameter: int, survey\_instance\_id

Parameter: Array of survey\_answers

Parameter: int, survey\_question\_id

Parameter: int, survey\_response\_id\*\*

Parameter: string, maxLen 512, survey\_response\_text

Parameter: int, survey\_response\_value\*\*

\*Note: As of API v1, no questions are ‘required’, so client users can submit empty or incomplete surveys. When POSTing empty or incomplete surveys, simply omit the survey\_answers array element for unanswered items.

\*\*Note: these parameters should be omitted for paragraph questions

Example 1: This is an example of submitting the survey shown in the GET /survey/suvey\_instance\_id, Example 2 above. This example shows that the user answered all of the survey questions. If they had not supplied an answer to the 2nd question, then the survey submission would simply exclude the green portion below.

{

"survey\_answers": [

{

"survey\_question\_id": 1003,

"survey\_response\_id": 1007,

"survey\_response\_text": "Strongly Agree",

"survey\_response\_value": 5

},

{

"survey\_question\_id": 1004,

"survey\_response\_id": 1012,

"survey\_response\_text": "True",

"survey\_response\_value": 100

},

{

"survey\_question\_id": 1005,

"survey\_response\_text": "Please contact me soon about hiring another Java dev."

}

],

"survey\_instance\_id": 3,

"survey\_name": "Employer Satisfaction 2016 Dec"

}

Example 2: This example shows that the user submitted a survey with no answers filled out:

{

"survey\_answers": [],

"survey\_instance\_id": 3,

"survey\_name": "Employer Satisfaction 2016 Dec"

}

# Appendix A – Developer Notes

* Although some parameters above state type “int”, if they are showed in quotes in the examples, they are exchanges as strings. The only parameter that is truly exchanges as a JSON int right now is the “code” parameter.
* We intend to use an email address for the username, please do light email validation on the client side.
* I didn’t want to convolute the survey model for MVP, so for now, please assume:
  1. If the question type is “MultipleChoice”
     1. A response is required
     2. Only 1 choice can be selected
  2. If the question type is “Paragraph”
     1. A response is optional
* This API document should cover the following MVP features (in order of priority). The highlights specify progress to date.
  + - Login / Forgot Password / Logout
    - Registration - Basic email address / password only
    - Weekly Score Card – Allow employers or apprentices to be sent a template survey to gather feedback
    - Notifications - Receive event notifications from the server
    - Daily Punch Card - Utilize geofences to check people in / out
    - User Profile – View and update user information
    - Course Schedule Calendar Interface with times & locations
    - Raise Hand / Ask Question - Allows users (apprentices, students, etc.) to ask questions, anonymously if desired