FOODND API

Version 1.1

Author: Brad Stalcup and Sean Murphy

Date: May 7, 2015

Overview

All API calls documented here consist of an HTTP method and a relative path; the path is relative to a configurable base URL.

Requests

Unless otherwise noted, all API calls receive requests in the form of JSON.

In general, the backend will balk if it receives unexpected parameters, so take care to send only the documented parameters and nothing else. All documented parameters are always required to be present unless otherwise noted

Responses

All responses, unless otherwise noted, are either formatted in JSON, whose MIME type is application/json, or HTML. Since JSON is typed, this documentation specifies the **TYPE** of each returned parameter, along with other notes.

JSON natively includes the types String, Number, Boolean, Array, Object as well as the singleton value null. In addition, this document uses *Integer* to refer to whole Numbers, *Datetime String* to refer to String values consisting of an absolute timestamp in the web-standard format ISO 8601, and *Money String* to refer to String values consisting of an unformatted representation of a floating point value.

All JSON responses include the errors array, which is the APIs mechanism for reporting errors.

PARAMETER	DESCRIPTION	TYPE
errors	A list of errors. This list is empty if and only if there were no errors in processing the request. Note that an array is always given; this member is never null or undefined.	Array of error objects (see **Error Object** below)

Error Object

PARAMETER	DESCRIPTION	TYPE
code	The error code.	Integer
message	The error message.	String

Resource overview

/users
/users/create
/users/login
/users/{userId}
/users/{userId}/votes
/users/{userId}/comments
/software/
/software/{softwareId}
/software/{softwareId}/comments
/software/{softwareId}/comments/{commentId}
/comments
/comments/{commentId}
/comments/{commentId}/votes

/users/

Action	Description
GET /users/	Get all user information
GET /users/login	For asynchronous logins
GET /users/create	For creating a user
GET /users/{userId}	Get information about a user
GET /users/{userId}/votes	Get a list of the user's votes
GET /users/{userId}/comments	Get a list of the user's comments

Data Model:

Field Name	Description	Type/Format	Required
userID	System-defined unique Customer identifier	20 character string	Y
name	Full name	String, up to 45 characters	Y
password	Hash using SHA-512 is stored, never full password	String, up to 45 characters	Y

JSON Form:

```
{
        name
        password
}
```

Error Messages:

Code	Text
9001	User Already Exists (upon creation)
9002	Incorrect User Password
9003	User Does not Exist

/Software/

Action	Description
GET /software	returns a list of all the software
GET /software/{softwareId}	Returns information about a particular software
GET /software/{softwareId}/comments	Returns a list of all the comments on a particular software
PUT /software/{softwareId}/comments	Creates or updates a comment
DELETE /software/{softwareId}/comments/{commentId}	Delete a comment

Data Model:

Field Name	Description	Type/Format	Required
Rank	1 or -1	Integer	Y
Comment	The content of the text	String	N
user	user's Id	Int	Y
software	software's Id	int	Y

JSON Form:

```
{
        rank
        comment
        userId
        softwareId
}
```

PUT Response:

Field Name	Description	Type/Format	Required
commentId	Generated by the API during PUT.	int	

Error Messages:

Code	Message
n/a	Some error happened in the query, the error message will be returned by the API

/Comments/

Action	Description
GET /comments	returns a list of all the comments
GET /comments/{commentId}	Returns information about a particular software
GET /comments/{commentId}/votes	Returns a list of all the votes on a particular comment
POST /comments/{commentId}/vote	Creates a vote on the comment with the logged in user
DELETE /comments/{commentId}/vote	Deletes a vote on the comment with the logged in user

Data Model:

Field Name	Description	Type/Format	Required
commentId	The desired Comment Id	Integer	Y

JSON Form:

```
{
        commentId
}
```

PUT Response:

Field Name	Description	Type/Format	Required

dishId	Generated by the API during PUT.	20 character string	

Error Messages:

Code	Message
9001	No logged in user