

RESTful Beer/Brewery API

Overall Description:

This API allows access to a database that holds information on breweries and beers. Beers are sold at breweries and users own breweries. Beer can only be added/removed from breweries if the user owns the brewery. The same goes for edit and delete functionality of breweries. Anyone can view information on beer or breweries.

BASE URL

All requests can be made to this base url: <https://finalproject-224219.appspot.com>

USERS

GET /users

- Gets a list of users.
- Response is an array of user's emails.

- ```
1 [{"email": "harrisri@baz.com"}, {"email": "harrisri@test.com"}]
```
- 200 status code returned on success

#### POST /users

- Creates a user
- Requires a JSON body of a username/password

```
{
 "username": "harrisri@test.com",
 "password": "abc123!"
}
```

- If password is not sufficiently complex, it will fail
- Example response:

```
1 - {
2 "email": "harrisri@baz.com",
3 "email_verified": false,
4 "updated_at": "2018-12-04T01:04:57.094Z",
5 "picture": "https://s.gravatar.com/avatar/0c58222f0ec0c04f0c3fccce26b4c4f?s=480&r=pg&d=
=https%3A%2F%2Fcdn.auth0.com%2Favatars%2Fha.png",
6 "user_id": "auth0I5c05d2b911bf6d2f2ce0e3d3",
7 - "identities": [
8 - {
9 "connection": "Username-Password-Authentication",
10 "user_id": "5c05d2b911bf6d2f2ce0e3d3",
11 "provider": "auth0",
12 "isSocial": false
13 }
14],
15 "created_at": "2018-12-04T01:04:57.094Z"
16 }
```

## POST /users/login

- Allows the user to 'login' and receive an id\_token. This token is needed for many operations.
- Requires username/password JSON body

```
{
 "username": "harrisri@test.com",
 "password": "abc123@!"
}
```

- Example response:

[illegible]

## DELETE /users/:username

- Deletes a user
- Requires an id\_token for the user for whom is to be deleted
  - ex: Greg can only delete Greg. Chris cannot delete Greg.

**PUT /users/:username**

- Change the users password.
  - Requires sufficient password complexity, else it will fail.
- Expects JSON body with the new password.

```
1 = {
2 "password": "newPassword"
3 }
```

- Requires an id\_token for the username in the url. Only that user can change their own password.

## BEER

Beer objects contain the below properties:

- Name
- Type
- alcoholPercentage

### GET /beer

- Gets a paginated (limit 5) list of beers.
- 'total\_number\_of\_beers' holds the total number of beers in the collection
- 'beers' holds an array of beer objects.
- 'next' holds a url for more beer objects (if more than 5 in the database)
- All beers have self links
- Able to send back application/json or text/html
- Every other type is rejected with a 406 code.
- Example response (JSON):

```
{
 "beer": [
 {
 "alcoholPercentage": 4.6,
 "name": "Lagunitas IPA",
 "type": "IPA",
 "id": "5178081291534336",
 "self": "http://localhost:8080/beer/5178081291534336"
 },
 {
 "type": "IPA",
 "alcoholPercentage": 4.6,
 "name": "Lagunitas IPA",
 "id": "5651124426113024",
 "self": "http://localhost:8080/beer/5651124426113024"
 },
 {
 "alcoholPercentage": 4.6,
 "name": "Lagunitas IPA",
 "type": "IPA",
 "id": "5656058538229760",
 "self": "http://localhost:8080/beer/5656058538229760"
 },
 {
 "alcoholPercentage": 4.6,
 "name": "Lagunitas IPA",
 "type": "IPA",
 "id": "5734055144325120",
 "self": "http://localhost:8080/beer/5734055144325120"
 },
 {
 "alcoholPercentage": 4.6,
 "name": "Lagunitas IPA",
 "type": "IPA",
 "id": "5741031244955648",
 "self": "http://localhost:8080/beer/5741031244955648"
 }
],
 "next": "http://localhost:8080/beer?cursor=CjASKmoVbXSmaWShbHB2p1Y3QtMjI0MjE5SchELEgRCZWVvGICAgICArpkKDBgAIAA=",
 "total_number_of_beers": 7
}
```

### GET /beer/:beerID

- Gets information on a single beer

- Able to send back application/json or text/html
- Every other type is rejected with a 406 code.
- Example response (JSON):

```
1 - {
2 "alcoholPercentage": 4.6,
3 "name": "Lagunitas IPA",
4 "type": "IPA",
5 "id": "5734055144325120",
6 "self": "http://localhost:8080/beer/5734055144325120"
7 }
```

## POST /beer

- Post a new beer
- Expects JSON body with below attributes. Any other attributes are ignored.

```
{
 "name": "Lagunitas IPA",
 "type": "IPA",
 "alcoholPercentage": 4.6
}
```

- Response contains id of newly created beer

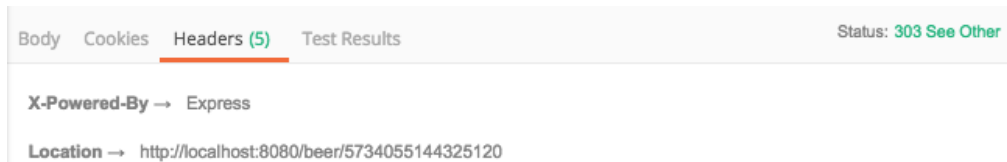
```
1 [{"id": 5734055144325120 }]
```

## PUT /beer/:beerID

- Edit a beer's properties
- Expects JSON body with below attributes. Any other attributes are ignored.

```
{
 "name": "Lagunitas IPA",
 "type": "IPA",
 "alcoholPercentage": 4.6
}
```

- All previous properties will be overwritten with the properties in the JSON body.
- On success, 303 status code is returned and Location header contains url to the updated beer.



```
Body Cookies Headers (5) Test Results Status: 303 See Other

X-Powered-By → Express
Location → http://localhost:8080/beer/5734055144325120
```

## DELETE /beer/:beerID

- Deletes a beer
- 204 returned on successful delete.

## BREWERIES

Brewery objects contain the below properties:

- Name
- Location
- Year\_Founded
- Owner
- Beer (list of beer sold at brewery)

### GET /breweries

- Gets a paginated (limit 5) list of breweries.
- Anyone can view.
- 'total\_number\_of\_breweries' holds the total number of breweries in the collection
- 'breweries' holds an array of brewery objects.
- 'next' holds a url for more brewery objects (if more than 5 in the database)
- All breweries have self links
- Able to send back application/json or text/html
- Every other type is rejected with a 406 code.
- Example response (JSON):

```
1 - {
2 - "breweries": [
3 - {
4 - "location": "Petaluma, CA",
5 - "beer": [],
6 - "owner": "harrisri@test.com",
7 - "name": "Lagunitas",
8 - "yearFounded": "1999",
9 - "id": "5654645158445056",
10 - "self": "http://localhost:8080/breweries/5654645158445056"
11 - },
12 - {
13 - "yearFounded": "1999",
14 - "location": "Petaluma, CA",
15 - "beer": [],
16 - "owner": "harrisri@test.com",
17 - "name": "Lagunitas",
18 - "id": "5660980839186432",
19 - "self": "http://localhost:8080/breweries/5660980839186432"
20 - },
21 - {
22 - "location": "Petaluma, CA",
23 - "beer": [],
24 - "owner": "harrisri@test.com",
25 - "name": "Lagunitas",
26 - "yearFounded": "1999",
27 - "id": "5736627494191104",
28 - "self": "http://localhost:8080/breweries/5736627494191104"
29 - },
30 - {
31 - "yearFounded": "1999",
32 - "location": "Petaluma, CA",
33 - "beer": [],
34 - "owner": "harrisri@test.com",
35 - "name": "Lagunitas",
36 - "id": "5737442330017792",
37 - "self": "http://localhost:8080/breweries/5737442330017792"
38 - },
39 - {
40 - "beer": [],
41 - "owner": "harrisri@test.com",
42 - "name": "Lagunitas",
43 - "yearFounded": "1999",
44 - "location": "Petaluma, CA",
45 - "id": "5745237494333440",
46 - "self": "http://localhost:8080/breweries/5745237494333440"
47 - }
48 -],
49 - "next": "http://localhost:8080/breweries?cursor=CjMSLWoVbXSmawShbHByb2plY3QtMjI0MjE5chQLEgdCcmV3ZXJ5GICAgMC1aJokDBgAIAA=",
50 - "total_number_of_breweries": 6
51 - }
```

### GET /breweries/:breweryID

- Gets information on a single brewery
- Able to send back application/json or text/html

- Every other type is rejected with a 406 code.
- Example response (JSON):

```

1 - {
2 "beer": [],
3 "owner": "harrisri@test.com",
4 "name": "Lagunitas",
5 "yearFounded": "1999",
6 "location": "Petaluma, CA",
7 "id": "5672330625810432",
8 "self": "http://localhost:8080/breweries/5672330625810432"
9 }

```

## POST /breweries

- Post a new brewery to the database
- Requires an id\_token from /users/login
- Only accepts application/json. Other types will be ignored and a 415 status sent.
- Expects JSON body with below properties. Note that beer is not included. If included, it will be ignored. Beer needs to be added with PUT /breweries/:breweryID/beer/:beerID (below)
  - Owner property will be populated with the provided id\_token

```

1 - {
2 "name": "Lagunitas",
3 "yearFounded": "1999",
4 "location": "Petaluma, CA"
5 }

```

- Response holds ID of newly created brewery.

```

{ 1 } { "id": 5654645158445056 }

```

- 204 sent on success, and location header populated with url of new brewery.

## PUT /breweries/:breweryID

- Edit a brewery
- Requires an id\_token from /users/login
  - id\_token must match owner of brewery. Else, the operation will be canceled and a 403 is sent.
- Only accepts application/json. Other types will be ignored and a 415 status sent.
- Expects JSON body with any of the brewery properties. ID will never be changed.

```

1 - {
2 "id": "not_the_right_id",
3 "name": "name_edit_success",
4 "location": "location_edit_success",
5 "yearFounded": 1000,
6 "beer": [2434, 32342, 423234, 4324324]
7 }

```

- 303 status code is sent on success, and Location header is populated with url of updated brewery.

Body Cookies Headers (5) Test Results Status: 303 See Other

X-Powered-By → Express

Location → http://localhost:8080/breweries/5662274280407040

Date → Sun, 02 Dec 2018 23:44:59 GMT

Connection → keep-alive

Content-Length → 0

**DELETE /breweries/:breweryID**

- Delete a brewery
- Requires an id\_token from /users/login
  - id\_token must match owner of brewery. Else, the operation will be canceled and a 403 is sent.
- 204 is sent on successful delete.

**PUT /breweries/:breweryID/beer/:beerID**

- Adds a beer to the brewery
- Requires an id\_token from /users/login
  - id\_token must match owner of brewery. Else, the operation will be canceled and a 403 is sent.
- Beer must not belong to another brewery, else 403 is sent.
- 303 is sent on success and Location header is populated with url of brewery

**DELETE /breweries/:breweryID/beer/:beerID**

- Removes a beer from the brewery
- Requires an id\_token from /users/login
  - id\_token must match owner of brewery. Else, the operation will be canceled and a 403 is sent.
- Beer must be in the brewery, else 403 is sent.
- 303 is sent on success and Location header is populated with url of brewery