Brent Johnson
Kennel Documentation
Final Project
CS 493-400 Fall 2018

This is a system for users to create pets and assign them to a kennel. See below for endpoints. JSON is the only supported format.

I had a hard time getting Postman to run these automatically. I was able to do so with a long delay (6000ms). It works when run manually one at a time in order. All requirements were met and should have tests in Postman except:

- *The collection must include a property that indicates how many total items are in the collection*
    - I spent some time trying to figure this out. What I was trying to do was get the collection without paging, and do a count. I kept getting promises errors, and was afraid I'd break the rest of the code if I kept trying.
- *user creation to Auth0*
    - Using the users already added to Auth0, the system does support Authentication and Authorization. I did not want to break things by trying to support this.
    - https://piazza.com/class/jm704r7q4ot3s1?cid=283 has what looks like good information about this if I was to attempt it.
- Deleting pets does not remove it from the kennel. I cannot get this to work. For some reason, the filter in line 361 isn't picking up the pet_id = req.params.id. It's the same basic code as I used in a different assignment which did work. I can confirm there are entities that it should pick up, but it doesn't. I checked Google's documentation and as near as I can tell, this is correct and should work. Other than asking for help, I'm out of ideas.

Some notes to help with grading:

- Pets are owned by users, kennels are not.
- GET /pets and GET /users are not restricted by JWT for grading and testing purposes of paging. It would be a simple matter of wrapping that into if statements to implement this.

ENDPOINTS

GET /pets

Returns an array of pets, and a next if there are more than 5 (only showing 2 for space considerations instead of 5)

Returns Status 200 and the following:


```
{
   "pets": [
      {
         "name": "Fluffy",
         "breed": "Cat",
         "desc": "White",
         "id": "5660313307316224",
         "self": "http://localhost:8080/pets/5660313307316224"
      },
      {
         "breed": "Poodle",
         "desc": "White",
         "name": "Fifi",
         "id": "5676533754626048",
         "self": "http://localhost:8080/pets/5676533754626048"
      }
   ],
   "next":
"http://localhost:8080/pets?cursor=CiYSIGoLc35maW5hbC00OTNyEQsSBFBldHMYgICAoIXZjwoMGAAgAA=="
}
```


GET /pets/:pet_id

Return one pet by pet id with status 200. Example:


```
[
   {
      "desc": "Small With Big Ears",
      "status": "Z2",
      "name": "Mini",
      "breed": "Mouse",
      "id": "5698529657880576",
      "self": "http://localhost:8080/pets/5698529657880576",
      "base": "http://localhost:8080/pets"
   } ]
```


POST /pets

Creates a new pet in the system. Returns status 201.
Requires login. Returns 401 if not logged in and 403 if forbidden.
Required body fields are:

```
{
    "name": "text",
    "breed": "text",
    "desc": "text"
}
```

DELETE /pets/:pet_id

Deletes pet by pet id. Returns 204. Only users who are logged in and the owner of the pet is allowed to delete it.
Requires login. Returns 401 if not logged in and 403 if forbidden.

PUT /pets/:pet_id

Updates name, breed and desc. Overwrites with “” if not there.
Requires login. Returns 401 if not logged in and 403 if forbidden.

```
{
    "name": "text",
    "breed": "text",
    "desc": "text"
}
```

PATCH /pets/:pet_id

Updates name, breed and desc. Only overwrites what is given. You can use any/all of the following:
Requires login. Returns 401 if not logged in and 403 if forbidden.

```
{
    "name": "text",
    "breed": "text",
    "desc": "text"
}
```

GET /kennels

Returns an array of kennels, and a next if there are more than 5. (only showing 2 for space considerations instead of 5)
Returns Status 200 and the following:


```
{
   "kennels": [
      {
         "number": "B1",
         "size": "medium",
         "desc": "simple crate",
         "id": "5632490240737280",
         "self": "http://localhost:8080/kennels/5632490240737280"
      },
      {
         "number": "B1",
         "size": "medium",
         "desc": "simple crate",
         "id": "5638350320959488",
         "self": "http://localhost:8080/kennels/5638350320959488"
      },
   ],
   "next":
"http://localhost:8080/kennels?cursor=CikSI2oLc35maW5hbC00OTNyFAsSB0tlbm5lbHMYgICAoJSQkAoMGAAgAA=="
}
```


GET /kennels/:kennel_id

Return one kennel by kennel id with status 200.


```
[
   {
      "number": "A1",
      "size": "small",
      "desc": "No outside access; for cats",
      "id": "5662784121470976",
      "self": "http://localhost:8080/kennels/5662784121470976",
      "base": "http://localhost:8080/kennels"
   }
]
```


POST /kennels

Creates a new kennel in the system. Returns status 201. Required body fields are:

```
{
    "number": "text",
    "size": "text",
    "desc": "text"
}
```

DELETE /kennels/:kennel_id

Deletes kennel by kennel id. Returns 204

PUT /kennels/:kennel_id

Updates kennel's number, size and description. Overwrites with "" if not there.

```
{
    "number": "text",
    "size": "text",
    "desc": "text"
}
```

PATCH /kennels/:kennel_id

Updates kennel's number, size and description. Only overwrites what is given. You can use any/all of the following:

```
{
    "number": "text",
    "size": "text",
    "desc": "text"
}
```

PUT /pets/:pet_id/kennels/:kennel_id

Assigns pet to a kennel. Returns 201.

Requires login. Returns 401 if not logged in and 403 if forbidden.

DELETE /pets/:pet_id/kennels/:kennel_id

Removes pet from a kennel. Returns 204.

Requires login. Returns 401 if not logged in and 403 if forbidden.

GET /kennels/:kennel_id/pets

Gets the pet information assigned to the kennel id. See GET /pets/:pet_id for the data that comes back.

Returns a 406 if kennel does not have a pet assigned to it.

GET /users

Returns an array of users, and a next if there are more than 5 (only showing 2 for space considerations instead of 5)
Returns Status 200 and the following:


```
{
   "users": [
      {
         "last": "GenericLastName6Here",
         "first": "GenericFirstName6Here",
         "email": "user6@someAddress.com",
         "id": "5203912365703168",
         "self": "http://localhost:8080/users/5203912365703168"
      },
      {
         "last": "GenericLastName3Here",
         "first": "GenericFirstName3Here",
         "email": "user3@someAddress.com",
         "id": "5630441608445952",
         "self": "http://localhost:8080/users/5630441608445952"
      }
   ],
   "next":
"http://localhost:8080/users?cursor=CicSIWoLc35maW5hbC00OTNyEgsSBVVzZXJzGICAgKCU9IkKDBgAIAA="
}
```


POST /users

Creates a new user in the system. Returns status 201. Required body fields are:


```
{
         "email": "text",
         "first": "text",
         "last": "text"
}
```


GET /users/:user_id

Requires login. Returns 401 if not logged in and 403 if forbidden.
Return one user by user id with status 200. Example:


```
[
  {
    "last": "Smithers",
    "first": "Jonny",
    "email": "someRandomUsername@someAddress.com",
    "id": "5647979100766208",
    "self": "http://localhost:8080/users/5647979100766208",
    "base": "http://localhost:8080/users"
  }
]
```


PUT /users/:user_id

Requires login. Returns 401 if not logged in and 403 if forbidden.
Updates email, first and last. Overwrites with “” if not there.


```
{
        "email": "text",
        "first": "text",
        "last": "text"
}
```


PATCH users/:user_id

Requires login. Returns 401 if not logged in and 403 if forbidden.
Updates email, first and last. Only overwrites what is given. You can use any/all of the following:


```
{
        "email": "text",
        "first": "text",
        "last": "text"
}
```


DELETE /users/:user_id

Requires login. Returns 401 if not logged in and 403 if forbidden.
Deletes user by user id. Returns 204

DELETE /pets
DELETE /kennels
DELETE /users

Not Supported, returns 405