Harbor Master RESTful API Documentation

https://a9-portfolio.wl.r.appspot.com

CS 493: Cloud Application Development

Spring 2023

Oregon State University

Last Update: June 11. 8:40 pm Pacific, parky8

Change log...

Data Model ....

Boat

Create a Boat ....

PUT a Boat..

PATCH a Boat ....

Get a Boat ....

List all Boats ....

Delete a Boat...

List all Loads on a Boat...

Assign a Load to a Boat ....

Unasssign a Load from a Boat...

Load

Create a Load ....

Get a Load ....

List all Loads ....

Delete a Load ....

PUT a Load ....

PATCH a Load ....

User

Get a User ....

List all Users ....

Change log

Version	Change	Date
1.0	Initial version.	June 11, 2023

Data Model

The app stores three kinds of entities in Datastore, Boats, Loads, and Users. JWT provided by Auth0. “sub” property from JWT used as the UID for each user. This UID is used to denote boat ownership and to protect the boat resource. A Bearer Token matching the boat owner’s UID is required to access boats. Load resources are not protected. Editing either load or boat will reset the assigned relationships if there exists one. Deleting a load will also delete it from the list of loads on the boat entity. Deleting a boat will not delete a load. User entity cannot be edited through RESTful requests. A user is automatically added to the database when a user signs up using Auth0.

Boats

Property	Data Type	Notes
id	Integer	The id of the boat. Datastore automatically generates it. Don't add it yourself as a property of the entity.
name	String	Name of the boat.
type	String	Type of the boat. E.g., Sailboat, Catamaran, etc.
length	Integer	The length of the boat in feet.
loads	Array	An array of loads assigned to the boat. Each load is represented by an object with the keys self and id.

Loads

Property	Data Type	Notes
id	Integer	The id of the load. Datastore automatically generates it. Don't add it yourself as a property of the entity.
volume	Integer	The volume of the load.
item	String	The item that the load contains.
creation_date	String	A manually entered string that denotes the creation date of the load.
carrier	Object	The boat that is assigned to the load. The object will contain the name, self and id of the boat if assigned. If not assigned, the value will be null.

Users

Property	Data Type	Notes
uid	String	The unique id of the user. This is the “sub” from auth0.
nickname	String	The nickname of the user.
id	String	The id of the user on Datastore.

Boat

- The following requests will return a 405 Method Not Allowed response from the server:

PUT /boats
DELETE /boats

Create a Boat

Allows you to create a new boat.

Requires a Bearer Token that is displayed in the web app after login. Owner automatically added based on the token sent in with the request. Loads added as empty upon request. To add a load, go to “Assign a Load to a Boat”.

POST /boats

Request

Path Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	**Description**	**Required?**
name	The name of the boat.	Yes
type	The type of the boat. E.g., Sailboat, Catamaran, etc.	Yes
length	Length of the boat in feet.	Yes

Request Body Example

```
{
    "name": "Sea Witch",
   "type": "Catamaran",
    "length": 28
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201 Created	The response body will contain a JSON detailing the created entity
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the boat must not be created, and 400 status code must be returned. You don't need to validate the values of the attributes and can assume that if the request contains any of the listed attributes, then that attribute's value is valid. You can also assume that the request will not contain any extraneous attribute (i.e., the request JSON will never contain any attribute that is not listed above).
	401 Unauthorized	If the request is missing or has an invalid JWT as the Bearer Token, a 401 status code will be returned
	415 Unsupported Media Type	If the content-type header is not set to application/json, a 415 status code will be returned
	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned

Response Examples

- Datastore will automatically generate an ID and store it with the entity being created.

Success

```
Status: 201 Created

{
    "id": 123,
    "name": "Sea Witch",
  "type": "Catamaran",
    "length": 28,
    "loads": [],
     "owner": “auth|owner_uid”,
    "self": "http://URL/boats/123
}
```

Failure

```
Status: 400 Bad Request

{
"Error":  " The request object is missing at least one of the required attributes"
}
```

```
Status: 401 Unauthorized

{
"Error":  " Missing or invalid jwt"
}
```

```
Status: 415 Unsupported Media Type

{
"Error":  "Server only accepts application/json data"
}
```

```
Status: 406 Not Acceptable

{
"Error":  "Server only accepts application/json data"
}
```

Put a Boat

Allows you to edit a boat. The PUT request requires that the body specifies 3 properties: name, type, and length. The owner stays the same in accordance with the Bearer Token. Note that all assigned loads are unassigned. The boat that has been edited will have an empty “loads” property.

Requires a Bearer Token that is displayed in the web app after login. To add a load, go to “Assign a Load to a Boat”.

```
PUT /boats/:boat_id
```

Request

Path Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
name	The name of the boat.	Yes
type	The type of the boat. E.g., Sailboat, Catamaran, etc.	Yes
length	Length of the boat in feet.	Yes

Request Body Example

```
{
    "name": "Sea Witch",
   "type": "Catamaran",
    "length": 28
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	303 See Other	The response body will contain a JSON detailing the edited entity. Property “self” contains the link to the edited entity.
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the boat must not be created, and 400 status code must be returned. You don’t need to validate the values of the attributes and can assume that if the request contains any of the listed attributes, then that attribute’s value is valid. You can also assume that the request will not contain any extraneous attribute (i.e., the request JSON will never contain any attribute that is not listed above).
	401 Unauthorized	If the request is missing or has an invalid JWT as the Bearer Token, a 401 status code will be returned
	415 Unsupported Media Type	If the content-type header is not set to application/json, a 415 status code will be returned
	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned
	404 Not Found	No boat with this boat_id exists
	403 Forbidden	If the boat owner does not match the Bearer Token, a 403 status code will be returned

Response Examples

- The id property will remain the same after the PUT request.

Success

```
Status: 303 See Other

{
    "id": 123,
    "name": "Sea Witch",
  "type": "Catamaran",
    "length": 28,
    "loads": [],
     "owner": “auth|owner_uid”,
    "self": "http://URL/boats/123
}
```

Failure

```
Status: 400 Bad Request
```

```
{
"Error":  " The request object is missing at least one of the required attributes "
}
```

Status: 401 Unauthorized

```
{
"Error":  " Missing or invalid jwt"
}
```

Status: 415 Unsupported Media Type

```
{
"Error":  "Server only accepts application/json data"
}
```

Status: 406 Not Acceptable

```
{
"Error":  "Server only accepts application/json data"
}
```

Status: 404 Not Found

```
{
"Error":  "No boat with this boat_id exists"
}
```

Status: 403 Forbidden

```
{
"Error":  " Boat is owned by different owner"
}
```

Patch a Boat

Allows you to edit a boat. The PATCH request has 3 optional body properties: name, type, and length. The PATCH request will only edit the properties in the body. The owner stays the same in accordance with the Bearer Token. Note that all assigned loads are unassigned. The boat that has been edited will have an empty “loads” property.

If you have property “id” in the request, the server will return a 400, as boat_id cannot be patched.

Requires a Bearer Token that is displayed in the web app after login. To add a load, go to “Assign a Load to a Boat”.

PATCH /boats/:boat_id

Request

Path Parameters

None

Request Body

Optional

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
name	The name of the boat.	No
type	The type of the boat. E.g., Sailboat, Catamaran, etc.	No
length	Length of the boat in feet.	No

Request Body Example

```
{
    "name": "Sea Witch",
    "type": "Catamaran"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	303 See Other	The response body will contain a JSON detailing the edited entity. Property “self” contains the link to the edited entity.
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the boat must not be created, and 400 status code must be returned. You don’t need to validate the values of the attributes and can assume that if the request contains any of the listed attributes, then that attribute’s value is valid. You can also assume that the request will not contain any extraneous attribute (i.e., the request JSON will never contain any attribute that is not listed above).
	401 Unauthorized	If the request is missing or has an invalid JWT as the Bearer Token, a 401 status code will be returned
	415 Unsupported Media Type	If the content-type header is not set to application/json, a 415 status code will be returned
	406 Not Acceptable	If application/json is not accepted as a response, a 406

		status code will be returned
	404 Not Found	No boat with this boat_id exists
	403 Forbidden	If the boat owner does not match the Bearer Token, a 403 status code will be returned

Response Examples

- The id property will remain the same after the PUT request.

Success

```
Status: 303 See Other

{
    "id": 123,
    "name": "Sea Witch",
  "type": "Catamaran",
    "length": 28,
    "loads": [],
     "owner": “auth|owner_uid”,
    "self": "http://URL/boats/123
}
```

Failure

```
Status: 400 Bad Request

{
"Error":  " The request object is missing at least one of the required attributes "
}
```

```
Status: 401 Unauthorized

{
"Error":  " Missing or invalid jwt"
}
```

```
Status: 415 Unsupported Media Type

{
"Error":  "Server only accepts application/json data"
}
```

```
Status: 406 Not Acceptable

{
"Error":  "Server only accepts application/json data"
}
```

```
Status: 404 Not Found

{
```

```
"Error":  "No boat with this boat_id exists"
}
```

```
Status: 403 Forbidden

{
"Error":  " Boat is owned by different owner"
}
```

Get a Boat

Allows you to get an existing boat.

Requires a Bearer Token that is displayed in the web app after login. Owner of the requested boat must match token’s indicated owner.

```
GET /boats/:boat_id
```

Request

Path Parameters

Name	Description
boat_id	ID of the boat

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	404 Not Found	No boat with this boat_id exists
	401 Unauthorized	If the request is missing or has an invalid JWT as the Bearer Token, a 401 status code will be returned
	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned
	403 Forbidden	If the boat owner does not match the Bearer Token, a 403 status code will be returned

Response Examples

Success

```
Status: 200 OK

{
    "id": 123,
    "name": "Sea Witch",
  "type": "Catamaran",
    "length": 28,
    "loads": [],
     "owner": “auth|owner_uid”,
    "self": "http://URL/boats/123
}
```

Failure

```
Status: 404 Not Found

{
"Error":  "No boat with this boat_id exists"
}
```

Status: 401 Unauthorized

```
{
"Error":  " Missing or invalid jwt"
}
```

Status: 406 Not Acceptable

```
{
"Error":  "Server only accepts application/json data"
}
```

Status: 403 Forbidden

```
{
"Error":  " Boat is owned by different owner"
}
```

List all Boats

List all the boats in which owner matches the Bearer Token.

Requires a Bearer Token that is displayed in the web app after login. Only boats that are owned by the owner matching the Bearer Token are returned. Key “count” holds the total count of boats returned.

Pagination: 5 Boats per page, with key "next" holding a URL to the next page.

```
GET /boats
```

Request

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	401 Unauthorized	If the request is missing or has an invalid JWT as the Bearer Token, a 401 status code will be returned
	406 Not Acceptable	If application/json is not accepted as a response, a 406

		status code will be returned

Response Examples

Success

```
{

        "count": 3,
        "boats": [
                {
                        "name": "Sea Witch",
                        "length": 28,
                        "loads": [],
                        "type": "Catamaran",
                        "owner": "auth|owner_uid",
                        "id": "5632499082330112"
                },
                {
                        "name": "Sea Wizard",
                        "length": 27,
                        "loads": [],
                        "type": "Catamaran",
                        "owner": "auth|owner_uid",
                        "id": "5634161670881280"
                },
                {
                        "name": "Sea Dragon",
                        "length": 48,
                        "loads": [],
                        "type": "Catamaran",
                        "owner": "auth|owner_uid",
                        "id": "5644004762845184"
                }
        ],
        "next":
"http://localhost:8080/boats?cursor=CigSImoNbX5hNC1yZXN0LWFwaXIRCxIEQm9hdBiAgICYlKaDCgw
YACAA"
}
```

Failure

```
Status: 401 Unauthorized

{
"Error":  " Missing or invalid jwt"
}
```

```
Status: 406 Not Acceptable

{
```

```
"Error":  "Server only accepts application/json data"
}
```

Delete a Boat

Allows you to delete a Boat.

Requires a Bearer Token that is displayed in the web app after login. Only boats that are owned by the owner matching the Bearer Token can be deleted.

If the Boat being deleted is assigned to loads, the loads are unassigned from the boat before deleting. The loads are not deleted and have a carrier of *null* after deletion.

```
DELETE /boats/:boat_id
```

Request

Path Parameters

Name	**Description**
boat_id	ID of the boat

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	204 No Content	
Failure	404 Not Found	No boat with this boat_id exists
	401 Unauthorized	If the request is missing or has an invalid JWT as the Bearer Token, a 401 status code will be returned
	403 Forbidden	If the boat owner does not match the Bearer Token, a 403 status code will be returned

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found

{
```

```
"Error":  "No boat with this boat_id exists"
}
```

```
Status: 401 Unauthorized

{
"Error":  " Missing or invalid jwt"
}
```

```
Status: 403 Forbidden

{
"Error":  " Boat is owned by different owner"
}
```

List all Loads on a Boat

List all the loads.

Requires a Bearer Token that is displayed in the web app after login. Only boats that are owned by the owner matching the Bearer Token can be requested.

```
GET /boats/:boat_id/loads
```

Request

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	404 Not Found	No boat with this boat_id exists
	401 Unauthorized	If the request is missing or has an invalid JWT as the Bearer Token, a 401 status code will be returned
	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned
	403 Forbidden	If the boat owner does not match the Bearer Token, a 403 status code will be returned

Response Examples

Success

```
{
    "loads": [
        {
            "id": "5707651547660288",
            "item": "Stuffed Animals",
            "creation_date": "10/19/2021",
            "volume": 10,
            "self": "http://127.0.0.1:8080/loads/5707651547660288"
        },
        {
            "id": "5704568633556992",
            "item": "Food",
            "creation_date": "10/19/2021",
            "volume": 3,
            "self": "http://127.0.0.1:8080/loads/5704568633556992"
        },
        {
            "id": "5144701594238976",
            "item": "Furnitures",
            "creation_date": "10/20/2021",
            "volume": 7,
            "self": "http://127.0.0.1:8080/loads/5144701594238976"
        },
        {
            "id": "5700228501995520",
            "item": "Toys",
            "creation_date": "10/21/2021",
            "volume": 4,
            "self": "http://127.0.0.1:8080/loads/5700228501995520"
        }
    ]
}
```

Failure

```
Status: 404 Not Found

{
"Error":  "No boat with this boat_id exists"
}
```

```
Status: 401 Unauthorized

{
"Error":  " Missing or invalid jwt"
}
```

Status: 406 Not Acceptable

```
{
"Error":  "Server only accepts application/json data"
}
```

Status: 403 Forbidden

```
{
"Error":  " Boat is owned by different owner"
}
```

Assign a Load to a Boat

Load assigned to a Boat.

Requires a Bearer Token that is displayed in the web app after login. Only boats that are owned by the owner matching the Bearer Token can be assigned to a load.

A load that is already assigned cannot be requested to be assigned to another boat. Use “Unassign a Load from a Boat” first before assigning the load to another boat.

A load cannot be assigned to multiple boats. A boat can carry multiple loads.

PUT /boats/:boat_id/loads/:load_id

Request

Path Parameters

Name	Description
boat_id	ID of the boat
load_id	ID of the load

Request Body

None

Note: Set Content-Length to 0 in your request when calling out to this endpoint.

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	Succeeds only if a boat exists with this boat_id, a load exists with this load_id and the load is not assigned.
Failure	403 Forbidden	The load is already assigned to a boat.
	404 Not Found	The specified boat and/or load does not exist
	401 Unauthorized	If the request is missing or has an invalid JWT as the Bearer Token, a 401 status code will be returned
	403 Forbidden	If the boat is owned by a different owner, a 403 status code will be returned

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 403 Forbidden

{
	"Error": "The load is already loaded on another boat"
}
```

```
Status: 404 Not Found

{
	"Error": "The specified boat and/or load does not exist"
}
```

```
Status: 401 Unauthorized

{
"Error":  " Missing or invalid jwt"
}
```

```
Status: 403 Forbidden

{
"Error":  " Boat is owned by different owner"
}
```

Unassign a Load from a Boat

Unassign a Load from a Boat.

Requires a Bearer Token that is displayed in the web app after login. Only boats that are owned by the owner matching the Bearer Token can be unassigned from a load.

```
DELETE /boats/:boat_id/loads/:load_id
```

Request

Path Parameters

Name	Description
load_id	ID of the load
boat_id	ID of the boat

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	Succeeds only if a boat exists with this boat_id, a load exists with this load_id and this load is at this boat.
Failure	404 Not Found	No load with this load_id is at the boat with this boat_id. This could be because no boat with this boat_id exists, or because no load with load_id exists, or even if both boat_id and load_id are valid, the load with this load_id is not at this boat with this boat_id
	401 Unauthorized	If the request is missing or has an invalid JWT as the Bearer Token, a 401 status code will be returned
	403 Forbidden	If the boat is owned by a different owner, a 403 status code will be returned

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found
{
	"Error": "No boat with this boat_id is loaded with the load with this load_id"
}
```

```
Status: 401 Unauthorized

{
"Error":  " Missing or invalid jwt"
}
```

```
Status: 403 Forbidden
```

```
{
"Error":  " Boat is owned by different owner"
}
```

Load

- Load does not require a Bearer Token.
- The following requests will return a 405 Method Not Allowed response from the server:

PUT /loads
DELETE /loads

Create a Load

Allows you to create a new load.

To assign a load to a boat, see “Assign a Load to a Boat”. This request will require a Bearer Token that matches the “owner” property of the boat.

POST /loads

Request

Path Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
volume	The volume of the load.	Yes
item	The item of the load.	Yes
creation_date	The date of creation.	Yes

Request Body Example

```
{
  "volume": 5,
  "item": "LEGO Blocks",
  "creation_date": "10/18/2021"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes

Success	201 Created	
Failure	400 Bad Request	If the request is missing the number attribute, the load must not be created, and 400 status code must be returned. You don't need to validate the values of this attribute and can assume that if the number attribute is specified, then its value is valid. You can also assume that the request will not contain any extraneous attribute (i.e., the request JSON will never contain any attribute other than number).
	415 Unsupported Media Type	If the content-type header is not set to application/json, a 415 status code will be returned
	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned

Response Examples

- Datastore will automatically generate an ID.
- The value of the attribute `carrier` lists the ID, name and self of the boat assigned to this load. If there is no boat assigned to this load, the value of `carrier` should be null.

Success

Status: 201 Created

```
{
    "id": "5636645067948032",
    "volume": 5,
    "carrier": null,
    "item": "LEGO Blocks",
    "creation_date": "10/18/2021",
    "self": "http://localhost:8080/loads/5636645067948032"
}
```

Failure

Status: 400 Bad Request

```
{
"Error":  "The request object is missing the required number"
}
```

Status: 415 Unsupported Media Type

```
{
"Error":  "Server only accepts application/json data"
}
```

Status: 406 Not Acceptable

```
{
"Error":  "Server only accepts application/json data"
}
```

Get a Load

Allows you to get an existing load.

```
GET /loads/:load_id
```

Request

Path Parameters

Name	Description
load_id	ID of the load

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	404 Not Found	No load with this load_id exists
	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned

Response Examples

Success

```
Status: 200 OK

{
        "volume": 5,
        "carrier": null,
        "item": "LEGO Blocks",
        "creation_date": "10/18/2021",
        "id": "5631671361601536",
        "self": "http://127.0.0.1:8080/loads/5631671361601536"
}
```

Failure

```
Status: 404 Not Found

{
        "Error": "No load with this load_id exists"
}
```

```
Status: 406 Not Acceptable

{
"Error":  "Server only accepts application/json data"
}
```

List all Loads

List all the loads.

Key “count” holds the total count of loads returned. Pagination: 5 Loads per page, with key "next" holding a URL to the next page.

```
GET /loads
```

Request

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	
Failure	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned

Response Examples

Success

```
Status: 200 OK

{
        "count": 3,
        "loads": [
                {
                        "volume": 21,
                        "carrier": null,
                        "item": "iPhone",
                        "creation_date": "10/18/2021",
                        "id": "5631671361601536"
                },
                {
                        "volume": 2,
                        "carrier": null,
                        "item": "Furniture",
                        "creation_date": "10/12/2021",
                        "id": "5636645067948032"
                },
                {
                        "volume": 5,
                        "carrier": null,
                        "item": "LEGO Blocks",
                        "creation_date": "10/11/2021",
                        "id": "5642368648740864"
                }
        ],
        "next":
"http://127.0.0.1:8080/loads?cursor=CigSImoNbX5hNC1yZXN0LWFwaXIRCxIETG9hZBiAgICYxfaCCgwY
ACAA"
}
```

Failure

```
Status: 406 Not Acceptable

{
"Error":  "Server only accepts application/json data"
}
```

Delete a Load

Allows you to delete a boat. Note that if the load is currently assigned to a boat, deleting the load will unassign the load from the assigned boat.

```
DELETE /loads/:load_id
```

Request

Path Parameters

Name	Description
load_id	ID of the load

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	404 Not Found	No load with this load_id exists

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found

{
"Error":  "No load with this load_id exists"
}
```

Put a Load

Allows you to edit a load. The PUT request requires that the body specifies 3 properties: volume, item, and creation_date. Note that the assigned boat is unassigned after a successful edit. The load that has been edited will have “carrier” property set to *null*.

To assign a boat, go to “Assign a Load to a Boat”. This request will require a Bearer Token matching the owner of the boat.

PUT /loads/:load_id

Request

Path Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
volume	The volume of the load.	Yes
item	The item of the load.	Yes
creation_date	The date of creation.	Yes

Request Body Example

```
{
  "volume": 5,
  "item": "LEGO Blocks",
  "creation_date": "10/18/2021"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	303 See Other	The response body will contain a JSON detailing the edited entity. Property “self” contains the link to the edited entity.
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the boat must not be created, and 400 status code must be returned.

		You don’t need to validate the values of the attributes and can assume that if the request contains any of the listed attributes, then that attribute’s value is valid. You can also assume that the request will not contain any extraneous attribute (i.e., the request JSON will never contain any attribute that is not listed above).
	415 Unsupported Media Type	If the content-type header is not set to application/json, a 415 status code will be returned
	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned
	404 Not Found	No boat with this boat_id exists

Response Examples

- The id property will remain the same after the PUT request.

Success

Status: 303 See Other

```
{
    "id": "5636645067948032",
    "volume": 5,
    "carrier": null,
    "item": "LEGO Blocks",
    "creation_date": "10/18/2021",
    "self": "http://localhost:8080/loads/5636645067948032"
}
```

Failure

Status: 400 Bad Request

```
{
"Error":  " The request object is missing at least one of the required attributes "
}
```

Status: 415 Unsupported Media Type

```
{
"Error":  "Server only accepts application/json data"
}
```

Status: 406 Not Acceptable

```
{
"Error":  "Server only accepts application/json data"
}
```

Status: 404 Not Found

```
{
"Error":  "No load with this load_id exists"
}
```

Patch a Load

Allows you to edit a load. The PATCH request has 3 optional body properties: volume, item, and creation_date. The PATCH request will only edit the properties in the body. Note that the assigned boat is unassigned after a successful edit. The load that has been edited will have “carrier” property set to *null*.

To assign a boat, go to “Assign a Load to a Boat”. This request will require a Bearer Token matching the owner of the boat.

PATCH /loads/:load_id

Request

Path Parameters

None

Request Body

Optional

Request Body Format

JSON

Request JSON Attributes

Name	**Description**	**Required?**
volume	The volume of the load.	No
item	The item of the load.	No
creation_date	The date of creation.	No

Request Body Example

```
{
  "volume": 5,
  "item": "LEGO Blocks"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	303 See Other	The response body will contain a JSON detailing the edited entity. Property “self” contains the link to the

		edited entity.
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the boat must not be created, and 400 status code must be returned. You don't need to validate the values of the attributes and can assume that if the request contains any of the listed attributes, then that attribute's value is valid. You can also assume that the request will not contain any extraneous attribute (i.e., the request JSON will never contain any attribute that is not listed above).
	415 Unsupported Media Type	If the content-type header is not set to application/json, a 415 status code will be returned
	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned
	404 Not Found	No boat with this boat_id exists

Response Examples

- The id property will remain the same after the PATCH request.

Success

```
Status: 303 See Other

{
    "id": "5636645067948032",
    "volume": 5,
    "carrier": null,
    "item": "LEGO Blocks",
    "creation_date": "10/18/2021",
    "self": "http://localhost:8080/loads/5636645067948032"
}
```

Failure

```
Status: 400 Bad Request

{
"Error":  " The request object is missing at least one of the required attributes "
}
```

```
Status: 415 Unsupported Media Type

{
"Error":  "Server only accepts application/json data"
}
```

```
Status: 406 Not Acceptable

{
"Error":  "Server only accepts application/json data"
}
```

```
Status: 404 Not Found

{
"Error":  "No load with this load_id exists"
}
```

User

- The following requests will return a 405 Method Not Allowed response from the server:

PUT /users
DELETE /users
POST /users

Get a User

Allows you to get an existing user.

GET /users/:uid

Request

Path Parameters

Name	**Description**
uid	UID of the user – note this is different from id

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	
Failure	404 Not Found	No user with this uid exists
	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned

Response Examples

Success

```
Status: 200 OK

{
	"uid": "auth0|64860d6c65d56b4968472c57",
	"nickname": "daniel",
	"self": "http://localhost:8080/users/auth0|64860d6c65d56b4968472c57"
}
```

Failure

```
Status: 404 Not Found
```

```
{
        "Error": "No user with this uid exists"
}
```

```
Status: 406 Not Acceptable

{
"Error":  "Server only accepts application/json data"
}
```

List all Users

List all the users.

No pagination. Property “count” indicates the number of users returned from the request.

```
GET /users
```

Request

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	
Failure	406 Not Acceptable	If application/json is not accepted as a response, a 406 status code will be returned

Response Examples

Success

```
Status: 200 OK

{
        "count": 3,
        "users": [
                {
                        "uid": "auth0|64860d6c65d56b4968472c57",
                        "nickname": "daniel",
                        "id": "5667525748588544"
                },
        {
            "uid": "auth0|64860d61d959151e3615102c",
            "nickname": "emily",
            "id": "5743670015819776"
        },
        {
            "uid": "auth0|64860608d959151e36150edf",
            "nickname": "emily",
            "id": "5816332004098048"
        }
    ]
}
```

Failure

```
Status: 406 Not Acceptable

{
"Error":  "Server only accepts application/json data"
}
```