Boat Freight API

Timothy Yoon

CS 493: Cloud Application Development

Spring 2022

Oregon State University

Last Update: June 16. 2:55 PM Pacific

Change log

Version	Change	Date
1.0	Initial version.	June 3, 2022
1.1	Removed sections on deployment.	June 16, 2022

Data Model

- The app stores three kinds of entities in Datastore: boats, loads, and users.
- The "Required" column in the tables below refers to whether a property must have a value in the database, and does not necessarily refer to whether a property is required to be specified in the request body when an entity is being created. In other words, a property can be required in the database without the user needing to specify a value for it during a POST request.
  - In the sections describing the endpoints, the "Required?" column refers to whether a property must be specified in the request body.

Boat

Property	Data Type	Required	Valid Values	Notes
Id	Integer	Yes	Automatically generated by Datastore	The ID of the boat
name	String	Yes	• Any of the ASCII characters with codes 32-126 (decimal) • 1-60 characters long	The name of the boat
type	String	Yes	• Any of the ASCII characters with codes 32-126 (decimal) • 1-60 characters long	The type of the boat. E.g., Sailboat, Catamaran, etc.
length	Integer	Yes	Between 1-6000 (inclusive)	The length of the boat in feet
owner	String	Yes	• Any of the ASCII characters with codes 32-126 (decimal) • 1-60 characters long • Unique identifier of an existing user • Based on the "sub" property from the JWT	The user who owns the boat
loads	Array	Yes	• Array of 0, 1, or more objects representing existing loads whose "carrier" property matches this boat • e.g., [{ "id": 123, "self": "http://localhost:8080/loads/123" }]	A list of loads that the boat is carrying

Load

Property	Data Type	Required	Valid Values	Notes
id	Integer	Yes	Automatically generated by Datastore	The ID of the load
volume	Integer	Yes	Between 1-6000 (inclusive)	The volume of the load
item	String	Yes	• Any of the ASCII characters with codes 32-126 (decimal) • 1-60 characters long	The item that the load consists of
creation_date	String	Yes	Must be in the general format	The date on which the

			"4/6/2021", "10/18/2021", etc.	load was created
carrier	Object	No	• Object representing the boat that is carrying this load • e.g., { "id": 30, "self": "http://localhost:8080/boats/30" }	The boat on which the load is being carried

User

Property	**Data Type**	**Required**	**Valid Values**	**Notes**
id	Integer	Yes	Automatically generated by Datastore	The ID of the user
user_id	String	Yes	• Any of the ASCII characters with codes 32-126 (decimal) • 1-60 characters long • Based on the "sub" property from the JWT	The unique ID of the user
boats	Array	Yes	• Array of 0, 1, or more objects representing boats whose "owner" property matches this user's user_id • e.g., [{ "id": 30, "self": "http://localhost:8080/boats/30" }]	A list of boats that the user owns

Relationship Between the Non-User Entities

- The non-user entities are the boat and the load.

- A boat can carry zero, one, or more loads, which are listed in the boat's "loads" property. A load starts off unassigned to any boat and can be assigned to only one boat at a time. If a load is on a boat, the load's "carrier" property will specify that boat.

- A load can be assigned to and removed from a boat using the endpoints of this API.

How the User Entity is Modeled

- The unique identifier for a user is the value of "sub" from the JWT associated with the user. When a new account is created via Auth0, a new user entity is created with the unique identifier in the "user_id" property of the user.

- If a request needs to supply a user identifier, the person making the request needs to provide a valid JWT as a Bearer Token in the Authorization header of the request.

- The application maps a supplied JWT to the identifier of a user as follows. First, the application passes the request into the function "verify_jwt", which verifies the JWT and returns the decoded JWT. Then the application extracts the value of the "sub" property from the decoded JWT and checks whether any user in the database has that identifier.

Relationship Between the User Entity and Non-User Entity (Boat)

- A boat is always owned by one user at any given time.
- A user can own zero, one, or more boats.

View all Users

View all the users currently registered in the app. The response is not paginated.

```
GET /users
```

Protected?

No, this endpoint does not require a JWT.

Request

Path Parameters

None

Request Header

Accept header must include "application/json".

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 200 OK

[
    {
        "boats": [
            {
                "id": 5765243099676672,
                "self": "http://localhost:8080/boats/5765243099676672"
            },
            {
                "id": 5671636300726272,
                "self": "http://localhost:8080/boats/5671636300726272"
            }
        ],
        "user_id": "auth0|6294e3d9dc4dea0068d78193"
    },
    {
        "boats": [
            {
                "id": 5738294193160192,
                "self": "http://localhost:8080/boats/5738294193160192"
            },
            {
                "id": 5653635052601344,
                "self": "http://localhost:8080/boats/5653635052601344"
            }
        ],
        "user_id": "auth0|6294f862e0b00b0069f8bc99"
    }
]
```

- The "user_id" property represents the unique identifier of a user.

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

Create a Boat

Allows you to create a new boat.

```
POST /boats
```

Protected?

Yes, this endpoint verifies whether the provided JWT is valid.

Side Effects

An object is appended to the "boats" array of the user entity associated with the provided JWT. An example of such an object is

```
{
  "id": 5765243099676672,
  "self": "http://localhost:8080/boats/5765243099676672"
}
```

Request

Path Parameters

None

Request Header

- Authorization header must be of type Bearer Token and must include a valid JWT.
- Accept header must include "application/json".

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

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

Request Body Example

```
{
    "name": "Ahoy",
    "type": "Airboat",
    "length": 30
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201 Created	
Failure	400 Bad Request	If the request is missing any of the required attributes, the boat will not be created and a 400 status code will be returned. The API assumes that if the request contains any of the listed attributes, then that attribute's value is valid. The API also assumes that the request will not contain any extraneous attribute (i.e., that the request JSON will never contain any attribute that is not listed above).
Failure	401 Unauthorized	If the request has a missing or invalid JWT, the request will be rejected with status code 401.
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

- Datastore automatically generates an ID and stores it with the entity being created. The API sends back this value in the response body as shown in the example.

Success

```
Status: 201 Created

{
    "id": 5704568633556992,
    "length": 30,
    "loads": [],
    "name": "Ahoy",
    "owner": "auth0|629009b19ddf7f006874639b",
    "self": "http://localhost:8080/boats/5704568633556992",
    "type": "Airboat"
}
```

Failure

```
Status: 400 Bad Request

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

Failure

```
Status: 401 Unauthorized

{
  "Error": "The request object has a missing or invalid JWT"
}
```

Failure

```
Status: 406 Not Acceptable
```

```
{
  "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

View a Boat

Allows you to view an existing boat.

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

Protected?

Yes, this endpoint verifies whether the provided JWT is valid and only returns the specified boat if it belongs to the user associated with the provided JWT.

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	401 Unauthorized	If the request has a missing or invalid JWT, the request will be rejected with status code 401.
Failure	403 Forbidden	If the boat specified in the URL belongs to another user (i.e., a user not associated with the provided JWT), the request will be rejected with status code 403.
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 200 OK

{
    "id": 5153611168350208,
    "length": 30,
    "loads": [],
    "name": "Ahoy",
    "owner": "auth0|629009b19ddf7f006874639b",
    "self": "http://localhost:8080/boats/5153611168350208",
    "type": "Airboat"
}
```

Failure

```
Status: 401 Unauthorized

{
    "Error": "The request object has a missing or invalid JWT"
}
```

Failure

```
Status: 403 Forbidden

{
    "Error": "The boat belongs to someone else"
}
```

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

View all Boats

- View all the boats owned by the user associated with the provided JWT
- This endpoint implements pagination and returns a maximum of 5 boats per page. Every page except the last will have a "next" link to get the next page of results.
- Every page includes the property "total_number_of_boats" which indicates how many total boats (owned by the user associated with the JWT) are in the collection.

GET /boats

Protected?

Yes, this endpoint verifies whether the provided JWT is valid and only returns boats owned by the user associated with the provided JWT.

Request

Path Parameters

- None in the initial request
- The limit (maximum number of results returned in one batch) and offset (number of results to skip before returning anything) parameters can be added to view different pages of results. If there is at least one more page available to view, the limit and offset will be automatically set as query parameters in the "next" link of the response body.
    - To view the next page, it is recommended to copy and paste the "next" link into the Postman URL field of the existing request instead of creating a new request.

Request Header

- Authorization header must be of type Bearer Token and must include a valid JWT.
- Accept header must include "application/json".

Request Body

None

Response

Response Body Format

- JSON
- If there is at least one more page available to view, the current response body will have a "next" link. If there are no more pages, the current response body will be the last page and will not have a "next" link.

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	
Failure	401 Unauthorized	If the request has a missing or invalid JWT, the request will be rejected with status code 401.
Failure	406 Not Acceptable	If the request's Accept header does not include

		"application/json", the request will be rejected with status code 406.

Response Examples

Success (page 1 of 3)

```
Status: 200 OK

{
    "boats": [
        {
            "id": 4806027115495424,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/4806027115495424",
            "type": "Airboat"
        },
        {
            "id": 5114509786087424,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/5114509786087424",
            "type": "Airboat"
        },
        {
            "id": 5143677177430016,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/5143677177430016",
            "type": "Airboat"
        },
        {
            "id": 5658646574792704,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/5658646574792704",
            "type": "Airboat"
        },
        {
            "id": 5677459739508736,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/5677459739508736",
            "type": "Airboat"
        }
    ],
    "next": "http://localhost:8080/boats?limit=5&offset=5",
    "total_number_of_boats": 15
}
```

Success (page 2 of 3)

```
Status: 200 OK

{
    "boats": [
        {
            "id": 5701666712059904,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/5701666712059904",
            "type": "Airboat"
        },
        {
            "id": 5706627130851328,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/5706627130851328",
            "type": "Airboat"
        },
        {
            "id": 5726966351134720,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/5726966351134720",
            "type": "Airboat"
        },
        {
            "id": 5745753746046976,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/5745753746046976",
            "type": "Airboat"
        },
        {
            "id": 5750704903815168,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/5750704903815168",
            "type": "Airboat"
        }
    ],
    "next": "http://localhost:8080/boats?limit=5&offset=10",
    "total_number_of_boats": 15
}
```

Success (page 3 of 3)

```
Status: 200 OK

{
    "boats": [
        {
            "id": 6199595021369344,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/6199595021369344",
            "type": "Airboat"
        },
        {
            "id": 6213401999048704,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/6213401999048704",
            "type": "Airboat"
        },
        {
            "id": 6240409692930048,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/6240409692930048",
            "type": "Airboat"
        },
        {
            "id": 6265843818168320,
            "length": 30,
            "loads": [],
            "name": "All Aboard",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/6265843818168320",
            "type": "Auxiliary Ship"
        },
        {
            "id": 6306619969241088,
            "length": 30,
            "loads": [],
            "name": "Ahoy",
            "owner": "auth0|6294e3d9dc4dea0068d78193",
            "self": "http://localhost:8080/boats/6306619969241088",
            "type": "Airboat"
        }
    ],
    "total_number_of_boats": 15
}
```

Failure

```
Status: 401 Unauthorized

{
  "Error": "The request object has a missing or invalid JWT"
}
```

Failure

```
Status: 406 Not Acceptable

{
  "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

Update a Boat

Allows you to update a boat via either the PATCH or PUT method. The boat's name, type, and length attributes can be updated.

PATCH /boats/:boat_id

Protected?

Yes, this PATCH endpoint verifies whether the provided JWT is valid and only updates the specified boat if it belongs to the user associated with the provided JWT.

Request

Path Parameters

Name	Description
boat_id	ID of the boat

Request Body

- Required
- At least one attribute must be specified. All three attributes (name, type, length) can be included.

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": "All Aboard",
    "type": "Auxiliary Ship"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	401 Unauthorized	If the request has a missing or invalid JWT, the request will be rejected with status code 401.
Failure	403 Forbidden	If the boat specified in the URL belongs to another user (i.e., a user not associated with the provided JWT), the

		request will be rejected with status code 403.
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 200 OK

{
    "id": 5075408403824640,
    "length": 100,
    "loads": [],
    "name": "All Aboard",
    "owner": "auth0|629009b19ddf7f006874639b",
    "self": "http://localhost:8080/boats/5075408403824640",
    "type": "Auxiliary Ship"
}
```

Failure

```
Status: 401 Unauthorized

{
    "Error": "The request object has a missing or invalid JWT"
}
```

Failure

```
Status: 403 Forbidden

{
    "Error": "The boat belongs to someone else"
}
```

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

PUT /boats/:boat_id

Protected?

Yes, this PUT endpoint verifies whether the provided JWT is valid and only updates the specified boat if it belongs to the user associated with the provided JWT.

Request

Path Parameters

Name	**Description**
boat_id	ID of the boat

Request Body

- Required
- All three attributes (name, type, length) must be included.

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": "Courageous",
    "type": "Cruise Ship",
    "length": 300
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	401 Unauthorized	If the request has a missing or invalid JWT, the request will be rejected with status code 401.
Failure	403 Forbidden	If the boat specified in the URL belongs to another user (i.e., a user not associated with the provided JWT), the request will be rejected with status code 403.
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 200 OK

{
    "id": 5117579211309056,
    "length": 300,
    "loads": [],
    "name": "Courageous",
    "owner": "auth0|62900f8c05373d006f1ac820",
    "self": "http://localhost:8080/boats/5117579211309056",
    "type": "Cruise Ship"
}
```

Failure

```
Status: 401 Unauthorized

{
    "Error": "The request object has a missing or invalid JWT"
}
```

Failure

```
Status: 403 Forbidden

{
    "Error": "The boat belongs to someone else"
}
```

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

Delete a Boat

Allows you to delete a boat. If the boat has any loads, they will be unloaded before the boat is deleted. In addition, the boat owner's "boats" array will be updated to no longer include the boat.

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

Protected?

Yes, this endpoint verifies whether the provided JWT is valid and only deletes the specified boat if it belongs to the user associated with the provided JWT.

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	401 Unauthorized	If the request has a missing or invalid JWT, the request will be rejected with status code 401.
Failure	403 Forbidden	If the boat specified in the URL belongs to another user (i.e., a user not associated with the provided JWT), the request will be rejected with status code 403.
Failure	404 Not Found	No boat with this boat_id exists
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 401 Unauthorized

{
    "Error": "The request object has a missing or invalid JWT"
}
```

Failure

```
Status: 403 Forbidden

{
    "Error": "The boat belongs to someone else"
}
```

Failure

```
Status: 404 Not Found

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

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

Create a Load

Allows you to create a new load. This load will begin unassigned to any boat.

```
POST /loads
```

Protected?

No, this endpoint does not require a JWT.

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 that the load consists of	Yes
creation_date	The date on which the load was created	Yes

Request Body Example

```
{
  "volume": 8,
  "item": "Wooden planks",
  "creation_date": "4/6/2021"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201 Created	
Failure	400 Bad Request	If the request is missing any of the required attributes, the load will not be created and a 400 status code will be returned. The API assumes that if the request contains any of the listed attributes, then that attribute's value is valid. The API also assumes that the request will not contain

		any extraneous attribute (i.e., that the request JSON will never contain any attribute that is not listed above).
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

- Datastore automatically generates an ID and stores it with the entity being created. The API sends back this value in the response body as shown in the example.
- The value of the "carrier" attribute is null because a newly created load begins unassigned to any boat.

Success

```
Status: 201 Created

{
    "carrier": null,
    "creation_date": "4/6/2021",
    "id": 5681556802764800,
    "item": "Wooden planks",
    "self": "http://localhost:8080/loads/5681556802764800",
    "volume": 8
}
```

Failure

```
Status: 400 Bad Request

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

Failure

```
Status: 406 Not Acceptable
{
  "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

View a Load

Allows you to view an existing load.

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

Protected?

No, a JWT is not required since the load entity is not related to the user entity.

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	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 200 OK

{
    "carrier": null,
    "creation_date": "4/6/2021",
    "id": 5723707007827968,
    "item": "Wooden planks",
    "self": "http://localhost:8080/loads/5723707007827968",
    "volume": 8
}
```

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

View all Loads

- View all the loads
- This endpoint implements pagination and returns a maximum of 5 loads per page. Every page except the last will have a "next" link to get the next page of results.
- Every page includes the property "total_number_of_loads" which indicates the total number of loads in the collection.

```
GET /loads
```

Protected?

No, a JWT is not required since the load entity is not related to the user entity.

Request

Path Parameters

- None in the initial request
- The limit (maximum number of results returned in one batch) and offset (number of results to skip before returning anything) parameters can be added to view different pages of results. If there is at least one more page available to view, the limit and offset will be automatically set as query parameters in the "next" link of the response body.
  - To view the next page, it is recommended to copy and paste the "next" link into the Postman URL field of the existing request instead of creating a new request.

Request Header

Accept header must include "application/json".

Request Body

None

Response

Response Body Format

- JSON
- If there is at least one more page available to view, the current response body will have a "next" link. If there are no more pages, the current response body will be the last page and will not have a "next" link.

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success (page 1 of 3)

```
Status: 200 OK

{
    "loads": [
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 4837131872632832,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/4837131872632832",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 4886651973140480,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/4886651973140480",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5097358505279488,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5097358505279488",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5118606849343488,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5118606849343488",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5136459887542272,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5136459887542272",
            "volume": 8
        }
    ],
    "next": "http://localhost:8080/loads?limit=5&offset=5",
    "total_number_of_loads": 15
}
```

Success (page 2 of 3)

```
Status: 200 OK

{
    "loads": [
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5168126949851136,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5168126949851136",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5660308458700800,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5660308458700800",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5667525748588544,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5667525748588544",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5681556802764800,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5681556802764800",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5699409840963584,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5699409840963584",
            "volume": 8
        }
    ],
    "next": "http://localhost:8080/loads?limit=5&offset=10",
    "total_number_of_loads": 15
}
```

Success (page 3 of 3)

```
Status: 200 OK

{
    "loads": [
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5723707007827968,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5723707007827968",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5731076903272448,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5731076903272448",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 5738709764800512,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/5738709764800512",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 6244506756186112,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/6244506756186112",
            "volume": 8
        },
        {
            "carrier": null,
            "creation_date": "4/6/2021",
            "id": 6294026856693760,
            "item": "Wooden planks",
            "self": "http://localhost:8080/loads/6294026856693760",
            "volume": 8
        }
    ],
    "total_number_of_loads": 15
}
```

Failure

```
Status: 406 Not Acceptable

{
  "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

Update a Load

Allows you to update a load via either the PATCH or PUT method. The load's volume, item, and creation_date attributes can be updated.

```
PATCH /loads/:load_id
```

Protected?

No, a JWT is not required since the load entity is not related to the user entity.

Request

Path Parameters

Name	Description
load_id	ID of the load

Request Body

- Required
- At least one attribute must be specified. All three attributes (volume, item, creation_date) can be included.

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
volume	The volume of the load	No
item	The item that the load consists of	No
creation_date	The date on which the load was created	No

Request Body Example

```
{
    "volume": 16,
    "item": "Iron bars"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 200 OK

{
    "carrier": null,
    "creation_date": "4/6/2021",
    "id": 5723707007827968,
    "item": "Iron bars",
    "self": "http://localhost:8080/loads/5723707007827968",
    "volume": 16
}
```

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

PUT /loads/:load_id

Protected?

No, a JWT is not required since the load entity is not related to the user entity.

Request

Path Parameters

Name	**Description**
load_id	ID of the load

Request Body

- Required
- All three attributes (volume, item, creation_date) must be included.

Request Body Format

JSON

Request JSON Attributes

Name	**Description**	**Required?**
volume	The volume of the load	Yes
item	The item that the load consists of	Yes
creation_date	The date on which the load was created	Yes

Request Body Example

```
{
    "volume": 32,
    "item": "Linen sheets",
    "creation_date": "5/6/2022"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 200 OK

{
    "carrier": null,
    "creation_date": "5/6/2022",
    "id": 5723707007827968,
    "item": "Linen sheets",
    "self": "http://localhost:8080/loads/5723707007827968",
    "volume": 32
}
```

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

Delete a Load

Allows you to delete a load. Deleting a load will update the boat that was carrying it.

```
DELETE /loads/:load _id
```

Protected?

No, this endpoint does not require a JWT.

Request

Path Parameters

Name	**Description**
load_id	ID of the load

Request Header

Accept header must include "application/json".

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
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found

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

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

Delete all Loads

This endpoint is not supported and will return a 405 status code.

```
DELETE /loads
```

Protected?

No, this endpoint does not require a JWT.

Request

Request Body

None

Response

No body

Response Body Format

No body (Content-Type of text/html)

Response Statuses

Outcome	Status Code	Notes
Failure	405 Method Not Allowed	The method is not supported by the API.

Response Examples

Failure

```
Status: 405 Method Not Allowed
```

Assign Load to Boat

Assign a load to a boat.

```
PUT /boats/:boat_id/loads/:load_id
```

Protected?

No, this endpoint does not require a JWT.

Request

Path Parameters

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

Request Header

Accept header must include "application/json".

Request Body

None

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

Response

No body

Response Body Format

No body

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	Succeeds only if a boat exists with this boat_id, and a load exists with this load_id.
Failure	403 Forbidden	The load cannot be assigned because it has already been assigned to another boat.
Failure	404 Not Found	The specified boat and/or load does not exist.
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 403 Forbidden

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

Failure

```
Status: 404 Not Found

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

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes
'application/json'"
}
```

Remove Load from Boat

Remove a load from a boat. It is possible to remove a load from a boat without deleting that load.

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

Protected?

No, this endpoint does not require a JWT.

Request

Path Parameters

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

Request Header

Accept header must include "application/json".

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 boat has this load.
Failure	404 Not Found	No boat with this boat_id exists, or no load with this load_id exists.
Failure	404 Not Found	The boat with this boat_id does not have the load with this load_id.
Failure	406 Not Acceptable	If the request's Accept header does not include "application/json", the request will be rejected with status code 406.

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found
{
  "Error": "The specified boat and/or load does not exist"
}
```

Failure

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

Failure

```
Status: 406 Not Acceptable

{
    "Error": "The request object does not have an Accept header that includes 
'application/json'"
}
```