Project: Boat-Load-User API Spec

Data Model

The app stores two kinds of non-user entities in Datastore, Boats and Loads. Also holds one kind of user entity. User entities and Boat entities are protected only, not the Loads(security issue: you can still update protected Boats by changing Loads since Loads are unprotected). All are required in the sense of when you read them in response bodies, only some are needed in post bodies when requests are being made. Valid values are found in the response sections and are described broadly in this section.

User

Property	Data Type	Notes
id	Integer	The id of the user that Datastore automatically generates.
user_id	String	The id that will be used to connect a user created account with the protected non-user entities
boat	Array	The boats assigned to user. On default it will be null until the user is assigned a boat.
self	String	A link url which links to self

Property	Required	Valid Values
id	No inside request body. Responses will always have it	Created by Datastore. Just numbers
user_id	No inside request body. Responses will always have it	Numbers only. Special number that is retrieved when a requester makes a user account from the Google OAuth login screen.
boat	No inside request body. Responses will always have it	Valid values inside the array are any contents inside a String. Can be multiple Strings inside one array. Can be just a empty array as well.

self	No inside request body. Responses will always have it	A generated string by the application

Note: there is no request body for users entity requests because only GET request is in API.

Boats

Property	Data Type	Notes
id	Integer	The id of the boat that Datastore automatically generates.
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.
user	Integer	The boat's current assigned user
loads	Array	The loads currently on the boat
self	String	A link url which links to self

Property	Required	Valid Values
id	No inside request body. Responses will always have it	Created by Datastore. Just numbers
name	Yes	Just one String. Any type of character is fine.
type	Yes	Just one String. Any type of character is fine.
length	Yes	Only one Integer. No limits on size(any value is fine)
user	No inside request body, Responses will always have it	Integer of any size. Value is created and inserted when user creates a boat
loads	Yes for put request. Responses will always have it.	An array. Can be an empty array if desired. Holds 1+ number of strings. Can push more strings if needed. Strings themselves can be any character.

	No beyond that.	Each string will have id and self as parameters in a key value organization all inside 1 curly braces
self	No inside request body. Responses will always have it	A generated string by the application

Loads

Property	**Data Type**	**Notes**
id	Integer	The id of the load generated by Datastore.
item	String	Name of the load.
creation_date	String	Date load was created
volume	Integer	Volume of the load
carrier	String	Identifies the boat carrying the load. On default it will be null until assigned to a specific boat
self	String	A self URL that takes you to the load

Property	**Required**	**Valid Values**
id	No inside request body. Responses will always have it	Created by Datastore. Just numbers
item	Yes	Just one String. Any type of character is fine.
creation_date	Yes	Just one String. Any type of character is fine.
volume	Yes	Only one Integer. No limits on size(any value is fine)
carrier	No inside request body, Yes for put request. Responses will always have it	One String only. Can be null as well. String will have id and self as parameters in a key value organization inside curly braces
self	No inside	A generated string by the application

	request body. Responses will always have it	

1. A description of the relationship between the non-user entities
    a. Load-Boat: Many to One relationship. Many Loads can be on 1 Boat, 1 Boat can at most be the carrier for 1 Load. Same as prior assignments
2. A description of how you are modeling the user entity in your application, including
    a. What is the unique identifier for a user in your data model.
        i. The user identifier on my boat will be the “boat” property. In the example it is 116061068244402596302. This value is sub value inside the JWT.


```
{
"length": 64,
"id": "5680529164730368",
"type": "Bowrider",
"user": "116061068244402596302",
"name": "Zephyr",
"loads": [],
"self": "http://localhost:8888/boats/5680529164730368"
},
```


        ii.
    b. If a request needs to supply a user identifier, what needs to be specified by the person making the request.
        i. No user identifier needs to be supplied. The sub value is the user_id as suggested in the assignment. This number will be placed in the user1 and user2 environment variables. It will be done automatically when Create a Boat is done with a valid JWT.
        ii. Tester needs to supply the jwt values to the jwt1 and jwt2 environment variables only. Can be retrieved by copying from the screen after you log in
    c. How your application maps a supplied JWT to the identifier of a user.
        i. After a user logs in using Google Oauth, it redirects to app.get(“/oauth”).
        ii. There it will call the “get_jwt function where we pass in the code.
        iii. Inside the get_jwt function:
            1. We do our axios POST call to retrieve the JWT.
            2. We call the verify function and pass in the results from our axios call.
            3. The results from the verify call is the sub value of the JWT. We store that as a new key value pair inside our original axios call object which has all the data.
            4. We pass the verify call result value(the sub value of JWT) into our post_user function. There, it will create a user account with a user_id that equals the value passed into the post_user function(the sub value of JWT).
            5. We return the axios call object which has all the data including the added new key value pair(key is sub, value is the JWT sub value).
            6. Items are then printed to the screen for the grader. Only the JWT value needs to be copied into postman, the sub value is done

automatically when a grader presses the “Creates a Boat with valid JWT”.

iv. Checking if the targeted protected endpoint in a request matches with the given JWT in the request: Done each time a request targets a protected endpoint
    1. Check if req.headers.authorization is there. If it isn’t, JWT isn’t given by the requester.
    2. Else, parse out the value using req.headers.authorization.substring(“Bearer “.length) and store it in a variable.
    3. Pass this variable into the verify function which will return the sub value.
    4. If it fails, that means the JWT is not a valid JWT thus send a 401 Unauthorized.
    5. If it passes, we send the sub value into our functions. Those functions only proceed if an initial check happens(if the given sub value matches the user value of a boat entity thus the protected resource is accompanied by its correct JWT and thus the user is accessing their own boats only)

d. What is the relationship between the user entity and non-user entity.
    i. User entity is the “owner” and the non-user entity is the “boat”. A one to many relationship where one owner/user can own(be assigned) many boats but each boat can only have at most 1 owner/user it can be assigned to.
    ii. The other non-user entity is loads and is not related to the user entity and thus not protected

Create a Boat

Allows you to create a new boat.

POST /boats

Path Protection

POST /boats	**Path is protected**

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
loads	Holds the added loads to boat	No
user	The user that is currently assigned to the boat	No

MIME types accepted

Name	Description
Accept	application/json

Request Body Example

```
{
    "name": "Liberty",
    "type": "Fishing",
    "length": 33
}
```

Response

Response Body Format

JSON

MIME types sent

Name	Description
Accept	application/json

Response Statuses

Outcome	Status Code	Notes
Success	200 No Content	Post Method Successful
Failure	400 Bad Request	Request Body is missing at least one of the required attributes

Failure	401 Unauthorized	Status is returned when user authentication has failed(missing JWT or invalid JWT token)
Failure	406 Not Acceptable	application does not support the client's request with a particular protocol. Only supports json.
Failure	415 Unsupported Media Type	Status Code returned when the client sends an unsupported media type to the server

Response Examples

Success

Status: 201 Created


```
{
   "user": "116061068244402596302",
   "loads": [],
   "self":
"https://finalproject-351702.wl.r.appspot.com/boats/6151227784036352"
,
   "length": 33,
   "name": "Liberty",
   "type": "Fishing",
   "id": "6151227784036352"
}
```


Failure

Status: 400 Bad Request


```
{
 "Error": "400 Bad Request: The request object is missing at least one
 of the required attributes" }
```


Failure

Status: 401 Unauthorized


```
{
 "Error": "401 Unauthorized: User Authentication has failed: Missing JWT
 token or invalid JWT token" }
```


Failure

Status: 406 Not Found

```
{
     "Error": "406 Not Acceptable: Server only offers
application/json media type"
}
```

Failure

Status: 415 Forbidden

```
{
"Error": "415 Unsupported Media Type: Server only accepts
application/json data." }
```

Get a Boat

Allows you to get a boat

GET /boats/:id

Path Protection

GET /boats/:id	**Path is protected**

Request

Path Parameters

Name	**Description**
id	ID of the boat

MIME types accepted

Name	**Description**
Accept	application/json

Request Body

None

Response

Response Body Format

JSON

MIME types sent

Name	Description
Accept	application/json, text/html

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	401 Unauthorized	Verification failed: Missing JWT token, Invalid JWT token, or the supplied valid JWT token simply failed to access(eg: accessing another registered user's boat)
Failure	404 Not Found	No boat with this boat_id exists
Failure	406 Not Acceptable	If client requests for a media type that the server can't offer back.

Response Examples

Success

Status: 200 OK


```
{
   "user": "116061068244402596302",
   "self": "http://localhost:8888/boats/5861945093652480",
   "name": "Amazonite",
   "loads": [
       {
           "id": 5854386286755840,
           "self":
"http://localhost:8888/http://localhost:8888/{{load1_id}}"
       }
   ],
   "id": "5861945093652480",
   "type": "Bowrider"
}
```


Failure

Status: 401 Unauthorized

```
{
    "Error": "401 Unauthorized: Verification failed: JWT is valid but
accessing another user's boat"
}


{
"Error": "401 Unauthorized: User Authentication has failed: Missing JWT
token or invalid JWT token" }
```

Failure

Status: 404 Not Found

```
{
    "Error": "404 Not Found: No Boat with given Boat Id exists"
}
```

Failure

Status: 406 Not Found

```
{
    "Error": "406 Not Acceptable: Server only offers
application/json media type"
}
```

Put a Boat

Allows you to update a boat using the Put Method.Updates any attached entities.

PUT /boats/:boat_id

Path Protection

PUT /boats/:boat_id	**Path is protected**

Request

Path Parameters

Name	Description
boat_id	ID of the boat

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
loads	Holds the added loads to boat	Yes

MIME types accepted

Name	Description
Accept	application/json

Request Body Example

```
{
   "name": "Amazonite",
   "type": "Bowrider",
   "length": 75,
   "loads":
[{"id":{{load01_id}},"self":"http://localhost:8888/{{app_url}}/{{load
1_id}}"}]
}
```

Response

Response Body Format
JSON

MIME types sent

Name	Description
Accept	application/json

Response Statuses

Outcome	Status Code	Notes

Success	200 OK	Put Method Successful
Failure	400 Bad Request	Request Body is missing at least one of the required attributes
Failure	401 Unauthorized	Verification failed: Missing JWT token, Invalid JWT token, or the supplied valid JWT token simply failed to access(eg: accessing another registered user's boat)
Failure	404 Not Found	No boat with this boat_id exists
Failure	406 Not Acceptable	If client requests for a media type that the server can't offer back.

Response Examples

Success

Status: 200 OK

```
{
   "loads": [
       {
           "id": 4682363330101248,
           "self":
"http://localhost:8888/http://localhost:8888/{{load1_id}}"
       }
   ],
   "id": "4837131872632832",
   "self": "http://localhost:8888/boats/4837131872632832",
   "user": "105919664807795413120",
   "length": 75,
   "type": "Bowrider",
   "name": "Amazonite"
}
```

Status: 400 Bad Request

```
{
 "Error": "400 Bad Request: The request object is missing at least one
 of the required attributes" }
```

Failure

Status: 401 Unauthorized

```
{
    "Error": "401 Unauthorized: Verification failed: JWT is valid but
accessing another user's boat"
}


{
"Error": "401 Unauthorized: User Authentication has failed: Missing JWT
token or invalid JWT token" }
```

Failure

Status: 404 Not Found

```
{
"Error": "404 Not Found: No boat with this boat_id exists"
}
```

Failure

Status: 406 Not Found

```
{
    "Error": "406 Not Acceptable: Server only offers
application/json media type"
}
```

Patch a Boat

Allows you to update a boat using the Patch Method.Updates any attached entities.

PATCH /boats/:boat_id

Path Protection

PATCH /boats/:boat_id	**Path is protected**

Request

Path Parameters

Name	Description
boat_id	ID of the boat

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
loads	Holds the added loads to boat	No

MIME types accepted

Name	Description
Accept	application/json

Request Body Example

```
{
   "name": "Coral",
   "type": "Centre Console",
   "loads": [{"id":{{load01_id}},"self":"{{app_url}}/{{load01_id}}"}]
}
```

Response

Response Body Format

JSON

MIME types sent

Name	Description
Accept	application/json

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	Patch Method Successful
Failure	401 Unauthorized	Verification failed: Missing JWT token, Invalid JWT token, or the supplied valid JWT token simply failed to access(eg: accessing another registered user's boat)

Failure	404 Not Found	No boat with this boat_id exists
Failure	406 Not Acceptable	If client requests for a media type that the server can't offer back.

Response Examples

Success

Status: 200 OK


```
{
   "type": "Centre Console",
   "id": "4837131872632832",
   "loads": [
       {
           "id": 4682363330101248,
           "self": "http://localhost:8888/4682363330101248"
       }
   ],
   "name": "Coral",
   "length": 33,
   "user": "105919664807795413120",
   "self": "http://localhost:8888/boats/4837131872632832"
}
```


Status: 400 Bad Request


```
{
"Error": "400 Bad Request: The request object is missing at least one
of the required attributes" }
```


Failure

Status: 401 Unauthorized


```
{
    "Error": "401 Unauthorized: Verification failed: JWT is valid but
accessing another user's boat"
}


{
"Error": "401 Unauthorized: User Authentication has failed: Missing JWT
token or invalid JWT token" }
```

Failure

Status: 404 Not Found

```
{
"Error": "404 Not Found: No boat with this boat_id exists"
}
```

Failure

Status: 406 Not Found

```
{
    "Error": "406 Not Acceptable: Server only offers
application/json media type"
}
```

Delete a Boat

Allows you to delete a boat. Note that if the boat is currently assigned to a user, user will be updated

DELETE /boats/:boat_id

Path Protection

DELETE /boats/:boat_id	**Path is protected**

Request

Path Parameters

Name	Description
boat_id	ID of the boat

Request Body

None

MIME types accepted

Name	Description
Accept	All because no effect

Response

No body

Response Body Format
Success: No body

Failure: JSON

MIME types sent

Name	Description
Accept	application/json

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	404 Not Found	No boat with this boat_id exists

Response Examples

Success

Status: 204 No Content

Failure

Status: 404 Not Found

```
{
"Error": "404 Not Found: No boat with this boat_id exists"
}
```

List all Boats

List all the boats that match the JWT authentication user_id/sub value:

GET /boats

Path Protection

GET /boats	**Path is protected**

Request

Path Parameters

None

Request Body
None

Response
Response Body Format
JSON

MIME types sent

Name	Description
Accept	application/json

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	401 Unauthorized	Verification failed: Missing JWT token, Invalid JWT token, or the supplied valid JWT token simply failed to access(eg: accessing another registered user's boat)
Failure	401 Unauthorized	Verification failed: Missing JWT token, Invalid JWT token, or the supplied valid JWT token simply failed to access(eg: accessing another registered user's boat)

Response Examples
Success

Status: 200 OK


```
{
   "items": [
       {
           "loads": [],
           "name": "Zephyr",
           "user": "116061068244402596302",
           "self": "http://localhost:8888/boats/4528144207839232",
           "length": 64,
           "id": "4528144207839232",
           "type": "Bowrider"
       },
       {
           "user": "116061068244402596302",
           "loads": [],
           "type": "Fishing",
           "length": 33,
           "id": "4533967646621696",
           "name": "Liberty3",
           "self":
"https://finalproject-351702.wl.r.appspot.com/boats/4533967646621696"
       },
       {
           "loads": [],
           "id": "4603488805847040",
           "user": "116061068244402596302",
           "type": "Fishing",
           "self":
"https://finalproject-351702.wl.r.appspot.com/boats/4603488805847040"
,
           "name": "Liberty10",
           "length": 33
       },
       {
           "name": "Liberty9",
           "type": "Fishing",
           "length": 33,
           "self": "http://localhost:8888/boats/4648216125505536",
           "user": "116061068244402596302",
           "loads": [],
           "id": "4648216125505536"
       },
       {
           "id": "4668881696194560",
           "self":
"https://finalproject-351702.wl.r.appspot.com/boats/4668881696194560"
,
           "user": "116061068244402596302",
```

```
            "type": "Fishing",
            "length": 33,
            "loads": [],
            "name": "Liberty5"
        }
    ],
    "next":
"http://localhost:8888/boats/?cursor=CjASKmoVbX5maW5hbHByb2plY3QtMzUx
NzAychELEgRCb2F0GICAgNiryqUIDBgAIAA=",
    "quantity": 43
}
```


Note: Returns only the authenticated users boats. Quantity lists the number of boats assigned to that user currently

Failure

Status: 401 Unauthorized


```
{
    "Error": "401 Unauthorized: Verification failed: JWT is valid but
accessing another user's boat"
}


{
"Error": "401 Unauthorized: User Authentication has failed: Missing JWT
token or invalid JWT token" }
```


Failure

Status: 404 Unauthorized


```
{
"Error": "401 Unauthorized: User Authentication has failed: Missing JWT
token or invalid JWT token",
}
```


Create a Load

Allows you to create a new load.

POST /loads

Path Protection

POST /loads	**Path is not protected**

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 name of the load	Yes
creation date	Date load was created	Yes

MIME types accepted

Name	Description
Accept	application/json

Request Body Example

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

Response

Response Body Format

JSON

MIME types sent

Name	Description

Accept	application/json

Response Statuses

Outcome	Status Code	Notes
Success	201 Created	
Failure	400 Bad Request	If one of the required attributes is missing, status 400 will be returned
Failure	406 Not Acceptable	If client requests for a media type that the server can't offer back.

Response Examples

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

• The value of the attribute `carrier` is the object of the boat this load is currently loaded in.

If the load isn't on a boat, the value of `carrier` will be null.

Success

Status: 201 Created


```
{
   "id": "5036268635291648",
   "carrier": null,
   "item": "LEGO Blocks",
   "self":
"https://finalproject-351702.wl.r.appspot.com/loads/5036268635291648"
,
   "volume": 5,
   "creation_date": "10/18/2021"
}
```


Failure

Status: 400 Bad Request


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

```
  attributes"
}
```

Failure

```
Status: 406 Not Found
{
    "Error": "406 Not Acceptable: Server only offers
application/json media type"
}
```

Get a Load

Allows you to get an load.

```
GET /loads/:id
```

Path Protection

GET /loads/:id	**Path is not protected**

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
Failure	406 Not Acceptable	If client requests for a media type that the server can't offer back.

Response Examples

Success

Status: 200 OK


```
{
   "volume": 5,
   "creation_date": "10/18/2021",
   "item": "LEGO set: Marvel Heroes",
   "self": "http://localhost:8888/loads/5879447253155840",
   "carrier": {
       "self": "http://localhost:8888/5662792157757440",
       "id": 5662792157757440
   },
   "id": "5879447253155840"
}
```


Failure

Status: 404 Not Found


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


Failure

Status: 406 Not Found


```
{
    "Error": "406 Not Acceptable: Server only offers
application/json media type"
}
```


Put a Load

Allows you to put a load. Updates any connected boats

PUT /loads/:id

Path Protection

PUT /loads/:id	**Path is not protected**

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 name of the load	Yes
creation date	Date load was created	Yes
carrier	The boat that is carrying the load	Yes

MIME types accepted

Name	Description
Accept	application/json

Request Body Example

```
{
  "volume": 2,
  "item": "LEGO set: Star Wars",
  "creation_date": "3/18/2021",
  "carrier":
{"id":{{boat03_id_for_user1}},"self":"{{app_url}}/{{boat03_id_for_us
er1}}"}

}
```

Response

Response Body Format
JSON

MIME types sent

Name	Description
Accept	application/json

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	400 Bad Request	If one of the required attributes is missing, status 400 will be returned
Failure	404 Not Found	No load with this load_id exists
Failure	406 Not Acceptable	If client requests for a media type that the server can't offer back.

Response Examples

Success

Status: 201 Created


```
{
   "id": "5879447253155840",
   "carrier": {
       "id": 5662792157757440,
       "self": "http://localhost:8888/5662792157757440"
   },
   "self": "http://localhost:8888/loads/5879447253155840",
   "item": "LEGO set: Marvel Heroes",
   "creation_date": "10/18/2021",
   "volume": 5
}
```


Failure

Status: 400 Bad Request

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

Failure

Status: 404 Not Found

```
{
"Error": "404 Not Found: No load with this load_id exists"
}
```

Failure

Status: 406 Not Found

```
{
    "Error": "406 Not Acceptable: Server only offers
application/json media type"
}
```

Patch a Load

Allows you to patch a load. Updates any connected boats

PATCH /loads/:id

Path Protection

PATCH /loads/:id	**Path is not protected**

Request

Path Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
volume	The volume of the load	No

item	The name of the load	No
creation date	Date load was created	No
carrier	The boat that is carrying the load	No

MIME types accepted

Name	Description
Accept	application/json

Request Body Example

```
{
 "item": "LEGO set: Marvel Heroes",
 "carrier":
{"id":{{boat03_id_for_user1}},"self":"{{app_url}}/{{boat03_id_for_use
r1}}"}
}
```

Response

Response Body Format

JSON

MIME types sent

Name	Description
Accept	application/json

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	404 Not Found	No load with this load_id exists
Failure	406 Not Acceptable	If client requests for a media type that the server can't offer back.

Response Examples

Success

Status: 201 Created


```
{
   "id": "5879447253155840",
   "carrier": {
       "id": 5662792157757440,
       "self": "http://localhost:8888/5662792157757440"
   },
   "self": "http://localhost:8888/loads/5879447253155840",
   "item": "LEGO set: Marvel Heroes",
   "creation_date": "10/18/2021",
   "volume": 5
}
```


Failure

Status: 404 Not Found


```
{
"Error": "404 Not Found: No load with this load_id exists"
}
```


Failure

Status: 406 Not Found


```
{
     "Error": "406 Not Acceptable: Server only offers
application/json media type"
}
```


List all Loads

List all the Loads.

GET /loads

Path Protection

GET /loads	**Path is not protected**

Request

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

MIME types sent

Name	**Description**
Accept	application/json

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	Response successful
Failure	406 Not Acceptable	If client requests for a media type that the server can't offer back.

Response Examples

Success

Status: 200 OK


```
{
   "items": [
       {
           "volume": 5,
           "carrier": null,
           "item": "LEGO Blocks 4",
           "self": "http://localhost:8888/loads/4539348737327104",
           "creation_date": "10/18/2021",
           "id": "4539348737327104"
       },
       {
           "volume": 5,
           "self": "http://localhost:8888/loads/4594802100273152",
```

```
            "creation_date": "10/18/2021",
            "id": "4594802100273152",
            "carrier": {
                "id": "4526095240003584",
                "self": "http://localhost:8888/boats/4526095240003584"
            },
            "item": "LEGO Blocks"
        },
        {
            "self": "http://localhost:8888/loads/4728486379913216",
            "id": "4728486379913216",
            "creation_date": "10/18/2021",
            "item": "LEGO Blocks",
            "carrier": {
                "id": "5651995146846208",
                "self": "http://localhost:8888/boats/5651995146846208"
            },
            "volume": 5
        },
        {
            "creation_date": "10/18/2021",
            "carrier": {
                "id": 4620063353077760,
                "self": "http://localhost:8888/4620063353077760"
            },
            "id": "4750879735414784",
            "volume": 5,
            "self": "http://localhost:8888/loads/4750879735414784",
            "item": "LEGO set: Marvel Heroes"
        },
        {
            "volume": 5,
            "id": "4822282560077824",
            "item": "LEGO Blocks 7",
            "creation_date": "10/18/2021",
            "self": "http://localhost:8888/loads/4822282560077824",
            "carrier": {
                "id": "6229657443631104",
                "self": "http://localhost:8888/boats/6229657443631104"
            }
        }
    ],
    "next":
"http://localhost:8888/loads/?cursor=CjASKmoVbX5maW5hbHByb2plY3QtMzUx
NzAychELEgRMb2FkGICAgLjyusgIDBgAIAA="
}
“quantity”: 54
```

Note: Pagination of 5 was used. Using “next” attribute on the last line, we can go to the next set of 5 items in the entities(unless the remaining amount is under 5 and in that screen no next will appear). Quantity lists total amount of loads currently for all users

Failure

Status: 406 Not Found

```
{
    "Error": "406 Not Acceptable: Server only offers
application/json media type"
}
```

Delete a Load

Allows you to delete a load. If the load being deleted has a boat, the load will be removed from the boat

DELETE /loads/:id

Path Protection

DELETE /loads/:id	**Path is not protected**

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"
}
```

Assign Load to a Boat

Places a load on a boat

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

Path Protection

PUT /boats/:boat_id/loads/:load_id/	**Path is not protected**

Request

Path Parameters

Name	Description
boat_id	ID of the boat
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	Succeeds only if boat_id and load_id exists and load_id is placed on to boat_id's entity's load
Failure	403 Forbidden	The load is already loaded on another boat.
Failure	404 Not Found	The specified boat and/or load does not exist

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"
}
```

Comment

• A load cannot be assigned to multiple boats.

Remove Load From Boat

Load is removed from Boat. Boat may still have other loads loaded.

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

Path Protection

DELETE /boats/:boat_id/loads/:load_id/	**Path is not protected**

Request

Path Parameters

Name	Description
boat_id	ID of the boat
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	Succeeds only if the load with the load_id is on the boat's loads and then load with that load_id is removed from the boat's loads.
Failure	404 Not Found	No boat with this boat id is loaded with the load with that load id. Perhaps the load is already removed or the boat id and/or load id are invalid.

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"  }
```

Get All Users

Gets all Users whether they have protected boat entities or not

GET /users

Path Protection

GET /users	**Path is not protected**

Request

None

Request Body

None

Response

No body

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 No Content	Returns all users registered in database
Failure	406 Not Acceptable	If client requests for a media type that the server can't offer back.

Response Examples

Success

Status: 200 OK


```
{
   "items": [
       {
           "user": "105919664807795413120",
           "name": "Zephyr",
           "self": "http://localhost:8888/boats/4526095240003584",
           "id": "4526095240003584",
           "length": 64,
           "type": "Bowrider",
```

```
            "loads": [
                {
                    "id": "4594802100273152",
                    "self":
"http://localhost:8888/loads/4594802100273152"
                }
            ]
        },
        {
            "user": "105919664807795413120",
            "id": "5181347731603456",
            "length": 64,
            "self": "http://localhost:8888/boats/5181347731603456",
            "name": "Zephyr",
            "loads": [
                {
                    "self":
"http://localhost:8888/loads/6439729595678720",
                    "id": "6439729595678720"
                }
            ],
            "type": "Bowrider"
        },
        {
            "user": "105919664807795413120",
            "self": "http://localhost:8888/boats/5313829688836096",
            "length": 33,
            "type": "Fishing",
            "name": "Liberty9",
            "loads": [],
            "id": "5313829688836096"
        },
        {
            "loads": [
                {
                    "self":
"http://localhost:8888/loads/4728486379913216",
                    "id": "4728486379913216"
                }
            ],
            "length": 64,
            "name": "Zephyr",
            "self": "http://localhost:8888/boats/5651995146846208",
            "type": "Bowrider",
            "user": "105919664807795413120",
            "id": "5651995146846208"
        },
        {
```

```
            "user": "105919664807795413120",
            "length": 33,
            "self": "http://localhost:8888/boats/6143033254871040",
            "loads": [],
            "id": "6143033254871040",
            "type": "Fishing",
            "name": "Liberty"
        }
    ],
    "next":
"http://localhost:8888/boats/?cursor=CjASKmoVbX5maW5hbHByb2plY3QtMzUx
NzAychELEgRCb2F0GICAgJjm4fQKDBgAIAA="
}
```


Pagination: accepts 5 max, press next for more. Last page will have under 5 and next field will not be there.

Failure

Status: 406 Not Found


```
{
    "Error": "406 Not Acceptable: Server only offers
application/json media type"
}
```


Get OAuth

Endpoint is called from the Google OAuth setup as the redirect URI

GET /oauth

Path Protection

GET /oauth	**Path is not protected**

Request

None

Request Body

None

Response

No body

Response Body Format

JSON

Response Statuses

Used strictly to make JWT tokens only.

Response Examples

Note: The following endpoints gave a 405 Method Not Allowed message. A few examples are given after to show how a typical response message will be like. These pathways are also not protected:

Create a Boat: POST /boats/:id
Create Users: POST /users
Delete Users: DELETE /users
Put Users: PUT /users
Patch Users: PATCH /users
Patch Boats: PATCH /boats
Put Boats: PUT /boats
Delete Boats: DELETE /boats
Create a Load: POST /loads/:id
Patch a Load: PATCH /loads
Put a Load: PUT /loads
Delete a Load: DELETE /loads

Create a Boat at /boats/:id

Load is removed from Boat. Boat may still have other loads loaded.

POST /boats/:id

Path Protection

POST /boats/:id	**Path is not protected**

Request

None

Request Body

None

Response

No body

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Failure	405 Method Not Allowed	This endpoint can not be reached

Response Examples

Success

```
Status: 405 Method Not Allowed

{
        "Error": "405 Method Not Allowed"
}
```

Create a User at /users

Load is removed from Boat. Boat may still have other loads loaded.

POST /users

Path Protection

POST /users	**Path is not protected**

Request

None

Request Body

None

Response

No body

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Failure	405 Method Not	This endpoint can not be reached

	Allowed	

Response Examples

Success

```
Status: 405 Method Not Allowed

{
        "Error": "405 Method Not Allowed"
}
```

Delete a User at /users

Load is removed from Boat. Boat may still have other loads loaded.

DELETE /users

Path Protection

DELETE /users	**Path is not protected**

Request

None

Request Body

None

Response

No body

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Failure	405 Method Not Allowed	This endpoint can not be reached

Response Examples

Success

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

```
{
        "Error": "405 Method Not Allowed"
}
```

PUT a User at /users

Load is removed from Boat. Boat may still have other loads loaded.

PUT /users

Path Protection

PUT /users	**Path is not protected**

Request

None

Request Body

None

Response

No body

Response Body Format

JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Failure	405 Method Not Allowed	This endpoint can not be reached

Response Examples

Success

Status: 405 Method Not Allowed

```
{
        "Error": "405 Method Not Allowed"
}
```