Portfolio Project: RESTful API

CS 493: Cloud Application Development

Fall 2020

Oregon State University

Last Update: Dec 07, 2020. 6:00 pm Pacific

Data Model Section

User Entities and their properties:

1. Upon use of this API, users will be prompted to login to the application. Upon success, they are added to the database where the 'sub' of the JWT is stored as their unique identifier along with email and their name. The unique identifier the user users of this app is their email and the unique identifier provided by google.
    a. After being added to the database the users can then do requests that require authorization with the API. For a user to do this, they need a valid JWT and then will be authorized/authenticated and able to make protected requests to the API.
    b. When you log-in you will be given a JWT, put that into the environment on postman in jwt1. A future log-in will give another JWT put that in jwt2. For placeholders, I have two accounts already put in jwt1 and jwt2 for testing.
    c. Note that certain tests require the user to be logged into another account, when you log into that account previous tests will fail since you will no longer have access to them. Ex) you are jwt1 but a test requires jwt2 bearer token.
2. Users can have many, 0, or 1 boats. Using JWT, a user can view all boats they currently have using a GET request to /boats; otherwise, sending a request without JWT will show a list of boats for everyone.
3. This application utilizes Auth0 to check the JWT to the identifier of a user. The user then takes this token and puts it in the postman collection for testing.

Non-User Entities and their properties:

1. Boats
    a. A boat can only have one owner under this API who is a user added to the database. 0, 1, or many boats per 1 owner. Many to one relationship
    b. A boat can have 0, 1, or many loads added to it. 0 to many relationship.
    c. Only an owner of a boat can add or remove loads to a specific boat
    d. When an owner adds a load to a boat, the load's datastore id is added to a 'loads' array as a foreign key.
    e. When an owner removes a load from a boat, or deletes a load, the boat's 'loads' array is updated and the loads id is removed.
2. Loads
    a. Loads can only be assigned to a single boat. Many to 1 relationship, a boat can have multiple loads.
    b. Initially, when a load is first created it's property on what boat it is assigned to is set to null.
    c. When an owner adds a load to their boat the property is updated with the id of the boat to which it was added.
    d. When an owner deletes a boat or a load is removed from a boat, the property that shows what boat the load is assigned to is then set to null.

Create a Boat

Allows you to create a new boat.

```
POST /boats
```

Request

Request Parameters

Valid JWT where the User is in the database

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
name	String	The name of the boat.	Yes
type	String	The type of the boat. E.g., Sailboat, Catamaran, etc.	Yes
length	Integer	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	

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 other than the ones that are listed).
Failure	401 Unauthorized	A valid JWT was not provided or either a JWT is valid but a user is not in the database, a 401 status code must be returned
Failure	406 Not Acceptable	A 406 status code must be returned if the MIMEtype is not allowed as either the application or json is not listed in the accept header.

Response Examples

- Datastore will automatically generate an ID and store it with the entity being created. This value needs to be sent in the response body as shown in the example.
- The `self` attribute will contain the live link to the REST resource corresponding to this boat. In other words, this is the URL to get this newly created boat. You must not store the `self` attribute in Datastore

Success

```
Status: 201 Created

{
  "id": "abc123",
  "name": "Sea Witch",
  "type": "Catamaran",
  "owner": user_id,
  "loads": "1",
  "length": 28,
  "self": "https://<your-app>/boats/abc123"
}
```

Failure

```
Status: 400 Bad Request

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

Failure

```
Status: 401 Unauthorized
{
"Error":  "Either this user is not in the database or the JWT is missing”
}
```

Failure

```
Status: 406 Not Acceptable
{
"Error":  "Requested MIMEtype is not allowed by this method”
}
```

Get a Boat

Allows you to get an existing boat

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

Request

Request Parameters

Name	Type	Description	Required?
boat_id	String	ID of the boat	Yes

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
Failure	406 Not Acceptable	A 406 status code must be returned if the MIMEtype is not allowed as either the application or json is not listed in the accept header.

Response Examples

Success

```
Status: 200 OK

{
  "id": "abc123",
  "name": "Sea Witch",
  "type": "Catamaran",
  "length": 28,
  "self": "https://<your-app>/boats/abc123"
}
```

Failure

```
Status: 404 Not Found

{
```

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

Failure

```
Status: 406 Not Acceptable
{
"Error":  "Requested MIMEtype is not allowed by this method”
}
```

List all Boats

List all the boats.

```
GET /boats
```

Request

Request Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	406 Not Acceptable	A 406 status code must be returned if the MIMEtype is not allowed as either the application or json is not listed in the accept header.

Response Examples

Success

```
Status: 200 OK

[
{
  "id": "abc123",
  "name": "Sea Witch",
  "type": "Catamaran",
  "length": 28,
  "self": "https://<your-app>/boats/abc123"
},
{
  "id": "def456",
  "name": "Adventure",
  "type": "Sailboat",
  "length": 50,
  "self": "https://<your-app>/boats/def456"
},
{
  "id": "xyz123",
  "name": "Hocus Pocus",
  "type": "Sailboat",
  "length": 100,
  "self": "https://<your-app>/boats/xyz123"
}
]
```

Failure

```
Status: 406 Not Acceptable


{
"Error":  "Requested MIMEtype is not allowed by this method”
}
```

List all Boats for Specific User

List all the boats.

```
GET /boats
```

Request

Request Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	

Response Examples

Success

```
Status: 200 OK

[
{
  "id": "abc123",
  "name": "Sea Witch",
  "type": "Catamaran",
  "length": 28,
  "self": "https://<your-app>/boats/abc123"
},
{
  "id": "def456",
  "name": "Adventure",
  "type": "Sailboat",
  "length": 50,
  "self": "https://<your-app>/boats/def456"
},
{
  "id": "xyz123",
  "name": "Hocus Pocus",
  "type": "Sailboat",
  "length": 100,
  "self": "https://<your-app>/boats/xyz123"
}
]
```

Edit a Boat

Allows you to edit a boat.

```
PATCH /boats/:boat_id
```

Request

Request Parameters

Name	Type	Description	Required?
boat_id	String	ID of the boat	Yes

Required

Valid JWT, user is in database and owner of boat.

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

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

Request Body Example

```
{
  "name": "Sea Witch",
  "type": "Catamaran",
  "length": 99
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	

Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the boat must not be updated, 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 other than the ones that are listed).
Failure	401 Unauthorized	A valid JWT was not provided or either a JWT is valid but a user is not in the database, a 401 status code must be returned
Failure	403 Forbidden	User is not Owner of boat
Failure	404 Not Found	No boat with this boat_id exists
Failure	406 Not Acceptable	A 406 status code must be returned if the MIMEtype is not allowed as either the application or json is not listed in the accept header.

Response Examples

- The `self` attribute will contain the live link to the REST resource corresponding to this boat. You must not store this attribute in Datastore.

Success

```
Status: 200 OK
{
  "id": "abc123",
  "name": "Sea Witch",
  "type": "Catamaran",
  "length": 99,
  "self": "https://<your-app>/boats/abc123"
}
```

Failure

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

Failure

```
Status: 401 Unauthorized
{
"Error":  "Either this user is not in the database or the JWT is missing”
}
```

Failure

```
Status: 403 Forbidden
{
"Error":  "User is not Owner of boat"
}
```

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

Failure

```
Status: 406 Not Acceptable
{
"Error":  "Requested MIMEtype is not allowed by this method”
}
```

Delete a Boat

Allows you to delete a boat.

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

Request

Request Parameters

Valid JWT with user in database

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	A valid JWT was not provided or either a JWT is valid but a user is not in the database, a 401 status code must be returned
Failure	403 Forbidden	User is not Owner of boat
Failure	404 Not Found	No boat with this boat_id exists

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 401 Unauthorized

{
"Error":  "Either this user is not in the database or the JWT is missing"
}
```

Failure

```
Status: 403 Forbidden

{
"Error":  "User is not Owner of boat"
}
```

Failure

```
Status: 404 Not Found

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

Create a Load

Allows you to create a new load.

```
POST /loads
```

Request

Request Parameters

Valid JWT with user in database

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
Weight	Integer	Weight of the load	Yes
Content	String	What the load consists of	Yes
Destination	String	Destination of load	Yes

Request Body Example

```
{
  "weight": 10,
  "content": “Toilets”,
  "destination": “Portland”
}
```

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).
Failure	401 Unauthorized	A valid JWT was not provided or either a JWT is valid but a user is not in the database, a 401 status code must be returned
Failure	406 Not Acceptable	A 406 status code must be returned if the MIMEtype is not allowed as either the application or json is not listed in the accept header.

Response Examples

- Datastore will automatically generate an ID and store it with the entity being created. This value needs to be sent in the response body as shown in the example.
- The value of the attribute `carrier` is the ID of the boat currently at this load. If there is no boat at with this load, the value of `carrier` should be null
- The value of the attribute `self` is a live link to the REST resource corresponding to this load. In other words, this is the URL to get this newly created load. You must not store this attribute in Datastore

Success

```
Status: 201 Created

{
  "id": "123abc",
  "weight": 10,
  "content": Toilets,
  "carrier": null,
  "destination": Portland,
  "self": "https://<your-app>/boats/123abc"
}
```

Failure

```
Status: 400 Bad Request

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

Failure

```
Status: 401 Unauthorized
{
"Error":  "Either this user is not in the database or the JWT is missing”
}
```

Failure

```
Status: 406 Not Acceptable
{
"Error":  "Requested MIMEtype is not allowed by this method”
}
```

Get a Load

List all Loads

List all the loads.

```
GET /loads
```

Request

Request Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	406 Not Acceptable	A 406 status code must be returned if the MIMEtype is not allowed as either the application or json is not listed in the accept header.

Response Examples

Success

```
Status: 200 OK

[
{
  "id": "123abc",
  "weight": 10,
  "content": Toilets,
  "carrier": null,
  "destination": Portland,
  "self": "https://<your-app>/boats/123abc"
},
{
  "id": "456def",
  "weight": 20,
  "content": Paper,
  "carrier": null,
  "destination": Seattle,
  "self": "https://<your-app>/boats/123abc"
}
]
```

Failure

```
Status: 406 Not Acceptable
{
"Error":  "Requested MIMEtype is not allowed by this method”
}
```

Edit a Load

Allows editing a load

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

Request

Request Parameters

Valid JWT with user being in database

Request Body

Required

Response

No body

Response Body Format

Success: No body

Failure: JSON

Request JSON Attributes

Name	Type	Description	Required?
Weight	Integer	Weight of the load	Yes
Content	String	What the load contains	Yes
Destination	String	Where the load is destined to go	Yes

Request Body Example

```
{
  “weight”: “10”,
  “content”: “Toilets”,
  “destination”: “Portland”
}
```

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the load must not be updated, and 400 status code must be returned.
Failure	401 Unauthorized	A valid JWT was not provided or either a JWT is valid but a user is not in the database, a 401 status code must be returned

Failure	404 Not Found	No boat with this boat_id exists, and/or no load with this load_id exits.
Failure	406 Not Acceptable	A 406 status code must be returned if the MIMEtype is not allowed as either the application or json is not listed in the accept header.

Response Examples

Success

```
Status: 200 OK
{
  "weight": "10",
  "content": "Toilets",
  "destination": "Portland"
}
```

Failure

```
Status: 400 Bad Request

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

Failure

```
Status: 401 Unauthorized
{
"Error":  "Either this user is not in the database or the JWT is missing”
}
```

Failure

```
Status: 404 Not Found

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

Failure

```
Status: 406 Not Acceptable
{
  "Error":  "Requested MIMEtype is not allowed by this method”
}
```

Load assigned to boat

Load is assigned and added to the cargo of a boat

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

Request

Request Parameters

Valid JWT with user being in database and user owning 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 the owner is the owner of this boat
Failure	401 Unauthorized	A valid JWT was not provided or either a JWT is valid but a user is not in the database, a 401 status code must be returned
Failure	403 Forbidden	User is not Owner of boat to be assigned
Failure	404 Not Found	No boat with this boat_id or load with this load_id.

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 401 Unauthorized
{
"Error":  "Either this user is not in the database or the JWT is missing”
}
```

Failure

```
Status: 403 Forbidden
```

```
{
"Error":  "A load with that load_id is already assigned to another boat
}
```

Failure

Status: 404 Not Found

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

List all the Loads Assigned to Specific Boat

Allows you to list any and all loads on a specific boat

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

Request

Request Parameters

None

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	A valid JWT was not provided or either a JWT is valid but a user is not in the database, a 401 status code must be returned
Failure	404 Not Found	No boat with this boat_id exists

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 401 Unauthorized
{
"Error":  "Either this user is not in the database or the JWT is missing”
}
```

Failure

```
Status: 404 Not Found

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

Remove Load from Boat

Allows owner to remove a load from a boat, effectively deleting a load if the owner owns it.

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

Request

Request Parameters

Valid JWT with user being in database and owner being owner of a specific 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	A valid JWT was not provided or either a JWT is valid but a user is not in the database, a 401 status code must be returned
Failure	404 Not Found	No boat with this boat_id or no load with this load_id.

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 401 Unauthorized
{
"Error":  "Either this user is not in the database or the JWT is missing”
}
```

Failure

```
Status: 404 Not Found


{
```

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