Users, Boats, Loads: API Spec

https://hw9-leeda3.ue.r.appspot.com
CS 493: Cloud Application Development
Fall 2021
Oregon State University

User Creation

You will need a Google login to interact with this API, as it utilizes Google's oAuth 2.0 service.

1. Follow welcome page (https://hw9-leeda3.ue.r.appspot.com) and login
2. Upon login, you will be provided with a JWT. This is what you will use to make calls to protected endpoints (boats) which correspond with your unique ID presented.

Data Model

The app stores a User entity as well as two non-user entities in Datastore: Boats and Loads.

User (Unprotected)

Property	Data Type	Notes
First name	String	First name obtained through Google People API
Last name	String	Last name obtained through Google People API
Id	Integer	Unique ID obtained through Google People API - Obtained via 'sub' property from oauth2 token - Different than Datastore's auto-generated ID value
JWT	JWT	JWT obtained through Google People API Obtained via 'id_token' from request to oauth2

Boats (Protected)

Property	Data Type	Notes
id	Integer	The id of the boat. Automatically generated by Datastore.
name	String	Name of the boat. Required property
type	String	Type of the boat. E.g., Sailboat, Catamaran, etc. Required property
length	Integer	The length of the boat in feet. Required property
owner	Integer	Unique ID of User identity (see User's Data Model above)
loads	List (of Entity loads)	List of loads. Newly created boats start with no load If empty, is an empty list: [] If not empty, has objects (entities of loads) in the list. See "View a Boat" (pg. 5) for more details

Loads (Unprotected)

Property	Data Type	Notes
id	Integer	The id of the load. Automatically generated by Datastore.
content	String	The content of the load. Required property
volume	Integer	The volume of the load. Required property
Price	Int/Float	The price of the load. Required property
carrier	Integer	The id of the boat carrying the load and its URL If load is not on a boat, is null Starts off empty upon creation of a load

Entity Relationships, Authentication, and Making Requests

Relationships: Boat and Loads

Loads can exist as a separate entity from boats, seen from the Data Model section. Upon creation of a Load, the ‘carrier’ property is automatically set to null. You can create a relationship between a boat and a load, which would update the ‘carrier’ property and set it as the id of the corresponding boat. This can also be seen under the ‘loads’ property under a specific boat. Boats can have multiple loads, but a load can be only on one boat at a time. If you wish to remove the relationship between a boat and a load, it will set the ‘carrier’ property under the load to null and the load will be removed from the corresponding boat.

Relationships: Users and Boats

Boats are a protected entity and can only be accessed if the proper JWT is provided along with the request. Boats can only be created, viewed, updated, or deleted with the associated JWT. Upon creation of a boat, the ‘owner’ property is automatically set to the user’s unique identifier (ID).

Authentication: User Entity

Upon sign in, a random, 20-character state variable is generated and stored in the Datastore. This state variable gets sent to Google’s oAuth endpoint along with the program’s client ID. After the user consents to logging in via Google, an access code with the same state variable is sent. Once the state variable is verified and the client has verified this request, a POST request is sent back with the access code so Google can verify the client and pass on the JWT which will be used in this application. Note that the JWT expires, but a new one can be obtained by relogging into this website and using the newly presented JWT.

The unique identifier for a user is the ID property obtained via the Google People API. This API provides a unique ID which is stored in the Datastore and will be used to validate access to protected endpoints. The google oAuth2 library is used in order to map this unique ID (now stored on the Datastore) with the JWT (specifically the id_token module). As long as the ID in the datastore matches the ID received from the requests sent through this library, it will allow access to protected endpoints.

Making Requests

The different types of requests and endpoints possible with this API is presented in the table of contents. For unprotected endpoints, no authorization is needed. In order to access a protected endpoint, the request must be sent in the Header of the request with the ‘Key:Value’ pair as follows: ‘Authorization:Bearer {{Supplied JWT}}’. If using Postman, the JWT can be supplied under the ‘Auth’ section with the Type set to Bearer Token.

View all Boats

List all the boats. Protected Endpoint.

GET /boats

Request

Authorization Header

JWT required

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	Returns 5 boats per page, at a time, via pagination If there are more than 5 boats, there will be a “next” URL where you can view the next page of boats.
Failure	401 Unauthorized	Missing or Invalid JWT sent with request. See Error message for further details.
Failure	405 Method Not Allowed	Non-get/post request (default Flask error)

Response Examples

Success

Status: 200 OK

```
{
    "boats": [
        {
            "id": 5643550469390336,
            "name": "Sea Witch",
            "type": "Catamaran",
            "length": 28,
            "owner": "102064530495094753303",
            "loads": [
                {
                    "self": "https://hw9-leeda3.ue.r.appspot.com/
                             loads/5750704903815168",
                    "id": 5750704903815168
                }
            ],
            "self": "https://hw9-leeda3.ue.r.appspot.com/
                     boats/5643550469390336"
        },
        {
            "id": 5708765252812800,
            "name": "Sea Witch v2",
            "type": "Catamaran v2",
            "length": 38,
            "owner": "102064530495094753303",
            "loads": [],
            "self": "https://hw9-leeda3.ue.r.appspot.com/
                     boats/5708765252812800"
        }
    ],
    "Total boats": 2
}
```

Failure

```
Status: 401 Unauthorized

{
"Error":  "Unauthorized: Missing JWT"
}
```

Create a Boat

Allows you to create a new boat. Protected Endpoint.

```
POST /boats
```

Request

Authorization Header

JWT required

Path Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

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

Request Body Example

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

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201 Created	Returns JSON body of boat created (see Response Examples below)
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the boat will not be created
Failure	401 Unauthorized	Missing or Invalid JWT sent with request. See Error message for further details.
Failure	405 Method Not Allowed	Non-get/post request (default Flask error)
Failure	415 Unsupported Media Type	Content-type must be sent with 'application/json' allowed

Response Examples

Success

```
Status: 201 Created

{
    "id": 5643550469390336,
    "name": "Sea Witch",
    "type": "Catamaran",
    "length": 28,
    "owner": "102064530495094753303",
    "loads": [],
    "self": "https://hw9-leeda3.ue.r.appspot.com/boats/5643550469390336"
}
```

Failure

Status: 400 Bad Request

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

Status: 401 Unauthorized

```
{
"Error":  "Unauthorized: Invalid JWT"
}
```

Status: 415 Unsupported Media Type

```
{
"Error":  "Unsupported Media Type"
}
```

View a Boat

Allows you to get an existing boat. Protected Endpoint.

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

Request

Authorization Header

JWT required

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	Returns JSON response of boat and its attributes created
Failure	401 Unauthorized	Invalid or missing JWT
Failure	403 Forbidden	Boat owned by someone else
Failure	404 Not Found	No boat with this boat_id exists
Failure	406 Not acceptable	Client must accept ‘application/json’

Response Examples

Success

Status: 200 OK

```
{
    "id": 5643550469390336,
    "name": "Sea Witch",
    "type": "Catamaran",
    "length": 28,
    "owner": "102064530495094753303",
    "loads": [],
    "self": "https://hw9-leeda3.ue.r.appspot.com/boats/5643550469390336"
}
```

Failure

Status: 401 Unauthorized

```
{
"Error":  "Unauthorized: Invalid JWT"
}
```

Status: 403 Forbidden

```
{
"Error":  "Forbidden: Boat owned by someone else"
}
```

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

Status: 406 Not Acceptable
 	
{
"Error": "Client must accept application/json"
}

Edit a Boat (PATCH)

Allows you to edit subset of an existing boat's attributes. Protected Endpoint.

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

Request

Authorization Header

JWT required

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	Returns JSON response of boat and its attributes created Note: Not all attributes are required for a PATCH request
Failure	400 Bad Request	Updating an invalid attribute
Failure	401 Unauthorized	Invalid or missing JWT
Failure	403 Forbidden	Boat owned by someone else
Failure	404 Not Found	No boat with this boat_id exists
Failure	415 Unsupported Media Type	Must accept 'application/json'

Response Examples

Success

Status: 200 OK

```
{
    "id": 5643550469390336,
    "name": "Updated Sea Witch",
    "type": "Catamaran",
    "length": 28,
    "owner": "102064530495094753303",
    "loads": [],
    "self": "https://hw9-leeda3.ue.r.appspot.com/boats/5643550469390336"
}
```

Failure

Status: 400 Bad Request

```
{
"Error":  "Updating invalid attribute "
}
```

Status: 401 Unauthorized

{
"Error": "Unauthorized: Invalid JWT"
}

Status: 403 Forbidden
 	
{
"Error": "Forbidden: Boat owned by someone else"
}

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

Status: 415 Not Acceptable
 	
{
"Error": "Unsupported Media Type"
}

Edit a Boat (PUT)

Allows you to edit an existing boat. Protected Endpoint.

PUT /boats/:boat_id

Request

Authorization Header

JWT required

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	Returns JSON response of boat and its attributes created Note: All attributes are required for a PUT request
Failure	400 Bad Request	Updating an invalid attribute
Failure	401 Unauthorized	Invalid or missing JWT
Failure	403 Forbidden	Boat owned by someone else
Failure	404 Not Found	No boat with this boat_id exists
Failure	415 Unsupported Media Type	Must accept 'application/json'

Response Examples

Success

Status: 200 OK

```
{
    "id": 5643550469390336,
    "name": "Updated Sea Witch",
    "type": "Updated Catamaran",
    "length": 38,
    "owner": "102064530495094753303",
    "loads": [],
    "self": "https://hw9-leeda3.ue.r.appspot.com/boats/5643550469390336"
}
```

Failure

Status: 400 Bad Request

```
{
"Error":  "Updating invalid attribute "
}
```

Status: 401 Unauthorized

{
"Error": "Unauthorized: Invalid JWT"
}

Status: 403 Forbidden
 	
{
"Error": "Forbidden: Boat owned by someone else"
}

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

Status: 415 Not Acceptable
 	
{
"Error": "Unsupported Media Type"
}

Delete a Boat

Allows you to delete a boat. Protected Endpoint.

Note: if the boat is currently holding a load, deleting the boat unloads any loads.

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

Request

Authorization Header

JWT required

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

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	Deleting a boat will unload any loads that were loaded onto it
Failure	401: Unauthorized	Invalid or missing JWT
Failure	403: Forbidden	Boat owned by someone else
Failure	404 Not Found	No boat with this boat_id exists
Failure	405: Method not allowed	Non get/patch/put/delete method

Response Examples

Success

Status: 204 No Content

Failure

Status: 401 Unauthorized

```
{
"Error":  "Unauthorized: Invalid JWT"
}
```

Status: 403 Forbidden

```
{
"Error":  "Forbidden: Boat owned by someone else"
}
```

Status: 404 Not Found

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

View all Loads

List all the loads. Unprotected endpoint.

```
GET /loads
```

Request

Authorization Header

JWT not required

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	Returns 5 loads per page, at a time, via pagination If there are more than 5 loads, there will be a “next” URL

Response Examples

Success

Status: 200 OK

```
{
    "loads": [
        {
            "id": 5658646574792704,
            "content": "Desks",
            "volume": 50,
            "price": 250,
            "carrier": null,
            "self": "https://hw9-leeda3.ue.r.appspot.com/
                     loads/5658646574792704"
        },
        {
            "id": 5673742378205184,
            "content": "Couch",
            "volume": 250,
            "price": 1250,
            "carrier": null,
            "self": "https://hw9-leeda3.ue.r.appspot.com/
                     loads/5673742378205184"
        },
        {
            "id": 5688703292801024,
            "content": "Chairs",
            "volume": 25,
            "price": 150,
            "carrier": null,
            "self": "https://hw9-leeda3.ue.r.appspot.com/
                     loads/5688703292801024"
        },
        {
            "id": 5718125026738176,
            "content": "Cushion",
            "volume": 25,
            "price": 50,
            "carrier": null,
            "self": "https://hw9-leeda3.ue.r.appspot.com/
                     loads/5718125026738176"
        },
        {
            "id": 5723707007827968,
            "content": "Monitor",
            "volume": 15,
            "price": 200,
            "carrier": null,
            "self": "https://hw9-leeda3.ue.r.appspot.com/
                     loads/5723707007827968"
        }
    ],
    "Total loads": 6,
    "next": "https://hw9-leeda3.ue.r.appspot.com/loads?limit=5&offset=5"
}
```

Create a Load

Allows you to create a new load. Unprotected endpoint.

```
POST /loads
```

Request

Authorization Header

JWT not required

Path Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
volume	The volume of the load	Yes
content	The content of the load	Yes
price	The price of the load	Yes

Request Body Example

```
{
  "volume": 10,
  "content": “Stool”,
  "price": 15
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201 Created	Returns JSON response of new load Newly created loads will be unassigned to any boat
Failure	400 Bad Request	If the request is missing the required attributes, the load will not be created. Price must be of type integer or float Price must be a positive value

Response Examples

Success

```
Status: 201 Created

{
    "id": 5750704903815168,
    "content": "Stool",
    "volume": 10,
    "price": 15,
    "carrier": null,
    "self": "https://hw9-leeda3.ue.r.appspot.com/loads/5750704903815168"
}
```

Failure

```
Status: 400 Bad Request

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

View a Load

Allows you to get an existing load. Unprotected endpoint.

GET /loads/:load_id

Request

Authorization Header

JWT not required

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	Returns JSON response with load attributes
Failure	404 Not Found	No load with this load_id exists

Response Examples

Success

Status: 200 OK

```
{
    "id": 5750704903815168,
    "content": "Stool",
    "volume": 10,
    "price": 15,
    "carrier": {
        "id": 5643550469390336,
        "self": "https://hw9-leeda3.ue.r.appspot.com/boats/5643550469390336"
    },
    "self": "https://hw9-leeda3.ue.r.appspot.com/loads/5750704903815168"
}
```

Failure

Status: 404 Not Found

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

Edit a Load (PATCH)

Allows you to edit a subset of an existing load’s attributes. Unprotected endpoint.

PATCH /loads/:load_id

Request

Authorization Header

JWT not required

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	Returns JSON response with updated load attributes
Failure	400 Bad Request	Updating invalid attribute - Price must be type int or float - Price must be positive value
Failure	404 Not Found	No load with this load_id exists

Response Examples

Success

Status: 200 OK

```
{
    "id": 5750704903815168,
    "content": "Updated Stool",
    "volume": 12,
    "price": 20.00,
    "carrier": {
        "id": 5643550469390336,
        "self": "https://hw9-leeda3.ue.r.appspot.com/boats/5643550469390336"
    },
    "self": "https://hw9-leeda3.ue.r.appspot.com/loads/5750704903815168"
}
```

Failure

Status: 400 Not Found

```
{
"Error":  "Price must be of type int or float"
}
```

Status: 404 Not Found

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

Edit a Load (PUT)

Allows you to edit an existing load's attributes. Unprotected endpoint.

PUT /loads/:load_id

Request

Authorization Header

JWT not required

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	Returns JSON response with updated load attributes Note: All attributes required
Failure	400 Bad Request	All attributes not present or updating invalid attribute - Price must be of type int or float - Price must be positive value
Failure	404 Not Found	No load with this load_id exists

Response Examples

Success

Status: 200 OK

```
{
    "id": 5750704903815168,
    "content": "Updated Stool",
    "volume": 15,
    "price": 25.00,
    "carrier": {
        "id": 5643550469390336,
        "self": "https://hw9-leeda3.ue.r.appspot.com/boats/5643550469390336"
    },
    "self": "https://hw9-leeda3.ue.r.appspot.com/loads/5750704903815168"
}
```

Failure

Status: 400 Bad Request

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

```
Status: 404 Not Found

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

Delete a Load

Allows you to delete a load.

Note: If the load is being held by a boat, deleting a load will update the boat’s load

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

Request

Authorization Header

JWT not required

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	Deleting a load should update the boat carrying it
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"
}
```

Assigning a Load to a Boat

Load is assigned a carrier (boat).

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

Request

Path Parameters

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

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	Succeeds only if a boat exists with this boat_id, a load exists with this load_id, and the load is unassigned. Updates “carrier” in loads as well as “loads” in boats
Failure	403 Forbidden	Load is already assigned to 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":  "Forbidden: Load is already assigned"
}
```

```
Status: 404 Not Found

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

Comment

- A load cannot be assigned to another boat without first being removed.

Removing a Load from a Boat

Remove the load from current boat.

Note: Load still exists as an entity

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

Request

Path Parameters

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

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	Succeeds only if a boat exists with this boat_id, a load exists with this load_id and this load is on the boat.
Failure	404 Not Found	No load with this load_id is on the boat with this boat_id

Response Examples

Success

```
Status: 204 No Content
```

Failure

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

View all Users

List all the users. Unprotected endpoint.

```
GET /users
```

Request

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	Returns list of all current users - Example below shows two different Google accounts owned by David Lee. If same account is used to login multiple times, Datastore checks to see if entity already exists and will not create a new entry.

Response Examples

Success

```
Status: 200 OK

{
  "Showing 2 users": [
    {
      "user_id": “102064530495094753303”,
      "first_name": "David",
      "last_name": "Lee",
    },
    {
      "user_id": “103561445722470375530”,
      "first_name": "David",
      "last_name": "Lee",
    },
  ],
}
```