Final Project

CS 493: Cloud Application Development

Fall 2019

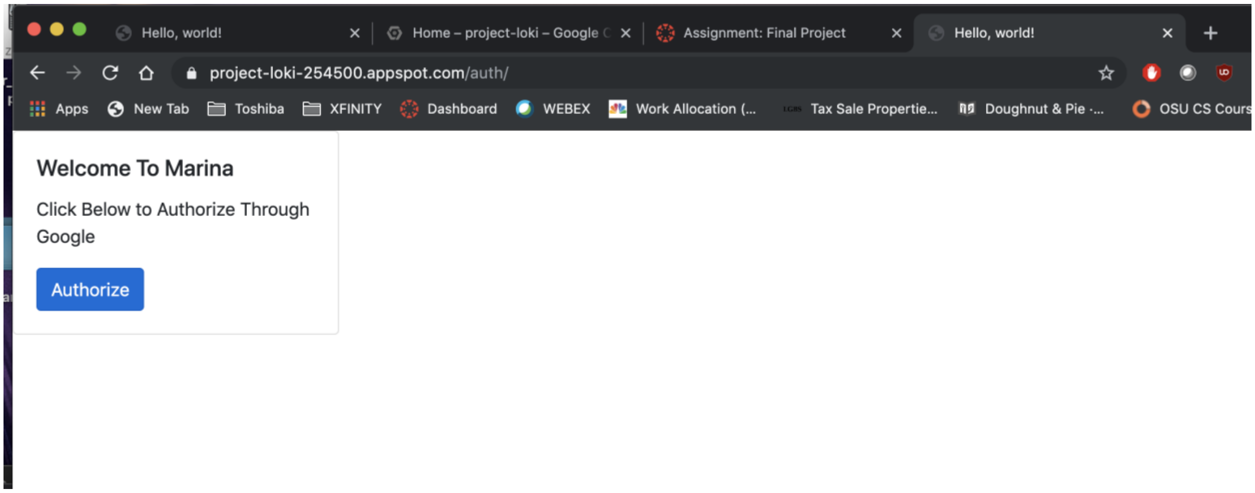
Oregon State University

Last Update: Nov 29, 2019. 9:00 pm Pacific

https://project-loki-254500.appspot.com

Getting Started

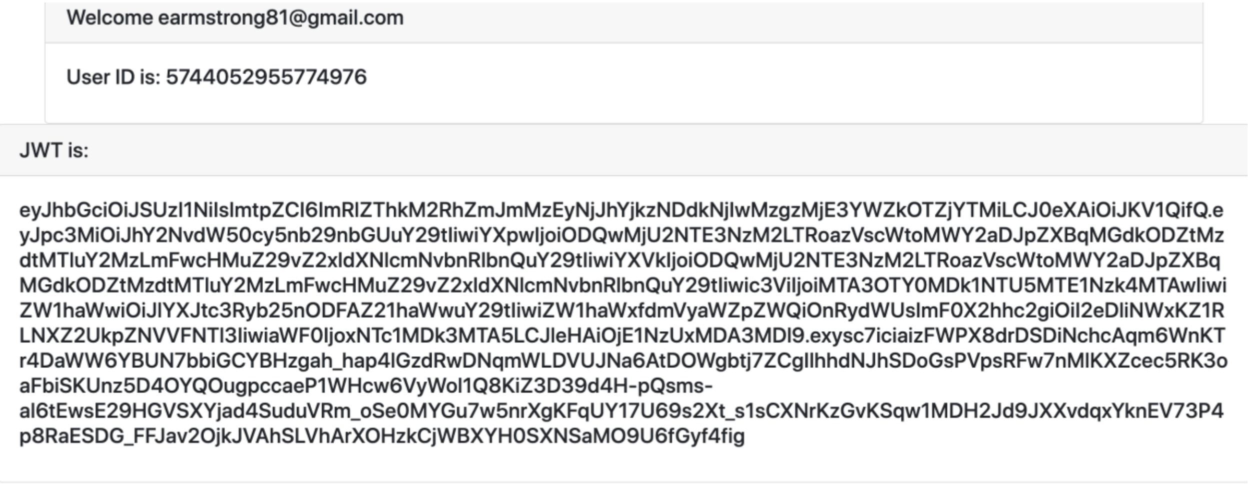
- Go to https://project-loki-254500.appspot.com and click ‘Authorize’
- Log in to google
- User Id and JWT will display
    - Use User Id for ***GET users/:user_id call***
    - Use JWT for Bearer token



Welcome To Marina

Click Below to Authorize Through Google

Authorize



Welcome earmstrong81@gmail.com

User ID is: 5744052955774976

JWT is:

```
eyJhbGciOiJSUzI1NiIsImtpZCI6ImRlZThkM2RhZmJmMzEyNjJhYjkzNDdkNjIwMzgzMjE3YWZkOTZjYTMiLCJ0eXAiOiJKV1QifQ.e
yJpc3MiOiJhY2NvdW50cy5nb29nbGUuY29tIiwiYXpwIjoiODQwMjU2NTE3NzM2LTRoazVscWtoMWY2aDJpZXBqMGdkODZtMz
dtMTluY2MzLmFwcHMuZ29vZ2xldXNlcmNvbnRlbnQuY29tIiwiYXVkIjoiODQwMjU2NTE3NzM2LTRoazVscWtoMWY2aDJpZXBq
MGdkODZtMzdtMTluY2MzLmFwcHMuZ29vZ2xldXNlcmNvbnRlbnQuY29tIiwic3ViIjoiMTA3OTY0MDk1NTU5MTE1Nzk4MTAwIiwi
ZW1haWwiOiJlYXJtc3Ryb25nODFAZ21haWwuY29tIiwiZW1haWxfdmVyaWZpZWQiOnRydWUsImF0X2hhc2giOiI2eDliNWxKZ1R
LNXZ2UkpZNVVFNTI3IiwiaWF0IjoxNTc1MDk3MTA5LCJleHAiOjE1NzUxMDA3MDl9.exysc7iciaizFWPX8drDSDiNchcAqm6WnKT
r4DaWW6YBUN7bbiGCYBHzgah_hap4lGzdRwDNqmWLDVUJNa6AtDOWgbtj7ZCgllhhdNJhSDoGsPVpsRFw7nMlKXZcec5RK3o
aFbiSKUnz5D4OYQOugpccaeP1WHcw6VyWol1Q8KiZ3D39d4H-pQsms-
al6tEwsE29HGVSXYjad4SuduVRm_oSe0MYGu7w5nrXgKFqUY17U69s2Xt_s1sCXNrKzGvKSqw1MDH2Jd9JXXvdqxYknEV73P4
p8RaESDG_FFJav2OjkJVAhSLVhArXOHzkCjWBXYH0SXNSaMO9U6fGyf4fig
```

https://project-loki-254500.appspot.com

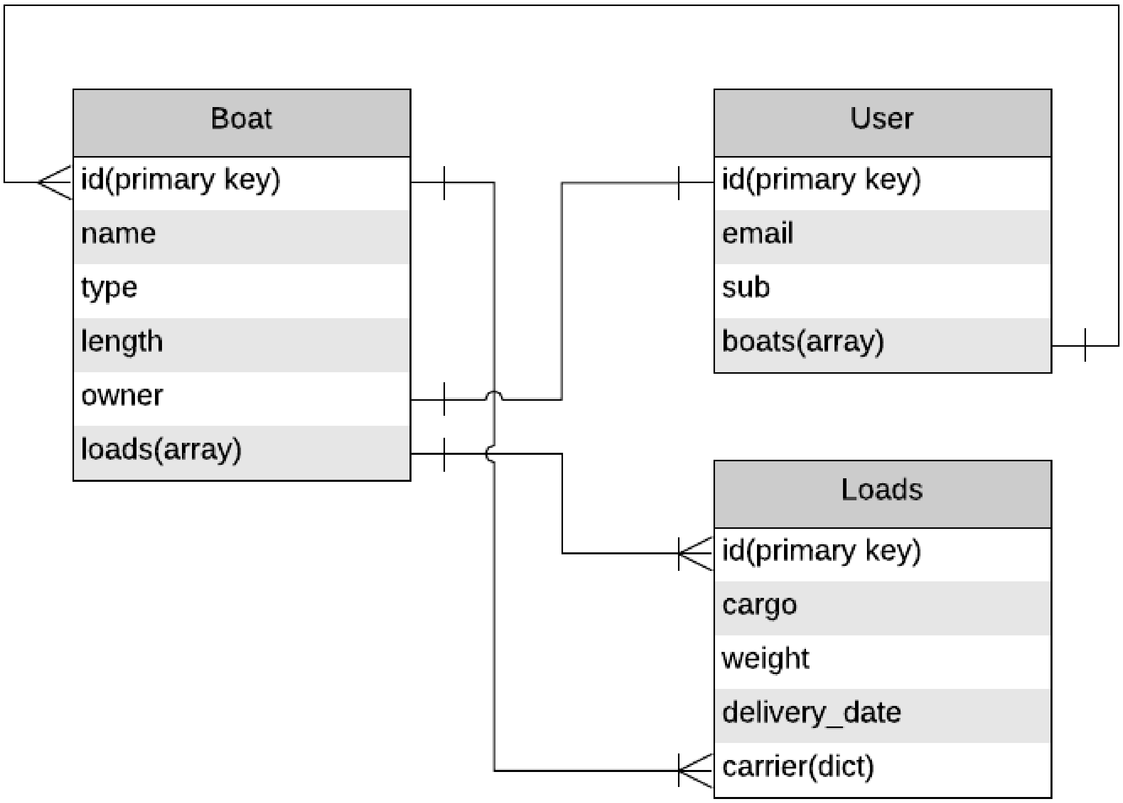


Figure 1 Database Structure

Create a Boat

Allows you to create a new boat.

POST /boats

Request

Request Headers

Content-Type: application/json

Authorization: Bearer {JWT}

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.
Failure	401 Unauthorized	Invalid JWT was used
Failure	406 Not Acceptable	Invalid format requested

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",
  "length": 28,
  "loads": [],
  "owner": 2883774838474
  "self": "https://<your-app>/boats/abc123"
}
```

Failure

Status: 400 Bad Request

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

Status: 401

{
"Error": "Invalid JWT"
}

Status: 406
{
"Error": "Content must be in JSON format"
}

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

Response Examples

Success

```
Status: 200 OK


  "id": "abc123",
  "name": "Sea Witch",
  "type": "Catamaran",
  "length": 28,
  "owner": 2838978709898,
  "loads": [{"id":"123abc",
             "self":"https://appspot.com/loads/123abc"},
            {"id":"456",
             "self":"https://appspot.com/loads/456"}
           ]
  "self": "https://<your-app>/boats/abc123"
}
```

Failure

```
Status: 404 Not Found

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

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	

Response Examples

Success

```
Status: 200 OK

[
{
  "id": "abc123",
  "name": "Sea Witch",
  "type": "Catamaran",
  “owner”: 32987497397323
  "length": 28,
  "loads": [{"id":"123abc",
             "self":"https://appspot.com/loads/123abc"},
            {"id":"456",
             "self":"https://appspot.com/loads/456"}
            ]

  "self": "https://<your-app>/boats/abc123"
},
{
  "id": "def456",
  "name": "Adventure",
  "type": "Sailboat",
  “owner”: 32987497397323
  "length": 50,
  "loads": [{"id":"123abc",
             "self":"https://appspot.com/loads/123abc"},
            {"id":"456",
             "self":"https://appspot.com/loads/456"}
            ]
  "self": "https://<your-app>/boats/def456"
},
{
  "id": "xyz123",
  "name": "Hocus Pocus",
   "type": "Sailboat",
  "length": 100,
  “owner”: 32987497397323
  "self": https://<your-app>/boats/xyz123
   "loads": [{"id":"123abc",
               "self":"https://appspot.com/loads/123abc"},
              {"id":"456",
               "self":"https://appspot.com/loads/456"}
             ]

}
   next:" https://<your-app>//boats?limit=3&offset=3"

]
```

Delete a Boat

Allows you to delete a boat. Note that if the boat is currently in a load, deleting the boat makes the load empty.

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

Request

Request Headers

Authorization: Bearer {JWT}

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	404 Not Found	No boat with this boat_id exists
Failure	405 Method Not Allowed	Invalid JWT used

https://project-loki-254500.appspot.com

Response Examples

Success

Status: 204 No Content

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

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
Cargo	String	name of cargo	Yes
Weight	Integer	Weight of cargo	Yes
Delivery_date	String	Date due to be delivered	Yes

Request Body Example

```
{
   "cargo":"LEGO Blocks",
   "weight":5,
   "delivery_date":"01/01/2020"
   "carrier:{}
}
```

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).

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 `current_boat` is the ID of the boat currently at this load. If there is no boat at this load, the value of `current_boat` 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

{
   "cargo":"LEGO Blocks",
   "weight":5,
   "delivery_date":"01/01/2020",
   "carrier":{},
   "id":6194625358331904,
   "self":http://127.0.0.1:5002/loads/6194625358331904
}
```

Failure

```
Status: 400 Bad Request

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

Get a Load

Allows you to get an existing load.

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

Request

Request Parameters

Name	Type	Description	Required?
load_id	String	ID of the load	Yes

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

Response Examples

Success

```
Status: 200 OK

{
   "cargo":"LEGO Blocks",
   "weight":5,
   "delivery_date":"01/01/2020",
   "carrier":{},
   "id":6194625358331904,
   "self":http://127.0.0.1:5002/loads/6194625358331904
}
```

Failure

```
Status: 404 Not Found

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

List all Loads In Boat

List all the loads.

```
GET /boats/<boat_id>/loads
```

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

[
{
   "cargo":"LEGOs",
   "weight":8,
   "delivery_date":"01/02/2020",
   "carrier":{<boat_id>, <boat_url>},
   "id":6194625358331904,
   "self":http://127.0.0.1:5002/loads/6194625358331904
},
{
   "cargo":"LEGO Blocks",
   "weight":5,
   "delivery_date":"01/01/2020",
   "carrier":{<boat_id>, <boat_url>},
   "id":6194625358331904,
   "self":http://127.0.0.1:5002/loads/6194625358331904
}
]
```

Delete a Load

Allows you to delete a load.

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

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	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"
}
```

Add Load to Boat

Boat has arrived at a load.

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

Request

Request Parameters

None

Request Body

Request JSON Attributes

Name	Type	Description	Required?
Id	Integer	Load Id	Yes

Request Body Example

```
{
   "id":"8249378326875798329",
}
```

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 boat is empty.
Failure	400 Forbidden	The load is already on a boat
Failure	404 Not Found	No boat with this boat_id exists, and/or no load with this load_id exits.

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 400 Forbidden

{
"Error":  " Load already assigned to a boat"
}
```

```
Status: 404 Not Found
```

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

}
```

Remove Load from boat

Boat has left the load to go to sea. The load is now empty.

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

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	Succeeds only if load is on a boat
Failure	404 Not Found	No load with this Id

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found

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

Get a User

Displays user and boats users.

```
GET users/:user_id
```

Request

Request Headers

Content-Type: application/json

Authorization: Bearer {JWT}

Request Body

Required

Request Body Format

JSON

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201 Ok	
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.
Failure	401 Unauthorized	Invalid JWT was used
Failure	406 Not Acceptable	Invalid format requested

Response Examples

- 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: 200 OK
{
  "email": "earmstrong81@gmail.com",
  "id": "6262364643000320",
  "boats": [],
  "sub": "107964095559115798100",
  "self": "http://127.0.0.1:8080/users/6262364643000320",
}
```

Failure

```
Status: 400 Bad Request

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

}

Status: 401
{
"Error": "Invalid JWT"
}

Status: 406
{
"Error": "Content must be in JSON format"
}