Brandon Lo

Rest Assignment 2

CS496

Google App Engine URL

https://rest-project-194300.appspot.com/

Model Entities

ID's were automatically created with ndb with urlsafe to allow for embedding into an URL, so ID's are omitted from the model entities

Boat

- name – string (required)
- type – string (required)
- length – integer (required)
- at_sea – bool (default set to true)
-

departure_history

- depature_date – string
- depBoat – string

Slip

- number – integer (required)
- current_boat– string
- arrival_date– integer
- departure_history – bool (extra credit)

URL

Create a Boat

POST /boats

Action: Creates a boat and returns it in a dictionary

Body:

{Name(required): String,

Type(required): String,

Length(required): integer,

at_sea: Boolean}

Response: Will return a 201 response if successful

Create a Slip

POST /slips

Action: Creates a slip and checks if it exists.

Body:

{number(required): integer}

Response: Returns a 201 response if successful. If slip exists or unsuccessful, it returns a 505 error.

Get All Boat Lists

GET /boats

Action: Gets all the boats as a list and returns it as a list

Response: Returns a 201 response if successful.

Get Single Boat by ID

GET /boats/<boatID>

Action: Get a boat by ID. Returns the list.

Response: Returns a 201 response if successful. Returns a 505 error if it is not found.

Get All Slips Lists

GET /slips

Action: Gets all the slips as a list and returns it as a list

Response: Returns a 201 response if successful.

Get Single Slip by ID

GET /slips/<slipID>

Action: Get a slips by ID. Returns the list.

Response: Returns a 201 response if successful. Returns a 505 error if it is not found.

Modify Boat Part

PATCH /boats/<boatID>

Action: Modifies a part of a boat based on the boatID

Body:

All are optional, but requires at least one modification

{name: string,

type: string,

length: integer}

Response: Returns a 201 response if successful. Returns a 505 error if it is not found.

Modify Entire Boat

PUT /boats/<boatID>

Action: Modifies entire boat based on the boatID

Body:

{name(required): string,

type(required): string,

Length(required): integer}

Response: Returns a 201 response if successful. Returns a 505 error if it is not found.

Modify Slip

PATCH /slips/<slipID>

Action: Modifies a slip based on the slipID. Also checks if the number currently exists.

Body:

{number(required): integer}

Response: Returns a 201 response if successful. Returns a 505 error if it is not found.

Delete A Boat

DELETE /boats/<boatID>

Action: Deletes a boat by ID and modifies the slip if the slip has the boat as docked. The departures history is also updated.

Response: Returns a 201 response if successful. Returns a 505 error if it is not found.

Delete A Slip

DELETE /slips/<slipID>

Action: Deletes a slip by ID and modifies the boat if the slip has the boat as docked. The departures history is also updated.

Response: Returns a 201 response if successful. Returns a 505 error if it is not found.

Put Ship to Sea

PUT /boats/<boatID>/atsea/<slipID>

Action: Puts ship out at sea, updates the depature history and the slip.

Body:

{departure_date(required): string}

Response: Returns a 201 response if successful. Returns a 402 if boat is already at sea. Returns a 403 is slip is empty. Returns 404 for invalid boatID. Returns 405 for invalid slipID.

Put Ship to into Slip (Dock ship

PUT /boats/<slipID>/dock/<boatID>

Action: Puts ship out at sea, updates the depature history and the slip.

Body:

{arrival_date(required): string}

Response: Returns a 201 response if successful. Returns a 402 if boat is already docked. Returns a 403 is slip is already full. Returns 404 for invalid boatID. Returns 405 for invalid slipID.