Mario Franco-Munoz
Assignment 3
Due: 7/15/2018

URL LISTING for Assignment 3 CS 496

ACTIVE URL: https://cs496-marina-210218.appspot.com/

'/' Mainpage doesn't do anything, just a simple "Hello World"

BoatHandler

- '/boats'
  - GET
    - When the GET method is called without an id in the URL, all boats in the database are returned in a single JSON object.
  - POST
    - POST method must be called with a body in the request that contains the *name*, *type,* and *length* of the boat that is to be created and added to the database.
    - Data validation:
      - Name is verified to not be a null string
      - Type is verified to not be a null string
      - Length is verified to be a positive non-zero/non-null value
- '/boats/{boat_id}'
  - GET
    - If the get method is called with an id in the url, the method checks to make sure that the id exists in the database and returns the data associated with that boat_id if it is valid as a JSON object
  - DELETE
    - This method verifies that the input id in the url is a valid url that exists in the database. If the url is valid, then the boat entry is removed from the database.
  - PATCH
    - Similar to the POST method, this method is used to update a given boat identified by the id in the PATCH url request. This method verifies that the id exists and that the body input data {"name": x, "type": x, "length": X} is valid.
    - This method has been written in such a way that when updating the fields, all fields must be updated.
    - Data validation: verifies that the input fields are not null and that the length is a non-negative non-zero/null value.

SlipHandler

- '/slips/{slip_id}
  - GET

        - When the GET method is called with an id in the URL, the handler checks to see if the ID is a valid and existing id of a slip. If it is, it returns as a JSON object the requested slip.
    - DELETE
        - Delete call must be called with an id in the URL. If the id is either null or invalid, the method returns an error. Calling this method deletes the slip and if a ship was docked there, the ship is set to be "at_sea = true."
    - PATCH
        - Patch method is used to update the *number* of a slip. The request URL must have a valid slilp id which is verified by the method. If the url is valid, the following checks are done on the id in the request body: Method checks that the id is non-null, non-negative and not already being used by another slip.
- '/slips
    - GET
        - When the GET method is called without a URL, all slips in the database are returned as a JSON object.
    - POST: When calling this url as a post method is created.
        - Input Data (as part of body of request): {"number":x } Since the id is a generated value, and the current_boat, arrival_date, and departure_history values should be updated via other interactions, I decided that the post method for this handler would only include one field: the slip number.
        - Data validation: this method checks that the number in the request body is not already in use and that it is non negative, and not null.

Arrival_DepartureHandler

- '/boat/{boat_id}/arrival'
    - POST: When calling this url as a post method with the {boat_id}, a boat is disassociated from the slip it was currently
        - Input Data (as part of body of request): {"number": x, "arrival_date": x}
        - This method checks to make sure that:
            - The boat actually exists
            - The slip number exists
            - Input data is not null
- '/boat/{boat_id}/departure'
    - PUT: When calling this url as a put method with the {boat_id}, a boat "departs" from the slip and is sent to sea
        - Input Data (as part of body request): {"departure_date": x}
        - Data validation: method makes sure that input date is not null

- '/marina'
    - GET only, calling this URL yields a full JSON output of all the *ships* and *slips*