Abreu, James
CS 496-400, Spring 2018

Assignment 01 – REST Planning and Implementation

Assignment Description/Goals: This is my first attempt to design and implement a RESTful API and my first time using google app engine to deploy a working application. The theme of the assignment is boats and slips (aka. docks). The focus of the design was to adhere to the core concepts of designing a RESTful API: URLs are nouns, interaction with the API uses HTTP verbs, and HTTP requests are mostly impodent (except for post requests). Boats and slips are created with POST, and can be retrieved with GET. Docking and undocking boats is done via a PATCH to a boat, and deletion of boats and slips can be done with DELETE requests. For more usage, visit the home ('/') directory of the applications URL, or see below.

Google App URL: https://reliable-sol-201922.appspot.com/

Project Source Code Github URL: https://github.com/JamCamAbreu/RESTful_Implementation-boats

Submitted Files Overview

File Name	Description
abreuj-CS496-RESTplanningImplementation.pdf	(This document)
abreuj-CS496-boatSlips-Postman	Directory containing Postman related files
RESTful API - Boats and Slips.postman_environment	Contains the environment for the postman tests
week03-BoatsSlips-Google.postman_collection	Contains the collection of postman tests
week03-BoatsSlips-Google.postman_test_run	Example results of running the tests while **using 450 ms delay between tests**
abreuj-CS496-boatsSlips-sourceCode	Contains a *copy* of the source code **(you can also visit the github project link above)**
main.py	Contains the python server code

Postman Notes: A delay of **at least 450 ms** should be set on the Postman tests to ensure the legitimacy of the data being tested.

URLs and Verbs

VERB url	**Description**	**Required**
GET /	Home directory. Lists usage of the app's API	
GET /boat	View all boats	
GET /boat/{boat_id}	View a specific boat	
GET /slip	View all slips	
GET /slip/{slip_id}	View a specific slip	
VERB url	**Description**	**Required**
POST /boat	Create a new boat. New boats start with an 'at_sea' status of "True".	Requires a name (string), type (string), and length in feet (positive integer): {"name": "Olyssius", "type": "skiff", "length": "8"}
POST /slip	Create a new slip.	Requires at least an empty json object. Can supply a number for the slip. If the slip number is already reserverd for another slip, then a new number will be supplied: {} OR {"number": "4"}
VERB url	**Description**	**Required**
DELETE /boat/{boat_id}	Delete a specific boat. Will first undock (aka. remove reference to any slip) before removal.	
DELETE /slip/{slip_id}	Delete a specific slip. Will first undock (aka. put boat out 'at_sea') before removal if needed.	
VERB url	**Description**	**Required**
PATCH /boat/{boat_id}	Used to dock or undock a ship. Will automatically modify references to boat on slips as needed. Telling a docked boat to dock or an undocked boat to undock will do as expected: nothing.	Requires desired boat status: {"at_sea": "False"} – to dock OR {"at_sea": "True"} – to undock