Conner Pappas, CS493: Assignment 3 - Build a Restful API
URL: https://restful-api-ships.appspot.com

GET /ships

Description: Get all ships stored in the datastore. The response is an array of objects that contain the properties of each ship in the datastore.

Example response:

```
Response code: 200
[
   {
      "name": "Test Ship 2",
      "type": "Ship 2",
      "length": 20,
      "id": "5649050225344512"
   },
   {
      "name": "Test Ship 1",
      "type": "Ship 1",
      "length": 10,
      "id": "5959639107633152"
   }
]
```

GET /ships/{shipID}

Description: Get a response containing a one element array containing an object that contains the properties of the requested shipID in the URL.

Example response:

```
Response code: 200
[
   {
      "name": "Test Ship 1",
      "type": "Ship 1",
      "length": 10,
      "id": "5959639107633152"
   }
]
```

POST /ships

Description: Create a new ship containing the properties specified in the request body. The properties that can be specified by the user are the name of the ship, the type of ship, and the length of the ship. The response is an object containing the ID of the new ship.

Example request:

```
{
        "name":"Test Ship 1",
        "type": "Ship 1",
        "length": 10
}
```

Example response:

```
Response code: 200
{"id": 5959639107633152}
```

PATCH /ships/{shipID}

Description: Modify a ship, specified in the URL, with the properties specified in the request body. The response is a one element array containing an object with the ship's modified properties.

Example request:

```
{
        "length": 42
}
```

Example response:

```
Response code: 200
[
   {
      "name": "Test Ship 1",
      "type": "Ship 1",
      "length": 42,
      "id": "5959639107633152"
   }
]
```

GET /slips

Description: Get all slips stored in the datastore. The response is an array of objects that contain the properties of each slip in the datastore.

Example response:

```
Response code: 200
[
   {
      "number": 1,
      "current_boat": null,
      "arrival_date": "",
      "id": "5660980839186432"
   },
   {
      "number": 2,
      "current_boat": null,
      "arrival_date": "",
      "id": "5719238044024832"
   }
]
```

GET /slips/{slipID}

Description: Get a response containing a one element array containing an object that contains the properties of the requested slipID in the URL.

Example response:

```
Response code: 200
[
   {
      "number": 1,
      "current_boat": null,
      "arrival_date": "",
      "id": "5660980839186432"
   }
]
```

POST /slips

Description: Create a new slip containing the properties specified in the request body. The only property that can be chosen by the user is the slip number since all slips are created empty. The response is an object containing the ID of the new slip.

Example request:

```
{
        "number": 1
}
```

Example response:

```
Response code: 200
{"id": 5660980839186432}
```

PATCH /slips/{slipID}

Description: Modify a slip, specified in the URL, with the properties specified in the request body. The only properties that can be changed without affecting a docked ship are the number and arrival date. This method can be used to make the current_boat property null to make the ship go back to being at sea. The response is a one element array containing an object with the slip's modified properties.

Example request:

```
{
        "number": 10
}
```

Example response:

```
Response code: 200
[
   {
      "number": 10,
      "current_boat": null,
      "arrival_date": "",
      "id": "5660980839186432"
   }
]
```

PUT slips/{slipID}/ships/{shipID}

Description: Dock a ship, specified by shipID, in a slip, specified by slipID. The request body should contain an arrival date to be placed in the slip's properties. The response is a one element array containing an object with the slip's new properties for current boat, arrival date, and an additional property for a link to the ship docked in that slip. If a ship is already in a slip while another ship occupies it then the server responds with a 403 response. A ship can be moved into another slip with this request, and the previous slip will be emptied.

Example request:

```
{
        "arrival_date": "1/10/19"
}
```

Example response:

```
[
   {
      "number": 10,
      "current_boat": "5959639107633152",
      "arrival_date": "1/10/19",
      "id": "5660980839186432",
      "ship_url": "https://restful-api-ships.appspot.com/ships/5959639107633152"
   }
]
```

PUT ships/{shipID}/slips/{slipID}

Description: Make a ship, specified by shipID, depart the slip, specified by slipID, it is currently occupying. If the ship isn't the slip then the server responds with a 403 response.

Example response:

Response code: 200 (upon success)
Response code: 403 (upon failure)

DELETE ships/{shipID}

Descriptions: Delete a ship, specified by shipID, from the datastore. Deleting a ship also frees up the slip it was occupying if it was in a slip.

Example response:

Response code: 200

DELETE slips/{slipID}

Descriptions: Delete a slip, specified by slipID, from the datastore.

Example response:

Response code: 200