Shopify Coding Challenge

For this challenge I decided implement the backend of the online store in Node.js using a Loopback framework which is based on Express. This allowed me to focus more on the functionality of the API's since Loopback provides you with a default structure to build off of while still allowing you to implement things using Express. The database used to store all of the products is MongoDB which is hosted in mlab which means it should be accessible on any system which runs this backend.

Set up

In order to run the application, you must install node on your system. If you do not have node set up, follow the instructions in the links below.

Windows: https://blog.teamtreehouse.com/install-node-js-npm-windows
Mac: https://blog.teamtreehouse.com/install-node-js-npm-mac
Linux: https://blog.teamtreehouse.com/install-node-js-npm-linux

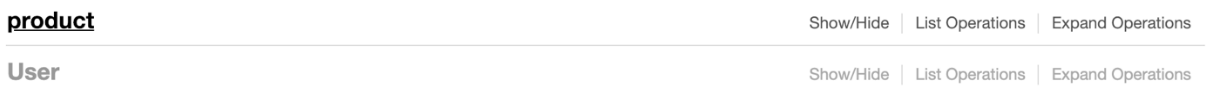
Once everything is installed open your terminal and change into the Spotify directory in the server folder within the unzipped zip file containing the challenge and run node command.

```
node server.js
```

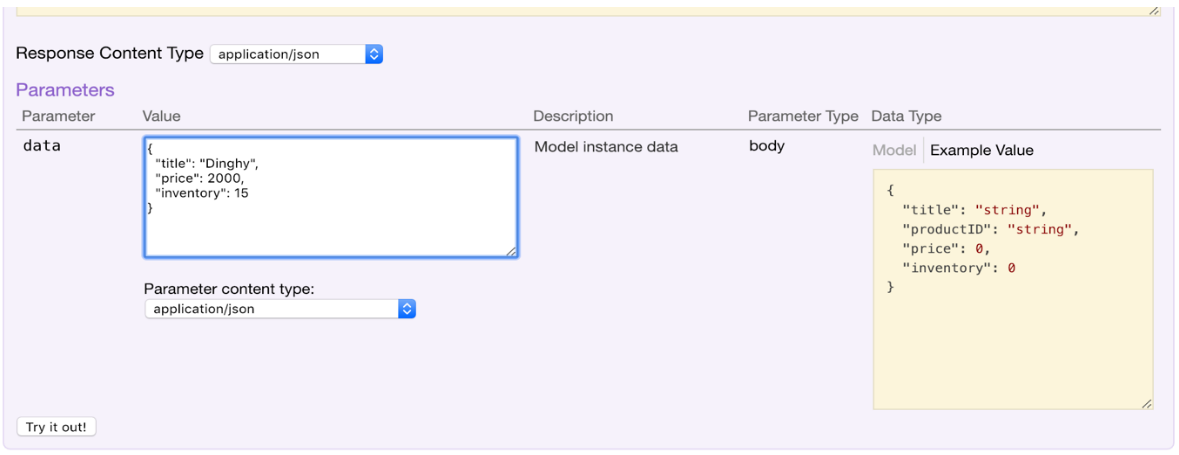
Copy the link http://0.0.0.0:3000/explorer or the Browse API link that may show up and paste it in your browser to test the APIs. If you can error concerning the database run the command below and try again.

```
npm install loopback-mongodb-connector --save
```

To test the API open the product model tab in the API explorer page


product
Show/Hide | List Operations | Expand Operations
User
Show/Hide | List Operations | Expand Operations

Then select an API and enter in arguments to test. The example below shows someone entering a product into the database.



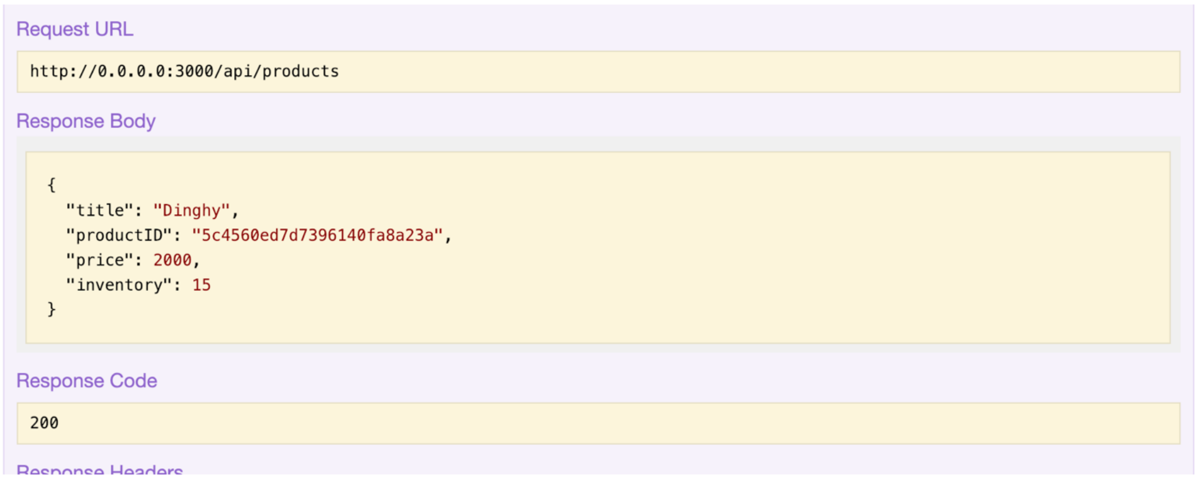
Response Content Type: application/json

Parameter	Value	Description	Parameter Type	Data Type
data	`{ "title": "Dinghy", "price": 2000, "inventory": 15 }`	Model instance data	body	Model / Example Value

```
{
 "title": "Dinghy",
 "price": 2000,
 "inventory": 15
}
```

Parameter content type: application/json

```
{
  "title": "string",
  "productID": "string",
  "price": 0,
  "inventory": 0
}
```


Request URL

```
http://0.0.0.0:3000/api/products
```

Response Body

```
{
  "title": "Dinghy",
  "productID": "5c4560ed7d7396140fa8a23a",
  "price": 2000,
  "inventory": 15
}
```

Response Code

```
200
```

Response Headers

*When entering a product into the database do not include the productid parameter as that will be automatically generated when persisting the model into the database

API

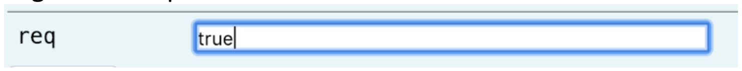
/products/findAllProducts

This API will fetch all available products in the store and filter products with no inventory when argument is passed through.

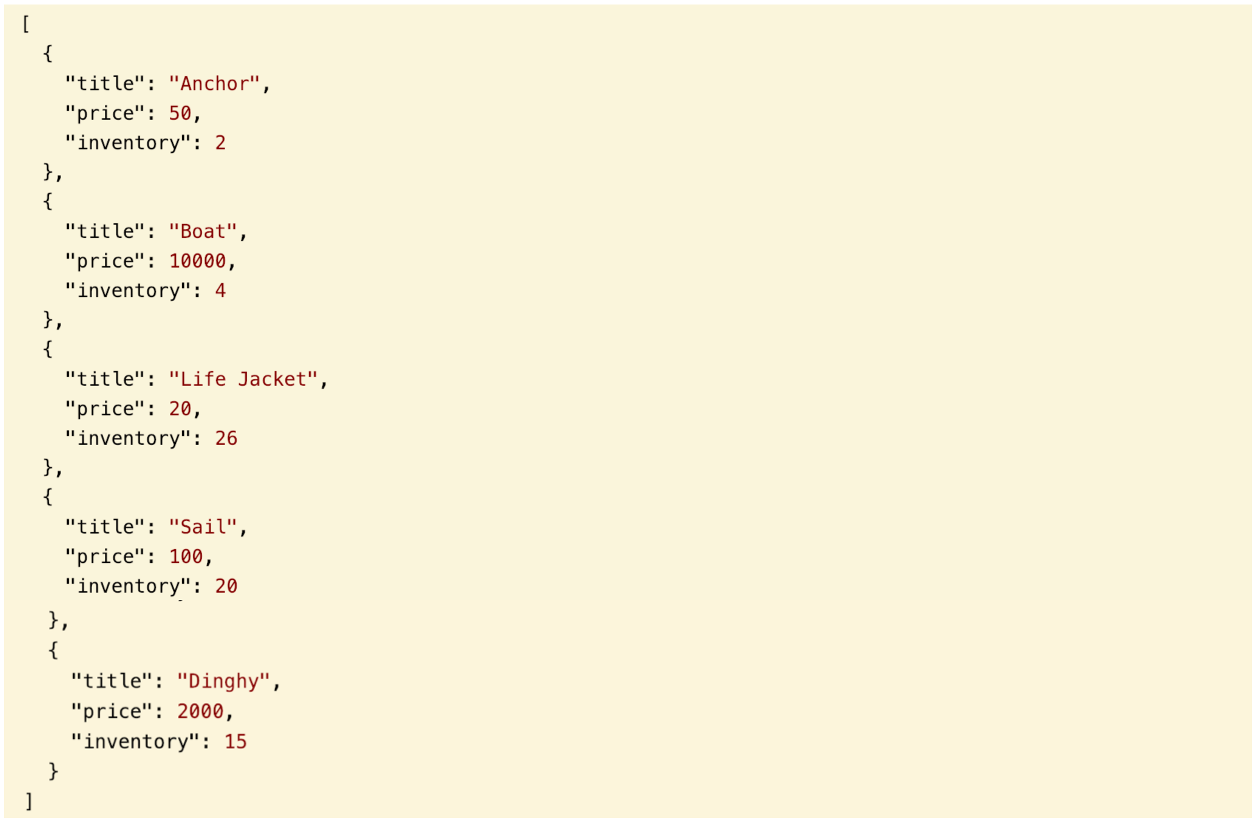
No argument response:

```
[
  {
    "title": "Anchor",
    "price": 50,
    "inventory": 2
  },
  {
    "title": "Boat",
    "price": 10000,
    "inventory": 4
  },
  {
    "title": "Rope",
    "price": 10000,
    "inventory": 0
  },
  {
    "title": "Life Jacket",
    "price": 20,
    "inventory": 26
  },
  {
    "title": "Sail",
    "price": 100,
    "inventory": 20
  },
  {
    "title": "Dinghy",
    "price": 2000,
    "inventory": 15
  }
]
```

Argument response:



```
req    true
```

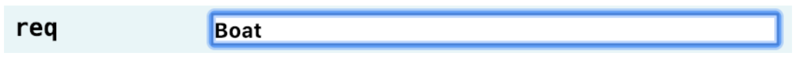


```
[
  {
    "title": "Anchor",
    "price": 50,
    "inventory": 2
  },
  {
    "title": "Boat",
    "price": 10000,
    "inventory": 4
  },
  {
    "title": "Life Jacket",
    "price": 20,
    "inventory": 26
  },
  {
    "title": "Sail",
    "price": 100,
    "inventory": 20
  },
  {
    "title": "Dinghy",
    "price": 2000,
    "inventory": 15
  }
]
```

/products/findProduct

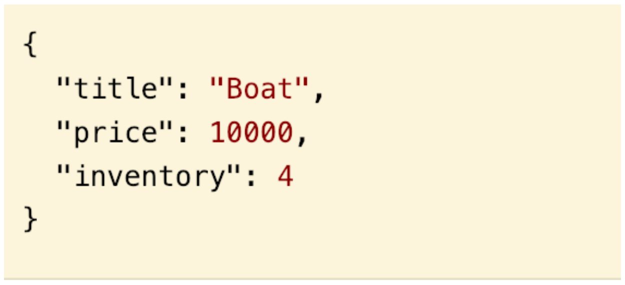
This API will fetch a single product provided by the argument. It will return an error if you try to fetch an item that has no inventory.

Product with inventory:



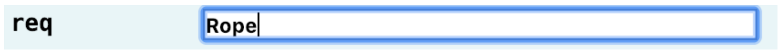
```
req    Boat
```

Response:



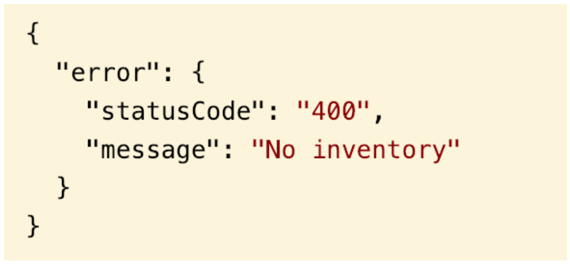
```
{
  "title": "Boat",
  "price": 10000,
  "inventory": 4
}
```

Product without inventory:



req	Rope

Response:

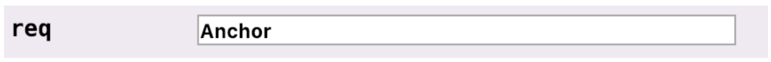


```
{
  "error": {
    "statusCode": "400",
    "message": "No inventory"
  }
}
```

/products/purchaseProduct

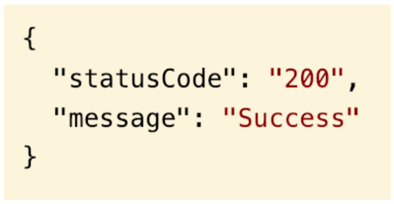
This API will purchase a specified item which will reduce the item inventory in the database by one. The response will be a success message meaning that the item has been purchased or a failure if there is no inventory for that product. To see the reduced inventory, you can fetch the product again using the findProduct API to see the inventory reduce.

Product with inventory:



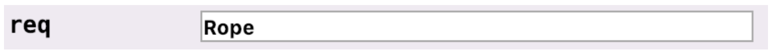
req	Anchor

Response:



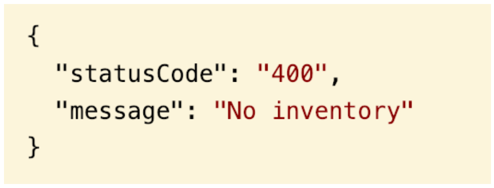
```
{
  "statusCode": "200",
  "message": "Success"
}
```

Product without inventory:



req	Rope

Response:



```
{
  "statusCode": "400",
  "message": "No inventory"
}
```

Code Base

To check the code base for this challenge, open the product.js file inside Shopify/common/models.