Express Bookstore

Download starter code <../express-bookstore.zip>

For this exercise, you will build an express.js application that validates a resource and then add tests to the application.

Getting started

You will be adding validation to an application that stores one resource, books. Here is an example of what a book object should look like:

```
{
  "isbn": "0691161518",
  "amazon_url": "http://a.co/eobPtX2",
  "author": "Matthew Lane",
  "language": "english",
  "pages": 264,
  "publisher": "Princeton University Press",
  "title": "Power-Up: Unlocking the Hidden Mathematics in Video Games",
  "year": 2017
}
```

Your application currently consists of the following routes:

GET /books
Responds with a list of all the books

POST /books
Creates a book and responds with the newly created book

GET /books/[isbn]
Responds with a single book found by its ***isbn***

PUT /books/[isbn]
Updates a book and responds with the updated book

DELETE /books/[isbn]
Deletes a book and responds with a message of “Book deleted”

Part One - Getting Started + Adding Validation

Before you get started, read through the code to make sure you understand what’s going on here.

Create your database and then run the ***data.sql*** file. You can find the database name in the ***config.js*** file.

Use JSONSchema to validate the creation and updating of your books! Display an error message containing all of the validation errors if book creation or updating fails.

Part Two - Add Tests

Add integration tests for each of your routes to make sure that the response expected is sent.

Think about certain edge cases for each of these routes and add tests for things like invalid input to make sure your schema validation is correct.

Also make sure to set ***process.env.NODE_ENV = "test"*** inside of your test file.

Further Study

- Create functionality for partially updating a book. You should not have to pass in all the fields to update a book, just the ones that need to be updated. Write unit tests for this function!
- Build a front-end for this application. Use ***axios*** for AJAX requests!
- Add a more complex database schema! What if you want to tag books by genre, or build a many-to-many between users and authors? Talk with your pair about how you could refine your data model, and what additional routes you'd need to support your model. Then build it!

Solution

View our solution <solution/index.html>