Final Project - Book Review Application

Estimated Time Needed: 2 hours

- In this final project, we will build a server-side online book review application and integrate it with a secure REST API server which will use authentication at the session level using JWT. You will then test your application using Promises callbacks or Async-Await functions.

Objectives:

After completing this lab, you will be able to:

1. Create APIs and perform CRUD operations on an Express server using Session & JWT authentication.
2. Use Async/Await or Promises with Axios in Node.js.
3. Create REST API endpoints and test them using Postman.

Set-up : Create application

1. Open a terminal window by using the menu in the editor: Terminal > New Terminal.
2. Change to your project folder, if you are not in the project folder already.

   ```
   cd /home/project
   ```

3. Please fork the Git repository that contains the starter code needed for this lab:

   ```
   https://github.com/ibm-developer-skills-network/expressBookReviews.git
   ```

4. Clone your forked Git repository, if it doesn't already exist.

   ```
   [ ! -d 'expressBookReviews' ] && git clone https://github.com/<your Github username>/expressBookReviews.git
   ```

5. Change to the directory **expressBookReviews/final_project/** directory to start working on the lab.

   ```
   cd expressBookReviews/final_project/
   ```

6. List the contents of this directory to see the artifacts for this lab.

   ```
   ls
   ```

Understanding the server application

1. In the files explorer open the **expressBookReviews/final_project/** folder and view `index.js`having the below code:

   ```
   const express = require('express');
   const jwt = require('jsonwebtoken');
   const session = require('express-session')
   const customer_routes = require('./router/auth_users.js').authenticated;
   const genl_routes = require('./router/general.js').general;
   const app = express();
   app.use(express.json());
   app.use("/customer",session({secret:"fingerprint_customer",resave: true, saveUninitialized: true}))
   app.use("/customer/auth/*", function auth(req,res,next){
   //Write the authenication mechanism here
   });

   const PORT =5000;
   app.use("/customer", customer_routes);
   app.use("/", genl_routes);
   app.listen(PORT,()=>console.log("Server is running"));
   ```

2. The packages required for this lab are defined in as dependencies in `packages.json` as below:

   ```
     "dependencies": {
   ```

```
    "express": "^4.18.1",
    "express-session": "^1.17.3",
    "jsonwebtoken": "^8.5.1",
    "nodemon": "^2.0.19"
}
```

Understanding the user routes

Navigate to the `router` directory having the below 3 files:

1. `booksdb.js` - This contains the the preloaded book information for this application.
2. `general.js` - This contains the skeletal implementations for the routes which a general user can access.
3. `auth_users.js` - This contains the skeletal implementations for the routes which an authorized user can access.

Updating the code for the authentication mechanism:

- Navigate to `index.js` and update the authentication code under `app.use("/customer/auth/*", function auth(req,res,next){`:

Hint: Use the session authorization feature (implemented in the Practice project lab) to authenticate a user based on the access token.

Update and test the general user routes in `general.js`

Note:

- Please remember to push your work to the GitHub repository after completing every task or if you're not completing the lab in one session. You can find the steps for pushing the repository to GitHub in Task 14. This will help you avoid losing your progress.
- If you encounter difficulties using Postman, you can alternatively use `curl` commands to test the API endpoints, as demonstrated in Module 3: Hands-on Lab: CRUD Operations with Node.js and Express. Please provide screenshots showing the JSON output of your `curl` commands in your submission for peer review.

Task 1:

- Complete the code for getting the list of books available in the shop under `public_users.get('/',function (req, res) {`.

Hint: Use the JSON.stringify method for displaying the output neatly.

- Run `npm install` for installing the required modules & start the server.
- Test the output on Postman.
- Please take a screenshot of the same and save it with the name `1-getallbooks.png` for submitting under Task 1 for the Peer Review Assignment.

Task 2:

- Complete the code for getting the book details based on ISBN under `public_users.get('/isbn/:isbn',function (req, res) {`.

Hint: Retrieve the ISBN from the request parameters

- Test the output on Postman.
- Please take a screenshot of the same and save it with the name `2-gedetailsISBN.png` for submitting under Task 2 for the Peer Review Assignment.

Task 3:

- Complete the code for getting the book details based on the author under `public_users.get('/author/:author',function (req, res) {`.

Hints:
1. Obtain all the keys for the 'books' object.
2. Iterate through the 'books' array & check the author matches the one provided in the request parameters.

- Test the output on Postman.
- Please take a screenshot of the same and save it with the name `3-getbooksbyauthor.png` for submitting under Task 3 for the Peer Review Assignment.

Task 4:

- Complete the code for getting the book details based on the title under `public_users.get('/title/:title',function (req, res) {`.

Hint: This will be similar to Exercise 3

- Test the output on Postman.
- Please take a screenshot of the same and save it with the name `4-getbooksbytitle.png` for submitting under Task 4 for the Peer Review Assignment.

Task 5:

- Complete the code for getting book reviews under `public_users.get('/review/:isbn',function (req, res) {`.

Hint: Get the book reviews based on ISBN provided in the request parameters.

- Please take a screenshot of the same and save it with the name `5-getbookreview.png` for submitting under Task 5 for the Peer Review Assignment.

Task 6:

- Complete the code for registering a new user

Hint: The code should take the 'username' and 'password' provided in the body of the request for registration. If the username already exists, it must mention the same & must also show other errors like eg. when username &/ password are not provided.

- Test the output on Postman.
- Please take a screenshot of the same and save it with the name `6-register.png` for submitting under Task 6 for the Peer Review Assignment.

Update and test the authenticated user routes in `auth_users.js`.

Task 7:

- Complete the code for logging in as a registered user.

Hint: The code must validate and sign in a customer based on the username and password created in Exercise 6. It must also save the user credentials for the session as a JWT.
As you are required to login as a customer, while testing the output on Postman, use the endpoint as "customer/login"

- Test the output on Postman.
- Please take a screenshot of the same and save it with the name `7-login.png` for submitting under Task 7 for the Peer Review Assignment.

Task 8:

- Complete the code for adding or modifying a book review.

Hint: You have to give a review as a request query & it must get posted with the username (stored in the session) posted. If the same user posts a different review on the same ISBN, it should modify the existing review. If another user logs in and posts a review on the same ISBN, it will get added as a different review under the same ISBN.

- Test the output on Postman.
- Please take a screenshot of the same and save it with the name `8-reviewadded.png` for submitting under Task 8 for the Peer Review Assignment.

Task 9:

Complete the code for deleting a book review under `regd_users.delete("/auth/review/:isbn", (req, res) => {`

Hint: Filter & delete the reviews based on the session username, so that a user can delete only his/her reviews and not other users'.

- Test the output on Postman.
- Please take a screenshot of the same and save it with the name `9-deletereview.png` for submitting under Task 9 for the Peer Review Assignment.

With this you have implemented and tested the codes for the general and authenticated user routes.

Improving the scope of Tasks 1-4 using Promises or Async-Await

You will now use Promise callbacks or Async-Await functions for doing the same functionality which we covered synchronously in Tasks 1-4.

> **Note:** Please take a screenshot of your code implementation using either async/await or Promises for Tasks 10-13. This screenshot will be used for peer review submission.

Task 10:

- Add the code for getting the list of books available in the shop (done in Task 1) using Promise callbacks or async-await with Axios.

Hint: Refer to this lab on Promises and Callbacks.

- Please ensure that the `general.js` file has the code for getting the list of books available in the shop using Promise callbacks or async-await with Axios is covered.
- Please take a screenshot of the same and save it with the name task10.png for submitting under Task 10 for the Peer Review Assignment.

Task 11:

- Add the code for getting the book details based on ISBN (done in Task 2) using Promise callbacks or async-await with Axios.

Hint: Refer to this lab on Promises and Callbacks.

- Please ensure that the `general.js` file has the code for getting the book details based on ISBN using Promise callbacks or async-await with Axios is covered.
- Please take a screenshot of the same and save it with the name task11.png for submitting under Task 11 for the Peer Review Assignment.

Task 12:

- Add the code for getting the book details based on Author (done in Task 3) using Promise callbacks or async-await with Axios.

Hint: Refer to this lab on Promises and Callbacks.

- Please ensure that the `general.js` file has the code for or getting the book details based on Author using Promise callbacks or async-await with Axios is covered.
- Please take a screenshot of the same and save it with the name task12.png for submitting under Task 12 for the Peer Review Assignment.

Task 13:

- Add the code for getting the book details based on Title (done in Task 4) using Promise callbacks or async-await with Axios.

Hint: Refer to this lab on Promises and Callbacks.

- Please ensure that the `general.js` file has the code for or getting the book details based on Title using Promise callbacks or async-await with Axios is covered.
- Please take a screenshot of the same and save it with the name task13.png for submitting under Task 13 for the Peer Review Assignment.

Github repo updation for peer review submission

Task 14:

- Please commit and push all the changes to your forked Github repo.

  Note: Please refer to this lab, if you need any help in committing and pushing changes to your repo.

- Your Github repo link will be used for the evaluation of Task 14 in the peer review assignment.

Congratulations! You have completed the Final project!

Summary:

In this lab, you have built a server-side online book review application, integrated it with a secure REST API using JWT based session level authentication, and tested the built application using Promises callbacks or Async-Await functions.

Author(s)

Lavanya T S

Sapthashree K S

K Sundararajan

© IBM Corporation. All rights reserved.