PROG2002 Assignment 2

Weight: 40% of your final mark

Due: Week 6 (Monday, 23:59 AEST)

Please note that this is an individual assignment

Specifications

Previously, you have develop the client-side of an online book store website, but didn't involve the server side, didn't connect to a database, and without an admin site to manipulate the data.

In real practice, a website application needs to reside at a server, connect to a database and is able to handle clients' requests, and generate a dynamic content to the clients based on their requirest. Therefore, we will use the same case study using AngularJS.

In this assessment, your task is to develop the server-side of an online book store website using AngularJS. The web application is expected to have the following requirements.

- A MySQL database using NodeJS that stores the available books.
- RESTful APIs to fetch the details of the list of available books, including insert, update and delete actions.
- A simple client-side Angular app to display the list of books
- A simple admin-side Angular app to display the list of books, including insert, update and delete actions.

Getting Help

This assignment is to be completed individually. It is the opportunity to gain an understanding of the concepts of building a dynamic website. It is important that you master these concepts yourself. You are permitted to work from the examples in the study guide or textbook but you must acknowledge assistance from other textbooks or classmates. In particular, you **must not** use online material or help from others, as this would prevent you from mastering these concepts.

Who can you get help from? Use this diagram to determine from whom you may seek help with your program.

🟩 Lecturer	🟩 Tutors	🟩 Online Forums
🟥 Relatives	🟥 Students outside unit	🟥 Hired coders
🟩 Classmates	🟩 Private Tutors	🟦 Other

Encouraged

Attribution Required

Ask tutor

Not acceptable

Please note that if your marker has any suspicion that you had help with your code or that your work is not your own, you will be asked to come to a meeting with your marker to explain your code. Any student who is unable to explain their code will be submitted for academic misconduct.

Part 1 – Setup MySQL database using NodeJS

You are required to complete the following requirements using MySQL and NodeJS. *This part is covered in Module 4.*

- Install a MySQL server, connect to the MySQL server, and create a new database called "**bookstore_db**" by creating a file named "***bookstore_db.js".***
- Create two new tables called ***book*** and ***category*** that store the list of available books and the book categories.
    - In the **book** table, create the following fields: book_ID (PK), title, author, price, description, category_ID (FK). A book can only be classified into a category.
    - In the **category** table, create the following fields: category_ID (PK), name
- For the two tables above, use appropriate data type for each of the fields
- Insert at least 10 records into the **book** table and at least 5 records into the **category** table

Part 2 – Create a RESTFul API

Create RESTful APIs using NodeJS and ExpressJS to manipulate the **book** table only in the database. The APIs should have the following requirements. *This part is covered in Module 4.*

- **Using GET method** to retrieve all books from the database

(see *Module 4 > Displaying database results on Client Side > Hosting RESTful API*)

- **Using GET method with 1 parameter** (categoryid) to retrieve a list of book based on the category id inputed from the database
  (see *Module 4 > Displaying database results on Client Side > Read: Passing on the details via the URL*)
- **Using POST method** to insert a new book record to the database
  (see *Module 4 > Displaying database results on Client Side > Create: Post method*)
- **Using PUT method** to update a book based on the book ID inputed
  (see *Module 4 > Displaying database results on Client Side > Update: Put method*)
- **Using DELETE method** to delete a book based on the book ID inputed
  (see *Module 4 > Displaying database results on Client Side > Delete: Deete method*)

Once you complete this part, you can test your APIs using Postman. *(covered in Module 4)*

Note:

- You are not required to create APIs for the category table.
- You are not required to use Injectable Service for this part.

Part 3 –Angular project for a simple client-side web

You are required to create a **simple one-page web application using AngularJS** to display a list of books stored in the database by calling the GET API (that you made in Part 2). The store customers will use this web page to see the list of available books. The page should have the following requirements. *This part is covered in Module 5.*

- Create an Angular project and import all the required modules such as ***HttpClient*** etc.
- Using Angular component, add a page in your Angular project to display the details of book and the categories (see Module 5).

You are free to decide the page layout as long as it shows the required information.

Note:

- Don't spend your time too much on the layout. Hint: You may use the same layout that you made in Assessment 1 Part 1 to save your time.
- Use one page only.
- Notice that this part is similar to Assessment 1 but excluding the Parts 2-4 (i.e. inquiry page and shopping cart page).

Part 4 –Angular project for a simple admin-side web

You are required to create a **simple one-page web application using AngularJS** to manipulate the books stored in the database by calling the APIs (that you made in Part 2). The store owner will use this webpage to manipulate the book data. The page should have the following requirements. *This part is covered in Module 5.*

- Display all the books by calling GET API
- Create a new form to insert a new book. Once the user presses a SAVE button, the button will call POST method created in Part 2. Validation should be performed to make sure the user fills in the required fields.
- Create a new form to update a book. Once the user presses a SAVE button, the button will call PUT method created in Part 2. Validation should be performed to make sure the user fills in the required fields.
- Create a REMOVE link next to each book to delete the book. Once the user presses the REMOVE link, the button will call DELETE method created in Part 2.

Note:

- You are free to decide the page layout as long as it can perform the required actions.
- You are not required to use multiple components but it doesn't limit you to use multiple components and Angular route to handle multiple components (see Module 6) if you want to.
- You are not required to manipulate the category table.
- For simpliclty, assume that the admin web doesn't need the user to log in.

Submission instruction:

- You are required to **upload your codes to the SCU server** so the website will be fully working. Each student has been sent a cPanel login to access your account (please check your email). There is a video to guide you on how to upload your website to the SCU cPanel server (see Additional videos to support your assessment)
- You are also required to upload your code to the submission link on Blackboard.