**Web Technologies PC Assessment (JavaScript, JQuery, Ajax, Java EE)**

**Time allowed**: 2 hours

**What to submit** : A .zip folder of your project. Also include a word document with Figures similar to Fig. 5,8,11 and 12 depending on how much you completed.

Step 1. Find All Books (test with browser) Completed

Step 2. Find a Book by id (test with browser ) Completed

Step 3. Display list of all books in browser (Figure 5) **[35 marks]**

Step 4 :Click on “More Info” icon (Figure 9) **[35 marks]**

Step 5 Search criteria – (Figure 10, 11,12) **[30 marks]**

NB: You have been given a number of files. The only client side file you need to create/edit is main.js. You will also need to add some methods in your server side java code in Step 5. Do not edit any other files such as the index.html or the css file. Otherwise it may not be possible to correct your assessment.

In the JavaScript use jQuery methods for fetching the data and manipulating the DOM.

**NB Notes on marking scheme**: This exercise needs to be completed on a step by step basis. Marks cannot be allocated for code written but not working. Marks will not be allocated for code that partially implements a Step or code that does not work.

**Step 1:Find all books API and display in browser**

Code for this is given.

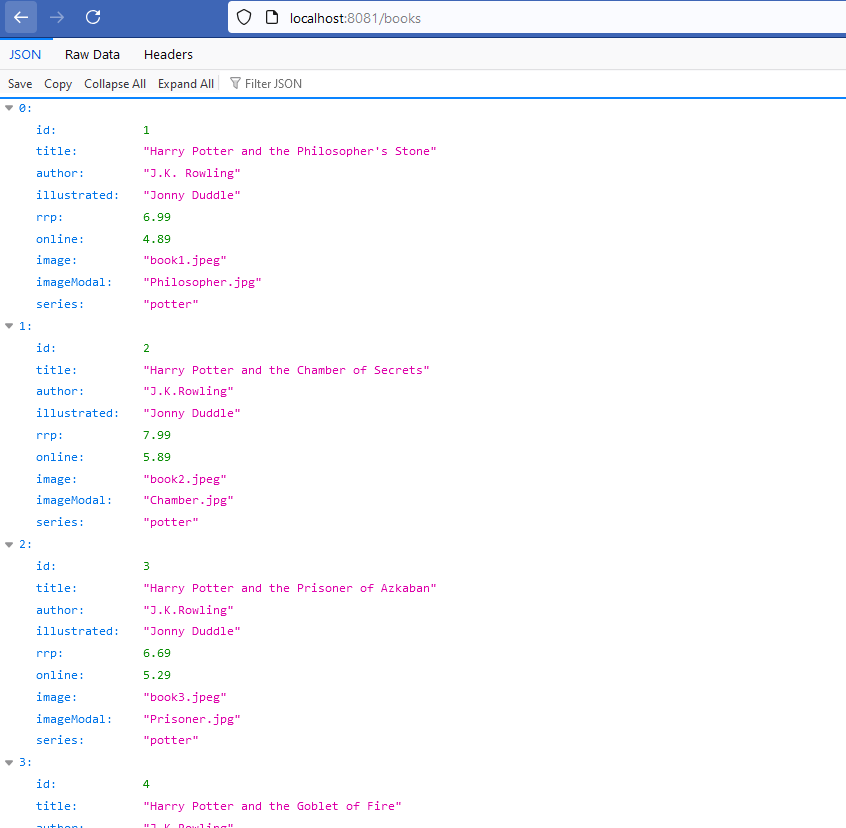


Figure 1: Browser – get all books

**Step 2:Find one book by its id and display result in browser**

Code given.

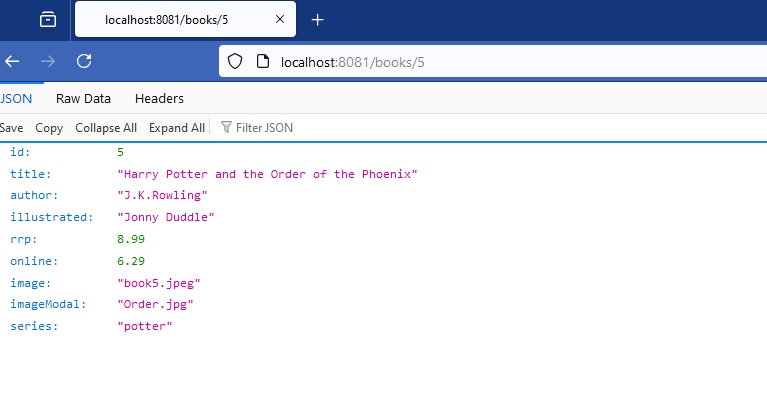


Figure 2: Output from browser – get book by id

**Step 3:Display a list of all books in the browser**

You have been given a html file and a css file ( index.html and style.css). Do **not** make any updates to these files. You have also been given a folder of images and some library css, fonts and js (empty).

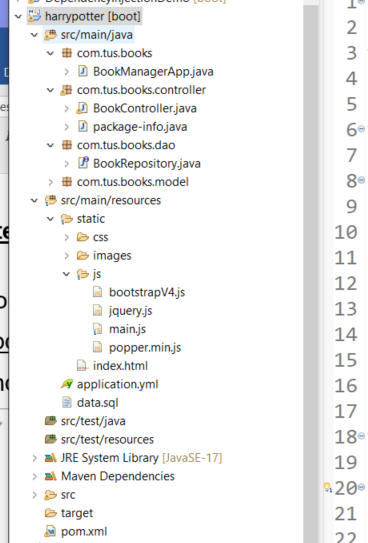


Figure 3: Project structure

Write the code needed in main.js to get the functionality described below. This is what the page looks like initially before you add any javascript code to main.js.

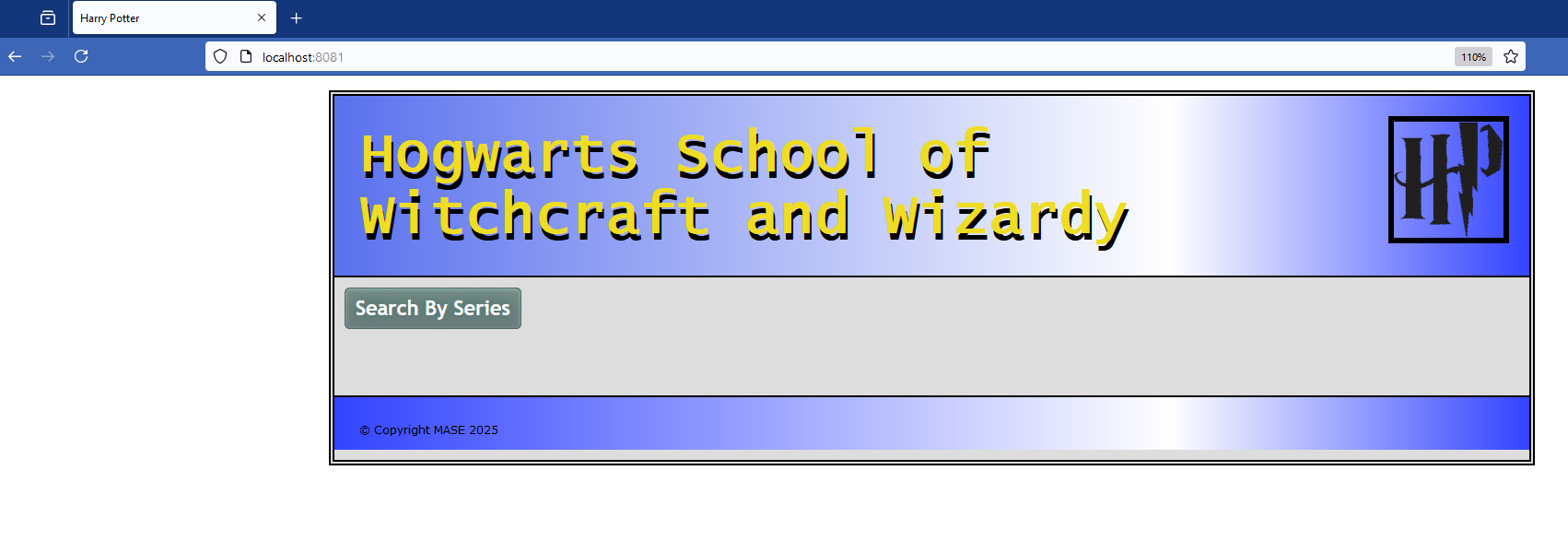


Figure 4: Browser output with no javascript in main.js

Add the javascript so that all the books are added to the <section id=”bookList”> that is contained in index.html. This is what the page should look like when the list of books is displayed.

A screenshot of the generated markup is shown on the next page. This is the markup that you need to generate using JavaScript. NB Generate this markup up with JavaScript – do not write it in HTML.





Figure 5: Browser output showing all books

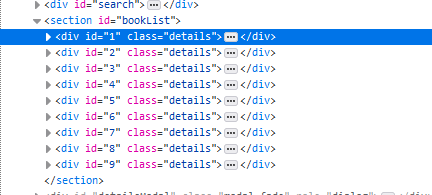


Figure 6: Example of generated markup – from inspect element

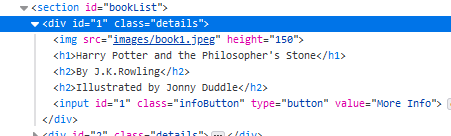


Figure 7: Example of generated markup for each div – from inspect element

Hint: If you don’t use the <div class=”details”> as a wrapper styling will not be applied correctly. All books should be displayed.

**Step 4:Click on the “More Info” button**

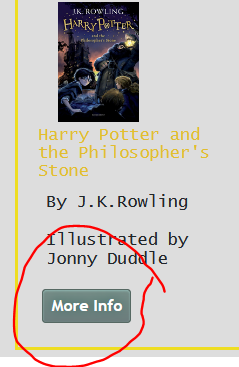


Figure 8

When the “More Info” button is clicked for a particular book, more details for that book are displayed in a model. Pay attention to the modal title and saving and % saving must be calculated from the rrp and online price. Use the modal with the id “detailsModal” that is provided in the html.

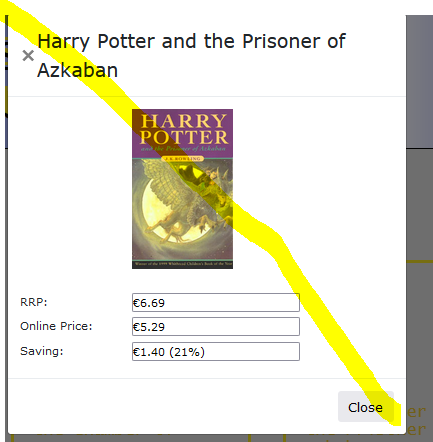
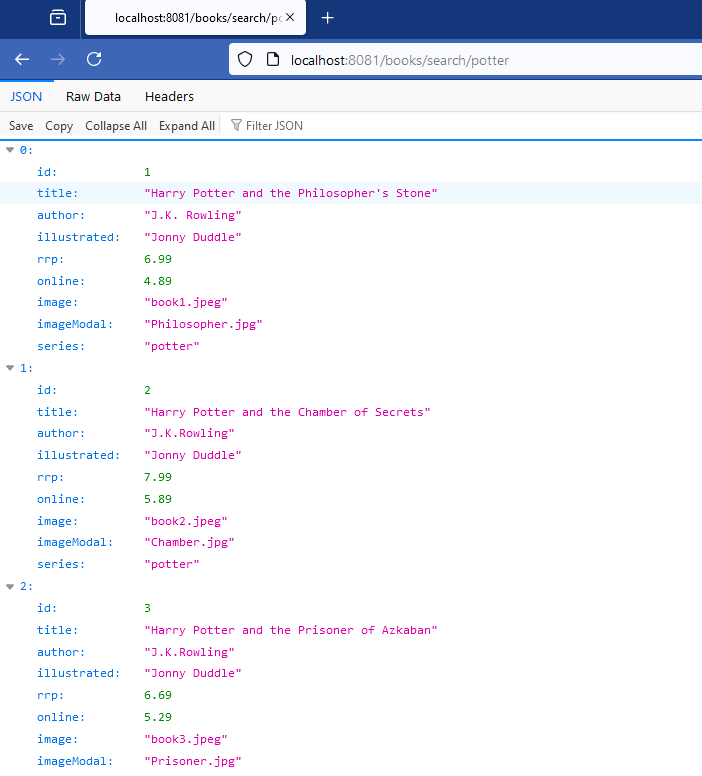
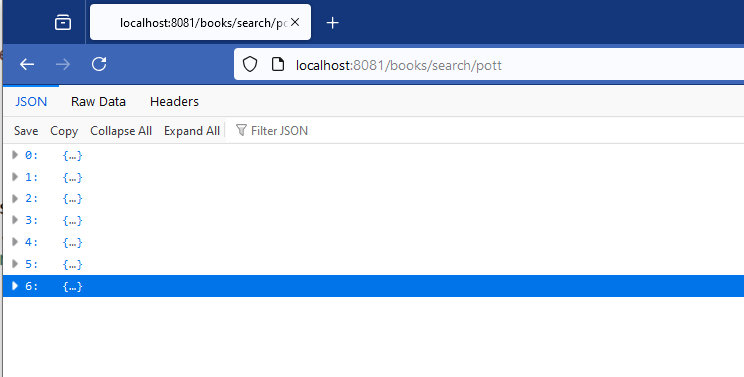


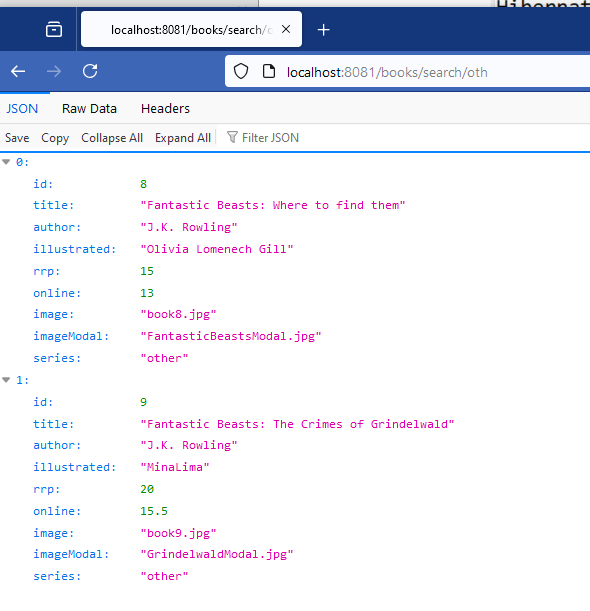
Figure 9: After clicking on “More Info” button

**Step 5: Adding search functionality on the server**

Add a search API option based on series on the server with url as shown. The search should work for part match e.g “pott” returns all the potter series.







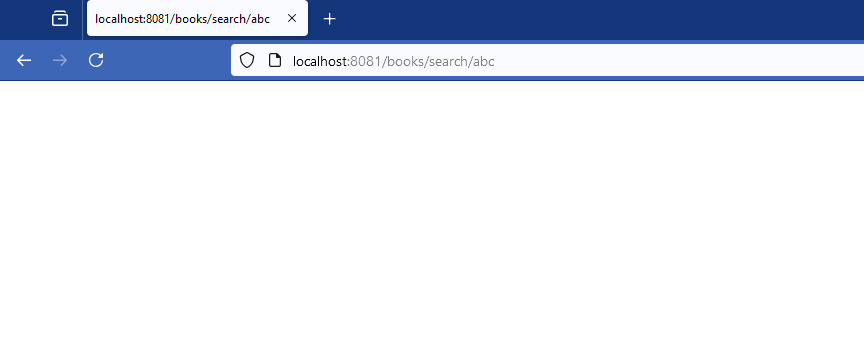


Figure 10: Search criteria

**Step 6: Filtering based on Search Criteria**

When the “Search By Series” button  is clicked, the modal with the id “filterModal” is displayed.

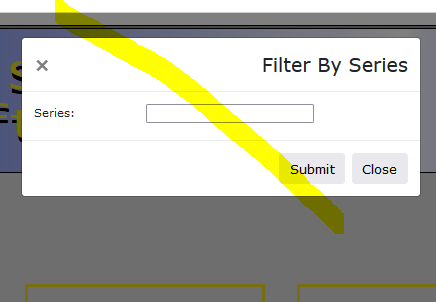


Figure 11: Filter Modal displayed

When the data is returned it is displayed as shown in Figure 11. If the search field is blank all books should be displayed. An unknown series results in no books being displayed

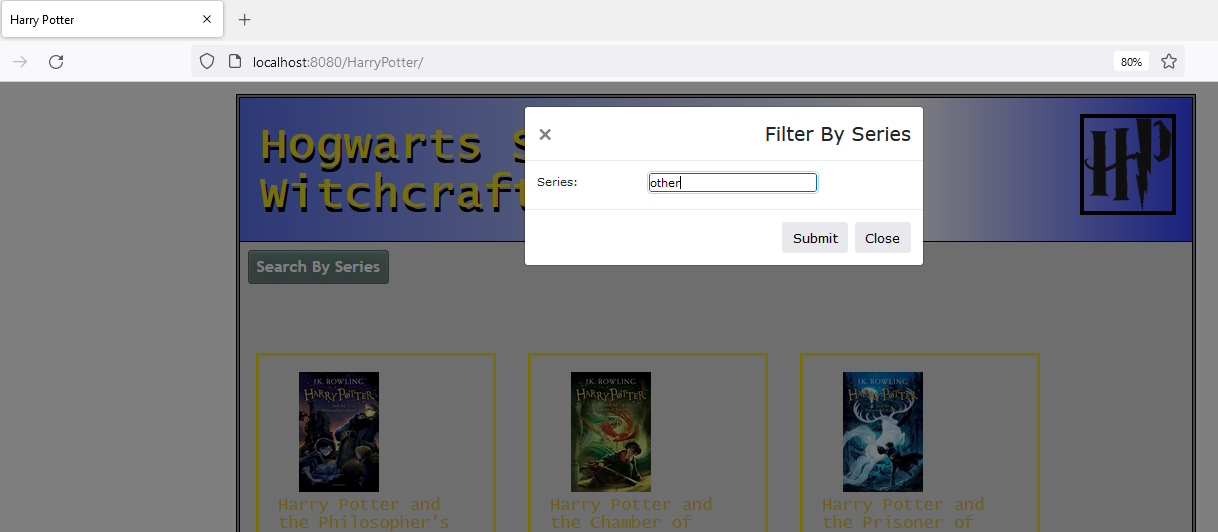




Figure 12 Search criteria