

HTML, CSS & Javascript

Building stuff with MVC

Books Shop (code name: book-shop)

Your challenge is to build a page that shows a list of books: *id*, *name*, *price*, *imgUrl*. We will allow the user – a shop owner – to manage the books.

1. Setup your project
 - a. Use a book-service to manage the books
 - b. The service will use the localStorage to load and store books.
2. Create your Model and show the books in a table. We will use a global variable `gBooks`, and a function `renderBooks()` that will draw the table

Now, let's handle CRUD (Create, Read, Update and Delete)

3. Add an Actions column to the table (something like this :)

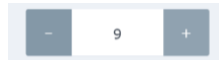
Welcome to my bookshop

[Create new book](#)

Id	Title	Price	Actions		
1	Learning Laravel	18.90	Read	Update	Delete
4	Beginning with Laravel	6.65	Read	Update	Delete
5	Java for developers	7.20	Read	Update	Delete

4. Handle delete - when the button clicked call `onRemoveBook` that will use the service's function `removeBook(bookId)`
5. Support adding a new book:
 - a. When clicked, call the function `onAddBook()` that will read (prompt) the details from the user: name and price, then will call a function `addBook(name, price)` that pushes a new book into the `gBooks` model. Then call the `renderBooks()` to redraw the table
6. Support updating a book:

- a. When clicked, call the function:
`onUpdateBook (bookId)` that will prompt for the book new price and call the service's function `updateBook(bookId, bookPrice)`.
Then Call the `renderBooks()` to redraw the table
7. Create a modal: **Book Details**, that shows the details of a selected book including its photo
 - a. When read is clicked, show the modal with details of the selected book.
 - b. Add a rate property for the book, set 0 as default, the rate should be a number between 0 and 10.
 - c. In the Book Details, allow the user to change the rate of the book by clicking the + / - buttons:



Bonus

1. Read the data from the user using an `<input>` instead of prompt
2. Make the header of the table clickable to support sorting by name or price
 - a. Note: Sorting is achieved in the same way that we filter the list (use: `gSortBy`)
3. Add paging:

