Library Project - Preparing for Phase 2 (UI and jQuery Event Handling)

Getting Started

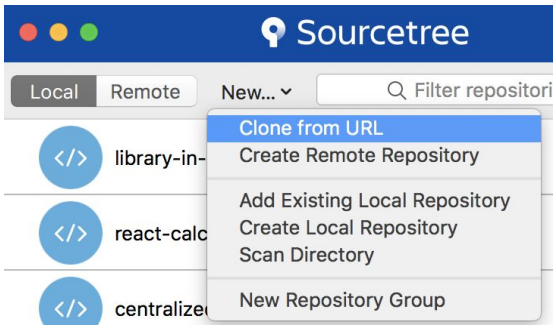
In Phase 1 of the Library Project, you built the engine that will drive your application using JavaScript. **Before you begin Phase 2, please commit and push your final copy of Phase 1 to your remote master if you have not already done so.** As we prepare to enter Phase 2, we will need to step back into the worlds of HTML, CSS, and Bootstrap to create the user-side interface for your application. Luckily for you, we have already taken care of this! This document will walk you through cloning the UI repository. We will also point out the purpose of the buttons, input fields, and modals included with the UI files. Afterward, we will explain precisely what you need to complete for this phase of the project.

Cloning The UI Repository

https://github.com/TechtonicAcademy/Academy-Library-UI

First you will navigate to the repository on GitHub listed above. Next you will need to clone the repository into a folder somewhere on your PC. You can do this through Sourcetree or command line, whichever you prefer.

Sourcetree



By going to the “New” dropdown menu and selecting “Clone from URL,” you then just need to provide a path to the folder you want to clone to and the URL to the repository **(https://github.com/TechtonicAcademy/Academy-Library-UI).** Then hit “Clone.”

Command Line

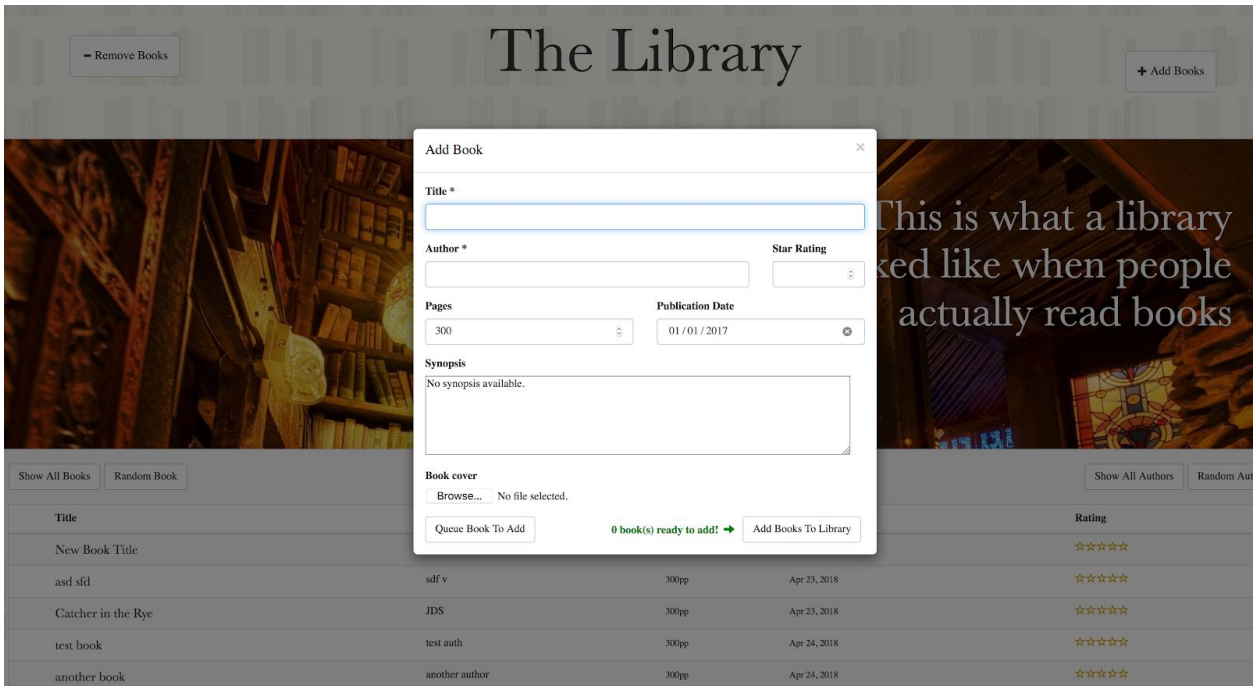
If you would rather use command line to clone the project, navigate to the folder you want to clone the UI repository into and type the following into the Terminal. Paste the URL, https://github.com/TechtonicAcademy/Academy-Library-UI, after the `git clone` command.

```
git clone URL HERE
```

What Has Been Cloned?

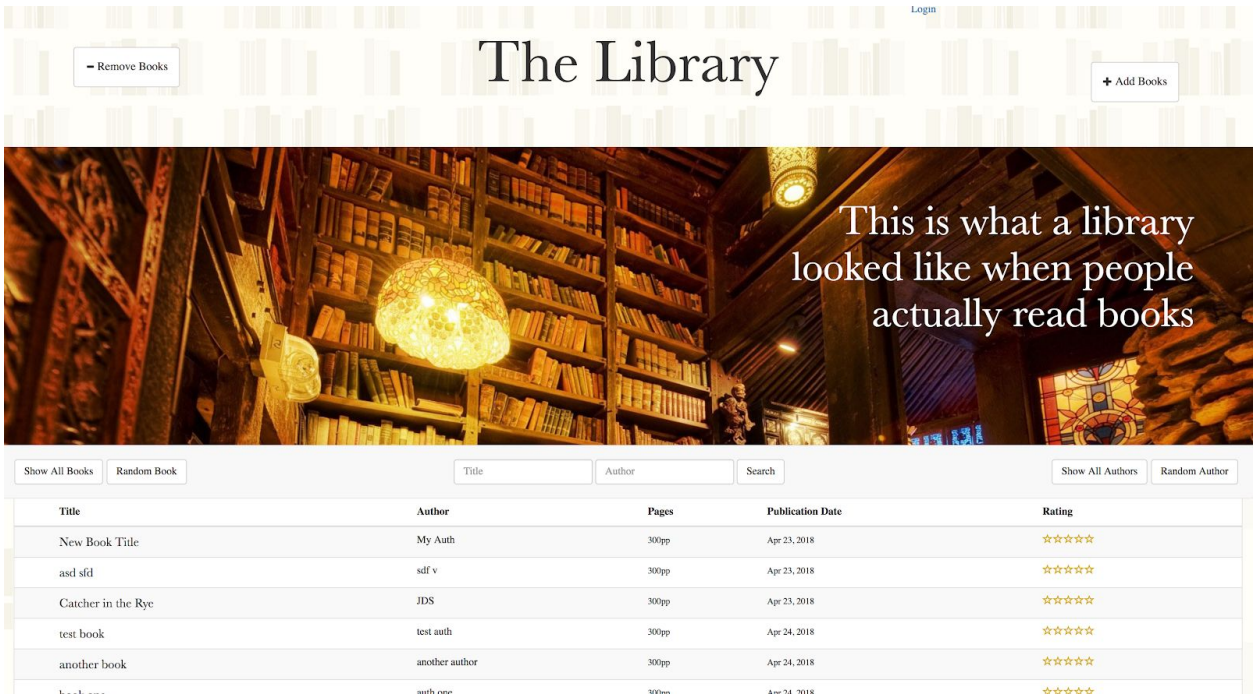
- The UI for our Library App uses the Bootstrap grid layout to design the page
    - https://getbootstrap.com/docs/3.3/examples/grid/
    - An Add Book modal (Figure 1): When this button is clicked it will allow users to add book objects.
    - This modal will open when the Add Book button is clicked.
    - The modal has 6 input fields and 1 button to upload a cover image file. These correspond to the book object properties (title, author, star rating, number of pages, publish date, synopsis, and a cover as a base64-encoded string)
        - The assets include book cover images in the correct format, but for further learning regarding base64 encoding, see below:
            - https://blogs.oracle.com/rammenon/base64-explained
            - https://www.base64-image.de/
    - Queue Book To Add button: When this button is clicked, the data from the form fields will be used to create a book object and then that book object will be added to an array of queued books and the modal form data will be cleared. Also the Book(s) Ready to Be Added Counter will be incremented.
    - Add Books To Library button: When this button is clicked the array of queued book objects should be added to the library bookshelf array and the queued book array will be emptied. Also the Book(s) Ready to Be Added Counter will be reset.

Figure 1:



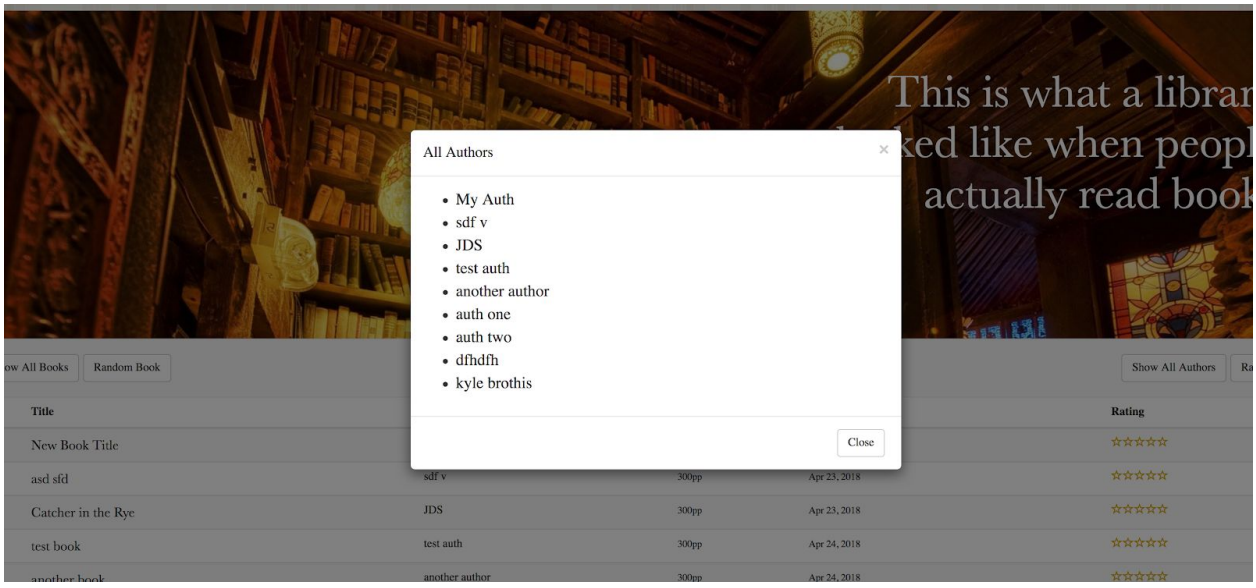
- The library UI includes a table (Figure 2) that displays each book's information
    - It consists of a header with labels for each column
    - Each row will have information for a single book object
        - A single column will be present for each of your book object's properties (cover, title, author, number of pages, publish date, rating, and delete)
        - A delete checkbox will also appear in the rightmost column of the table

Figure 2:



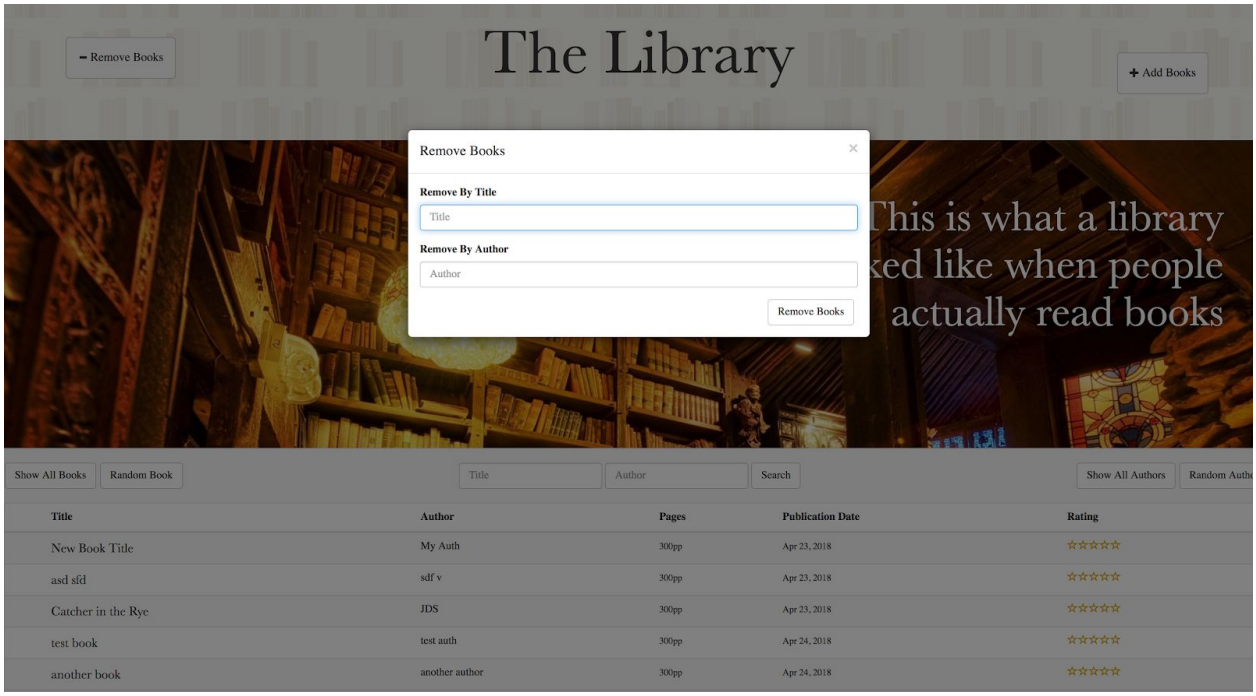
- The Show All Authors button will open a modal (Figure 3) that displays a list of unique (no repeats) authors from your bookshelf.

Figure 3:



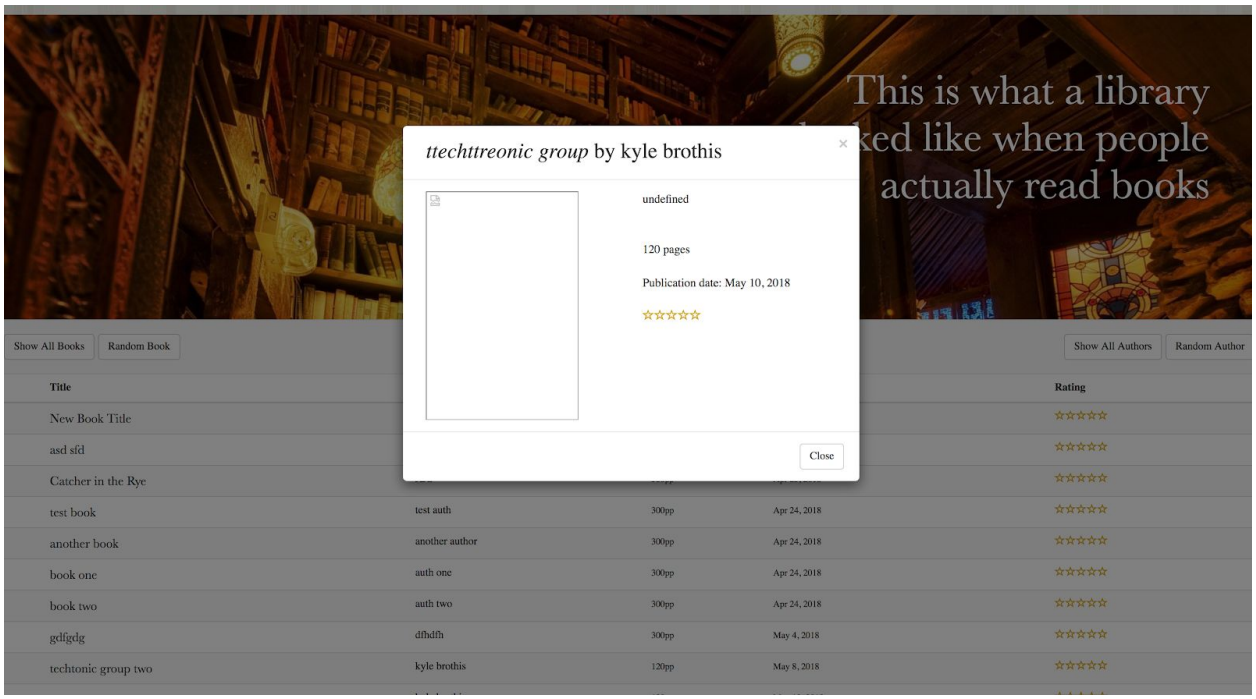
- The Remove Books button opens a modal (Figure 4) with an input field/button that will allow you to remove books by a single, unique author or title.

Figure 4:



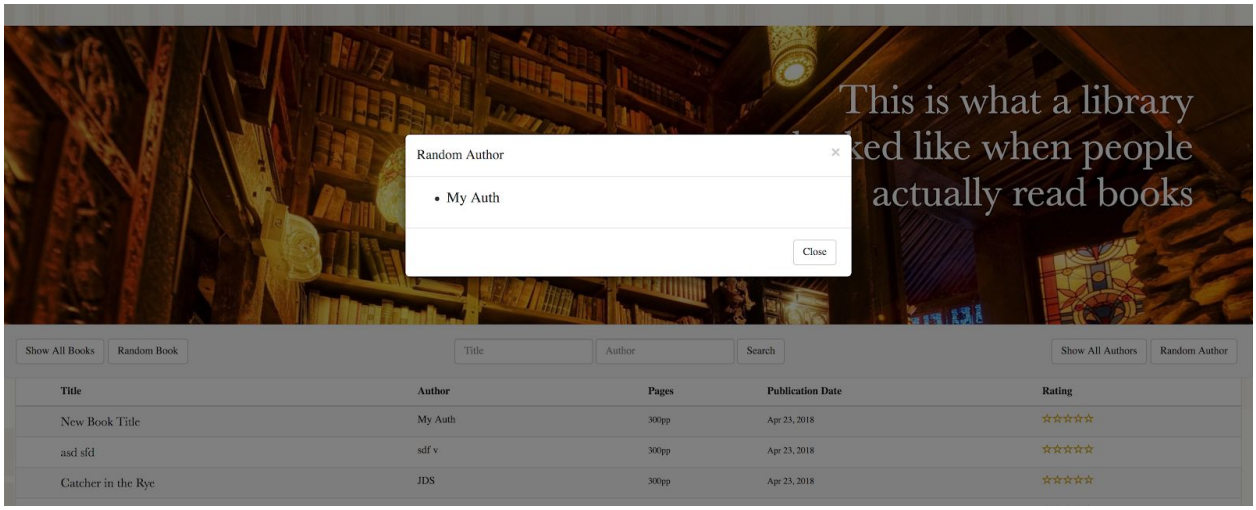
- The Random Book button will open a modal (Figure 5) that displays a random book from your bookshelf:
    - It will include all the data fields of your book, displayed as seen below:

Figure 5:



- The Random Author button will open a modal (Figure 6) displaying a single random author from your bookshelf with a display area for the result.

Figure 6:



- There is a title and author input field and button for searching through your library bookshelf contents for a specific title or author match just above the table.
    - If matches are found, the table should be updated to show only the matched books from your bookshelf.
- The Show All Books button will update your table to display all of the books on your library bookshelf when clicked. It will be used to refresh your table after a search.
- As you include script tags be sure to place them just before the </body> tag to match industry standards.

What to Do

1. Clone the repository
2. Get familiar with the UI. Identify ids, classes, and attributes that might be used to hook up functionality.
3. Use jQuery selectors and available or self-created ids, classes, and attributes to hook up functionality for each of the following modules described below. Instead of having you build your entire app in one file (main.js), we are having you create each separate module (a module is a self-contained, encapsulated chunk of code that achieves an objective) as its own file. Each of the necessary files has already been created for you. Use dataTable.js as an example for how to call and initialize each module. Hint: It should be fairly straightforward if you follow the example for the document ready and reset the context for each module, but you will have to target differing containers. You will have the following modules/files within your repo.

a. **book.js** - This module simply contains your Book class constructor. This is also where you will need to add in your editBook(book) object. You will need to refactor it to match your book constructor.

b. **util.js** - This module contains utility functions that will take care of ensuring that dates and table entries are well-formatted and that your added books appear as Book objects.

c. **engine.js** - This module is the engine you built in the console as Phase 1. It contains most of the methods you should need (including local storage setting and getting), but will require you to define the logic of the search functionality. You will need to think through the logic of setting up your robust search function and include it in this module. You will also need to consider refreshing the table through a click on the Show All Books button in the user interface.

d. **dataTable.js** - This module contains everything you will need for your table to display appropriately. It already initializes your bindEvents and bindCustomListeners, takes care of updating your table and local storage, fills in your star ratings, and creates the header columns and rows for your tables' data. It has spaces for your bindEvents and bindCustomListeners. You will need to add to these sections with the appropriate jQuery calls to hook in your functionality using methods like .on('click') events, $.proxy, and eventHandlers. You may also have to use delegation and preventDefault here. This will be where you need to set up the UI functionality of the search button by connecting it with the search logic you created in the engine.js module. This will also be the appropriate place to set up a function to display all books or refresh your table upon clicking the Show All Books button.

e. **addBooksModal.js** - This module contains the code for the Add Books modal. You will need to handle queueing, incrementing the queued books counter, adding books to your bookshelf, and clearing your queue. There is already a helper function here to help your images upload correctly.

f. **removeBooksModal.js** - This is where you will leverage what you've already created to remove books by title and author.

g. **showAuthorsModal.js** - This is where you will create an unordered list of all unique authors to be displayed in the modal.

h. **randomAuthorModal.js** - This is where you will display the name of a single random author in the modal.

i. **suggestBookModal.js** - This is where you will display the cover image and other book data for a single book within the modal.

Helpful Materials

-https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Building_blocks/Events
-https://api.jquery.com/serializeArray/
-https://oscarotero.com/jquery/
-https://medium.com/intrinsic/proxies-and-reflection-in-javascript-334412028f69
-https://bootstrapcreative.com/resources/bootstrap-4-css-classes-index/
-https://vickylai.com/verbose/iterating-over-objects-and-arrays-frequent-errors/
-https://medium.freecodecamp.org/object-oriented-programming-concepts-21bb035f7260
-https://codeburst.io/using-html5-web-storage-a450294484bb
-https://javascript.info/object