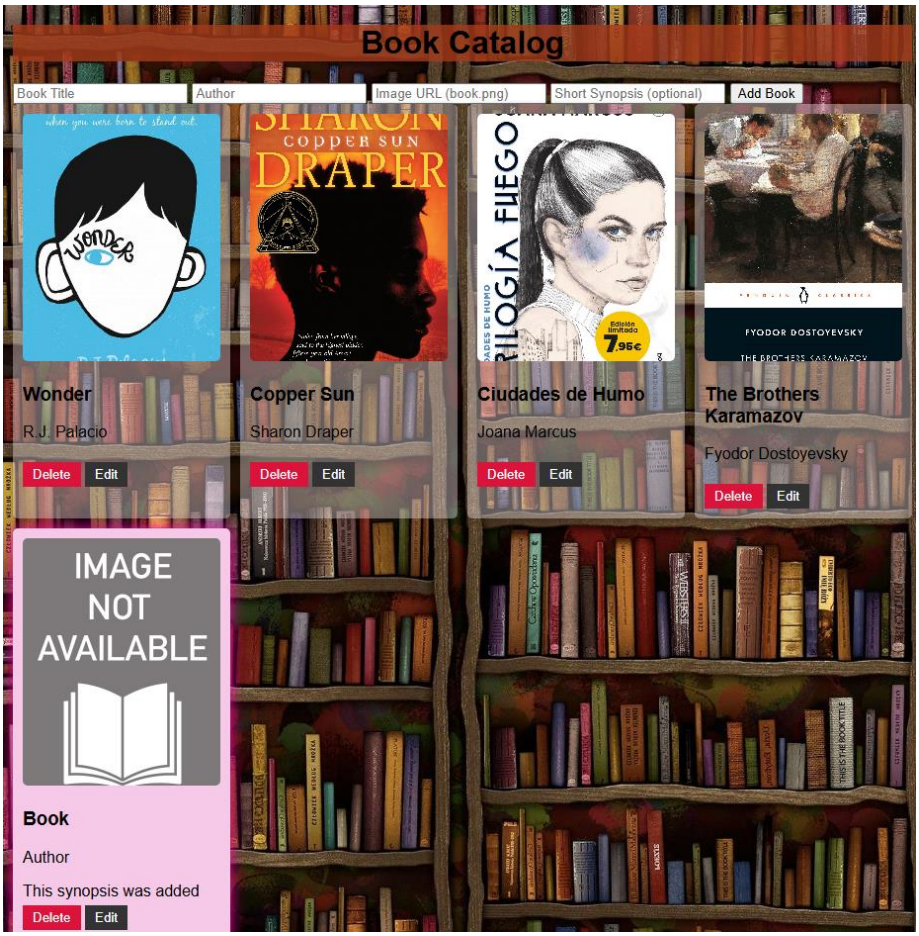
Pages Link: https://elabrapurdue.github.io/CRUD_Project/

Repo: elabrapurdue/CRUD_Project

Note:

To add an image (optional just like Synopsis), right-click the image you want, select “View Image” or “Open Image in New Tab,” then copy the image URL in this format: “https://...jpg”.

Be sure all project files are saved in the same folder. Open that folder in VS Code, right-click the HTML file, and select “Open with Live Server” so the webpage functions properly. (Exception with GitHub Pages link since it works without Live Server).



Documentation:

We collect user input from form fields using jQuery.
If no image is provided, we set a default.

We push the new book object into the books array and call renderBooks() to refresh the display.

```
$("#add-book").click(function () {
    const title = $("#book-title").val().trim();
    const author = $("#book-author").val().trim();
    let image = $("#book-image").val().trim();
    let synopsis = $("#book-synopsis").val().trim();

    if (!title || !author) {
        alert("Please enter both a book title and an author.");
        return;
    }
    if (!image) { image = "book.png"; }
    if (!synopsis) { synopsis = "No synopsis available."; }

    books.push({ id: nextId++, title, author, image, synopsis });
    renderBooks();
    highlightLastCard();
});
```

We loop through the books array and dynamically create HTML elements using jQuery, inserting them into the page.

```
function renderBooks() {
    $("#book-list").empty();
    books.forEach(book => {
        $("#book-list").append(`
            <div class="book-card" data-id="${book.id}">
                <img src="${book.image}" alt="${book.title}">
                <h3>${book.title}</h3>
```

```
                <p>${book.author}</p>
                <div class="synopsis" style="display:none;">${book.synopsis ||
"No synopsis available."}</div>
                <button class="delete-btn">Delete</button>
                <button class="edit-btn">Edit</button>
            </div>
        `);
    });
}
```

When the edit button is clicked, we use prompt() to get new values and update the matching object in the array.

```
$(document).on("click", ".edit-btn", function () {
    const card = $(this).parent();
    const id = parseInt(card.attr("data-id"));
    const currentTitle = card.find("h3").text();
    const currentAuthor = card.find("p").text();
    const currentImage = card.find("img").attr("src");

    const newTitle = prompt("Edit title:", currentTitle);
    const newAuthor = prompt("Edit author:", currentAuthor);
    const newImage = prompt("Edit image URL:", currentImage);

    if (newTitle && newAuthor && newImage) {
        updateBook(id, newTitle, newAuthor, newImage);
    }
});
```

Clicking the delete button removes the book from the array using filter() and re-renders the list.

```
$(document).on("click", ".delete-btn", function () {
    const id = parseInt($(this).parent().attr("data-id"));
```

```
    $(this).parent().fadeOut(300, function () {
        deleteBook(id);
    });
});
```

- JSON Lists: books.json + loaded via $.getJSON()
- Content to Page: Uses renderBooks() with jQuery to refresh the DOM
- CREATE: User adds a book through form inputs, array updates, and page refreshes
- READ: Click on a book card shows its synopsis detail
- UPDATE: Edit button updates title, author, and image using prompt()
- DELETE: Delete button removes the book with a fade-out effect
- UI Design: Custom background, hover effects, colors
- Effects: Highlight on create, fade-out on delete, card highlight on click