Laboratory Activity 2: Populating Pages

Part 1.

```php
FinalPro > app > Http > Controllers >  LandingPageController.php
1    <?php
2
3    namespace App\Http\Controllers;
4
5    use Illuminate\Http\Request;
6
     Codeium: Refactor | Explain
7    class LandingPageController extends Controller
8    {
         Codeium: Refactor | Explain | Generate Function Comment | X
9        public function index()
10       {
11           // Simulated featured books data
12           $featuredBooks = [];
13
14           // Generate 12 books dynamically
15           for ($i = 1; $i <= 12; $i++) {
16               $featuredBooks[] = [
17                   'id' => $i, // Unique ID for the book
18                   'title' => "Book Title $i",
19                   'image' => "assets/img/cover$i.jpg", // Assuming different images for each book
20                   'rating' => 0, // Default to 0 for unrated books
21                   'description' => "Description for Book $i. Nullam id dolor id nibh ultricies vehicula ut id elit. Cras justo odio.",
22               ];
23           }
24
25           return view('LandingPage', compact('featuredBooks'));
26       }
27   }
28
```

The LandingPageController class extends the Controller class and defines a single method *index()*.
The *index()* method generates an array of 12 books with unique IDs, titles, images, ratings, and descriptions. Each book's data is stored in an associative array and added to the $featuredBooks array.
Finally, the method returns a view called 'LandingPage' and passes the $featuredBooks array as a variable to the view.

```php
20
21    Route::get('/', [LandingPageController::class, 'index'])->name('landing');
22
```

The landing page route is a simple route that displays the landing page view.
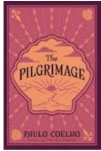The route is named 'landing' so that it can be easily referenced in code.
The route is handled by the index method of the *LandingPageController* class.

Part 2:

```
Codelum: Refactor | Explain | Generate Function Comment | X
 9      public function index()
10      {
11          // Simulated featured books data
12          $featuredBooks = [];
13
14          // Generate 12 books dynamically
15          for ($i = 1; $i <= 12; $i++) {
16              $featuredBooks[] = [
17                  'id' => $i, // Unique ID for the book
18                  'title' => "Book Title $i",
19                  'image' => "assets/img/cover$i.jpg", // Assuming different images for each book
20                  'rating' => 0, // Default to 0 for unrated books
21                  'description' => "Description for Book $i. Nullam id dolor id nibh ultricies vehicula ut id elit. Cras justo odio.",
22              ];
23          }
24
25          return view('LandingPage', compact('featuredBooks'));
26      }
27  }
```

This method inside the controller dynamically generates a list of **featured books** and passes it to the `LandingPage` view. Below is the content created by the controller.
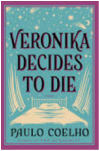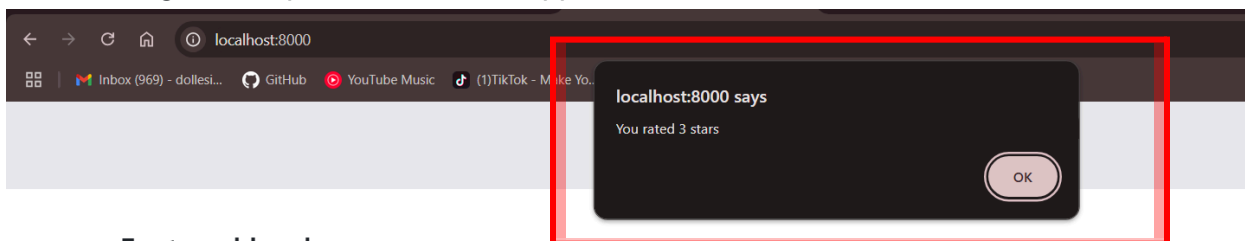
```
FinalPro > public > assets > bootstrap > js > JS star-rating.js > 🔘 document.addEventListener("DOMContentLoaded") callback > 🔘 ratingContainers.forEach() callback > 🔘 stars.forEach() callback
 1    document.addEventListener("DOMContentLoaded", function () {
 7
 8        // Initialize each rating container
 9        ratingContainers.forEach((container) => {
10            const bookId = container.getAttribute("data-book-id");
11            const stars = container.querySelectorAll(".fa-star");
12
13            // Set initial state based on saved ratings
14            const initialRating = savedRatings[bookId] || 0;
15            for (let i = 0; i < initialRating; i++) {
16                stars[i].style.color = "var(--bs-yellow)"; // Yellow for active stars
17            }
18
19            // Add click event listeners to stars
20            stars.forEach((star, index) => {
21                star.addEventListener("click", () => {
22                    // Set the rating
23                    const rating = index + 1;
24
25                    // Update the stars visually
26                    stars.forEach((s, i) => {
27                        if (i < rating) {
28                            s.style.color = "var(--bs-yellow)"; // Highlight active stars in yellow
29                        } else {
30                            s.style.color = "rgb(174, 174, 174)"; // Set unselected stars to gray
31                        }
32                    });
33
34                    // Show alert with the rating
35                    alert(`You rated ${rating} star${rating > 1 ? 's' : ''}`);
36
37                    // Save the rating in localStorage
38                    savedRatings[bookId] = rating;
39                    localStorage.setItem("bookRatings", JSON.stringify(savedRatings));
40                });
41            });
```

This JavaScript code adds interactivity to a **star rating system** for books. When the user clicks the star, an alert message appears.

After clicking OK, it updates the visual appearance of the stars.

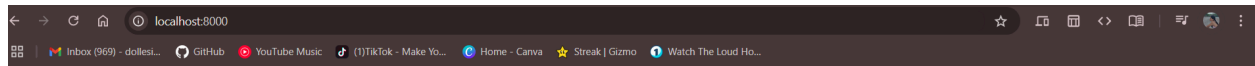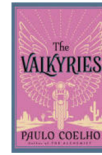# Featured books

view all

Book Title 1
★★★☆☆

Description for Book 1. Nullam id dolor id nibh ultricies vehicula ut id elit. Cras justo odio.

Book Title 2
☆☆☆☆☆

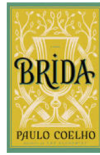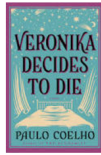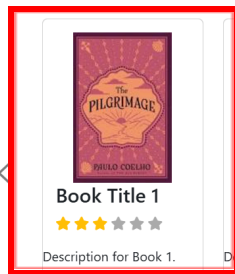Description for Book 2. Nullam id dolor id nibh ultricies vehicula ut id elit. Cras justo odio.
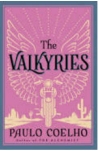
Book Title 3
☆☆☆☆☆

Description for Book 3. Nullam id dolor id nibh ultricies vehicula ut id elit. Cras justo odio.

Book Title 4
☆☆☆☆☆

Description for Book 4. Nullam id dolor id nibh ultricies vehicula ut id elit. Cras justo odio.

Book Title 5
☆☆☆☆☆

Description for Book 5. Nullam id dolor id nibh ultricies vehicula ut id elit. Cras justo odio.

Book Title 6
☆☆☆☆☆

Description for Book 6. Nullam id dolor id nibh ultricies vehicula ut id elit. Cras justo odio.

Subjects Sections

Most viewed