

1. Take screenshots of controllers, [routes](#), and rendered pages.

```
app > Http > Controllers > DashboardController.php
1  <?php
2
3  // app/Http/Controllers/DashboardController.php
4
5  namespace App\Http\Controllers;
6
7  use Illuminate\Http\Request;
8  use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
9  use Illuminate\Support\Facades\Auth;
10 use App\Models\User;
11
12
13 class DashboardController extends Controller
14 {
15     public function index()
16     {
17         return view('dashboard');
18     }
19
20     public function showBook($book_id)
21     {
22         // Dummy book data
23         $books = [
24             1 => [
25                 'title' => 'The Great Gatsby',
26                 'author' => 'F. Scott Fitzgerald',
27                 'review' => 'A fascinating story of the Jazz Age.',
28                 'rating' => 5
29             ],
30             2 => [
31                 'title' => '1984',
32                 'author' => 'George Orwell',
33                 'review' => 'A dystopian novel about totalitarianism.',
34                 'rating' => 4
35             ]
36         ];
37
38         // Check if the book exists in the dummy data
39         if (!isset($books[$book_id])) {
40             abort(404, 'Book not found');
41         }
42
43         // Return the view with the book details
44         return view('book', ['book' => $books[$book_id]]);
45     }
46
47
48 }
```

- o The DashboardController manages views and logic related to the dashboard and book details.

- The index method renders the dashboard view, which is the main page for authenticated users.

- The showBook method takes a book\_id parameter, looks for the corresponding book in a dummy dataset, and returns the book view with the book's details if found. If the book\_id is invalid, it returns a 404 error page.

```
app > Http > Controllers > HomeController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class HomeController extends Controller
8  {
9      public function index ()
10     {
11         return view ('landing-page');
12     }
13 }
14
```

- The HomeController is responsible for rendering the landing page.

- o The index method simply returns the landing-page view, which is the main entry point for users visiting the site.

```

routes > web.php
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4  use App\Http\Controllers\HomeController;
5  use App\Http\Controllers\DashboardController;
6  use App\Http\Controllers\AuthController;
7
8  // Landing Page
9  Route::get('/', [HomeController::class, 'index'])->name('landing-page');
10
11
12 // Authentication Routes (only for guests)
13 Route::middleware('guest')->group(function() {
14     Route::get("/login", [AuthController::class, 'login'])->name('login');
15     Route::post("/login", [AuthController::class, "loginPost"]->name("login.post");
16     Route::get("/register", [AuthController::class, "register"]->name("register");
17     Route::post("/register", [AuthController::class, "registerPost"]->name("register.post");
18 });
19
20 // Authenticated Routes (for logged-in users)
21 Route::middleware('auth')->group(function() {
22     Route::get('/contact', function () {
23         return view('contact');
24     });
25
26     Route::get('/home', function () {
27         return view('home');
28     });
29     Route::get('/dashboard', action: [DashboardController::class, 'index']->name('dashboard.show');
30
31     Route::get('/dashboard/book/{book_id}', [DashboardController::class, 'showBook']);
32
33     Route::post('/logout', function () { // Logout Route
34         Auth::logout(); // Logs out the user
35         return redirect(route("landing-page"));
36     })->name('logout');
37
38 });

```

### Routes Explanation

- Landing Page Route
  - Route: /
  - Handler: HomeController@index
  - Description: Displays the landing page for all users, regardless of authentication status.

### Authentication Routes (Guest-Only)

- Middleware: guest
- Routes:
  - 1) /login (GET): Displays the login form (AuthController@login).
  - 2) /login (POST): Processes the login data (AuthController@loginPost).
  - 3) /register (GET): Displays the registration form (AuthController@register).
  - 4) /register (POST): Processes the registration data (AuthController@registerPost).

Description: These routes ensure only unauthenticated users can access login and registration functionalities.

### Authenticated Routes (Logged-In Users Only)

- Middleware: auth
- Routes:
  - 1) /contact: Returns a contact view.
  - 2) /home: Returns a home view.
  - 3) /dashboard (GET): Displays the dashboard page (DashboardController@index).
  - 4) /dashboard/book/{book\_id} (GET): Displays details for a specific book based on the book\_id parameter (DashboardController@showBook).
  - 5) /logout (POST): Logs out the user and redirects to the landing page.
- Description: These routes are restricted to authenticated users, ensuring protected content is only accessible after logging in.

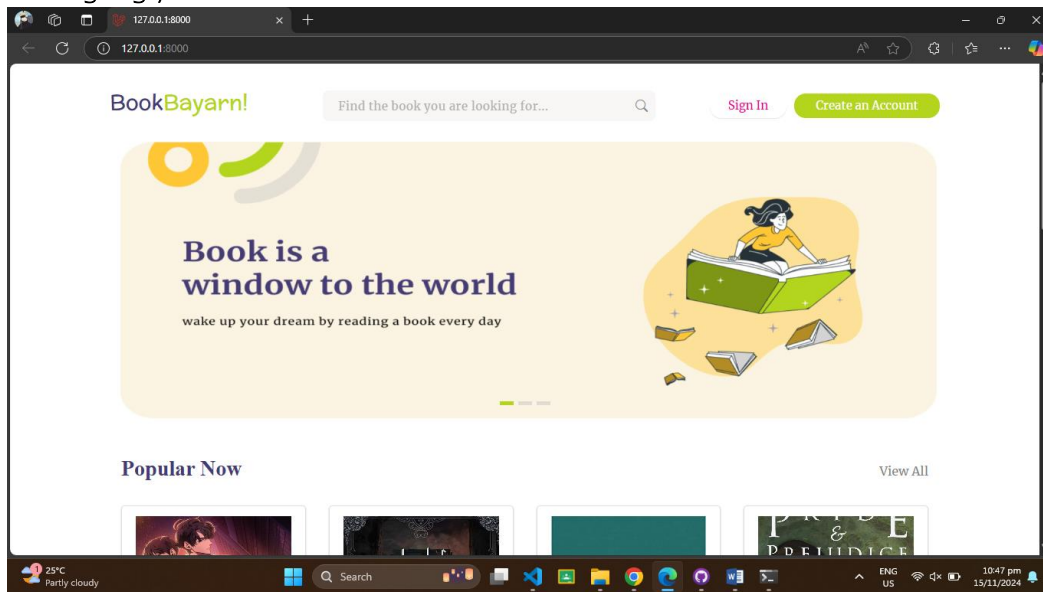
### Special Route Features

- 1) Named Routes: Provides identifiers like landing-page, login, and dashboard.show for easier redirection in the application.
- 2) Dynamic Route Parameters: The /dashboard/book/{book\_id} route uses a placeholder {book\_id} to dynamically fetch and display specific book details.
- 3) Middleware: Ensures users are either guests or authenticated before accessing certain routes.

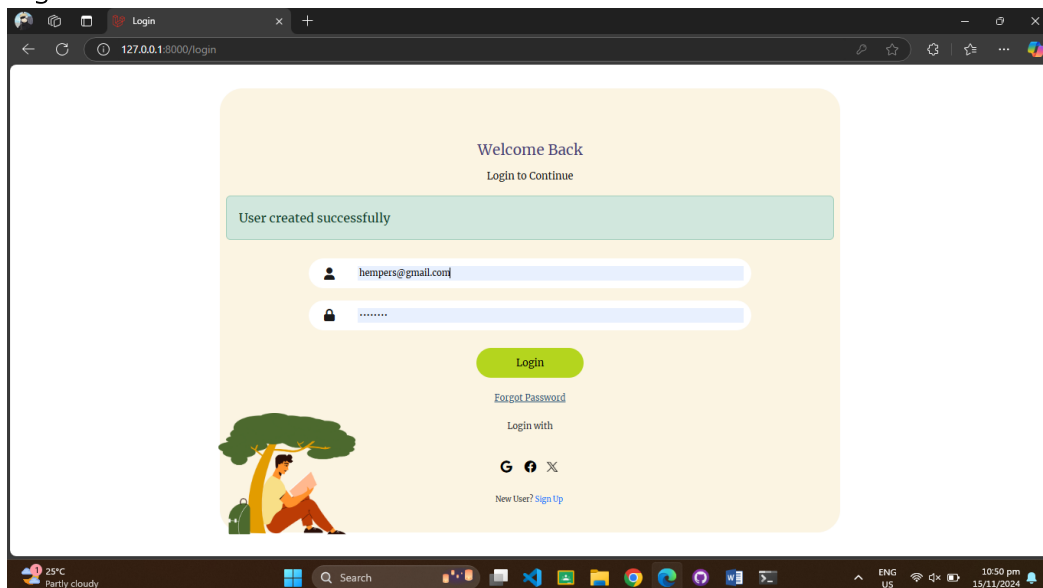
### Rendered Pages:

(The user interface may still undergo changes as it has not been finalized yet.)

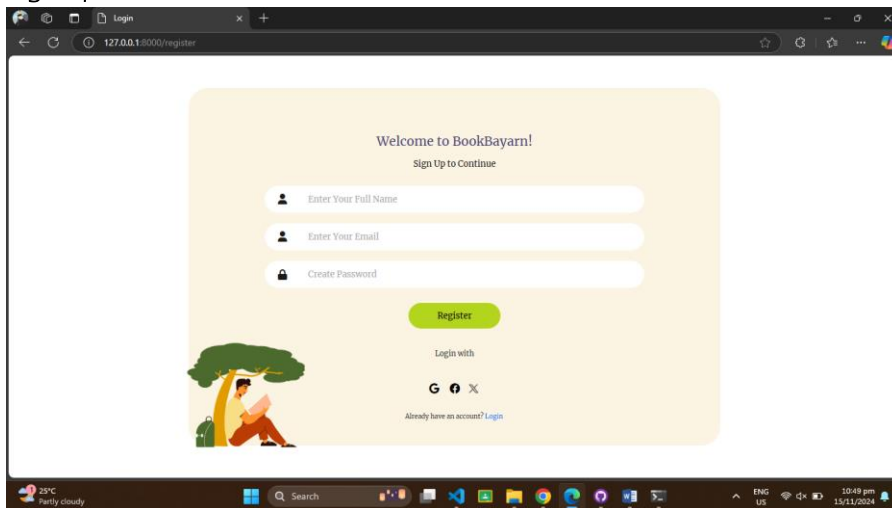
- Landing Page/Home:



- Login:

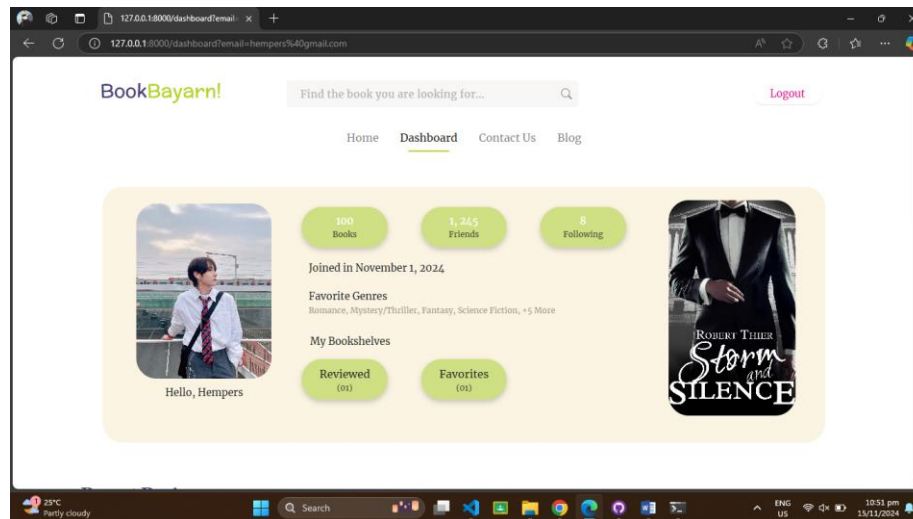


- Sign Up



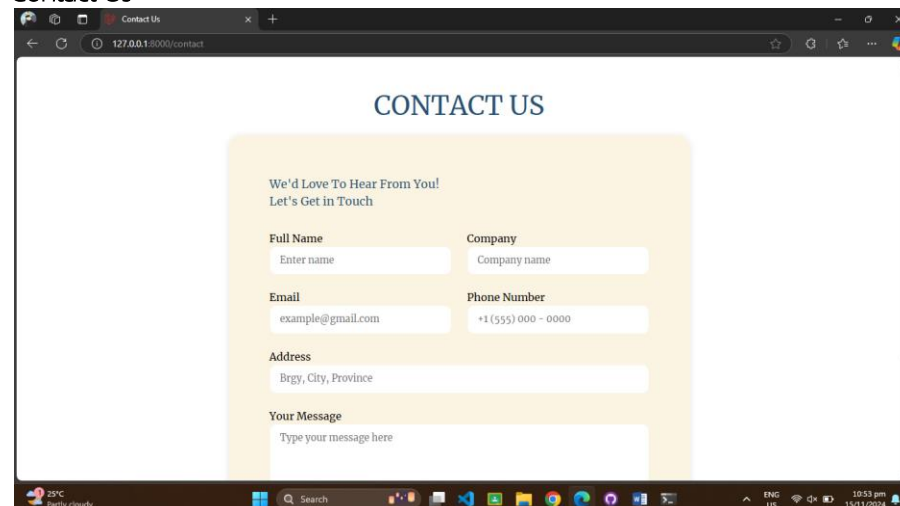
A screenshot of a web browser showing the 'Sign Up' page for 'BookBayarn!'. The page has a light yellow background with a central white box containing the registration form. The form includes fields for 'Enter Your Full Name', 'Enter Your Email', and 'Create Password', each with a corresponding icon (person, envelope, and lock). Below these fields is a green 'Register' button. Underneath the button is a 'Login with' section featuring Google and Facebook icons, and a link for 'Already have an account? Login'. To the left of the form is a small illustration of a person reading a book under a tree. The browser's address bar shows '127.0.0.1:8000/register'. The Windows taskbar at the bottom shows the date as 15/11/2024 and the time as 10:49 pm.

- Dashboard



A screenshot of the 'Dashboard' page for a user named 'Hempers' on the 'BookBayarn!' website. The page features a navigation bar with 'Home', 'Dashboard' (highlighted), 'Contact Us', and 'Blog'. A search bar and a 'Logout' link are also present. The main content area displays a user profile with a profile picture of a person in a school uniform, the name 'Hello, Hempers', and statistics: '100 Books', '1,245 Friends', and '8 Following'. It also shows the user joined in November 1, 2024, and lists favorite genres: Romance, Mystery/Thriller, Fantasy, Science Fiction, and +5 More. Below this, there are sections for 'My Bookshelves' with 'Reviewed (01)' and 'Favorites (01)' buttons, and a featured book cover for 'Storm and Silence' by Robert Tiller. The browser's address bar shows '127.0.0.1:8000/dashboard?email=hempers%40gmail.com'. The Windows taskbar at the bottom shows the date as 15/11/2024 and the time as 10:51 pm.

- Contact Us



A screenshot of the 'Contact Us' page for 'BookBayarn!'. The page has a light yellow background with a central white box containing the contact form. The form is titled 'CONTACT US' and includes the text 'We'd Love To Hear From You! Let's Get in Touch'. The form fields are arranged in two columns: 'Full Name' (with a sub-field 'Enter name') and 'Company' (with a sub-field 'Company name') in the top row; 'Email' (with a sub-field 'example@gmail.com') and 'Phone Number' (with a sub-field '+1 (555) 000 - 0000') in the second row; and a single 'Address' field (with a sub-field 'Brgy, City, Province') in the third row. Below these fields is a 'Your Message' section with a text area labeled 'Type your message here'. The browser's address bar shows '127.0.0.1:8000/contact'. The Windows taskbar at the bottom shows the date as 15/11/2024 and the time as 10:53 pm.

2. Write brief explanations of controller logic, parameter handling, and route assignments.

- The `HomeController` is responsible for rendering the landing page, with its `index` method returning the `landing-page` view, which serves as the entry point for all users. The `DashboardController` manages the logic for authenticated user pages. Its `index` method renders the `dashboard` view, while the `showBook` method handles dynamic parameters by accepting a `book_id`, checking it against a predefined dataset, and returning a `book` view with the corresponding details. If an invalid `book_id` is provided, it returns a 404 error page.
- Routes are categorized into three groups: public, guest-only, and authenticated routes. The public route `/` invokes the `HomeController@index` method to display the landing page. Guest-only routes, protected by the `guest` middleware, include `/login` and `/register` for handling login and registration through `AuthController` methods. Authenticated routes, protected by the `auth` middleware, include `/dashboard`, which uses `DashboardController@index` to render the dashboard, and `/dashboard/book/{book_id}`, which passes the `book_id` parameter to `DashboardController@showBook` for displaying specific book details. Additional authenticated routes include `/contact` and `/home` for rendering static views, and `/logout`, which logs out the user and redirects them to the landing page. These route assignments ensure proper user access control and dynamic content handling.