**Books Collection Application**

### Document Properties

| Item | Details |
| --- | --- |
| Document | Architecture Design Document |
| Application | Books Collection application |
| Author | Anant Nerkar |
| Version | V1.0 |
| Creation Date | 06/09/2020 |
| Last Updated | 07/09/2020 |
| Status | Completed |

Contents

[Document Properties 1](#_Toc50375247)

[**1.** **Introduction** 3](#_Toc50375248)

[**2.** **Use-Case View** 3](#_Toc50375249)

[**3.** **Flow Chart** 4](#_Toc50375250)

[**4.** **Components** 5](#_Toc50375251)

[**5.** **Services** 5](#_Toc50375252)

[**6.** **Validations** 5](#_Toc50375253)

[**7.** **Tools and Libraries** 5](#_Toc50375254)

[**8.** **Unit Testing** 5](#_Toc50375255)

[**9.** **Responsive** 6](#_Toc50375256)

[**10.** **Accessibility** 6](#_Toc50375257)

[**11.** **List other enhancement (at least top 3)** 6](#_Toc50375258)

[**12.** **Setup** 7](#_Toc50375259)

# **Introduction**

In this document, we can see the overview for the application which covers its basic flow and scenarios of the Books collection application.

It is mainly developed for managing Books with performing CRUD operations.

Also, there is a login and sign up functionality provided for the user.

# **Use-Case View**

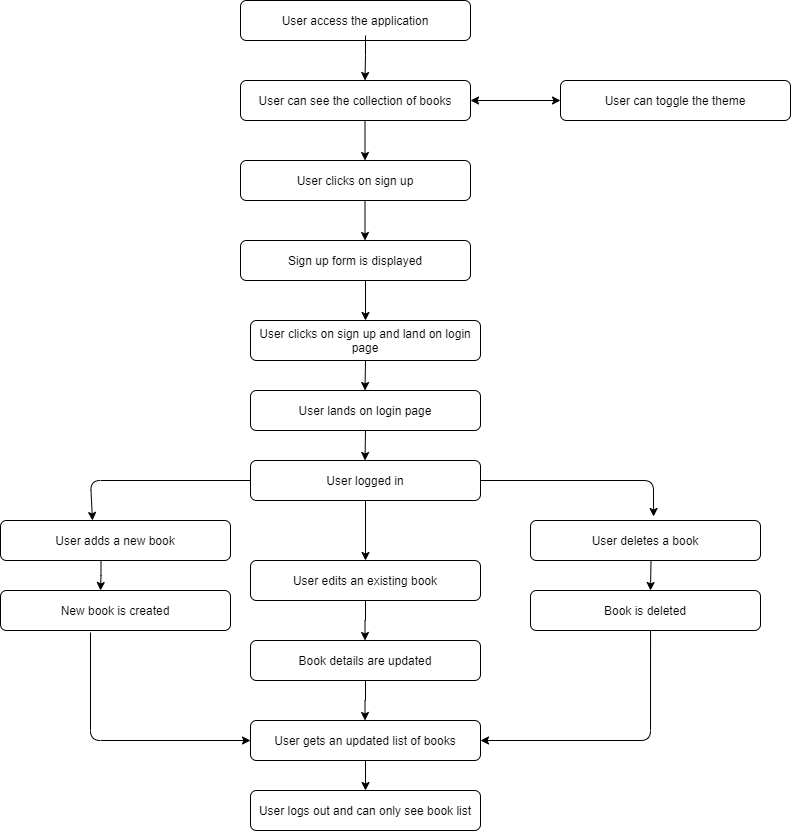
The application use cases for which user authentication is mandatory are -

* **Add a Book** – The application provides the functionality for adding a new Book after user login.
* **Update a Book** – This functionality provides the facility to update the existing Book for the user.
* **Delete a Book –** The user has an option to delete the Book.

Use cases for which the user authentication is not mandatory.

* **List all Books** -The user has the option to view all the Books and their description.
* **Login**– The user can login into the application with the credentials.
* **Sign up** – If the user wants to modify any of the Book or create a new Book, user has to sign up, and then login to perform the CRUD operations.

# **Flow Chart**



# **Components**

* AppComponent – This component is used for Application Bootstrapping
* HomeComponent – This is the home page of the application
* BookListComponent – This is the component used to display the list of the Books and from here the user can perform the actions related to the Books.

From this component the user has an option to delete the Book.

* SignUpComponent – This component manages the signing up and login flow of the user
* CreateBookComponent – This component is used to manage the edit/create flow of the Book.

# **Services**

1. AuthService – This service takes care of the API requests for authentication of the user like login and sign up.

2. BooksService – This service takes care of the API requests for CRUD operations of the Book.

# **Validations**

1)When an unregistered user tries to login, the application does not allow it to do so, first user needs to register/sign up then he can login.

2) Login and Sign up buttons are disabled until user enters all the fields.

**Validation strategy:**

- Used Reactive form for validation

- Used window.alerts and window.confirm for showing error messages from API and for confirmation response.

# **Tools and Libraries**

- Angular version: 10.0

- TypesScript: 3.9

- Rxjs: 6.5

- Jest: 26.4.2

This is a unit testing framework; it consists of tools/scripts to run the test cases.

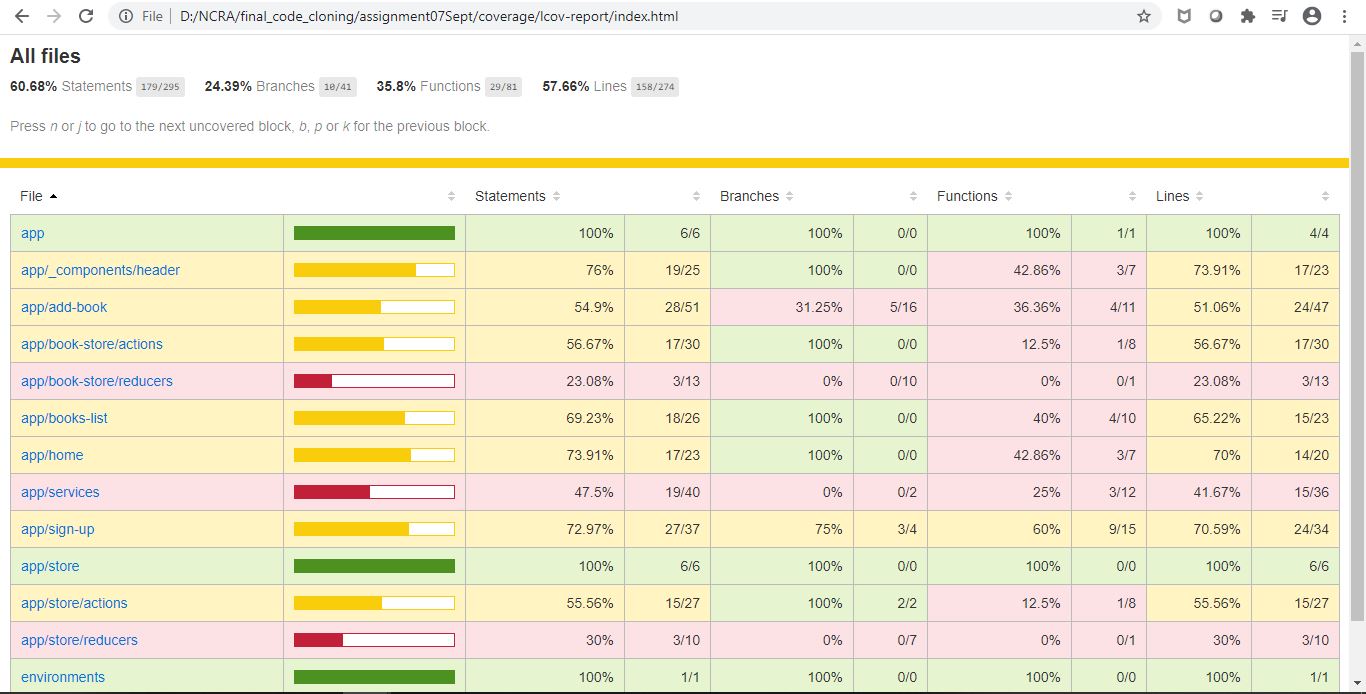
# **Unit Testing**

Added Unit Test cases for services.

It consists of two services -

* + Books.service.spec.ts – It contain test cases for the all the API calls for CRUD operations for Books.
  + Auth.service.ts - It contain test cases for API calls for the authentication.

Please find an attached test coverage report.



# **Responsive**

* Created responsive table which grows/shrinks in width according to the device.

# **Accessibility**

For Accessibility, the HTML markup contains aria-labels for buttons and tab indexes for input fields and table data.

# **List other enhancement (at least top 3)**

* **Feature enhancements** -

1. Provided theme toggling for switching the theme.
2. User data management.
3. Enhance application with user management and more accessibility.
4. Used json server for data management via fake API calls

* **Other enhancements** –

1. Server-side rendering support.
2. Client-side caching features can be enhanced.
3. More role-based operations can be performed.
4. Localization can be performed in application.

# **Setup**

1. Download and install node.js (version: 12.16)
2. Install npm and install @angular (version: 10.0)
3. Git clone https://github.com/anantnerkar/tavisca-product-listing.git origin master
4. Navigate to cloned folder & run npm install
5. Run ng serve
6. For testing use: npm run test