

EXPERIMENT NO. 8

Name of Student	<u>Shreyash Ganesh Dhekane</u>
Class Roll No	<u>16</u>
D.O.P.	<u>01-04-2025</u>
D.O.S.	<u>08-04-2025</u>
Sign and Grade	

AIM: To study AngularJS

PROBLEM STATEMENT:

- a) Demonstrate with an AngularJS code one way data binding and two-way data binding in AngularJS
- b) Implement a basic authentication system for a web application using AngularJS. Create a simple login page that takes a username and password, and upon submission, checks for a hardcoded set of credentials. If the credentials are valid, display a success message; otherwise, show an error message. Demonstrate AngularJS controller, module and form directives.
- c) Users want to search for books by title, author, or genre. To accomplish this, develop an AngularJS custom filter named bookFilter and include it into the application.
- d) Create a reusable and modular custom AngularJS service to handle user authentication. Include this service into an application.

THEORY:

- 1) What are directives? Name some of the most commonly used directives in AngularJS application

Directives in AngularJS are **special markers** (attributes, elements, or classes) that extend **HTML functionality** by attaching custom behaviors to DOM elements.

They enable **dynamic content manipulation** and are essential in AngularJS applications for building reusable components.

Commonly used Directives in Angular JS

- **ng-app** – Defines the root element of an AngularJS application.
- **ng-model** – Binds an input field to a variable in the scope (two-way data binding).
- **ng-repeat** – Loops through an array to display dynamic lists.
- **ng-if** – Conditionally renders elements based on an expression.
- **ng-show / ng-hide** – Shows or hides elements based on a condition.
- **ng-click** – Adds a click event listener to elements.

- 2) What is data binding in AngularJS?

Data binding in AngularJS is the **automatic synchronization** of data between the **model (JavaScript variables)** and the **view (HTML UI elements)**. It helps in building **dynamic applications** without manually manipulating the DOM.

Types of Data Binding in AngularJS

1. One-Way Data Binding (Interpolation & Expressions)

- Updates the view when the model changes but **not vice versa**.
- Achieved using `{{ expression }}` (interpolation) or directives like `ng-bind`.

```
<p>Hello, {{ username }}!</p>  
<p ng-bind="username"></p>
```

2. Two-Way Data Binding (ng-model)

- Synchronizes data **both ways**—when the user updates the UI, the model updates, and vice versa.

```
<input type="text" ng-model="username">  
<p>Your name: {{ username }}</p>
```

Data binding in AngularJS makes the application **responsive and interactive** by **automatically updating** the UI when the data changes, reducing the need for manual DOM manipulation

3. How is form validation done in angularJS

AngularJS provides **built-in form validation** using directives and the ng-model directive to track user inputs. It helps ensure **data correctness** before submission.

Key Features of Form Validation in AngularJS

1. **Uses AngularJS directives** like ng-required, ng-minlength, ng-pattern, etc.
2. **Real-time validation** – Errors appear as users type.
3. **Built-in validation states** – \$valid, \$invalid, \$dirty, \$pristine track form status.
4. **Custom validation** – Developers can define custom validation rules.

AngularJS **simplifies form validation** with built-in directives, real-time error handling, and easy tracking of form states. This ensures **better user experience** and **data integrity**.

4. What is the use of AngularJS Controllers in the application?

In AngularJS, **controllers** are used to manage the **application logic** and **data**. They act as an interface between the **view (HTML)** and the **model (data)**, making applications **dynamic and interactive**.

Key Uses of AngularJS Controllers

1. **Data Binding** – Controls how data is displayed in the view using \$scope.
2. **Business Logic** – Defines functions to handle user actions and process data.
3. **Communication with Services** – Calls APIs or services to fetch/update data.
4. **Event Handling** – Manages user interactions like button clicks and form submissions.
5. **Separation of Concerns** – Keeps business logic separate from the view (HTML).

5. What is the use of AngularJS Filters in the application?

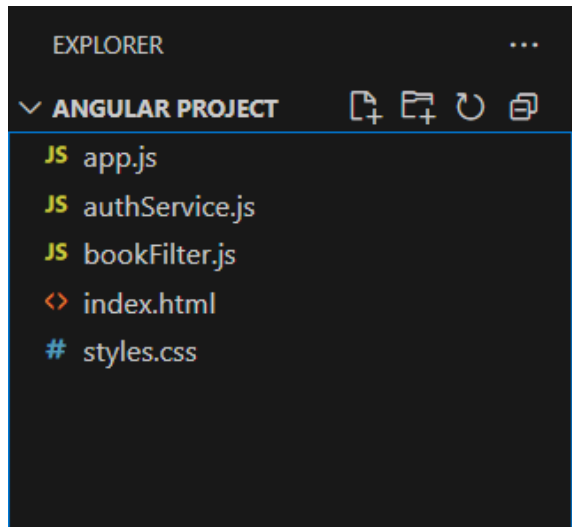
In AngularJS, **filters** are used to **format, modify, or transform** data before displaying it in the view. They help in presenting data in a **more readable and user-friendly format** without changing the original data in the model.

Key Uses of AngularJS Filters

1. **Formatting Data** – Modify text, numbers, or dates for better readability.
2. **Filtering Data** – Select specific data from a list (e.g., search results).
3. **Sorting Data** – Arrange lists in ascending or descending order.
4. **Currency & Number Formatting** – Display numbers in a currency format or with specific decimal places.
5. **Custom Transformations** – Create custom filters for specific data manipulations.

Code : -

DIRECTORY STRUCTURE



Index.html

```
<!DOCTYPE html>
<html ng-app="bookApp" ng-cloak>
<head>
  <title>BookHub</title>
  <meta charset="utf-8">
  <meta name="viewport"
content="width=device-width,
initial-scale=1">
  <link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500&display=swap" rel="stylesheet">
  <link
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
  <script src="app.js"></script>
  <script src="bookFilter.js"></script>
  <script src="authService.js"></script>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div class="app-container"
ng-app="bookApp"
ng-controller="MainController">
    <!-- Header -->
    <header class="app-header">
      <h1><i class="fas fa-book-open"></i> BookHub</h1>
      <div class="auth-status">
        <span ng-if="isAuthenticated"
class="welcome-msg">
          Welcome, {{currentUser}}!
```

```
      <button ng-click="logout()"
class="btn-logout">Logout</button>
    </div>
  </header>

  <!-- Login Form -->
  <section ng-if="!isAuthenticated"
class="card login-section">
    <h2><i class="fas fa-lock"></i>
Secure Login</h2>
    <form ng-submit="login()"
class="login-form">

      <div class="form-group">
        <label><i class="fas fa-user"></i> Username</label>
        <input type="text"
ng-model="credentials.username" required
class="form-control">
      </div>
      <div class="form-group">
        <label><i class="fas fa-key"></i> Password</label>
        <input type="password"
ng-model="credentials.password" required
class="form-control">
      </div>
      <button type="submit"
class="btn-primary">Login</button>

      <div ng-class="{ 'alert-success':
authSuccess, 'alert-error': !authSuccess }"
class="auth-message">
        {{authMessage}}
```

```

        </div>
    </form>
</section>

<!-- Book Search -->
    <section ng-if="isAuthenticated"
class="card book-section">
        <div class="search-header">
            <h2><i class="fas fa-search"></i>
Book Explorer</h2>
            <div class="search-box">
                <i class="fas fa-search
search-icon"></i>
                <input type="text"
                    ng-model="searchQuery"
                    placeholder="Search by
title, author or genre..."
                    class="search-input">
            </div>
        </div>

        <div class="book-grid">
            <!-- Updated ng-repeat to use
bookFilter -->
            <div ng-repeat="book in books |
bookFilter:searchQuery" class="book-card">

```

App.js

```

angular.module('bookApp', [])
    .controller('MainController', function
($scope, AuthService) {
        console.log("MainController Loaded");
        // ✓ Debugging log

        // Authentication state
        $scope.isAuthenticated =
AuthService.isAuthenticated();
        $scope.currentUser =
AuthService.getCurrentUser();
        $scope.authMessage = "";

        $scope.authSuccess = false;
        $scope.credentials = {
            username: "",
            password: ""
        };
        $scope.searchQuery = ""; // Initialize
search query

        // Books data

```

```

        <div class="book-cover">
            <i class="fas
fa-book-open"></i>
        </div>
        <div class="book-details">
            <h3>{{book.title}}</h3>

            <p><strong>Author:</strong>
{{book.author}}</p>
            <p><strong>Genre:</strong>

            <span
class="genre-tag">{{book.genre}}</span></p>

            <p><strong>Year:</strong>
{{book.year}}</p>
        </div>
    </div>
</section>

    <footer class="app-footer">
        © 2025 BookHub
    </footer>
</div>
</body>
</html>

```

```

$scope.books = [
    { title: 'The Immortals of Meluha',
author: 'Amish Tripathi', genre: 'Mythology',
year: 2010 },
    { title: 'The White Tiger', author:
'Aravind Adiga', genre: 'Fiction', year: 2008
},
    { title: 'Wings of Fire', author: 'A.P.J.
Abdul Kalam', genre: 'Biography', year:
1999 },
    { title: 'Chanakya's Chant', author:
'Ashwin Sanghi', genre: 'Thriller', year: 2010
},
    { title: 'Train to Pakistan', author:
'Khushwant Singh', genre: 'Historical
Fiction', year: 1956 },
    { title: 'Sita: Warrior of Mithila',
author: 'Amish Tripathi', genre: 'Mythology',
year: 2017 },
    { title: 'The God of Small Things',
author: 'Arundhati Roy', genre: 'Fiction',
year: 1997 },
    { title: 'Sacred Games', author:
'Vikram Chandra', genre: 'Thriller', year:
2006 },
    { title: 'Midnight's Children', author:
'Salman Rushdie', genre: 'Fiction', year:
1981 },

```

```

        { title: 'The Glass Palace', author:
'Amitav Ghosh', genre: 'Historical Fiction',
year: 2000 },
        { title: 'I Am Malala', author: 'Malala
Yousafzai', genre: 'Biography', year: 2013 }
    ];

    // Login function using AuthService
    $scope.login = function () {
        var response =
AuthService.login($scope.credentials);

        if (response.success) {
            $scope.isAuthenticated = true;
            $scope.currentUser =
AuthService.getCurrentUser();
            $scope.authSuccess = true;
            $scope.authMessage =
response.message;
        } else {
            $scope.authSuccess = false;
            $scope.authMessage =
response.message;
        }
    };

    // Logout function
    $scope.logout = function () {
        AuthService.logout();
        $scope.isAuthenticated = false;
        $scope.currentUser = "";
        $scope.credentials = { username: "",
password: "" };
        $scope.authMessage = 'You have
been logged out';
        $scope.searchQuery = "";
    };
});

```

bookFilter.js

```

angular.module("bookApp").filter("bookFilter", function () {
    return function (books, searchText) {
        if (!searchText) return books; // Return all books if searchText is empty

        searchText = searchText.toLowerCase();
        return books.filter(function (book) {
            return book.title.toLowerCase().includes(searchText) ||
                book.author.toLowerCase().includes(searchText) ||
                book.genre.toLowerCase().includes(searchText);
        });
    };
});

```

authService.js

```

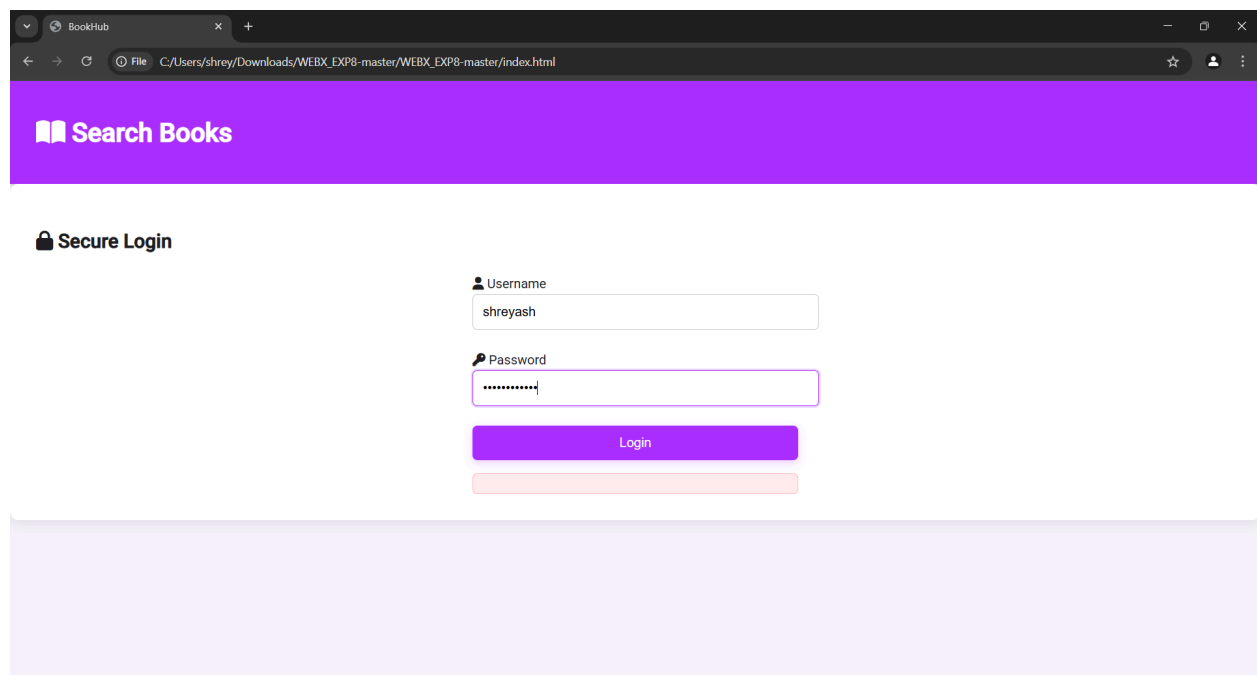
angular.module('bookApp').service('AuthService', function() {
    var currentUser = null; // Store the logged-in user
    var validUsers = [
        { username: "admin", password: "password123" },
        { username: "shreyash", password: "shreyash123" }
    ];
    return {
        login: function(credentials) {
            var user = validUsers.find(u => u.username === credentials.username && u.password
=== credentials.password);

            if (user) {
                currentUser = credentials.username;
                return { success: true, message: "Login Successful!" };
            } else {
                return { success: false, message: "Invalid Credentials!" };
            }
        },
    };
});

```

```
logout: function() {
    currentUser = null;
},
isAuthenticated: function() {
    return currentUser !== null;
},
getCurrentUser: function() {
    return currentUser;
}
};
});
```

OUTPUT: -



BookHub

File C:/Users/shrey/Downloads/WEBX_EXP8-master/WEBX_EXP8-master/index.html

Search Books

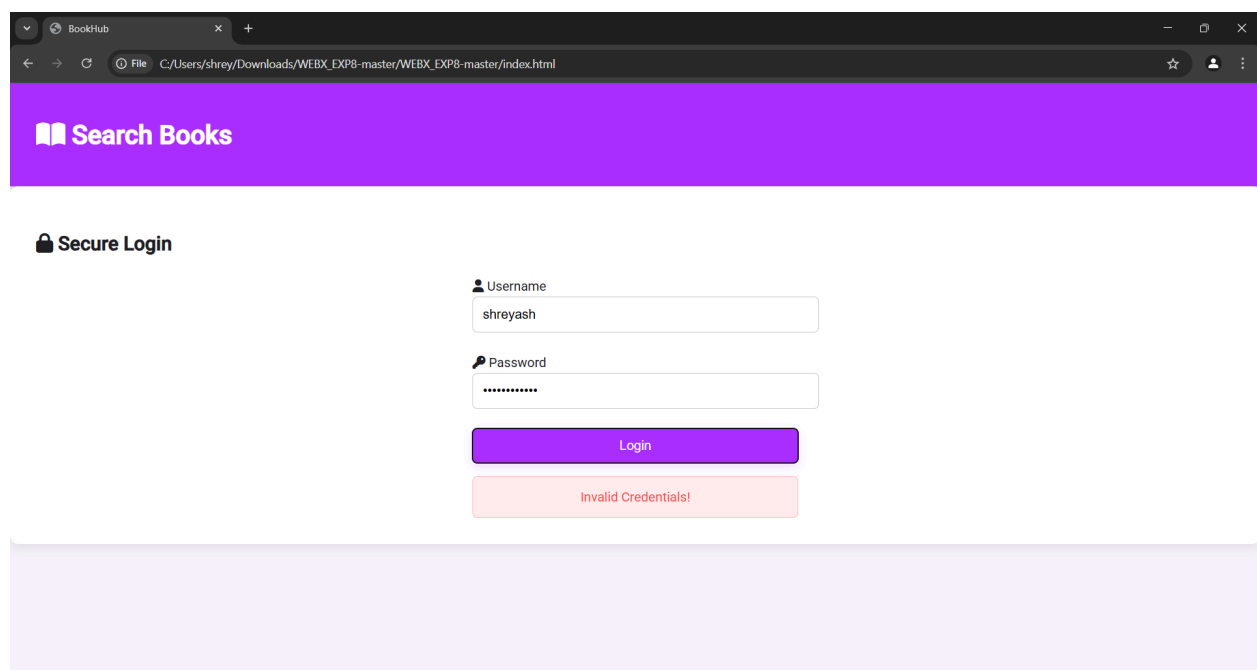
Secure Login

Username
shreyash

Password

Login

- When user logs in with incorrect details, it will display as “Invalid Credentials”



BookHub

File C:/Users/shrey/Downloads/WEBX_EXP8-master/WEBX_EXP8-master/index.html

Search Books

Secure Login

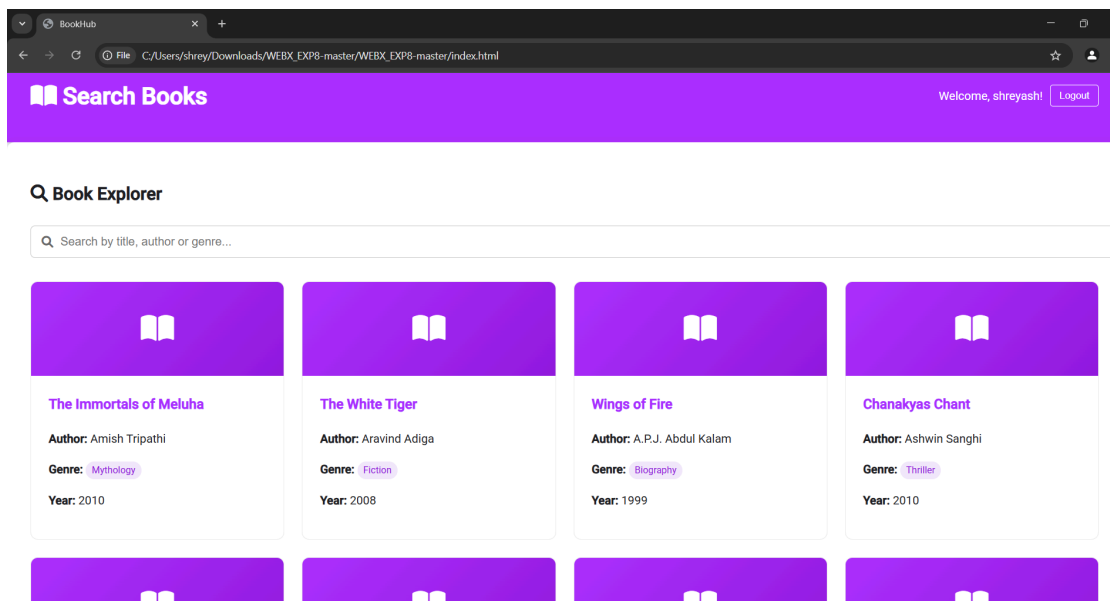
Username
shreyash

Password

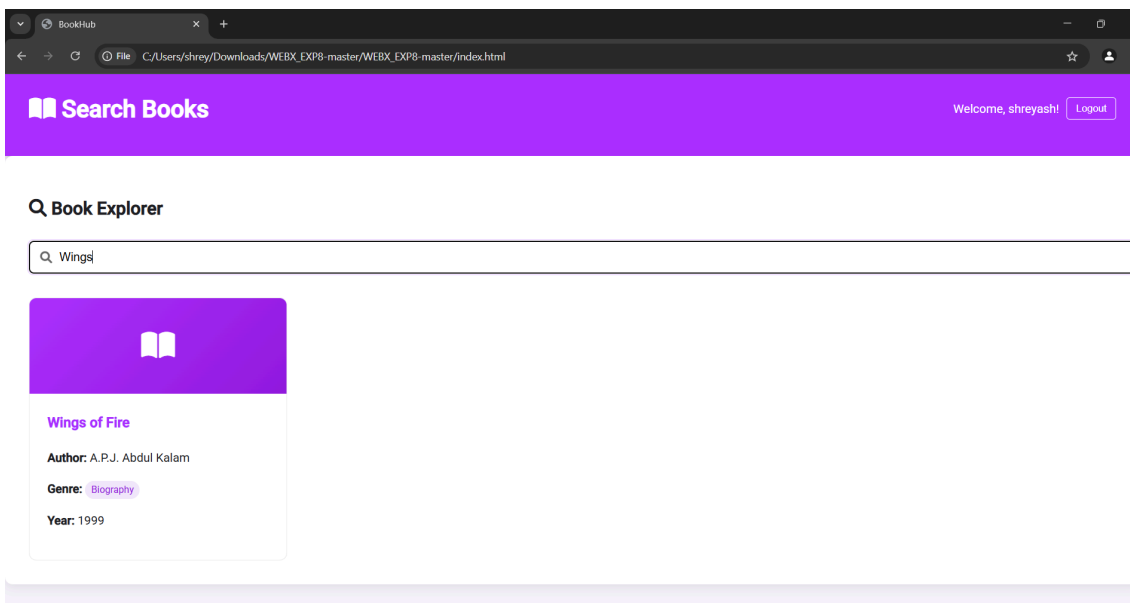
Login

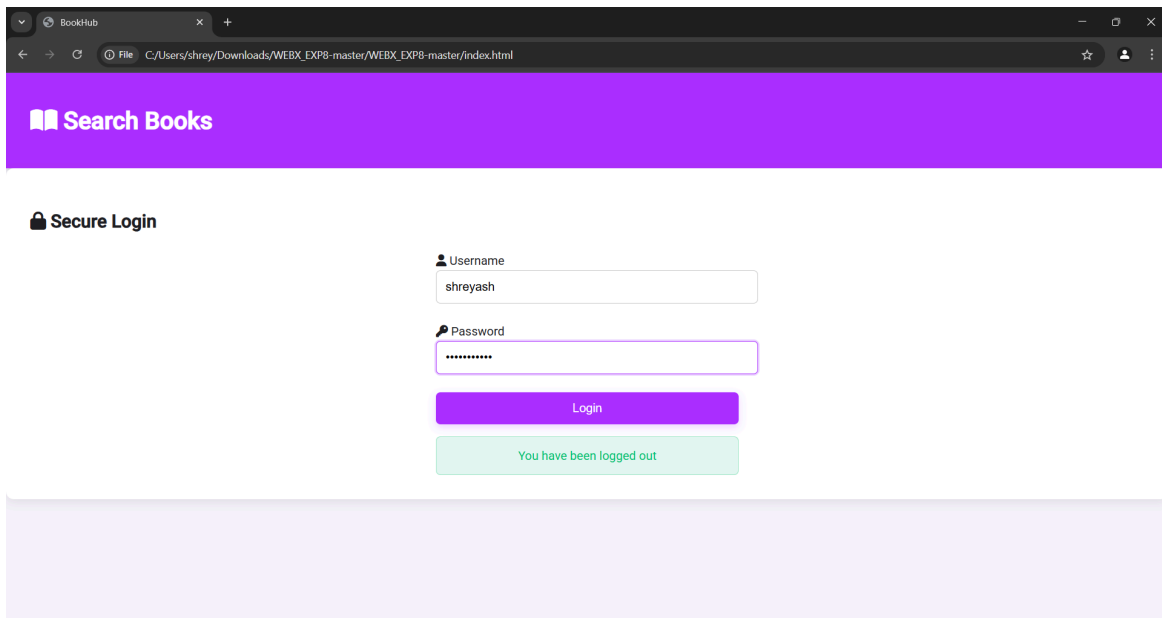
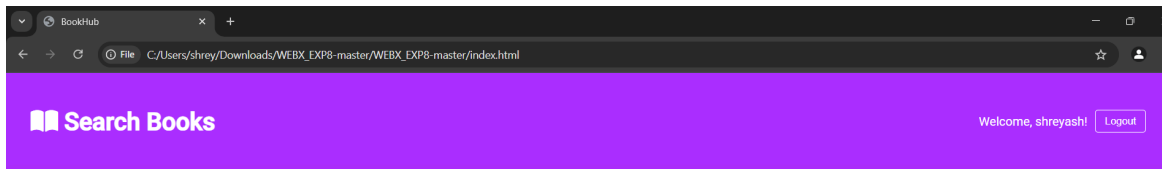
Invalid Credentials!

- After successful authentication, main page will be displayed



- User can search by title, author and genre





Conclusion: -

This practical explores key AngularJS concepts, including data binding, authentication, custom filters, and services. It demonstrates one-way and two-way data binding, showcasing how data flows between the model and view. A basic authentication system is implemented using AngularJS controllers, modules, and form directives. A custom filter (bookFilter) is created for searching books by title, author, or genre, while a modular authentication service ensures reusability and maintainability. These implementations highlight AngularJS's capabilities in building dynamic and interactive web applications.