Lab Mid-Term Report - Advanced Web Technologies

Name: Muhammad Jawad Afzal
Reg No.: SP22-BSE-031
Course: CSC337 - Advanced Web Technologies
Instructor: Yasmeen Jana

Project Title: Book Management Web Application

Objective:

To develop a full-stack web application that allows users to:

- View a list of books.
- Search books by author.
- Add new books to the system.

Tools & Technologies Used:

- **Frontend:** HTML, CSS (Bootstrap), JavaScript
- **Backend:** Node.js, Express.js
- **Database:** MongoDB (using Compass for local database)
- **Package Manager:** npm

Project Structure:

```
book-app/
├── backend/
│   ├── models/
│   │   └── bookModel.js
│   ├── routes/
│   │   └── bookRoutes.js
│   ├── .env
│   ├── server.js
├── frontend/
│   ├── index.html
│   ├── add.html
```

```
│   └── style.css (optional)
```

Backend Functionality:

- **/api/books [GET]:** Retrieves all books or filters by author using query string.
- **/api/books [POST]:** Adds a new book to MongoDB.
- **Mongoose model:** Defined in bookModel.js with fields: title, author, price.
- **MongoDB Connection:** Uses Mongoose to connect using .env configuration.

add.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Add Book</title>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">
</head>
<body class="p-4">
  <h1>Add a New Book</h1>
  <form id="addBookForm">
    <input class="form-control mb-2" name="title" placeholder="Title" required />
    <input class="form-control mb-2" name="author" placeholder="Author" required />
    <input class="form-control mb-2" name="price" type="number" placeholder="Price" required
/>
    <button class="btn btn-success">Submit</button>
  </form>
  <script>
    document.getElementById('addBookForm').addEventListener('submit', async (e) => {
      e.preventDefault();
      const formData = new FormData(e.target);
      const data = Object.fromEntries(formData);
      await fetch('http://localhost:5000/api/books', {
```

```
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(data)
    });
    alert('Book added!');
    window.location.href = 'index.html';
  });
 </script>
</body>
</html>
```

├── index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Book List</title>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">
  <link rel="stylesheet" href="style.css">

</head>
<body class="bg-light">
  <div class="container py-4">
    <h1 class="text-center mb-4">📚 Book Collection</h1>

    <input type="text" id="authorSearch" placeholder="Search by author..." class="form-control mb-3"
/>

    <div id="bookList" class="row g-4"></div>
```

```
  <div class="text-center mt-4">
    <a href="add.html" class="btn btn-success">➕ Add New Book</a>
  </div>
</div>

<script>
  const listContainer = document.getElementById('bookList');
  const searchInput = document.getElementById('authorSearch');

  async function fetchBooks(author = '') {
    try {
      const res = await fetch(`http://localhost:5000/api/books?author=${author}`);
      const books = await res.json();

      if (books.length === 0) {
        listContainer.innerHTML = `<p class="text-center">No books found.</p>`;
        return;
      }

      listContainer.innerHTML = books.map(book => `
        <div class="col-md-4">
          <div class="card shadow-sm">
            <div class="card-body">
              <h5 class="card-title">${book.title}</h5>
              <h6 class="card-subtitle mb-2 text-muted">${book.author}</h6>
              <p class="card-text">Price: <strong>$${book.price}</strong></p>
            </div>
          </div>
        </div>
      `).join('');
    } catch (err) {
      listContainer.innerHTML = `<p class="text-danger text-center">Error fetching books.</p>`;
```

```
      console.error(err);
    }
  }

  searchInput?.addEventListener('input', e => fetchBooks(e.target.value));
  fetchBooks();
 </script>
</body>
</html>
```

├── routes/
│ │ └── bookRoutes.js

```
const express = require('express');
const router = express.Router();
const Book = require('../models/Book');

router.get('/', async (req, res) => {
 try {
  const { author } = req.query;
  const query = author ? { author: new RegExp(author, 'i') } : {};
  const books = await Book.find(query);
  res.json(books);
 } catch (err) {
  console.error('Error fetching books:', err);
  res.status(500).json({ error: 'Failed to fetch books' });
 }
});

router.post('/', async (req, res) => {
 try {
```

```
    const { title, author, price } = req.body;

    // Validate required fields
    if (!title || !author || !price) {
      return res.status(400).json({ error: 'Missing required fields' });
    }

    const newBook = new Book({ title, author, price });
    await newBook.save();
    res.status(201).json(newBook);
  } catch (err) {
    console.error('Error saving book:', err);
    res.status(500).json({ error: 'Failed to save book' });
  }
});

module.exports = router;
```

├── models/

│ │ └── bookModel.js

```
const mongoose = require('mongoose');

const bookSchema = new mongoose.Schema({
  title: { type: String, required: true },
  author: { type: String, required: true },
  price: { type: Number, required: true }
});

module.exports = mongoose.model('Book', bookSchema);
```