

Lab Mid

Program: BS Software Engineering

Department of Computer Science



Submitted to: Ma'am Yasmeen

Submitted by: Habiba Ashfaq

Roll no: SP22-BSE-032

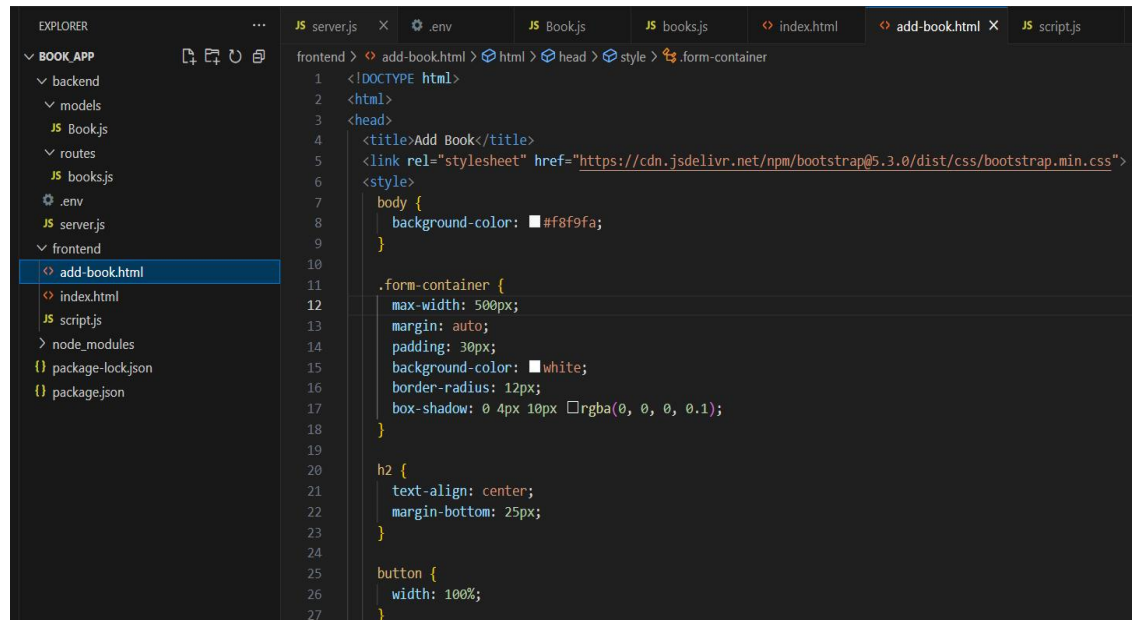
Course Title: Advanced Web Technologies

**COMSATS University Islamabad, Vehari
Campus.**

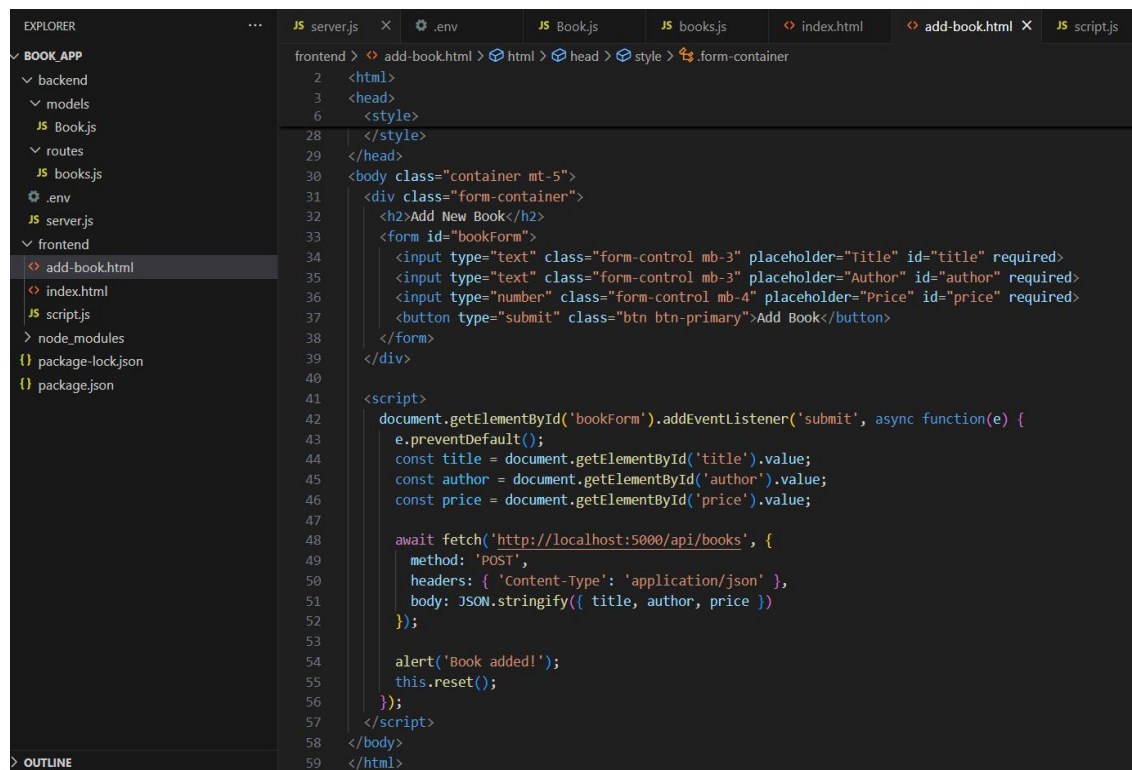
Project Name: Book Management System

Frontend:

Add Book:

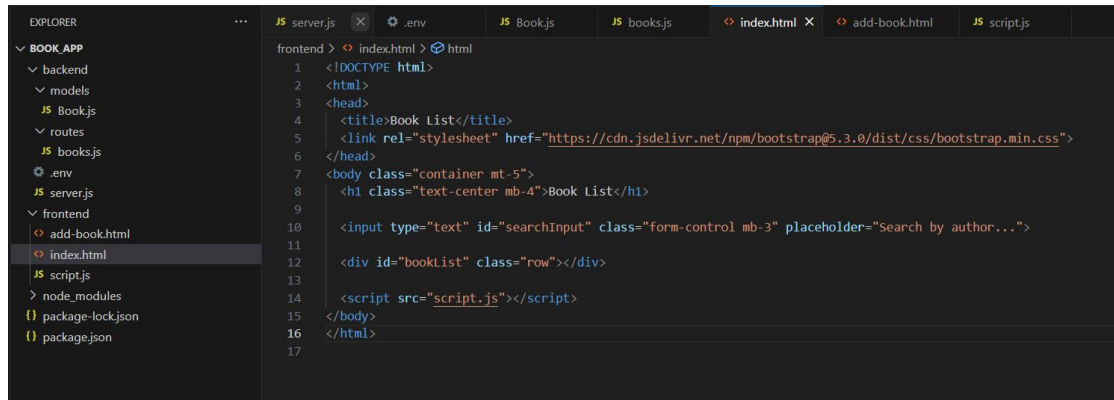


```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Add Book</title>
5 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">
6 <style>
7   body {
8     background-color: #f8f9fa;
9   }
10
11   .form-container {
12     max-width: 500px;
13     margin: auto;
14     padding: 30px;
15     background-color: white;
16     border-radius: 12px;
17     box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
18   }
19
20   h2 {
21     text-align: center;
22     margin-bottom: 25px;
23   }
24
25   button {
26     width: 100%;
27   }
```



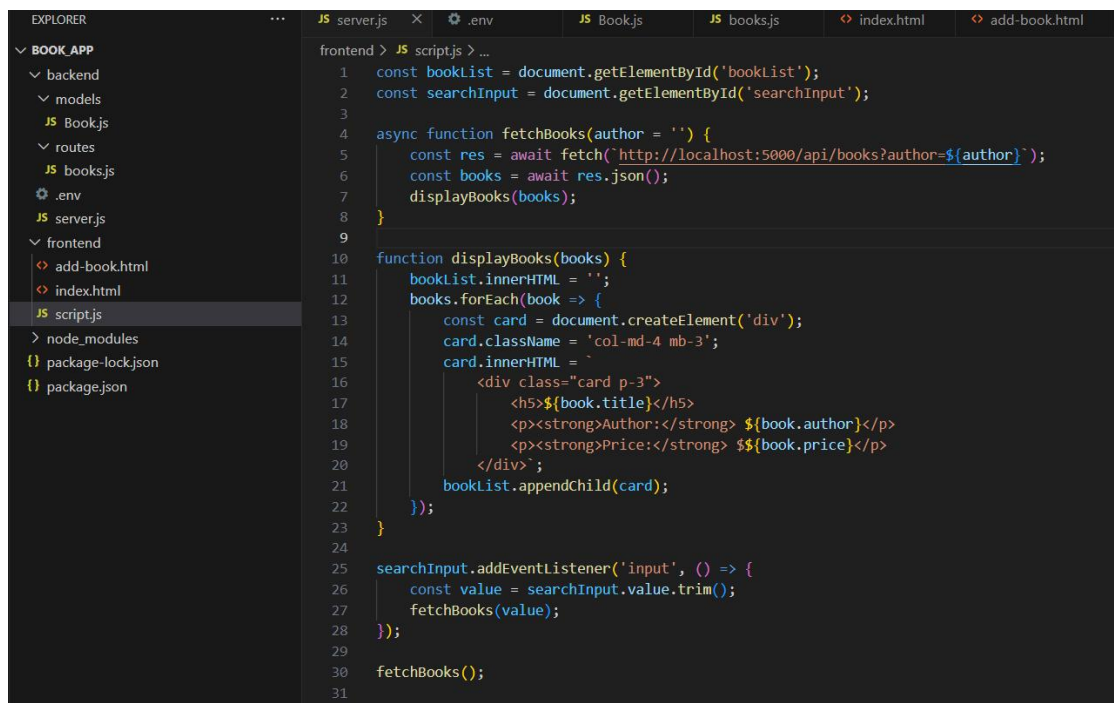
```
1 <html>
2 <head>
3 <style>
4
5 </style>
6 </head>
7 <body class="container mt-5">
8 <div class="form-container">
9 <h2>Add New Book</h2>
10 <form id="bookForm">
11   <input type="text" class="form-control mb-3" placeholder="Title" id="title" required>
12   <input type="text" class="form-control mb-3" placeholder="Author" id="author" required>
13   <input type="number" class="form-control mb-4" placeholder="Price" id="price" required>
14   <button type="submit" class="btn btn-primary">Add Book</button>
15 </form>
16 </div>
17
18 <script>
19   document.getElementById('bookForm').addEventListener('submit', async function(e) {
20     e.preventDefault();
21     const title = document.getElementById('title').value;
22     const author = document.getElementById('author').value;
23     const price = document.getElementById('price').value;
24
25     await fetch('http://localhost:5000/api/books', {
26       method: 'POST',
27       headers: { 'Content-Type': 'application/json' },
28       body: JSON.stringify({ title, author, price })
29     });
30
31     alert('Book added!');
32     this.reset();
33   });
34 </script>
35 </body>
36 </html>
```

Index file for Book List:



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Book List</title>
5   <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">
6 </head>
7 <body class="container mt-5">
8   <h1 class="text-center mb-4">Book List</h1>
9
10  <input type="text" id="searchInput" class="form-control mb-3" placeholder="Search by author...">
11
12  <div id="bookList" class="row"></div>
13
14  <script src="script.js"></script>
15 </body>
16 </html>
17
```

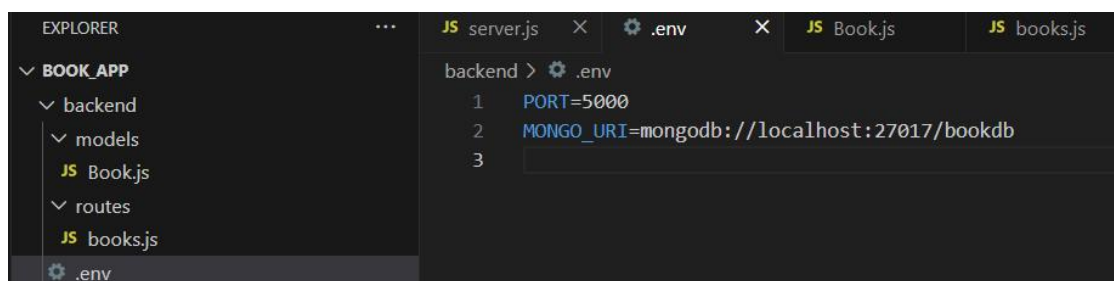
Script File:



```
1 const bookList = document.getElementById('bookList');
2 const searchInput = document.getElementById('searchInput');
3
4 async function fetchBooks(author = '') {
5   const res = await fetch('http://localhost:5000/api/books?author=${author}');
6   const books = await res.json();
7   displayBooks(books);
8 }
9
10 function displayBooks(books) {
11   bookList.innerHTML = '';
12   books.forEach(book => {
13     const card = document.createElement('div');
14     card.className = 'col-md-4 mb-3';
15     card.innerHTML = `
16       <div class="card p-3">
17         <h5>${book.title}</h5>
18         <p><strong>Author:</strong> ${book.author}</p>
19         <p><strong>Price:</strong> ${book.price}</p>
20       </div>`;
21     bookList.appendChild(card);
22   });
23 }
24
25 searchInput.addEventListener('input', () => {
26   const value = searchInput.value.trim();
27   fetchBooks(value);
28 });
29
30 fetchBooks();
31
```

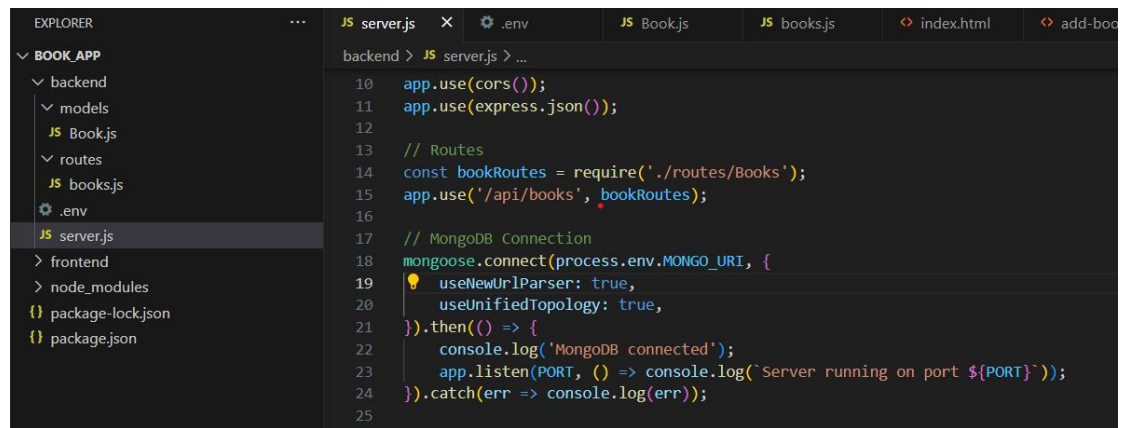
Backend:

.env file:



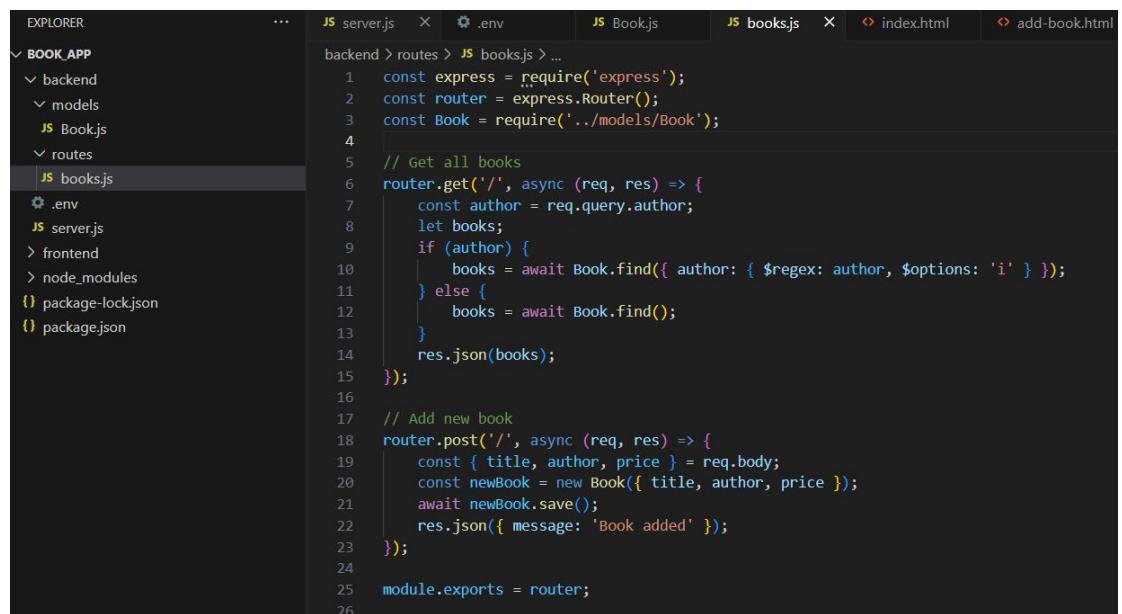
```
1 PORT=5000
2 MONGO_URI=mongodb://localhost:27017/bookdb
3
```

Server.js file:



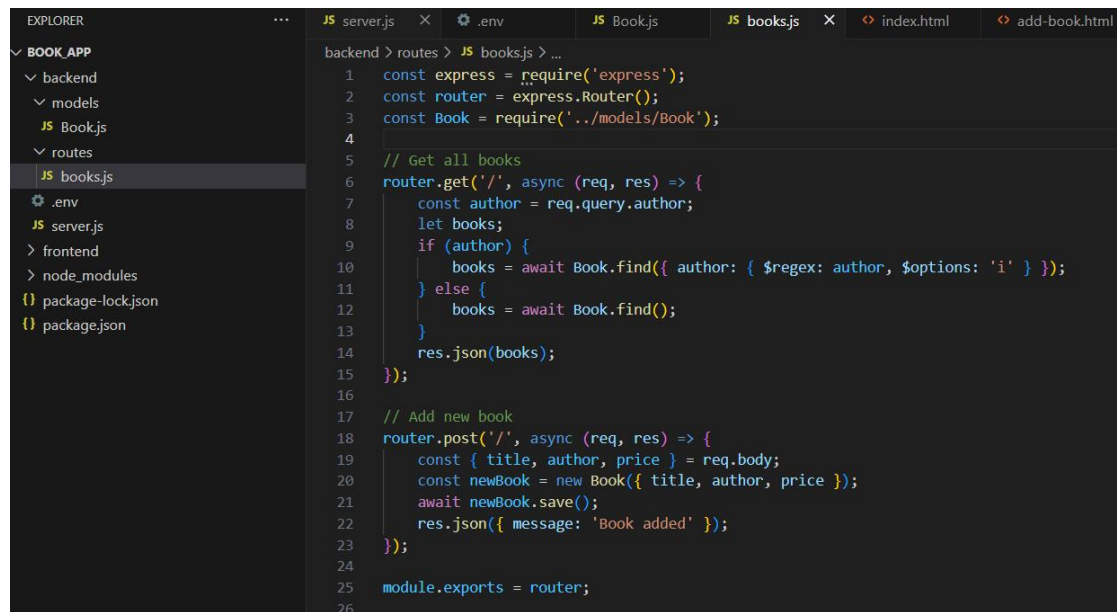
```
10 app.use(cors());
11 app.use(express.json());
12
13 // Routes
14 const bookRoutes = require('./routes/Books');
15 app.use('/api/books', bookRoutes);
16
17 // MongoDB Connection
18 mongoose.connect(process.env.MONGO_URI, {
19   useNewUrlParser: true,
20   useUnifiedTopology: true,
21 }).then(() => {
22   console.log('MongoDB connected');
23   app.listen(PORT, () => console.log(`Server running on port ${PORT}`));
24 }).catch(err => console.log(err));
25
```

Routes file:



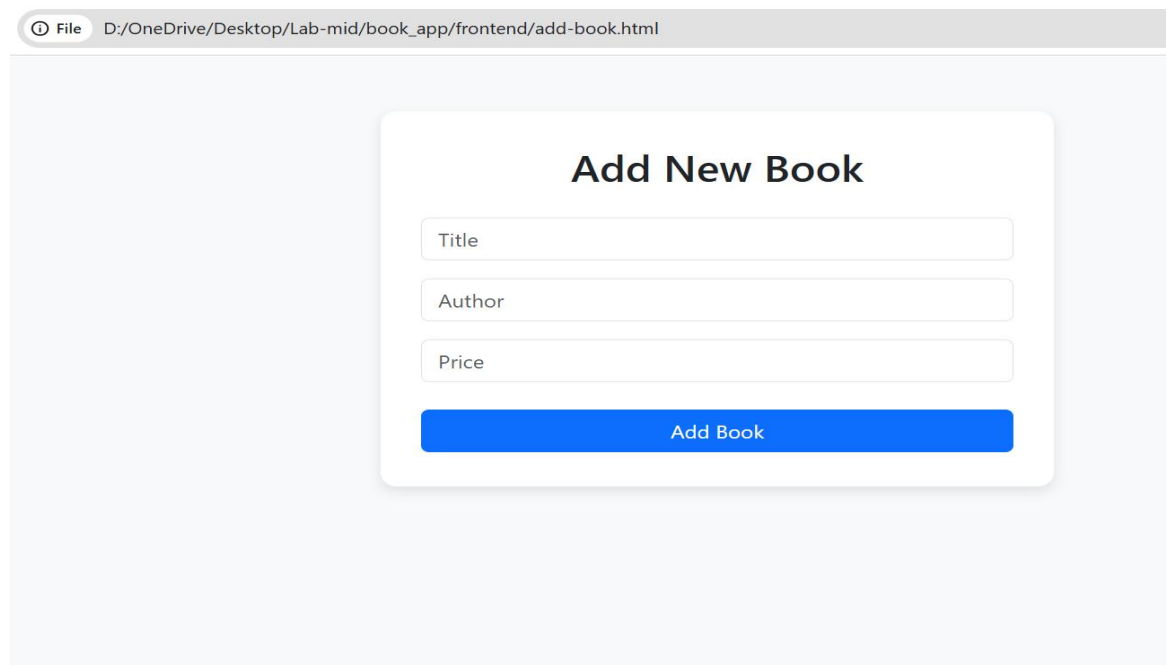
```
1 const express = require('express');
2 const router = express.Router();
3 const Book = require('../models/Book');
4
5 // Get all books
6 router.get('/', async (req, res) => {
7   const author = req.query.author;
8   let books;
9   if (author) {
10     books = await Book.find({ author: { $regex: author, $options: 'i' } });
11   } else {
12     books = await Book.find();
13   }
14   res.json(books);
15 });
16
17 // Add new book
18 router.post('/', async (req, res) => {
19   const { title, author, price } = req.body;
20   const newBook = new Book({ title, author, price });
21   await newBook.save();
22   res.json({ message: 'Book added' });
23 });
24
25 module.exports = router;
26
```

Models file:



```
backend > routes > JS books.js > ...
1  const express = require('express');
2  const router = express.Router();
3  const Book = require('../models/Book');
4
5  // Get all books
6  router.get('/', async (req, res) => {
7    const author = req.query.author;
8    let books;
9    if (author) {
10     books = await Book.find({ author: { $regex: author, $options: 'i' } });
11   } else {
12     books = await Book.find();
13   }
14   res.json(books);
15 });
16
17 // Add new book
18 router.post('/', async (req, res) => {
19   const { title, author, price } = req.body;
20   const newBook = new Book({ title, author, price });
21   await newBook.save();
22   res.json({ message: 'Book added' });
23 });
24
25 module.exports = router;
26
```

Output:

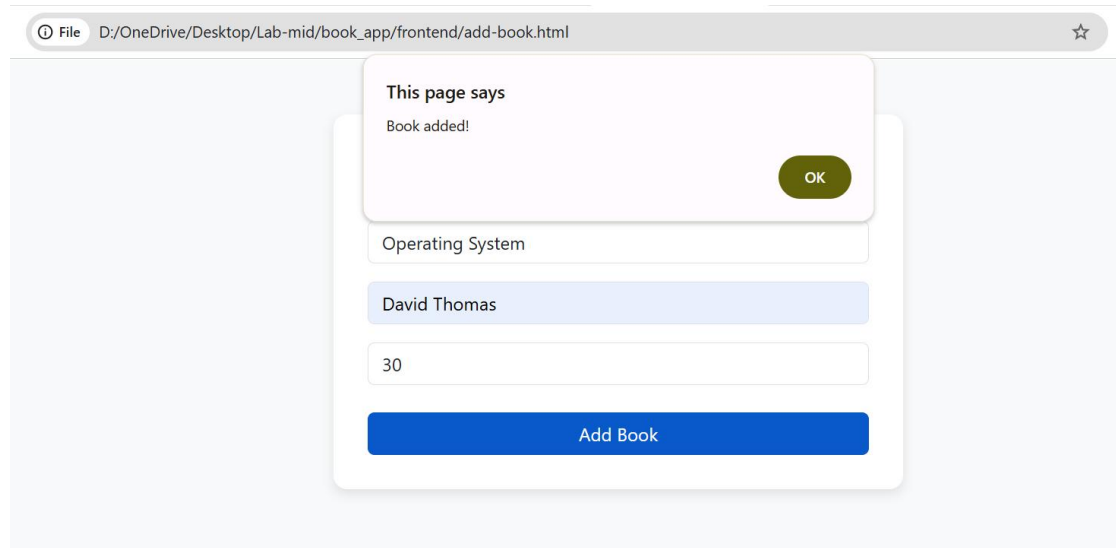


File D:/OneDrive/Desktop/Lab-mid/book_app/frontend/add-book.html

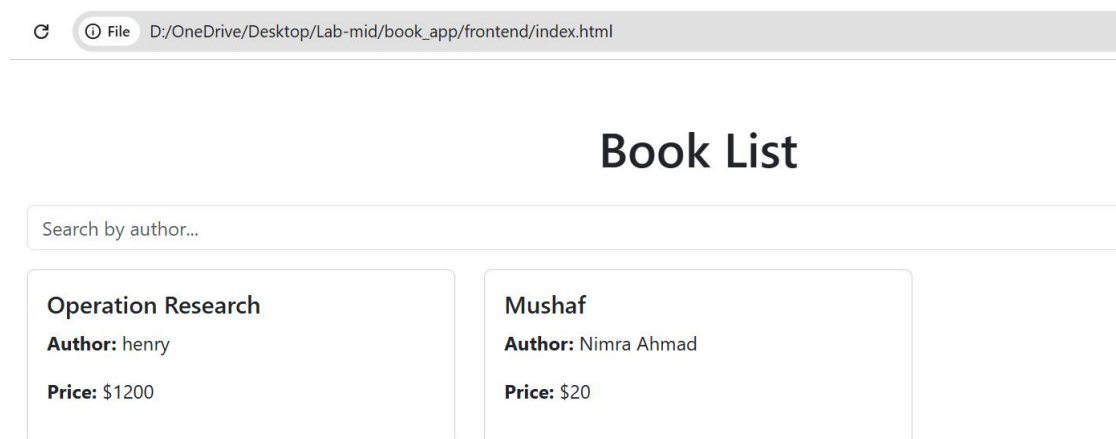
Add New Book

Add Book

Book added successfully:

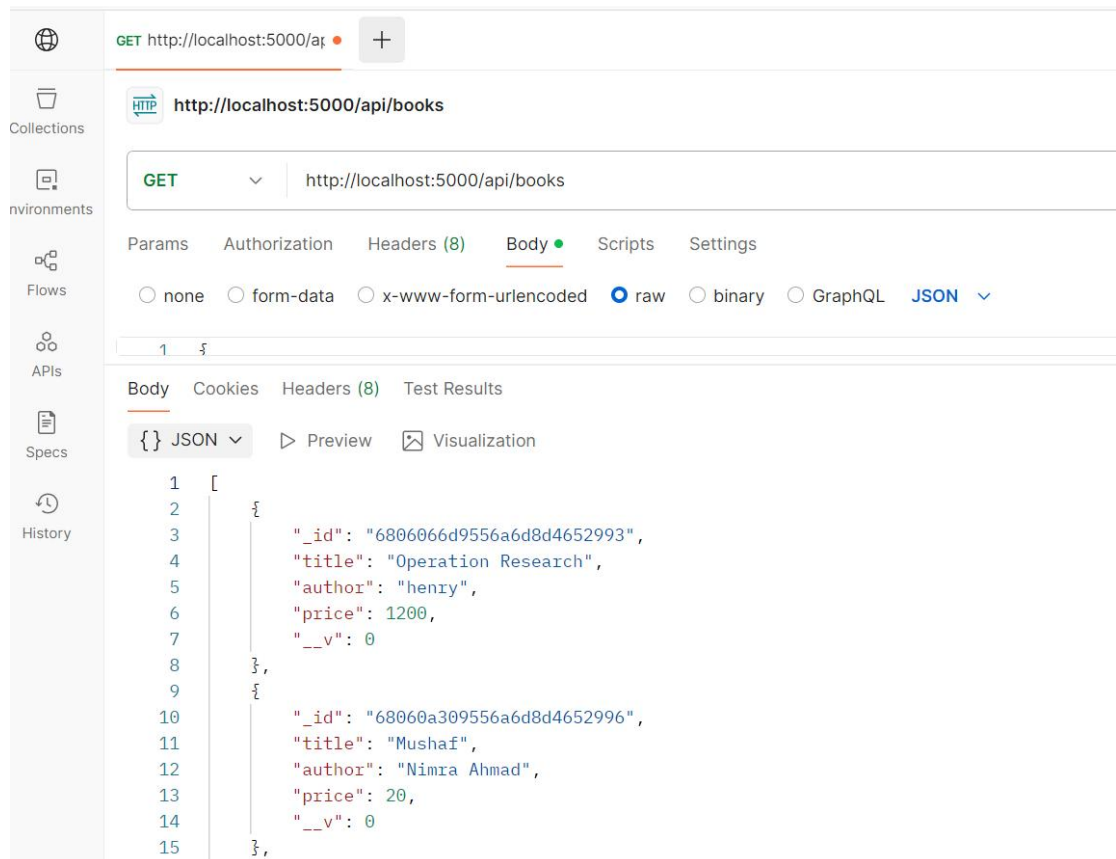


Book List:



Methods Output:

Get Method:



GET http://localhost:5000/api/books

HTTP http://localhost:5000/api/books

GET http://localhost:5000/api/books

Params Authorization Headers (8) Body Scripts Settings

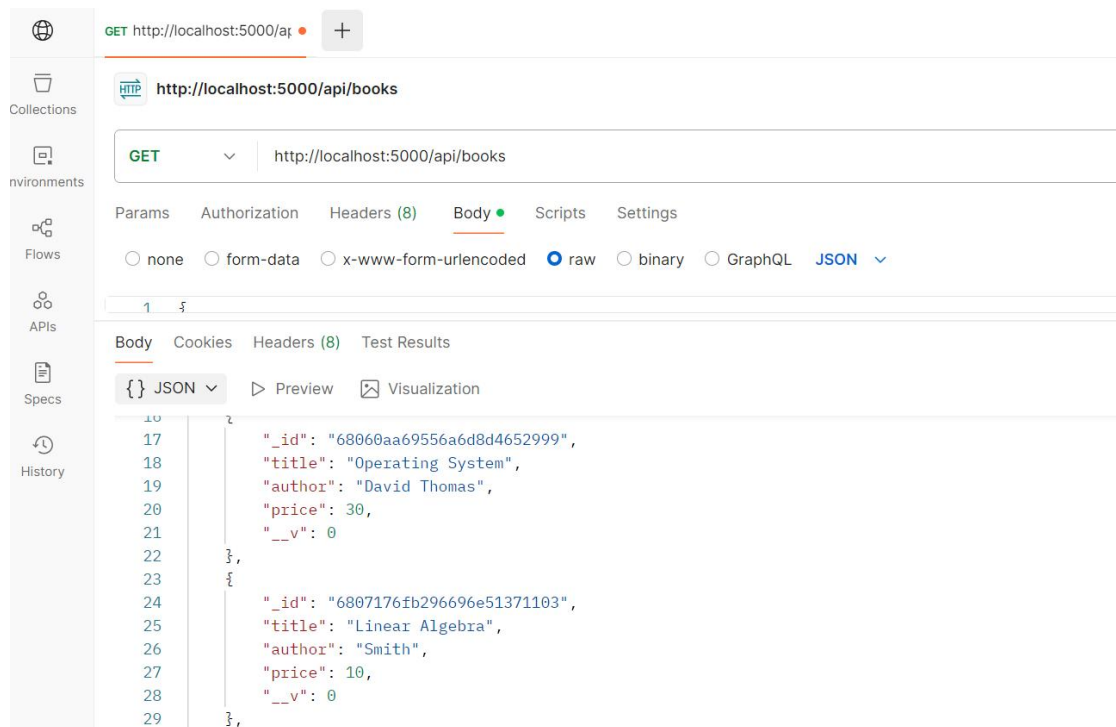
☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON

1 5

Body Cookies Headers (8) Test Results

{ } JSON Preview Visualization

```
1  [
2    {
3      "_id": "6806066d9556a6d8d4652993",
4      "title": "Operation Research",
5      "author": "henry",
6      "price": 1200,
7      "__v": 0
8    },
9    {
10     "_id": "68060a309556a6d8d4652996",
11     "title": "Mushaf",
12     "author": "Nimra Ahmad",
13     "price": 20,
14     "__v": 0
15   },
16 ]
```



GET http://localhost:5000/api/books

HTTP http://localhost:5000/api/books

GET http://localhost:5000/api/books

Params Authorization Headers (8) Body Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON

1 5

Body Cookies Headers (8) Test Results

{ } JSON Preview Visualization

```
10  [
17    {
18      "_id": "68060aa69556a6d8d4652999",
19      "title": "Operating System",
20      "author": "David Thomas",
21      "price": 30,
22      "__v": 0
23    },
24    {
25      "_id": "6807176fb296696e51371103",
26      "title": "Linear Algebra",
27      "author": "Smith",
28      "price": 10,
29      "__v": 0
30    },
31    {
32      "_id": "6807176fb296696e51371104",
33      "title": "Calculus",
34      "author": "John Doe",
35      "price": 15,
36      "__v": 0
37    },
38    {
39      "_id": "6807176fb296696e51371105",
40      "title": "Discrete Mathematics",
41      "author": "Alice",
42      "price": 25,
43      "__v": 0
44    }
45  ]
```


Post Method:

The screenshot shows the Postman interface. On the left is a sidebar with icons for Collections, Environments, Flows, APIs, Specs, and History. The main area displays a POST request to `http://localhost:5000/api/books`. The 'Body' tab is selected, showing a raw JSON payload:

```
{  "author": "smith",  "title": "0R",  "price": 210}
```

. Below the request, the 'Test Results' tab shows the response body:

```
{  "message": "Book added"}
```

.

Mongodb:

The screenshot shows the MongoDB Compass interface. On the left is a sidebar with 'Connections' and 'My Queries'. The main area displays the 'books' collection in the 'bookdb' database. The 'Documents' tab is selected, showing a list of documents. The first document is:

```
{  "_id": ObjectId('6806066d9556a6d8d4652993'),  "title": "Operation Research",  "author": "henry",  "price": 1200,  "__v": 0}
```

. The second document is:

```
{  "_id": ObjectId('68060a309556a6d8d4652996'),  "title": "Mushaf",  "author": "Nimra Ahmad",  "price": 20,  "__v": 0}
```

. The third document is:

```
{  "_id": ObjectId('68060aa69556a6d8d4652999'),  "title": "Operating System",  "author": "David Thomas",  "price": 30,  "__v": 0}
```

.

Compass

{ } My Queries

CONNECTIONS (1)

Search connections

▼ localhost:27017

▶ admin

▼ bookdb

books

▶ config

▶ local

books



localhost:27017 > bookdb > books

Documents 4

Aggregations

Schema

Indexes 1

Validation



Type a query: { field: 'value' } or [Generate query](#)

ADD DATA

EXPORT DATA

UPDATE

DELETE

```
_id: ObjectId('68060a309556a6d8d4652996')
title: "Mushaf"
author: "Nimra Ahmad"
price: 20
__v: 0
```

```
_id: ObjectId('68060aa69556a6d8d4652999')
title: "Operating System"
author: "David Thomas"
price: 30
__v: 0
```

```
_id: ObjectId('6807176fb296696e51371103')
title: "Linear Algebra"
author: "Smith"
price: 10
__v: 0
```