Lab Mid

Program: BS Software Engineering
Department of Computer Science

Submitted to: Ma'am Yasmeen
Submitted by: Misbah Jabbar
Roll no: SP22-BSE-011

Course Title: Advanced Web Technologies

COMSATS University Islamabad, Vehari Campus.

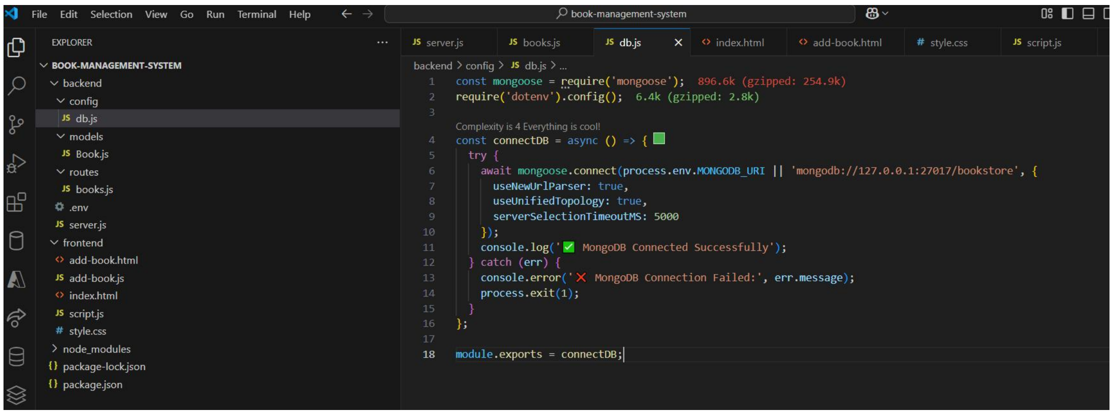
Question:no:01

You have been hired as a Full Stack Developer. Your task is to develop a complete web application that allows users to view a list of available books, search for books by author, and add new books to the system. You are required to build both the frontend and the backend components of this application.

Solution:

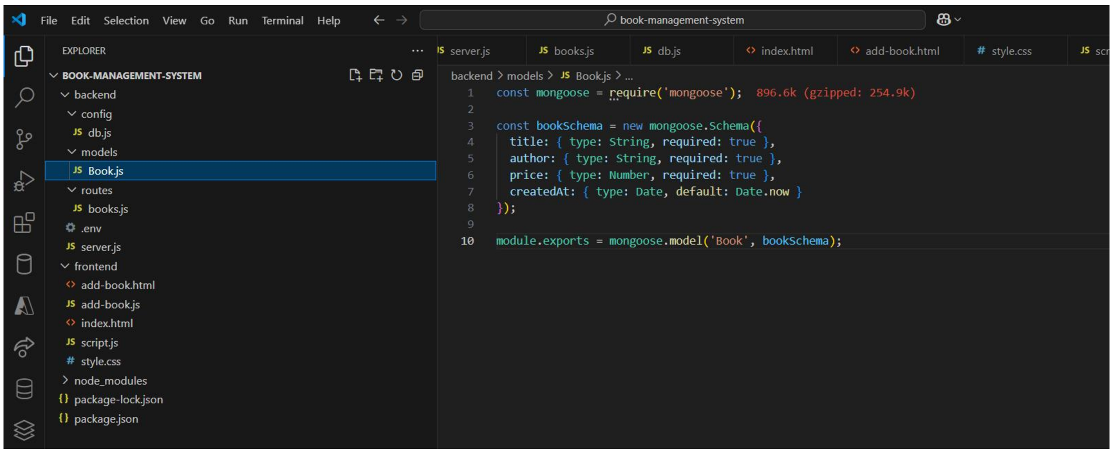
Backend System

Config(db.js):



```
const mongoose = require('mongoose');  896.6k (gzipped: 254.9k)
require('dotenv').config();  6.4k (gzipped: 2.8k)

Complexity is 4 Everything is cool!
const connectDB = async () => {
  try {
    await mongoose.connect(process.env.MONGODB_URI || 'mongodb://127.0.0.1:27017/bookstore', {
      useNewUrlParser: true,
      useUnifiedTopology: true,
      serverSelectionTimeoutMS: 5000
    });
    console.log('✅ MongoDB Connected Successfully');
  } catch (err) {
    console.error('❌ MongoDB Connection Failed:', err.message);
    process.exit(1);
  }
};

module.exports = connectDB;
```

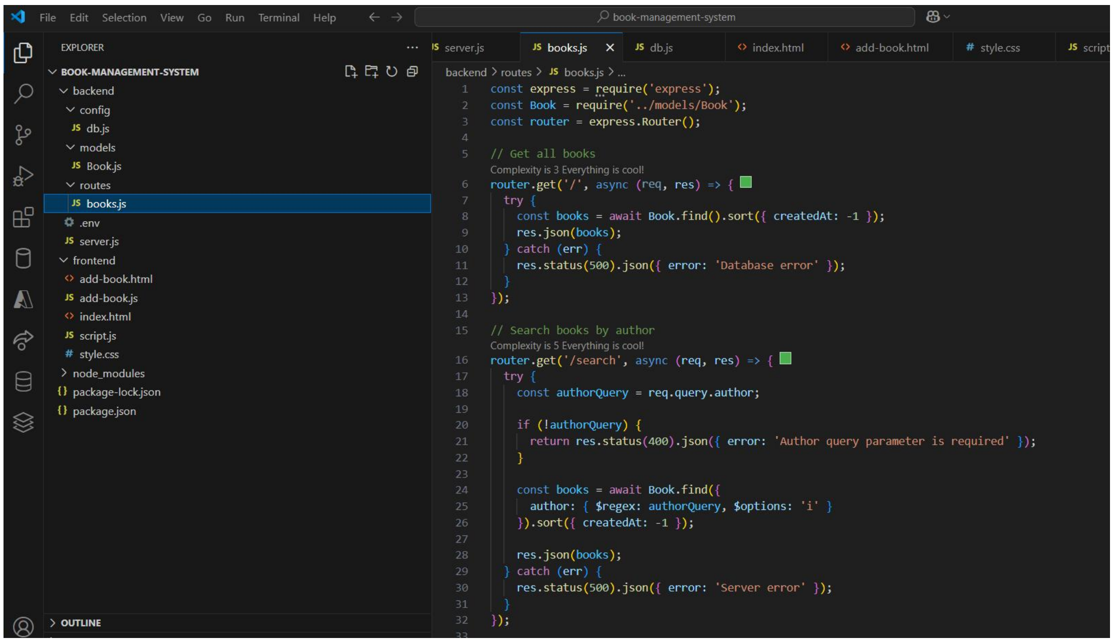
Model(Book.js)



```
const mongoose = require('mongoose');  896.6k (gzipped: 254.9k)

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

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

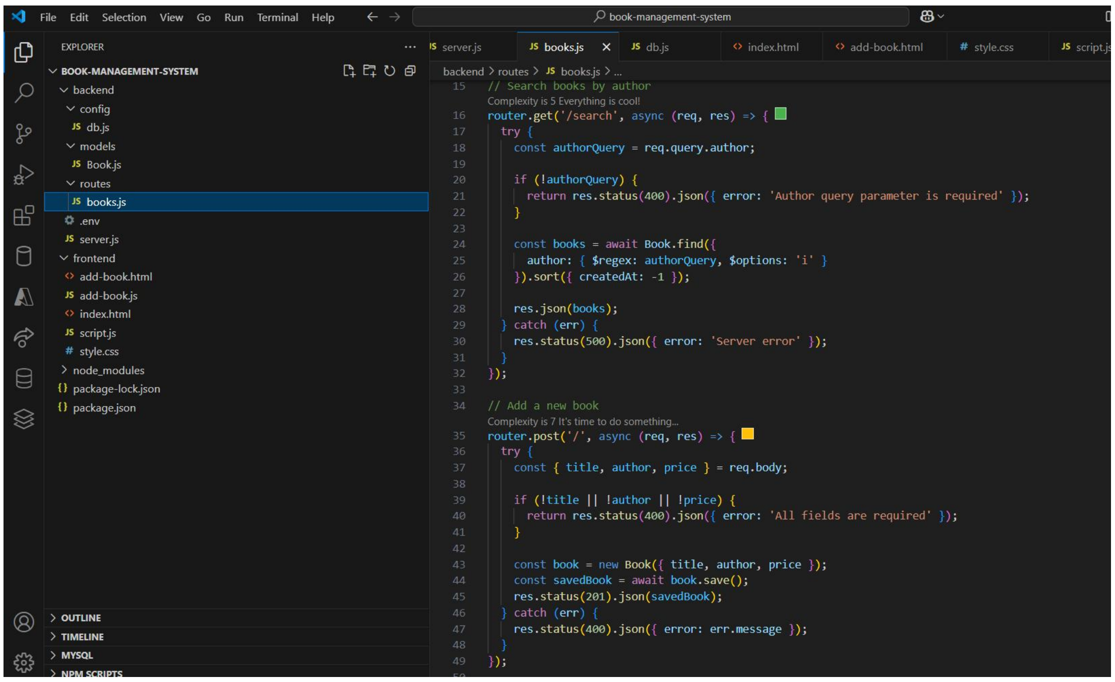
Routes(books.js)

A)



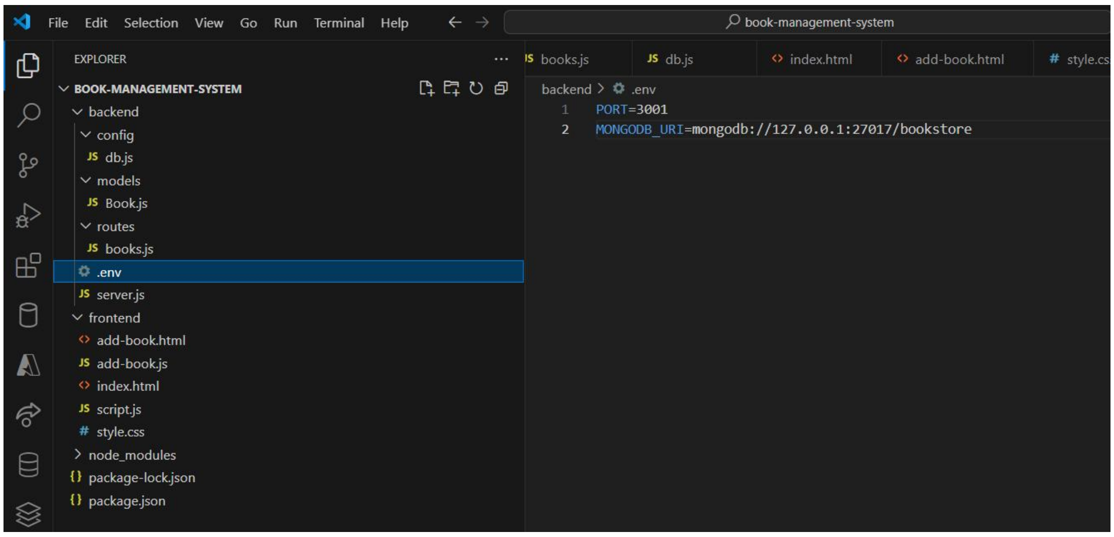
```
const express = require('express');
const Book = require('../models/Book');
const router = express.Router();

// Get all books
Complexity is 3 Everything is cool!
router.get('/', async (req, res) => {
  try {
    const books = await Book.find().sort({ createdAt: -1 });
    res.json(books);
  } catch (err) {
    res.status(500).json({ error: 'Database error' });
  }
});

// Search books by author
Complexity is 5 Everything is cool!
router.get('/search', async (req, res) => {
  try {
    const authorQuery = req.query.author;

    if (!authorQuery) {
      return res.status(400).json({ error: 'Author query parameter is required' });
    }

    const books = await Book.find({
      author: { $regex: authorQuery, $options: 'i' }
    }).sort({ createdAt: -1 });

    res.json(books);
  } catch (err) {
    res.status(500).json({ error: 'Server error' });
  }
});
```

B)



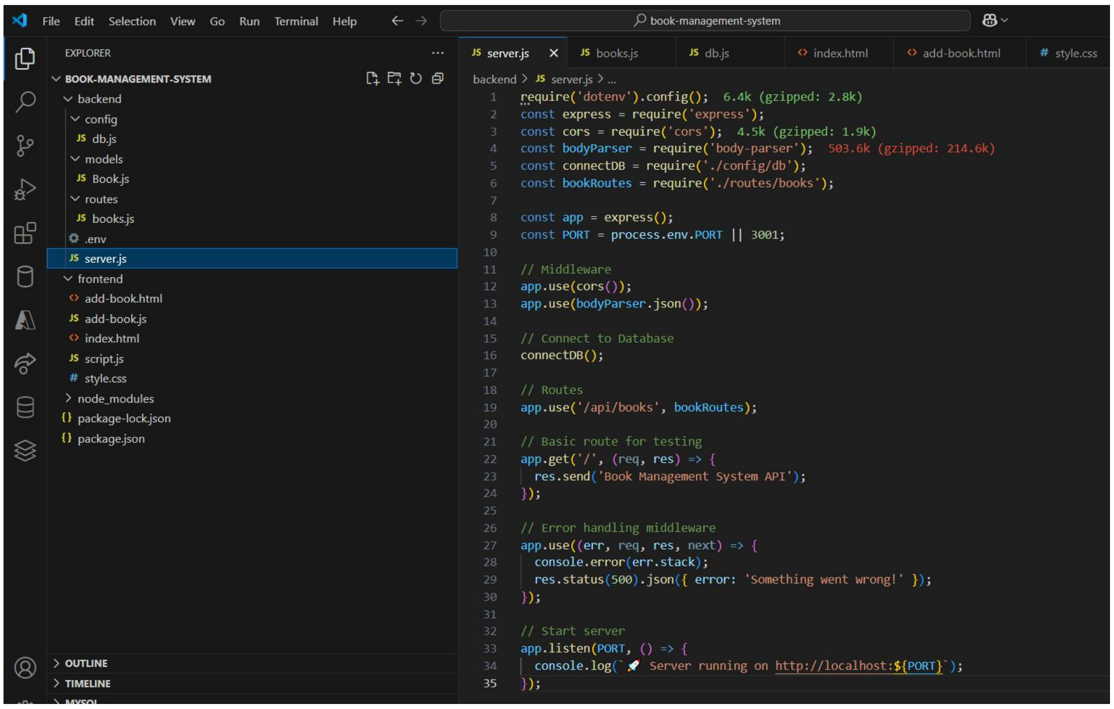
```
// Search books by author
Complexity is 5 Everything is cool!
router.get('/search', async (req, res) => {
  try {
    const authorQuery = req.query.author;

    if (!authorQuery) {
      return res.status(400).json({ error: 'Author query parameter is required' });
    }

    const books = await Book.find({
      author: { $regex: authorQuery, $options: 'i' }
    }).sort({ createdAt: -1 });

    res.json(books);
  } catch (err) {
    res.status(500).json({ error: 'Server error' });
  }
});

// Add a new book
Complexity is 7 It's time to do something...
router.post('/', async (req, res) => {
  try {
    const { title, author, price } = req.body;

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

    const book = new Book({ title, author, price });
    const savedBook = await book.save();
    res.status(201).json(savedBook);
  } catch (err) {
    res.status(400).json({ error: err.message });
  }
});
```

Env File:



```
PORT=3001
MONGODB_URI=mongodb://127.0.0.1:27017/bookstore
```

Server.js

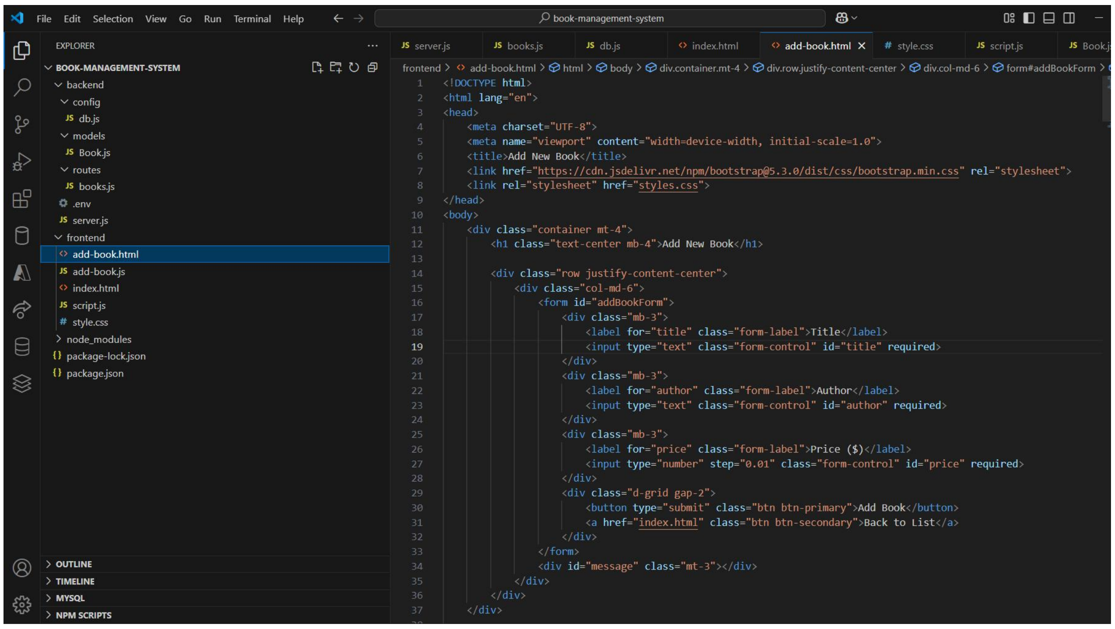


```
require('dotenv').config();  6.4k (gzipped: 2.8k)
const express = require('express');
const cors = require('cors');  4.5k (gzipped: 1.9k)
const bodyParser = require('body-parser');  503.6k (gzipped: 214.6k)
const connectDB = require('./config/db');
const bookRoutes = require('./routes/books');

const app = express();
const PORT = process.env.PORT || 3001;

// Middleware
app.use(cors());
app.use(bodyParser.json());

// Connect to Database
connectDB();

// Routes
app.use('/api/books', bookRoutes);

// Basic route for testing
app.get('/', (req, res) => {
  res.send('Book Management System API');
});

// Error handling middleware
app.use((err, req, res, next) => {
  console.error(err.stack);
  res.status(500).json({ error: 'Something went wrong!' });
});

// Start server
app.listen(PORT, () => {
  console.log(`🚀 Server running on http://localhost:${PORT}`);
});
```

Frontend System

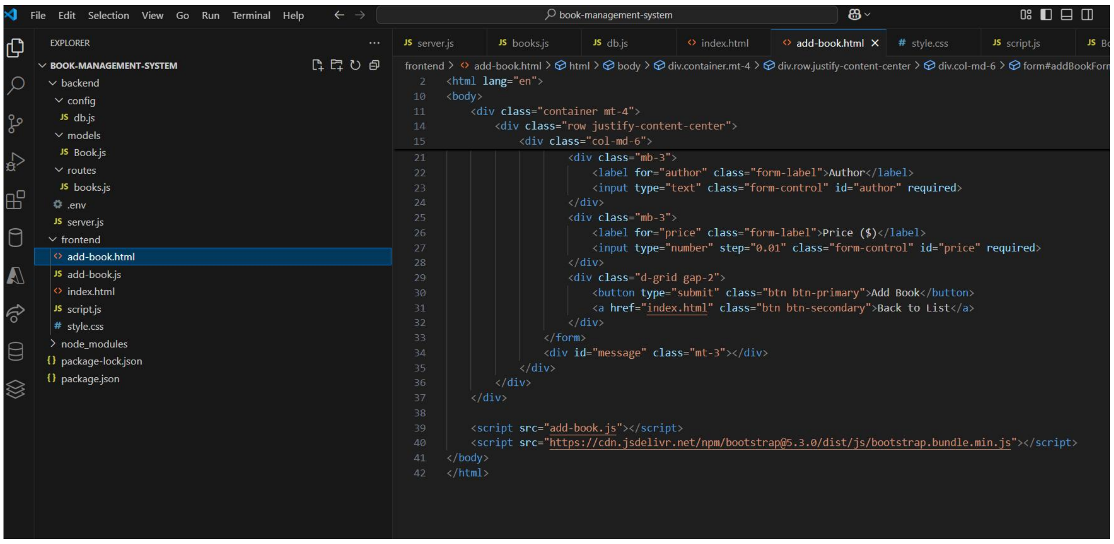
Add-book.html:

A)



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Add New Book</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <div class="container mt-4">
        <h1 class="text-center mb-4">Add New Book</h1>

        <div class="row justify-content-center">
            <div class="col-md-6">
                <form id="addBookForm">
                    <div class="mb-3">
                        <label for="title" class="form-label">Title</label>
                        <input type="text" class="form-control" id="title" required>
                    </div>
                    <div class="mb-3">
                        <label for="author" class="form-label">Author</label>
                        <input type="text" class="form-control" id="author" required>
                    </div>
                    <div class="mb-3">
                        <label for="price" class="form-label">Price ($)</label>
                        <input type="number" step="0.01" class="form-control" id="price" required>
                    </div>
                    <div class="d-grid gap-2">
                        <button type="submit" class="btn btn-primary">Add Book</button>
                        <a href="index.html" class="btn btn-secondary">Back to List</a>
                    </div>
                </form>
                <div id="message" class="mt-3"></div>
            </div>
        </div>
    </div>
```

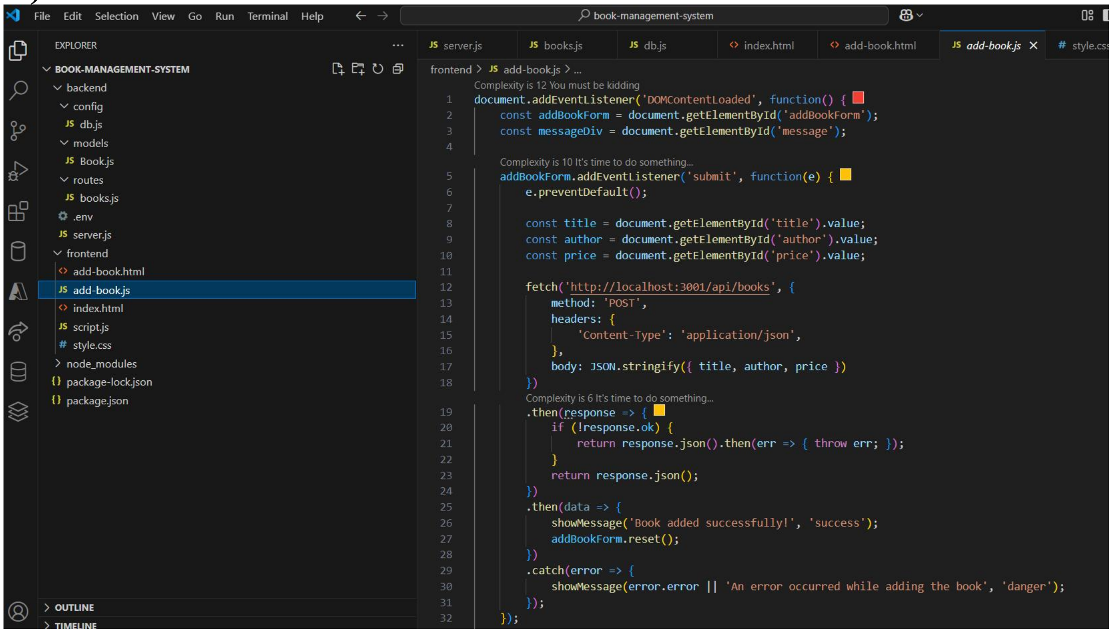
B)



```html
<html lang="en">
<body>
    <div class="container mt-4">
        <div class="row justify-content-center">
            <div class="col-md-6">
                    <div class="mb-3">
                        <label for="author" class="form-label">Author</label>
                        <input type="text" class="form-control" id="author" required>
                    </div>
                    <div class="mb-3">
                        <label for="price" class="form-label">Price ($)</label>
                        <input type="number" step="0.01" class="form-control" id="price" required>
                    </div>
                    <div class="d-grid gap-2">
                        <button type="submit" class="btn btn-primary">Add Book</button>
                        <a href="index.html" class="btn btn-secondary">Back to List</a>
                    </div>
                </form>
                <div id="message" class="mt-3"></div>
            </div>
        </div>
    </div>

    <script src="add-book.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

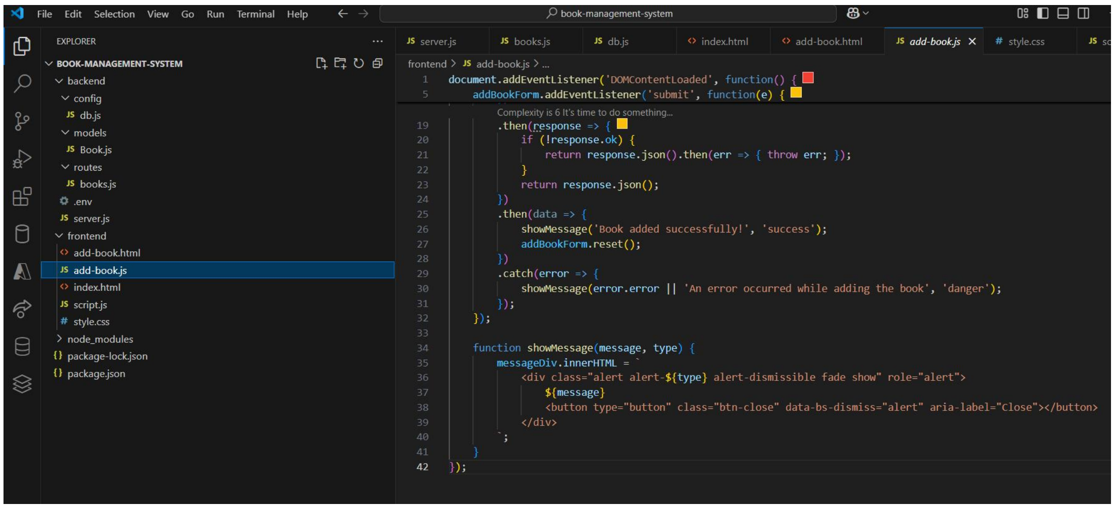
Add-book.js:

A)



```
document.addEventListener('DOMContentLoaded', function() {
    const addBookForm = document.getElementById('addBookForm');
    const messageDiv = document.getElementById('message');

    addBookForm.addEventListener('submit', function(e) {
        e.preventDefault();

        const title = document.getElementById('title').value;
        const author = document.getElementById('author').value;
        const price = document.getElementById('price').value;

        fetch('http://localhost:3001/api/books', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json',
            },
            body: JSON.stringify({ title, author, price })
        })
        .then(response => {
            if (!response.ok) {
                return response.json().then(err => { throw err; });
            }
            return response.json();
        })
        .then(data => {
            showMessage('Book added successfully!', 'success');
            addBookForm.reset();
        })
        .catch(error => {
            showMessage(error.error || 'An error occurred while adding the book', 'danger');
        });
    });
```

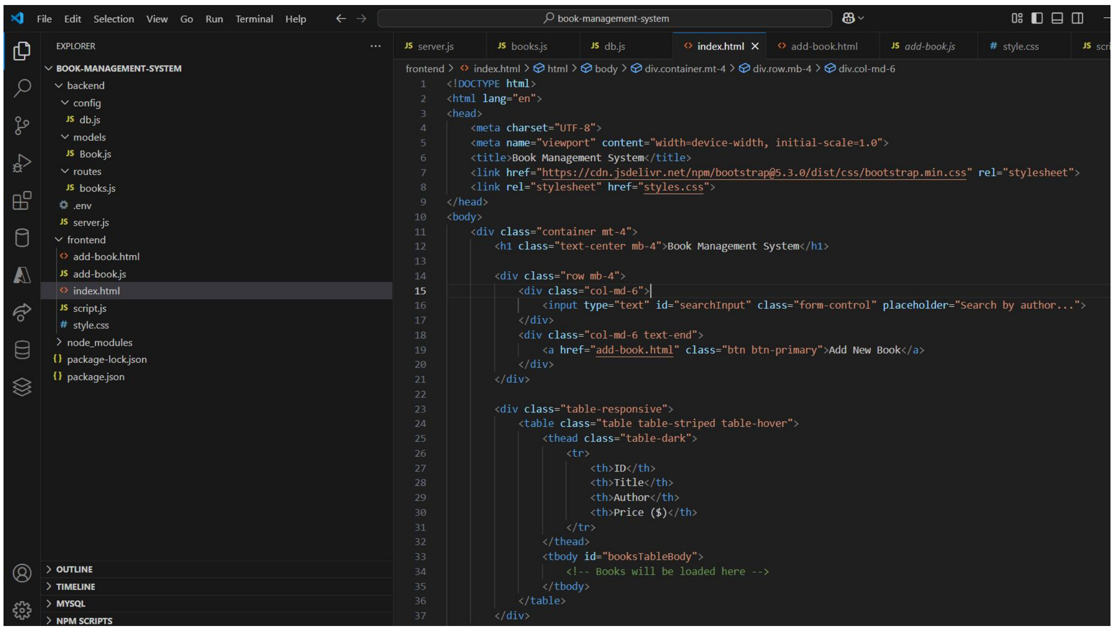
B)



```
document.addEventListener('DOMContentLoaded', function() {
    addBookForm.addEventListener('submit', function(e) {

        .then(response => {
            if (!response.ok) {
                return response.json().then(err => { throw err; });
            }
            return response.json();
        })
        .then(data => {
            showMessage('Book added successfully!', 'success');
            addBookForm.reset();
        })
        .catch(error => {
            showMessage(error.error || 'An error occurred while adding the book', 'danger');
        });
    });

    function showMessage(message, type) {
        messageDiv.innerHTML = `
            <div class="alert alert-${type} alert-dismissible fade show" role="alert">
                ${message}
                <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>
            </div>
        `;
    }
});
```

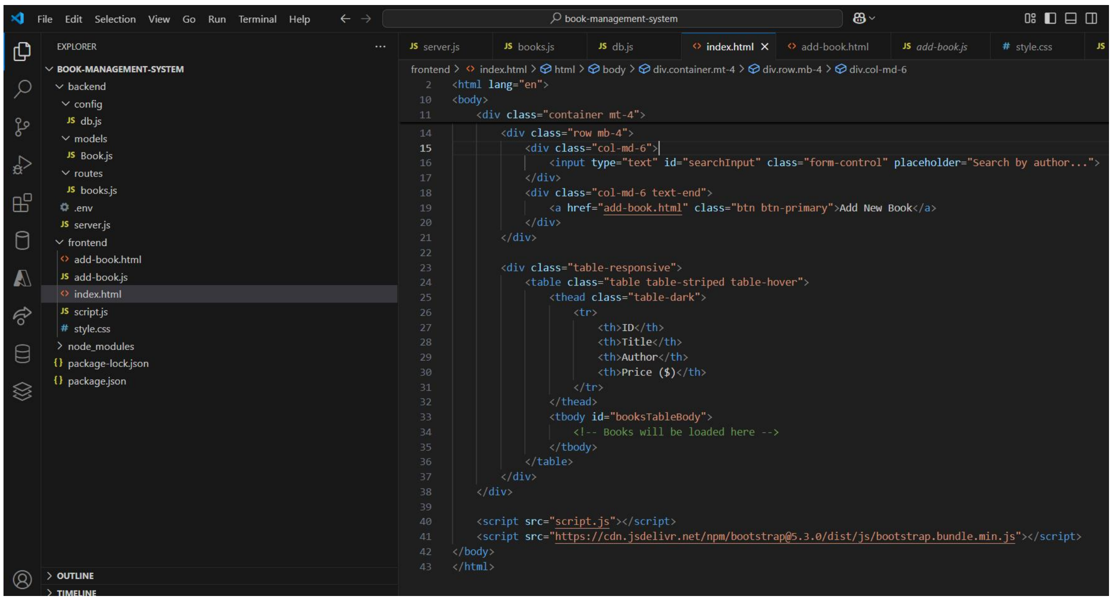
Index.html:

A)



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Book Management System</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <div class="container mt-4">
        <h1 class="text-center mb-4">Book Management System</h1>

        <div class="row mb-4">
            <div class="col-md-6">
                <input type="text" id="searchInput" class="form-control" placeholder="Search by author...">
            </div>
            <div class="col-md-6 text-end">
                <a href="add-book.html" class="btn btn-primary">Add New Book</a>
            </div>
        </div>

        <div class="table-responsive">
            <table class="table table-striped table-hover">
                <thead class="table-dark">
                    <tr>
                        <th>ID</th>
                        <th>Title</th>
                        <th>Author</th>
                        <th>Price ($)</th>
                    </tr>
                </thead>
                <tbody id="booksTableBody">
                    <!-- Books will be loaded here -->
                </tbody>
            </table>
        </div>
```

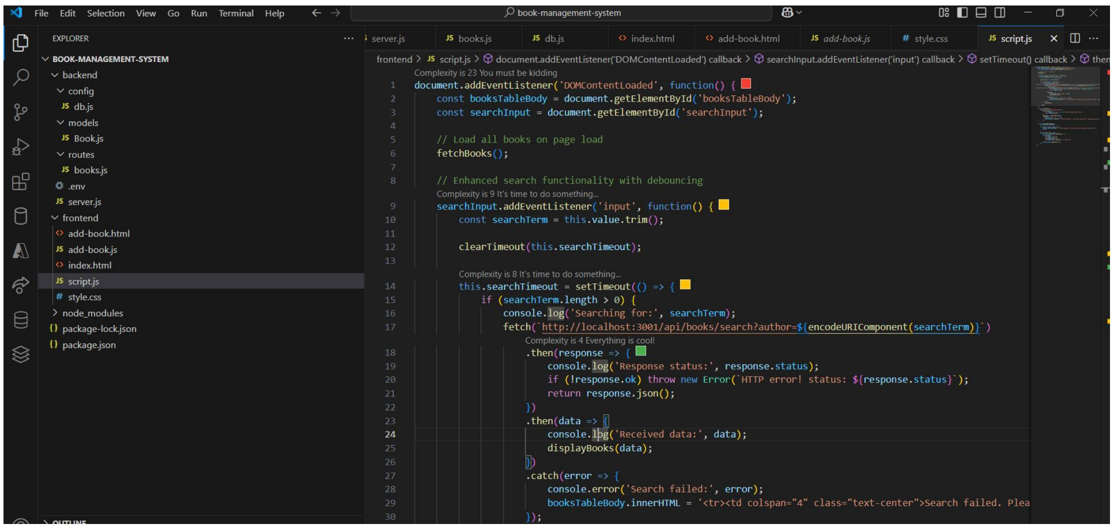
B)



```html
<html lang="en">
<body>
    <div class="container mt-4">
        <div class="row mb-4">
            <div class="col-md-6">
                <input type="text" id="searchInput" class="form-control" placeholder="Search by author...">
            </div>
            <div class="col-md-6 text-end">
                <a href="add-book.html" class="btn btn-primary">Add New Book</a>
            </div>
        </div>

        <div class="table-responsive">
            <table class="table table-striped table-hover">
                <thead class="table-dark">
                    <tr>
                        <th>ID</th>
                        <th>Title</th>
                        <th>Author</th>
                        <th>Price ($)</th>
                    </tr>
                </thead>
                <tbody id="booksTableBody">
                    <!-- Books will be loaded here -->
                </tbody>
            </table>
        </div>
    </div>

    <script src="script.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

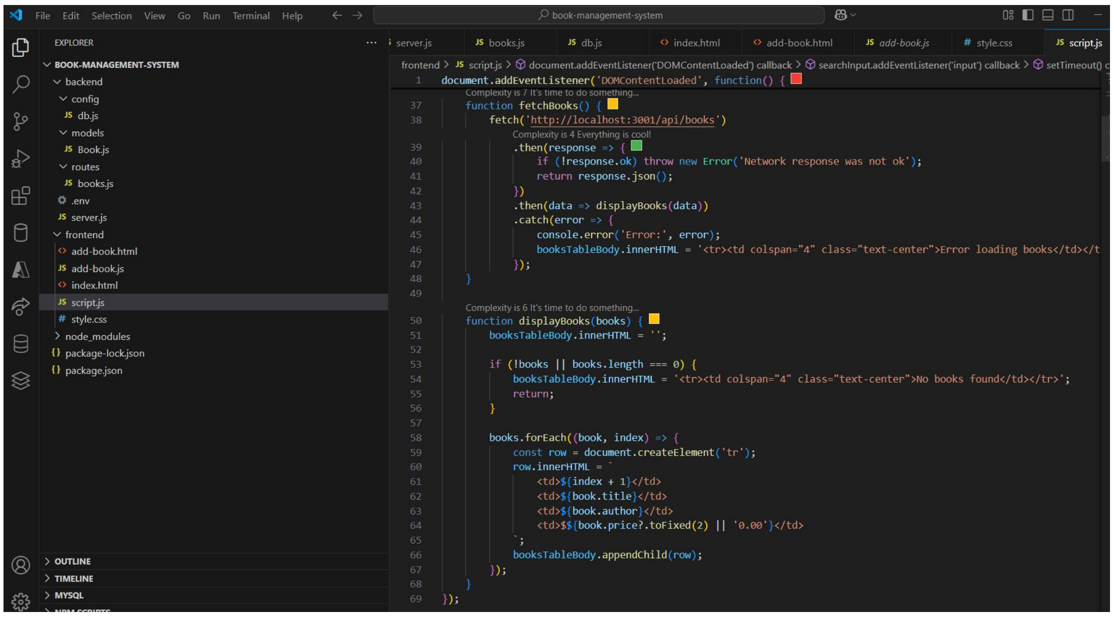
Script.js:

A)



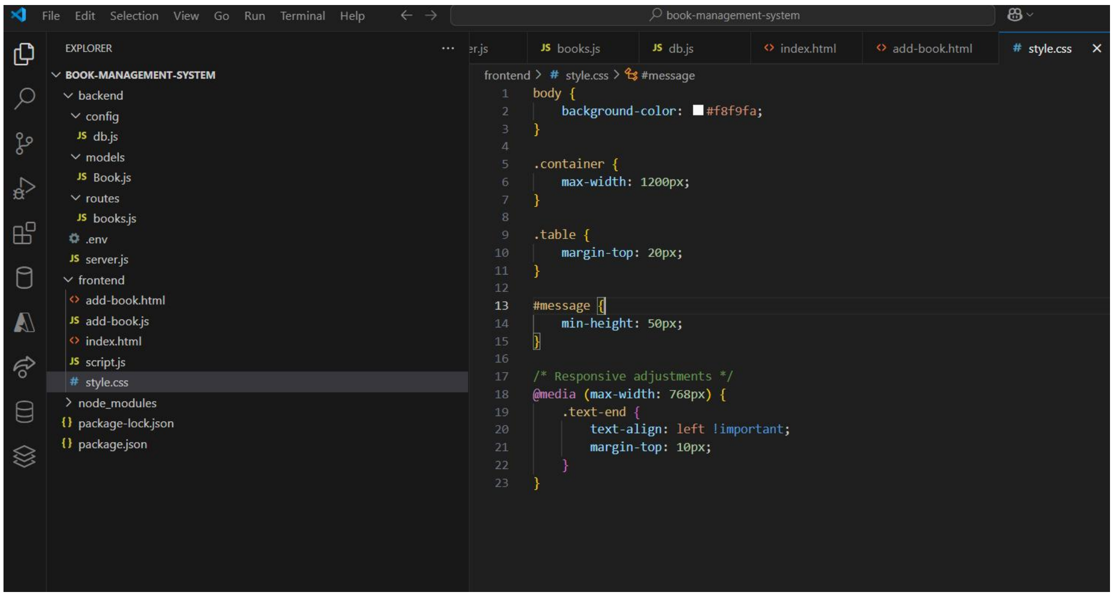
```
Complexity is 23 You must be kidding
document.addEventListener('DOMContentLoaded', function() {
    const booksTableBody = document.getElementById('booksTableBody');
    const searchInput = document.getElementById('searchInput');

    // Load all books on page load
    fetchBooks();

    // Enhanced search functionality with debouncing
    Complexity is 9 It's time to do something...
    searchInput.addEventListener('input', function() {
        const searchTerm = this.value.trim();

        clearTimeout(this.searchTimeout);

        Complexity is 8 It's time to do something...
        this.searchTimeout = setTimeout(() => {
            if (searchTerm.length > 0) {
                console.log('Searching for:', searchTerm);
                fetch(`http://localhost:3001/api/books/search?author=${encodeURIComponent(searchTerm)}`)
                    Complexity is 4 Everything is cool!
                    .then(response => {
                        console.log('Response status:', response.status);
                        if (!response.ok) throw new Error(`HTTP error! status: ${response.status}`);
                        return response.json();
                    })
                    .then(data => {
                        console.log('Received data:', data);
                        displayBooks(data);
                    })
                    .catch(error => {
                        console.error('Search failed:', error);
                        booksTableBody.innerHTML = '<tr><td colspan="4" class="text-center">Search failed. Plea
                    });
```

B)



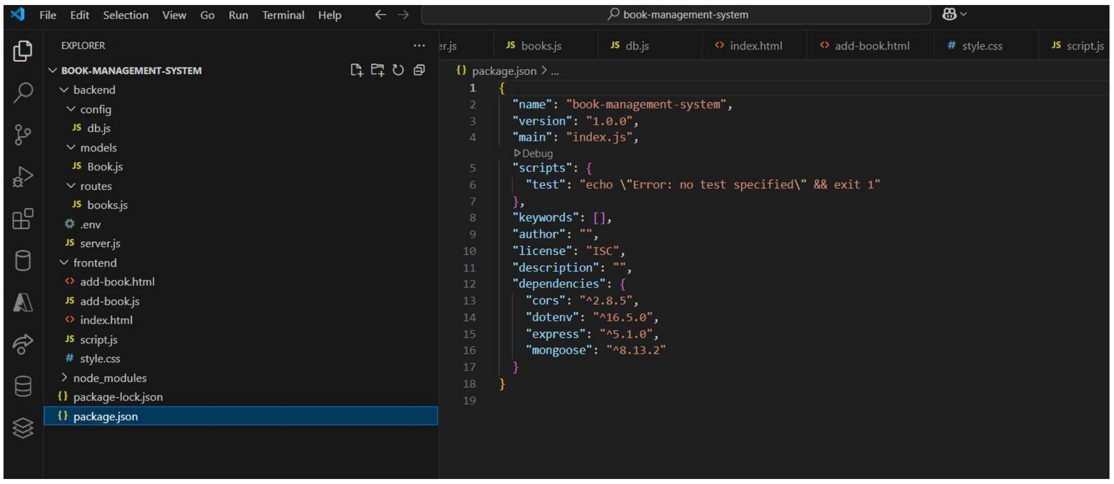
```
document.addEventListener('DOMContentLoaded', function() {
    Complexity is 7 It's time to do something...
    function fetchBooks() {
        fetch('http://localhost:3001/api/books')
            Complexity is 4 Everything is cool!
            .then(response => {
                if (!response.ok) throw new Error('Network response was not ok');
                return response.json();
            })
            .then(data => displayBooks(data))
            .catch(error => {
                console.error('Error:', error);
                booksTableBody.innerHTML = '<tr><td colspan="4" class="text-center">Error loading books</td></t
            });
    }

    Complexity is 6 It's time to do something...
    function displayBooks(books) {
        booksTableBody.innerHTML = '';

        if (!books || books.length === 0) {
            booksTableBody.innerHTML = '<tr><td colspan="4" class="text-center">No books found</td></tr>';
            return;
        }

        books.forEach((book, index) => {
            const row = document.createElement('tr');
            row.innerHTML = `
                <td>${index + 1}</td>
                <td>${book.title}</td>
                <td>${book.author}</td>
                <td>$${book.price?.toFixed(2) || '0.00'}</td>
            `;
            booksTableBody.appendChild(row);
        });
    }
});
```

Styles.css:



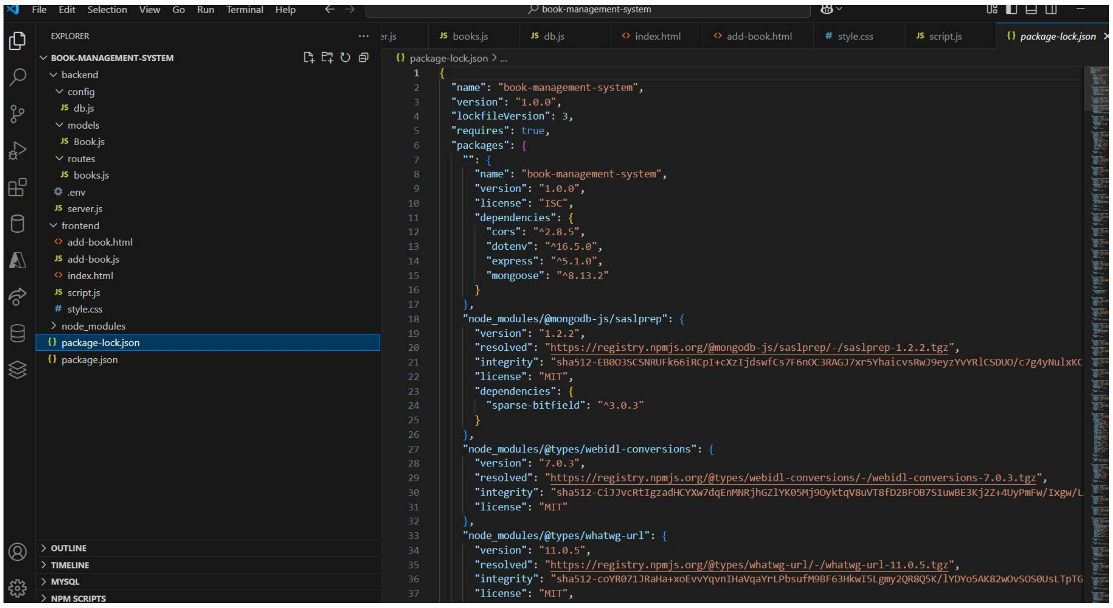
```css
body {
    background-color: #f8f9fa;
}

.container {
    max-width: 1200px;
}

.table {
    margin-top: 20px;
}

#message {
    min-height: 50px;
}

/* Responsive adjustments */
@media (max-width: 768px) {
    .text-end {
        text-align: left !important;
        margin-top: 10px;
    }
}
```

Package.json:



```json
{
  "name": "book-management-system",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "cors": "^2.8.5",
    "dotenv": "^16.5.0",
    "express": "^5.1.0",
    "mongoose": "^8.13.2"
  }
}
```

Package-Lock.json:

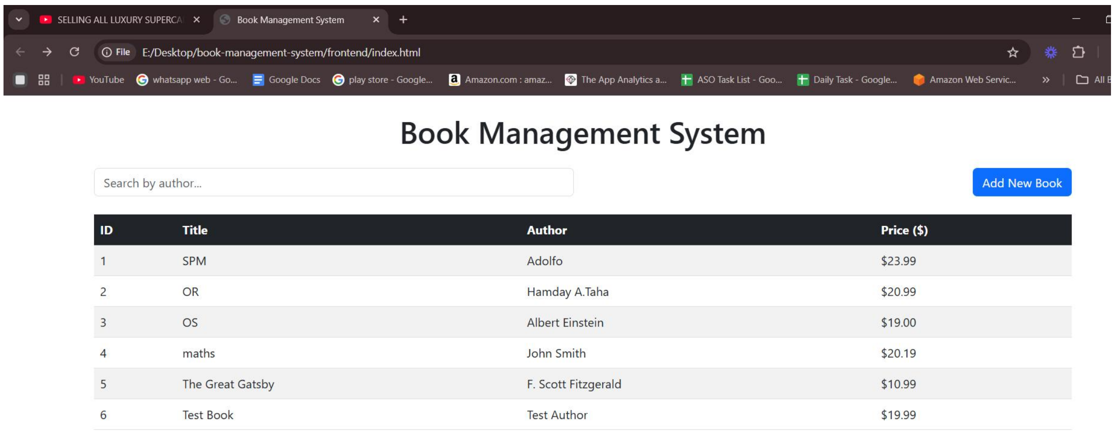


```json
{
  "name": "book-management-system",
  "version": "1.0.0",
  "lockfileVersion": 3,
  "requires": true,
  "packages": {
    "": {
      "name": "book-management-system",
      "version": "1.0.0",
      "license": "ISC",
      "dependencies": {
        "cors": "^2.8.5",
        "dotenv": "^16.5.0",
        "express": "^5.1.0",
        "mongoose": "^8.13.2"
      }
    },
    "node_modules/@mongodb-js/saslprep": {
      "version": "1.2.2",
      "resolved": "https://registry.npmjs.org/@mongodb-js/saslprep/-/saslprep-1.2.2.tgz",
      "integrity": "sha512-EB0O3SCSNRUFk66iRCpI+cXzIjdswfCs7F6nOC3RAGJ7xr5YhaicvsRwJ9eyzYvYRlCSDUO/c7g4yNulxKC
      "license": "MIT",
      "dependencies": {
        "sparse-bitfield": "^3.0.3"
      }
    },
    "node_modules/@types/webidl-conversions": {
      "version": "7.0.3",
      "resolved": "https://registry.npmjs.org/@types/webidl-conversions/-/webidl-conversions-7.0.3.tgz",
      "integrity": "sha512-CiJJvcRtIgzadHCYXw7dqEnMNRjhGZlYK05Mj9OyktqV8uVT8fD2BFOB7S1uwBE3Kj2Z+4UyPmFw/Ixgw/L
      "license": "MIT"
    },
    "node_modules/@types/whatwg-url": {
      "version": "11.0.5",
      "resolved": "https://registry.npmjs.org/@types/whatwg-url/-/whatwg-url-11.0.5.tgz",
      "integrity": "sha512-coYR071JRaHa+xoEvvYqvnIHaVqaYrLPbsufM9BF63HkwI5Lgmy2QR8Q5K/lYDYo5AK82wOvSOS0UsLTpTG
      "license": "MIT",
```

Output

Frontend:

Index Page:

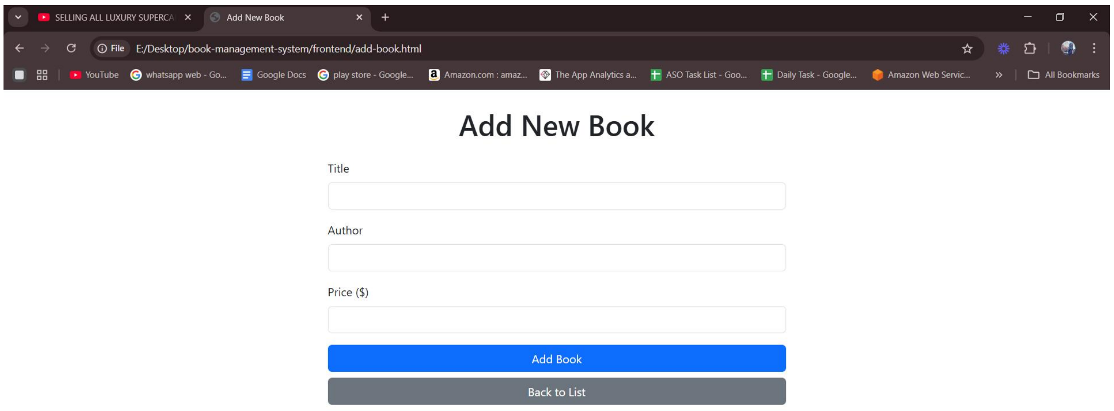


Book Management System

ID	Title	Author	Price ($)
1	SPM	Adolfo	$23.99
2	OR	Hamday A.Taha	$20.99
3	OS	Albert Einstein	$19.00
4	maths	John Smith	$20.19
5	The Great Gatsby	F. Scott Fitzgerald	$10.99
6	Test Book	Test Author	$19.99

Add Book Page:

A)



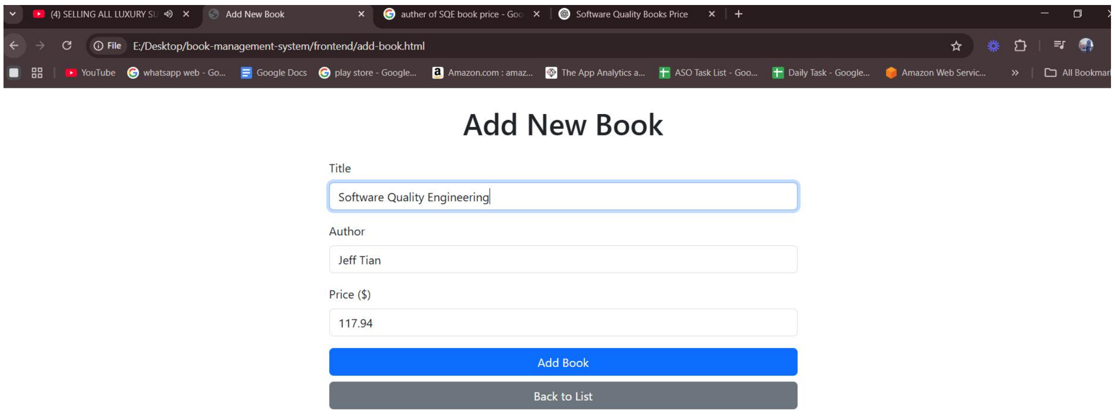
Add New Book

Title

Author

Price ($)

B)



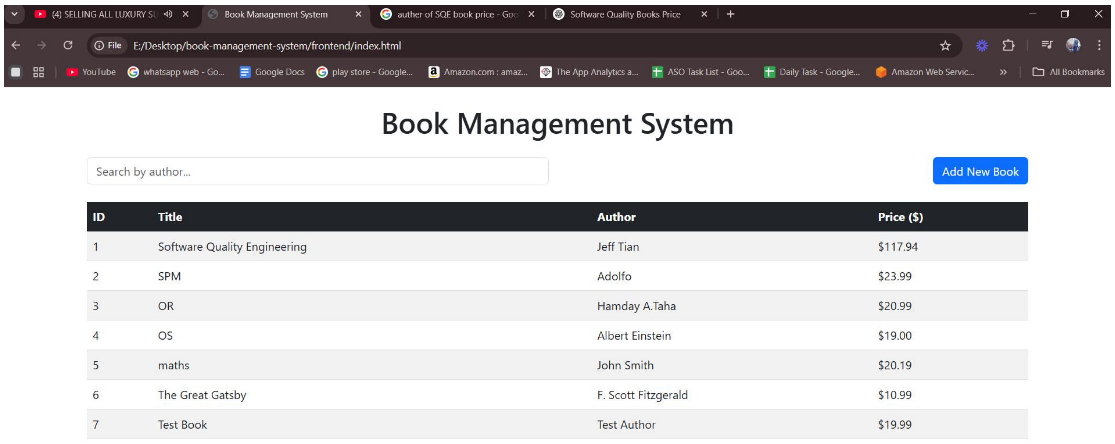
Add New Book

Title: Software Quality Engineering

Author: Jeff Tian

Price ($): 117.94

Index Page while new book add:



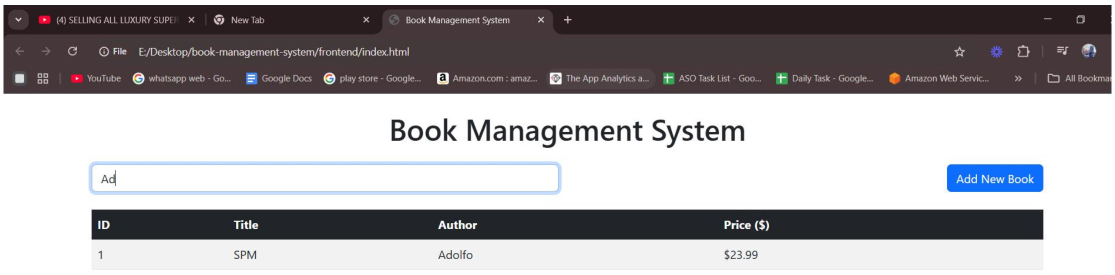
Book Management System

Search by author...

Add New Book

ID	Title	Author	Price ($)
1	Software Quality Engineering	Jeff Tian	$117.94
2	SPM	Adolfo	$23.99
3	OR	Hamday A.Taha	$20.99
4	OS	Albert Einstein	$19.00
5	maths	John Smith	$20.19
6	The Great Gatsby	F. Scott Fitzgerald	$10.99
7	Test Book	Test Author	$19.99

Search Page:



Book Management System

Ad

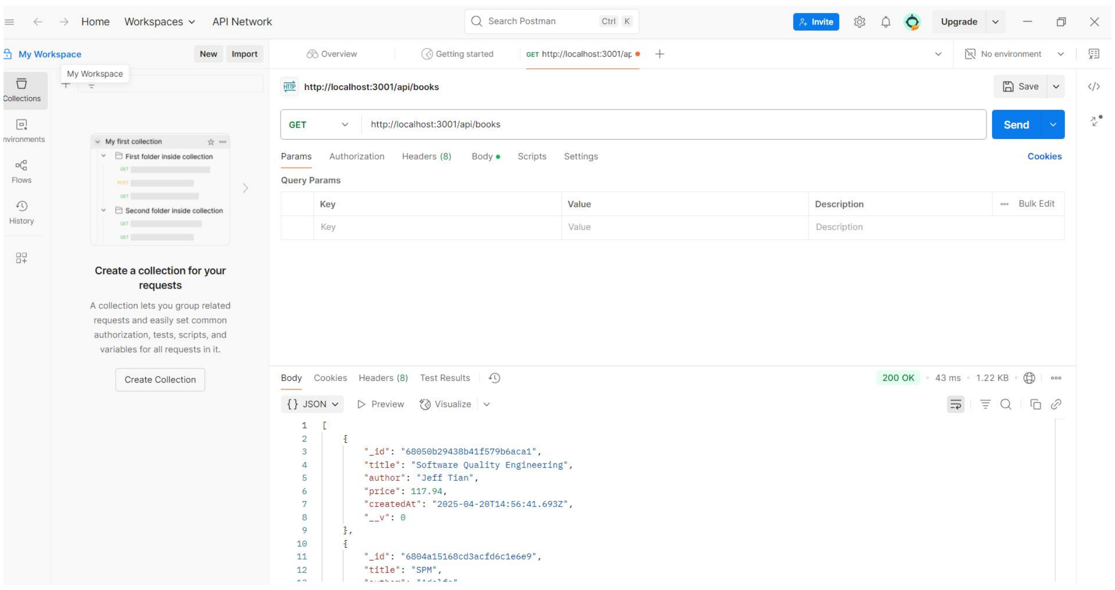
Add New Book

ID	Title	Author	Price ($)
1	SPM	Adolfo	$23.99

Backend

Get Method:

A)

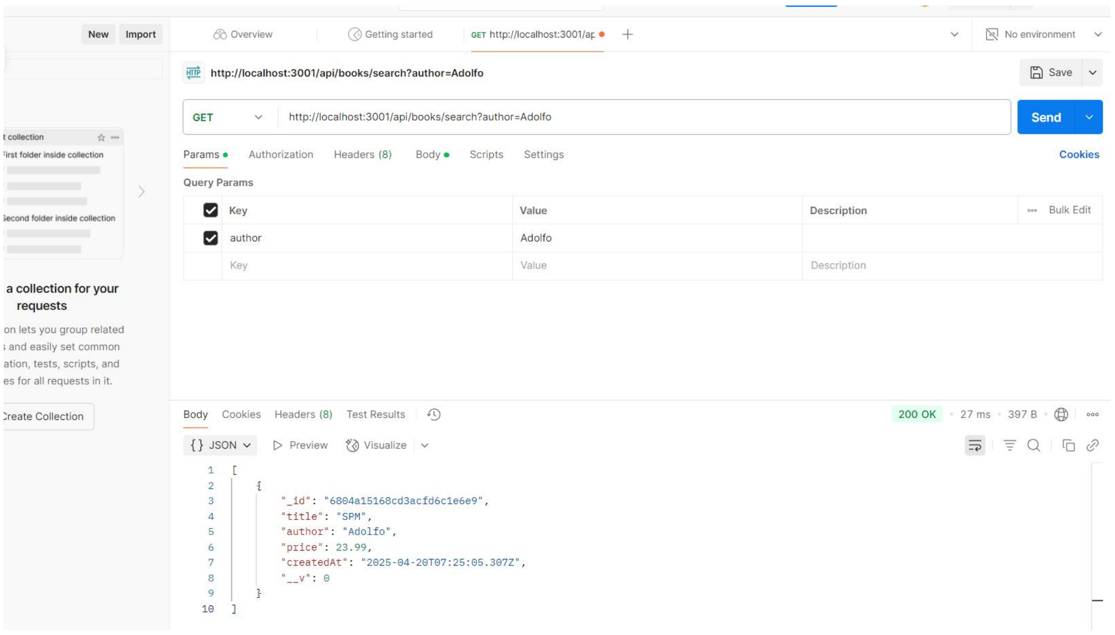


GET http://localhost:3001/api/books

200 OK · 43 ms · 1.22 KB

```
[
    {
        "_id": "68050b29438b41f579b6aca1",
        "title": "Software Quality Engineering",
        "author": "Jeff Tian",
        "price": 117.94,
        "createdAt": "2025-04-20T14:56:41.693Z",
        "__v": 0
    },
    {
        "_id": "6804a15168cd3acfd6c1e6e9",
        "title": "SPM",
```

B)



GET http://localhost:3001/api/books/search?author=Adolfo

Query Params

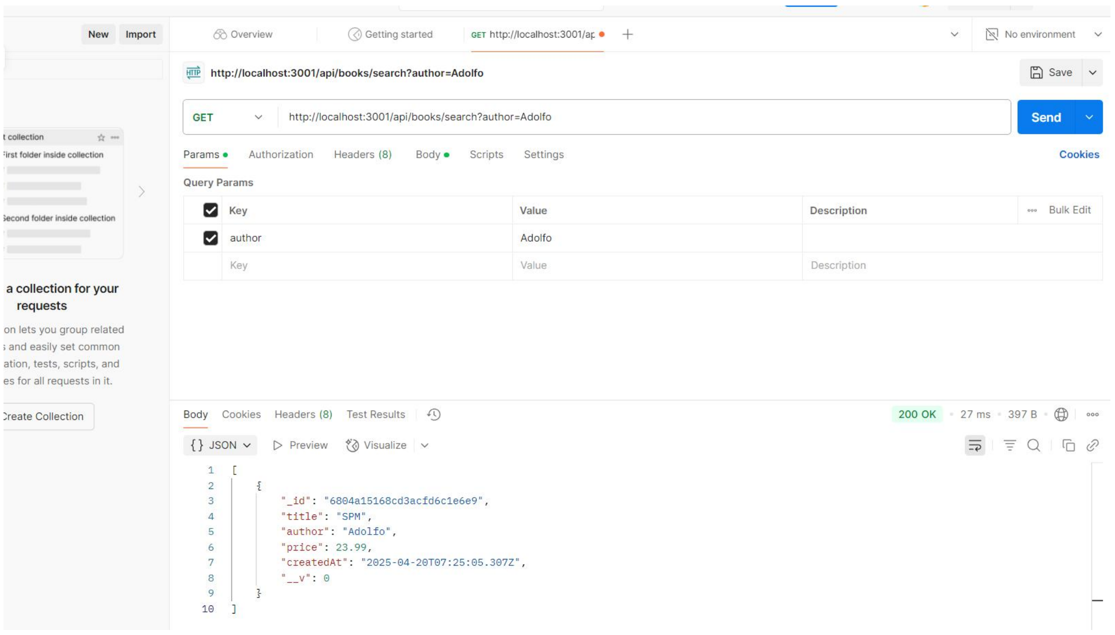
Key	Value	Description
author	Adolfo	

200 OK · 27 ms · 397 B

```
[
    {
        "_id": "6804a15168cd3acfd6c1e6e9",
        "title": "SPM",
        "author": "Adolfo",
        "price": 23.99,
        "createdAt": "2025-04-20T07:25:05.307Z",
        "__v": 0
    }
]
```

C)

Get method by Search:



GET http://localhost:3001/api/books/search?author=Adolfo

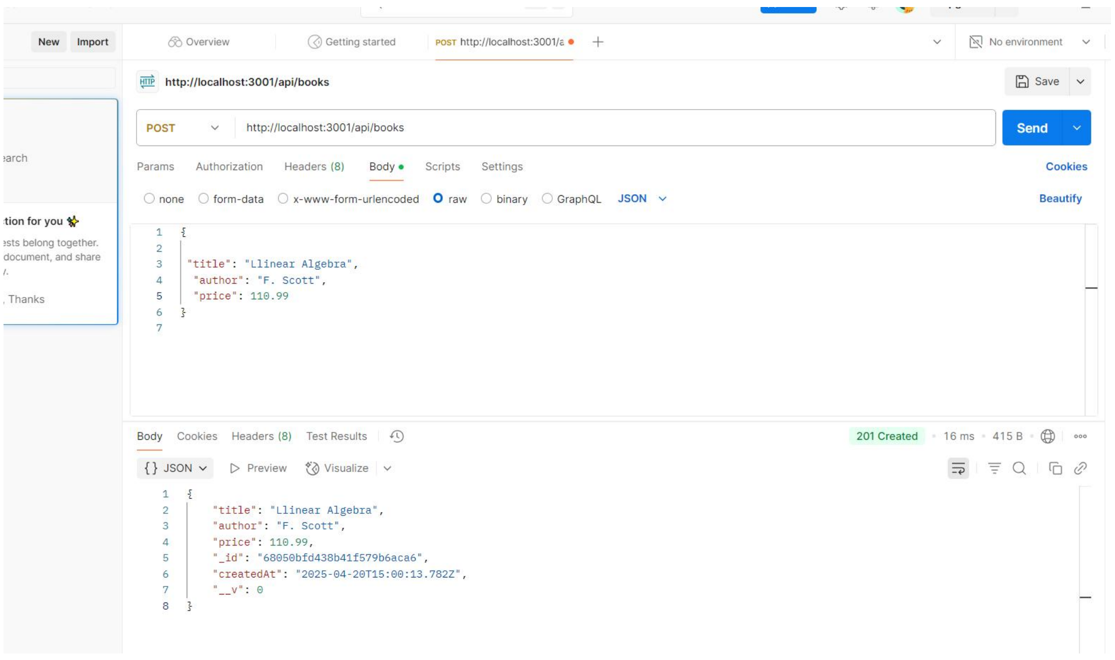
Query Params

Key	Value	Description
author	Adolfo	

200 OK · 27 ms · 397 B

```
[
    {
        "_id": "6804a15168cd3acfd6c1e6e9",
        "title": "SPM",
        "author": "Adolfo",
        "price": 23.99,
        "createdAt": "2025-04-20T07:25:05.307Z",
        "__v": 0
    }
]
```

Post Method:



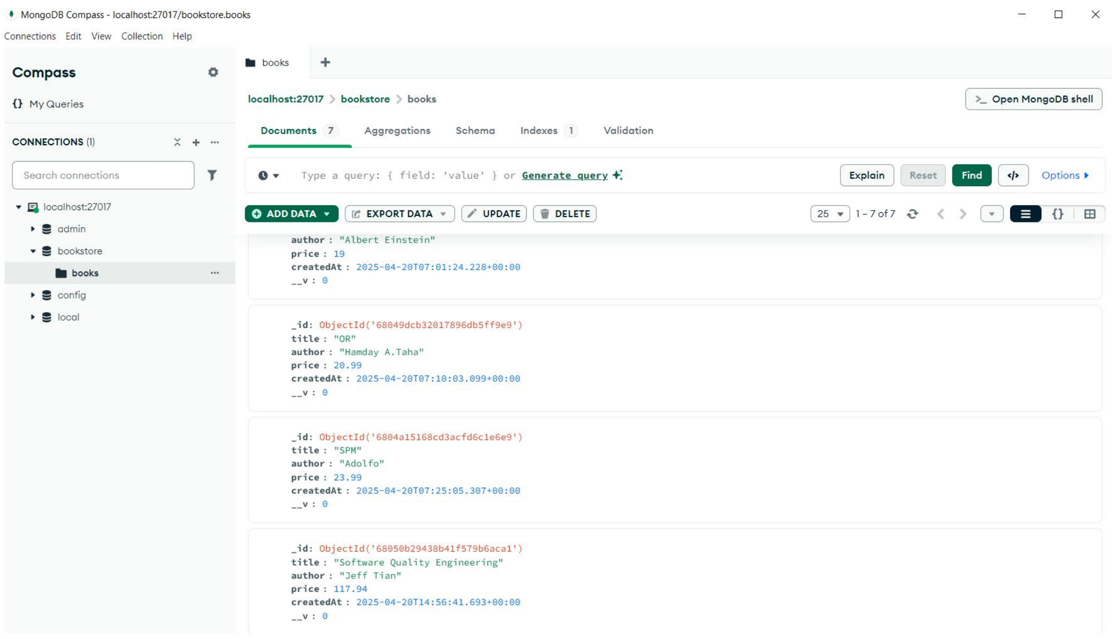
POST http://localhost:3001/api/books

```
{

"title": "Llinear Algebra",
 "author": "F. Scott",
 "price": 110.99
}

```

201 Created · 16 ms · 415 B

```
{
    "title": "Llinear Algebra",
    "author": "F. Scott",
    "price": 110.99,
    "_id": "68050bfd438b41f579b6aca6",
    "createdAt": "2025-04-20T15:00:13.782Z",
    "__v": 0
}
```

Mongodb



```
author : "Albert Einstein"
price : 19
createdAt : 2025-04-20T07:01:24.228+00:00
__v : 0

_id: ObjectId('68049dcb32017896db5ff9e9')
title : "OR"
author : "Hamday A.Taha"
price : 20.99
createdAt : 2025-04-20T07:10:03.099+00:00
__v : 0

_id: ObjectId('6804a15168cd3acfd6c1e6e9')
title : "SPM"
author : "Adolfo"
price : 23.99
createdAt : 2025-04-20T07:25:05.307+00:00
__v : 0

_id: ObjectId('68050b29438b41f579b6aca1')
title : "Software Quality Engineering"
author : "Jeff Tian"
price : 117.94
createdAt : 2025-04-20T14:56:41.693+00:00
__v : 0
```