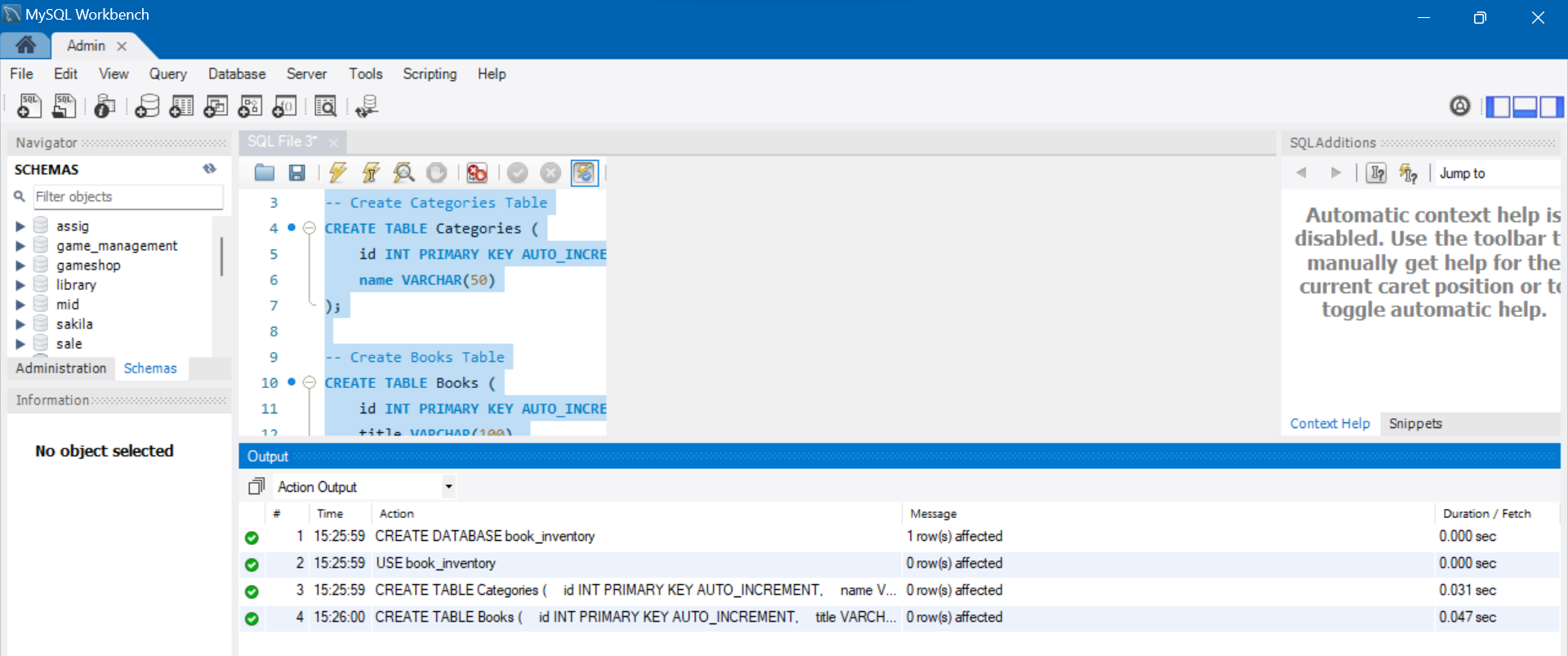
Documentation:

1. Open your terminal and navigate to the folder where you want to create your project.
2. Run the following command:

npm init –y

npm install express mysql2 sequelize

Open MYSQL login and run the script below.



CREATE DATABASE book\_inventory;

USE book\_inventory;

-- Create Categories Table

CREATE TABLE Categories (

id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(50)

);

-- Create Books Table

CREATE TABLE Books (

id INT PRIMARY KEY AUTO\_INCREMENT,

title VARCHAR(100),

author VARCHAR(100),

category\_id INT,

FOREIGN KEY (category\_id) REFERENCES Categories(id)

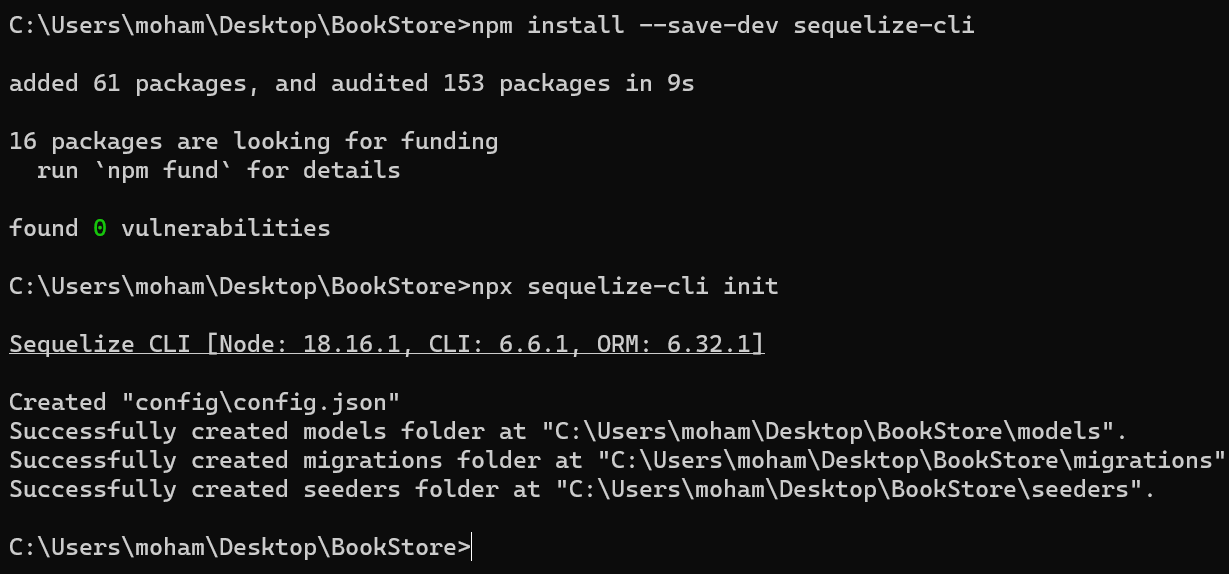
);

RUN THIS:

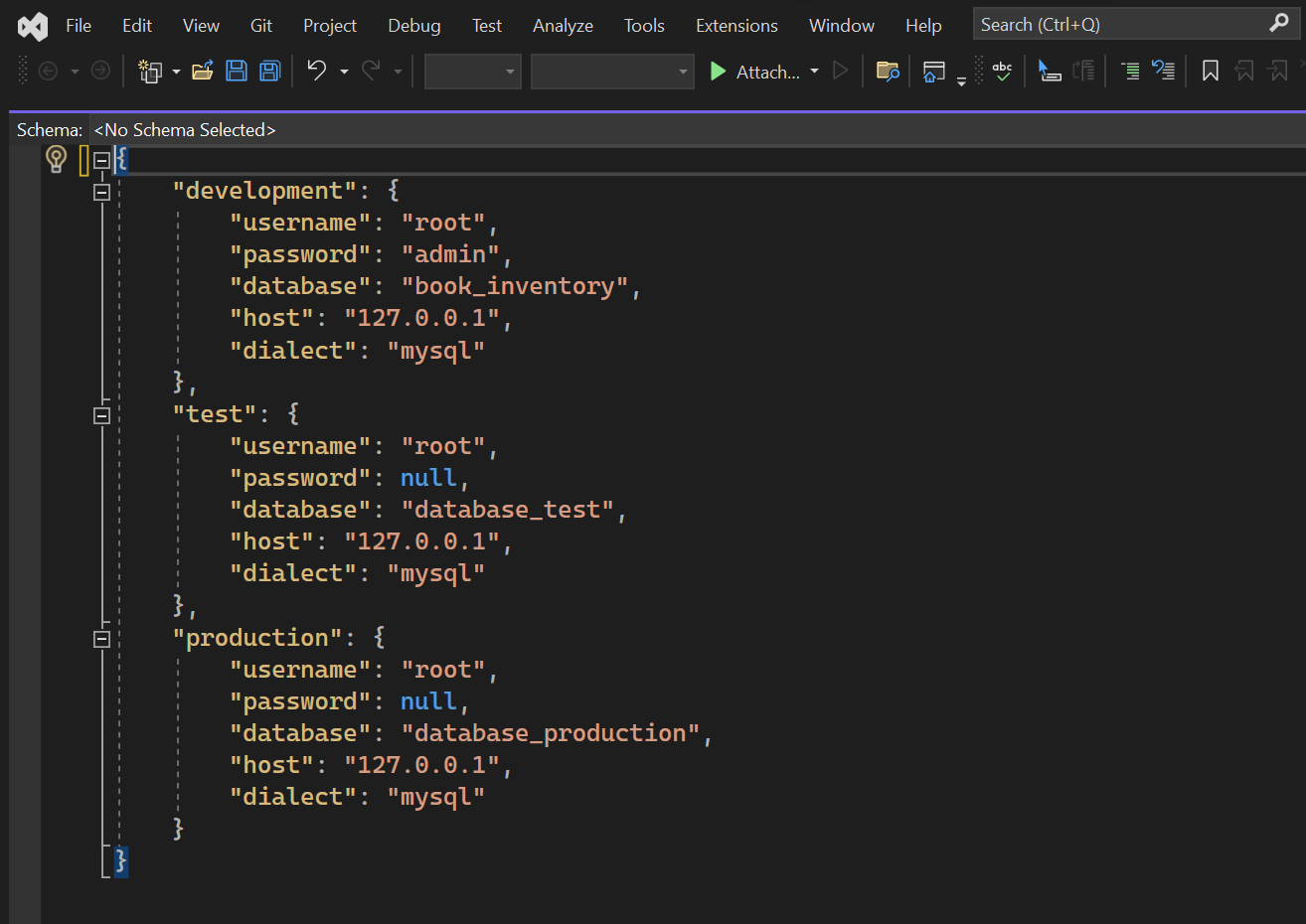
**npm install --save-dev sequelize-cli**

**npm install cors**

**npx sequelize-cli init**



Edit your config.json like this:



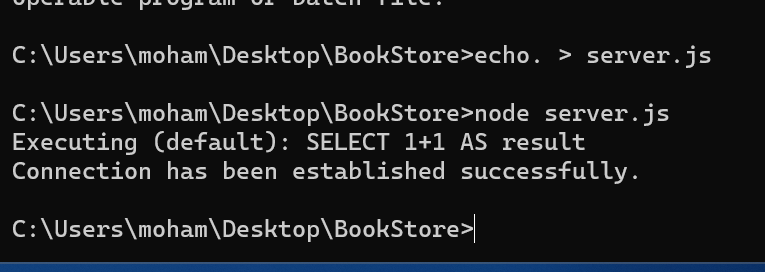
Username, password and database name would be of your own system. Mine is root, admin and book\_in..

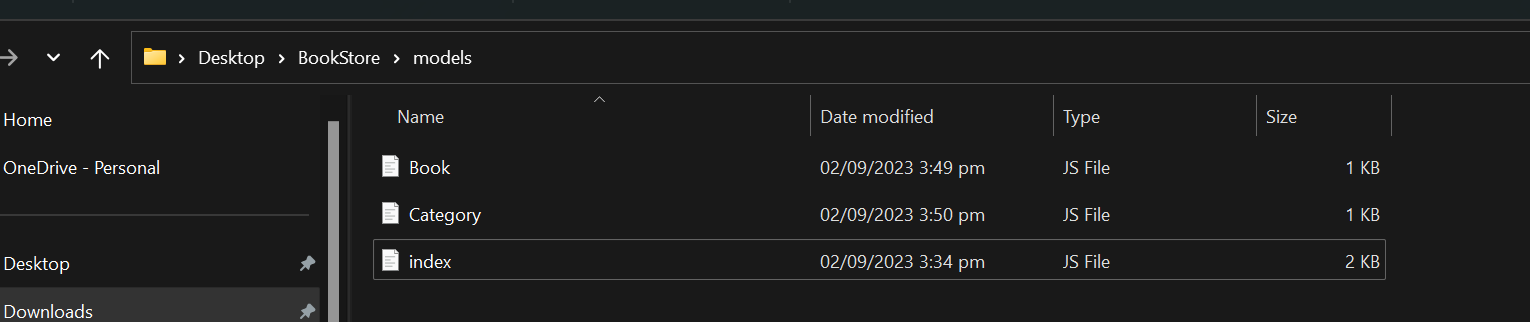
Create server.js manually (in bookstore main directory) or by running

echo. > server.js

Run this:

node server.js



Create these: 

Book.js:

*const* { Sequelize, DataTypes, Model } = require('sequelize');

*const* sequelize = new Sequelize('book\_inventory', 'root', 'admin', {

  host: 'localhost',

  dialect: 'mysql'

});

class Book *extends* Model {}

Book.init({

  title: {

    type: DataTypes.STRING,

    allowNull: false

  },

  author: {

    type: DataTypes.STRING,

    allowNull: false

  },

  category\_id: {

    type: DataTypes.INTEGER,

    references: {

      model: 'Categories', *// name of Target model*

      key: 'id', *// key in Target model that we're referencing*

    },

    allowNull: false

  },

}, {

  sequelize, *// passing the `sequelize` instance is required*

  modelName: 'Book', *// We need to choose the model name*

  timestamps: false  *// disable timestamps*

});

module.exports = Book;

Category.js:

*const* { Sequelize, DataTypes, Model } = require('sequelize');

*const* sequelize = new Sequelize('book\_inventory', 'root', 'admin', {

  host: 'localhost',

  dialect: 'mysql'

});

class Category *extends* Model {}

Category.init({

  name: {

    type: DataTypes.STRING,

    allowNull: false

  }

}, {

  sequelize,

  modelName: 'Category',

  timestamps: false  *// disable timestamps*

});

module.exports = Category;

NOW API:

Update server.js:

*const* express = require('express');

*const* cors = require('cors');  *// Import the CORS package*

*const* { Sequelize } = require('sequelize');

*const* Book = require('./models/Book');

*const* Category = require('./models/Category');

*const* app = express();

*const* PORT = 3000;

*// Enable CORS for all routes*

app.use(cors());

*// Middleware to parse JSON bodies*

app.use(express.json());

*// Test database connection*

*const* sequelize = new Sequelize('book\_inventory', 'root', 'admin', {

  host: 'localhost',

  dialect: 'mysql'

});

sequelize.authenticate()

  .then(() => {

    console.log('Connection has been established successfully.');

  })

  .catch(err => {

    console.error('Unable to connect to the database:', err);

  });

*// Import routes*

*const* bookRoutes = require('./routes/bookRoutes');

*const* categoryRoutes = require('./routes/categoryRoutes');

*// Use routes*

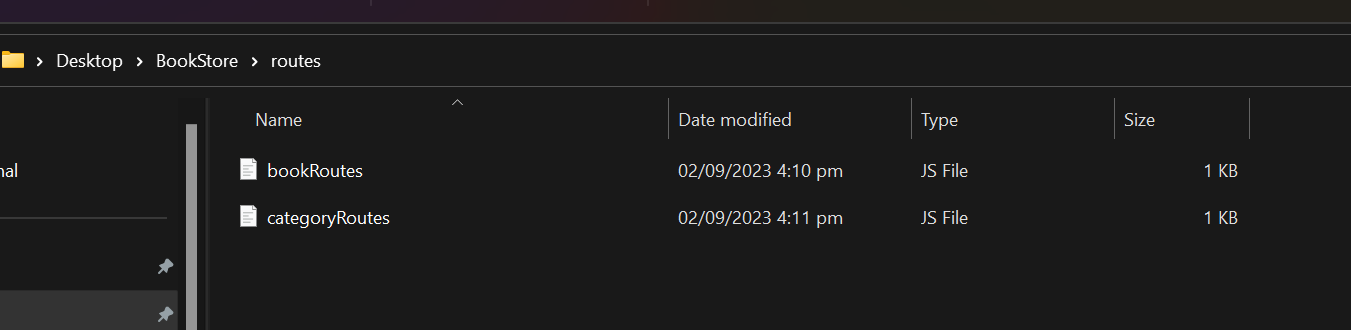
app.use('/api/books', bookRoutes);

app.use('/api/categories', categoryRoutes);

app.listen(PORT, () => {

  console.log(`Server is running on port ${PORT}`);

});



New folder routes with:

bookRoutes.js:

const express = require('express');

const Book = require('../models/Book');

const router = express.Router();

// Get all books

router.get('/', async (req, res) => {

const books = await Book.findAll();

res.json(books);

});

// Add a new book

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

const { title, author, category\_id } = req.body;

const newBook = await Book.create({ title, author, category\_id });

res.json(newBook);

});

module.exports = router;

categoryRoutes:

const express = require('express');

const Category = require('../models/Category');

const router = express.Router();

// Get all categories

router.get('/', async (req, res) => {

const categories = await Category.findAll();

res.json(categories);

});

// Add a new category

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

const { name } = req.body;

const newCategory = await Category.create({ name });

res.json(newCategory);

});

module.exports = router;

Create seedDatabase.js in directory:

const { Sequelize } = require('sequelize');

const Book = require('./models/Book');

const Category = require('./models/Category');

// Initialize Sequelize

const sequelize = new Sequelize('book\_inventory', 'root', 'admin', {

host: 'localhost',

dialect: 'mysql'

});

// Seed Categories

const seedCategories = async () => {

const categories = [

{ name: 'Fiction' },

{ name: 'Non-Fiction' },

{ name: 'Science' },

];

for (const category of categories) {

await Category.create(category);

}

};

// Seed Books

const seedBooks = async () => {

const books = [

{ title: 'To Kill a Mockingbird', author: 'Harper Lee', category\_id: 1 },

{ title: '1984', author: 'George Orwell', category\_id: 1 },

{ title: 'A Brief History of Time', author: 'Stephen Hawking', category\_id: 3 },

];

for (const book of books) {

await Book.create(book);

}

};

// Run the seeder

const runSeeder = async () => {

await sequelize.sync();

await seedCategories();

await seedBooks();

console.log('Database seeded!');

process.exit();

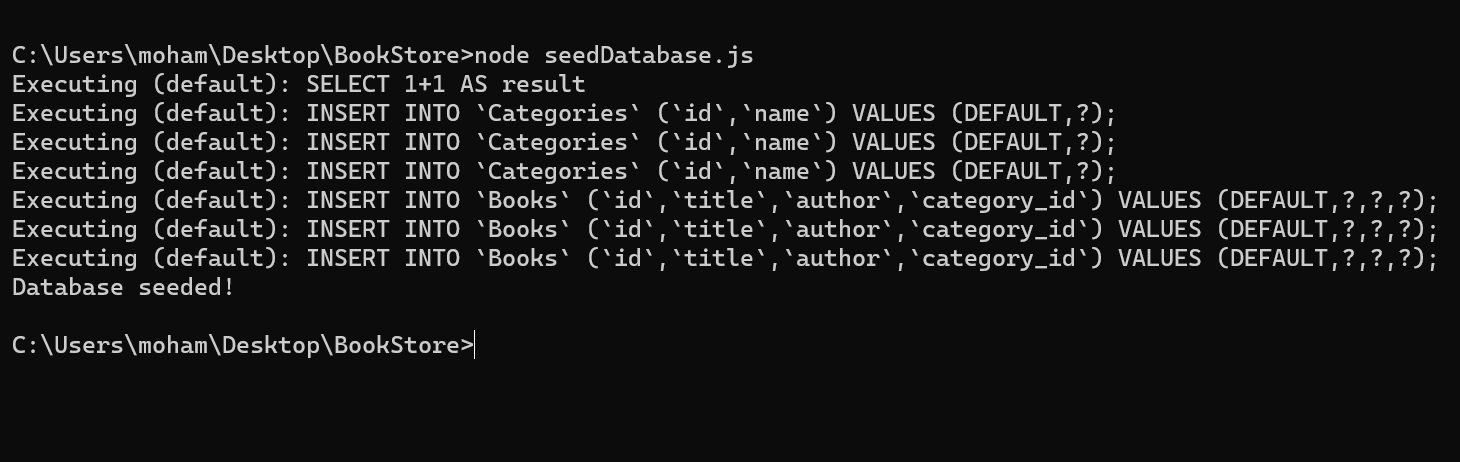
};

runSeeder().catch((error) => {

console.error('Seeding failed:', error);

process.exit();

});



Run:

node seedDatabase.js

Open postman

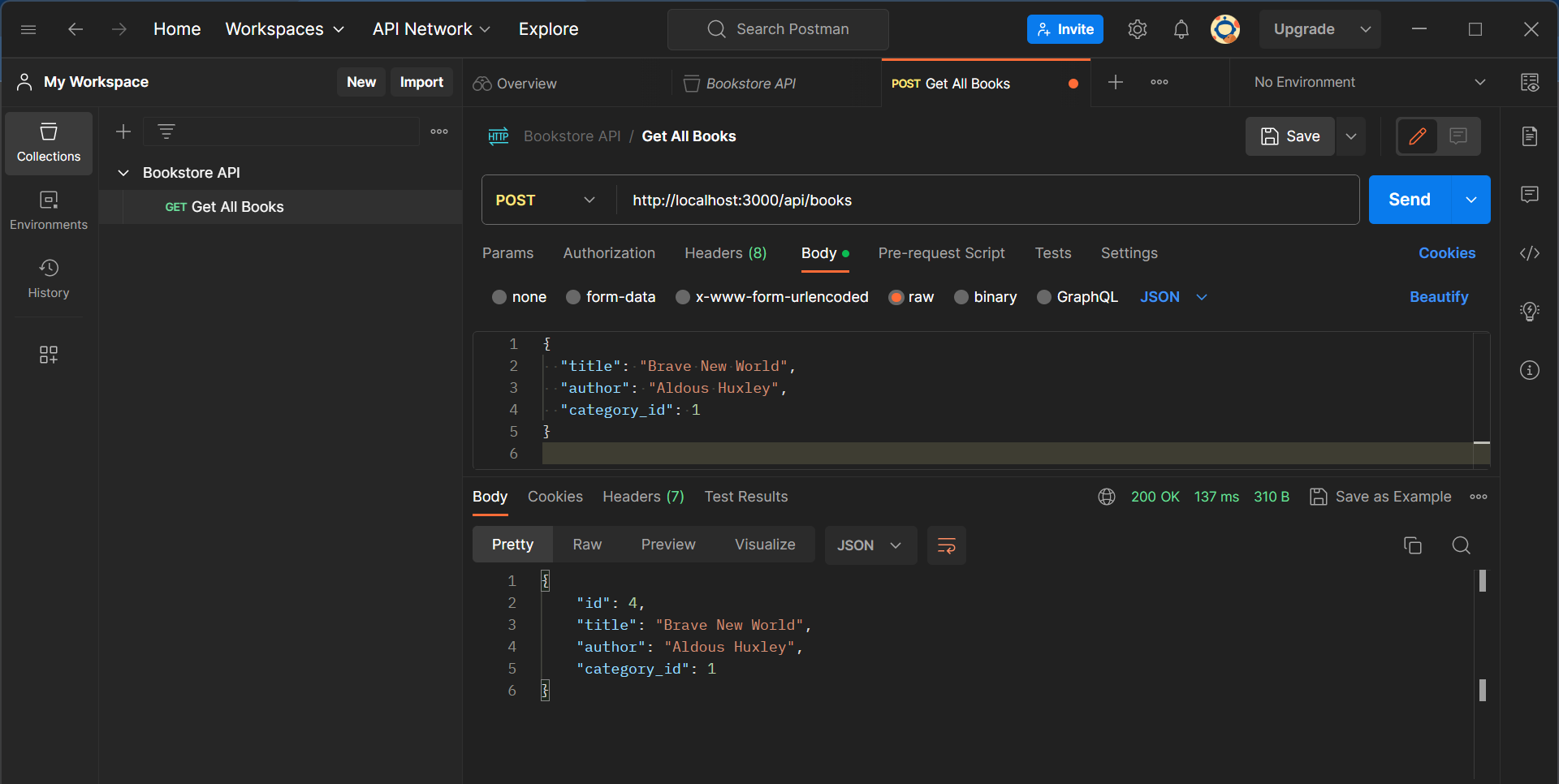
Create a new file

Keep post id :

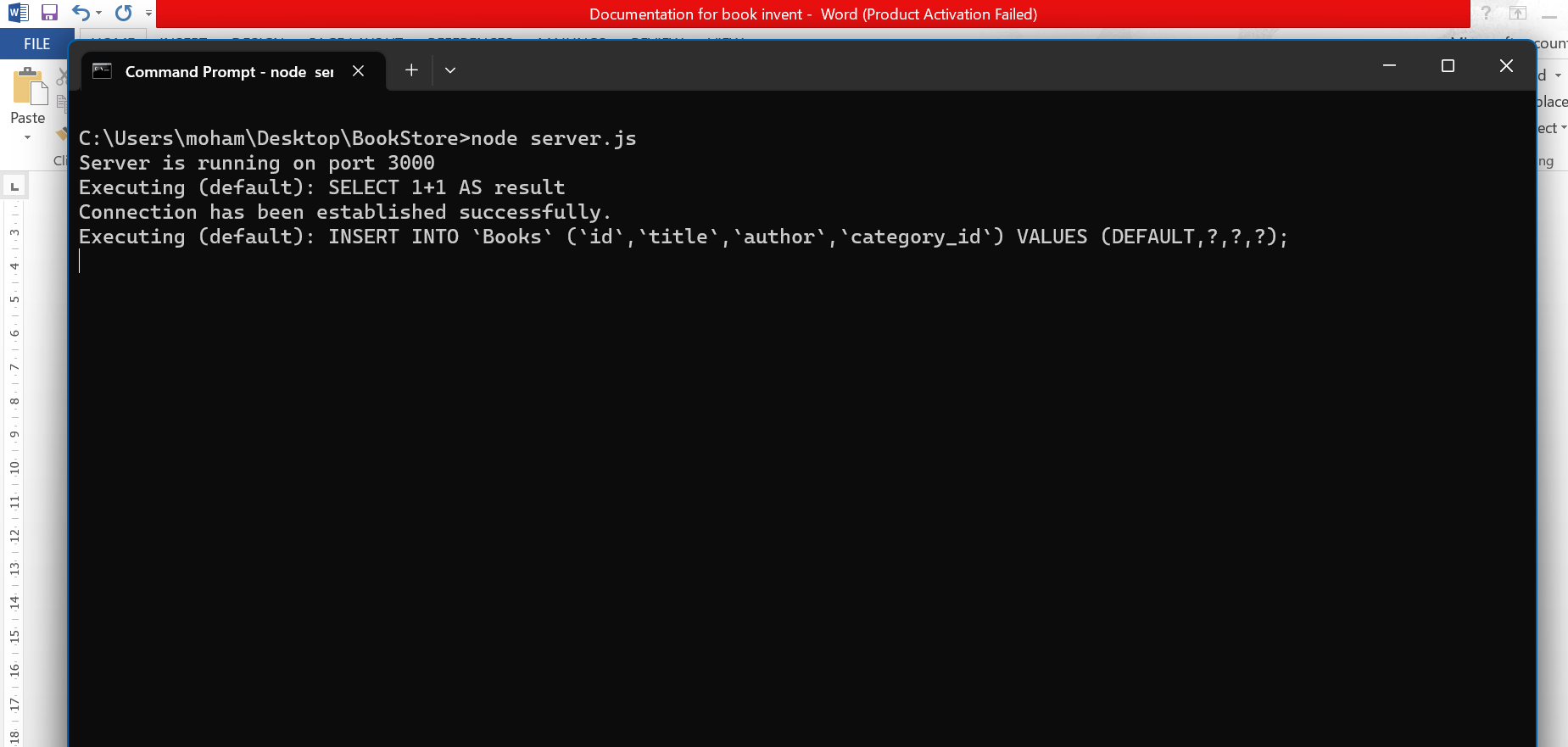
Click body> raw> and BLUE TEXT AS JSON

Press Send with some data as I did.

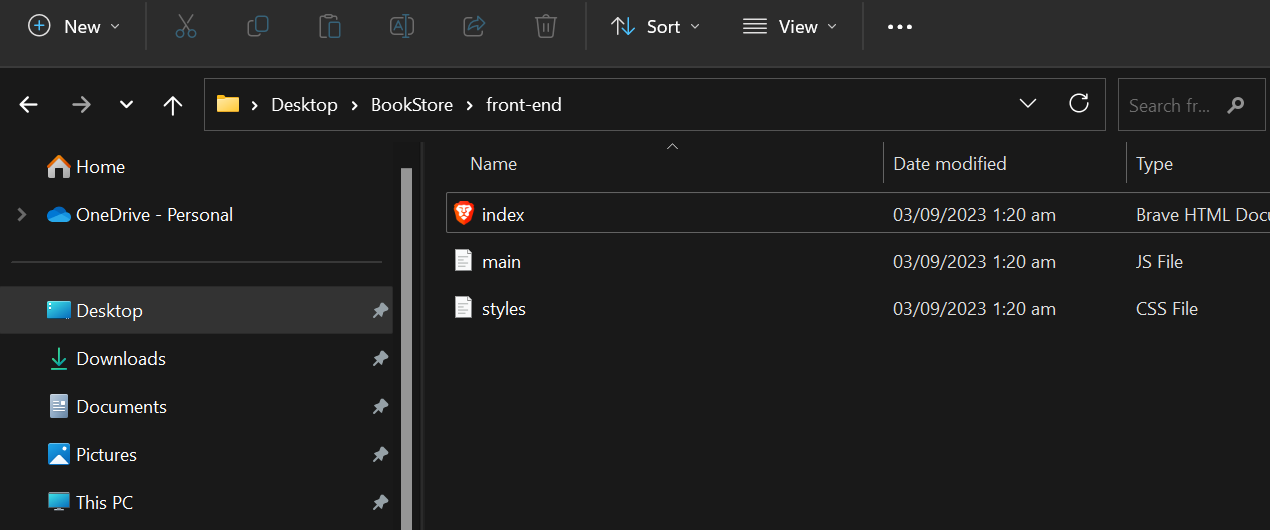
(Make sure node server.js is running)



Output would be like:



Create a folder :



Index.html:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Book Inventory</title>

  <link rel="stylesheet" href="styles.css">

</head>

<body>

  <h1>Book Inventory</h1>

  <div id="book-list">

*<!-- Books will be displayed here -->*

  </div>

  <script src="main.js"></script>

</body>

</html>

Main.js:

*// Fetch all books and display them on the page*

*async* function fetchAllBooks() {

  try {

*const* response = *await* fetch('http://localhost:3000/api/books');

*const* books = *await* response.json();

*const* bookList = document.getElementById('book-list');

    bookList.innerHTML = '';

    books.forEach(book => {

*const* bookItem = document.createElement('div');

      bookItem.textContent = `${book.title} by ${book.author}`;

      bookList.appendChild(bookItem);

    });

  } catch (error) {

    console.error('There was a problem fetching the data:', error);

  }

}

*// Call the function to fetch all books when the page loads*

window.addEventListener('load', fetchAllBooks);

styles.css:

body {

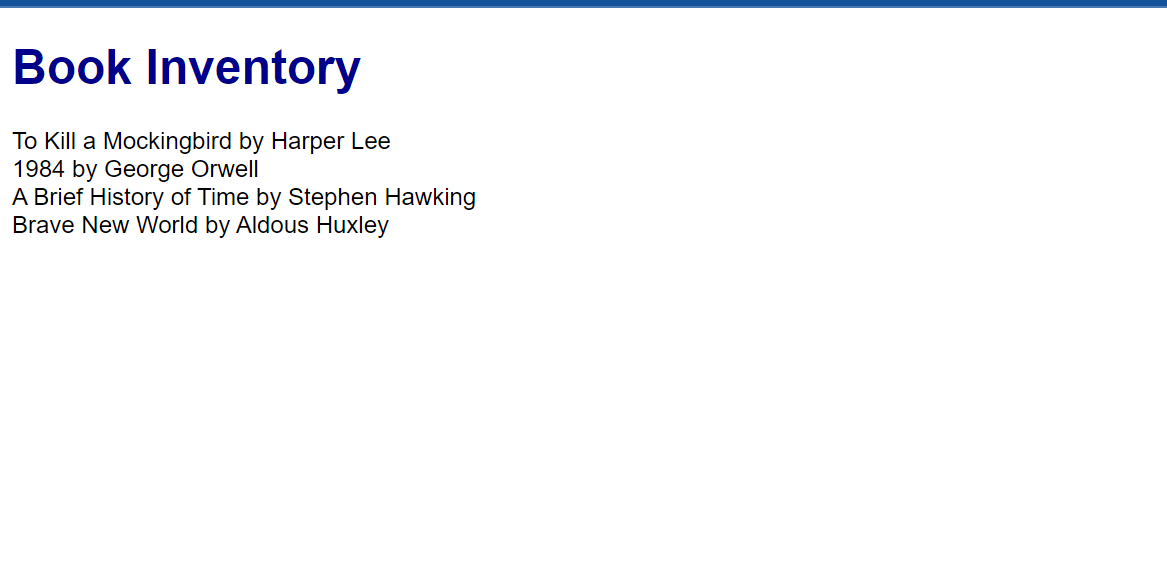
  font-family: Arial, sans-serif;

}

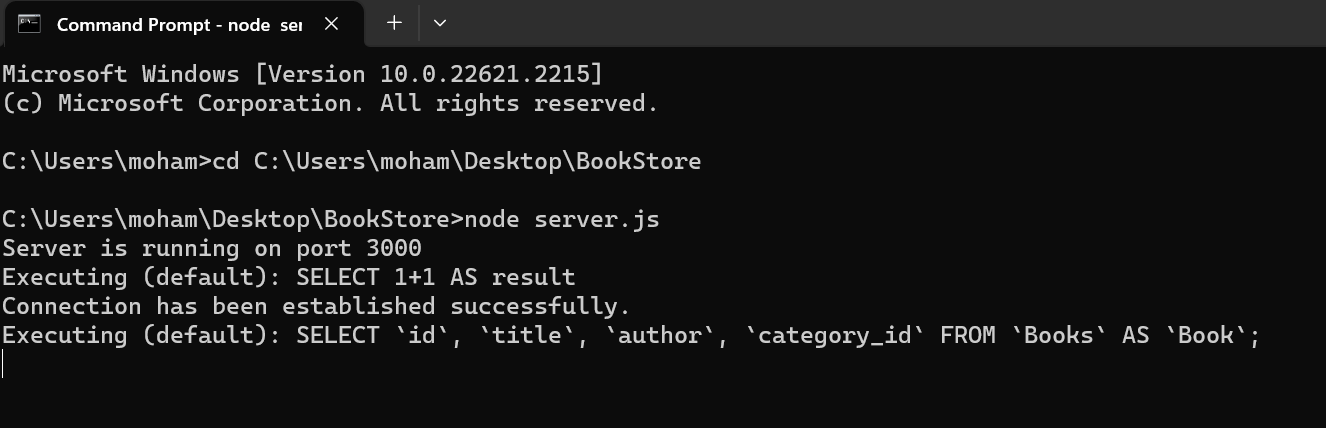
h1 {

  color: darkblue;

}

Open: index.html and you will see: 

Always make sure node js is running before opening index,html:



LAST THING WE WILL MAKE THE FRONT END LOOK MORE APPEALING:

UPDATE:

Index.html:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Book Inventory</title>

    <link rel="stylesheet" href="styles.css">

</head>

<body>

    <h1>Book Inventory</h1>

    <div id="book-list">

*<!-- Books will be inserted here -->*

    </div>

    <h2>Add a New Book</h2>

    <form id="add-book-form">

        <label for="title">Title:</label>

        <input type="text" id="title" required>

        <label for="author">Author:</label>

        <input type="text" id="author" required>

        <label for="category">Category:</label>

        <input type="number" id="category" required>

        <button type="submit">Add Book</button>

    </form>

    <script src="main.js"></script>

</body>

</html>

Style.css:

body {

  font-family: Arial, sans-serif;

  margin: 20px;

}

h1, h2 {

  color: #333;

}

#book-list {

  display: grid;

  grid-template-columns: repeat(3, 1fr);

  gap: 20px;

}

.book-card {

  border: 1px solid #ccc;

  padding: 15px;

  border-radius: 8px;

  text-align: center;

}

#add-book-form {

  margin-top: 30px;

}

label, input {

  margin-bottom: 10px;

}

Main.js:

*// Fetch all books and display them*

*async* function fetchAllBooks() {

  try {

*const* response = *await* fetch('http://localhost:3000/api/books');

*const* books = *await* response.json();

*const* bookList = document.getElementById('book-list');

      bookList.innerHTML = '';

      books.forEach(book => {

          bookList.innerHTML += `

              <div class="book-card">

                  <h3>${book.title}</h3>

                  <p>Author: ${book.author}</p>

                  <p>Category ID: ${book.category\_id}</p>

              </div>

          `;

      });

  } catch (error) {

      console.error('There was a problem fetching the data:', error);

  }

}

*// Add a new book*

document.getElementById('add-book-form').addEventListener('submit', *async* function(event) {

  event.preventDefault();

*const* title = document.getElementById('title').value;

*const* author = document.getElementById('author').value;

*const* category = document.getElementById('category').value;

  try {

*const* response = *await* fetch('http://localhost:3000/api/books', {

          method: 'POST',

          headers: {

              'Content-Type': 'application/json',

          },

          body: JSON.stringify({

              title,

              author,

              category\_id: category,

          }),

      });

      if (response.ok) {

          alert('Book added successfully.');

          fetchAllBooks();  *// refresh the book list*

      } else {

          alert('Failed to add book.');

      }

  } catch (error) {

      console.error('There was a problem adding the book:', error);

  }

});

*// Initial fetch to populate the book list*

fetchAllBooks();

FINAL LOOK:

