**Documentation**

The Book Inventory Management System is a web application designed to help users manage a collection of books. It allows users to add new books, filter existing books based on various criteria, and export the inventory data in both CSV and JSON formats.

**Technologies Used:**

Nodejs, Express, MySql, json2csv, HTML, CSS, Javascript

**Installation:**

1. **Clone the repository:**

git clone <repository-url>

cd <repository-directory>

1. **Install dependencies:**

npm install

1. **Set up the database:**

Create a database in MySQL.

Use the provided SQL script to create the Inventory table.

**Script:**

CREATE DATABASE BookInventory; USE BookInventory; CREATE TABLE Inventory ( entry\_id INT AUTO\_INCREMENT PRIMARY KEY, title VARCHAR(255) NOT NULL, author VARCHAR(255) NOT NULL, genre VARCHAR(100), publication\_date DATE, isbn VARCHAR(20) UNIQUE NOT NULL );

This SQL script does the following:

* Creates a database called BookInventory.
* Defines an Inventory table with columns for book details, including entry\_id as the primary key and isbn as a unique field.

1. **Start the server:**

node index.js

1. **Access the application:**

Open your web browser and navigate to <http://localhost:3000/index.html>

**Features in the application:**

* **Adding Books:** Fill out the "Add New Book" form with the required information (title, author, ISBN) and click "Add Book" to submit.
* **Filtering Books:** Use the "Filter Books" form to search for specific books based on title, author, genre, or publication date. Click "Search" to view the results.
* **Exporting Data:** Click the "Export as CSV" or "Export as JSON" button to download the entire book inventory in the selected format.

**API Endpoints:**

1. **Add New Book**

**Endpoint:** POST /books

**Body:**

{

"title": "Book Title",

"author": "Author Name",

"genre": "Genre",

"publication\_date": "YYYY-MM-DD",

"isbn": "ISBN Number"

}

1. **Filter Books**

**Endpoint:** GET /books

**Query Parameter:**

* title: Filter by book title.
* author: Filter by author name.
* genre: Filter by genre.
* publication\_date: Filter by publication date.

1. **Export Books**

**Endpoint:** GET /books/export

**Query Parameter:**

Format – either csv or json

**Database Schema:**

**Inventory Table:**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Datatype** | **Constraints** |
| Entry Id | INT | Primary key, Auto\_Increment |
| Title | VARCHAR(255) | NOT NULL |
| Author | VARCHAR(255) | NOT NULL |
| Genre | VARCHAR(100) | NULL |
| Publication Date | DATE | NULL |
| ISBN | VARCHAR(20) | NOT NULL UNIQUE |

**Design Decisions:**

* The application uses a RESTful API approach for clean separation between the front-end and back-end logic.
* The database schema is designed to be simple yet flexible, allowing for easy expansion in the future if more attributes for books are required.

**Challenges Faced:**

* **Input Validation:** Ensuring the user inputs are valid and appropriately handled for both the server and client sides.
* **Database Connection Management:** Properly managing the connection to the MySQL database and handling errors that may arise.

**Conclusion:**

The Book Inventory Management System provides a user-friendly interface for managing book data efficiently. This project serves as a foundational system that can be further expanded with additional features such as user authentication, advanced filtering options, or a more sophisticated front-end framework.