**Book Reservation API Documentation**

1. **Overview**

The Book Reservation API manages a collection of books, providing CRUD operations for books and handling reservations. This API is built using C# with .NET 7.0, utilizes an SQLite database through Entity Framework Core, and is containerized with Docker. Swagger is integrated for easy exploration and testing.

1. **Technologies Used**

* **Programming Language:** C#
* **Framework:** .NET 7.0
* **Database:** SQLite
* **ORM:** Entity Framework Core
* **Containerization:** Docker
* **API Documentation:** Swagger

1. **Project Structure**

* **Dockerfile:** Configuration for building the Docker image.
* **SystemDbContext.cs:** Entity Framework DbContext for interacting with the SQLite database.
* **BooksController.cs:** Controller defining API endpoints for book operations.
* **ReservationController.cs:** Controller defining API endpoints for reservation operations.
* **Book.cs:** Entity class representing a book.
* **Reservation.cs:** Entity class representing a reservation.
* **ReservedBook.cs:** Model for displaying reserved book information on the UI.
* **ReservationHistory.cs:** Model for displaying reservation history for the books.

1. **Running the Application**

To run the application using Docker:

docker-compose build

docker-compose up

The API is accessible at [http://localhost:32768](http://localhost:32768/), and Swagger documentation is available at Swagger UI.

1. **Endpoints**
   1. **Book**

* **GET /v1/Book:** Retrieve all books.
* **POST /v1/Book:** Create a new book.
* **GET /v1/Book/{id}:** Retrieve details of a specific book by ID.
* **PUT /v1/Book/{id}:** Update a book by ID.
* **DELETE /v1/Book/{id}:** Delete a book by ID.
* **GET /v1/Book/search:** Search book by Title or Author.
  1. **Reservation**
* **GET /v1/Reservation/reserved\_book:** Get a list of reserved books with reservation comment.
* **GET /v1/Reservation/available\_book:** Get a list of available (not reserved) books.
* **POST /v1/Reservation/reserve/{bookId}:** Reserve a book by the book ID.
* **POST /v1/Reservation/return/{bookId}:** Return a book by the book ID.
* **GET /v1/Reservation/History:** Get all reservation history.

1. **Database**

The application uses an SQLite database, and the database file is named books.db.

1. **Additional Features**

* **Swagger UI:** Access Swagger documentation at Swagger UI.
* **Docker Support:** Containerized using Docker for easy deployment and portability.

1. **Notes**

* No authorization is required for API access.
* Entity Framework Core is used for database interaction.