Project Name: BookStore API

Project Description:

The BookStore API provides a RESTful interface for managing an online bookstore's inventory, customer orders, and author information. The API will allow users to view available books, manage inventory, handle customer orders, and retrieve author details.

Requirements

1. API Overview

- **Purpose**: To provide a RESTful API for managing books, authors, and orders.
- **Scope**: Enable users to access book information, place orders, and manage author details.
- **Audience**: Primarily used by front-end applications, admin dashboards, and potentially third-party developers integrating bookstore functionality.

2. Core Functionality

Entities:

- **Books**: CRUD operations for book management.
- **Authors**: Manage author profiles and link books to authors.
- **Orders**: Enable customers to place and track orders.

Endpoints:

- **Books**:
  - `GET /books`: Fetch a list of all books.
  - `GET /books/{id}`: Fetch details of a specific book.
  - `POST /books`: Add a new book (admin access only).
  - `PUT /books/{id}`: Update details of a specific book (admin access only).
  - `DELETE /books/{id}`: Delete a specific book (admin access only).
- **Authors**:
  - `GET /authors`: Fetch a list of all authors.
  - `GET /authors/{id}`: Fetch details of a specific author.
  - `POST /authors`: Add a new author (admin access only).
  - `PUT /authors/{id}`: Update author information (admin access only).
  - `DELETE /authors/{id}`: Delete an author (admin access only).
- **Orders**:
  - `POST /orders`: Create a new order.
  - `GET /orders/{id}`: Get details of a specific order.
  - `PUT /orders/{id}`: Update the status of an order (admin access only).

3. Data Format

- **Request and Response Format**: JSON.
- **Example of Book Object**:

```
{
  "id": 1,
  "title": "The Great Gatsby",
  "authorId": 5,
  "price": 10.99,
  "stock": 120,
  "publishedDate": "1925-04-10"
}
```

4. Authentication & Authorization

- **Authentication**: JWT (JSON Web Tokens) for user authentication.
- **Authorization**: Role-based access control (RBAC).
  - **Admin Role**: Full access to all endpoints.
  - **Customer Role**: Read access to books and authors, can create orders.

5. Error Handling

- **Standard HTTP Codes**:
  - `200 OK` for successful requests.
  - `400 Bad Request` for invalid data input.
  - `401 Unauthorized` for requests without valid authentication.
  - `404 Not Found` for non-existent resources.
  - `500 Internal Server Error` for unexpected server issues.
- **Error Response Example**:

```
{
  "error": "Invalid request",
  "message": "The 'price' field must be a positive number."
}
```

6. Data Storage

- **Database**: SQLServer for structured data storage.
- **Tables**:
  - `Books`: Stores book information.
  - `Authors`: Stores author profiles.
  - `Orders`: Manages customer orders.
- **Data Validation**: Ensures required fields (e.g., title, authorId, price) are present and valid.

7. Documentation

- **API Documentation**: Use Swagger for interactive API documentation.
- **Endpoint Details**: Include endpoint descriptions, request parameters, example requests and responses.