

SwaggerUI API documentation for a bookstore web API

Table of Contents

Description	2
Tools and technologies used	2
Code snippets	2
Program.cs	2
bookstore.csproj	2
BooksController.cs	3
Output: Swagger UI screenshots	3
GET	4
POST	4
GET	4
GET with execution	5
PUT	5
DELETE	5
DELETE with execution	6

Description

The Bookstore API is a .NET Core application featuring RESTful endpoints, GET, POST, PUT, DELETE, for managing bookstore inventory. This project demonstrates my ability to integrate Swagger for automatic API documentation using Swashbuckle.

Tools and technologies used

- Microsoft Visual Studio 2022
- .NET 8.0
- ASP.NET Core MVC
- Swashbuckle.AspNetCore 6.6.2

Code snippets

Program.cs

```
builder.Services.AddControllers();
// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen(s =>
{
    s.SwaggerDoc("v1", new Microsoft.OpenApi.Models.OpenApiInfo
    {
        Title = "API documentation for Bookstore",
        Version = "v1",
        Description = "API documentation for Bookstore",
        Contact = new Microsoft.OpenApi.Models.OpenApiContact()
        {
            Name = "Priya Dalvi",
            Email = "dalvipriya09@gmail.com"
        },
    });
    var xmlPath = Path.Combine(System.AppContext.BaseDirectory, "bookstore.xml");
    s.IncludeXmlComments(xmlPath);
});

var app = builder.Build();

// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}
```

bookstore.csproj

```
<Project Sdk="Microsoft.NET.Sdk.Web">

  <PropertyGroup>
    <TargetFramework>net8.0</TargetFramework>
    <GenerateDocumentationFile>true</GenerateDocumentationFile>
    <Nullable>enable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
  </PropertyGroup>
  <ItemGroup>
    <PackageReference Include="Swashbuckle.AspNetCore" Version="6.6.2" />
  </ItemGroup>
</Project>
```

BooksController.cs

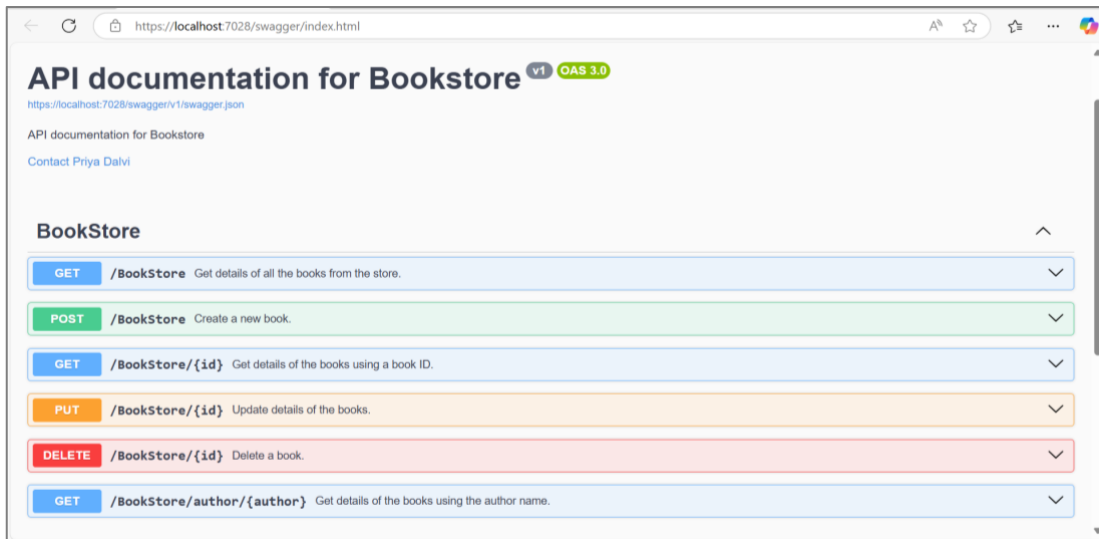
```
namespace Bookstore.Controllers
{
    [ApiController]
    [Route("[controller]")]
    public class BookStoreController : Controller
    {
        // In-memory list to simulate a database
        private static List<Book> Books = new List<Book>
        {
            new Book { Id = 1, Title = "The Catcher in the Rye", Author = "J.D. Salinger", Price = 10.99m },
            new Book { Id = 2, Title = "To Kill a Mockingbird", Author = "Harper Lee", Price = 12.99m },
            new Book { Id = 3, Title = "1984", Author = "George Orwell", Price = 14.99m },
            new Book { Id = 4, Title = "The Great Gatsby", Author = "F. Scott Fitzgerald", Price = 9.99m },
            new Book { Id = 5, Title = "Moby Dick", Author = "Herman Melville", Price = 15.99m },
            new Book { Id = 6, Title = "Pride and Prejudice", Author = "Jane Austen", Price = 8.99m },
            new Book { Id = 7, Title = "War and Peace", Author = "Leo Tolstoy", Price = 20.99m },
            new Book { Id = 8, Title = "The Hobbit", Author = "J.R.R. Tolkien", Price = 13.99m },
            new Book { Id = 9, Title = "Crime and Punishment", Author = "Fyodor Dostoevsky", Price = 16.99m },
            new Book { Id = 10, Title = "The Alchemist", Author = "Paulo Coelho", Price = 11.99m }
        };

        // GET: api/BookStore
        /// <summary>
        /// Get details of all the books from the store.
        /// </summary>
        /// <remarks>This API should be used to fetch all books from the store.</remarks>
        /// <returns></returns>
        /// <response code="200">Successfull excecution</response>
        /// <response code="500">Internal server error found</response>

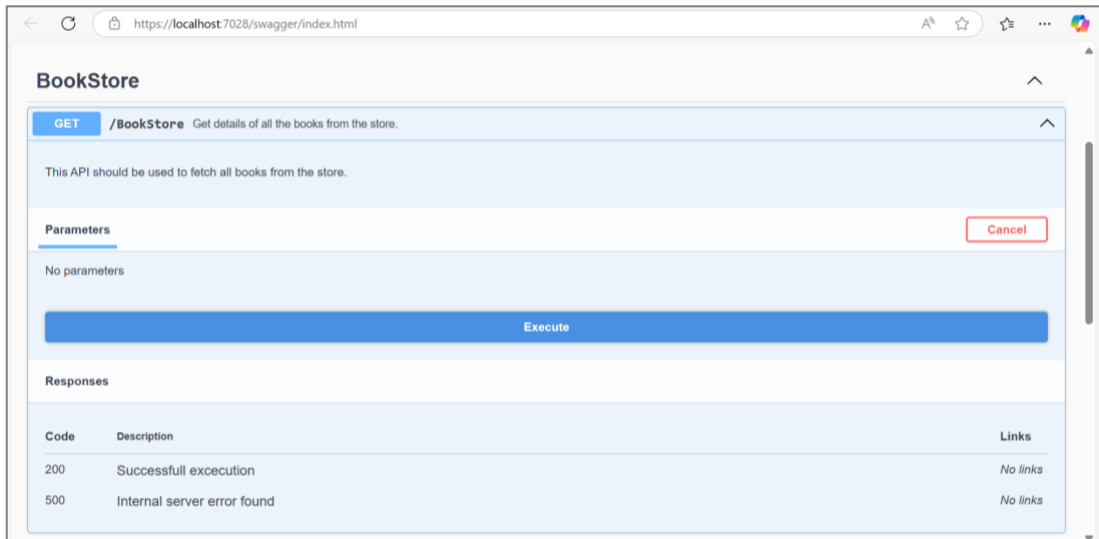
        [HttpGet]
        public IActionResult GetAllBooks()
        {
            return Ok(Books);
        }
    }
}
```

Output: Swagger UI screenshots

The following screenshots display the API endpoints, parameters, code responses, and execution results.



GET



The image shows a Swagger UI interface for a GET endpoint. The browser address bar shows `https://localhost:7028/swagger/index.html`. The endpoint is `GET /BookStore` with the description "Get details of all the books from the store." Below the description, it says "This API should be used to fetch all books from the store." The "Parameters" section is empty, with a "Cancel" button. A large blue "Execute" button is present. The "Responses" section contains a table with two entries:

Code	Description	Links
200	Successfull excecution	No links
500	Internal server error found	No links

POST



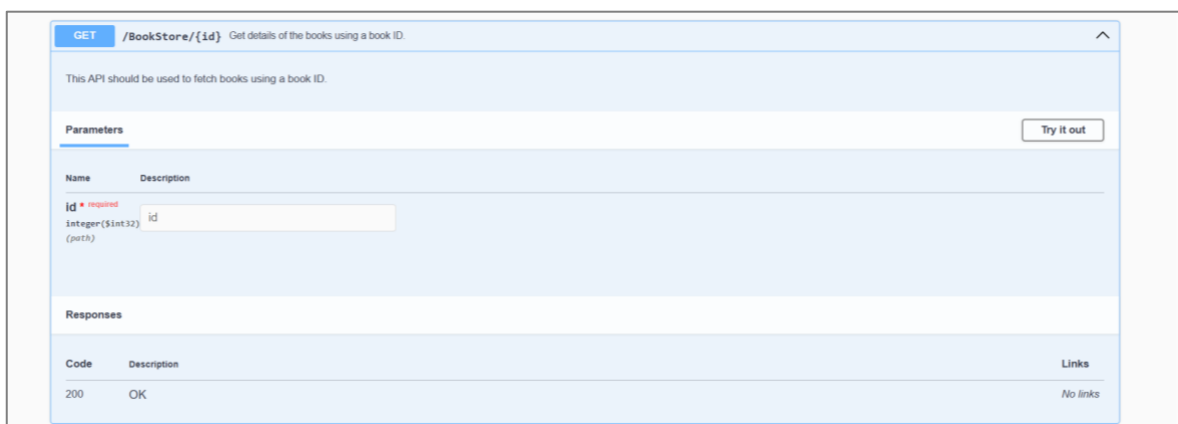
The image shows a Swagger UI interface for a POST endpoint. The endpoint is `POST /BookStore` with the description "Create a new book." Below the description, it says "This API should be used to add details of a new book in the system." The "Parameters" section is empty, with a "Try it out" button. The "Request body" section is set to `application/json`. An "Example Value" is shown in a dark box:

```
{
  "id": 0,
  "title": "string",
  "author": "string",
  "price": 0
}
```

The "Responses" section contains a table with one entry:

Code	Description	Links
200	OK	No links

GET



The image shows a Swagger UI interface for a GET endpoint. The endpoint is `GET /BookStore/{id}` with the description "Get details of the books using a book ID." Below the description, it says "This API should be used to fetch books using a book ID." The "Parameters" section has a "Try it out" button and a table with one parameter:

Name	Description
<code>id</code> * required integer(\$int32) (path)	id

The "Responses" section contains a table with one entry:

Code	Description	Links
200	OK	No links

GET with execution

The screenshot shows the Swagger UI for the endpoint `GET /BookStore/{id}`. The interface is light blue. The title is "GET /BookStore/{id} Get details of the books using a book ID." Below the title, it says "This API should be used to fetch books using a book ID." The "Parameters" section has a table with one entry: `id` (required), `Integer($int32)`, `id`, `(path)`. The value `5` is entered in the input field. Below the parameters are "Execute" and "Clear" buttons. The "Responses" section shows a "200 OK" response with a "Response body" containing a JSON object: `{ "id": 5, "title": "The Hobbit", "author": "J.R.R. Tolkien", "price": 12.99 }`. The "Request URL" is `https://localhost:7028/swagger/index.html` and the "Server response" is `200 OK`.

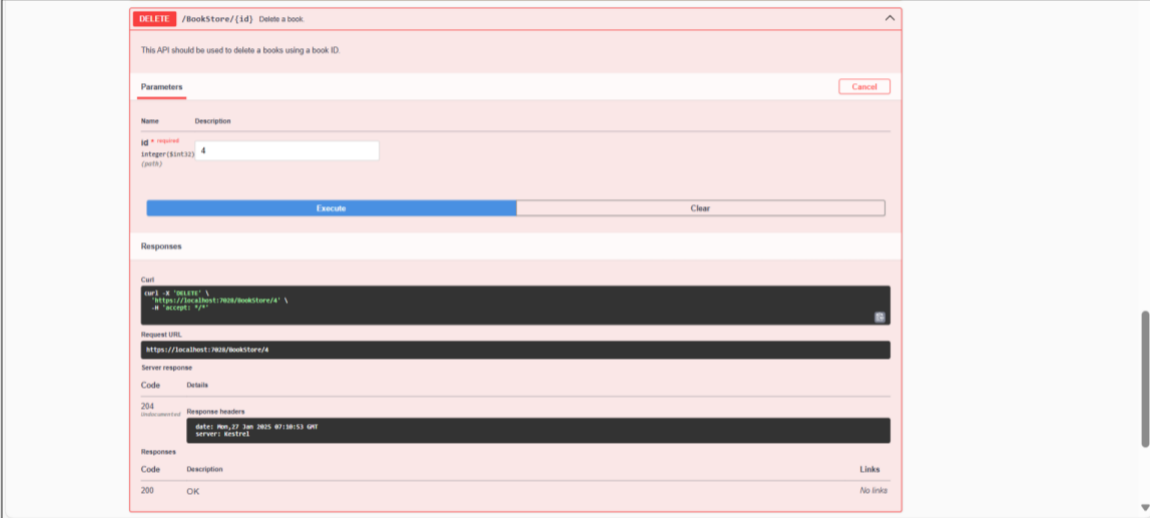
PUT

The screenshot shows the Swagger UI for the endpoint `PUT /BookStore/{id}`. The interface is light orange. The title is "PUT /BookStore/{id} Update details of the books." Below the title, it says "This API should be used to update a book that you can first search using a book ID." The "Parameters" section has a table with one entry: `id` (required), `Integer($int32)`, `id`, `(path)`. The "Request body" section has a dropdown menu set to `application/json`. Below it, the "Example Value" is a JSON object: `{ "id": 5, "title": "The Hobbit", "author": "J.R.R. Tolkien", "price": 12.99 }`. The "Responses" section shows a "200 OK" response with a "Response body" containing a JSON object: `{ "id": 5, "title": "The Hobbit", "author": "J.R.R. Tolkien", "price": 12.99 }`. The "Request URL" is `https://localhost:7028/swagger/index.html` and the "Server response" is `200 OK`.

DELETE

The screenshot shows the Swagger UI for the endpoint `DELETE /BookStore/{id}`. The interface is light red. The title is "DELETE /BookStore/{id} Delete a book." Below the title, it says "This API should be used to delete a books using a book ID." The "Parameters" section has a table with one entry: `id` (required), `Integer($int32)`, `id`, `(path)`. The "Responses" section shows a "200 OK" response with a "Response body" containing a JSON object: `{ "id": 5, "title": "The Hobbit", "author": "J.R.R. Tolkien", "price": 12.99 }`. The "Request URL" is `https://localhost:7028/swagger/index.html` and the "Server response" is `200 OK`.

DELETE with execution



The screenshot displays a REST client interface for a DELETE endpoint. The title bar reads "DELETE /BookStore/[id] Delete a book." Below the title, a note states: "This API should be used to delete a books using a book ID." The "Parameters" section contains a table with one entry: "id" (required, path) with a value of "4". Below the parameters are "Execute" and "Clear" buttons. The "Responses" section shows the following details:

- curl:**

```
curl -X DELETE -i \
  "https://localhost:7030/bookstore/" \
  -H "Accept: */*"
```
- Request URL:** https://localhost:7030/bookstore/4
- Server response:**
- Code:** 204
- Details:** Response Headers: date: Mon, 22 Jan 2025 07:08:53 GMT; server: Kestrel
- Response:**
- Code:** 200
- Description:** OK
- Links:** No links