Scenario :

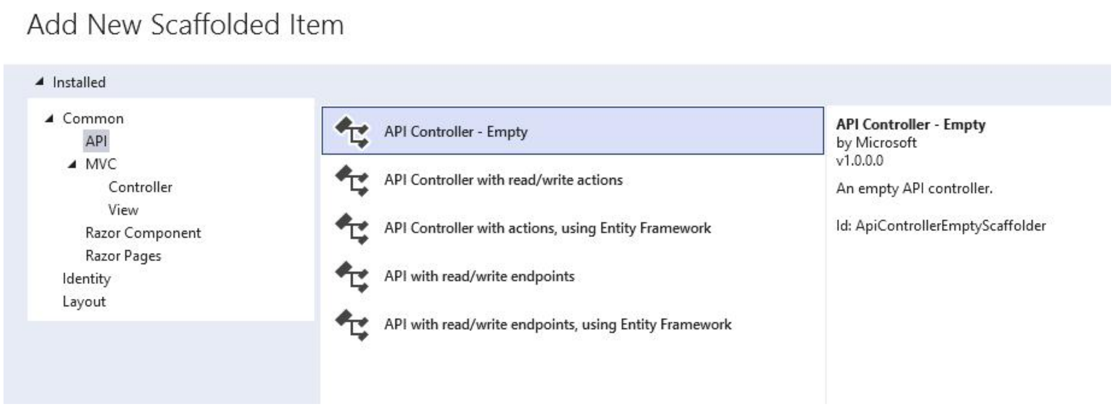
Sarah is an avid reader and has decided to open a small independent bookshop in her neighborhood. She wants to create a Web API application to manage and showcase the available books to customers online.

Help her by creating a web API application and making her work more efficient.

Exercise Steps :

1. Create a new web API core project in visual studio and choose the .net 6.0 framework.

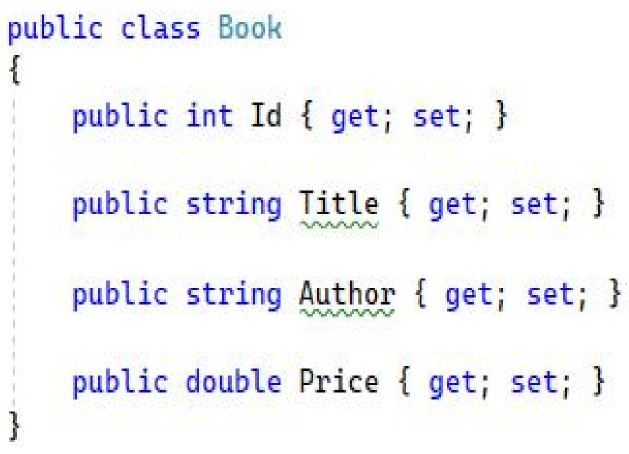
2. Add a new controller in the Controller folder called "**BookController**".


Add New Scaffolded Item
Installed
Common
API
MVC
Controller
View
Razor Component
Razor Pages
Identity
Layout
API Controller - Empty
API Controller with read/write actions
API Controller with actions, using Entity Framework
API with read/write endpoints
API with read/write endpoints, using Entity Framework

API Controller - Empty
by Microsoft
v1.0.0.0
An empty API controller.
Id: ApiControllerEmptyScaffolder

Make sure that the controller is an empty API Controller.

3. Create a new folder called **Models**. Inside that folder, create a class named **Book.cs**. Declare the mentioned properties in the Book.cs class.

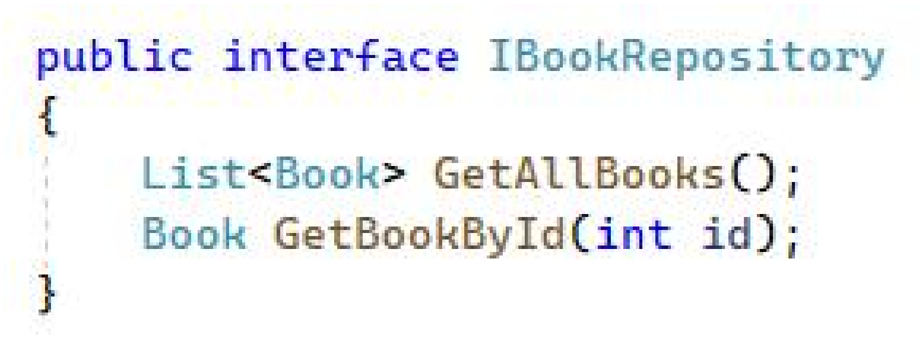


```
public class Book
{
    public int Id { get; set; }

    public string Title { get; set; }

    public string Author { get; set; }

    public double Price { get; set; }
}
```

4. Create a new folder called **Interface**. Inside that folder, create an interface named **IBookRepository.cs**.

Declare the given method in the IBookRepository.cs interface.

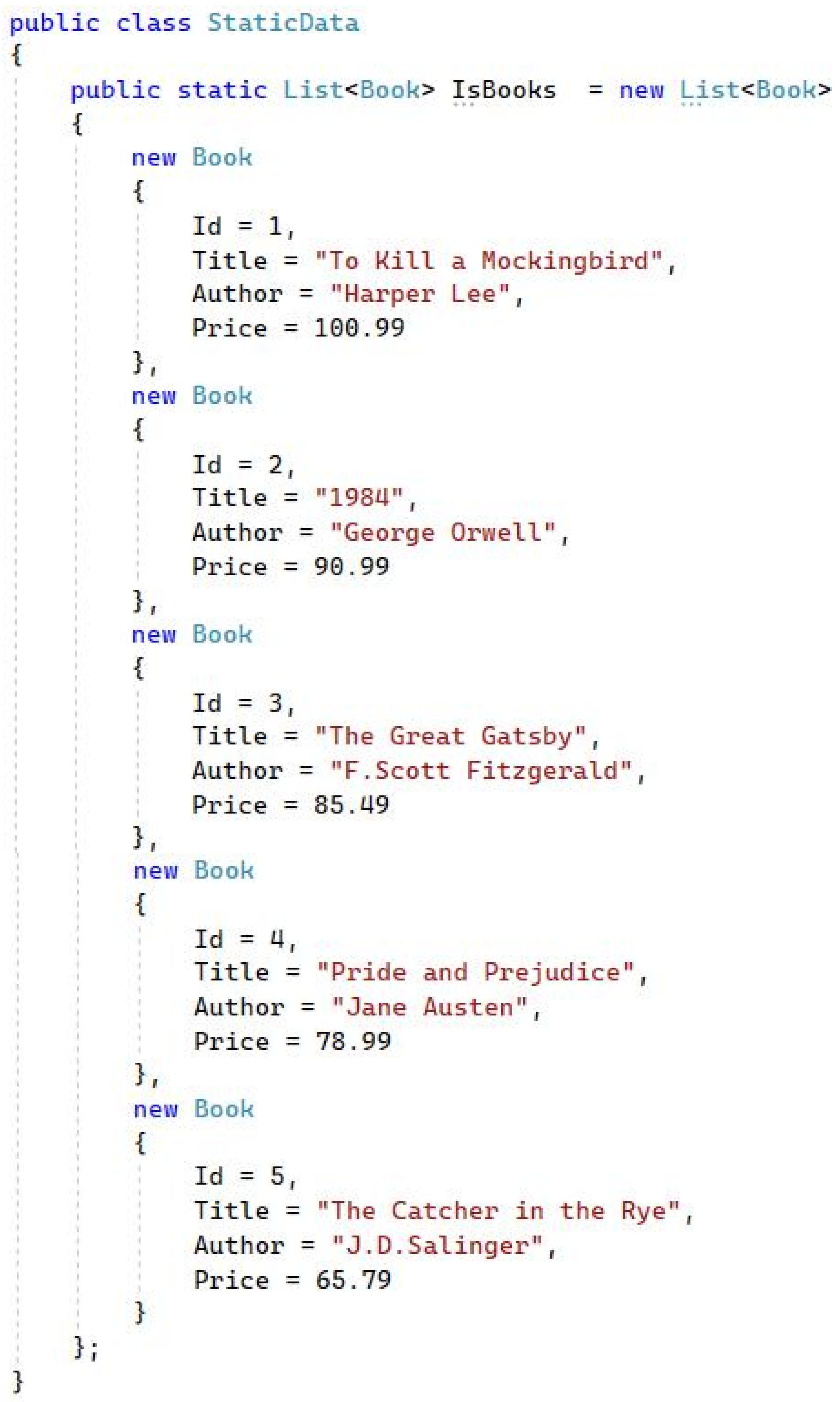
Method
List<Book> GetAllBooks()
Book GetBookById(int id)



```
public interface IBookRepository
{
    List<Book> GetAllBooks();
    Book GetBookById(int id);
}
```

5. Create a new folder called **Repository**. Inside that folder, create two classes, namely **StaticData**.cs and **BookRepository**.cs.

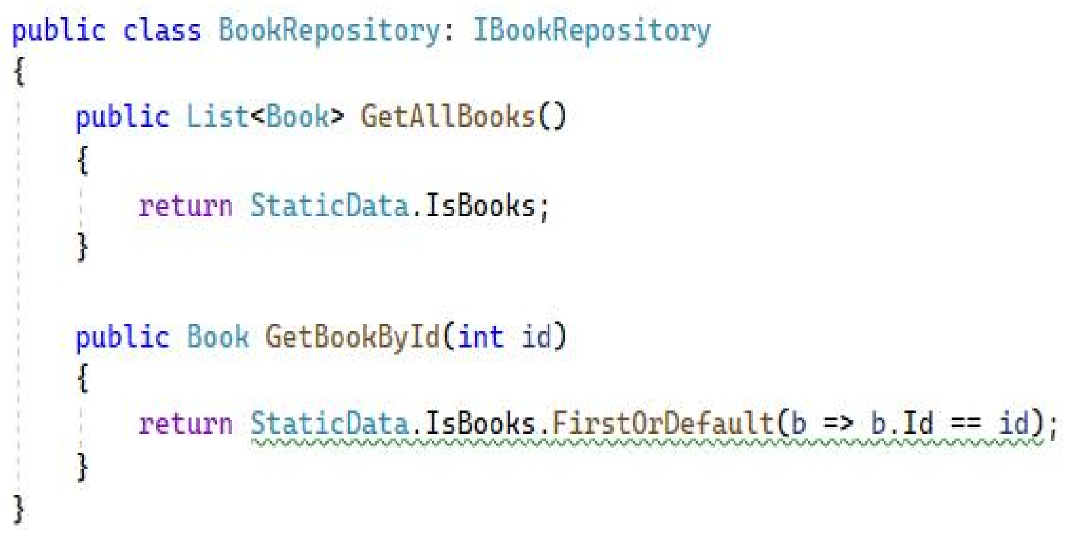
Declare the given list of objects in the **StaticData**.cs class.



```
public class StaticData
{
    public static List<Book> IsBooks  = new List<Book>
    {
        new Book
        {
            Id = 1,
            Title = "To Kill a Mockingbird",
            Author = "Harper Lee",
            Price = 100.99
        },
        new Book
        {
            Id = 2,
            Title = "1984",
            Author = "George Orwell",
            Price = 90.99
        },
        new Book
        {
            Id = 3,
            Title = "The Great Gatsby",
            Author = "F.Scott Fitzgerald",
            Price = 85.49
        },
        new Book
        {
            Id = 4,
            Title = "Pride and Prejudice",
            Author = "Jane Austen",
            Price = 78.99
        },
        new Book
        {
            Id = 5,
            Title = "The Catcher in the Rye",
            Author = "J.D.Salinger",
            Price = 65.79
        }
    };
}
```

6. Implement the **interface** methods in the **BookRepository**.cs. The logic must retrieve the Books from the list of objects **IsBooks**.

Method	**Functionality**
public List<Book> GetAllBooks()	This method is used to get all the books from the **IsBooks** list.
public Book GetBookById(int id)	This method is used to get a single book by **id** from the **IsBooks** list.



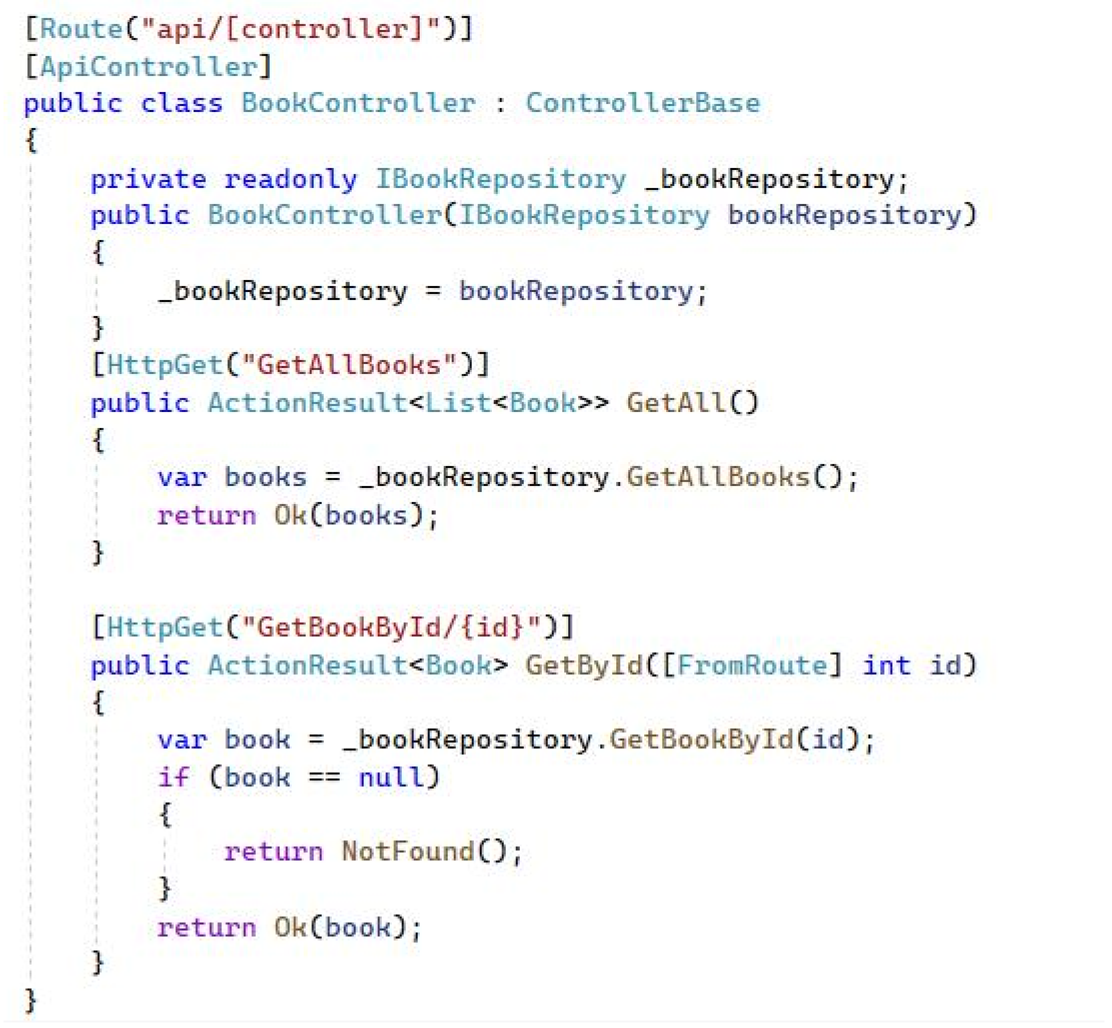
```
public class BookRepository: IBookRepository
{
    public List<Book> GetAllBooks()
    {
        return StaticData.IsBooks;
    }

    public Book GetBookById(int id)
    {
        return StaticData.IsBooks.FirstOrDefault(b => b.Id == id);
    }
}
```

7. Now all the declarations and definitions have been given and done. Let us move to the **BookController**.cs and invoke the method present in the interface **IBookRepository**.cs using the dependency injection. In the Controller, we have 2 action methods namely,

GetAll() method which has no input parameters. This method retrieves all data from the **IsBooks** list of objects. It is an **HttpGet** method with a route of **[Route("api/[controller]/GetAllBooks")]**. Return the **list of books** with the result **Ok** as the status.

GetById() method has one integer input parameter. This method retrieves specific data from the **IsBooks** list of objects based on the **Id** passed. It is an **HttpGet** method with route of **[Route("api/[controller]/GetBookById/{id}")].** Return the **book object** with the result **Ok** as the status. Otherwise, return **NotFound**.

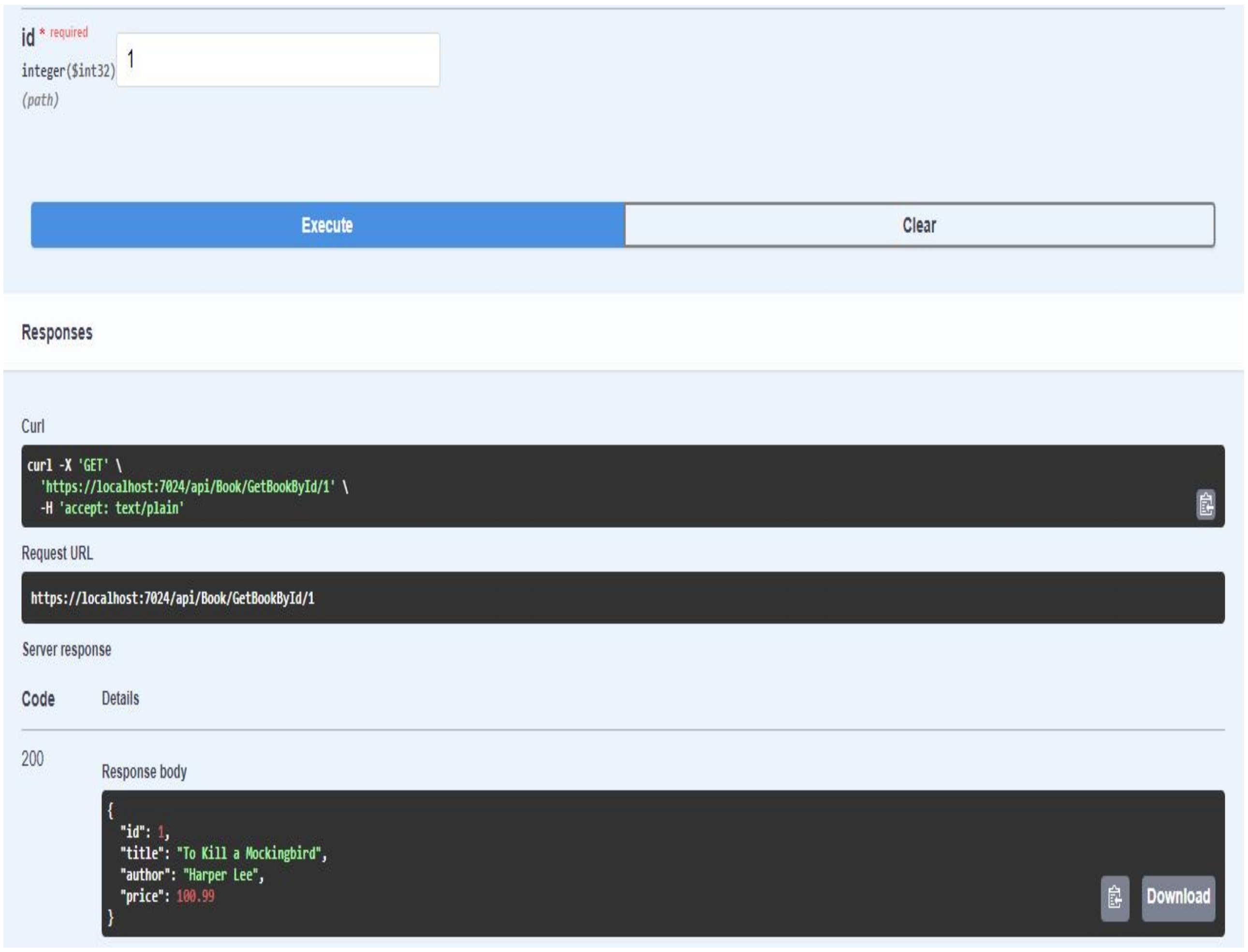
Service	**Http Type & Return Type**	**Functionality**	**Repository (Check BookRepository Section)**
api/[controller]/ GetAllBooks	GET Method ActionResult <List<Book>>	This service is used to get all books from IsBooks and return the result as List<Book>.	Call "GetAllBooks"
api/[controller]/ GetBookById/{id}	GET Method ActionResult<Book>	This service is used to get a single book by id. Parameter type is 'int'. Return the Book details as a Book object.	Call "GetBookById"



```
[Route("api/[controller]")]
[ApiController]
public class BookController : ControllerBase
{
    private readonly IBookRepository _bookRepository;
    public BookController(IBookRepository bookRepository)
    {
        _bookRepository = bookRepository;
    }
    [HttpGet("GetAllBooks")]
    public ActionResult<List<Book>> GetAll()
    {
        var books = _bookRepository.GetAllBooks();
        return Ok(books);
    }

    [HttpGet("GetBookById/{id}")]
    public ActionResult<Book> GetById([FromRoute] int id)
    {
        var book = _bookRepository.GetBookById(id);
        if (book == null)
        {
            return NotFound();
        }
        return Ok(book);
    }
}
```

8. We have completed a **GET** web API with the necessary declarations and definitions. Let us run the project and see the output.

Pass the input in the path, - GetById

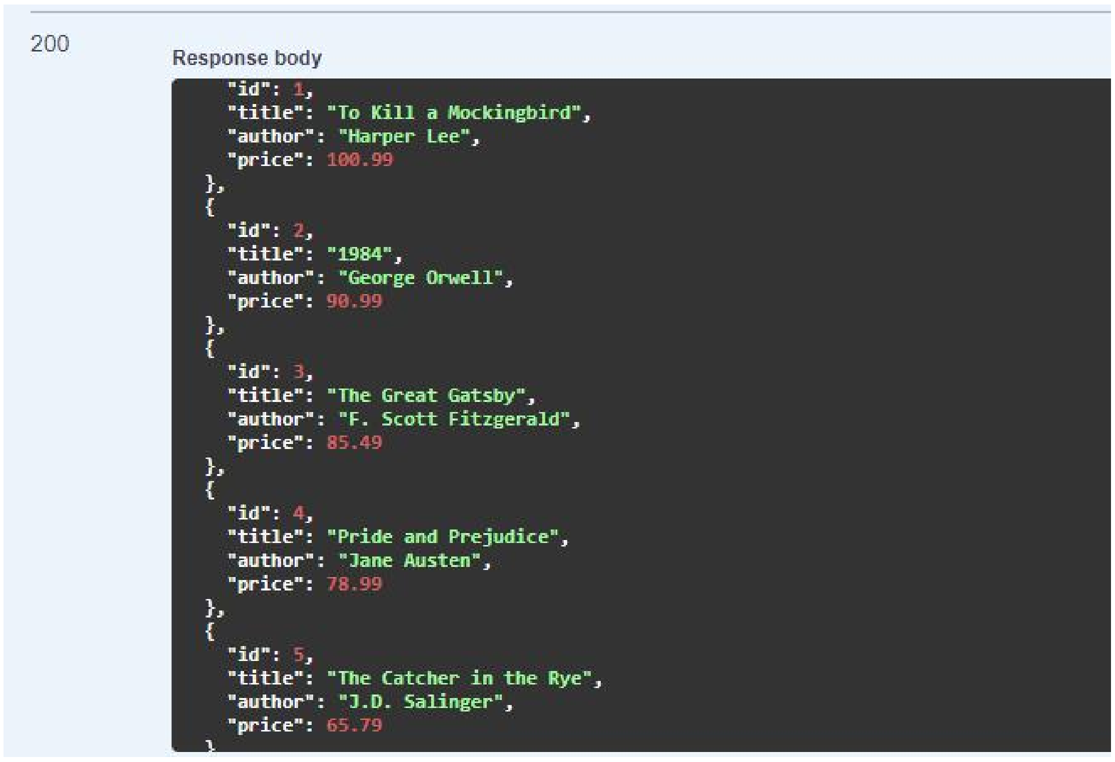


```
curl -X 'GET' \
  'https://localhost:7024/api/Book/GetBookById/1' \
  -H 'accept: text/plain'
```

```
https://localhost:7024/api/Book/GetBookById/1
```

```
{
  "id": 1,
  "title": "To Kill a Mockingbird",
  "author": "Harper Lee",
  "price": 100.99
}
```

GetAll



```
    "id": 1,
    "title": "To Kill a Mockingbird",
    "author": "Harper Lee",
    "price": 100.99
  },
  {
    "id": 2,
    "title": "1984",
    "author": "George Orwell",
    "price": 90.99
  },
  {
    "id": 3,
    "title": "The Great Gatsby",
    "author": "F. Scott Fitzgerald",
    "price": 85.49
  },
  {
    "id": 4,
    "title": "Pride and Prejudice",
    "author": "Jane Austen",
    "price": 78.99
  },
  {
    "id": 5,
    "title": "The Catcher in the Rye",
    "author": "J.D. Salinger",
    "price": 65.79
  }
```

To Summarize,

We have learned about how to retrieve data using the web API GET method, and learned about the dependency injection concept. We also learned about how to navigate a web API and how to pass data in the URI path.