Scenario :

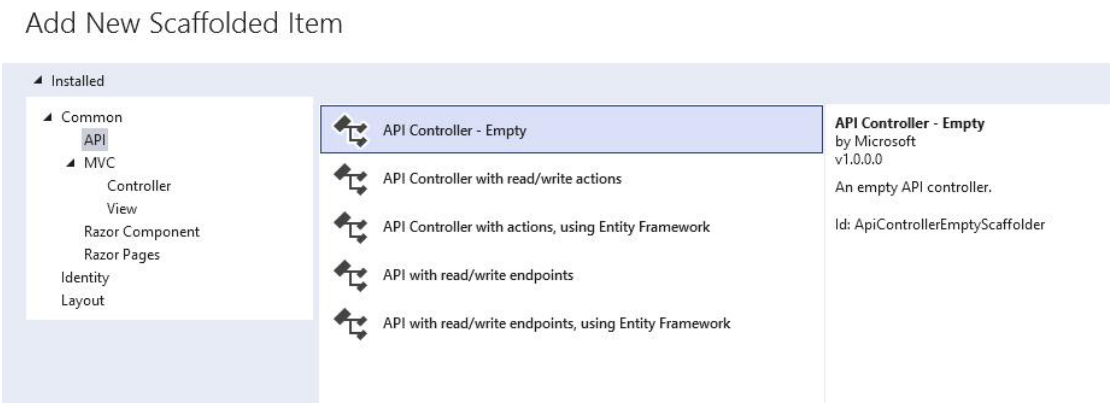
Sarah is an avid reader and has decided to open a small independent bookshop in her neighborhood. She wants to create a Web API application to manage and showcase the available books to customers online.

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**".



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
bool DeleteBook(int id)

```
public interface IBookRepository
{
    bool DeleteBook(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 delete the Book object from the list of objects **IsBooks**.

Method	Functionality
public bool DeleteBook(int id)	This method is used to remove the book by their Id. If the book is removed successfully, it should return true if not, it should return false

```
public class BookRepository: IBookRepository
{
    public bool DeleteBook(int id)
    {
        var bookToRemove = StaticData.IsBooks.FirstOrDefault(b => b.Id == id);
        if (bookToRemove != null)
        {
            StaticData.IsBooks.Remove(bookToRemove);
            return true;
        }
        return false;
    }
}
```

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 are passing the **book id** as a path parameter and this is an **HttpDelete** method with the route of **[Route("api/[controller]/DeleteBook/{id} ")].**

Then we are calling the method **DeleteBook** present in the interface. If the returned result is true, then return **Ok** with the returned **result**. Otherwise, return the status **NotFound**.

Service	Http Type & Return Type	Functionality	Repository (Check BookRepository Section)
api/[controller]/ DeleteBook/{id}	DELETE Method IActionResult	This service is used to delete the book using the Id Parameter	Call "DeleteBook"

		type Id - int. If the result is false it should return the status NotFound OR return OK.	

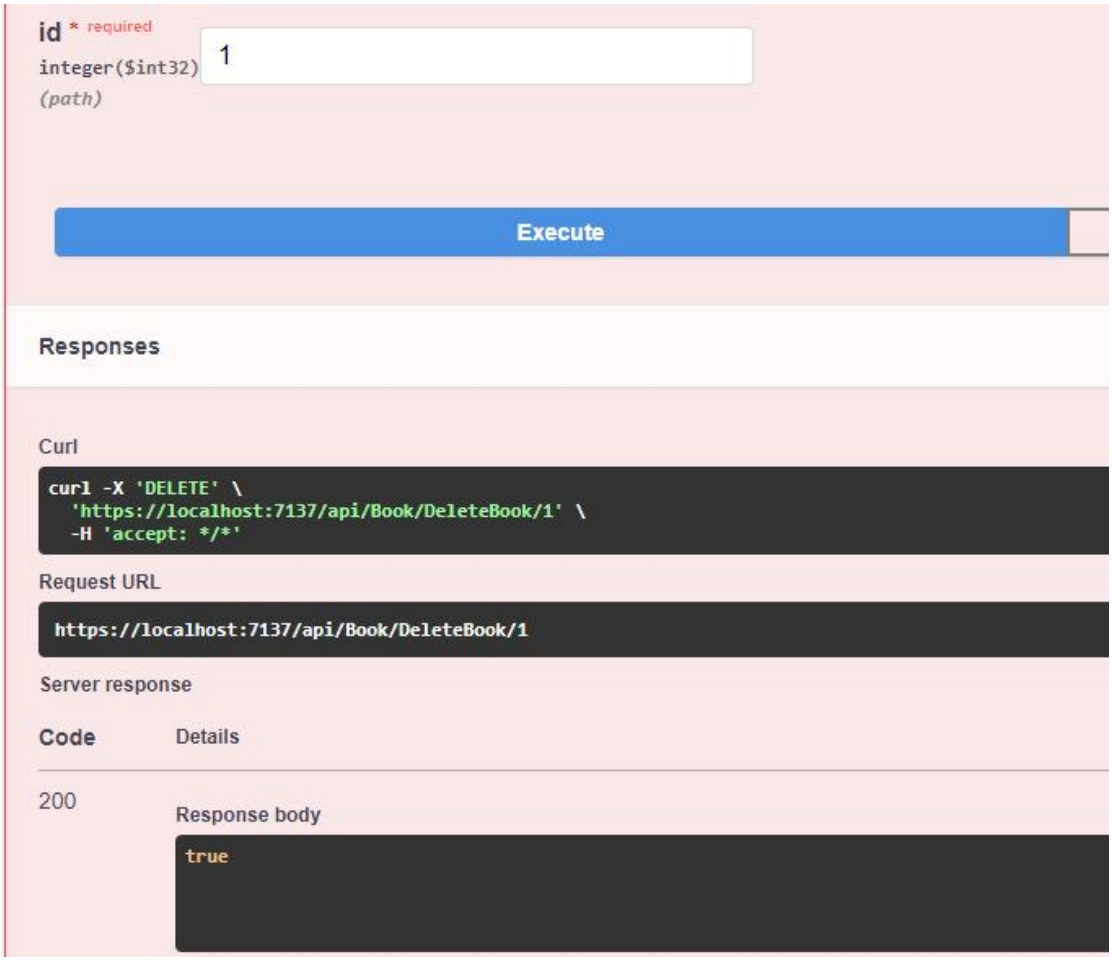
```
[Route("api/[controller]")]
[ApiController]
public class BookController : ControllerBase
{
    private readonly IBookRepository _bookRepository;

    public BookController(IBookRepository bookRepository)
    {
        _bookRepository = bookRepository;
    }

    [HttpDelete("DeleteBook/{id}")]
    public IActionResult Delete([FromRoute] int id)
    {
        var book = _bookRepository.DeleteBook(id);
        if (!book)
        {
            return NotFound();
        }
        return Ok(book);
    }
}
```

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

Pass the input in the path,



We can see the output

So the book details are deleted from the lsBooks list of objects.

To Summarize,

We have learned about how to delete data using the web API DELETE method. 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 and the body.