Project 3 - BookList

The aim of this project is for you to get familiar with the MVC pattern. You don‘t have to use any css in this project unless you really want to. This means that your views (html) do not have to be pretty.

You are to create a application that supports the following routes:

- **/**
- **/Home**
- **/Home/Index**
    - All the above routes should show **a list** of all books
    - This should have a heading(h1) that says: **Book List**
    - The Title, Author and Genre of each book should be shown
    - The Title of each Book should be clickable and should lead to a page which only displays the given book at the /Home/Index/id

- **/Home/Index/id**
    - Should show a page that displays the book with the given id, if there is no book with the given id an Error page should be shown, which should have the text NOT FOUND! In a h1 heading and a link that says HOME that should redirect to /Home/Index when clicked
    - The Title, Author and genre of the given book should be shown here

- **Home/Index?Genre=SomeGenre**
    - The genre should be displayed in a h2 heading at the top of the page
    - a list of books filtered by a certain genre should be displayed
        - The Title and Author of each book should be displayed

- **/Home/Create**
    - Should let the user add a book to the list. There should be a input field for the name of the book, the author and the genre. Then there should be a submit button that says “Add book!” that submits the form to the server once it is clicked. On the server side the new book should get an id. The id should be the highest id of the books incremented by one. I.E if the highest id is 12 the new book should get the id 13. Once the book has been added to the fake database the action should redirect to Home/Index/ where the user should see the new book at the end of the book list

In order to impliment this you will need to persist some data, since we have not yet discussed databases you will be provided with a fake database in this project. This fake database is just a static class that has a static list called Books. To use a static class in your controller you don't need to create an instance of the class. You can store this file at the root of the project (at the same level as the Startup.cs file)

For example: You can access the List of books in a controller like this:

```
FakeDatabase.Books;
```

You should have the following folder structure in your project:

- Controllers
- Models
- Views
  - There should be some subfolders in the Views folder

Do note that one controller(HomeController) is enough for this project.

There will only be one model class called **Book**, you should examine the fakeDatabase.cs file to see what properties the Book class should have.

All views should be a complete html page with a header tag and a body tag.

For all links you should use **tag-helpers**

You should also use **tag-helpers** in the form that submits a new book

Make sure that you add the changes to Startup.cs (see lecture from 27.Feb)

You should create the project using the command: **dotnet new web**

The project must have the name **BookList** because the given fake database expects the project to have that name!

Handin

You can handin in groups of two. The group member that hands in the project should state the full name and RU username of the other group member in a comment.

You are to handin a .zip file containing your entire project