# Exercise: Working with ASP.NET MVC

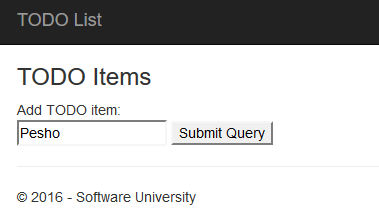
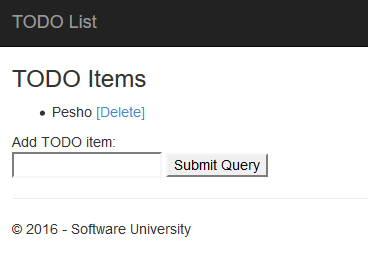
The purpose of this exercise is to make you **familiar** with ASP.NET MVC. After completing it, you should be able to **create a simple web application**.

This lab is part of the [“Software Technologies” course @ SoftUni](https://softuni.bg/courses/software-technologies).

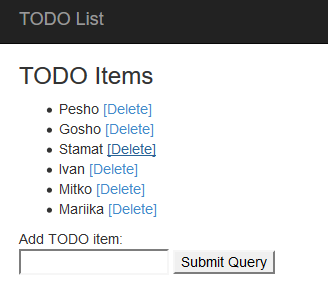
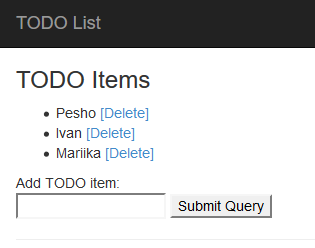
# Part I: To Do List

If you successfully complete all steps you will have “To Do List” application with the following functionality:

* **Add Item** – adds an item to the list. It should look like this:

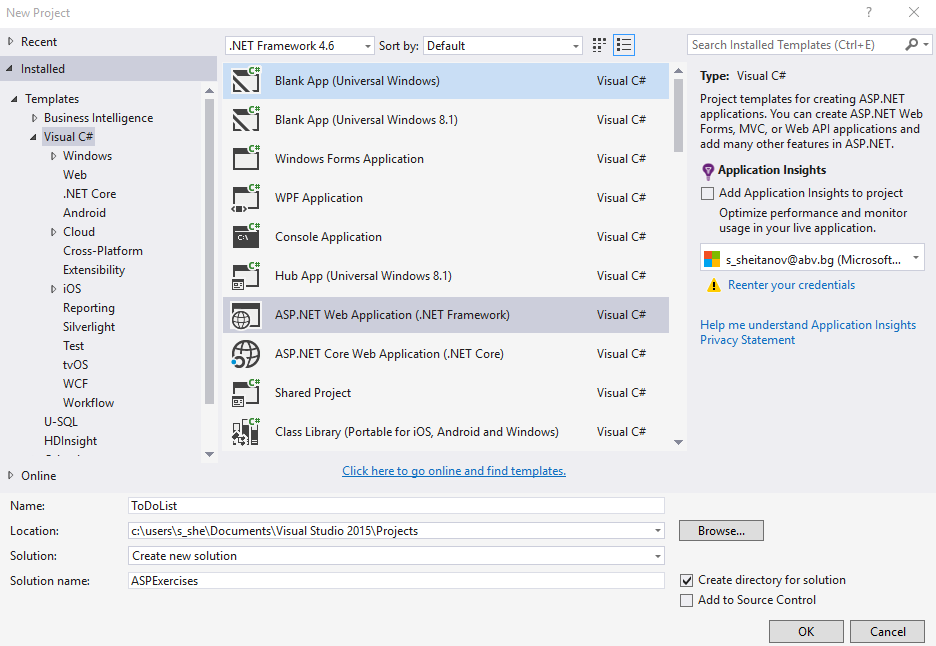
* **Remove Item** – removes the item from the list. It should look like this:

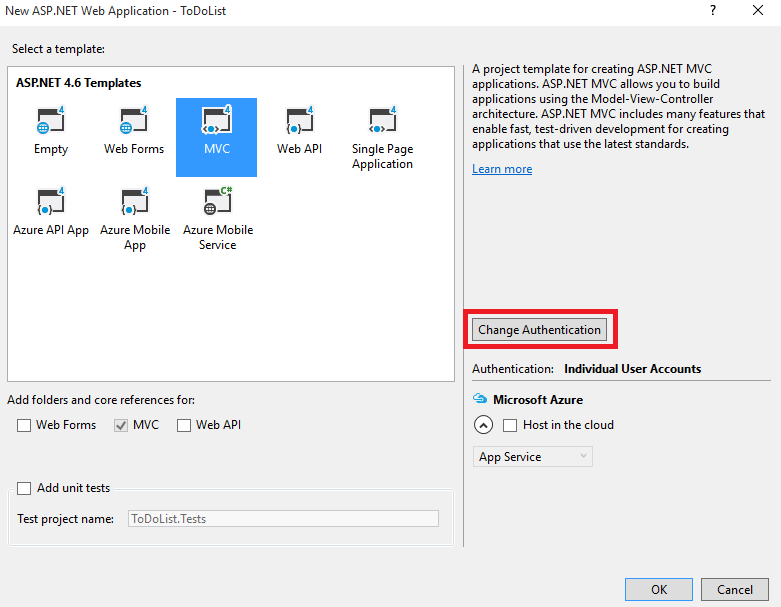
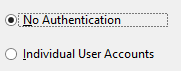
 

It should also keep your data if you exit the site (not the browser) and come back later.

## Create New Project

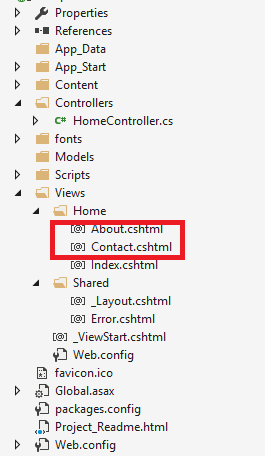
Let’s create new ASP.NET Project.



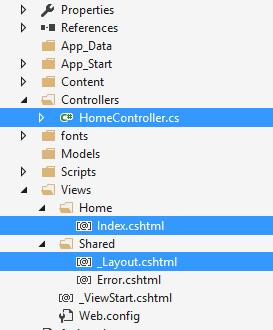
 

Change the authentication to “**No Authentication**”. Click OK, and you are done with this part.

Now we should delete the unnecessary files:



We will work with only 3 files to achieve our project:



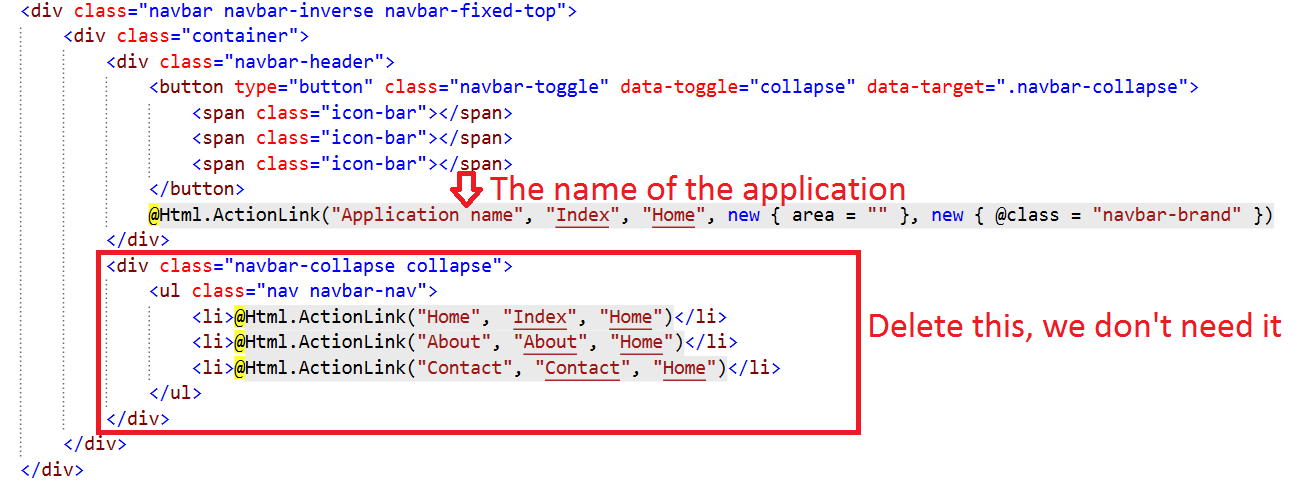
\_Layout.cshtml – We have our main template/layout for the application

Index.cshtml – The main (and only) page in our site

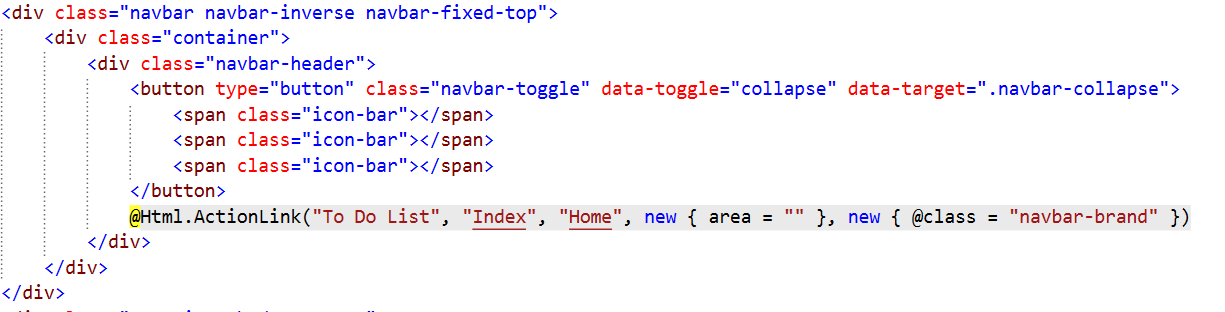
HomeController.cs – We will create our application logic in that file.

## Edit the Layout

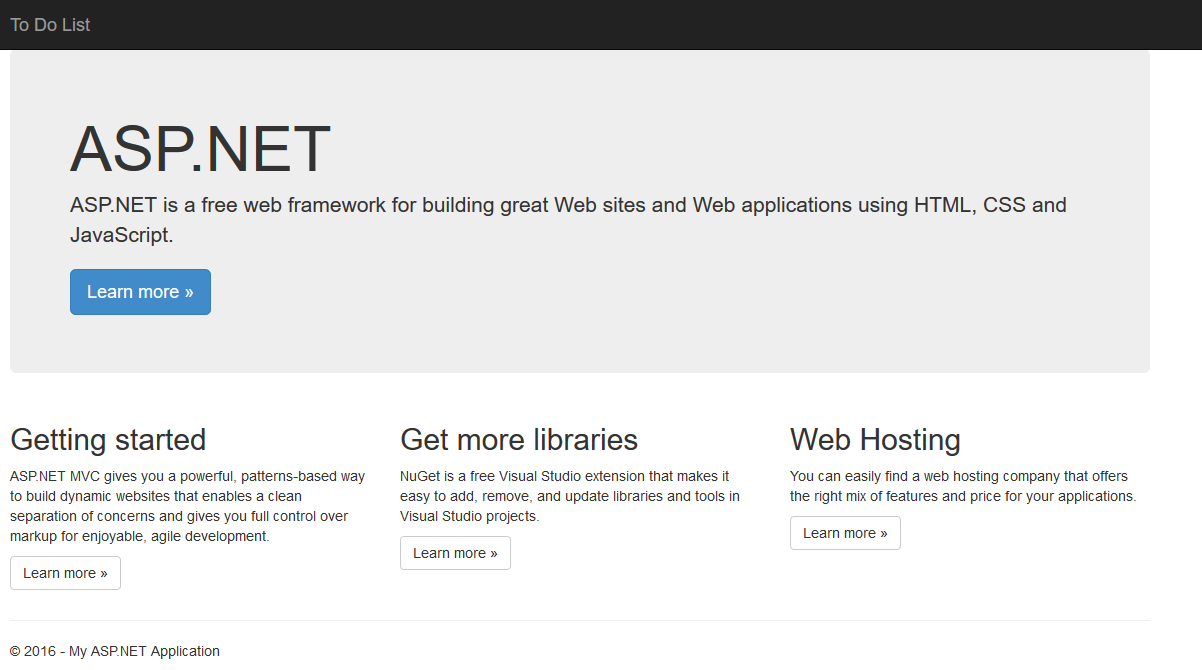
We will edit our \_Layout.cshtml now. Let’s make those changes:



We don’t need navigation menu, so we can delete the div that creates it. The only thing left to change is to rename our application. We should have something like this:



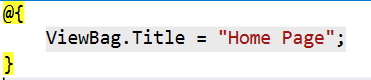
If we **start the application**, we should see this:



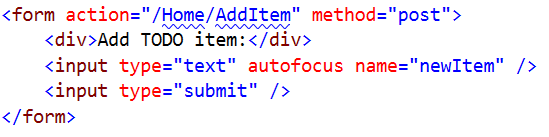
You can **edit the footer** in the same file, it shouldn’t be that hard.

## Edit the Main Page

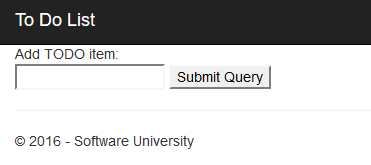
Our file for this step is index.cshtml. You should see bunch of stuff, but don’t worry, we will delete everything except for this part:



While you’re at it, **change the Title** to “To Do List”. Now we should **create a simple HTML form** like this one:



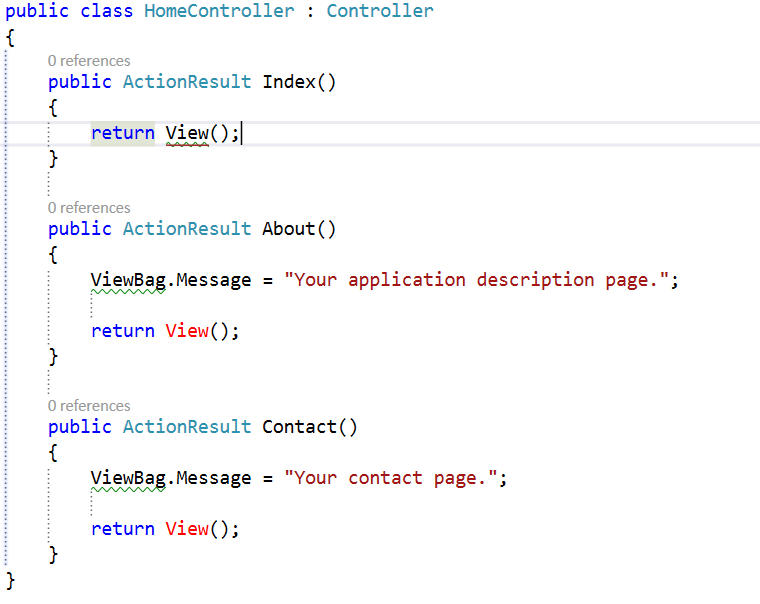
This form will add items to our list by **calling a method** in the HomeController.cs. Don’t worry for that now, let’s see what we have now:



Not bad, but we have more work to do. Before we show the contents of the list, we have to edit the controller.

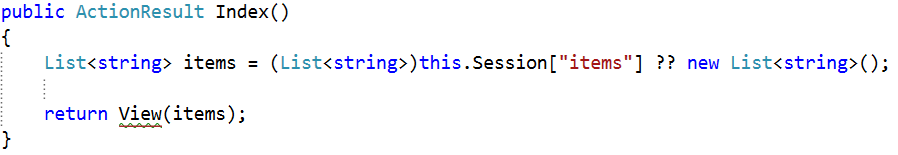
## Edit the Controller

Go to HomeController.cs. You should see something like this:



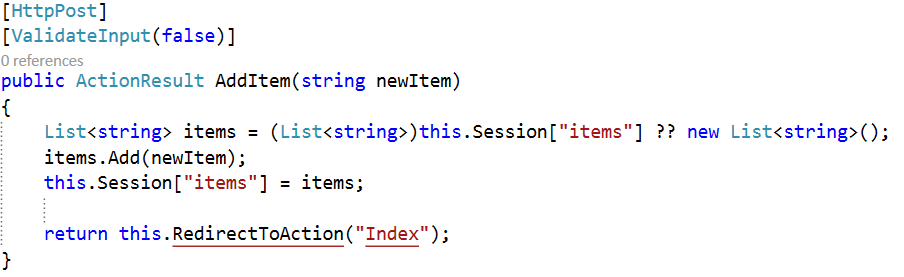
Delete the About() and Contact() methods. We don’t need them. Now we have only our Index() method.

We need to do slight adjustments to it. First we need to **get the list** from the browsers **session** (we will keep it there), and if there is nothing in there, we should **create a new list**. Then we need to **pass this list to the view**, so it can **render** **it**.



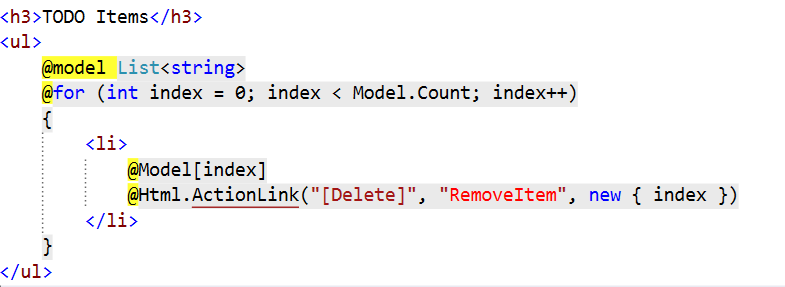
You can check the ‘??’ operator in google. It’s called **null-coalescing operator**. **If the left part is null**, it **returns** the **right** **part**.

Now we need to **write a simple method** that **adds** our **items** from the **textbox** to **our** **list**:



## Print the List

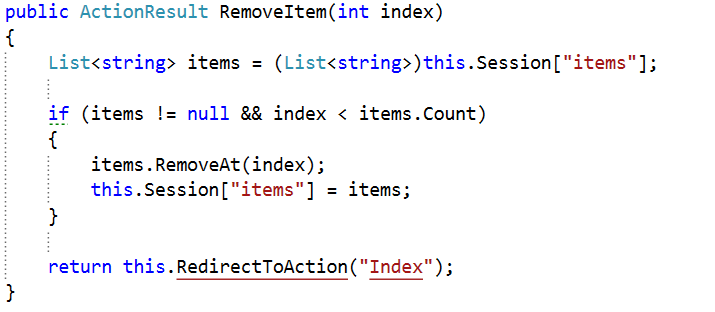
Now we should go back to the Index.cshtml and print the list, using single for loop. Write this code **above** our HTML form:



What we do here? First we get our list and start a for loop that will print our items. For each item, we add ActionLink to it, which will later help with deleting the item. It basically **binds a hyperlink to a method** in our **controller**. The last parameter is the **index of the element** that this link will delete. It’s complicated, but you will get used to it, don’t worry. We are done with this file.

## Delete from the List

The last thing we need to do is to create a method that **deletes an item** from the list. As you probably already know, we will do that in the HomeController.cs. Let’s write the following method:



This method gets our list. If the **list exists** and the **index is less than** the **count** ofitems, it removes the element. **You should** **add** a **validation** that **checks** if the index is **more than zero**.

That is everything! Now you should fully functional application, like the one on the pictures in the beginning.

You can continue with the “Book Library” application on the next page.

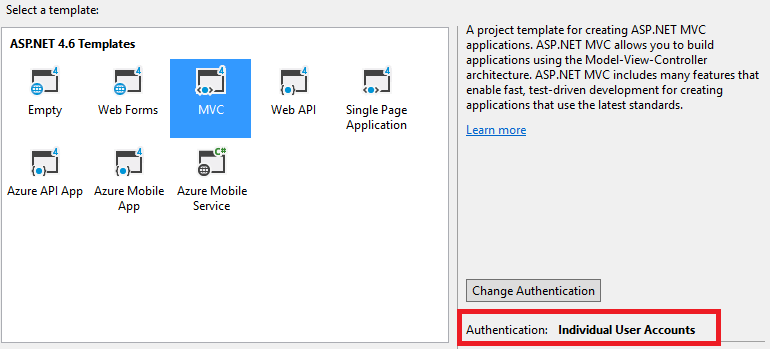
# Part II: Book Library

If you successfully complete all steps you will have “To Do List” application with the following functionality:

* **Register User**
* **Login User**
* **Create New Book**
* **Edit Existing Book**
* **Delete Existing Book**
* **List All Books**

## Create New Project

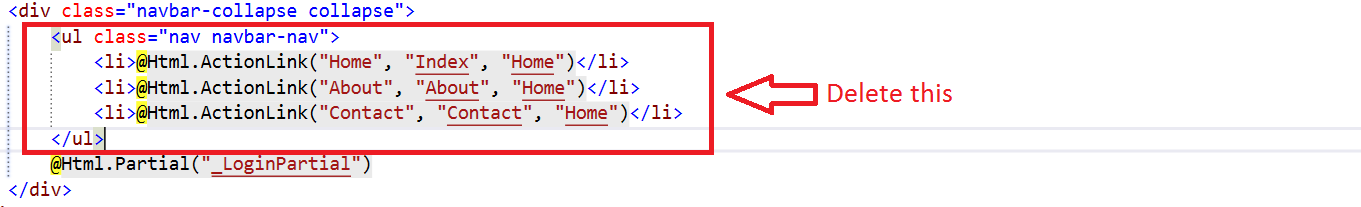
Creating the project is similar to the “To Do List”. This time leave the authentication to “**Individual User Authentication**”:



**Important: Don’t run the project before step 10.**

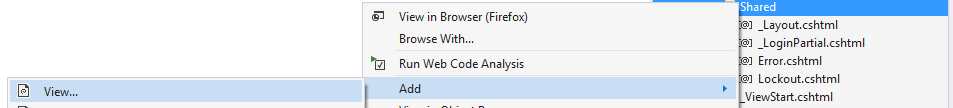
## Edit the Main Layout

Let’s start by deleting the unnecessary lines from „\_Layout.cshtml“.

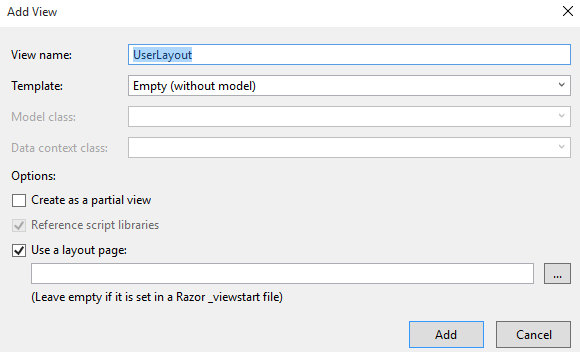


## Create User Layout

Now we must right-click on the “Shared” folder and add new **View**:

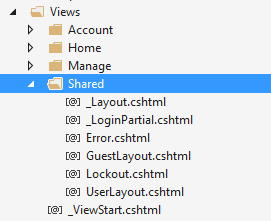


Name that view “UserLayout”:



## Create Guest Layout

Create a new layout and name it “GuestLayout” and you should have the following files in your “Solution Explorer”

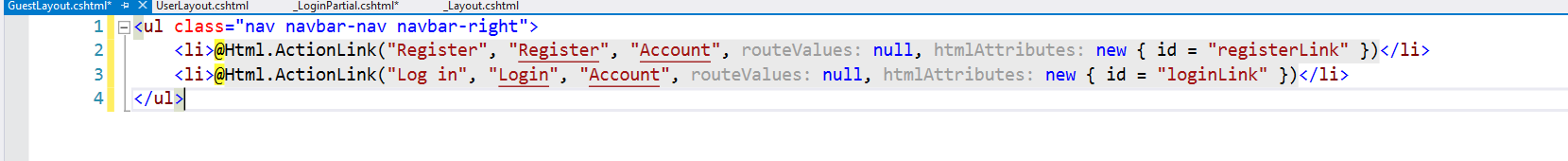


## Edit the Authentication Layout

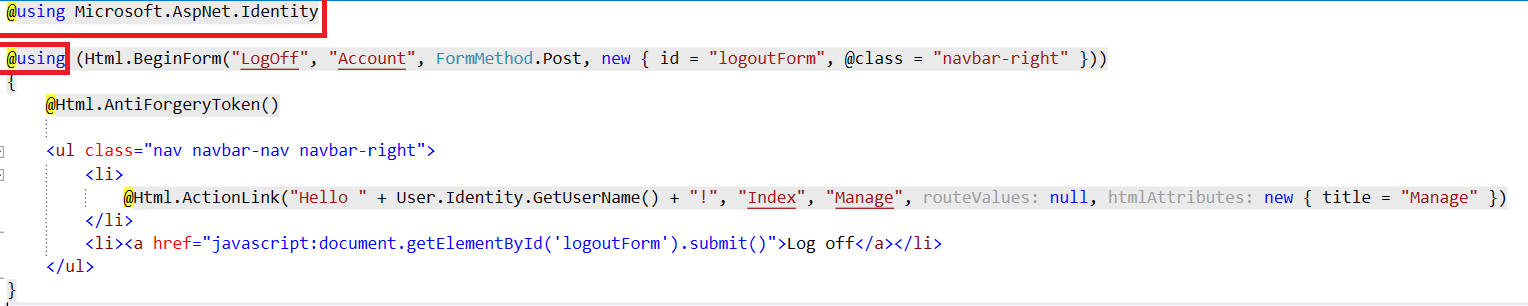
Now we must go to “\_LoginPartial.cshtml”. We will see something like this:



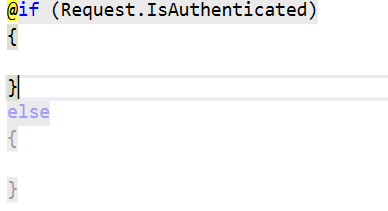
Delete everything from your “GuestLayout.cshtml”. Copy the contents from the else statement and paste it to your “GuestLayout.cshtml”. You should have something like this:



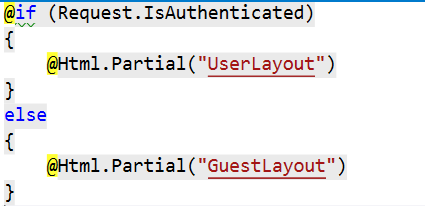
Now copy the code from the if statement and paste it to your “UserLayout.cshtml” (delete everything else from the user layout). Your “UserLayout.cshtml” should look like this:



The “\_LoginPartial.cshtml” should look like this:

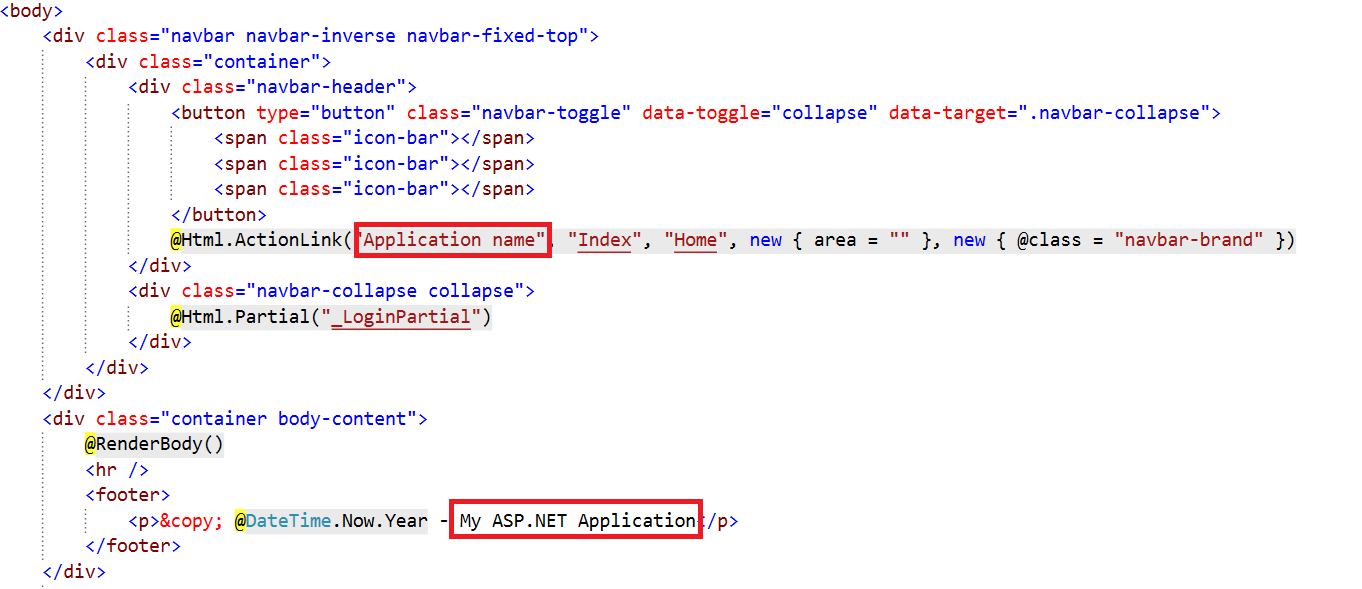


Add the following lines:



## Edit the Application Name

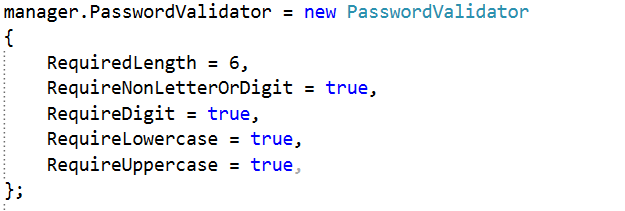
Go to your main layout and edit the following lines:



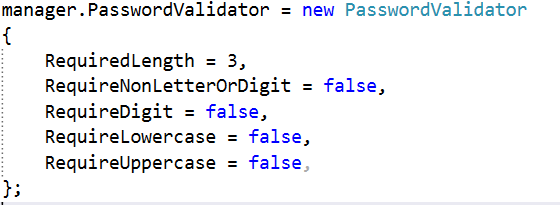
We are done with the layout for now.

## Simplify the Registration

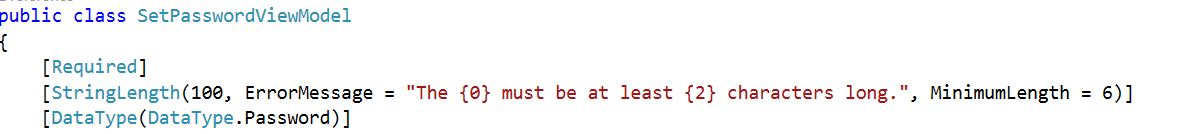
Go to your “App\_Start/IdentityConfig.cs” file. Find the following lines:



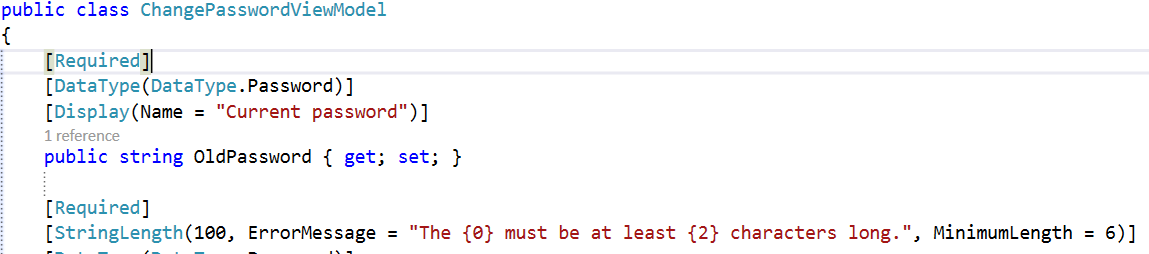
Edit them to receive this result:



This will simplify our password. Now we need to edit the models that validate our data. Go to the “ManageViewModels.cs”. Find this line:



Edit the minimum length to 3 symbols. You will find similar line in the ChangePasswordViewModel class. Edit the minimum length there too:



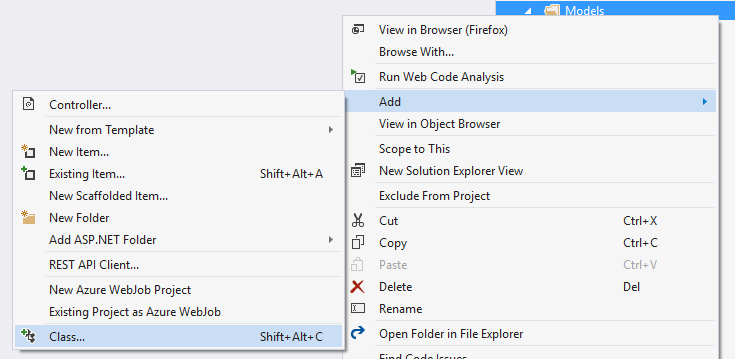
Use 3 symbols again.

Finally, we should go to the AccountViewModels.cs file. Find the lines that validate the password length, and change the value to 3 symbols. Here is how it should look like:

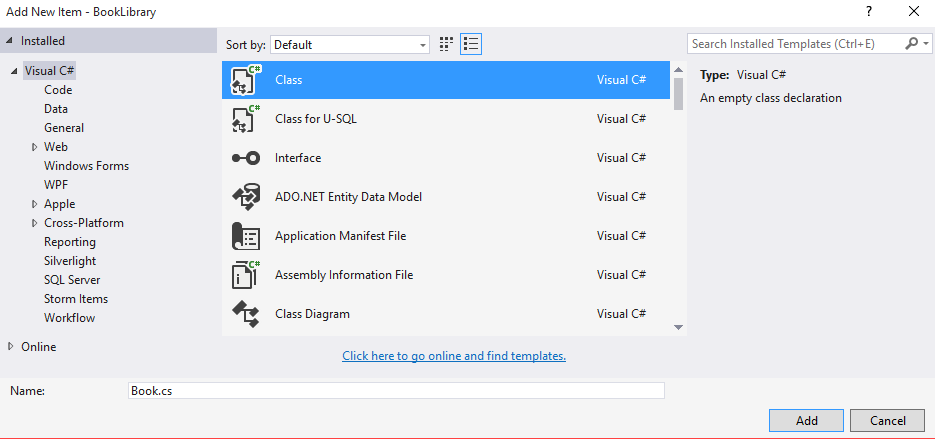


## Create the Book Model

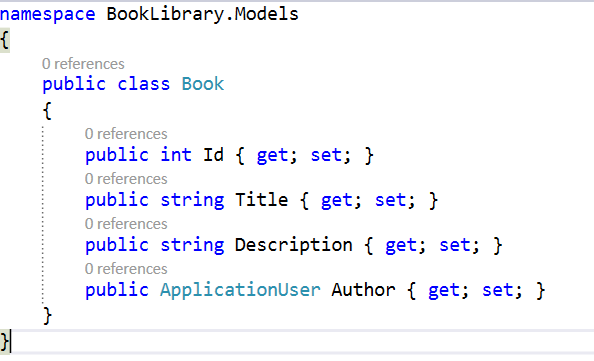
Create a new class in your **Models** folder:



Name the class Book.cs:



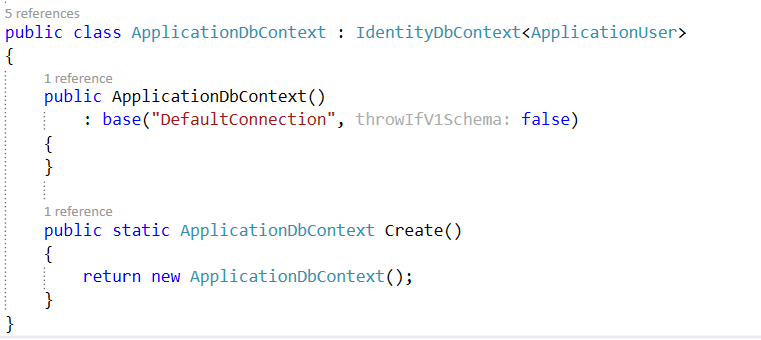
Add the following code:



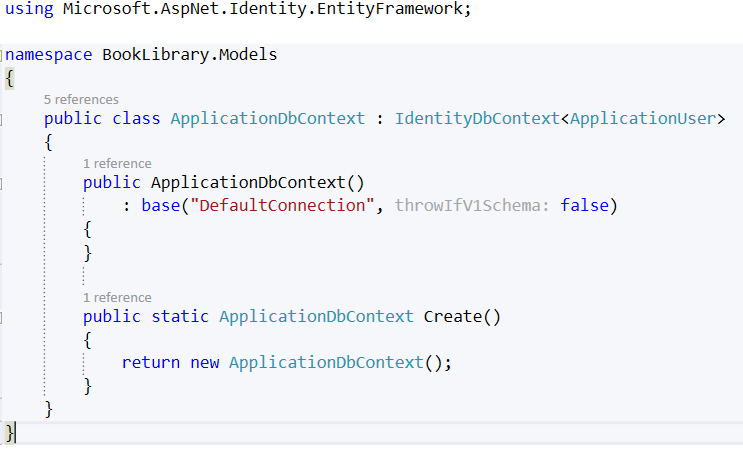
## Extract the DbContext

Create a new model and name it “ApplicationDbContext”.

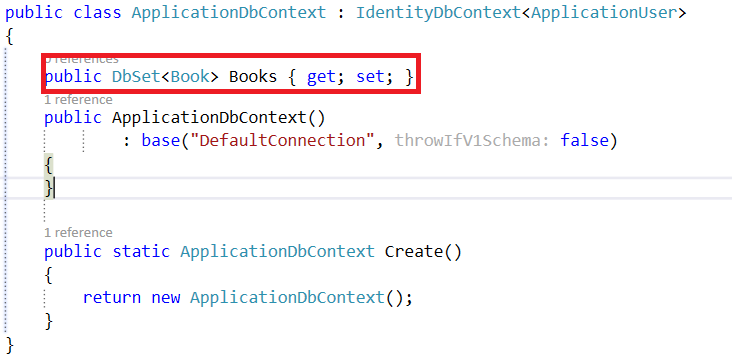
Go to your IdentityModels.cs file and find the ApplicationDbContext class.



**Cut** the class and paste it to the file you’ve just created. **Replace** the **existing** **class** in the file. You should have this:

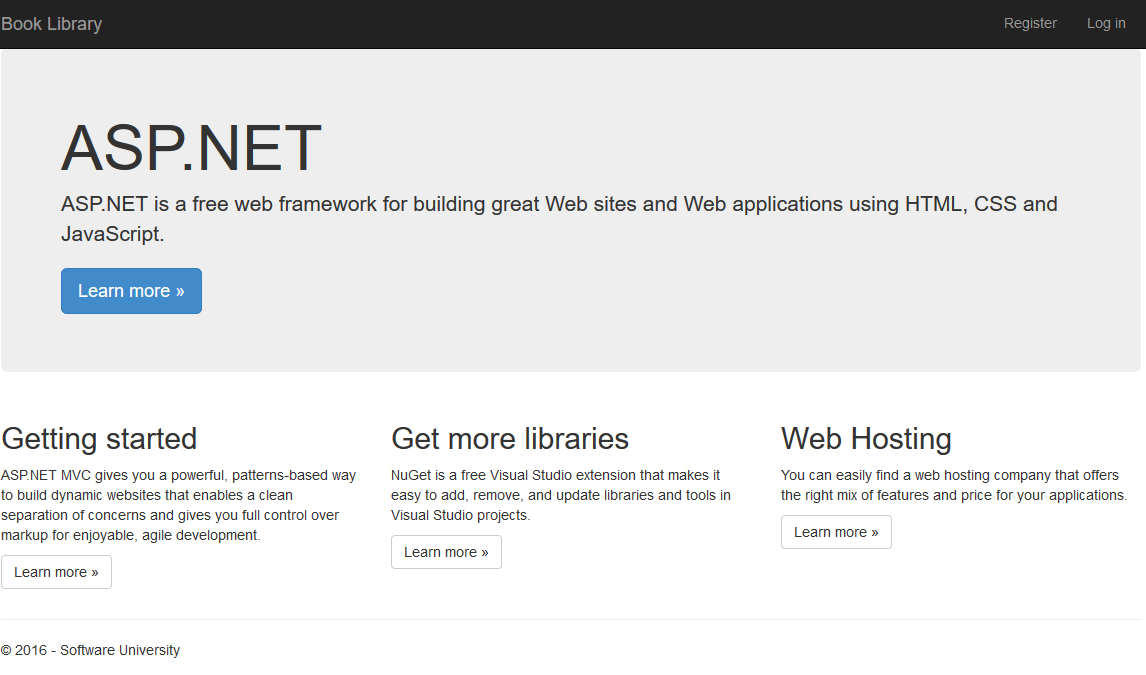


Add the following code:



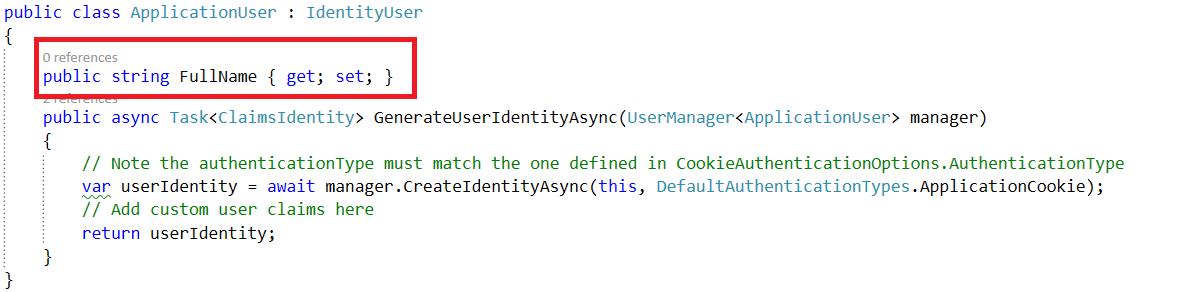
## Run the Application for the First Time

Don’t register a new user just yet! You should see this:

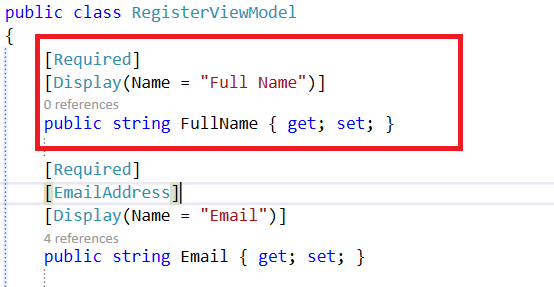


## Add Full Name to Our User

Go to your IdentityModels.cs file and write the following line:



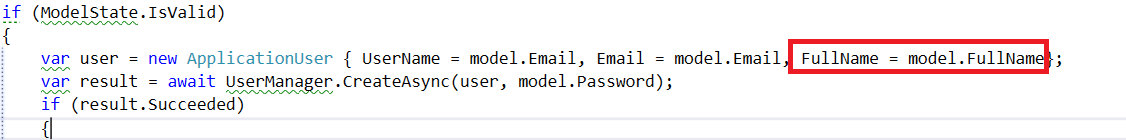
Now go to the AccountViewModels.cs and find the RegisterViewModel class. Add the following lines:



We are almost done. Go to your AccountController.cs. Find the Register **post** method:



Add this, to the existing line:

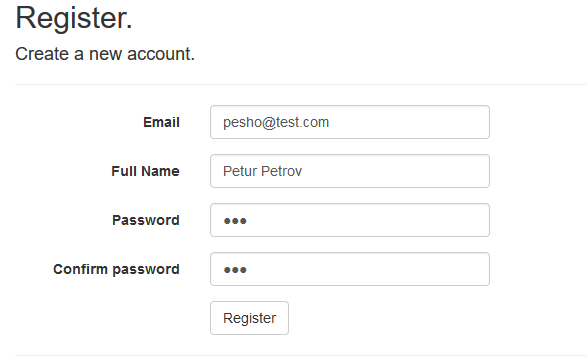


Finally go to the register view located in “Views/Account/Register.cshtml”. Write the following code:



## Register New User

Run the application and try to register a new user:

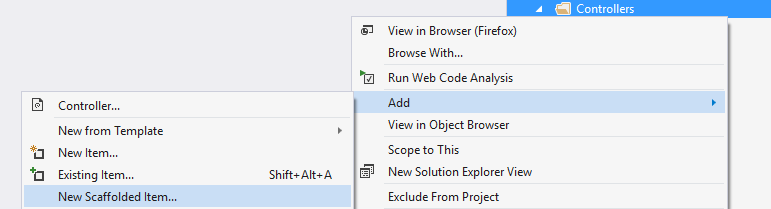


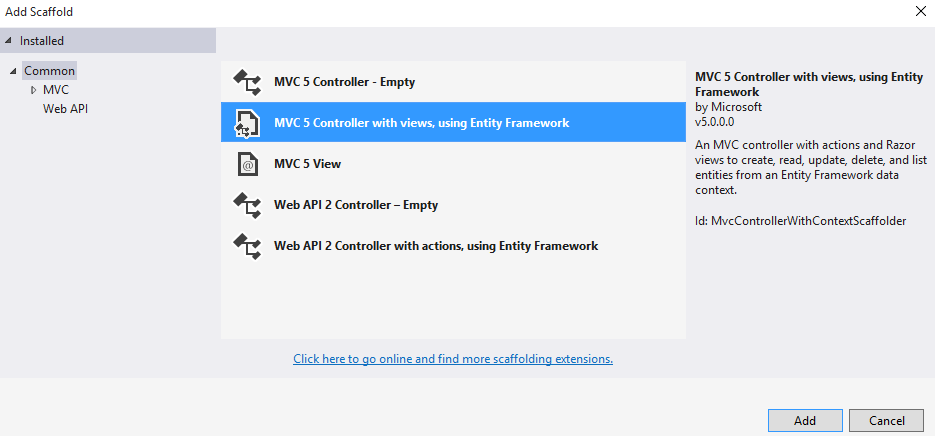
If it works you will be logged in, and you will see welcome message like this:



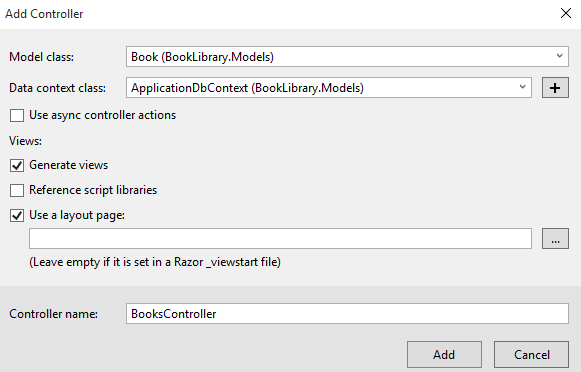
## Create the Book Controller

Add new scaffolded item to your **Controllers** folder:



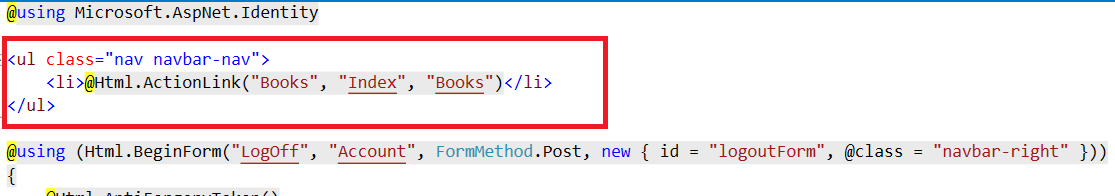


Use your **Book** **Model** and your **ApplicationDbContext** as shown in the image below:



## Edit the User Layout

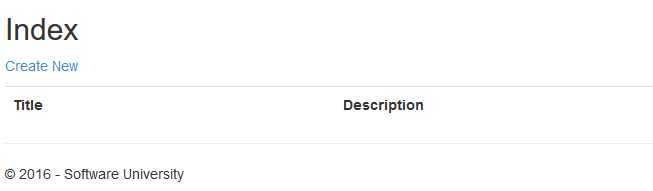
Go to your user layout and add the following code:



Now if you login, you will see this button:



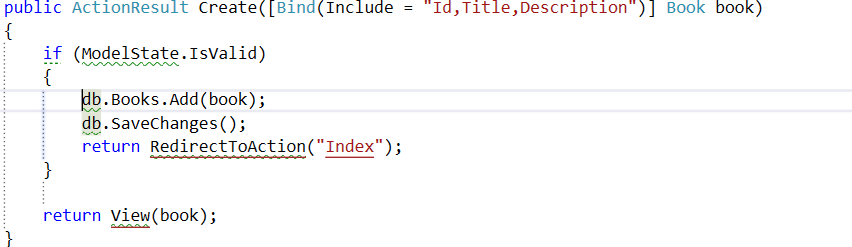
When you click on it, you will see this page:



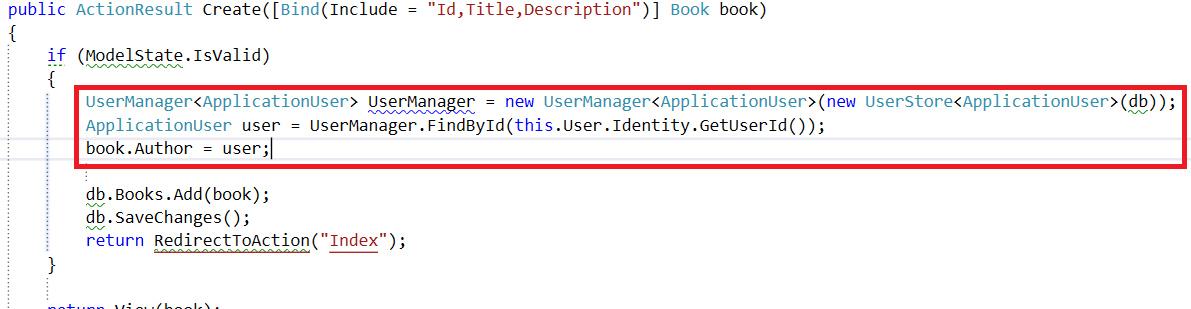
Don’t create new books yet!

## Add Author Info

Right now our **CRUD** operations don’t add the current user to our DB. We need to fix that. Find the Create method in your BooksController. It should look like this:



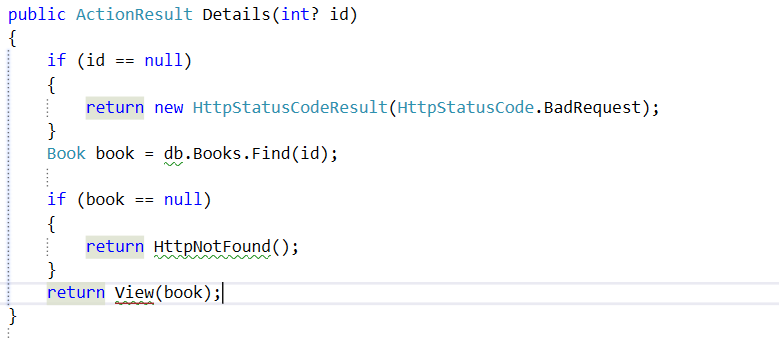
Add the following lines of code to the if statement:



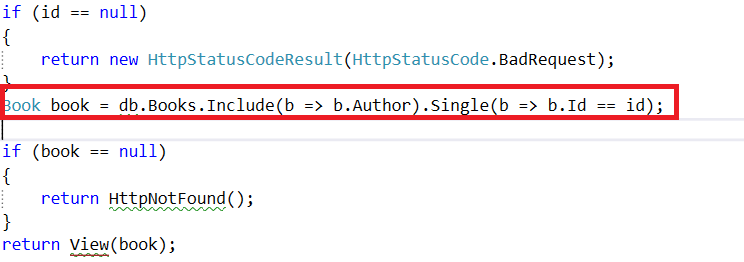
That will add the author to the book object.

## Show the Author

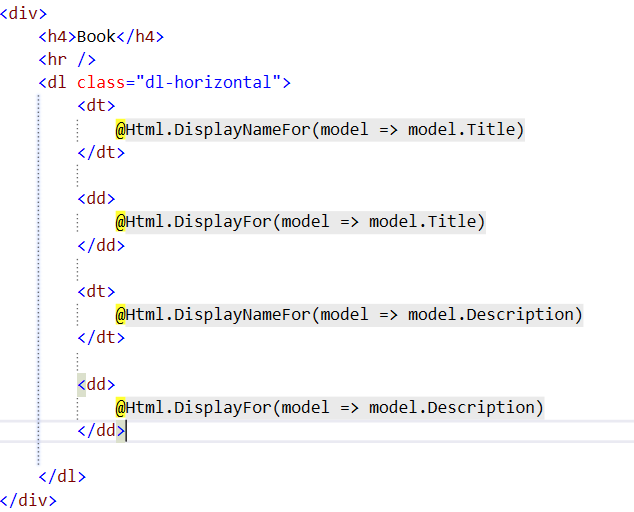
Our final step is to show the author of the book. We will do that by finding the Details method in our BookController. You should have something like this:



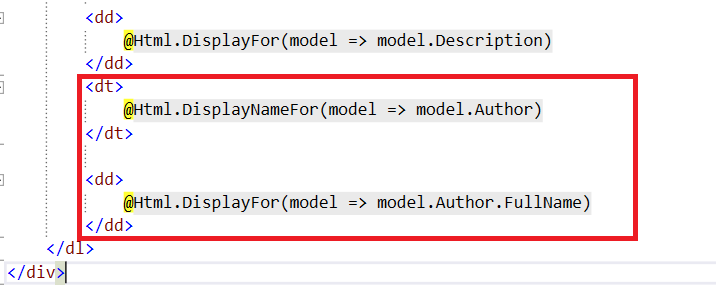
We will edit one of those lines, with LINQ query:



We are almost done. Let’s go to our “Views/Books/Details.cshtml” file. It should contain a div tag, that looks like this:



Add the following element:



You are ready to play with your Book Library now!