# Exercises: ASP.NET Core Introduction

This document defines the **exercise assignments** for the ["C# MVC Frameworks" course @ Software University](https://softuni.bg/courses/asp-net-mvc).

# Fluffy Duffy Munchkin Cats (FDMC)

You have been tasked to create a very simple, very non-quality code web application. The offer sounds pretty absurd and disturbing to you since you are a good (non-Indian) developer and such low-quality applications are not in your style, but hey… they pay well.

The application you must implement is called FDMC, an online Cat Exhibition website which should support very basic functionality.

Use ASP.NET Core. It’s up to you to **decide whether to create a Razor page or to use the MVC pattern**. I recommend that you try out both, just to get a little taste of them.

## Implementing the Data Layer

Your employers wanted you to store the database on a **.txt file** but you deeply **DISAGREED** with that idea. And they said you can use your black magic databases. You decided to use **Entity Framework Core**.

You have a single model – Cat, which has a name, age, breed and imageUrl.

## Implementing the Index page

On the home route (“/”), you should see a heading (h1) with the name of the project (Fluffy Duffy Munchkin Cats).

Below it you should see a very simple unordered list of items – each item being a cat. The items should be links, which should hold the cat’s name and should lead to a route “/cats/{catId}”.

Below all the links, there should be a button “Add Cat”, which leads to route “/cats/add”, where a simple form is rendered to **add** a **cat** to the application’s databae.

## Implementing the Cat Add Page

The cat add page should hold a simple form, with 4 input fields, for the:

* Cat Name – text
* Cat Age – number
* Cat Breed – text
* Cat Image Url – text

And it should submit a POST request to route “/cat/add”.

The cat add page should hold a heading (h1) – “Add Cat”.

## Implementing the Cat page

Each cat should have a page for itself, which is accessed through the link on the Index page.

It has a heading (h1), holding the **cat’s name**.

It has an image with an alt attribute – the **cat’s name** and a **max width** of 300px.

It has a paragraph, holding the **cat’s age**, and a paragraph, holding the **cat’s breed**. Both should be **bold**, and should have a prefix with the **property’s name**.

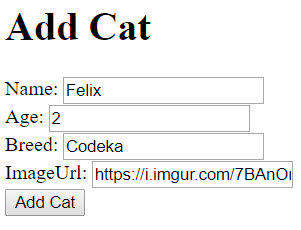
## Visual Looks

The application pages should look like this:

### Index page

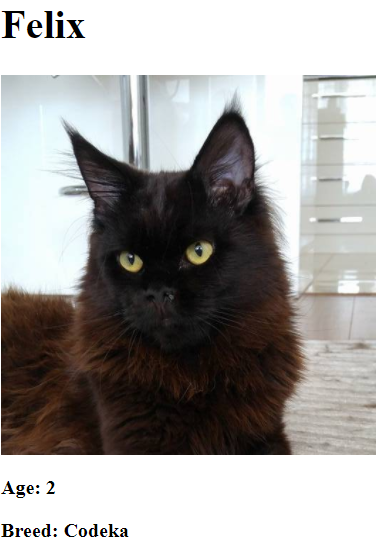


### Add Cat Page



You can think about displaying “ImageUrl” in a better way. There are a lot of ways to do that.

### Cat page



## Refactoring

Now, we all can agree that such an application wouldn’t look well no matter how well it has been coded, but make sure it follows at least a bit of the rules of the high-quality code design. Search the internet if you have to for hints on how to make it better.