# IN710 OOSD 2017 Practical 11.2 – Dog Selector

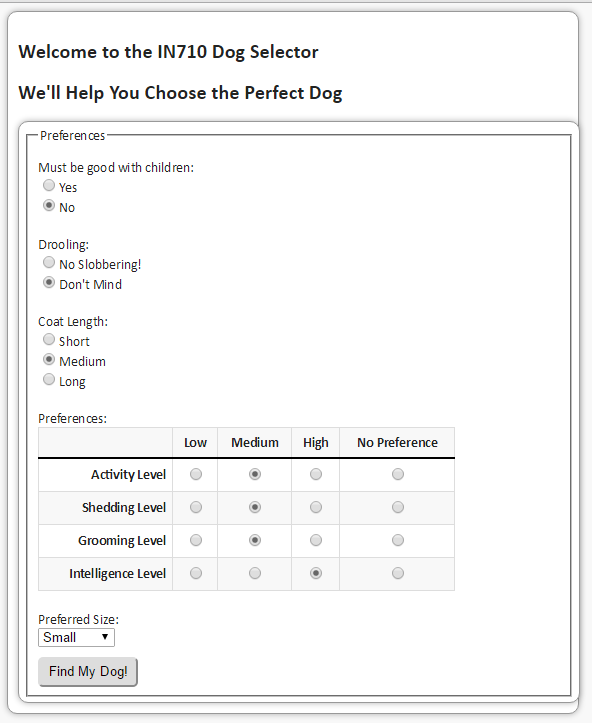
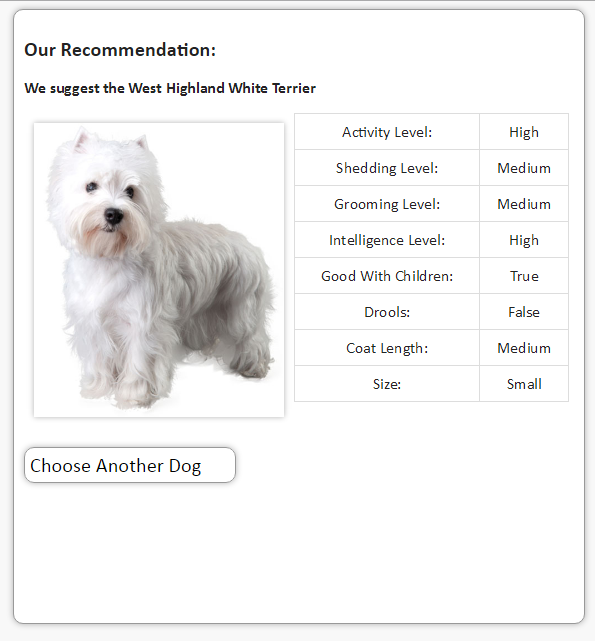
Use ASP.NET MVC to build a web site that helps people to identify the best breed of dog for their lifestyles and preferences. The site should present a form that queries the user’s preferences on (at a minimum):

* Activity level
* Grooming requirements
* Intelligence
* Amount of shedding
* Coat length
* Size
* Behaviour with children
* Tendency to drool

Be sure to use appropriate controls for each field in your Form.

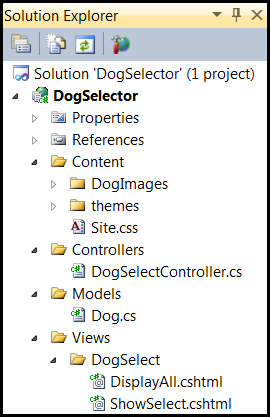
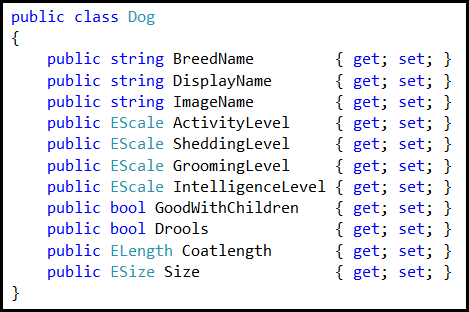
Based on the user’s input data, the site recommends the best matching dog, and presents a picture and details of the dog’s features.

Examples of the two screens from my solution are:

### Project Architecture

Before starting to build your site, carefully plan your Model class(es), desired Views and required Controllers. For comparison, the relevant portion of my Solution Explorer is shown below (other architectures are also possible). My Model contains one class (also shown) and some enum types (not shown).

### Simulating a Database

It is, of course, possible to use a real back-end database with ASP.NET MVC web applications. One can use either the ADO.NET classes, or the Entity Framework ORM. We will look at how to do this in our next lecture. For this practical, however, you are to simulate your database with an in-memory collection of Dog objects. On the I: drive you will find images for 15 different dogs; your application must choose from this set of dogs (add more if you wish, but you have to use at least these 15). You will also find a method to populate a List<Dog> that can serve as the database for this application. This code assumes the necessary enums have been declared somewhere in scope. Modify the code as required for your architecture if you wish to use it; it is only provided to save you some typing.

### Making the Recommendation

Your project must compute a sensible recommendation based on the user’s input data. You must devise and implement a reasonable algorithm for this.

### Maintaining good OO

Although the MVC architecture is very different from the desktop architectures we have been using so far this semester, the C# controllers are still OO and should still display all the qualities of good OO code such as proper modularity, high cohesion, low coupling, sensible variable naming, no integer literals, etc. Similarly, the Views should be written in clean HTML with clean CSS. Always keep your code elegant.