.NET MVC Applications

Three main components:

* Model
* View
* Controller

For this particular application, these are the files that really shape its core behavior and appearance:

‘Models’ Folder:

* InquiryViewModel.cs

‘Controllers’ Folder:

* InquiryController.cs

‘Views’ Folder:

* Inquiry/Success.cshtml
* Inquiry/Test.cshtml

How does it all work together?

1. Controller processes Model data and updates the Model
2. Model holds the application data
3. The View renders data from the Model

In .NET MVC:

1. View triggers controller through form submissions, hyperlink clicks, or URL visits in address bar (POST or GET requests, respectively)
2. Controller methods return Views in the form of an ActionResult Object

Concrete examples of process:

GET Request

1. User visits Inquiry/Test in browser
2. Test() InquiryController method is triggered
3. Test.cshtml View page is returned and displayed in browser

POST Request

1. User makes form submission on Inquiry/Test page
2. Test() InquiryController method with [HTTPPost] attribute is triggered
3. Test() method returns view based on Model state

Razor Syntax

In brief, Razor exists to allow HTML to be rendered dynamically based on application state. For example, if we wanted to render a table of data on our website, but weren’t sure if the table were 10 or 20 rows, we would want the HTML to be written according to how many rows there currently are.

Razor has an HtmlHelper class that has a bunch of C# methods for generating HTML rather than us explicitly writing the HTML. I’m sure this has some use, but I experimented a little and saw that we can write basic HTML instead.

As a comparison, I created a mirror version of Test.cshtml, called Form.cshtml – and this works just the same.