MONASH University

MONASH INFORMATION TECHNOLOGY

FIT5192 Module 2 Internet Applications Development Lecture 10 – Part I

GROUP OF EIGHT AUSTRALIA

MONASH University

Lecture Overview

1. Introduction
2. MVC pattern
3. ASP.NET MVC
4. Razor View Engine

GROUP OF EIGHT AUSTRALIA

MONASH University

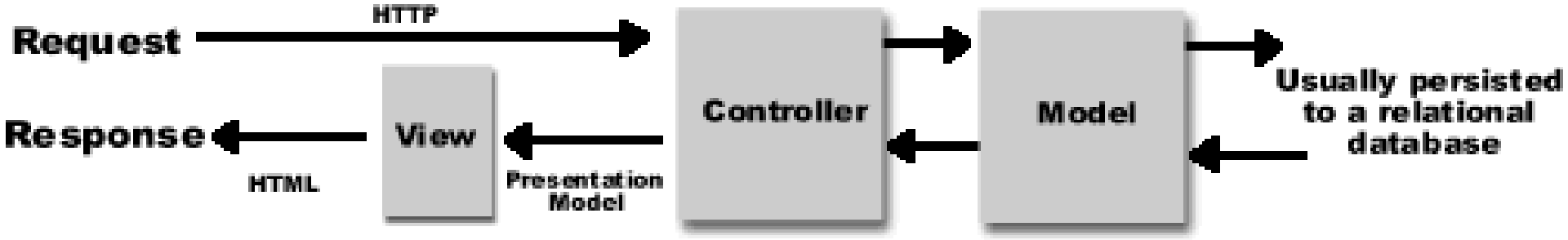
Introduction

MVC Pattern

Model

View

Controller



Model

- A class or set of classes that describes all the business logics and additionally handles data access for an application.
- Also contains code that defines its relationship with other models
- Defines the data validation rules to be used when adding or updating data.

Controller

- Controls the application flow or logic of the application
- Controller logic decides what response is to be generated
- Controller logic normally contains calls to models to access data, and also other functionalities like access control checks etc.
- Controller passes the response (output) to the view

View

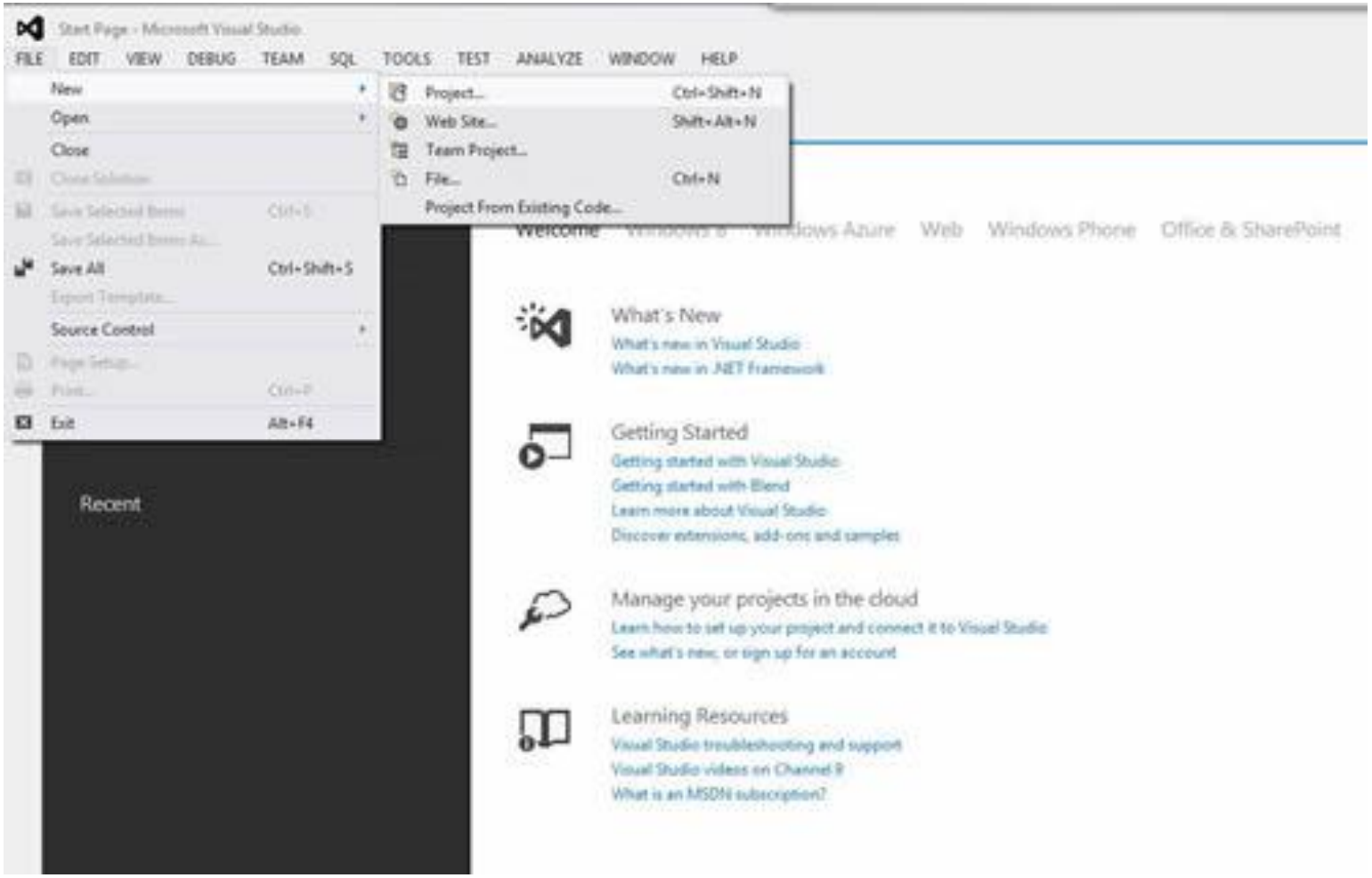
- Is the outputs or responses that are sent back to the user once a request is processed.
- Consist of markup (like HTML) code with embedded .NET code
- Can also be other forms of output like XML, PDF documents etc
- Views can be thought of as the presentation layer of an application and ideally should be as "dumb" as possible

Advantages of ASP.NET MVC

- Enables clean separation of concerns
- Ideal for Test Driven Development
  - was developed with TDD in mind
- Provides full control over rendered HTML
- SEO and REST friendly URL patterns
  - Search Engine optimised
- Easy integration with various JavaScript frameworks
- No Viewstate and Postback
- Easily extensible and pluggable components

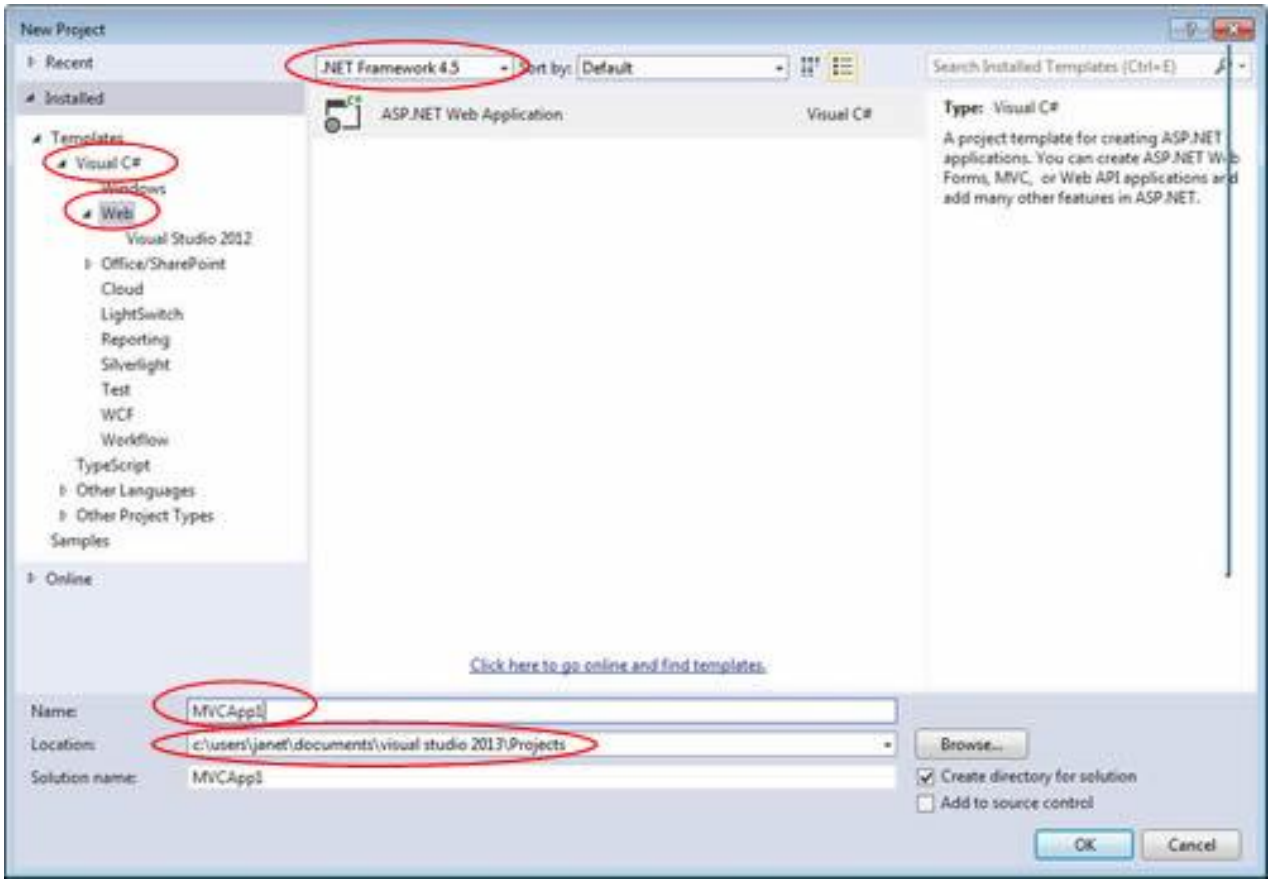
MVC in ASP.NET

Creating an MVC Project



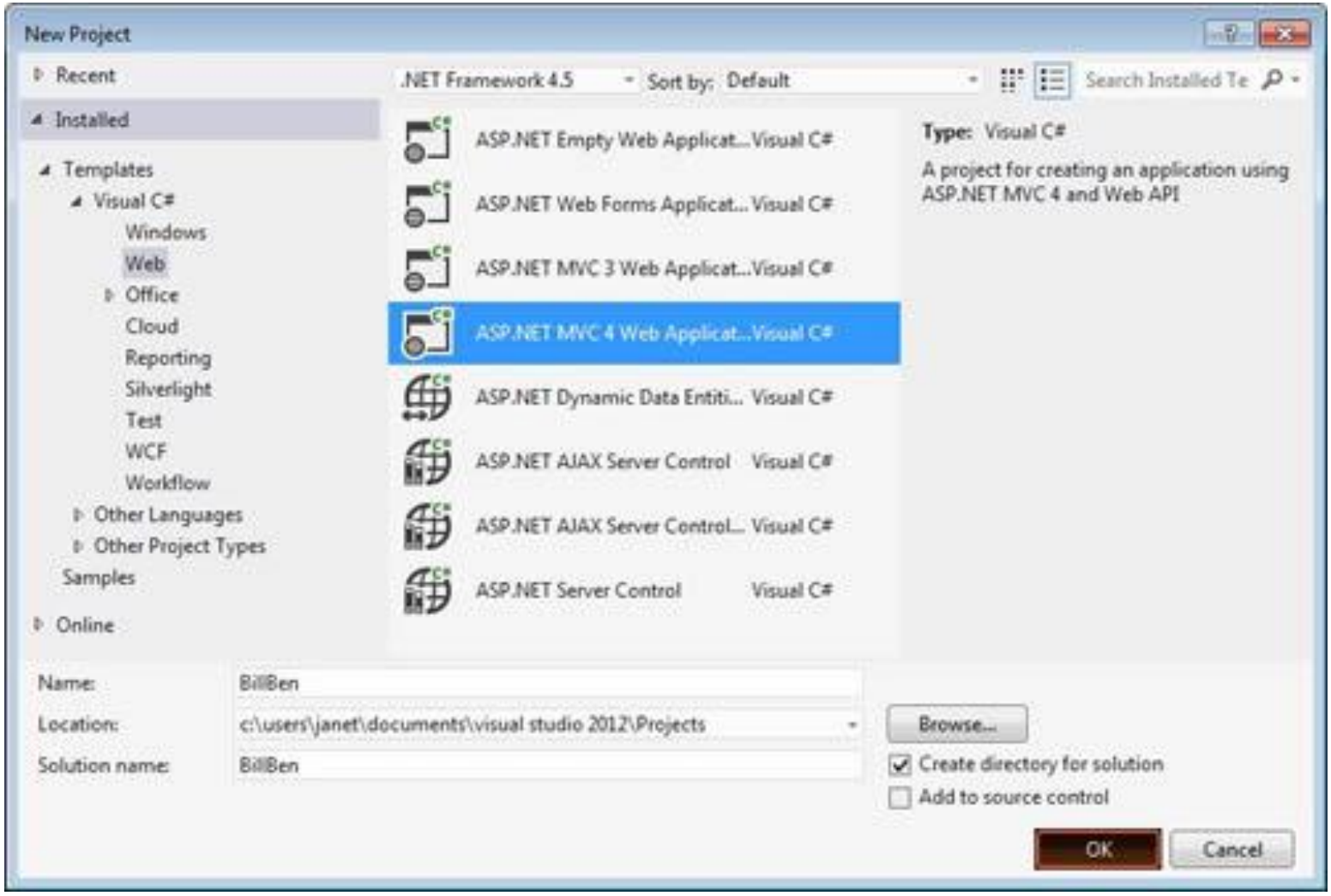
For Razor's view projects use this approach

(Recommended for MVC Task, see Topic 11.2)

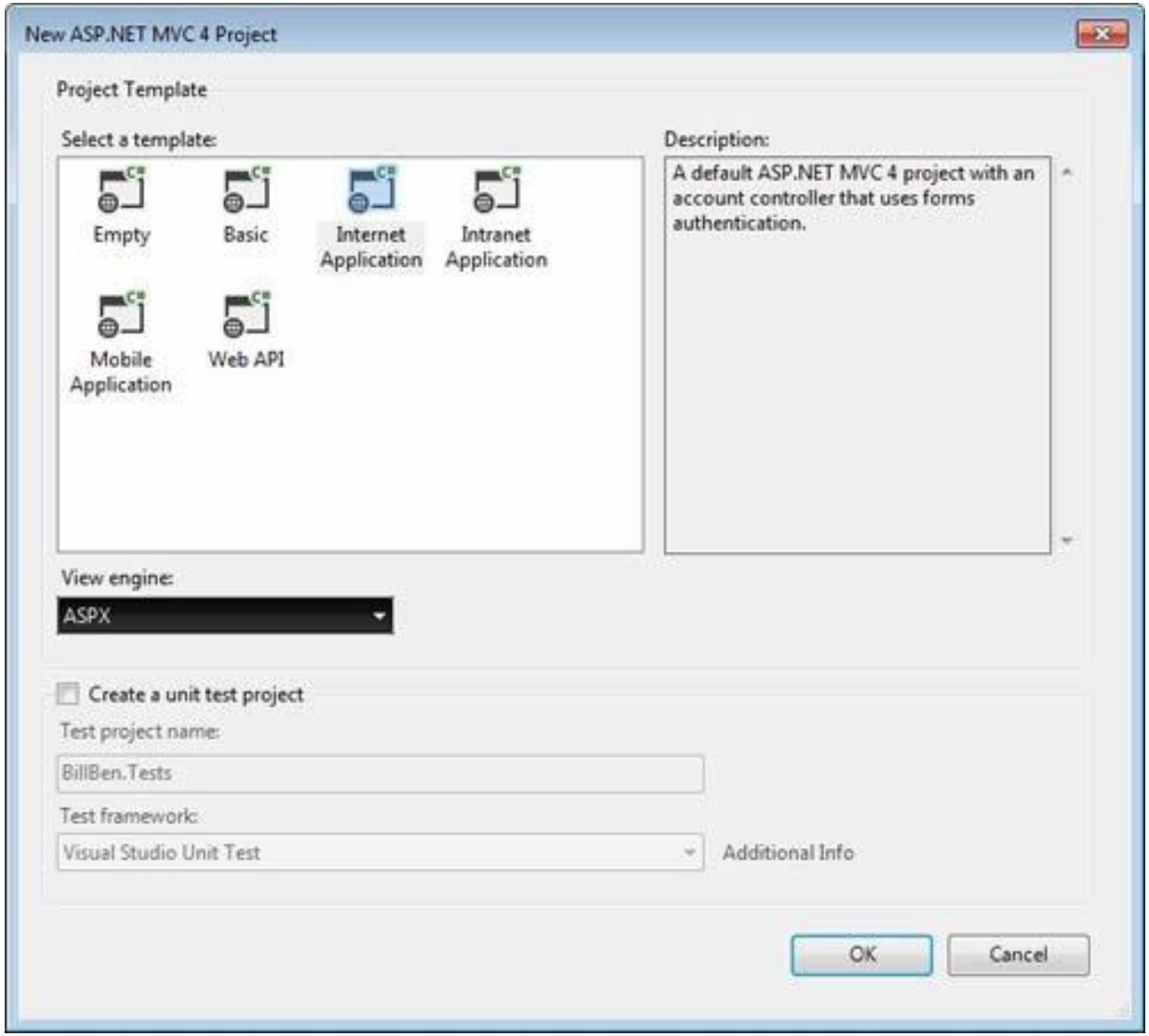


For aspx approach, see following slides

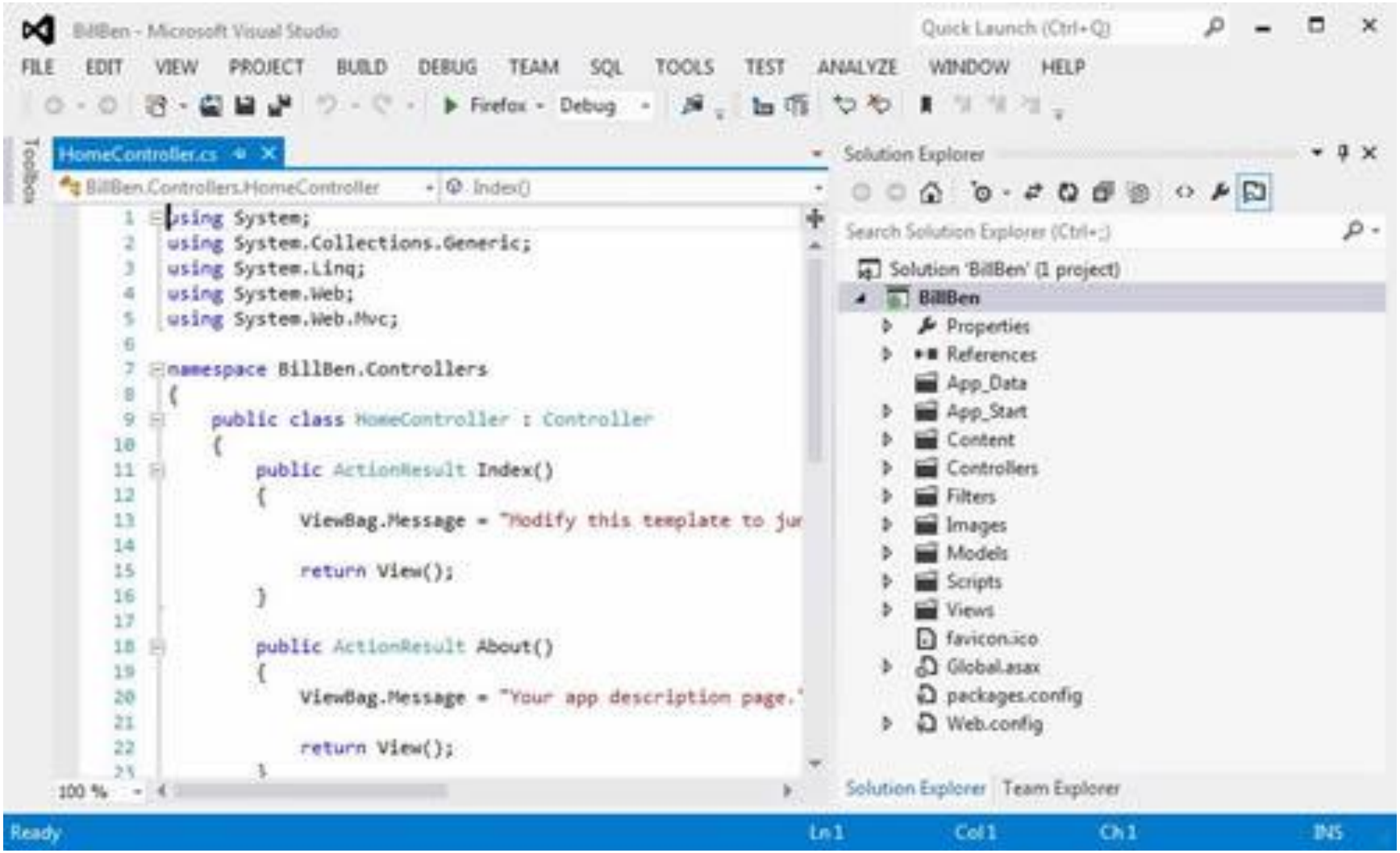
Select MVC C# Project



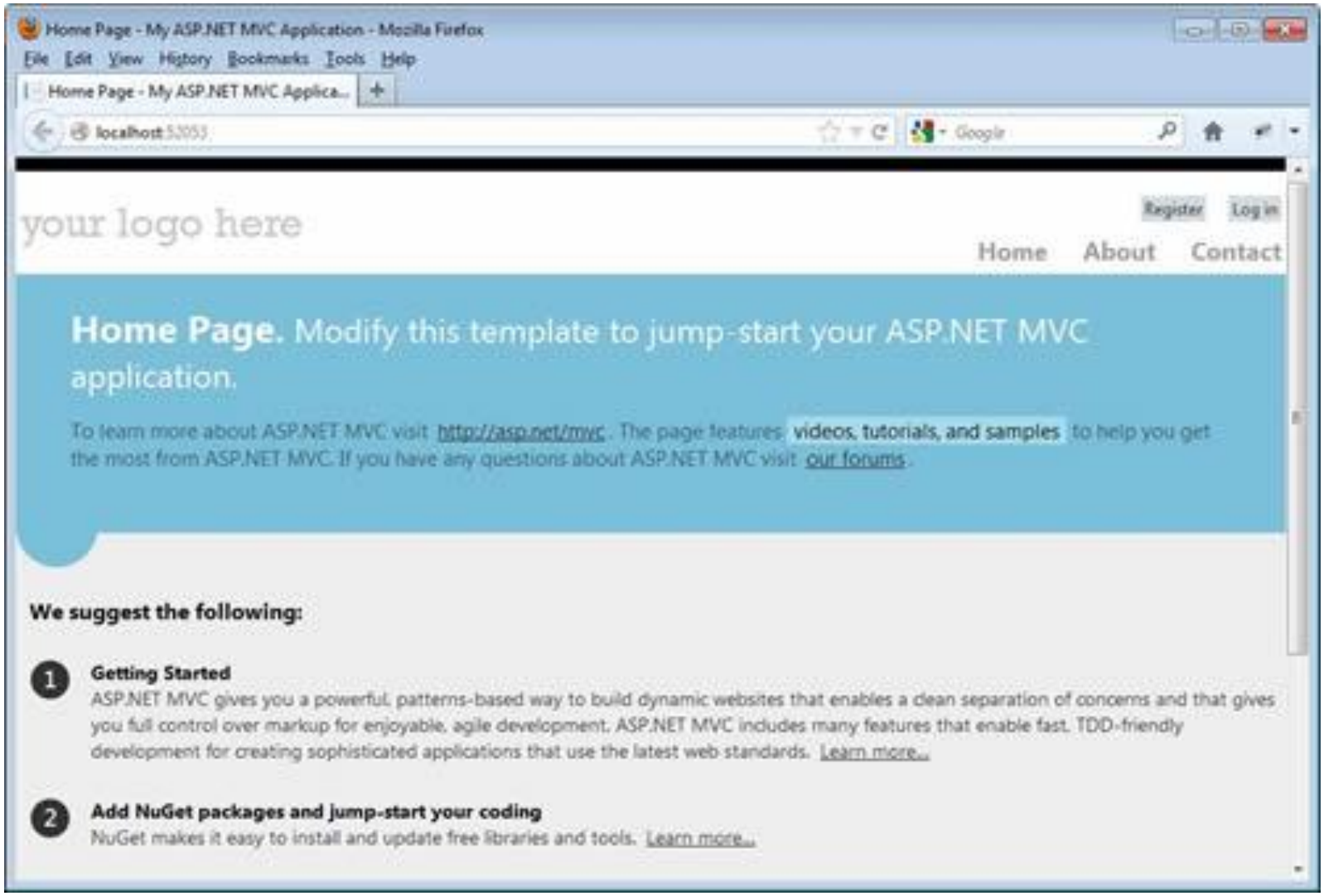
ASPX View Engine is used (Internet Application)



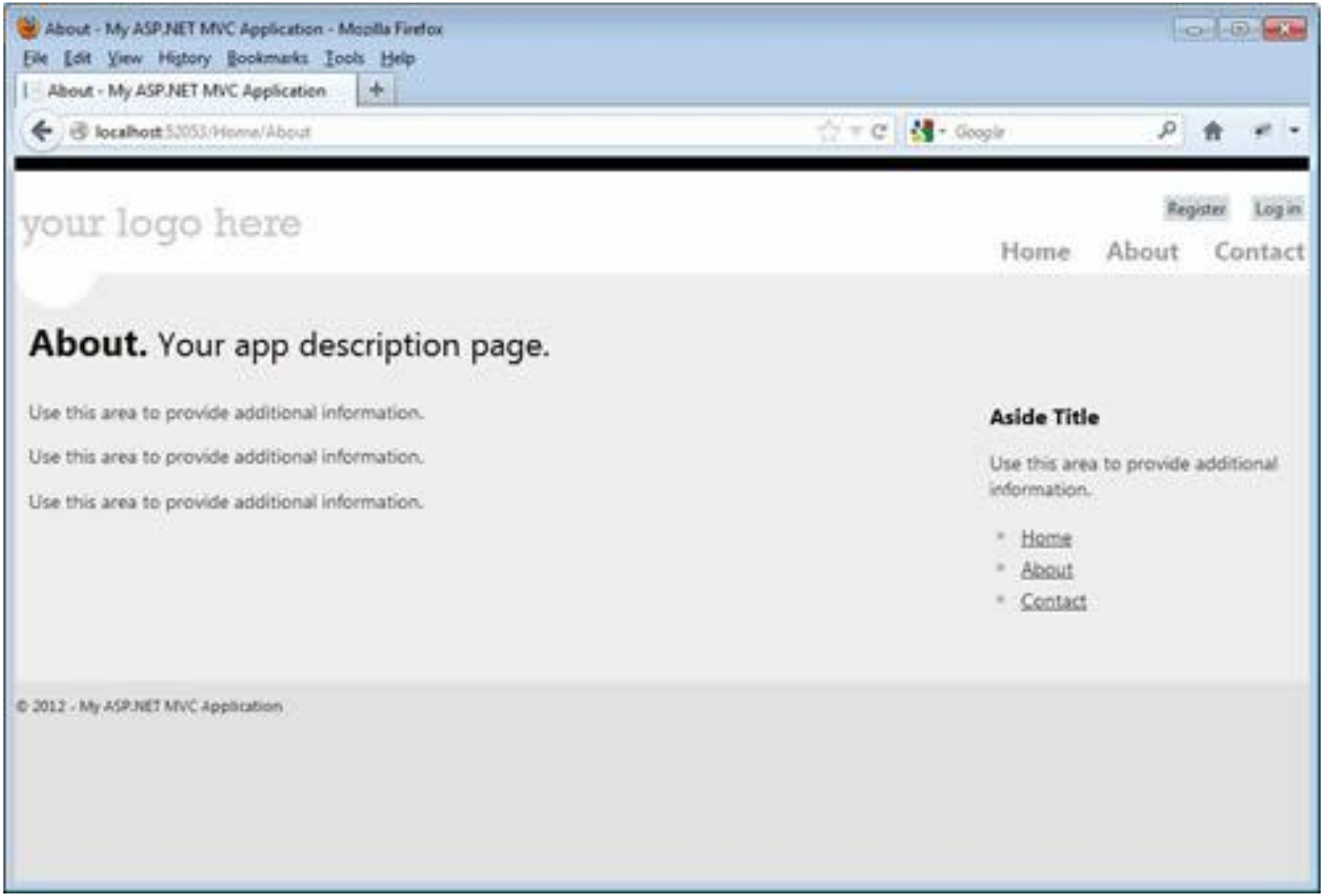
A Default MVC Project is created



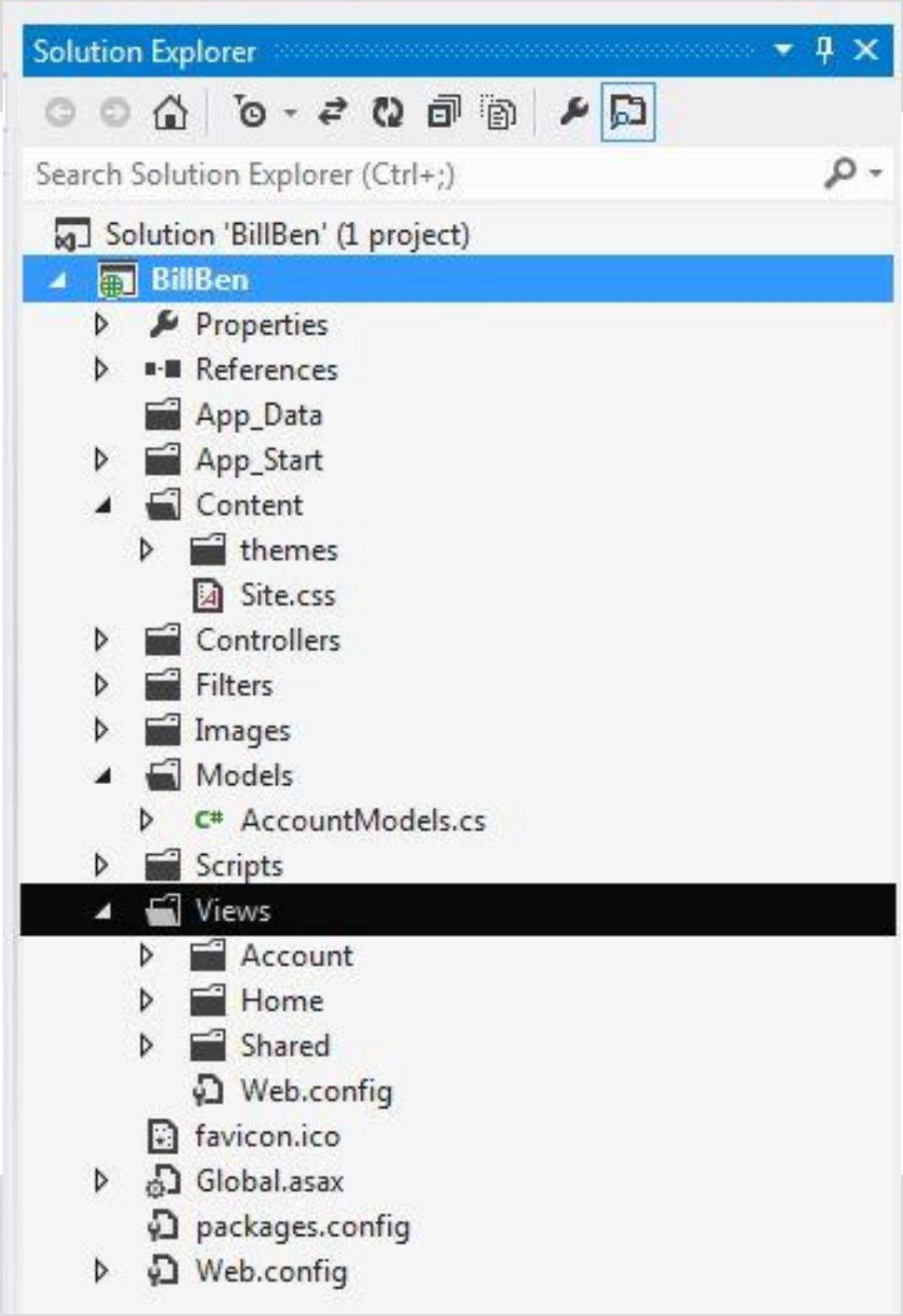
Running the Default MVC Project



Clicking on About Page URL: http://localhost:52053/Home/About



No such file


Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'BillBen' (1 project)
BillBen
Properties
References
App_Data
App_Start
Content
themes
Site.css
Controllers
Filters
Images
Models
AccountModels.cs
Scripts
Views
Account
Home
Shared
Web.config
favicon.ico
Global.asax
packages.config
Web.config

MVC code details

Global.asax (Part 1)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Http;
using System.Web.Mvc;
using System.Web.Optimization;
using System.Web.Routing;
…..
```

Global.asax (Part 2)

```
namespace BillBen {
    public class MvcApplication : System.Web.HttpApplication {
        protected void Application_Start(){
            AreaRegistration.RegisterAllAreas();
            WebApiConfig.Register(GlobalConfiguration.Configuration);
            FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
            RouteConfig.RegisterRoutes(RouteTable.Routes);
            BundleConfig.RegisterBundles(BundleTable.Bundles);
            AuthConfig.RegisterAuth();
        }
    }
}
```

RouteConfig.cs (Part 1)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using System.Web.Routing;
.....
```

RouteConfig.cs (Part 2)

```
namespace BillBen {
public class RouteConfig {
        public static void RegisterRoutes(RouteCollection routes) {
                routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
                routes.MapRoute(
                        name: "Default",
                        url: "{controller}/{action}/{id}",    //URL with parameters
                        defaults: new { controller = "Home", action = "Index",
                        id = UrlParameter.Optional }      //Parameter defaults );
                }
        }
}
```

Default Routes

The default route table consists of one route.

breaks all incoming requests into three segments (a URL segment is anything between forward slashes '/').

The first segment is mapped to a controller name, the second segment is mapped to an action name, and the final segment is mapped to a parameter passed to the action named Id.

For example, consider the following URL:

/Product/Details/3

parsed into three parameters like this:

Controller = Product

Action = Details

Id = 3

Controllers

HomeController.cs

```
....
namespace BillBen.Controllers {
    public class HomeController : Controller {
        public ActionResult Index() {
        ViewBag.Message = "Modify this template to jump-start your ASP.NET
        MVC application."; return View(); }
        public ActionResult About() {
        ViewBag.Message = "BillnBen About Us"; return View(); }
        public ActionResult Contact() {
        ViewBag.Message = "Your contact page."; return View(); }
    }
}
```

Controllers Cont

Developers need to be aware of the fact that any public method is exposed as a controller action.

This means that any public method contained within a controller can be accessed by entering the appropriate URL into a browser.

Views

Views

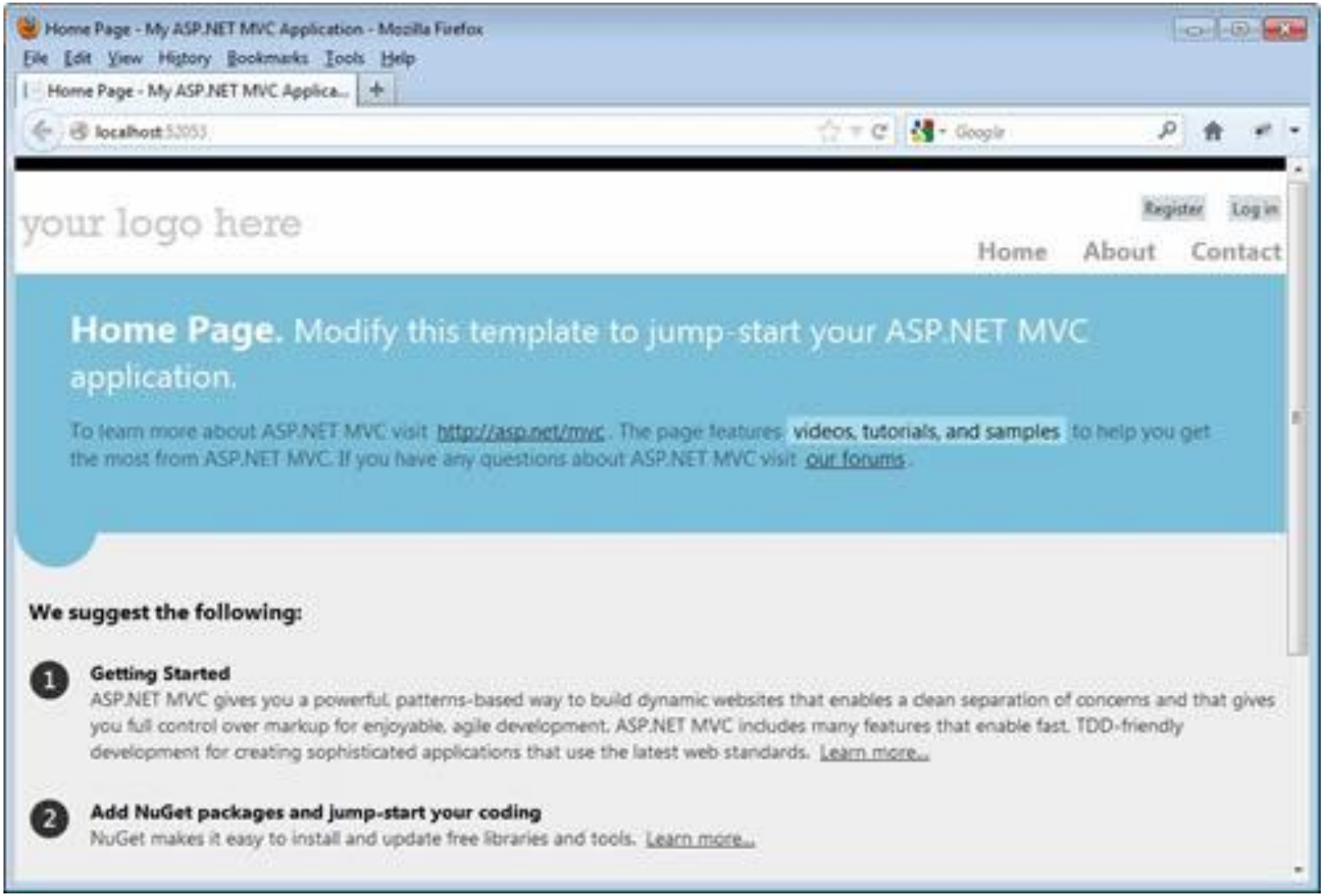
If you want to return a view from a controller action,

- you need to create a sub folder in the Views folder with the same name as the controller.
- In this subfolder you need to create an .aspx file with the same name as the controller action.

Index.aspx

```
<%@ Page Language="C#"
  MasterPageFile="~/Views/Shared/Site.Master"
  Inherits="System.Web.Mvc.ViewPage" %>

<asp:Content ID="indexFeatured"
  ContentPlaceHolderID="FeaturedContent" runat="server">
  <section class="featured">

  <hgroup class="title"> <h1>Home Page.</h1>

  <h2><%: ViewBag.Message %></H2>

  </section>

</asp:Content>

…..
```

Index.aspx page



HTML helpers

Html.ActionLink

ActionLink takes 3 string arguments

1. link text
2. action - the name of the controller method which will be called by this link
3. controller - the name of the controller

So for *<%: Html.ActionLink("Home", "Index", "Home")%>*

will display *Home* as the text

and call the *Index()* method

in the home controller -*HomeController.cs*

Virtual paths for images *Url.Content*

MVC elements are not Server Controls and therefore the ~ character will not work in resolving a virtual path.

Url.Content helper outputs application relative URL's, e.g.

```
<img src="<%: Url.Content("~/Content/Pix/BillnBenHead.gif") %>" />
```

Used to encode user data, to ensure it can be displayed.

<%: ... %> syntax, which automatically encodes it output

Similar to <%= ... %> a shorthand syntax for Response.Write in order to display a value on a web page

BillNBen MVC application

See demo on moodle, Topic 11.2.

MONASH
University

Razor View Engine

History

The Razor view engine was developed in 2010/11

intended to be more compact

easier to learn than the coding required for .aspx files.

No requirement for <%: ... %> coding blocks.

Razor views have a .cshtml file extension.

Master Pages vs Layouts

Razor views use a *layout* to determine their look and feel, rather than a Master Page.

The *default* view is held in *Views/Shared/layout.cshtml*.

layout.cshtml (Part 1)

```
<!DOCTYPE html>
<html lang="en">
 <head> <title>@ViewBag.Title - BillnBen Nursery
@Styles.Render("~/Content/css")
@Scripts.Render("~/bundles/modernizr")
@Scripts.Render("~/bundles/jquery")
@RenderSection("scripts", required: false)
</head> …….
```

layout.cshtml (Part 2)

```
……
<body> <div class="page">
<img src="@Url.Content("~/Images/BillnBenHead.gif")" />
  <div id="menucontainer"> <ul id="menu">
  <li>@Html.ActionLink("Home", "Index", "Home")</li>
  <li>@Html.ActionLink("About Us", "About", "Home")</li>
  <li>@Html.ActionLink("Register", "Register", "Home")</li>
  </ul> </div> <div id="content">
@RenderSection("content", required: false)
@RenderBody()
```

Rendering (Part 1)

```
@Styles.Render("~/Content/css")
```

is rendered as

```
<link href="/Content/site.css" rel="stylesheet"/>
```

The lines

```
@Scripts.Render("~/bundles/modernizr")
@Scripts.Render("~/bundles/jquery")
```

are rendered as

```
<script src="/Scripts/modernizr-2.5.3.js"></script>
<script src="/Scripts/jquery-1.7.1.js"></script>
```

The JavaScript librarys for *jQuery* and *modernizr* are built into the MVC framework.

Rendering (Part 2)

@RenderSection("content", required: false)

renders the *content* section of the View file, not mandatory to have such a named section

@RenderBody()

renders any content not within a named section.

```
@{
   ViewBag.Title = "Home Page";
}
@section content {
<h2>@ViewBag.Message</h2>
}
```

This is where the body of the page goes

Demo on Moodle

See the Razors View demo example on moodle, Topic 11.3

MONASH University

Summary

1 Introduction

2 MVC pattern

3 ASP.NET MVC

4 Razor View Engine

GROUP OF EIGHT AUSTRALIA

MONASH University

What you will do in the Studio

Try topic 11 examples

Read ahead ASP examples from topic 12

Run using Visual Studio 2013

GROUP OF EIGHT AUSTRALIA

MONASH University

Thanks and See you in the Studio!