2.12 Routing



This section will guide you to:

* Create an ASP.NET MVC project to demonstrate various routing techniques

**Development Environment:**

* Windows 10
* Visual Studio 2019 Community Version

This guide has nine subsections, namely:

2.12.1 Creating an ASP.NET MVC project to demonstrate various routing techniques

2.12.2 Creating StudentInfo1.cshmtl View to show student info

2.12.3 Creating TeacherInfo.cshmtl as a placeholder page

2.12.4 Adding routes to Startup.cs

2.12.5 Making changes to HomeController to handle the new Views

2.12.6 Changing Index.cshtml to show links to the Views

2.12.7 Building the project

2.12.8 Publishing and running the project

2.12.9 Pushing the code to your GitHub repositories

**Step 2.12.1:** Creating an ASP.NET MVC project to demonstrate various routing techniques

* Open Visual Studio.
* From the top menu, select **File->New->Project.**
* **In Create A New Project** screen, select **ASP.NET Core Web Application** from the list of available Project types and click on **Next.**
* Enter **Project Name** as Phase3Section2.24 and click on **Create.**
* From the list of project sub-types, choose **Web Application (Model-View-Controller),**  uncheck **Configure for HTTPS** andclick on **Create.**
* This will create the files for an ASP.NET MVC Project.

**Step 2.12.2:** Creating StudentInfo1.cshmtl View to show student info

* In **Solution Explorer,** expand **Views->Home.** Right click **Home** and choose **Add->View.**
* Put **View Name** as StudentInfo1 and click on **Add.**
* Add the following script:

@{

ViewData["Title"] = "StudentInfo1";

}

<**h2**>StudentInfo Using Page Directive Routing</**h2**>

<**div** class="row">

<**div** class="col-sm-7">

Id: @ViewData["id"] <**br** />

Name: @ViewData["name"]<**br** />

</**div**>

</**div**>

**Step 2.12.3:** Adding routes to Startup.cs

* In **Solution Explorer,** double click **Startup.cs.**
* Add the following script:

**using** System;

**using** System.Collections.Generic;

**using** System.Linq;

**using** System.Threading.Tasks;

**using** Microsoft.AspNetCore.Builder;

**using** Microsoft.AspNetCore.Hosting;

**using** Microsoft.AspNetCore.Http;

**using** Microsoft.AspNetCore.Mvc;

**using** Microsoft.Extensions.Configuration;

**using** Microsoft.Extensions.DependencyInjection;

**namespace** Phase3Section2.\_24

{

**public** **class** Startup

{

**public** Startup(IConfiguration configuration)

{

Configuration = configuration;

}

**public** IConfiguration Configuration { **get**; }

// This method gets called by the runtime. Use this method to add services to the container.

**public** **void** ConfigureServices(IServiceCollection services)

{

services.Configure<CookiePolicyOptions>(options =>

{

// This lambda determines whether user consent for non-essential cookies is needed for a given request.

options.CheckConsentNeeded = context => **true**;

options.MinimumSameSitePolicy = SameSiteMode.None;

});

services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version\_2\_1);

}

// This method gets called by the runtime. Use this method to configure the HTTP request pipeline.

**public** **void** Configure(IApplicationBuilder app, IHostingEnvironment env)

{

**if** (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

**else**

{

app.UseExceptionHandler("/Home/Error");

}

app.UseStaticFiles();

app.UseCookiePolicy();

app.UseMvc(routes =>

{

routes.MapRoute(

name: "default",

template: "{controller=Home}/{action=Index}/{id?}");

routes.MapRoute(

name: "studentId",

template: "Student/{id}",

defaults: **new** { controller = "Home", action = "StudentInfo1" });

routes.MapRoute(

name: "studentIdName",

template: "Student/{id}/{name}",

defaults: **new** { controller = "Home", action = "StudentInfo1" });

});

}

}

}

**Step 2.12.5:** Making changes to HomeController to handle the new Views

* In **Solution Explorer,** expand **Controllers** and double click HomeController.
* Enter the following code:

**using** System;

**using** System.Collections.Generic;

**using** System.Diagnostics;

**using** System.Linq;

**using** System.Threading.Tasks;

**using** Microsoft.AspNetCore.Mvc;

**using** Phase3Section2.\_24.Models;

**namespace** Phase3Section2.\_24.Controllers

{

**public** **class** HomeController : Controller

{

**public** IActionResult Index()

{

**return** View();

}

**public** IActionResult About()

{

ViewData["Message"] = "Your application description page.";

**return** View();

}

**public** IActionResult Contact()

{

ViewData["Message"] = "Your contact page.";

**return** View();

}

**public** IActionResult Privacy()

{

**return** View();

}

[ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = **true**)]

**public** IActionResult Error()

{

**return** View(**new** ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });

}

**public** IActionResult StudentInfo1(**string** id, **string** name)

{

ViewData["id"] = id;

ViewData["name"] = name;

**return** View();

}

}

}

**Step 2.12.6:** Changing Index.cshtml to show links to the Views

* In **Solution Explorer,** expand **Views->Home** and double-click index.cshtml.
* Enter the following script:

@{

ViewData["Title"] = "Home Page";

}

<**div** class="container">

<**br** />

<**div** class="row">

<**div** class="col-sm-7">

<**a** href="/Home/StudentInfo1">Student Info Page Using Default Routing</**a**><**br** />

<**a** href="/Home/StudentInfo1?id=198&name=Walter">Student Info Page With Parameters Using Default Routing</**a**><**br** /><**br** />

<**a** href="/Student/190">Student Info Page Using Custom Routing</**a**><**br** />

</**div**>

</**div**>

</**div**>

**Step 2.12.7:** Building the project

* In Visual Studio, from the top menu, choose **Build->Build Solution.**
* If any compile errors are shown, fix them as required.

**Step 2.12.8:** Publishing and running the project

* From the top menu, select **Debug->Start Without Debugging.**
* This will execute the program in the default browser.
* To see the student pages, go to the url : http://localhost:xxxx/students.

**Step 2.12.9:** Pushing the code to your GitHub repositories

Open your command prompt and navigate to the folder where you have created your files.

cd <folder path>

Initialize your repository using the following command:

git init

Add all the files to your git repository using the following command:

git add .

Commit the changes using the following command:

git commit -m “Changes have been committed.”

Push the files to the folder you created initially using the following command:

git push -u origin master