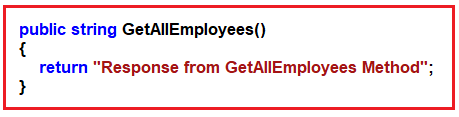
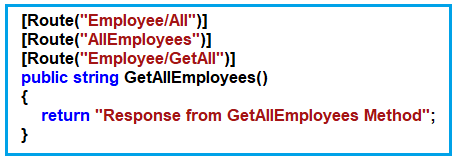
**How to set up Multiple URLs for a Single Resource in ASP.NET Core Web API**

##### **How to access a Single Resource with Multiple URLs in ASP.NET Core Web API:**

Let us understand how to access a single resource with Multiple URLs with an example. Suppose, we have the following resource available in our Employee Controller.



Now we want to access the above resource with three different URLs such as Employee/All, AllEmployees, Employee/GetAll, then how we can do this? If this is our requirement, then we need to decorate the GetAllEmployees action method with three different Route attributes as shown in the below image.



With this, now you can access the GetAllEmployees resource with three different URLs in the ASP.NET Core Web API Application. Let us prove this. First, modify the EmployeeController class as shown in the below code.

**using** *Microsoft.AspNetCore.Mvc;*

**namespace** *RoutingInASPNETCoreWebAPI.Controllers*

**{**

**[**ApiController**]**

**public** **class** EmployeeController : ControllerBase

**{**

**[**Route**(**"Employee/All"**)]**

**[**Route**(**"AllEmployees"**)]**

**[**Route**(**"Employee/GetAll"**)]**

**public** string GetAllEmployees**()**

**{**

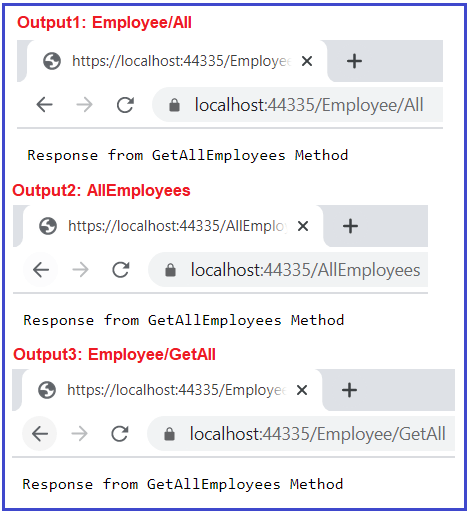
**return** "Response from GetAllEmployees Method";

**}**

**}**

**}**

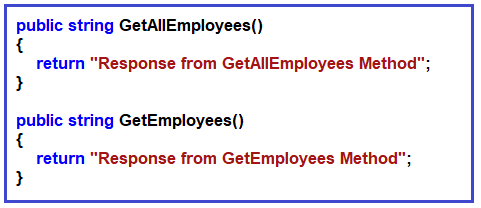
Now run the application and access the above resource using the different URLs as shown in the below image.



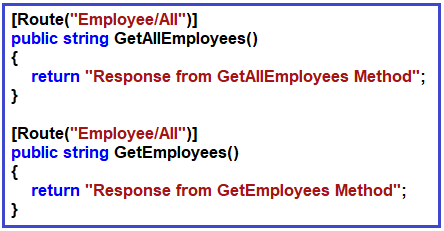
As you can see in the above image, we are now able to access the same resource using three different URLs in ASP.NET Core Web API Application. As you can see each URL is unique. So, as long as the URLs are unique, you can access a particular resource using different URLs.

##### **What happens if we use the same URL for multiple resources?**

This is not accepted in ASP.NET Core Web API. Let us prove this with an example. Let say we have the following two resources.



Now, let us try to access both the above resources using the URL Employee/All, so what we will do is, we will decorate both the resources with the same Route Attribute as shown in the below image.



Let us modify the EmployeeController class as shown below.

**using** *Microsoft.AspNetCore.Mvc;*

**namespace** *RoutingInASPNETCoreWebAPI.Controllers*

**{**

**[**ApiController**]**

**public** **class** EmployeeController : ControllerBase

**{**

**[**Route**(**"Employee/All"**)]**

**public** string GetAllEmployees**()**

**{**

**return** "Response from GetAllEmployees Method";

**}**

**[**Route**(**"Employee/All"**)]**

**public** string GetEmployees**()**

**{**

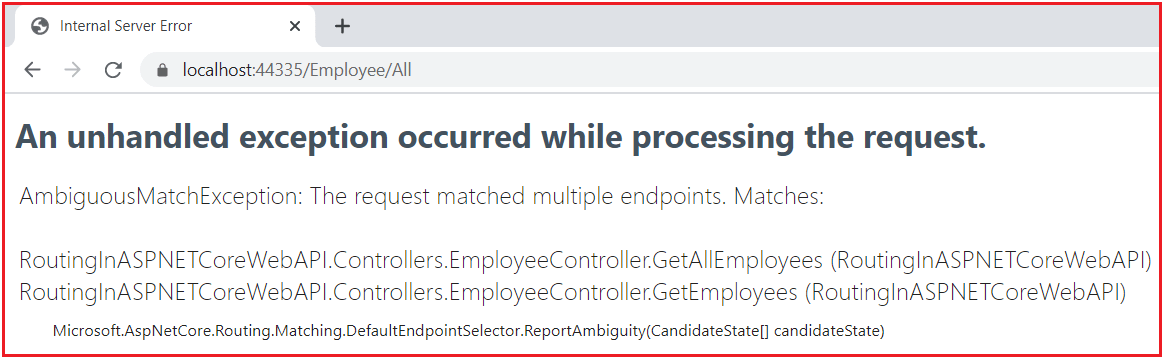
**return** "Response from GetEmployees Method";

**}**

**}**

**}**

With the above changes in place, now run the application and navigate to **Employee/All** URL as shown in the below image.



As you can see in the above image, it is throwing an Internal Server Error saying AmbiguousMatchException: The request matched multiple endpoints. This is because the application finds to resource for the same URL and gets confused about who is going to handle the request and hence throwing the AmbiguousMatchException.

**Note:** So, the point that you need to remember is, each resource must have a unique URL, and also it is possible that a resource can be accessed using multiple URLs as long as all the URLs are unique. But it is not possible to access two or more different resources using a single URL in ASP.NET Core Web API Application.

Now change the Route Attribute of both the resources as shown in the below code to give different URLs.

**using** *Microsoft.AspNetCore.Mvc;*

**namespace** *RoutingInASPNETCoreWebAPI.Controllers*

**{**

**[**ApiController**]**

**public** **class** EmployeeController : ControllerBase

**{**

**[**Route**(**"Employee/GetAll"**)]**

**[**Route**(**"EmployeeAll"**)]**

**public** string GetAllEmployees**()**

**{**

**return** "Response from GetAllEmployees Method";

**}**

**[**Route**(**"Employee/All"**)]**

**public** string GetEmployees**()**

**{**

**return** "Response from GetEmployees Method";

**}**

**}**

**}**

With the above changes in place, now you can access the **GetAllEmployees** resource using two URLs i.e. **Employee/GetAll** and **EmployeeAll**. On the other hand, you can access the **GetEmployees** resource using the URL **Employee/All** as shown in the below image.

