Routing Lab

Perform these labs on your own computer using Visual Studio 2022 to ensure you understand the lessons presented in the corresponding videos and lectures.

Lab 1: Add Route Attribute

Open the **CustomerController.cs** file and add two [Route] attributes (shown in **bold** below) to the Get() method.

```
[HttpGet]
[Route("")]
[Route("GetAllCustomers")]
[ProducesResponseType(StatusCodes.Status200OK)]
[ProducesResponseType(StatusCodes.Status404NotFound)]
public ActionResult<IEnumerable<Customer>> Get()
{
  // REST OF THE CODE HERE
}
```

Try it Out

Run the application and you should now see two routes (shown in Figure 1 below).

Both routes call the same Get() method.

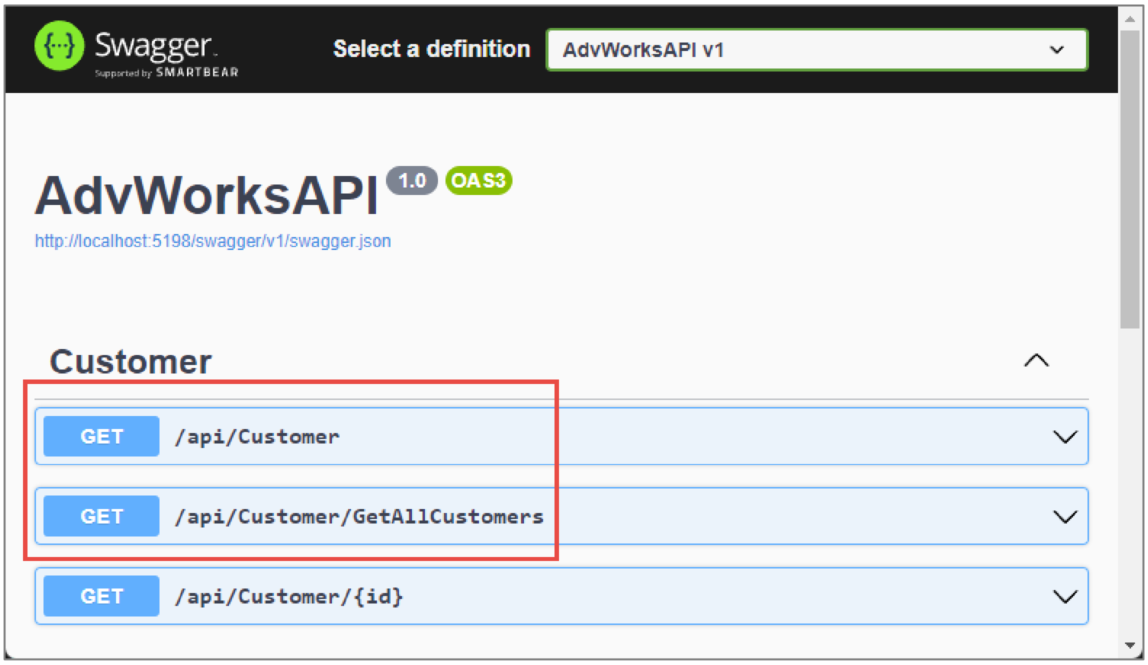

Figure 1: Two routes can call the same Web API method

After you have viewed and run these APIs to ensure they work, **remove** both the **[Route(…)]** attributes on the Get() method as you don't really need to do this.

Lab 2: Search for Title

Open the **CustomerController.cs** file and add a new method to search for all customers that **exactly match** a specific title.

```
[HttpGet]
[Route("SearchByTitle")]
[ProducesResponseType(StatusCodes.Status200OK)]
[ProducesResponseType(StatusCodes.Status404NotFound)]
public ActionResult<IEnumerable<Customer>>
SearchByTitle(string title)
{
  ActionResult<IEnumerable<Customer>> ret;
  List<Customer> list;

  // Get all data
  list = new CustomerRepository().Get()
    .Where(row => row.Title == title)
    .OrderBy(row => row.LastName).ToList();

  if (list != null && list.Count > 0) {
    ret = StatusCode(StatusCodes.Status200OK, list);
  }
  else {
    ret = StatusCode(StatusCodes.Status404NotFound,
      $"No Customers found matching the title
'{title}'.");
  }

  return ret;
}
```

Try it Out

Run the application and click on the **GET /api/Customer/SearchByTitle** button.

Enter the value **Mr.** (make sure you include the period at the end) into the **title** field.

Click the **Execute** button to view the results.

Notice that Swagger shows you how to make this request via a URL query.

```
http://localhost/api/Customer/SearchByTitle?title=Mr.
```

Lab 3: Search for FirstName and LastName

Open the **CustomerController.cs** file and add a new method to search by the first name and the last name.

```
[HttpGet]
[Route("SearchByFirstLast/First/{first}/Last/{last}")]
[ProducesResponseType(StatusCodes.Status200OK)]
[ProducesResponseType(StatusCodes.Status404NotFound)]
public ActionResult<IEnumerable<Customer>>
SearchByFirstLast(string first, string last)
{
  ActionResult<IEnumerable<Customer>> ret;
  List<Customer> list;

  // Get all data
  list = new CustomerRepository().Get()
    .Where(row => row.FirstName.Contains(first,
StringComparison.InvariantCultureIgnoreCase) &&
           row.LastName.Contains(last,
StringComparison.InvariantCultureIgnoreCase))
    .OrderBy(row => row.LastName).ToList();

  if (list != null && list.Count > 0) {
    ret = StatusCode(StatusCodes.Status200OK, list);
  }
  else {
    ret = StatusCode(StatusCodes.Status404NotFound,
      "No Customers found matching the criteria passed
in.");
  }

  return ret;
}
```

Try it Out

Run the application

Click on the **GET /api/Customer/SearchByFirstLast/…** button.

Enter 'A' for the first name.

Enter 'B' for the last name.

Click the **Execute** button to view the results.

Notice that Swagger shows you how to make this request via a URL query.

```
http://localhost:5198/api/Customer/SearchByFirstLast/Fir
st/A/Last/B
```