#### Questions for this assignment

What is Routing?

How Routing works in ASP.NET Core?

What are the important route constraints?

What is the purpose of the wwwroot folder?

How do you change the path of wwwroot folder?

What is Routing?

Routing is functionality that map incoming request to the route handler. The route can have route parameters to receive values from the URL. Using the route, routing can find a route handler based on the URL. All the routes are registered when the application is started. There are two types of routing supported by ASP.NET Core

1. The conventional routing
2. Attribute routing

The Routing uses routes to map incoming requests with the route handler and Generates URL that is used in response. Mostly, the application has a single collection of routes and this collection is used for the process of the request. The RouteAsync method is used to map incoming requests (that match the URL) with available in route collection.

How Routing works in ASP.NET Core?

Routing is used to handle incoming HTTP requests for the app. Routing finds matching executable endpoint for incoming requests. These endpoints are registered when app starts. Matching process use values from incoming request url to process the requests. You can configure the routing in middleware pipeline of configure method in startup class.

app.UseRouting(); // It adds route matching to middleware pipeline

// It adds endpoints execution to middleware pipeline

app.UseEndpoints(endpoints =>

{

endpoints.MapGet("/", async context =>

{

await context.Response.WriteAsync("Hello World!");

});

});

What are the important route constraints?

In ASP.NET Core, route constraints are used to restrict the values that can be matched for route parameters. They are an essential part of defining more specific and controlled routes. Here are some important route constraints you can use:

1. **int**: Constrains the parameter to be an integer.
2. **long**: Constrains the parameter to be a long integer.
3. **bool**: Constrains the parameter to be a Boolean value, i.e., "true" or "false".
4. **double**: Constrains the parameter to be a double-precision floating-point number.
5. **float**: Constrains the parameter to be a floating-point number.
6. **guid**: Constrains the parameter to be a GUID (Globally Unique Identifier).
7. **datetime**: Constrains the parameter to be a valid date and time value.
8. **alpha**: Constrains the parameter to contain only letters (no digits or special characters).
9. **regex**: Allows you to define a custom constraint using a regular expression pattern.
10. **length**: Constrains the parameter to have a specific length. For example, **{id:length(5)}** will only match when the **id** parameter has a length of 5 characters.
11. **min** and **max**: Allows you to specify minimum and maximum values for numeric parameters. For example, **{age:min(18)}** will only match if the **age** parameter is 18 or greater.
12. **range**: Similar to min and max, but allows you to specify a range of values. For example, **{year:range(1900, 2023)}** will only match if the **year** parameter is between 1900 and 2023.
13. **required**: Indicates that the parameter is required and must be present in the URL for the route to match.
14. **nonempty**: Ensures that the parameter is not empty (not null, empty string, or whitespace).
15. **maxlength** and **minlength**: Restricts the length of a string parameter. For example, **{username:maxlength(20)}** will only match if the **username** parameter has a length of 20 characters or less.

What is the purpose of the wwwroot folder?

The **wwwroot** folder is a special folder in an ASP.NET Core web application that serves as the web root. Its purpose is to store static files, such as HTML, CSS, JavaScript, images, and other client-side assets that need to be directly accessible by the web browser.

When a web application receives a request, the web server looks for the requested resource within the **wwwroot** folder. If the resource is found in this folder, the web server serves it directly to the client without involving the ASP.NET Core middleware pipeline.

How do you change the path of wwwroot folder?

We need to set path of the wwwroot folder in the WebRootPath propertyof the WebApplicationOptions class.

var builder = WebApplication.CreateBuilder(new WebApplicationOptions() {

WebRootPath = "foldername"

});