Arithmetic Operations Web Application – Razor Pages

Objective:

Create a web application that processes various arithmetic operations based on parameters received either through query strings or route parameters. This application will be developed using ASP.NET Core with Razor Pages. We will utilize optionally Postman to send requests to test the application.

Functional Requirements:

1. **Arithmetic Operations:** The application should be able to perform the following operations based on the input parameters:
    - Sum (SUM) and Product (MUL)
    - Minimum (Min) and Maximum (MAX)
    - Least Common Multiple (LCM) and Greatest Common Divisor (GCD)
2. **Razor Pages and Handlers:**
    - **SumMul Razor Page:** Handles **SUM** and **MUL** operations. This page should accept three route parameters via Get Request. It should display the summation and the multiplication of the received values.
    - **MinMax Razor Page:** Handles **Min** and **MAX** operations. This page should accept three route parameters via Get Request. It should display the minimum and the maximum of the received values.
    - **LcdGcd Razor Page:** Handles **LCM** and **GCD** operations. This page should accept two route parameters via Get Request. It should display the Least Common Multiple and the Greatest Common Divisor of the received values.
3. **Service Classes:** All logic for performing the arithmetic operations should be encapsulated within service classes. These services should be injected into the Razor Pages using Dependency Injection (DI).
4. **Handling Default Values:** If **fewer** than the required number of parameters are passed in the route parameters for any operation except (LCM and GCD), the page handler should assign default values.
5. **Route Constraints:** Implement constraints on the route parameters to validate the input values. Test these constraints by trying to send queries that do not match the expected patterns.

6. **Layout Customization:** Modify the application layout to include a customized footer and header. Remove any unrelated links and menus to streamline the user interface. (_Layout.cshtml).

7. **Testing Non-Matching Routes:** Attempt to send a request where no route template matches, to ensure the application handles it gracefully.

Technical Notes:

- To extract route parameters from the request in a Razor Page, you might use the **RouteData.Values["parameterName"]** method, but we should be based on the binding concepts in this exercice.