Assignment 1: Building a .NET Core Application with Middleware

Objective: Develop a basic .NET Core application that uses middleware to handle HTTP requests and serves static files. This assignment will help you understand the core features of .NET Core, middleware configuration, and serving static content.

Problem Statement

Create a .NET Core web application that demonstrates the use of middleware to handle different types of HTTP requests. The application should also serve static files (HTML, CSS, JavaScript) and apply basic security measures for these static files.

User Stories and Expectations

User Story 1: Create a .NET Core Application

- **As a developer**, you need to create a new .NET Core web application using the .NET Core CLI or Visual Studio.
- **Acceptance Criteria**:
    - The application should be a basic web project with a startup class configured to use middleware.
    - Implement middleware to log incoming requests and responses.

User Story 2: Configure Middleware Components

- **As a developer**, you need to configure middleware components in the `Startup.cs` file to handle requests and responses.
- **Acceptance Criteria**:
    - Add middleware to log request details and response status codes.
    - Configure middleware to handle errors by returning a custom error page for unhandled exceptions.

User Story 3: Serve Static Files

- **As a developer**, you need to configure the application to serve static files from a `wwwroot` folder.
- **Acceptance Criteria**:
    - Create a `wwwroot` folder and add an HTML file, a CSS file, and a JavaScript file.
    - Ensure that these static files are served correctly when accessed via the browser.

User Story 4: Implement Basic Security

- **As a developer**, you need to apply basic security considerations to the serving of static files.
- **Acceptance Criteria**:
    - Configure middleware to enforce HTTPS for serving static files.
    - Implement a content security policy to mitigate risks such as Cross-Site Scripting (XSS).

Assignment Outcome

By completing this assignment, you will gain hands-on experience with .NET Core application setup, middleware configuration, and static file handling. You will also understand basic security practices related to static content.

Best Practices for Submission:

1. **Code Quality**: Ensure your code is clean and follows best practices for middleware and static file serving.
2. **Documentation**: Comment your code to explain the purpose of each middleware component and the security measures implemented.
3. **Testing**: Verify that the middleware is logging requests and responses correctly and that static files are served properly.

Assignment 2: Implementing Razor Pages with Page Models

Objective: Create a Razor Pages application to demonstrate understanding of Razor Pages architecture, syntax, and PageModel class. This assignment will involve creating and configuring Razor Pages and utilizing Razor syntax for server-side code.

Problem Statement

Build a Razor Pages application that includes several pages demonstrating different features of Razor syntax and the PageModel class. The application should allow users to interact with the pages and display data dynamically.

User Stories and Expectations

User Story 1: Create and Configure Razor Pages

- **As a developer**, you need to create a Razor Pages application and configure it to use multiple pages.
- **Acceptance Criteria**:
    - Create a Razor Pages application with at least two pages: one for displaying a list of items and another for adding a new item.

- Ensure that Razor Pages are correctly configured with a folder structure and naming conventions.

User Story 2: Use Razor Syntax

- **As a developer**, you need to use Razor syntax to mix HTML with server-side code.
- **Acceptance Criteria**:
  - Implement Razor syntax to dynamically generate content on the pages. For example, display a list of items using a `foreach` loop and handle form submissions using Razor syntax.

User Story 3: Implement the PageModel Class

- **As a developer**, you need to create and utilize the PageModel class for handling page logic and data binding.
- **Acceptance Criteria**:
  - Create a `PageModel` class for each page that handles data binding, form submissions, and page interactions.
  - Use property binding to manage data and handle user input.

User Story 4: Display Dynamic Data

- **As a developer**, you need to ensure that the Razor Pages display dynamic data based on user input or other interactions.
- **Acceptance Criteria**:
  - Ensure that the list of items displayed on the page updates dynamically based on data changes.
  - Implement form handling to allow users to add new items and see the updates reflected on the page.

Assignment Outcome

By completing this assignment, you will gain practical experience with Razor Pages, including creating pages, using Razor syntax, and implementing the PageModel class. You will also learn how to handle dynamic data and user interactions within Razor Pages.

Best Practices for Submission:

1. **Code Organization**: Ensure that Razor Pages are well-organized according to the folder structure and naming conventions.
2. **Documentation**: Include comments and explanations in your code to clarify the use of Razor syntax and PageModel logic.
3. **Testing**: Test the pages to ensure they correctly handle user input, display dynamic data, and adhere to the expected functionality.

Daily Coding Assignment Wipro NGA .NET Cohort