Project Report: ASP.NET Core MVC Book Store Application

WEB PROGRAMMING HOMEWORK

B211202036 OZAN GÜL

B211202006 SELİN AKKAŞ

B201202046 BEKİRCAN KÜÇÜKAKIN

1. Introduction:

The ASP.NET Core MVC Book Store application was developed as a semester project, adhering to the MVC architecture. This report outlines the key aspects of the project, including the utilization of Model-View-Controller, Routing, Layout, ViewBag/ViewData, Tag Helpers, Forms, Form Validations, Partial Views, and Entity Framework.

2. Architecture Overview:

The application follows the Model-View-Controller architectural pattern, separating concerns into models, views, and controllers. This approach promotes code organization, reusability, and maintainability.

3. Controllers:

The project comprises a minimum of two controllers:

BookManagementController: Manages operations related to books such as listing, creating, editing, and deleting.

CategoryController: Handles category-related actions, including listing categories and displaying books within a specific category.

4. Views:

Each controller has a minimum of two views:

BookManagementController Views:

Index.cshtml: Displays a list of books.

Create.cshtml: Allows the user to add a new book.

Edit.cshtml: Permits the user to modify book details.

Delete.cshtml: Provides confirmation for book deletion.

CategoryController Views:

Index.cshtml: Lists all available book categories.

ShowBooks.cshtml: Displays books belonging to a specific category.

5. Frontend Development:

The frontend is developed using HTML and CSS, ensuring a visually appealing and responsive user interface.

Utilized Tag Helpers for cleaner and more readable HTML markup.

6. Entity Framework:

Entity Framework is employed for data access, providing an Object-Relational Mapping (ORM) approach.

The Book and BookCategory models are designed to represent the database schema.

7. Routing and Layout:

Custom routes are defined to ensure user-friendly URLs.

A consistent layout is established across views for a seamless user experience.

8. Forms and Form Validations:

Forms are implemented for user input in creating and editing books.

Form validations are in place to ensure data integrity and provide a smooth user experience.

9. Partial Views:

Partial views are utilized for code reusability and to enhance modularity.

Common elements like headers and footers are placed in partial views.

10. ViewData and ViewBag:

ViewData and ViewBag are used to pass data from controllers to views.

These mechanisms facilitate communication between the controller and the view.

11. Conclusion:

The ASP.NET Core MVC Book Store project successfully incorporates the specified topics, demonstrating proficiency in MVC architecture, routing, layout design, form handling, and data access through Entity Framework. The use of HTML/CSS contributes to an engaging frontend, ensuring a holistic and well-rounded web application.

work parts of group members

**1. Selin,Bekir,Ozan:**

Responsibilities:

Implement HTML and CSS for the user interface.

Ensure the frontend is visually appealing and responsive.

Utilize Tag Helpers for cleaner HTML markup.

Collaborate with the team to integrate frontend components with backend functionality.

Test and optimize the user interface for a seamless experience.

**2. Ozan,Bekir:**

Responsibilities:

Develop controllers and actions following the MVC architecture.

Implement routing for user-friendly URLs.

Design and implement models for Book and BookCategory using Entity Framework.

Handle CRUD operations for books and categories.

Implement form handling and validations.

Ensure proper communication between the frontend and backend using ViewData and ViewBag.

Collaborate with the team to integrate frontend and backend components.

**3. Bekir,Ozan Selin:**

Responsibilities:

Design and create the database schema for the Book Store application.

Implement Entity Framework for data access and ORM.

Optimize database queries for efficient performance.

Ensure data integrity and consistency.

Collaborate with the backend developer to establish a smooth connection between the application and the database.

Handle database migrations and updates.

**4. Collaborative Tasks:**

All Team Members:

Conduct regular team meetings to discuss progress, challenges, and updates.

Collaborate on the integration of frontend and backend components.

Perform testing, debugging, and optimization as a team.

Document the project for future reference and maintenance.

This role distribution aims to leverage each team member's skills and expertise, ensuring a collaborative and efficient development process. Regular communication and coordination among team members are crucial for the success of the project.