Explanation of BestStoreMVC File Structure

Tuesday, March 4, 2025 5:16 PM

1 wwwroot/ - Static Files

This folder contains **public static files** that users can access directly, like:

- css/ → Stores CSS files for styling.
- js/ → Stores JavaScript files for client-side scripts.
- lib/ → Contains external libraries (Bootstrap, jQuery, etc.).
- products/ → Stores uploaded product images.
- favicon.ico → The website's favicon.

▨ **When a user uploads an image, it is saved inside the wwwroot/products/ folder.**

2 Controllers/ - Handle User Requests

- **HomeController.cs** → Handles requests for the home page.
- **ProductsController.cs** → Manages product-related operations, such as:
    - Index() → Fetches all products from the database and passes them to the view.
    - Create(ProductDto productDto) → Saves a new product to the database.
    - Edit(int id, ProductDto productDto) → Updates an existing product.
    - DeleteConfirmed(int id) → Deletes a product.

▨ **Controllers act as the "brain" of the application, fetching and sending data between Views and Models.**

3 Migrations/ - Database Schema Management

- 20250302201416_InitialCreate.cs → A migration file that defines how the database schema is created/modified.
- ApplicationDbContextModelSnapshot.cs → Represents the current database schema snapshot.

▨ **This folder is used by Entity Framework Core to manage database updates and structure.**

4 Models/ - Data Representation (Database Tables)

- **Product.cs**

```
public class Product
{
   public int Id { get; set; }
   public string Name { get; set; }
   public string Brand { get; set; }
   public string Category { get; set; }
   public decimal Price { get; set; }
   public string Description { get; set; }
   public string ImageFileName { get; set; }
   public DateTime CreatedAt { get; set; }
}
```

▨ **Represents the Products table in the database.**

- **ProductDto.cs**

```
public class ProductDto
{
   public string Name { get; set; }
   public string Brand { get; set; }
   public string Category { get; set; }
   public decimal Price { get; set; }
   public string Description { get; set; }
   public IFormFile? ImageFile { get; set; }
}
```

▨ **ProductDto (Data Transfer Object) is used for handling form data but does not directly map to the database.**

  - It is used when receiving data from the user in Create and Edit actions.

5 Services/ - Application Database Context

- **ApplicationDbContext.cs**

```
csharp
CopyEdit
public class ApplicationDbContext : DbContext
{
   public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) :
base(options) { }
   public DbSet<Product> Products { get; set; }
}
```

▨ **Manages the connection between the application and the database using Entity Framework Core.**

  - Defines a Products table using DbSet<Product>.

6 Views/ - User Interface (Frontend HTML Templates)

◇ Views/Home/

- Index.cshtml → Homepage template.
- Privacy.cshtml → Privacy policy page.

◇ Views/Products/

- Create.cshtml → **Form to create a new product.**
- Edit.cshtml → **Form to edit a product.**
- Index.cshtml → **Displays all products in a table.**

◇ Views/Shared/

- _Layout.cshtml → **Main layout template (header, footer, etc.).**
- _ValidationScriptsPartial.cshtml → **Includes client-side validation scripts.**
- Error.cshtml → **Error handling page.**

7 Configuration Files

- appsettings.json → **Stores database connection strings and app settings.**
- Program.cs → **Entry point of the application, configures services like MVC and Entity Framework.**

🔄 How Data Flows in Your Application

1️⃣ **User requests a page** (e.g., localhost:5000/Products).
2️⃣ **Controller (ProductsController.cs) handles the request** and fetches data from the database using ApplicationDbContext.
3️⃣ **Data is passed to a View (Index.cshtml)** where products are displayed.
4️⃣ **If a user submits a form** (e.g., Create/Edit Product), the form data is captured using ProductDto.
5️⃣ **Controller processes the data, saves it in the database**, and redirects to Index.cshtml.
6️⃣ **If an image is uploaded**, it is stored in wwwroot/products/.

✅ Summary

- **Controllers handle HTTP requests and interact with the database.**
- **Models represent database tables and business logic.**
- **Views display HTML pages using Razor syntax.**
- **The database is managed using Entity Framework Core with migrations.**
- **Static files like CSS, JS, and images are stored in wwwroot/.**
- **Images uploaded by users are stored in wwwroot/products/.**

💡 **Your project follows a clean and structured MVC architecture. Let me know if you need improvements or explanations!** 🚀 🔥