Team Name: CAI2_SWD5_S10_Team1

Project: TechXpress E-commerce Platform Documentation

Objective:

The goal of the TechXpress e-commerce platform is to build a scalable and maintainable web application for selling electronics. The project leverages design patterns like NTier Architecture, Repository Pattern, and Unit of Work to ensure efficient development and separation of concerns.

Description:

TechXpress is a full-featured e-commerce platform that allows users to browse electronics (laptops, mobiles, cameras), add them to a shopping cart, and complete purchases using integrated payment gateways. The platform includes an admin panel for managing products, categories, and orders.

Features:

- **User Authentication:** Secure login and registration with role-based access control
- **Product Catalog:** Browse and search for products with filtering options
- **Shopping Cart:** Add, remove, and update items before checkout
- **Payment Gateway:** Secure payment processing using Stripe
- **Order Management:** Order tracking, history, and status updates
- **Admin Panel:** Manage products, categories, and orders with ease
- **Responsive UI:** Optimized design for desktop and mobile devices

Target Audience:

- **Customers:** Browse products, make purchases, and track orders
- **Admins:** Manage product listings, orders, and user roles
- **Vendors:** Potential future integration to allow multiple sellers

Technologies Used:

- **Backend:** ASP.NET Core MVC, Entity Framework Core
- **Frontend:** JQuery, DataTables, Toaster JS, Bootstrap
- **Authentication:** ASP.NET Identity
- **Payment Integration:** Stripe
- **Deployment:** Microsoft Azure

Prerequisites:

- .NET SDK (latest version)
- SQL Server and SQL Server Management Studio (SSMS)
- Visual Studio
- Git for version control
- Postman (for API testing)

Project Structure:

- TechXpress.Web (Presentation Layer) → Contains MVC views, controllers, and static assets.
- TechXpress.Services (Business Logic Layer) → Contains service classes for business operations.
- TechXpress.Data (Data Access Layer) → Implements repositories, entity models, and database context.
- TechXpress.Tests (Testing) → Contains unit and integration tests.
- TechXpress.API (Optional) → REST API endpoints for the frontend.

Testing Strategy:

- **Unit Testing:**
    - Test business logic functions in the service layer.
    - Validate repository operations (CRUD) using mock data.
- **Integration Testing:**
    - Test API endpoints for correct request/response handling.
    - Ensure database transactions work as expected.
- **User Acceptance Testing:**
    - Verify UI components for usability and performance.
    - Conduct end-to-end order processing tests.

Design Patterns:

- **Ntier Architecture:** Separation of the application into Presentation (UI), Business Logic (Service Layer), and Data Access (Repository/Entity Framework) layers.
- **Repository Pattern:** Abstraction for database interactions, ensuring clean API for querying and persisting data.
- **Unit of Work Pattern:** Ensures that multiple related changes are grouped together in a single transaction, reducing data inconsistencies.
- **Dependency Injection:** Inject services (repositories, business logic) into controllers to promote testability and loose coupling.

Database Design:

- **Tables:** Products, Categories, Orders, OrderDetails, Users
- **ORM:** Entity Framework Core with DbContext as Unit of Work
- **Repositories:** Generic repository for CRUD operations, specific repositories for Product and Category management