**Technical Documentation: EasyGames**

**1. System Overview**

EasyGames is an ASP.NET Core MVC web application, which is designed as a student project to replicate a real-world store for an online shop for books, games, and toys.

The overall system has two roles which affect your view for the website:

* **User:** As a User, you can see the Home, Catalog, Cart pages. You can browse stock, add **them to cart and checkout.**
* **Admin:** As an Admin, you can manage stocks, manage users and subscribers, monitor for any low-stock items and action accordingly.

**Note:** Without any of these roles, User can still browse the Home and Catalog page, but it requires login/register to see further pages.

The goal of this system is to provide a simple e-commerce workflow for admin to manage business operations.

**2. Models**

Models are the logic and data of the application, holding together the data, storage and processing of the application. Below are the models in the application EasyGames:

* **AdminDashboard:** It holds the summary stats for total users, categories, stock items, subscribers, and a section for monitoring low-stock items with alerts for the admin dashboard.
* **CartItem:** It represents a single item inside a shopping cart, including its properties like quantity, price, and stock limits.
* **Category:** It defines the product categories such as Books, Toys, and Games available in the store. This is used for organizing stock items.
* **ErrorViewModel:** It displays error details such as request ID if something goes wrong in the app.
* **HomeViewModel:** It provides data for the home page for section like “What’s New” and “Trending” products.
* **StockItem:** It represents an individual product with details of its properties such as title, description, price, quantity, image, and category.
* **Subscriber:** It stores email addresses of the users who agreed to subscribe for updates or offers for the store.
* **Users:** It stores the user details, email, password, roles and extends the ASP.NET Identity user with additional details like full name.

**3. Controllers**

* + **AccountController:** Handles the login, register, verify email and password change.
  + **CartController:** Manages add, update, remove and checkout for the shopping cart.
  + **CatalogController:** Displays products by category and search query.
  + **HomeController:** Handles homepage, subscription form, and admin dashboard.
  + **StockItemsController:** Provides CRUD (Create, Read, Update, Delete) for stock items (admin only).
  + **SubscribersController:** Allows admin to view subscriber list.
  + **UserController:** Allows admin to view and delete users.

**4. Services**

* **Cart.cs:**
  + Stores cart in session.
  + Ensures the quantities added by users do not exceed available stock.
  + Performs stock checks and reduces stock when user checks out.
* **SeedService.cs:**
  + Creates roles (Admin and User).
  + Seeds a default admin account.
  + Ensures base categories (Books, Games, Toys) exist.
  + Adds sample demo stock if stocks are empty.

**5. Data Flow**

1. **User workflow:**
   * Register/Login → Browse Catalog → Add to Cart → Update quantities/Delete anu item (if needed) → Checkout.
   * Checkout decreases stock in database.
2. **Admin workflow:**
   * Login → Dashboard → View user count, stock count, subscriber count, and low-stock alerts.
   * Manage stock with Low Stock Alert on Admin Dashboard.
   * Add new items under the Categories from Admin Dashboard.

**6. Database Design**

The database uses SQL Server LocalDB with Entity Framework Core. This is opened and managed in the SQL Server locally. The database is named NewUserRoles.

Main tables:

* **AspNetUsers / AspNetRoles:** Identity tables for authentication.
* **Categories:** Stores category names.
* **StockItems:** Stores product details.
* **Subscribers:** Stores newsletter emails.

A screenshot of a computer

AI-generated content may be incorrect.

**7. Special Features**

* **Low Stock Alerts:** Admin dashboard highlights products with stock ≤ 5.

A screenshot of a computer

AI-generated content may be incorrect.

* **Subscription Management:** User can subscribe via homepage, and admin can view subscribers in the Admin Dashboard.

A screenshot of a video game

AI-generated content may be incorrect.

A green and white rectangle with a white background

AI-generated content may be incorrect.

* **Cart Quantity Control:** Cart automatically limits to available stock, when users check out with max quantity.
* **Secure Authentication:** ASP.NET Identity handles login/logout, password hashing, and role management. Passwords are setup with validation on password constraints.