Software Design Specification (SDS)

Project Name: “Chapter Find”– Online Bookstore
Prepared By: Ahmed Amr, Mohamed Wael, Mohamed Eslam, Malek Mohamed, Ahmed Wael
Date:09-Nov-24

---

1. Introduction

1.1 Purpose

The purpose of this document is to describe the design, architecture, and technical specifications of the [“Chapter Find”– Online Bookstore]. It outlines the functionality, system components, and design decisions to be followed during the development process.

1.2 Scope

This SDS covers the design and implementation details of the [“Chapter Find”– Online Bookstore]. The software will perform the following major tasks:

- **Book Browsing and Search**: Allow users to search, browse, and view detailed book information.
- **Purchase and Payment Processing**: Facilitate secure book purchases with online payment options.
- **Order Tracking**: Enable users to track the status of their orders.
- **Admin Book Management**: Allow admins to add, edit, and delete book listings.
- **Shipping Management**: Allow users to select and track shipping methods for purchased books.
- **Book Categorization**: Enable users to browse books by category, genre, or theme for easier discovery.

---

2. System Overview

The system consists of the following components:

- **Frontend**: HTML, CSS, java script [Razor page using ASP.net]
- **Backend**: C#
- **Database**: SQL Server

---

3. System Architecture

3.1 Architectural Design

This project follows the architecture, where:

- **Frontend** communicates with the backend using Protocol.
- **Backend** interacts with the database to manage and retrieve data.

3.2 Data Flow

1. **User Interaction**: Users interact with the frontend interface to browse, search, and make purchases.
2. **Request Processing**: The frontend sends an API request to the backend server for specific actions.
3. **Data Handling**: The backend processes the request, interacts with the database, and fetches or updates the necessary data.
4. **Response**: The backend sends the response back to the frontend, updating the UI.

---

4. Database Design

4.1 Database Schema

The system uses a **relational database (MySQL)** with the following entities and relationships:

Table 1: Categories

- **CategoryID**: Unique identifier for categories (Primary Key).
- **CategoryName**: Name of the category.
- **img**: Path to the category image.

Table 2: Authors

- **AuthorID**: Unique identifier for authors (Primary Key).
- **Name**: Author's full name.
- **TopCategoryID**: ID of the top category the author is associated with (Foreign Key to Categories).
- **Description**: Short bio of the author.
- **img**: Path to the author's image.

Table 3: Books

- **BookID**: Unique identifier for books (Primary Key).
- **Title**: Book title.
- **AuthorID**: ID of the author (Foreign Key to Authors).
- **CategoryID**: ID of the category (Foreign Key to Categories).
- **Price**: Book price.
- **IsDiscount**: Indicates if the book is discounted (0 or 1).
- **Discount**: Discount amount.
- **InStock**: Quantity in stock.
- **SDescription**: Short description of the book.
- **Description**: Detailed description of the book.
- **ReleaseDate**: Release year.
- **NuOfPage**: Number of pages.

- **Collection**: Indicates if the book is part of a collection (0 or 1).
- **img**: Path to the book's image.
- **Visabilty**: Visibility status (1 for visible, 0 for hidden).

Table 4: Customers

- **CustomerID**: Unique identifier for customers (Primary Key).
- **Username**: Customer's username.
- **Password**: Customer's password.
- **Name**: Customer's full name.
- **Email**: Customer's email address.
- **PhoneNumber**: Customer's phone number.

Table 5: Orders

- **OrderID**: Unique identifier for orders (Primary Key).
- **CustomerID**: ID of the customer placing the order (Foreign Key to Customers).
- **OrderDate**: Date of the order.
- **TotalAmount**: Total order amount.
- **Status**: Order status (e.g., "Shipped", "Processing").
- **ShippingAddress**: Address where the order is shipped.
- **PhoneNumber**: Contact phone number for the order.

Table 6: OrderDetails

- **OrderID**: Order identifier (Foreign Key to Orders).
- **BookID**: Book identifier (Foreign Key to Books).
- **Quantity**: Number of copies ordered.
- **UnitPrice**: Price of each book.
- **Primary Key**: Combination of OrderID and BookID.

Table 7: Admin

- **Username**: Admin's unique username (Primary Key).
- **Password**: Admin's password.
- **Name**: Admin's name.
- **Email**: Admin's email.
- **PhoneNumber**: Admin's contact number.
- **Title**: Admin's title.

Table 8: Staff

- **Username**: Staff member's unique username (Primary Key).
- **Password**: Staff member's password.
- **Name**: Staff member's name.
- **Email**: Staff member's email.
- **PhoneNumber**: Staff member's contact number.
- **AuthorityLevel**: Access authority level.

Table 9: Cart

- **CustomerID**: Customer identifier (Foreign Key to Customers).
- **BookID**: Book identifier (Foreign Key to Books).
- **Quantity**: Number of items in the cart.
- **Primary Key**: Combination of CustomerID and BookID.

Table 10: ShippingCost

- **City**: Name of the city (Primary Key).
- **Cost**: Shipping cost for the city.

Table 11: CustomersAddress

- **CustomerID**: Customer identifier (Foreign Key to Customers).
- **City**: City name (Foreign Key to ShippingCost).
- **Address**: Detailed address of the customer.

Relationships:

1. **Categories ↔ Authors**: One-to-Many (Authors are linked to one TopCategoryID).
2. **Categories ↔ Books**: One-to-Many (Books belong to a single category).
3. **Authors ↔ Books**: One-to-Many (Books are written by one author).
4. **Customers ↔ Orders**: One-to-Many (Customers can place multiple orders).
5. **Orders ↔ OrderDetails**: One-to-Many (OrderDetails detail the books in each order).
6. **Customers ↔ CustomersAddress**: One-to-Many (Customers can have multiple addresses).
7. **Customers ↔ Cart**: One-to-Many (Customers can have multiple items in their cart).
8. **ShippingCost ↔ CustomersAddress**: One-to-Many (Addresses link to shipping costs by city).

---

5. Technology Stack

- **Frontend**: HTML, CSS, java script [Razor page using ASP.net]
- **Backend**: ASP.NET,C#
- **Database**: SQL management system
- **Hosting**: AWS, Heroku, Google Cloud, myASP

---

6. Testing Plan

6.1 Unit Testing

Each module and function will undergo unit testing to ensure that individual components are working as expected.

6.2 Integration Testing

Integration tests will validate that different modules (frontend and backend, or backend and database) work together as expected.

6.3 User Acceptance Testing (UAT)

End users will be involved in testing the system to verify that it meets their requirements and expectations.

6.4 Performance Testing

Stress and load testing will be conducted to ensure the system can handle the required number of users and operations without degradation in performance.

---

7. Conclusion

The [“Chapter Find”– Online Bookstore] is designed to fulfill the specified functional and non-functional requirements as described in this SDS. The design outlined here will ensure that the system is robust, scalable, and user-friendly, providing the intended value to its users.