DC2410 Internet Applications and Techniques – Book Store

# File Locations

Website files – public\_html - contains main webpages  
 public\_html/css - contains css files  
 public\_html/img - contains favicon  
 public\_html/js - contains website javascript code  
 public\_html/resources/configs - contains website configuration file  
 public\_html/resources/php - contains website php code  
 public\_html/resources/db - database creation scripts

# Abstract

The Book Store project is an individual assignment to develop an online book store for university students to be able to buy books from the university book shop. It has been developed using PHP, MySQL, HTML, JavaScript + AJAX, and CSS for styling.

My system allows students and staff to register (staff registration has been disabled through a configuration file on the host server) to the Book Store website.

When students and staff can log in, they are always able to see their account balance in a bar across the top of the page. They can also browse books. They can filter the books by category, and can also search the list of books. The search will match on any column of the table so this will include searching on Title, Authors, Category, Quantity, and Price. By clicking on a book in the table, more details on the book will appear as well as an option to add it to the user’s basket. The user may add as many books as are in stock to their basket after which it is not possible to add more to the basket.

The user can then view their basket, and adjust the quantity of any item in their basket (increase/decrease quantity, remove an item completely, or empty the basket) if they wish. All stock is handled when adding/removing items from the basket. They can then click the checkout button to place an order with the Book Store. Once an order is placed, a visual confirmation card is shown to the user with their order id, items ordered, total cost, and instructions on how to complete the order.

Staff can access the staff console page which will allow them to add new books to the Book Store by filling in a form. Staff may also view all users of the Book Store and top up their account balance. Finally, staff members may view all active orders in the Book Store. The orders will appear in a table much like the table for viewing books. This table can be searched on all columns (Order Id, username, and date). By clicking on an order, it will show the details of the order including its cost and a button to complete the order. The order will disappear from the orders list once it has been completed. If a user does not have enough balance in their account, then the order will not be completed.

# User Credentials

|  |  |  |
| --- | --- | --- |
| Username | Password | User Type |
| shopuser | BookStore123 | STUDENT |
| shopstaff | ShopStaff123 | STAFF |

The website can be accessed from this link: <http://www.thakerp.eas-dc2410-1516.aston.ac.uk/>

# Stretchers Implemented

S2 – View book items by Category including the amount in stock and price.

S4 – Remove an item from their basket.

S5 – View their basket and account balance.

T1 – Allow students to top up their account with a cash payment.

T3 – View a student’s basket and account balance.

Searching the book list using certain attributes (the book list can be searched on all columns (title, authors, category, quantity, and price)

I have modified the design slightly to keep baskets and orders as separate entities. Baskets are only controllable and viewable by the owner of that basket. When a user wishes to purchase the items in their basket, an order is placed which can then be seen by staff members so that they can complete the order.

I have implemented the Post-Redirect-Get pattern on webpages which submit to the database. This is so that form resubmission is not possible and so users cannot mistakenly submit a form that has already been submitted.

# Security

I have implemented a few security measures to the website. Since the MD5 and SHA algorithms are very insecure, I have decided to use the native password hashing API functions from PHP - password\_hash(). The reason I’ve decided to use this is that it is far more secure since it uses the BCRYPT algorithm. It also uses a cryptographic salt which means that the output of the hashing algorithm cannot be looked up in rainbow tables.

Another security consideration I have made is in regards to user input. All user input which is input into the database is validated against a regex pattern which restricts the characters and length that can be input. Special characters which are very commonly used in coding languages are forbidden. The most lenient regex within the Book Store is one which is used to validate user input when new books are added. This will only allow alphanumerics, whitespace, commas, periods, ?, !, :, - , single and double quotes, and ampersand. These restrictions mean that no code can be injected into the website or database since the common coding characters such as $, semicolon, and brackets are not allowed.

A further security measure I have taken is to use PDO prepared statements. This is so that every single database query is escaped properly using the PDO library so as to prevent SQL injection attacks.

# Assumptions

Upon usage of the Book Store, it is assumed that the user will know that the rows within the tables that show books, user orders, and user accounts, are clickable in order to view more details.

It is also assumed that when adding books to the system, the user will read the information tips on the form so that it is filled out correctly. It is also assumed that the nomenclature 1-\* means one to many.