Folio

An Ebook Retailer

Zana Ahmad   Hugh Alun-Jones   Thomas Akers   Emmanuel Adedeji

Contents

List of Tables

System Specification

Folio is a website for users to buy and download e-books. Users have the option to search for an e-book using an author name or a title of a book. The homepage will present the user with various books that are randomly chosen. There will also be a browse page to look for a book via publisher, date of publication, author or genre. The project itself consists of 3 systems. Please see the relevant subsections to find out more.

Other pages that would be included within the website include the login and register pages where the user will be asked to enter their details to either register or login. Once this is completed, they can buy a book they would like which then gets added to the Order System for recording who has bought what. Folio consists of four systems that are allocated between four people. More details of each system are shown below under each subsection.

At the backend, we have access to the database so we will directly access data and do all the relevant actions such as addition, deletion, filtration and modification, listing and validation. A form will be designed to take in input for example for adding a new book entry.

Pre-defined SQL queries will be designed to instantly output relevant data for ease of use. For example, a query will be made to return all the users in the database or all books that are bought by a user. The result will return a list.

User Management System

This system is run by Zana who will be creating the following functionality:

As mentioned before, this system will consist of two parts; a backend facing windows form application and a frontend website.

It will carry out functions such as:

- **Adding a user:** for the backend facing form, an admin can add a user by a form that has been provided for them that consists of the same fields as the frontend page will. It will ask for information such as first name, last name, date of birth, email and telephone number.
- **Editing a user:** at the backend, the admin will be able to search for a user and then bring up all the editable fields that were provided and modify them as they please as long as it can be validated. On the frontend, the user can do the same thing but only for their own account.
- **Deleting a user:** the admin can search for a user and completely delete the record. At the frontend the user can choose to delete their own record.
- **Listing users or a single user:** on the backend, an admin can choose to see all the users or search for specific ones. Once this has been done, they can then view the details of that user. A user on the frontend can see their own details on their profile page.
- **Finding users:** an admin can find another user at the backend. This will not be possible as of yet for the frontend. Although it may be an extra feature that could be useful if enough time was allocated.
- **Filtering users:** an admin can use the windows form at the backend to filter out only the users they require to see. For example, an admin can ask the database to retrieve data from a user that is born on April 8$^{\text{th}}$ and that they have bought 32 books.
- **Validating users:** any information should be validated that are going into the database. For example, a user should be at least at minimum age range, possibly at 16 which ensures a user legally can make a purchase. Validation is going to be done at the backend of the application that talks to the database. However, the frontend user will still be notified if something goes wrong such as too few characters for a password.

It is worth mentioning that the windows form will not have a login form for admins. We expect that the machine will be kept secure by providing a password for their machine.

User Schema

Below is the schema for a user. On the register page they would have to provide the following details.

1. First name
2. Last name
3. Date of birth
4. Email
5. Telephone number

Then these data will be sent to the server to be validated. An email also will be sent to the user to verify their information which is essential if they want to buy an e-book so that we know that they will receive an email for the receipt and the pdf of the desired book.

Table 1: User Schema Table

Attribute	Type (`datatype`)	Key
user_id	String `varchar(25)`	Private Key
user_fullname	String `varchar(50)`	
user_password	String `varchar(50)`	
user_dob	Date `date`	
user_email	String `varchar(255)`	
user_tel	String `varchar(13)`	
user_numof_books_bought	Integer `int`	
user_is_email_verified	Boolean `bit`	

With every purchase, we will increment the `user_numof_books_bought` field to keep record of how many books each individual has bought. This may come in handy to see how popular the website as a whole is.

When a user is authenticated whether through the frontend or backend, the rest of the application is handled by the other systems.

Book Management System

Order & Record System

This system will be run by Emmanuel, creating the following functionality:

1. Frontend user order/record table `My Orders`. This page will consist of pulled previous-order information related to the particular `UserID` attempting to access the order records and displays this for each user. The information will be accessed from the larger order-record table which holds the orders of all customers. Each order will have a download button beside it, allowing the user to download the particular book related to the `OrderID`.

2. Backend order addition, deletion, filtration and modification, listing and validation.

3. Frontend user order confirmation. This page will provide each user with an order-number (`OrderID`) and itinerary confirming the details of the book they have purchased and redirection link to the `My Orders` page where they can download their book.

Order & Record Schema

On the table below, the template for each user order on the `My Orders` is shown.

On the user order page, each customer will have the opportunity to enter an integer between 1-5 representing their level of satisfaction of the order. Helping gauge the popularity of each book, particularly in identifying the sites best sellers.

Table 2: Order & Record Schema Table

Attribute	Type (`DataType`)	Key
OrderID	Integer `int`	Primary Key
UserID	String `varchar(25)`	Foreign Key
BookID	String `varchar(255)`	Foreign Key
DateOfPurchase	Date `date`	
isClaimed	Boolean `bit`	
Satisfaction	Integer `int`	

Simultaneous to every additional order in the system, there will be an increment in the `OrderID` value.

There is a `one to zero or many` and `one to one (mandatory)` relationship between `UserID` and `OrderID`.