|  |  |  |
| --- | --- | --- |
|  | | |
|  | **CMPS 350 Project Phase 1 – WebApp UI Design and Implementation**  **E Commerce Platform**  **(15% of the course grade)**  **The project code is accessible on the following (github) link: https://github.com/moumenkhha/Project.git** | |
| **Group Id:** | |  |
| **Group Members:** | | Moumen Abd Alkadr Alkhateb Alhasani (201910156)  M.Reyad Ahmad Tuama Halabi (201902988)  **Emails:** ma1910156@student.qu.edu.qa; mt1902988@student.qu.edu.qa |

**Grading Rubric - In the Functionality column please specify either: *Working (completed x%)*, *Not Working (completed x%)* or *Not done*.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Criteria** | **Weight%** | **Functionality\*** | **Quality of the implementation** | **Your Grade** |
| 1) Design and implement the app Web UI and navigation using HTML, CSS and JavaScript. Including designing the App Web UI and navigation. | 30 | Working (completed 100%) |  |  |
| 2) Design and implement the Web API and the server-side data access repositories to read/write the app data JSON files. | 60 | Not done. Local Storage is used for Phase1 |  |  |
| **3) Application Design:** Entities, Repositories and Web API class diagrams. | 5 | Not done. |  |  |
| **4) Testing documentation** using screen shots illustrating the testing results.  - Discussion of the project contribution of each team member. Members should collaborate and contribute equally to the project. | 5 | Working (completed 100%) |  |  |
| **Total** | 100 |  |  |  |

***Important remark:*** *In case of copying and/or plagiarism or not being able to explain or answer questions about the implementation, you lose the whole grade.*

**\* Criteria for grading the functionality:**

- The functionality is working: you get 70% of the assigned grade.

- The functionality is not working: you lose 40% of assigned grade.

- The functionality is not implemented: you get 0.

- The remaining grade in all cases from above **is assigned to the quality of the implementation**,

- The grades are distributed on the various use cases, when the design/implementation is partial, you get only the grades of designed/implemented use cases.

Code quality criteria, include:

- Use of meaningful identifiers for variables and functions (e.g. using JavaScript naming conventions)

- Pages are responsive

- Clean code: simple and concise code, no redundancy

- Clean implementation without unnecessary files/code

- Use of comments where necessary

- Proper code formatting and indentation.

**You lose marks** for code duplication, poor/inefficient coding practices, poor naming of identifiers, unclean/untidy submission, and unnecessary complex/poor user interface design.

**Important Remark**:

**[Grades: 100-85]:** Will be given only to **fully functional application** with **all the quality criteria cited above met** and the project has excellent **design for the various functionalities**. **The report is professional**.

**[Grades: 85-80]:** Will be given only **to fully functional application** **with most of all the quality criteria cited above met** and the project has good design for the various functionalities. **The report is professional**.

**[Grades: 80-75]:** 80% of the application functionalities are functional. The project respects partially the quality criteria. **The report is professional** but misses some iformation.

The grades are not negotiable. We expect that only a small portion (around 15%) of the class will be able to meet the criteria for the grades **[100-85]. You should work hard to and demonstrate the merits of your application to earn those grades.**

# Description of your proposed platform: A website to purchase books consisting of 3 pages: index, login, and my Activity to show user history of purchasing. It starts with a home page with a header containing navigations and search bar, samples of books, and a footer. A user can search for a specific book but cannot proceed adding it to cart unless he is login. He can see its title, authors, ISBN, publisher, and price. Also, he cannot visit my Activity page unless he is login. Taping the login link will lead to another page called login. It is simple: the book store logo and a box welcoming the user to enter his Id and password. A pre-written JSON file with users’ IDs and their passwords is prepared. If the username is wrong, the userId label will be red. If it is correct but the password is wrong, the password label will be red and the first will be green. If both, add the user information the local Storage, then return to main.

# The search bar is not located well. My friend Reyad designed the home page. It was ok, but when I took his code and started building on it, it moved a little, I do not know why.

# Typing in the search bar will fire an event connected to JSON file containing books details. Selecting a book will remove main contents of index.html, replacing it with a table of book details and a button add to cart, and adding the book details to local Storage. Logging is a pre-condition for adding to cart. If adding, a form check-out will appear, to select zone, street, building, unit, quantity. Selecting a quantity will lead the total price to be automatically shown. Every input is required. If the user missed one and confirmed the order, an alert followed by a message under the field missed will be shown.

# If everything is ok, the quantity selected by user is not more than what is in the bookstore, his balance allows to purchase the desired quantity, the location is fully stated, an alert saying “order confirmed. Redirecting to Main” will appear. If balance is less or quantity is not enough, also an alert will notify the user.

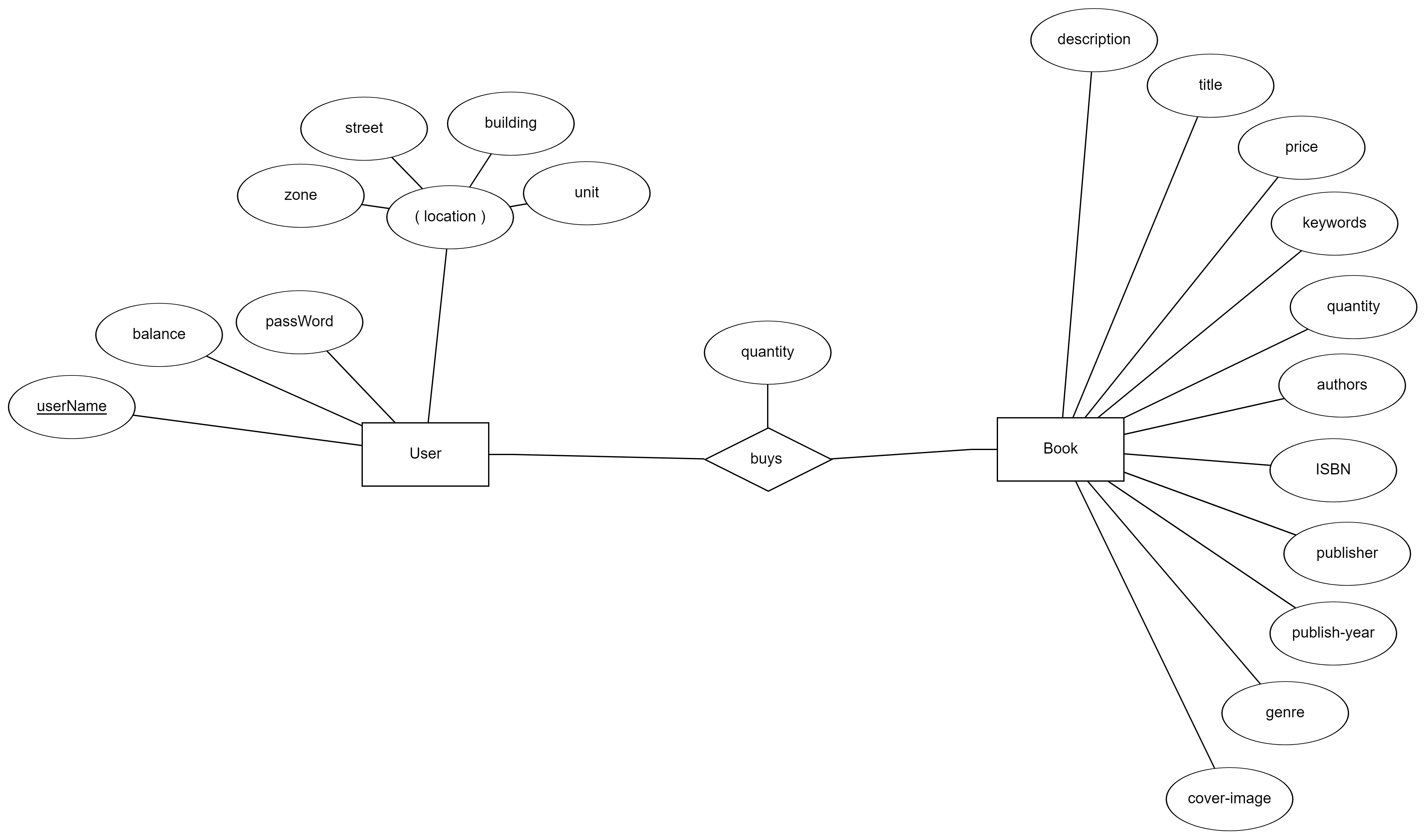
# Clicking on my activity will lead to a new page to display the user history. It will show the date, book title, quantity purchased, and total price taken from the user balance.

# Each page has the same design (header) except the login page. Signing out will clear local storage. When dividing the project between us, my use cases was login, purchase an item, view the purchase history, and Reyad took the rest. I think I did the three use cases.

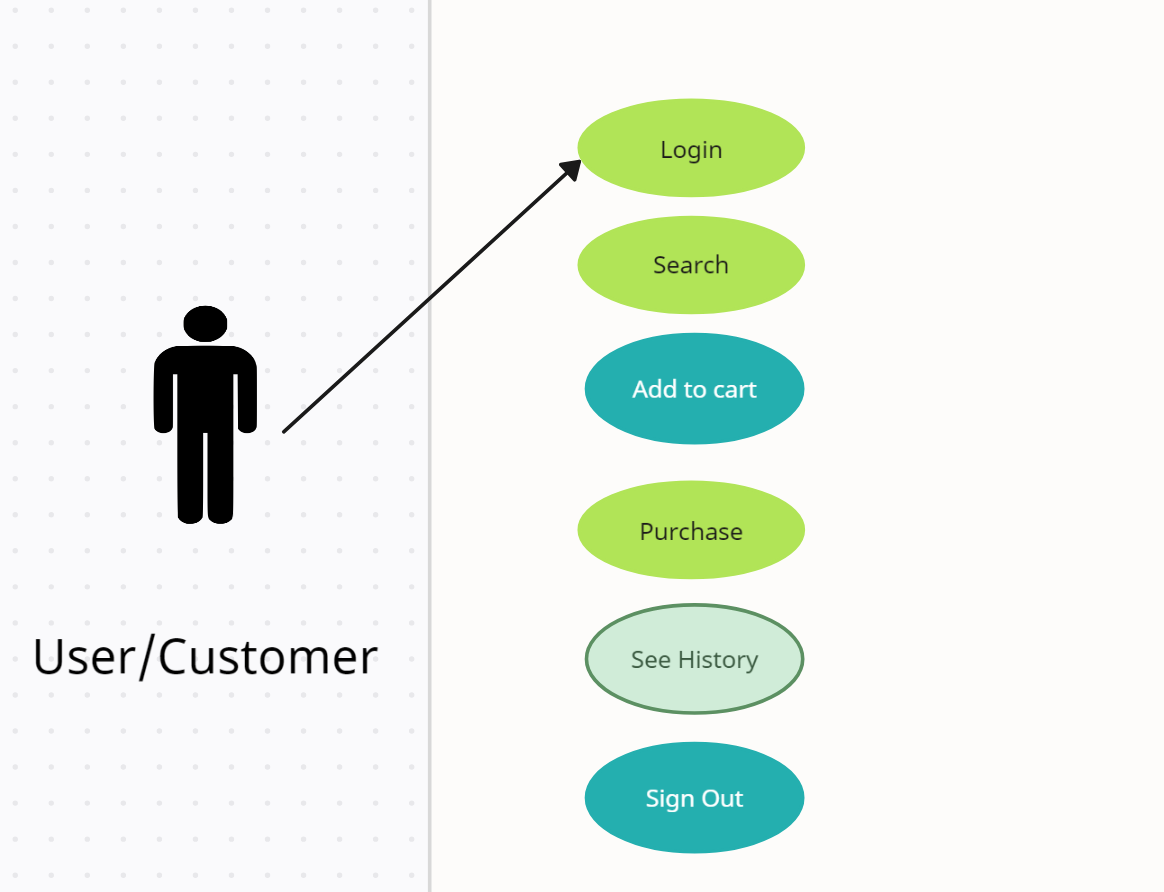
# Also, every book should hold its image, but we did not do that.

# Application Design

# Entities class diagram



# Use case diagram



# Web API class diagram

# Implementation

# Implemented use-cases:

# 1 [Login]: Working. Checks and validates username and password. Showing if wrong input not matching.

# 2 [Search Available]: Working. It is functioning well, displaying suggestions whenever the user change the input field.

# 3 [Purchase an item]: Working. Total price and user balance and quantity of a book is linked to local storage. From that is handling the data.

# 4 [View the purchase history]: Working. Also linked to local Storage. each purchase operation will be displayed in a div separating from the others. Displays the date, title, quantity, price.

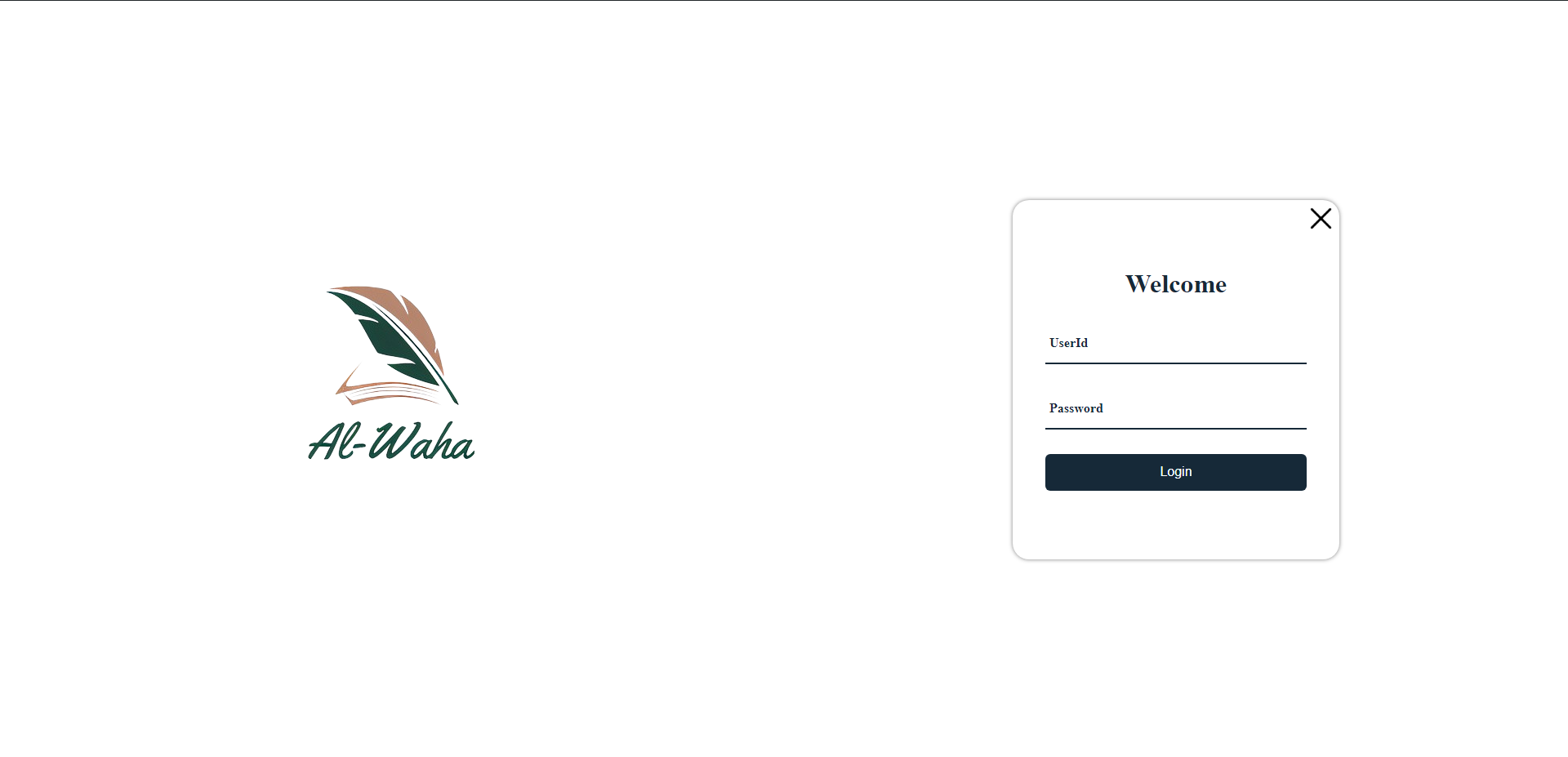
# Unimplemented use-cases and not functioning parts

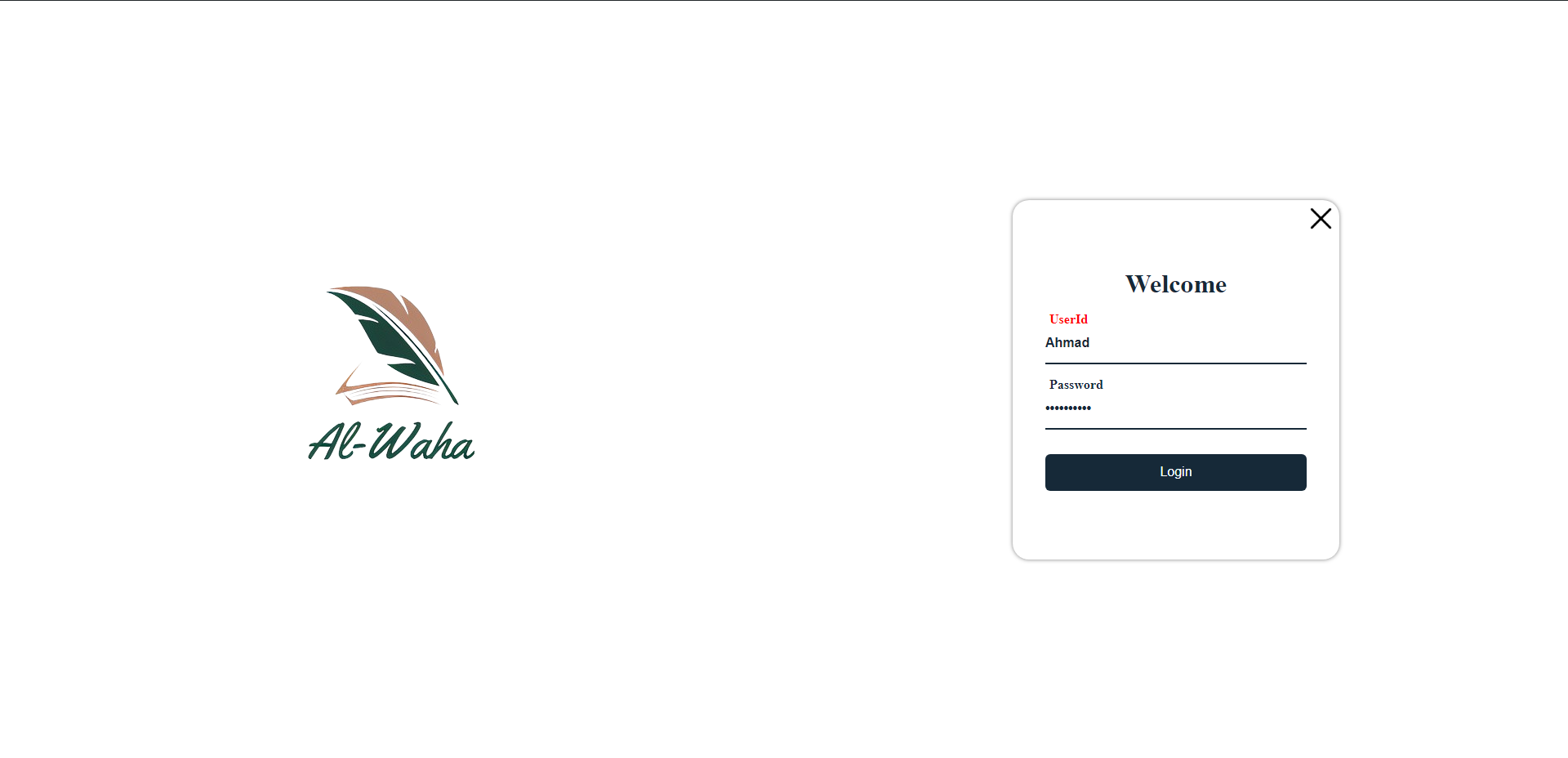
# 5 [View available items on sale and sale history]: It was required from Reyad.

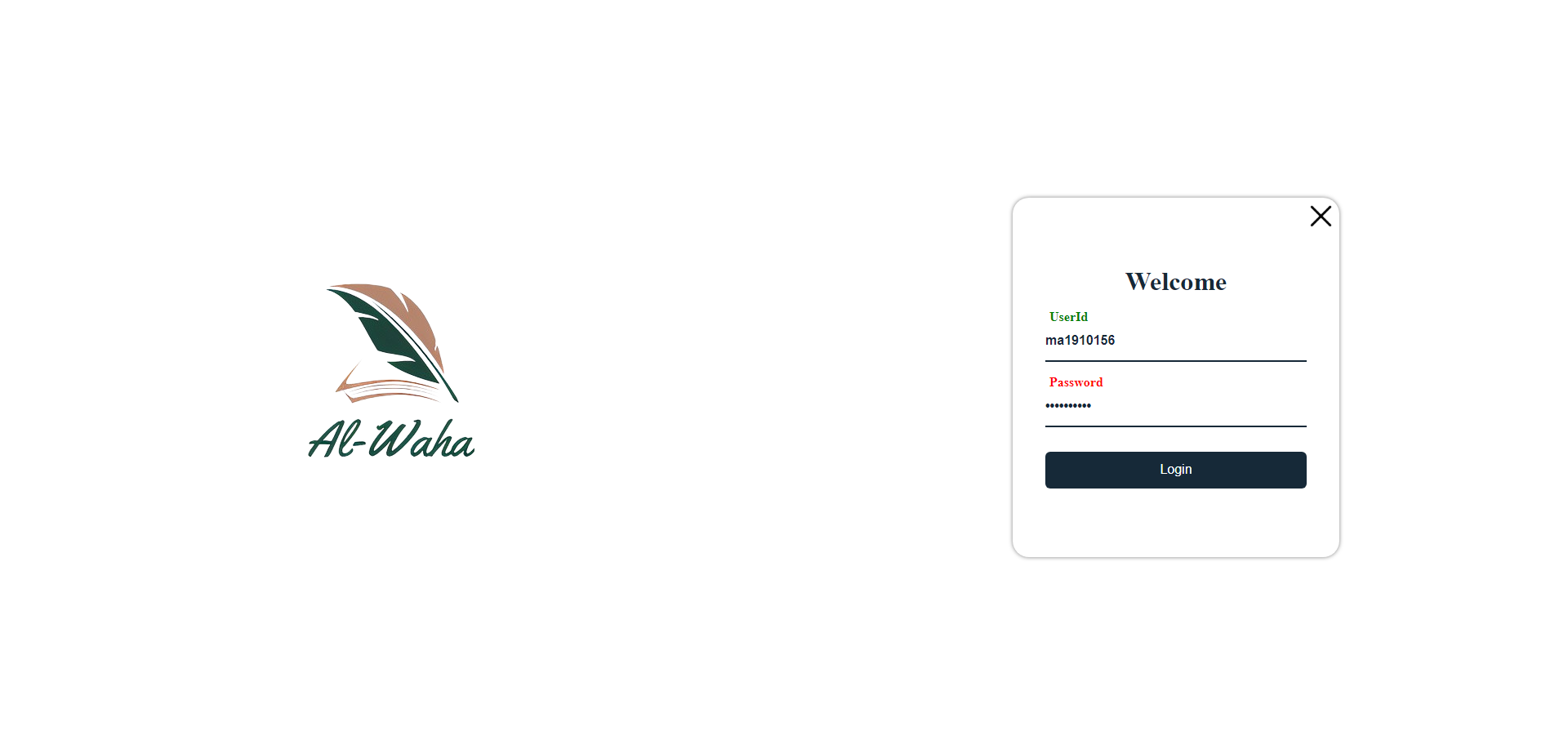
# 6 [Upload an item to be sold]: It was required from Reyad.

# Testing

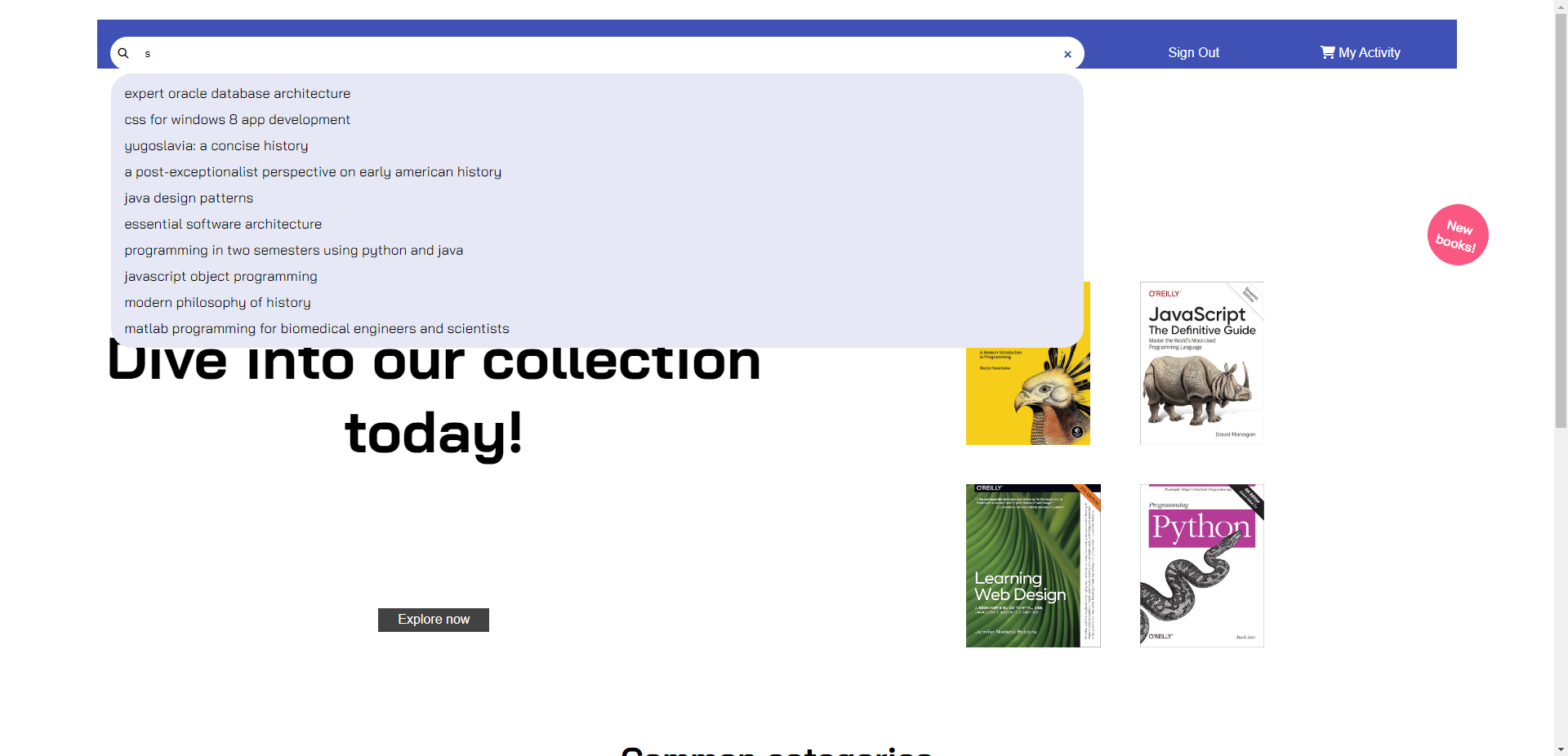
# Use case 1



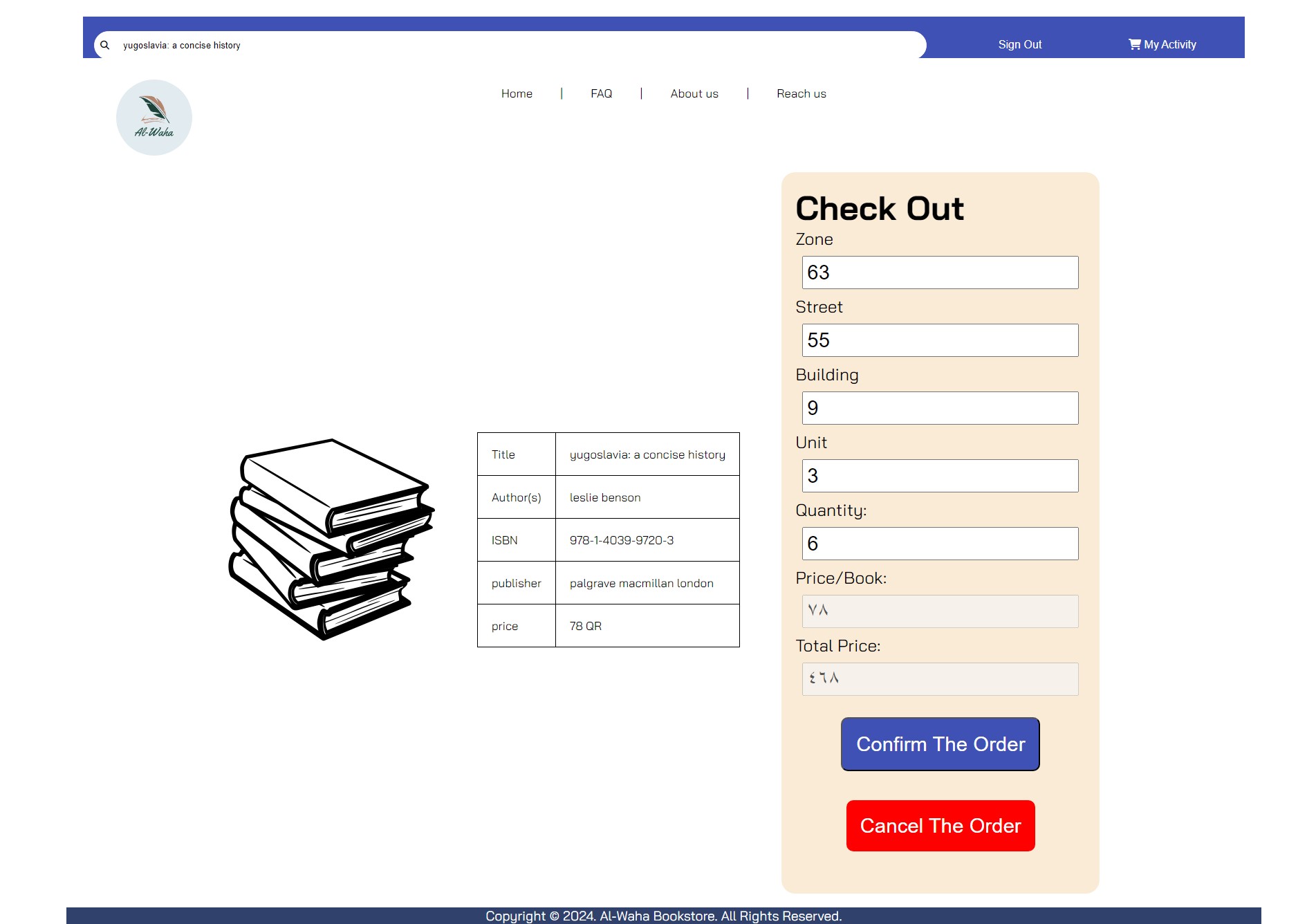




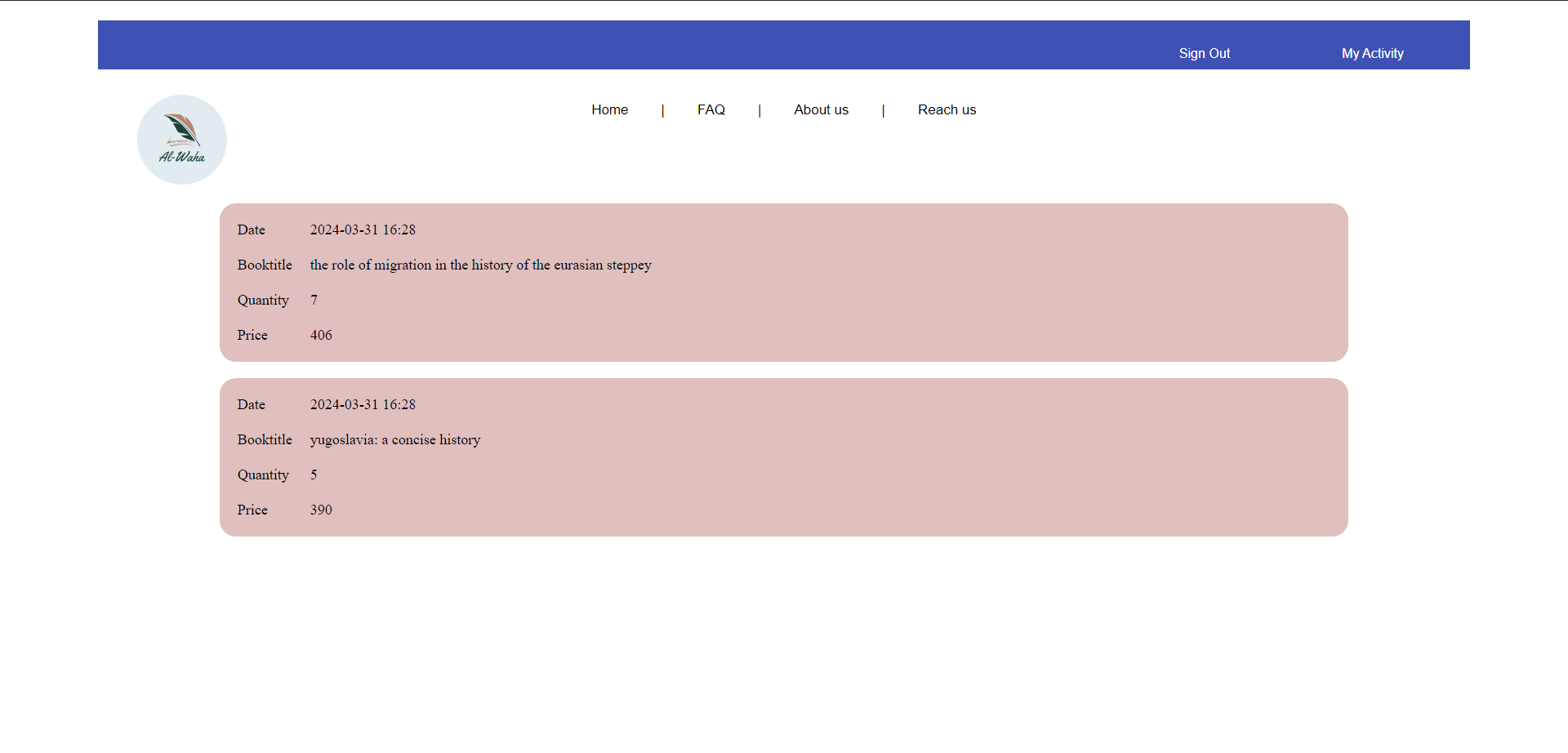
# Use case 2



# Use case 3



# Use case 4



# Use case 5: Not implemented.

# Use case 6: Not implemented.

# Discussion of the project contribution of each team member: I took Use cases 1,3,4. Reyad took 2,5,6. I did them and he did use case 2. We were meeting regularly and discussing face to face & online. He is good in design. He likes to do the best, and was discussing everything, small and big.