

Code Developer Project:

Revenue and User Management Website

Goal: The primary aim of this project is to develop a comprehensive revenue and user management bookstore website.

I – Requirements Gathering and Project Planning

Part I – Requirements Survey

Following a client meeting, the project involves the creation of a bookstore website with the following features:

For Any User:

- View books
- Search for books by various fields (Book details)
- Register user

For Registered Users:

- Submit interest in buy the book (if flag is “Available”) Button “interested in buy”
- Comment on books
- Rate books (1 to 5)
- Access a personal area
- Maintain a list of favorite books

For Administrators:

- Manage users, with the ability to block them
- Manager books:
 - Adding and editing all fields
- Managing flags (counting flag “sold”, search /counting Button "Interested in buy" + user email +Book details, change the “available” to “sold”)

Books Details:

- List of Books (Title, Author, Editor)
- Genders (Unkown, Fiction gender, Non Fiction gender)
- Language (portugues, english, other)
- Flag (available, sold, not available)

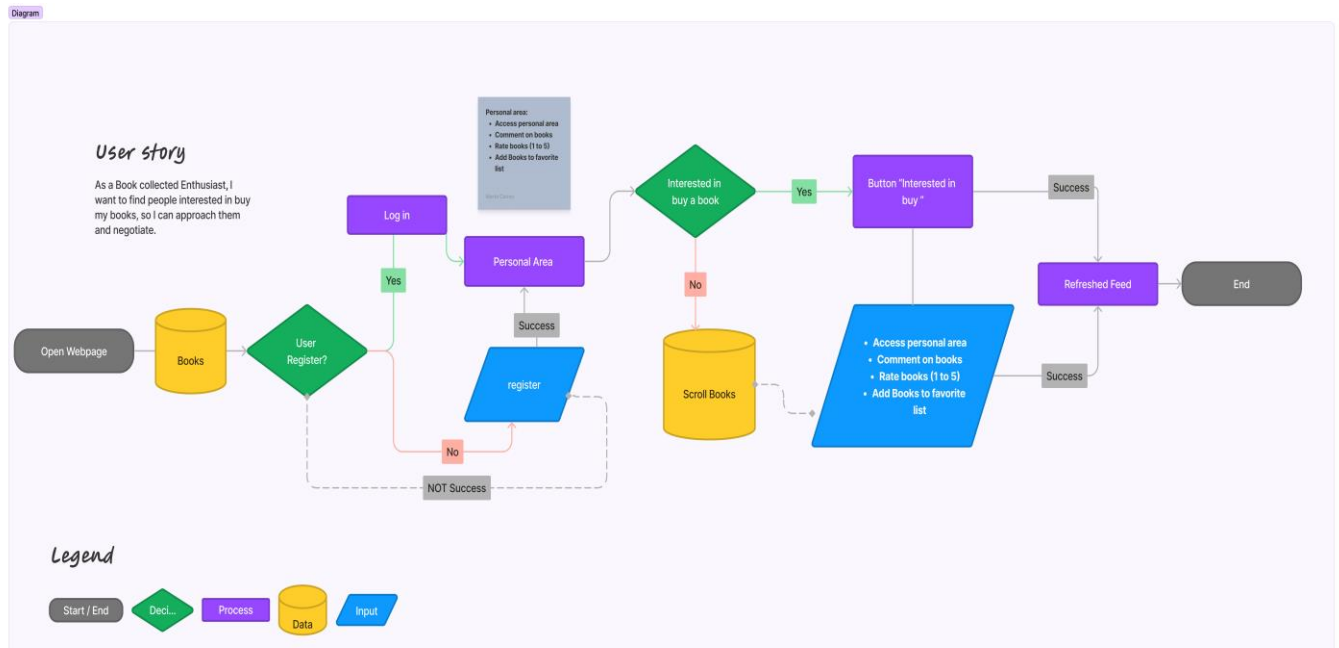
Registered User's Personal Area:

- Name
- Email
- Password

Current Stage Objectives:

- Create a list of functionalities and sub-functionalities associated with the identified needs.
- Develop use case diagrams (flowcharts) and application functionalities, along with their pseudocode .
- Design mockups/layouts of website forms for both the front office and back office.

Note: conceptualizing the structure for all elements, as the remaining diagrams will be essential for the comprehensive development of the project.



List of functionalities and sub-functionalities:

View Books:

- Sub-functionality: Browse books by title, author, or editor.
- Sub-functionality: Display book details.

Search for Books:

- Sub-functionality: Search books by various fields (title, author, editor).
- Sub-functionality: Filter books based on gender, language, and availability.

User Registration:

- Sub-functionality: Provide user details (name, email, password).

Registered Users:

- Sub-functionality: Submit interest in buying a book (if available).
- Sub-functionality: Comment on books.
- Sub-functionality: Rate books (1 to 5).
- Sub-functionality: Access a personal area.
- Sub-functionality: Maintain a list of favorite books.

Administrators:

- Sub-functionality: Manage users, including the ability to block them.
- Sub-functionality: Manage books, including adding/editing all fields.
- Sub-functionality: Manage flags (counting, searching, and updating availability).

Diagrams & Pseudocode

(step-by-step guides for someone who's not into coding)

1. User Registration:

What it Does:

Helps you sign up for the website.

Steps:

- 1.1 You fill in your name, email, and password.
- 1.2 The system checks if everything looks good.
- 1.3 If it's okay, you're all set!

Diagram:

Pseudocode:

if user submits registration form:

 validate user input

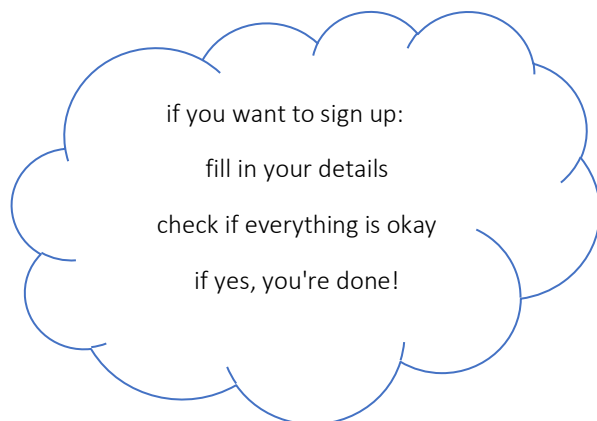
 if input is valid:

 create user account

 redirect to user area

 else:

 display error messages



2. View Books:

What it Does:

Lets you see what books are there.

Steps:

2.1 You say you want to look at books.

2.2 The system shows you the list of books.

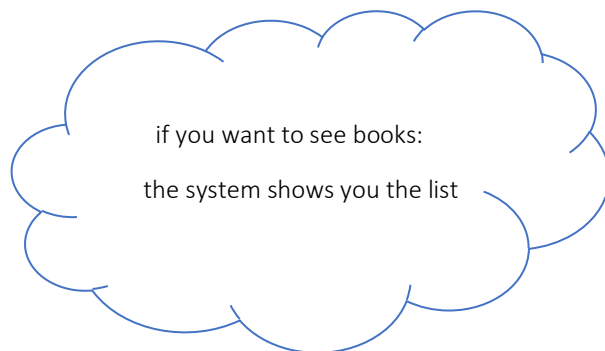
Diagram:

Pseudocode:

if user selects "View Books":

 retrieve list of books from the database

 display books on the front end



3. Submit Interest in Buying a Book:

What it Does:

Tells the system you want to buy a book.

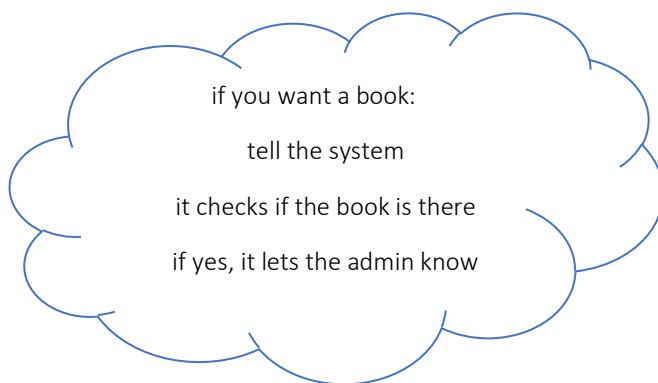
Steps:

- 3.1 You express interest in a book.
- 3.2 The system checks if the book is available.
- 3.3 If yes, it tells the admin you want it.

Diagram:

Pseudocode:

```
if user submits interest in a book:  
    check availability of the book  
    if available:  
        notify administrator  
        update book status to "Interested"  
    else:  
        display message that the book is not available
```



4. Admin Manage Books:

What it Does:

Helps the admin handle books.

Steps:

4.1 The admin can see a list of books.

4.2 They can add or edit book details.

4.3 They can manage flags like available/sold/not available.

Diagram:

Pseudocode:

if admin wants to manage books:

 show list of books

 allow adding/editing book details

 allow managing flags (counting, searching, updating availability)



Desing Mockups/Layouts

(look at the webpage, you'll see clear sections for entering your details, exploring book information, and the admin having control over users and books.)

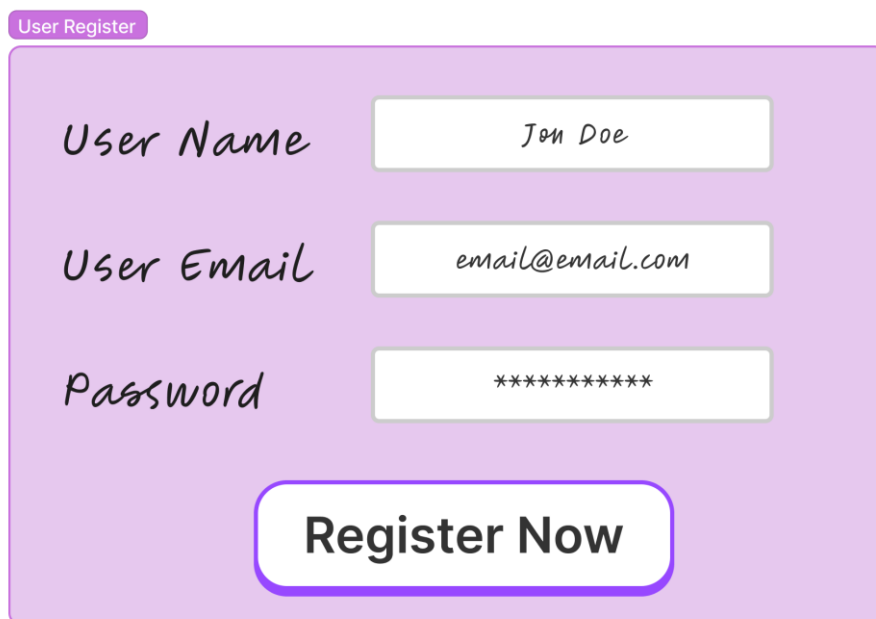
1. User Registration Form:

Fields:

- **Name:** Your full name goes here.
- **Email:** Your email address, so we can keep in touch.
- **Password:** Your secret key to access your account.

Button:

- **Register:** Once you've filled in your details, hit this button to officially join.



A mockup of a user registration form. It features a light purple background with a darker purple border. At the top left, there is a small purple pill-shaped button with the text "User Register". Below this, there are three input fields stacked vertically. The first field is labeled "User Name" in a cursive font and contains the text "Jon Doe". The second field is labeled "User Email" in a cursive font and contains the text "email@email.com". The third field is labeled "Password" in a cursive font and contains ten asterisks "*****". Below the input fields is a large, rounded rectangular button with a purple border and the text "Register Now" in a bold, sans-serif font.

2. Book Details Page:

Display:

- **Title:** The name of the book.
- **Author:** The person who wrote the book.
- **Editor:** The person who prepared the book for publishing.
- **Gender:** Whether the book is Fiction, Non-Fiction, or Unknown.
- **Language:** The language the book is written in.
- **Flag:** This shows if the book is available, sold, or not available.

Button:

- **Submit "Interest to Buy":** If you really want to buy this book, hit this button.
- **Comment (Area):** Share your thoughts on the book in this space.
- **Rate (from 1-5):** Give the book a rating from 1 to 5 stars.



3. Admin Panel:

User Management Section:

- **List of Users:** See a list of everyone using the website.
- **Block/Unblock Buttons:** Control who can and can't use the website.

Book Management Section:

- **Add/Edit Books:** Put new books into the system or change details of existing ones.
- **Manage Flags:** Keep track of how many books are sold, search for books with user interest, and update availability.

“Interested in Buy” List:

- List of user that press interested in buy button in each book details.

User Management

1	User X	emailX@email.com	Block / Unblock
2	User Y	<u>emailY@email.com</u>	Block / Unblock
3	User W	<u>emailW@email.com</u>	Block / Unblock

Book Management

1	Book Title X	Author X	Editor X	Language	Gender	Flag
2	Book Title Y	<u>Author Y</u>	Editor Y	Language	Gender	Flag
3	Book Title W	<u>Author W</u>	Editor W	Language	Gender	Flag

"Interested in Buy" List

1	User W	emailW@email.com	Book Title X	Author X	Editor X	Language	Gender
2	User X	emailX@email.com	Book Title Y	Author Y	Editor Y	Language	Gender
3	User X	emailX@email.com	Book Title W	Author W	Editor W	Language	Gender

Part II - Backend

1. Create the model layer (entities) of the application, with the description of each entity and your relationships
2. Create the project's persistence layer (DAL) using the Entity Framework
3. Create the Web API (REST) layer

The. This layer must be protected by authentication

4. Create the services layer for communication between Web API and DAL
5. Use Postman / SoapUI to communicate and test and save all requests in one collection

Note 1: each layer must be in separate projects, within the same solution.

Note 2: at least one class, repository, service and controller must be developed for books. It means that CRUD operations must be functional for revenue.

These requests (as well as the authentication method) must be delivered to a collection of the Postman/SoapUI.

Note 3: When running the program, it must check whether a local database exists, otherwise you must automatically carry out the migration as well as seed the database with the information they find relevant to test the API.