




Name: MUHAMMAD DANIEL HAKIM BIN KAMAROLZAMAN		
Id Number: AM2211012877		
Lecturer Name: SIR MOHD AKMAL BIN MOHD AZMER		Section No.: SECTION 01
Course Name and Course Code: SECURE PROGRAMMING / SWC3503		Submission Date: 13 NOVEMBER 2023
Assignment Title: INDIVIDUAL FINAL PROJECT		Extension & Late Submission: Allowed / Disallowed
Assignment Type: INDIVIDUAL PROJECT	% of Assignment Mark: 40%	Returning Date:
Penalties: <ol style="list-style-type: none"> 10% of the original mark will be deducted for every one-week period after the submission date No work will be accepted after two weeks of the deadline If you were unable to submit the coursework on time due to extenuating circumstances, you may be eligible for an extension Extension will not exceed one week 		
Declaration: I/We the undersigned confirm that I/we have read and agree to abide by these regulations on plagiarism and cheating. I/we confirm that this work is my/our own. I/we consent to appropriate storage of our work for checking to ensure that there is no plagiarism/academic cheating.		
<div style="text-align: center;">  </div>		
Signature(s):		
Name: (MUHAMMAD DANIEL HAKIM BIN KAMAROLZAMAN)		
This section may be used for feedback or other information		

Table of Contents

Introduction.....	4
Overall Functionality.....	4
Project Background.....	4
Justification on the Chosen Programming / Framework	5
PHP	5
Java.....	5
CSS	6
Design and Architecture	7
Case Diagram	7
Flow Diagram.....	9
Class Diagram.....	10
Contents	12
The functionality of the system	12
Register New Books	12
View Books	13
Update Existing Books.....	14
Delete Existing Books	15
Roles Based Account.....	16
1. User Roles:	16
2. Librarian Role:	16
3. Ordinary User Role:	16
4. Authentication Mechanism:	16
5. Authorization Mechanism:	16
6. Example Scenarios:.....	17
7. Flexibility for Future Roles:.....	17
8. Security Measures:	17
Implementation of Security Measure	18
1. Authentication and Authorization:.....	18
2. Secure Communication:	18
3. Input Validation:	19
4. Output Encoding:	19

5. Password Security:	19
6. Secure Database Practices:	19
7. Error Handling and Logging:	20
8. Securing Admin Pages:	20
9. Regular Security Audits and Updates:	20
Testing	21
Admin Part	21
Login Page	21
Dashboard Page	23
User Part	27
User Login Page	27
Dashboard Page	28
Security Measure	29
Reflective Conclusion	30
Positive Aspects:	30
Role-Based Access Control (RBAC):	30
SSL/TLS Encryption:	30
Input Validation:	30
Password Security:	30
Secure Database Practices:	30
Secure Communication with HTTPS:	30
Suggestions for Improvement:	31
Error Handling and Logging:	31
Session Management:	31
Regular Security Audits:	31
Security Headers:	31
Client-Side Security:	31
Authentication Best Practices:	31
User Account Management:	31
Conclusion	31
References	32
Appendix	32

Software Security Design and Development Report

Introduction

Overall Functionality

The Library Management System is a comprehensive solution designed to efficiently manage the library's book inventory. It provides a user-friendly interface for librarians and users to interact securely with the system, ensuring smooth book management and user registration.

Library Functions:

- Log in with credentials
- Create, read, update and delete book records
- View the list of available books
- Search for books by various criteria
- View the list of registered users
- Update, create, and delete user records

User Functions:

- Log in with credentials
- View the list of available books

Project Background

The library management system requires a modern and robust system to streamline its operations. The system aims to automate the process of book management, user registration, and interactions between librarians and users, enhancing the overall efficiency of the library.

Justification on the Chosen Programming / Framework

PHP, JavaScript, and CSS were chosen as the primary technologies for this project due to several reasons:

PHP

Web Development Focus:

PHP is a server-side scripting language specifically designed for web development. It excels at handling server-side tasks and interacting with databases, making it well-suited for building web applications like your library system.

Wide Adoption in Web Development:

PHP is widely used in the industry and has a large community of developers. This ensures a vast pool of resources, libraries, and frameworks that can be leveraged for faster development and problem-solving.

Integration with Databases:

PHP has excellent support for various databases, including MySQL, which is commonly used in web applications. This is crucial for a library inventory system that needs to interact with a local database.

Java

Platform Independence:

Java is known for its "Write Once, Run Anywhere" philosophy, meaning Java applications can run on any device with the Java Virtual Machine (JVM). This makes it a versatile choice for a system that may need to be deployed on different platforms.

Scalability and Robustness:

Java is highly scalable and known for its robustness. It can handle large-scale applications effectively, providing stability and performance, which is crucial for a library system that may have a considerable amount of data and users.

Extensive Ecosystem:

Java has a mature ecosystem with a wide range of libraries and frameworks, such as Spring for web development. This can significantly speed up development and contribute to the overall quality of the system.

CSS

Frontend Styling:

CSS (Cascading Style Sheets) is essential for defining the presentation of web pages. It allows for consistent styling across different pages of the system, providing a unified and visually appealing user interface.

Responsive Design:

CSS is crucial for creating responsive designs that adapt to different screen sizes. This is important for ensuring that librarians and users can access the system seamlessly from various devices.

Separation of Concerns:

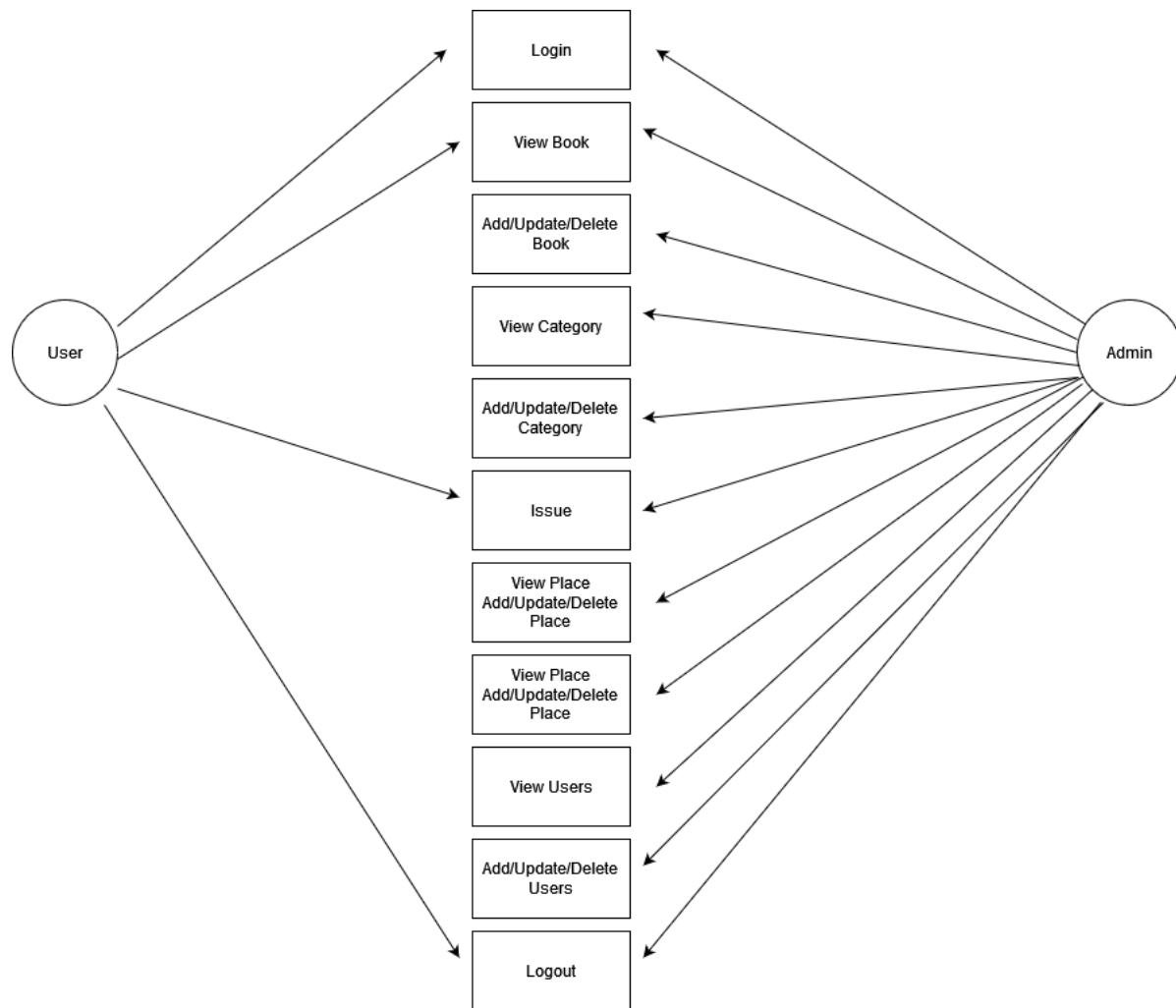
CSS facilitates the separation of concerns by keeping the styling separate from the HTML and PHP/Java logic. This makes the codebase cleaner, more maintainable, and easier to collaborate on.

In summary, PHP is chosen for its web development focus and integration with databases, Java for its platform independence, scalability, and robustness, and CSS for its role in frontend styling and creating a responsive design. Together, they form a strong foundation for developing a secure, efficient, and user-friendly Library Management System.

Design and Architecture

Functional requirements and technical requirements of the secure system

Case Diagram



The use case diagram shows the interactions between the Librarian and User actors and the Library System. The Librarian actor can perform all eleven use cases, while the User actor can only perform the "Login, View, Issue and Logout" use case.

The use cases can be explained as follows:

Create book: The Librarian creates a new book entry in the system.

Update book: The Librarian updates an existing book entry in the system.

Delete book: The Librarian deletes a book entry from the system.

View all books: The Librarian views a list of all books in the system.

Search book: The Librarian searches for a book in the system by title, author, ISBN number, or keyword.

View all users: The Librarian views a list of all registered users in the system.

Create user: The Librarian creates a new user account in the system.

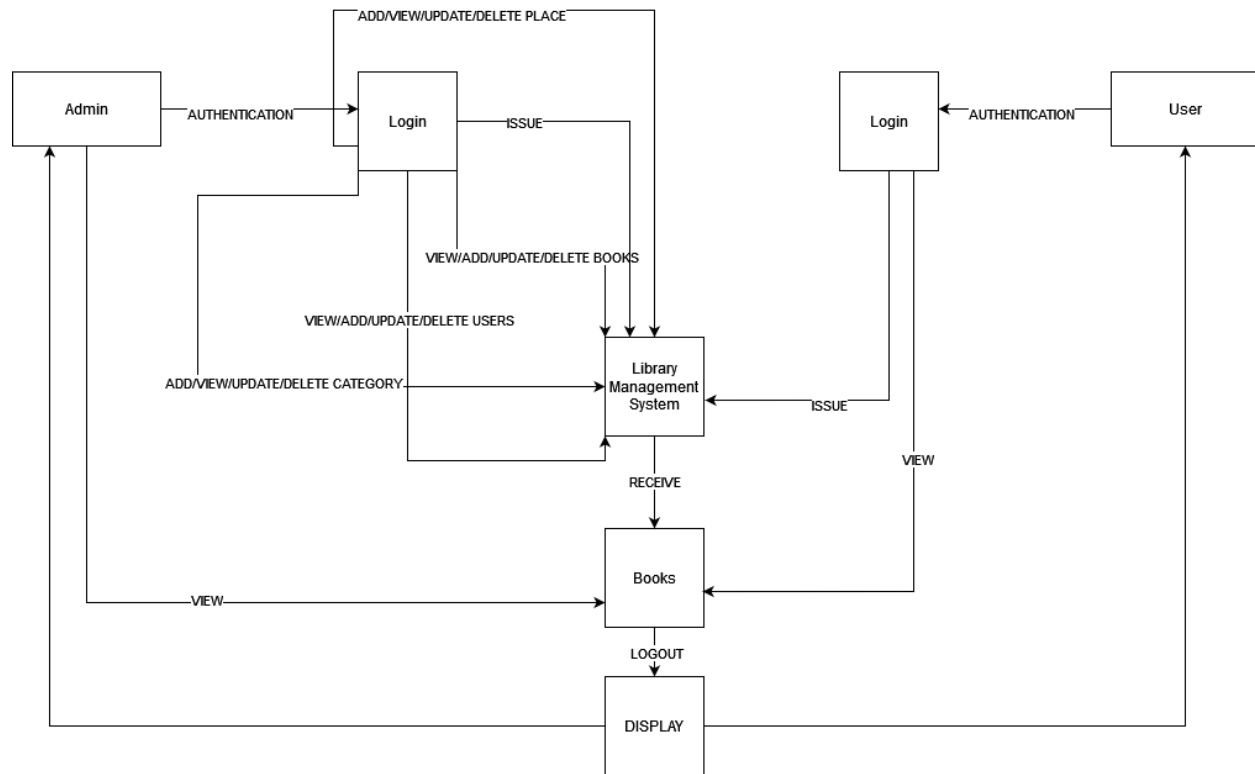
Update user: The Librarian updates an existing user account in the system.

Delete user: The Librarian deletes a user account from the system.

View available books: The User views a list of all books in the system that are available for borrowing.

The use case diagram provides a high-level overview of the Library System's functionality. It can be used to identify and prioritize the system's features, as well as to develop detailed requirements for each use case.

Flow Diagram



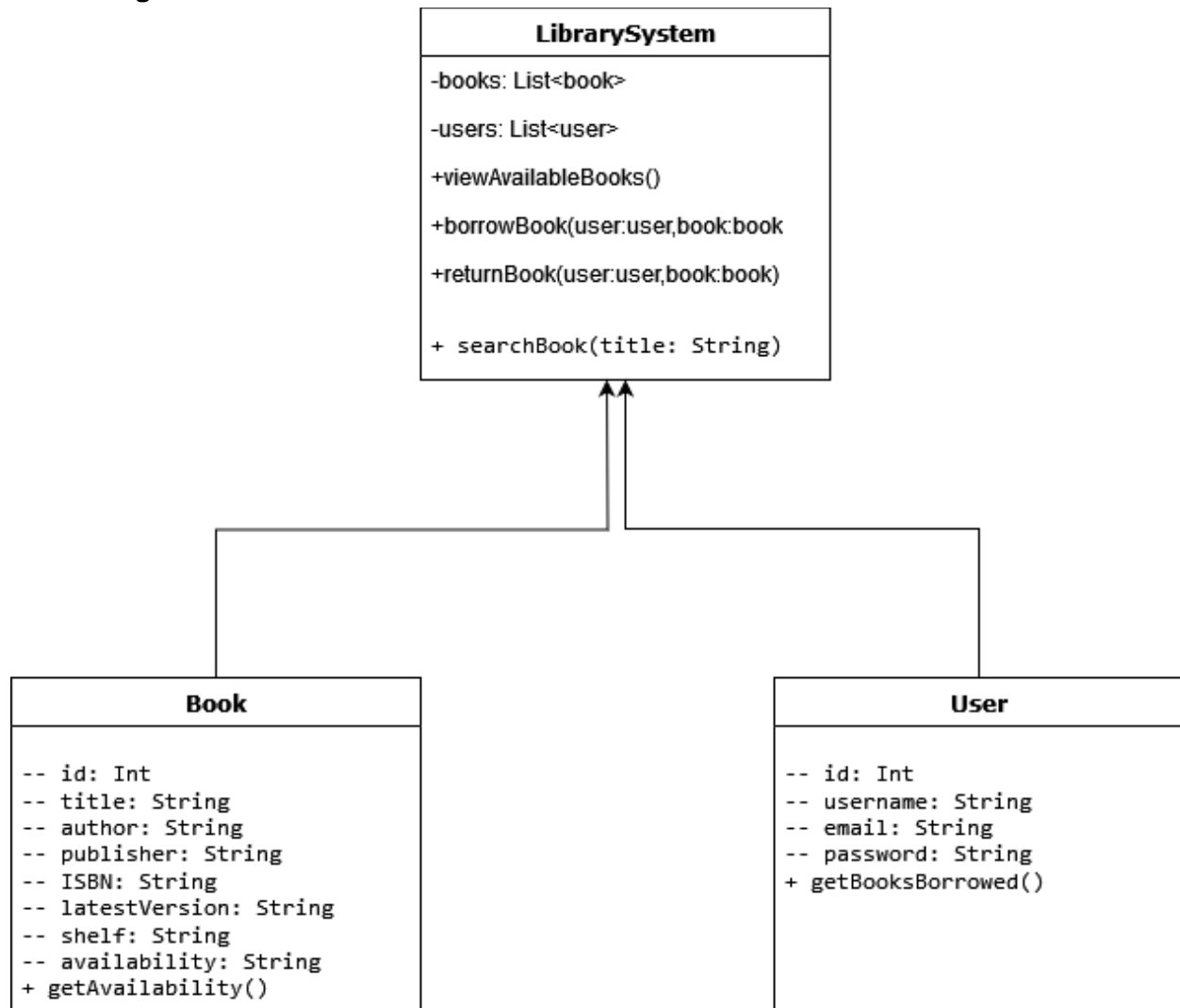
In this Flow Diagram, we can see the flow of the Admin and the User itself. Authentication are the secure measure that are imply in this Library Management System.

Since the Admin is superior it can access all the table in the database and make all the changes while User only can Issue and View the book. The scenario is quite simple since to get the books the user needs to issue it first to the system and the admin can approved or rejected it.

The flow diagram illustrates the following steps:

1. The User logs in to the system.
2. The system displays a list of all available books.
3. The User selects a book to view more information.
4. The system displays the book's information.
5. The User can then select to borrow the book or return to the list of available books.

Class Diagram



The class diagram provides a high-level overview of the structure of the Library System and the relationships between the different classes. It can be used to guide the development of the system and to ensure that the system is consistent and well-designed.

The following are some of the key elements of the class diagram:

LibrarySystem: This class represents the overall library system. It has attributes for storing information about the books and users in the system. It also has methods for managing the books and users.

Book: This class represents a book in the library system. It has attributes for storing information about the book, such as its title, author, publisher, ISBN, latest version, shelf, and availability.

User: This class represents a user of the library system. It has attributes for storing information about the user, such as their username, email address, and password.

The class diagram shows that the LibrarySystem class has a one-to-many relationship with the Book class and the User class. This means that a LibrarySystem object can have many Book objects and User objects associated with it, but a Book object can only be associated with one LibrarySystem object, and a user object can only be associated with one LibrarySystem object.

The class diagram also shows that the User class has a many-to-many relationship with the Book class. This means that a user object can borrow many Book objects, and a Book object can be borrowed by many User objects.

Secure Measures

The following are some of the secure measures that could be implemented for the Library System:

Authentication: Users should be required to authenticate themselves before they can access the system. This could be done using a username and password, or a more secure method such as two-factor authentication.

Authorization: Users should only be granted access to the resources they need to perform their tasks. For example, librarians should have access to all of the system's data, while users should only be able to view the books that are available for borrowing.

Data encryption: Sensitive data, such as user passwords, should be encrypted to protect it from unauthorized access.

Input validation: Input fields should be validated to prevent users from entering malicious code or data.

Logging: All activities in the system should be logged to facilitate auditing and incident response.

The Library System is a valuable resource for the community, and it is important to protect it from unauthorized access and misuse. By implementing a comprehensive set of secure measures, the organization can safeguard its data, protect its users, and ensure that the Library System continues to be a valuable resource for the community.

Contents

The functionality of the system

Describe how the system able to register new books, view books and update as well as delete existing books.

Register New Books

Librarian Access:

The librarian logs into the system using valid credentials, gaining access to administrative functions.

Navigate to Book Registration:

The librarian navigates to the "Add Book" section, typically accessible through a dedicated admin dashboard or menu.

Input Book Details:

The librarian enters relevant information about the new book, such as the entry number, book name, author, publisher, ISBN number, and location.

Submit and Database Update:

Upon completion, the librarian submits the information. The system validates the inputs and adds the new book details to the database.

Confirmation:

The system provides a confirmation message, indicating that the book has been successfully registered.

View Books

User (Librarian or Ordinary User) Access:

Both librarians and ordinary users log into the system using their respective credentials.

Navigate to Book View:

Users navigate to the "View Books" section, typically available in the user interface.

Retrieve Book List:

The system retrieves the list of available books from the database.

Display Book Information:

The system displays relevant information about each book, including entry number, name, author, publisher, ISBN number, and location for Librarian

Search Functionality:

Users have the option to search for specific books based on criteria such as title, author, or ISBN number.

Update Existing Books

Librarian Access:

The librarian logs into the system with administrative privileges.

Navigate to Book Update:

The librarian goes to the "Update Books" section, often found in the admin dashboard.

Search for the Book to Update:

The librarian can search for the specific book using entry number, title, or other identifying information.

Modify Book Details:

The librarian selects the book to update and modifies the necessary details, such as the author, publisher, version, or location.

Submit Changes:

Upon completing the modifications, the librarian submits the changes. The system updates the book details in the database.

Confirmation:

The system provides a confirmation message, confirming that the book details have been successfully updated.

Delete Existing Books

Librarian Access:

The librarian logs into the system with administrative privileges.

Navigate to Book Deletion:

The librarian goes to the "Delete Books" section, usually available in the admin dashboard.

Search for the Book to Delete:

The librarian searches for the specific book using entry number, title, or other identifying information.

Confirm Deletion:

The librarian selects the book to delete and confirms the deletion. The system removes the book entry from the database.

Confirmation:

The system provides a confirmation message, confirming that the book has been successfully deleted.

These processes collectively allow the system to manage the lifecycle of books in the library, from registration to viewing and potential updates or deletions by authorized librarians. The user interfaces for these functionalities should be intuitive, providing a seamless experience for both librarians and ordinary users.

Roles Based Account

Describe how the role-based account works for the system.

The role-based account system in the library management system (LMS) governs the access and permissions granted to users based on their roles. Here's a description of how the role-based account system works:

Role-Based Account System:

1. User Roles:

The system defines distinct user roles, primarily "Librarian" and "Ordinary User."

2. Librarian Role:

Access and Privileges:

Librarians, as administrators, have access to the administrative functions of the system.

They can register new books, update existing book details, delete books, and manage user accounts.

Dashboard:

Librarians typically have access to an admin dashboard or control panel where they can find all administrative functions.

3. Ordinary User Role:

Access and Privileges:

Ordinary users have more restricted access compared to librarians.

They can view the list of available books and search for specific books based on criteria like title, author, or ISBN.

No Administrative Functions:

Ordinary users do not have permissions to register, update, or delete books. Their role is primarily to browse and search for information.

4. Authentication Mechanism:

Login Process:

To access the system, both librarians and ordinary users must authenticate by providing valid credentials (username and password).

5. Authorization Mechanism:

Role-Based Access Control (RBAC):

The system employs Role-Based Access Control to determine what actions each user can perform based on their assigned role.

6. Example Scenarios:

Scenario 1 - Librarian Actions:

When a librarian logs in, they see an admin dashboard with options for book registration, update, and deletion.

Librarians can also manage user accounts, including creating, updating, and deleting user accounts.

Scenario 2 - Ordinary User Actions:

When an ordinary user logs in, they see a simplified interface showing the list of available books and a search functionality.

Ordinary users cannot access administrative functions like book registration or deletion.

7. Flexibility for Future Roles:

Scalability:

The role-based account system is designed to be scalable. If additional roles are introduced in the future (e.g., "Library Assistant" with limited administrative rights), the system can easily adapt.

8. Security Measures:

Authentication and Authorization:

The system ensures secure authentication to verify the identity of users.

Authorization mechanisms prevent unauthorized access to specific features or data.

This role-based account system ensures that different users have appropriate access levels, aligning with their responsibilities within the library context. It contributes to a secure and organized management of books and user accounts in the library management system.

Implementation of Security Measure

Justification on how the implementation of the security measures on the system can be done.

Implementing security measures in the library management system (LMS) is crucial to protect sensitive information, ensure user privacy, and prevent unauthorized access or data breaches. Below is a justification on how the implementation of security measures can be done:

1. Authentication and Authorization:

Justification:

User Identification:

Authentication is implemented to verify the identity of users logging into the system. This ensures that only authorized individuals have access to sensitive information.

Role-Based Access Control (RBAC):

Authorization mechanisms, such as RBAC, are implemented to assign specific roles to users. This ensures that each user has the appropriate level of access based on their responsibilities.

Security Tokens or Session Management:

Security tokens or robust session management techniques are used to maintain secure user sessions, preventing session hijacking and unauthorized access.

2. Secure Communication:

Justification:

SSL/TLS Encryption:

All communication between the client and the server is secured using SSL/TLS encryption. This prevents eavesdropping and man-in-the-middle attacks, ensuring data integrity and confidentiality.

Secure Data Transmission:

Input and output data from the system are transmitted securely, protecting sensitive information such as login credentials and user details.

3. Input Validation:

Justification:

Prevention of SQL Injection and XSS Attacks:

Input validation techniques are implemented to sanitize and validate user inputs, preventing common security vulnerabilities such as SQL injection and cross-site scripting (XSS).

Data Integrity:

Validating user inputs ensures data integrity and protects the system from malicious input that could compromise the database.

4. Output Encoding:

Justification:

Prevention of Cross-Site Scripting (XSS):

Output encoding is applied to all user-generated content displayed in the web interface. This prevents XSS attacks by ensuring that user input is treated as plain text rather than executable code.

5. Password Security:

Justification:

Hashing and Salting:

User passwords are securely stored using strong hashing algorithms and unique salts. This ensures that even if the database is compromised, passwords remain secure.

Password Complexity Requirements:

Users are required to create strong passwords with a combination of letters, numbers, and special characters to enhance password security.

6. Secure Database Practices:

Justification:

MySQL Injection Prevention:

Prepared statements and parameterized queries are implemented to prevent MySQL injection attacks, ensuring that user inputs are not executed as SQL commands.

Regular Database Audits:

Regular audits of the database are conducted to identify and address potential vulnerabilities. This includes reviewing user permissions, checking for unused accounts, and monitoring database activity.

7. Error Handling and Logging:

Justification:

Detection and Mitigation of Issues:

Comprehensive error handling is implemented to detect and log errors. This information is valuable for identifying and mitigating issues promptly.

Security Incident Response:

Detailed logging of security-related events enables the development team to respond effectively to security incidents, track potential breaches, and implement necessary security measures.

8. Securing Admin Pages:

Justification:

Access Control Lists (ACLs):

Access Control Lists are employed to secure admin pages, ensuring that only authorized users (librarians) have access. This prevents ordinary users from accessing sensitive administrative functionalities.

URL Whitelisting:

URL whitelisting is implemented to restrict access to admin pages only to users with the appropriate permissions. This helps in preventing direct access by unauthorized users.

9. Regular Security Audits and Updates:

Justification:

Proactive Security Measures:

Regular security audits are conducted to identify and address potential vulnerabilities before they can be exploited.

System and Software Updates:

The system and all software components are kept up-to-date with the latest security patches and updates. This ensures that known vulnerabilities are patched promptly.

Conclusion:

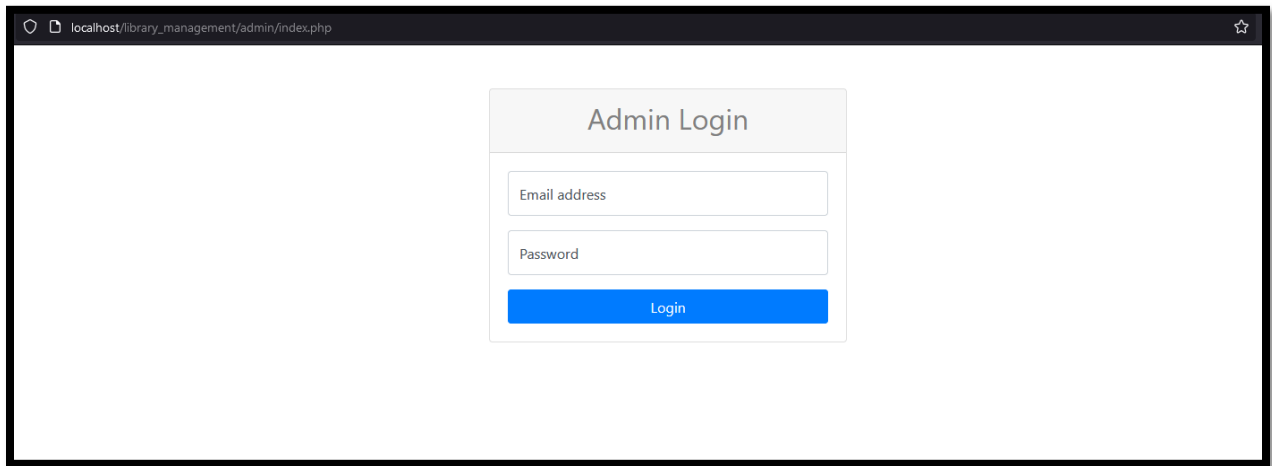
Implementing these security measures ensures that the library management system is fortified against a range of potential threats, providing a robust and secure environment for managing books and user information. Regular monitoring, updates, and proactive security practices contribute to the overall resilience of the system against evolving security challenges.

Testing

Show the testing process and result from the designated possible input.

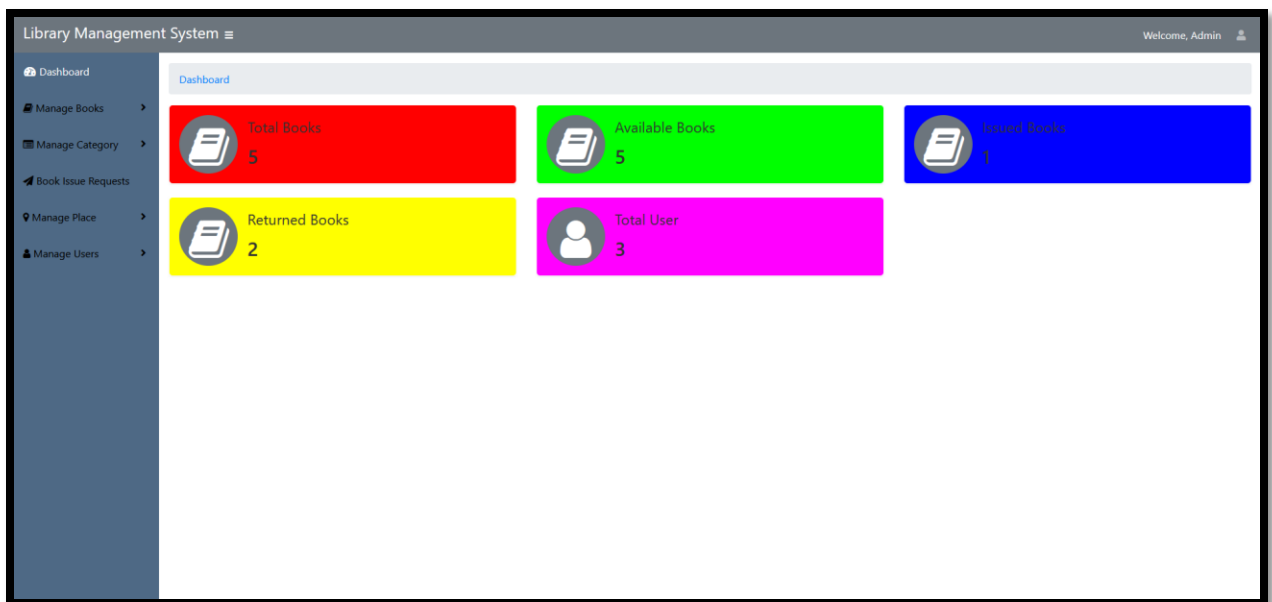
Admin Part

Login Page

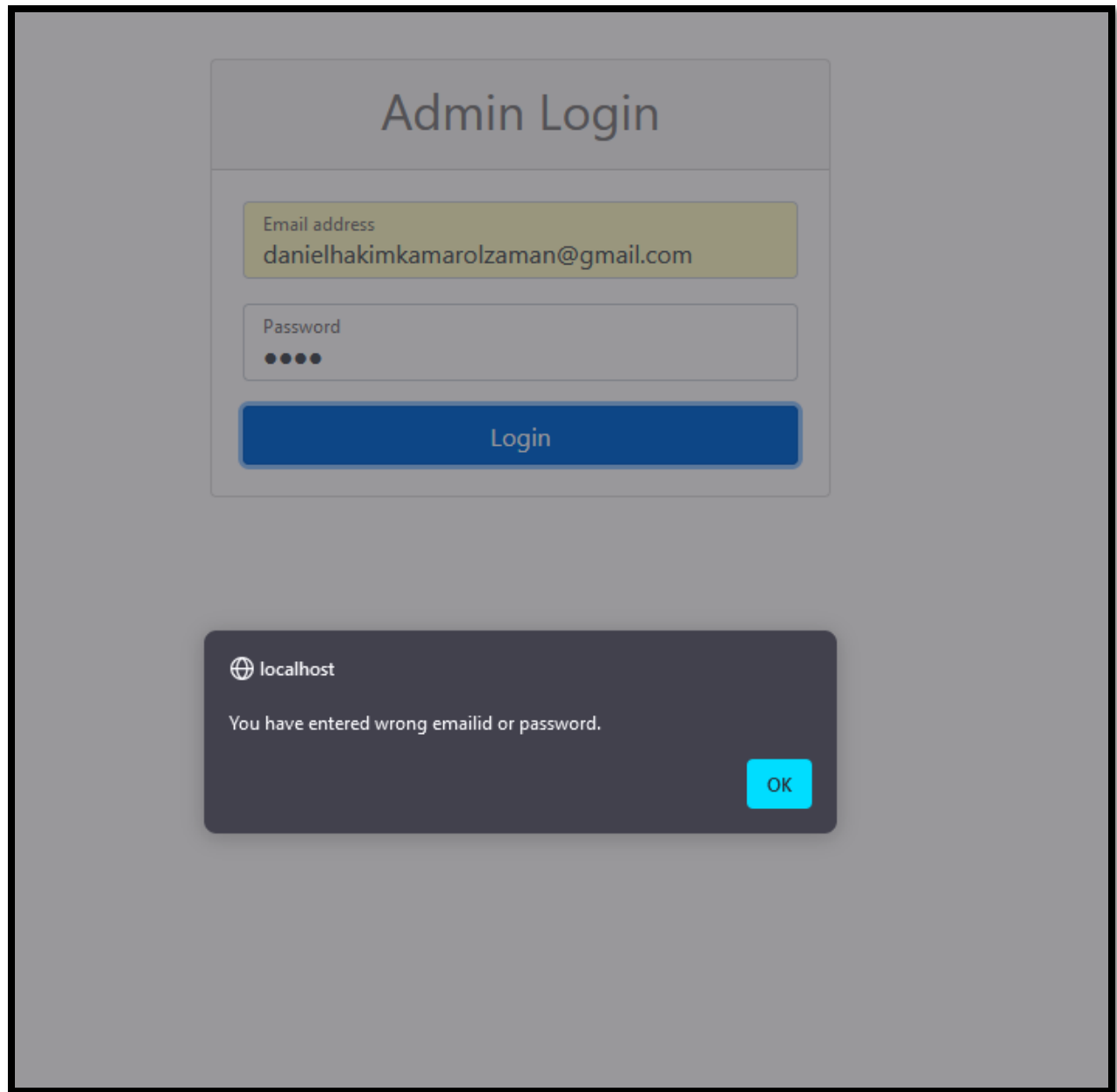


The screenshot shows a web browser window with the address bar displaying 'localhost/library_management/admin/index.php'. The main content area features a centered 'Admin Login' form. The form has a title 'Admin Login' at the top, followed by two input fields: 'Email address' and 'Password'. Below these fields is a blue 'Login' button.

Admin Login is the login just for admin only to prevent any unauthorized user to access the page.



After insert the **correct** credentials, user will go to dashboard page.



If the credentials are wrong, a message will come out to tell the user that the credentials inserted are wrong.

Dashboard Page

Library Management System Welcome, Admin

[Dashboard](#)

[Manage Books](#)

[Manage Category](#)

[Book Issue Requests](#)

[Manage Place](#)

[Manage Users](#)

[Add Book](#)

Submit Book Details

Book Name *

Category *

ISBN *

Author *

Publisher *

Price *

Quantity *

Location *

Availability *

[Submit](#)

Admin can add book into the system

Library Management System Welcome, Admin

[Dashboard](#)

[Manage Books](#)

[Manage Category](#)

[Book Issue Requests](#)

[Manage Place](#)

[Manage Users](#)

[View Book](#)

View Book Details

Show entries

Search:

S.No.	Name	Category	ISBN	Author	Publisher	Price	Quantity	Place	Status	Action
1	The Great Gatsby	Classic	978-0-7432-7356-5	F. Scott Fitzgerald	Scribner	10.99	3	A1	Available	Edit Delete
2	To Kill a Mockingbird	Fiction	978-0-06-112008-4	Harper Lee	Harper Perennial Modern Classics	14.99	5	A2	Available	Edit Delete
3	1984	Fiction	978-0-452-28423-4	George Orwell	Signet Classics	9.99	5	A1	Available	Edit Delete
4	test	Fantasy	321312321321	J.D. Salinger	test	13.99	5	A3	Available	Edit Delete
5	The Catcher in the Rye	Fiction	978-0-316-76948-0	J.D. Salinger	Little, Brown and Company	13.99	5	A3	Available	Edit Delete

Showing 1 to 5 of 5 entries

[Previous](#) [1](#) [Next](#)

Admin can view the book details specifically

Library Management System ⌵

Welcome, Admin 👤

Dashboard
Manage Books
Manage Category
Book Issue Requests
Manage Place
Manage Users

Add Category

Submit Details

Category Name *

Status *

[Submit](#)

Admin can add category into the system

Library Management System ⌵

Welcome, Admin 👤

Dashboard
Manage Books
Manage Category
Book Issue Requests
Manage Place
Manage Users

View Category

View Details

Show entries Search:

S.No.	Category Name	Status	Action
1	Fiction	Active	Edit Delete
2	Fantasy	Active	Edit Delete
3	Classic	Active	Edit Delete

Showing 1 to 3 of 3 entries Previous **1** Next

Admin can view the category details specifically

Library Management System ⌵

Welcome, Admin 👤

Dashboard
Manage Books
Manage Category
Book Issue Requests
Manage Place
Manage Users

View Book Issue Requests

Issue Requests

Show entries Search:

S.No.	Book Name	User Name	Quantity	Action
1	The Great Gatsby	Daniel	3	Book Issued

Showing 1 to 1 of 1 entries Previous **1** Next

Admin can view the book issue request from any user

Library Management System ≡ Welcome, Admin

[Dashboard](#)

[Add Place](#)

Submit Details

Place Name *

Status *

[Submit](#)

Admin can add place into the system

Library Management System ≡ Welcome, Admin

[Dashboard](#)

[View Place](#)

View Details

Show entries Search:

S.No.	Name	Status	Action
1	A1	Active	Edit Delete
2	A2	Active	Edit Delete
3	A3	Active	Edit Delete

Showing 1 to 3 of 3 entries Previous **1** Next

Admin can view the place details specifically

Library Management System ≡ Welcome, Admin

[Dashboard](#)

[Add User](#)

Submit Details

User Name *

Emailid *

Password *

Role *

Status *

[Submit](#)

Admin can add user into the system

Library Management System Welcome, Admin

Dashboard

Manage Books

Manage Category

Book Issue Requests

Manage Place

Manage Users

View Users

View Details

Show 10 entries

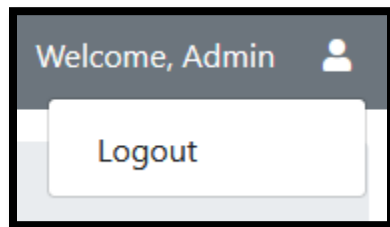
Search:

S.No.	User Name	Emailid	Role	Status	Action
1	Admin	admin@library.com	Admin	Active	Edit Delete
2	Daniel	danielhakimkamarolzaman@gmail.com	User	Active	Edit Delete
3	Sir	sir@user.com	User	Active	Edit Delete
4	madam	madam@user.com	User	Active	Edit Delete

Showing 1 to 4 of 4 entries

Previous 1 Next

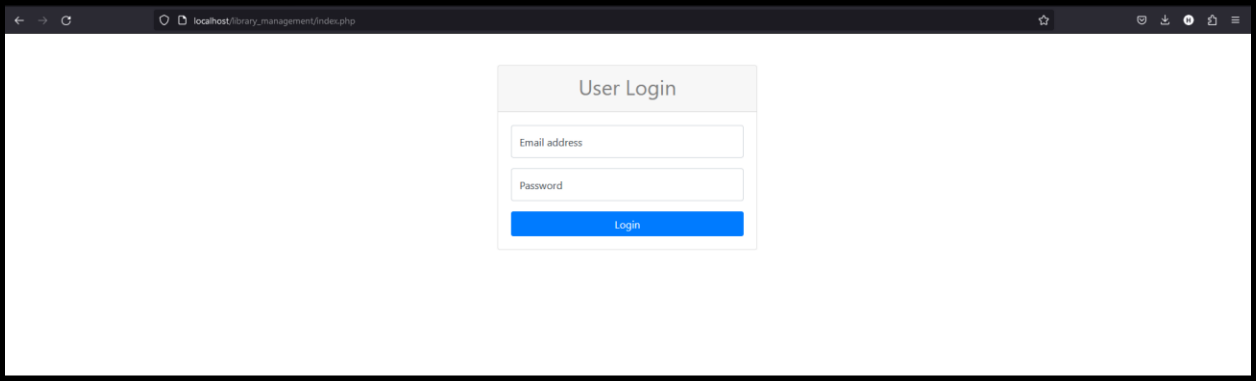
Admin can view all the users details specifically



Admin can logout to prevent any unauthorized access

User Part

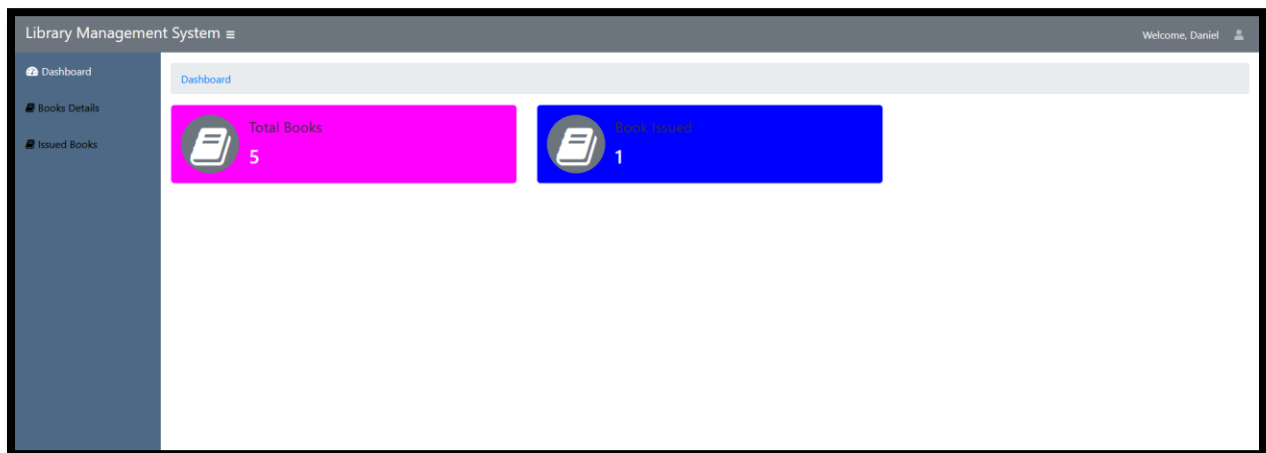
User Login Page



A screenshot of a web browser displaying a "User Login" form. The browser's address bar shows "localhost/library_management/index.php". The form is centered on the page and contains the following elements:

- A title "User Login" in a light gray box.
- An input field labeled "Email address".
- An input field labeled "Password".
- A blue button labeled "Login".

User Login is the login just for user only to prevent any unauthorized user to access the page.



Once login, the user will only see this information on Dashboard

Dashboard Page

Library Management System

Welcome, Daniel

Dashboard

Books Details

Issued Books

View Book

View Book Details

Show 10 entries

Search:

S.No.	Name	Category	Availability	Action
1	The Great Gatsby	Classic	Available	Issue
2	To Kill a Mockingbird	Fiction	Available	Issue
3	1984	Fiction	Available	Issue
4	test	Fantasy	Available	Issue
5	The Catcher in the Rye	Fiction	Available	Issue

Showing 1 to 5 of 5 entries

Previous 1 Next

User can view the book details but necessary part only.

Library Management System

Welcome, Daniel

Dashboard

Books Details

Issued Books

View Book

View Book Details

Show 10 entries

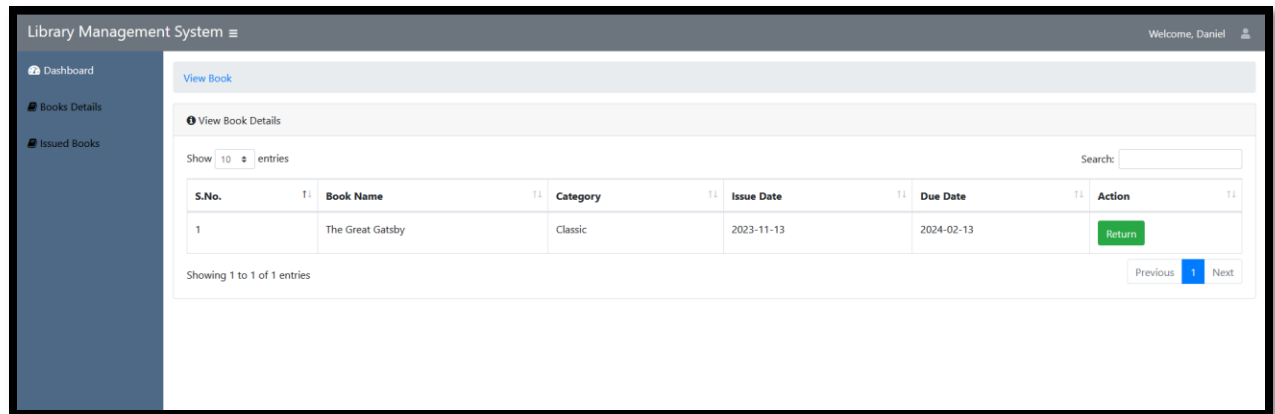
Search:

S.No.	Name	Category	Availability	Action
1	The Great Gatsby	Classic	Available	Issue
2	To Kill a Mockingbird	Fiction	Available	Issue
3	1984	Fiction	Available	Issue
4	test	Fantasy	Available	Request Sent
5	The Catcher in the Rye	Fiction	Available	Issue

Showing 1 to 5 of 5 entries

Previous 1 Next

If user want to borrow the book, user can click Issue button and it will change into Request Sent meaning that the request has been sent to admin.



User can view the book details with issued book that been sent and been accepted by admin.

Security Measure

	id	user_name	emailid	password	role	status	created_at
					1=admin, 2=user	1=active, 0=inactive	
<input type="checkbox"/> Edit <input type="copy"/> Copy <input type="delete"/> Delete	1	Admin	admin@library.com	6b8d5de3b53751bac499eb1bef762a2a	1	1	2023-05-27 21:29:44
<input type="checkbox"/> Edit <input type="copy"/> Copy <input type="delete"/> Delete	2	Daniel	danielhakimkamarolzaman@gmail.com	202cb962ac59075b964b07152d234b70	2	1	2023-11-13 14:05:33
<input type="checkbox"/> Edit <input type="copy"/> Copy <input type="delete"/> Delete	3	Sir	sir@user.com	202cb962ac59075b964b07152d234b70	2	1	2023-11-13 14:52:47
<input type="checkbox"/> Edit <input type="copy"/> Copy <input type="delete"/> Delete	4	madam	madam@user.com	202cb962ac59075b964b07152d234b70	2	1	2023-11-13 14:53:05

Once the admin created an account for user, the details will show as above. Since this project need to implement security measures the password that had been set by admin will change into hash password. It means more secure and protect the details of any users and can prevent any unauthorized person gain the details.

Reflective Conclusion

In assessing the overall secure coding implementation on a web system, it's essential to consider the specific measures taken to address common security threats and vulnerabilities. Based on the information provided earlier, here's a technical opinion and comments on the secure coding implementation:

Positive Aspects:

Role-Based Access Control (RBAC):

Implementing RBAC is a strong security practice, ensuring that users only have access to the functionalities relevant to their roles. This minimizes the risk of unauthorized access and actions.

SSL/TLS Encryption:

Using SSL/TLS encryption for secure communication is crucial for protecting data in transit. This ensures that sensitive information, such as login credentials, is encrypted, reducing the risk of eavesdropping.

Input Validation:

Implementing input validation is a fundamental security practice to prevent common vulnerabilities like SQL injection and XSS attacks. Validating and sanitizing user inputs contribute to data integrity and user safety.

Password Security:

The use of strong hashing algorithms and unique salts for password storage is commendable. It enhances password security and protects user credentials even if the database is compromised.

Secure Database Practices:

Employing prepared statements and parameterized queries is crucial for preventing SQL injection attacks. Regular database audits contribute to maintaining a secure database environment.

Secure Communication with HTTPS:

The adoption of HTTPS ensures secure communication between clients and servers. This protects against various security threats, including man-in-the-middle attacks.

Suggestions for Improvement:**Error Handling and Logging:**

While comprehensive error handling is in place, it's important to ensure that error messages revealed to users are generic and do not disclose sensitive information. Log entries should be detailed enough for debugging but without exposing critical data.

Session Management:

Ensure that session management is robust, including mechanisms to prevent session fixation, session hijacking, and session timeout policies. Implementing secure session management is crucial for maintaining user authentication.

Regular Security Audits:

Explicitly mention the frequency and methodology of security audits. Regular audits should include code reviews, penetration testing, and vulnerability assessments to proactively identify and address potential security risks.

Security Headers:

Consider implementing security headers such as Content Security Policy (CSP), Strict-Transport-Security (HSTS), and X-Content-Type-Options to enhance browser security and protect against certain types of attacks.

Client-Side Security:

Emphasize the importance of secure coding practices on the client side, especially if the system includes client-side scripting (JavaScript). Implement measures to prevent Cross-Site Scripting (XSS) vulnerabilities.

Authentication Best Practices:

Implement multi-factor authentication (MFA) if possible. Additionally, consider periodic reviews of authentication mechanisms to ensure they remain resilient against evolving threats.

User Account Management:

Provide functionality for users to change their passwords and consider implementing account lockout mechanisms to mitigate brute-force attacks.

Conclusion

The overall secure coding implementation appears to be robust, with several positive aspects such as RBAC, encryption, and password security. However, continuous improvement and vigilance are essential to stay ahead of emerging security threats. Regular security audits, updates, and adherence to best practices will contribute to a strong and resilient web system that prioritizes user data protection and system integrity.

References

- Anurag. (2023, October 30). Library Management System | Library management software in India. SkoolBeep. <https://www.skoolbeep.com/blog/library-management-system/>
- IndiaStudyChannel. (n.d.). UML Diagrams For The Case Studies Library Management System And Online Mobile Recharge. Retrieved from <https://www.indiastudychannel.com/resources/150271-UML-Diagrams-For-The-Case-Studies-Library-Management-System-And-Online-Mobile-Recharge.aspx>
- GeeksforGeeks. (n.d.). Use Case Diagram for Library Management System. Retrieved from <https://www.geeksforgeeks.org/use-case-diagram-for-library-management-system/>
- Creately. (n.d.). Library Management System - Context Diagram. Retrieved from <https://creately.com/diagram/example/ingwfn842/library-management-system-context-diagram>
- Algorithms. (n.d.). 4 Key Data Security Features in Library Management System- iSLIM. Retrieved from <https://slimkm.com/blog/4-key-data-security-features-in-library-management-system-islim/>

Appendix

https://github.com/Slushiee26/SWC_3503.git