Course Title: Web Application Development (CYB325)

M9

Group Name: Group 4

Students:

محمد عادل الصبحي 4400777

عمار مسلم الجهني 4406104

عمار مسيعيد الرفيعي 4400537

فهد ناجي الحربي 4401448

يامن غسان حسين 4405521

Section 1: Problem Domain

1.1 Authentication and Access Control:

- The authentication module is designed to ensure security by validating user identities and managing access. It also ensures that the website differentiate between regular students and administrators.

  Key challenges solved:

  1- Secure password storage by using hashes.

  2- Access control by implementing Role-Based Access Control to ensure that unauthorized users can't access restricted pages.

  3- Session security by using secure session flags.

  4- Overall security by following OWASP guidelines against common web application attacks.

1.2 The Admin Module:

- is responsible for managing the entire book system, including adding books, editing existing books, deleting books, and handling user suggestions. The main problem addressed in this module is maintaining data integrity, preventing unauthorized modification, and ensuring that the library data remains clean and up-to-date.

  Key challenges solved:

  1- Ensuring only Admin users can access book management features using Role-Based Access Control (RBAC).

  2-Validating all inputs before inserting or updating book records.

  3- Preventing duplicate or stuck suggestions by showing only pending ones.

  4- Automatically removing related loan records when a book is deleted using SQL ON DELETE CASCADE.

  5- Organizing admin tasks in a clean dashboard interface.

1.3 The main page:

- Users need a homepage for a digital library that allows them to browse all books, search by title or author, and view book details. The system must load data from the database securely, show available books, and handle errors without exposing system information.

  Key Problems Solved:

  1. Safe searching: Uses secure queries so people can search by title or author without risking SQL injection.
  2. Auto-loading books: Pulls all books from the database and shows them instantly.
  3. Easy-to-browse layout: Displays books in clean, simple cards with covers and details.
  4. Clear availability: Shows whether each book is “available” or “borrowed.”
  5. Error protection: Logs real errors privately while showing safe messages to the user.
  6. Quick navigation: Each book has a button that takes you to its details page.

1.4 The loans module:

addresses the need for users to borrow and return books while maintaining accurate book availability status across the system. The problem domain behind this module is ensuring that a borrowed book cannot be loaned again until it is returned and preventing lost or inconsistent loan records.

Key challenges solved:

1. Maintaining availability status of each book by updating the database when a loan is created or returned.

2. Preventing a user from borrowing the same book multiple times at once.
3. Blocking borrowing attempts when a book is already marked as "borrowed."
4. Automatically recording loan dates and return dates for tracking purposes and preventing overdue misuse.
5. Providing a personalized dashboard where users can view only their own borrowed books securely through session-based access control.

This module improves the user experience by supporting self-service borrowing and returning while ensuring database integrity and preventing unauthorized interactions with loan records.

1.5 Suggestions Module:

The suggestions module enables users to request the addition of new books to the digital library. It solves the problem of determining which new titles students need without requiring direct communication with the administrator.

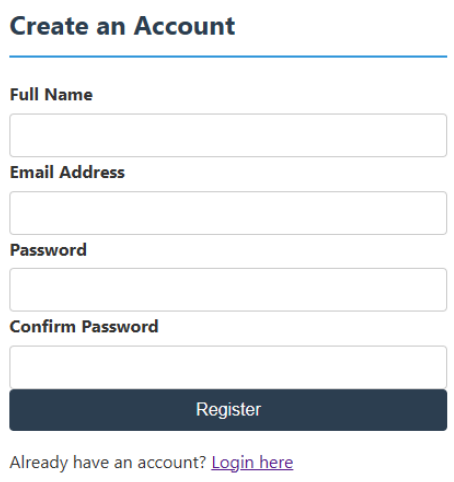
This feature provides:

1. A suggestion form accessible only to authenticated users.
2. Secure storage of suggestions in the database with predefined statuses (pending, approved, rejected).
3. An admin-only inbox where requests can be reviewed and decided upon.
4. Role-based access control preventing students from viewing or modifying suggestion decisions.

Overall, the module enhances library content by collecting user-driven recommendations while maintaining full administrative control.

Section 2: Main Project Interfaces

2.1 Registration Page:



Create an Account

Full Name

Email Address

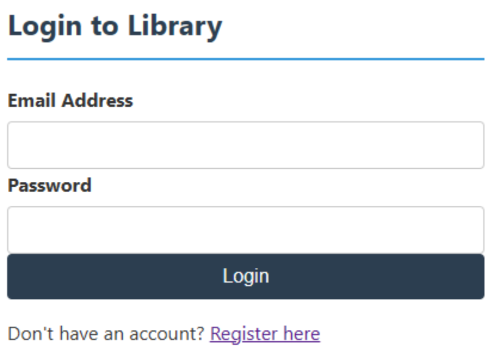
Password

Confirm Password

Register

Already have an account? Login here

2.2 Login Page:



Login to Library

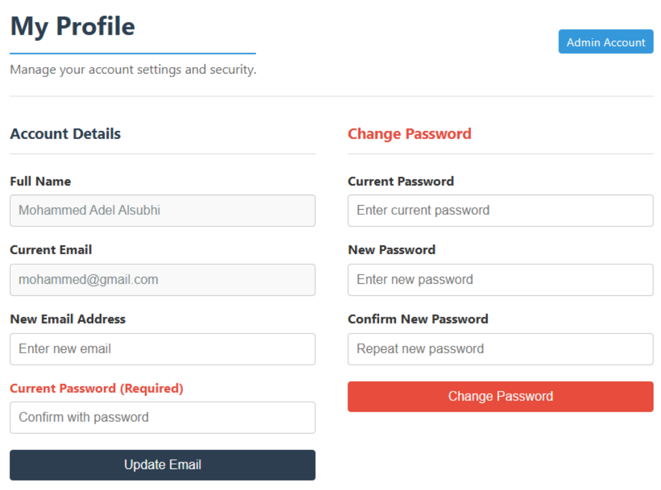
Email Address

Password

Login

Don't have an account? Register here

2.3 Profile Page:



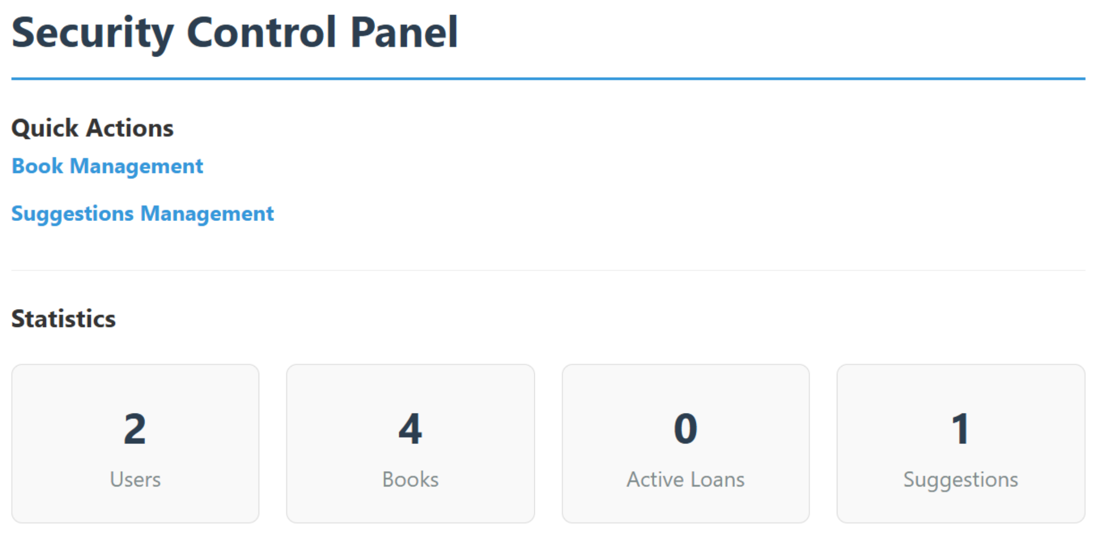
My Profile

Admin Account

Manage your account settings and security.

Account Details	Change Password
Full Name: Mohammed Adel Alsubhi	Current Password: Enter current password
Current Email: mohammed@gmail.com	New Password: Enter new password
New Email Address: Enter new email	Confirm New Password: Repeat new password
Current Password (Required): Confirm with password	Change Password
Update Email	

2.4 Admin Security Control Page



Security Control Panel

Quick Actions

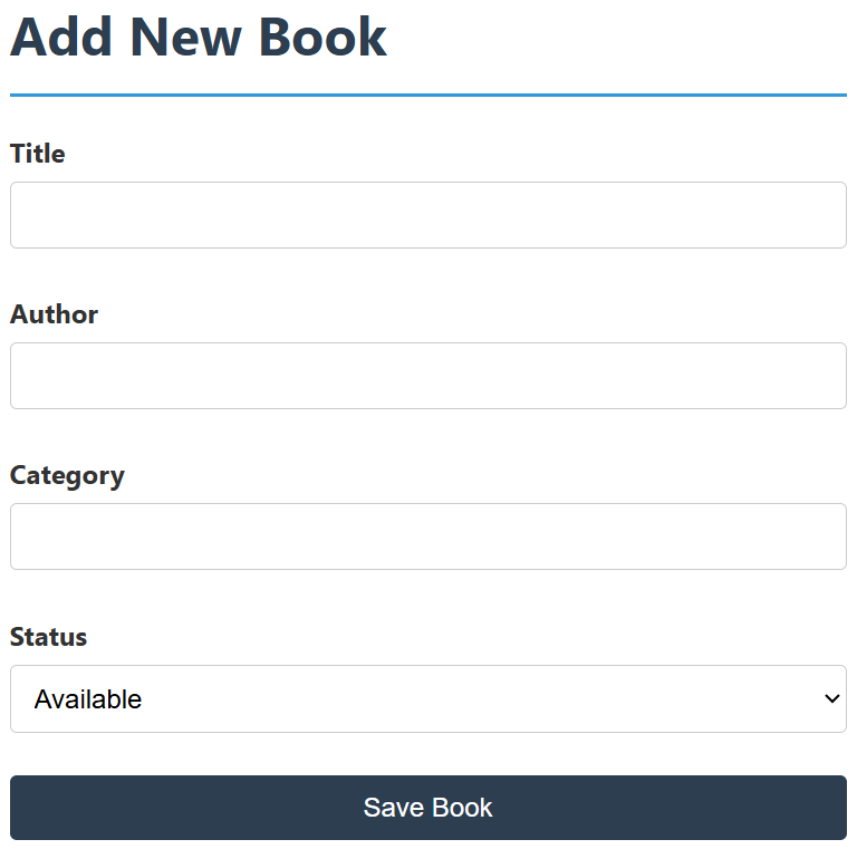
Book Management

Suggestions Management

Statistics

2	4	0	1
Users	Books	Active Loans	Suggestions

2.5 Add New Book



Add New Book

Title

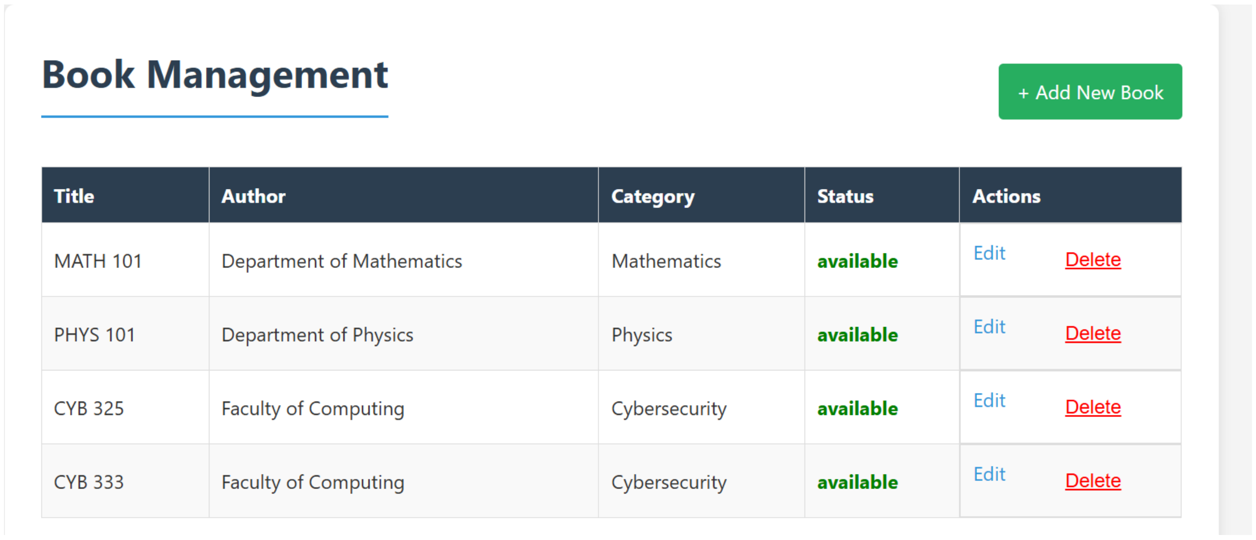
Author

Category

Status

Available

2.6 Edit and delete Book

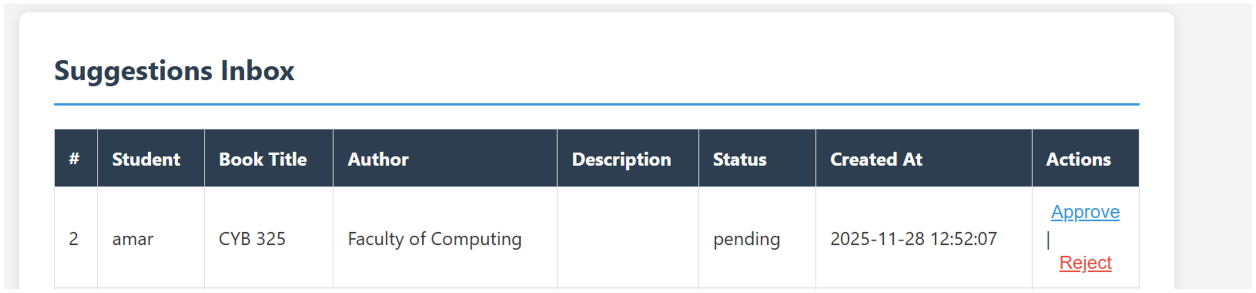


Book Management

+ Add New Book

Title	Author	Category	Status	Actions
MATH 101	Department of Mathematics	Mathematics	available	Edit Delete
PHYS 101	Department of Physics	Physics	available	Edit Delete
CYB 325	Faculty of Computing	Cybersecurity	available	Edit Delete
CYB 333	Faculty of Computing	Cybersecurity	available	Edit Delete

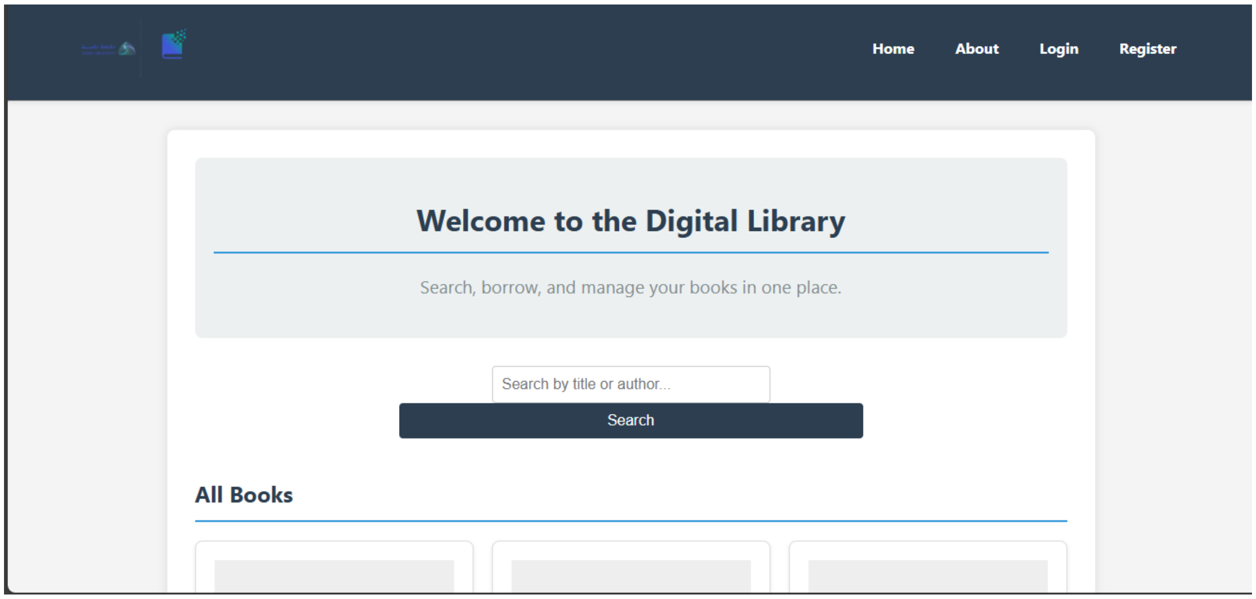
Suggestions from the Users



Suggestions Inbox

#	Student	Book Title	Author	Description	Status	Created At	Actions
2	amar	CYB 325	Faculty of Computing		pending	2025-11-28 12:52:07	Approve | Reject

2.7 Main page

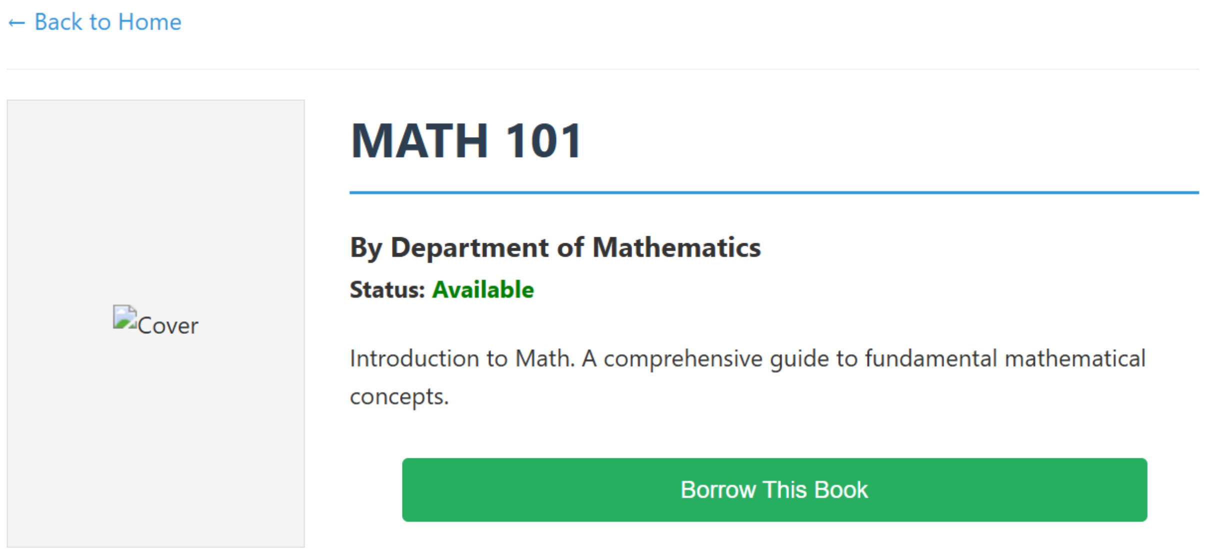


Welcome to the Digital Library

Search, borrow, and manage your books in one place.

All Books

2.8 Borrow Book Page



← Back to Home

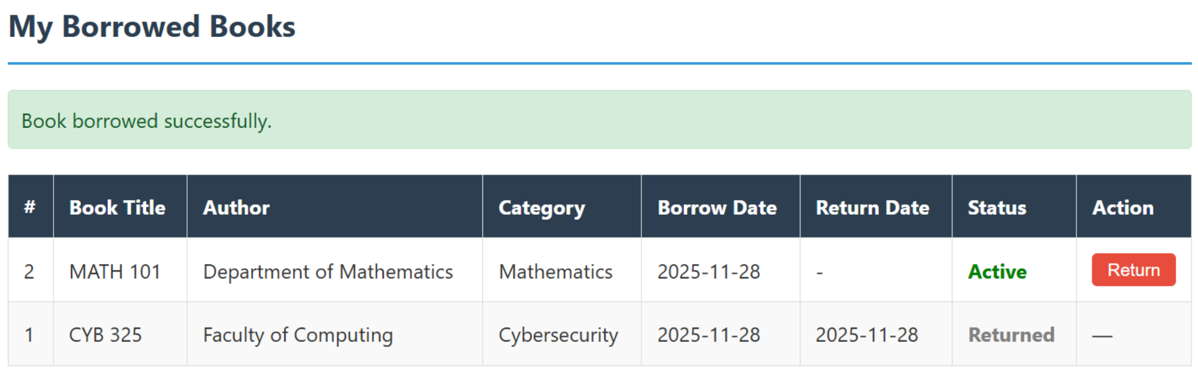
MATH 101

By Department of Mathematics

Status: Available

Introduction to Math. A comprehensive guide to fundamental mathematical concepts.

2.9 Returning a Borrowed Item

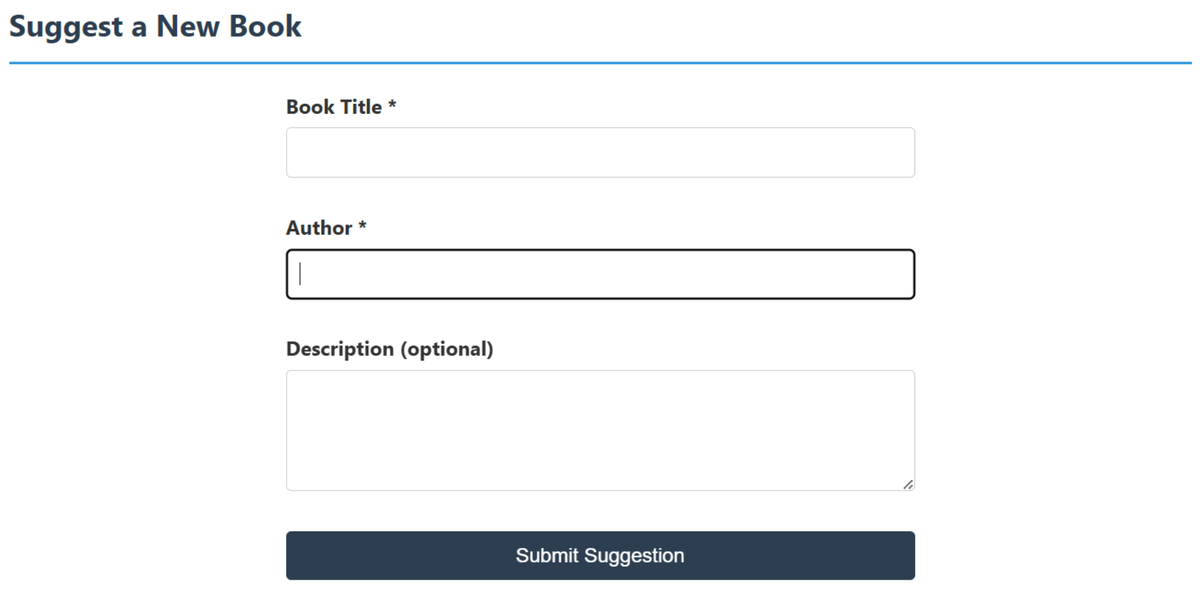


My Borrowed Books

Book borrowed successfully.

#	Book Title	Author	Category	Borrow Date	Return Date	Status	Action
2	MATH 101	Department of Mathematics	Mathematics	2025-11-28	-	**Active**	Return
1	CYB 325	Faculty of Computing	Cybersecurity	2025-11-28	2025-11-28	**Returned**	—

2.10 Suggest Book Page



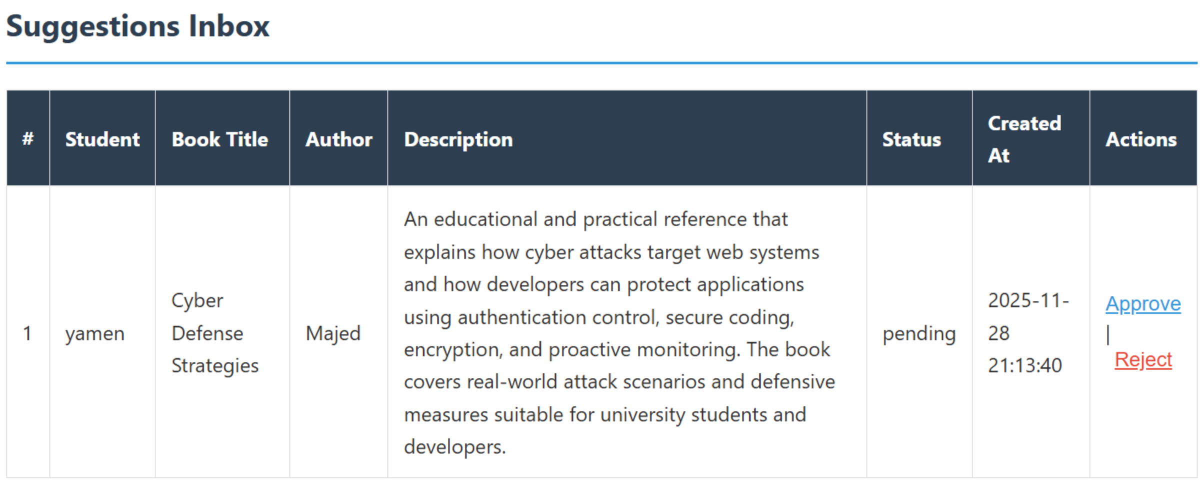
Suggest a New Book

Book Title *

Author *

Description (optional)

2.11 Review Suggestions Page (Admin Panel)

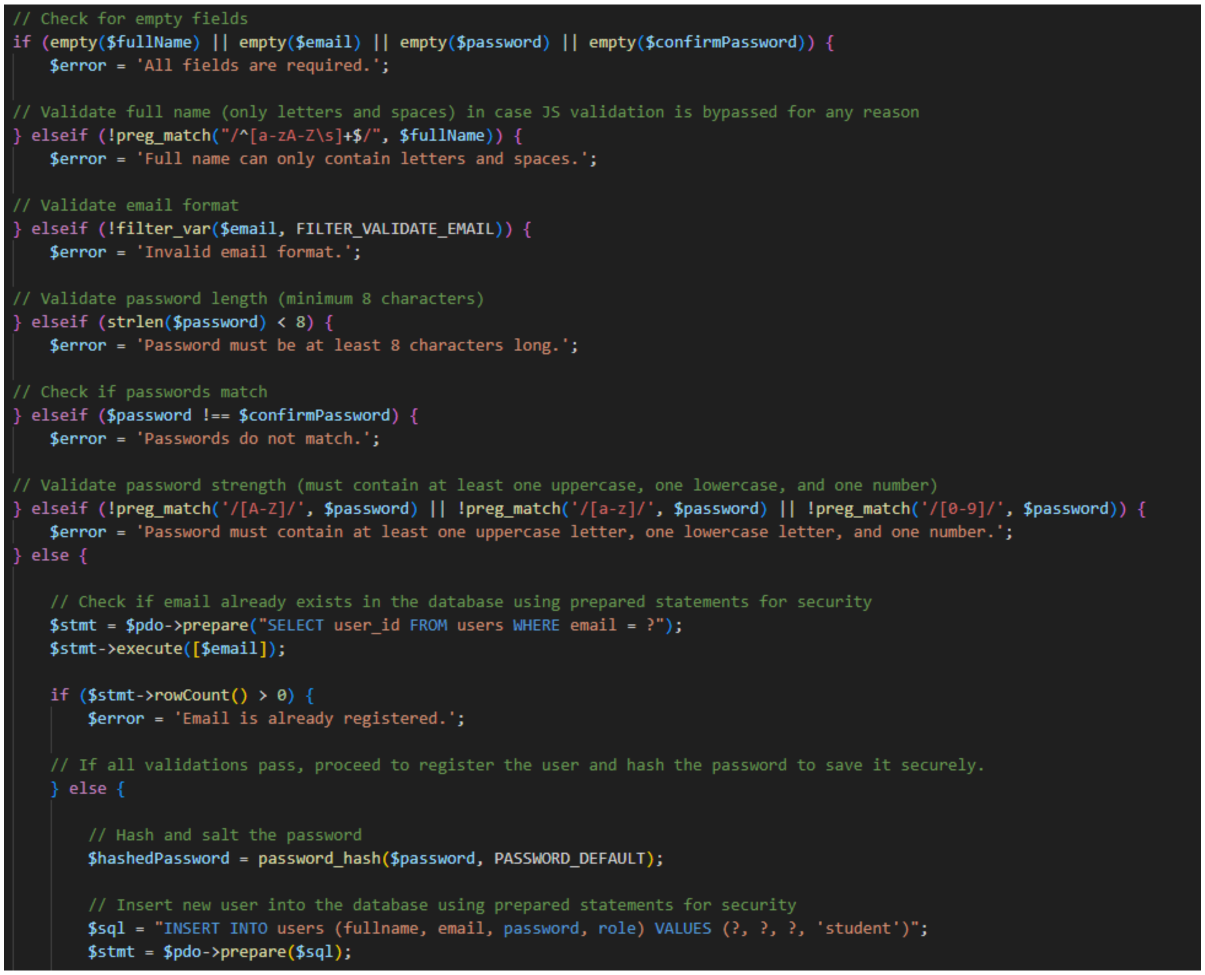


Suggestions Inbox

#	Student	Book Title	Author	Description	Status	Created At	Actions
1	yamen	Cyber Defense Strategies	Majed	An educational and practical reference that explains how cyber attacks target web systems and how developers can protect applications using authentication control, secure coding, encryption, and proactive monitoring. The book covers real-world attack scenarios and defensive measures suitable for university students and developers.	pending	2025-11-28 21:13:40	Approve | Reject

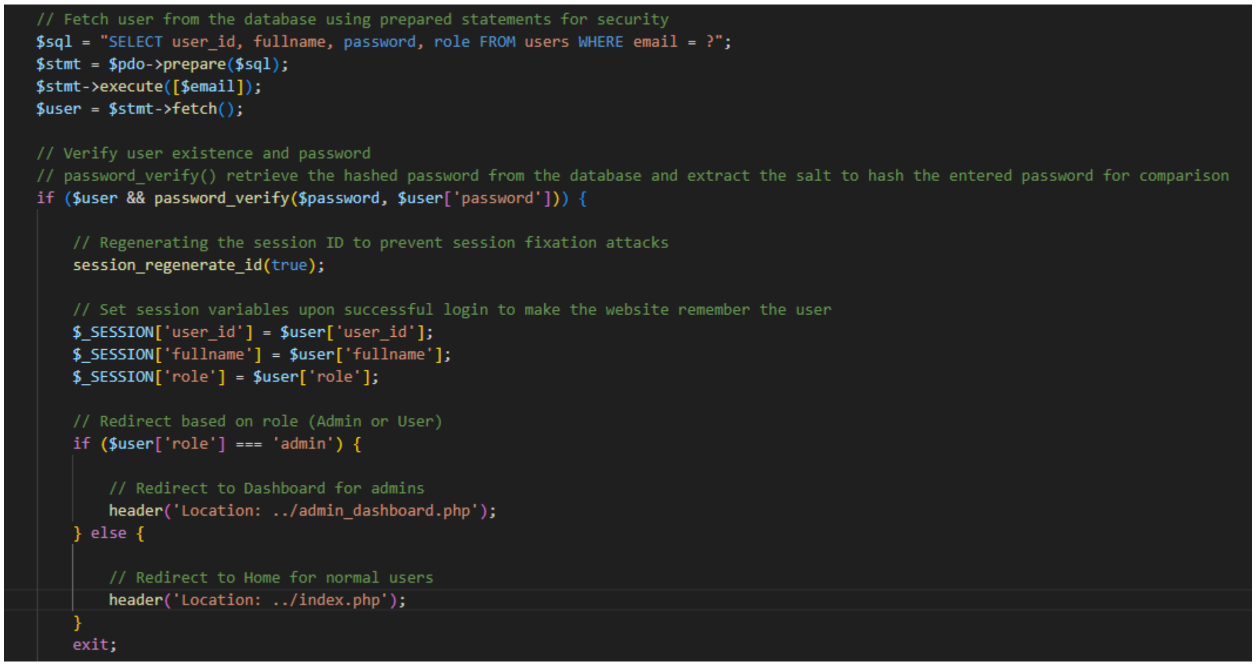
Section 3: Representative Code Excerpts

3.1 register.php: server-side validation, registration handling, and hashing.



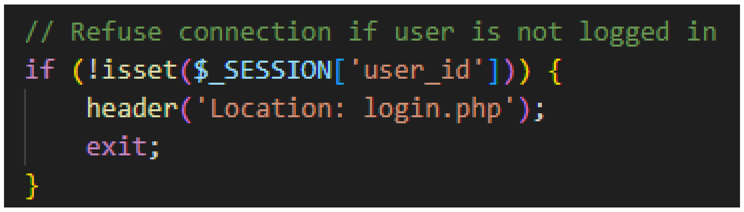
```php
// Check for empty fields
if (empty($fullName) || empty($email) || empty($password) || empty($confirmPassword)) {
    $error = 'All fields are required.';

// Validate full name (only letters and spaces) in case JS validation is bypassed for any reason
} elseif (!preg_match("/^[a-zA-Z\s]+$/", $fullName)) {
    $error = 'Full name can only contain letters and spaces.';

// Validate email format
} elseif (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
    $error = 'Invalid email format.';

// Validate password length (minimum 8 characters)
} elseif (strlen($password) < 8) {
    $error = 'Password must be at least 8 characters long.';

// Check if passwords match
} elseif ($password !== $confirmPassword) {
    $error = 'Passwords do not match.';

// Validate password strength (must contain at least one uppercase, one lowercase, and one number)
} elseif (!preg_match('/[A-Z]/', $password) || !preg_match('/[a-z]/', $password) || !preg_match('/[0-9]/', $password)) {
    $error = 'Password must contain at least one uppercase letter, one lowercase letter, and one number.';
} else {

    // Check if email already exists in the database using prepared statements for security
    $stmt = $pdo->prepare("SELECT user_id FROM users WHERE email = ?");
    $stmt->execute([$email]);

    if ($stmt->rowCount() > 0) {
        $error = 'Email is already registered.';

    // If all validations pass, proceed to register the user and hash the password to save it securely.
    } else {

        // Hash and salt the password
        $hashedPassword = password_hash($password, PASSWORD_DEFAULT);

        // Insert new user into the database using prepared statements for security
        $sql = "INSERT INTO users (fullname, email, password, role) VALUES (?, ?, ?, 'student')";
        $stmt = $pdo->prepare($sql);
```

3.2 **login.php: Database operations, input validation, and handling success case**

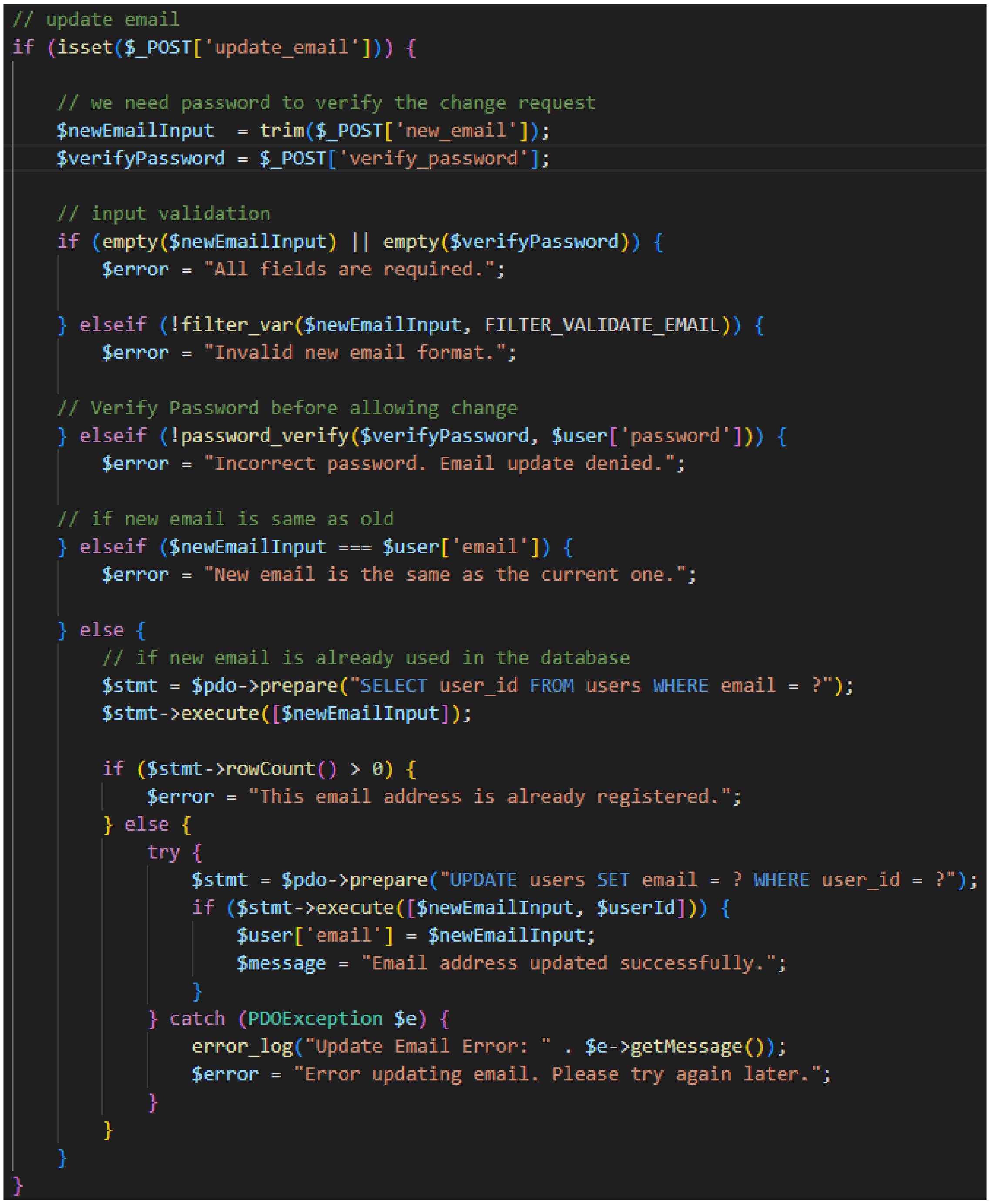


```php
    // Fetch user from the database using prepared statements for security
    $sql = "SELECT user_id, fullname, password, role FROM users WHERE email = ?";
    $stmt = $pdo->prepare($sql);
    $stmt->execute([$email]);
    $user = $stmt->fetch();

    // Verify user existence and password
    // password_verify() retrieve the hashed password from the database and extract the salt to hash the entered password for comparison
    if ($user && password_verify($password, $user['password'])) {

        // Regenerating the session ID to prevent session fixation attacks
        session_regenerate_id(true);

        // Set session variables upon successful login to make the website remember the user
        $_SESSION['user_id'] = $user['user_id'];
        $_SESSION['fullname'] = $user['fullname'];
        $_SESSION['role'] = $user['role'];

        // Redirect based on role (Admin or User)
        if ($user['role'] === 'admin') {

            // Redirect to Dashboard for admins
            header('Location: ../admin_dashboard.php');
        } else {

            // Redirect to Home for normal users
            header('Location: ../index.php');
        }
        exit;
```

3.3 profile.php: profile info update, and sessions security.

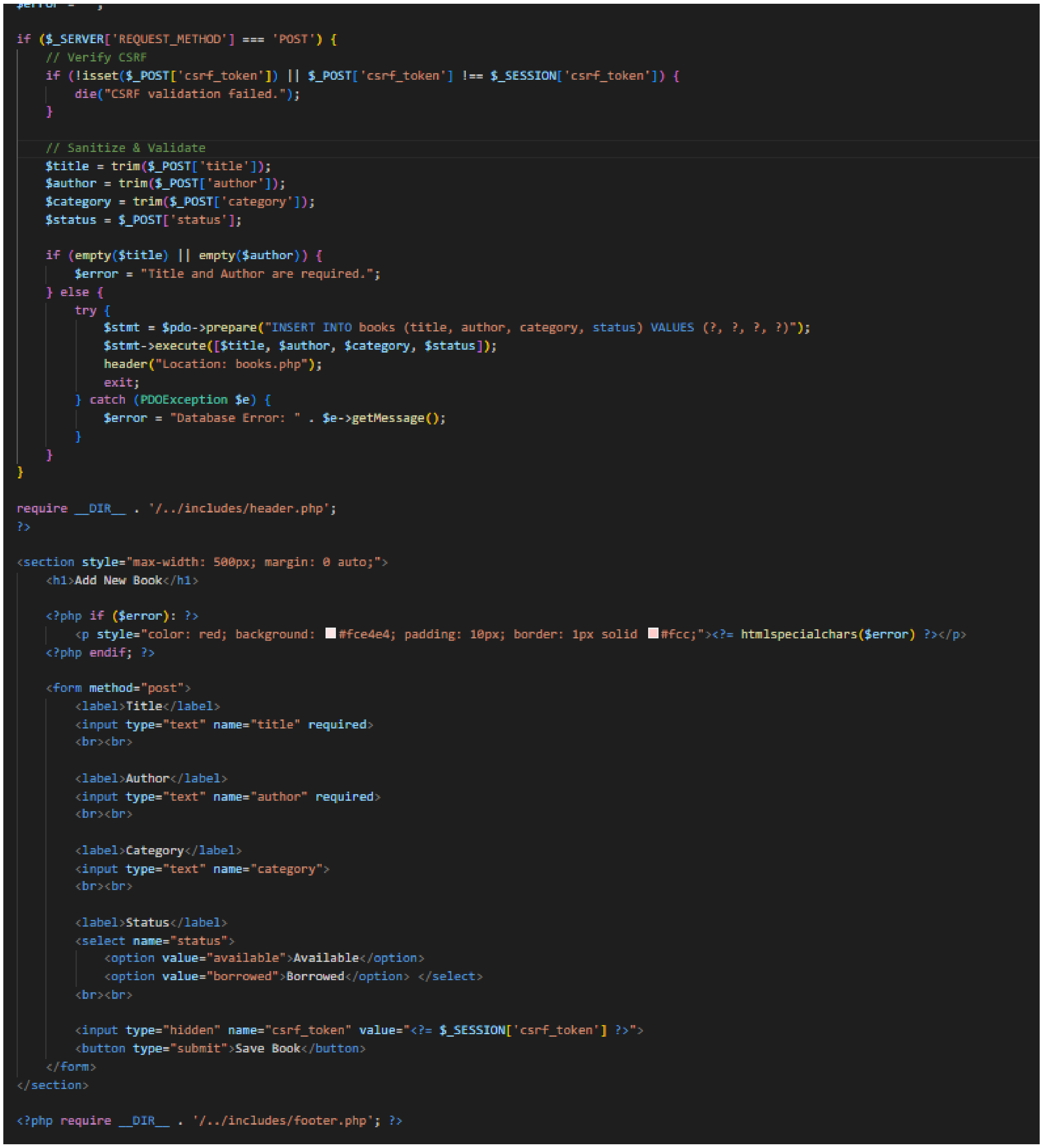


```php
// Refuse connection if user is not logged in
if (!isset($_SESSION['user_id'])) {
    header('Location: login.php');
    exit;
}
```



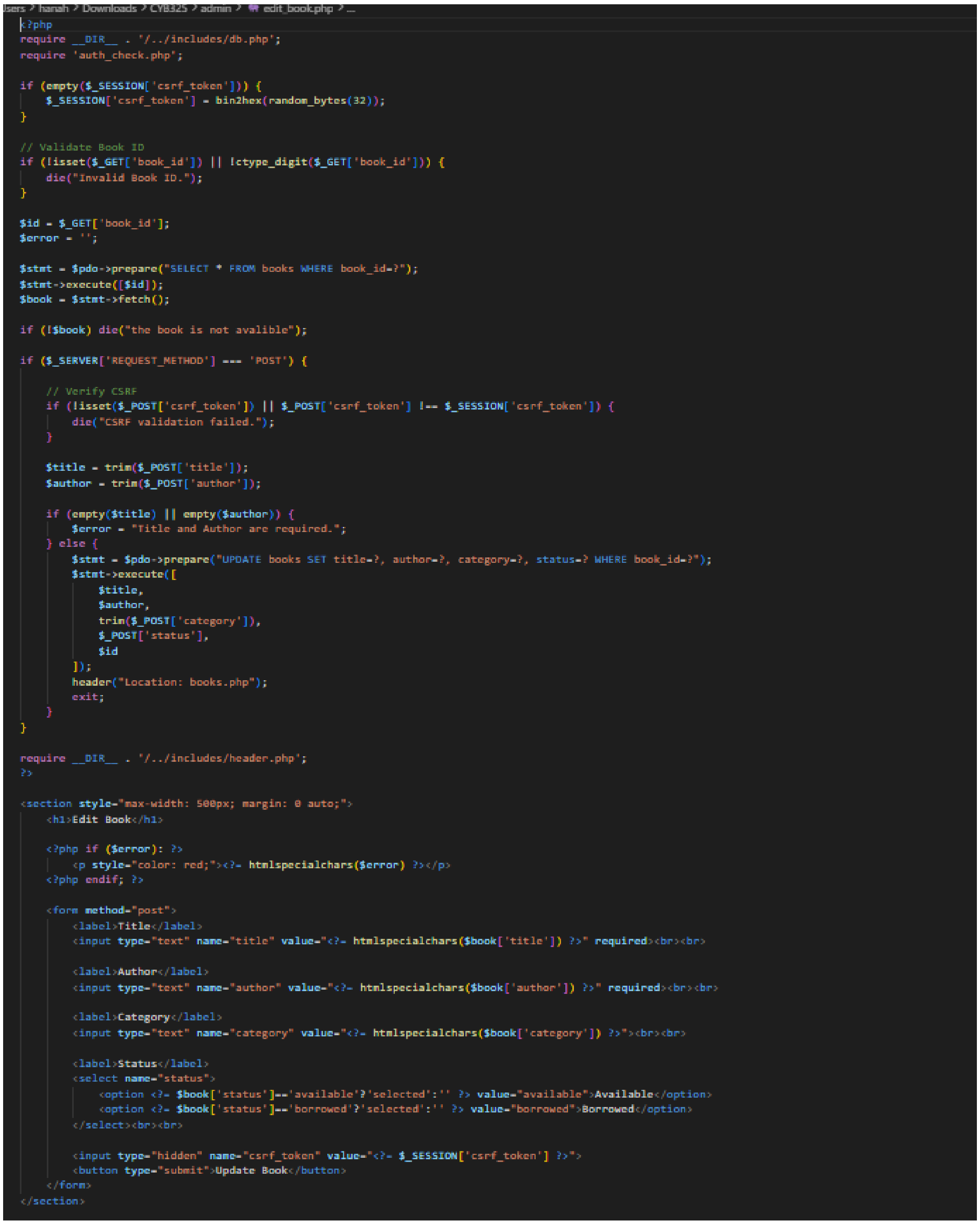
```php
// update email
if (isset($_POST['update_email'])) {

    // we need password to verify the change request
    $newEmailInput  = trim($_POST['new_email']);
    $verifyPassword = $_POST['verify_password'];

    // input validation
    if (empty($newEmailInput) || empty($verifyPassword)) {
        $error = "All fields are required.";

    } elseif (!filter_var($newEmailInput, FILTER_VALIDATE_EMAIL)) {
        $error = "Invalid new email format.";

    // Verify Password before allowing change
    } elseif (!password_verify($verifyPassword, $user['password'])) {
        $error = "Incorrect password. Email update denied.";

    // if new email is same as old
    } elseif ($newEmailInput === $user['email']) {
        $error = "New email is the same as the current one.";

    } else {
        // if new email is already used in the database
        $stmt = $pdo->prepare("SELECT user_id FROM users WHERE email = ?");
        $stmt->execute([$newEmailInput]);

        if ($stmt->rowCount() > 0) {
            $error = "This email address is already registered.";
        } else {
            try {
                $stmt = $pdo->prepare("UPDATE users SET email = ? WHERE user_id = ?");
                if ($stmt->execute([$newEmailInput, $userId])) {
                    $user['email'] = $newEmailInput;
                    $message = "Email address updated successfully.";
                }
            } catch (PDOException $e) {
                error_log("Update Email Error: " . $e->getMessage());
                $error = "Error updating email. Please try again later.";
            }
        }
    }
}
```

3.4 add_Book Validates input and insert book using prepared SQL statements



```php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Verify CSRF
    if (!isset($_POST['csrf_token']) || $_POST['csrf_token'] !== $_SESSION['csrf_token']) {
        die("CSRF validation failed.");
    }

    // Sanitize & Validate
    $title = trim($_POST['title']);
    $author = trim($_POST['author']);
    $category = trim($_POST['category']);
    $status = $_POST['status'];

    if (empty($title) || empty($author)) {
        $error = "Title and Author are required.";
    } else {
        try {
            $stmt = $pdo->prepare("INSERT INTO books (title, author, category, status) VALUES (?, ?, ?, ?)");
            $stmt->execute([$title, $author, $category, $status]);
            header("Location: books.php");
            exit;
        } catch (PDOException $e) {
            $error = "Database Error: " . $e->getMessage();
        }
    }
}

require __DIR__ . '/../includes/header.php';
?>

<section style="max-width: 500px; margin: 0 auto;">
    <h1>Add New Book</h1>

    <?php if ($error): ?>
        <p style="color: red; background: #fce4e4; padding: 10px; border: 1px solid #fcc;"><?= htmlspecialchars($error) ?></p>
    <?php endif; ?>

    <form method="post">
        <label>Title</label>
        <input type="text" name="title" required>
        <br><br>

        <label>Author</label>
        <input type="text" name="author" required>
        <br><br>

        <label>Category</label>
        <input type="text" name="category">
        <br><br>

        <label>Status</label>
        <select name="status">
            <option value="available">Available</option>
            <option value="borrowed">Borrowed</option> </select>
        <br><br>

        <input type="hidden" name="csrf_token" value="<?= $_SESSION['csrf_token'] ?>">
        <button type="submit">Save Book</button>
    </form>
</section>

<?php require __DIR__ . '/../includes/footer.php'; ?>
```

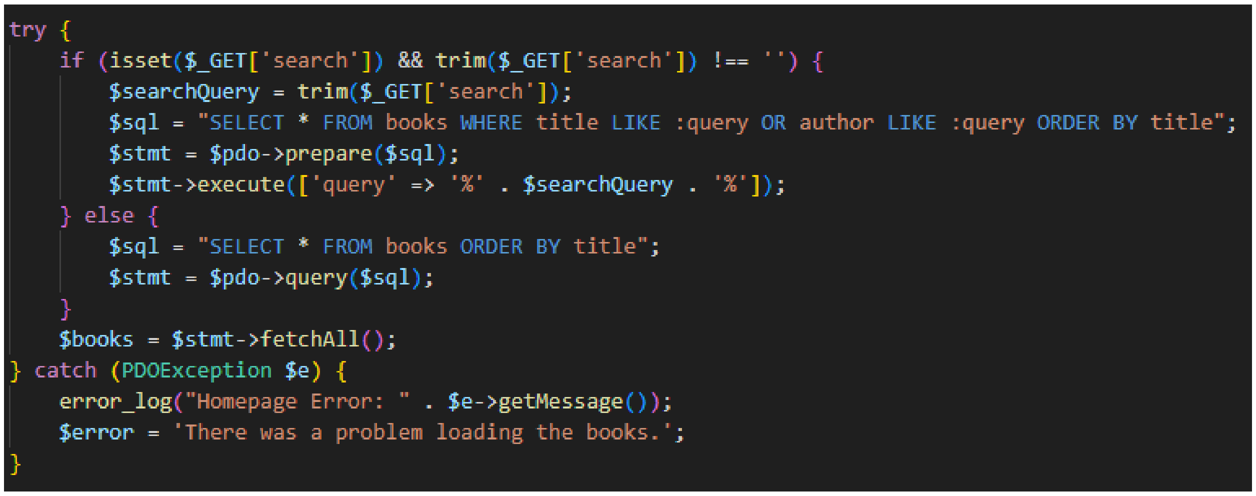
3.5 Fetches Book info and Updates database security



Users > hanah > Downloads > CYB325 > admin > edit_book.php > ...

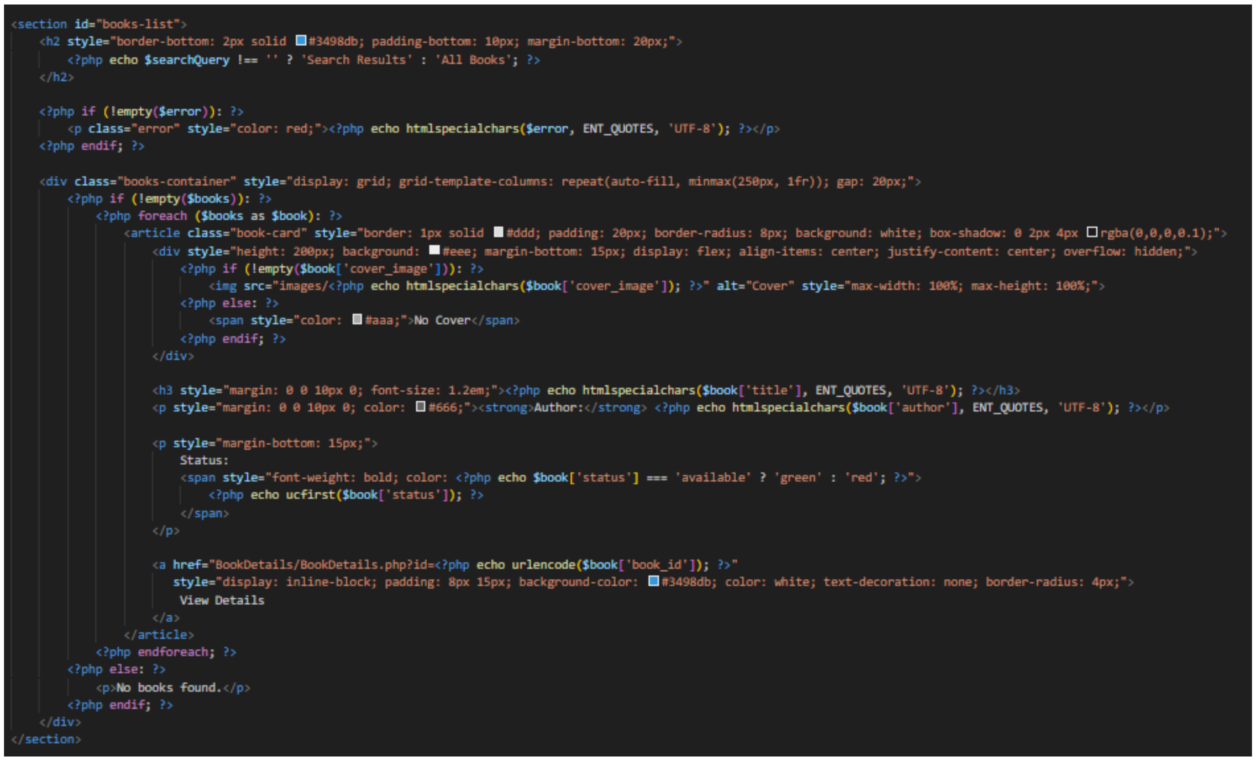
```php
<?php
require __DIR__ . '/../includes/db.php';
require 'auth_check.php';

if (empty($_SESSION['csrf_token'])) {
    $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
}

// Validate Book ID
if (!isset($_GET['book_id']) || !ctype_digit($_GET['book_id'])) {
    die("Invalid Book ID.");
}

$id = $_GET['book_id'];
$error = '';

$stmt = $pdo->prepare("SELECT * FROM books WHERE book_id=?");
$stmt->execute([$id]);
$book = $stmt->fetch();

if (!$book) die("the book is not avalible");

if ($_SERVER['REQUEST_METHOD'] === 'POST') {

    // Verify CSRF
    if (!isset($_POST['csrf_token']) || $_POST['csrf_token'] !== $_SESSION['csrf_token']) {
        die("CSRF validation failed.");
    }

    $title = trim($_POST['title']);
    $author = trim($_POST['author']);

    if (empty($title) || empty($author)) {
        $error = "Title and Author are required.";
    } else {
        $stmt = $pdo->prepare("UPDATE books SET title=?, author=?, category=?, status=? WHERE book_id=?");
        $stmt->execute([
            $title,
            $author,
            trim($_POST['category']),
            $_POST['status'],
            $id
        ]);
        header("Location: books.php");
        exit;
    }
}

require __DIR__ . '/../includes/header.php';
?>

<section style="max-width: 500px; margin: 0 auto;">
    <h1>Edit Book</h1>

    <?php if ($error): ?>
        <p style="color: red;"><?= htmlspecialchars($error) ?></p>
    <?php endif; ?>

    <form method="post">
        <label>Title</label>
        <input type="text" name="title" value="<?= htmlspecialchars($book['title']) ?>" required><br><br>

        <label>Author</label>
        <input type="text" name="author" value="<?= htmlspecialchars($book['author']) ?>" required><br><br>

        <label>Category</label>
        <input type="text" name="category" value="<?= htmlspecialchars($book['category']) ?>"><br><br>

        <label>Status</label>
        <select name="status">
            <option <?= $book['status']=='available'?'selected':'' ?> value="available">Available</option>
            <option <?= $book['status']=='borrowed'?'selected':'' ?> value="borrowed">Borrowed</option>
        </select><br><br>

        <input type="hidden" name="csrf_token" value="<?= $_SESSION['csrf_token'] ?>">
        <button type="submit">Update Book</button>
    </form>
</section>
```

3.6 Loading the books from the database when searching for them:



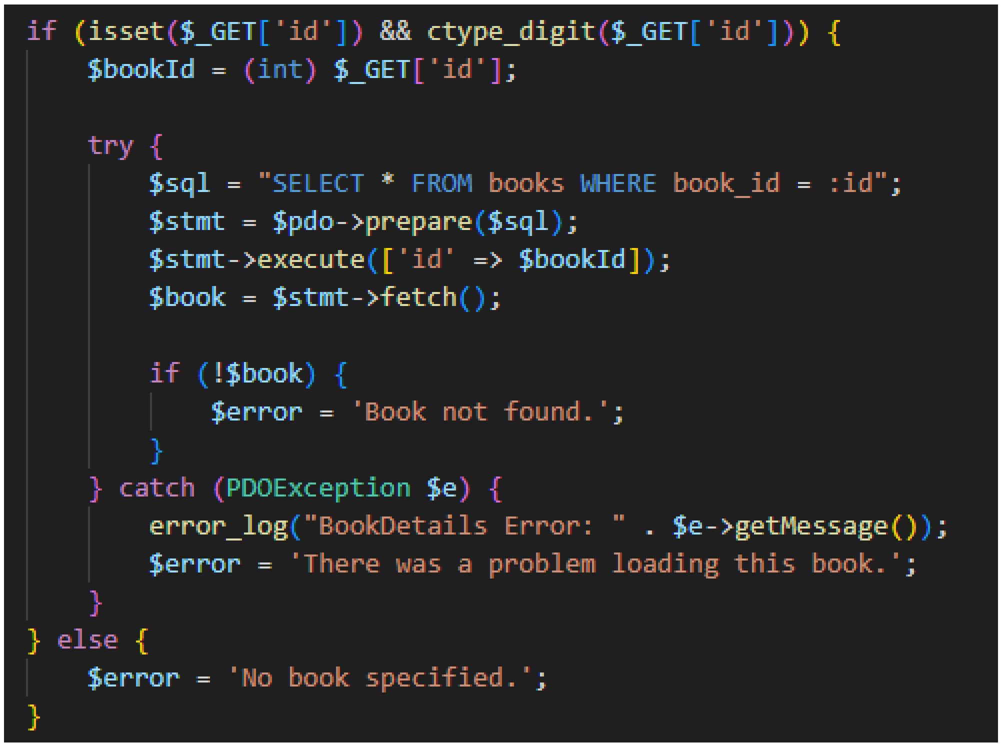
```php
try {
    if (isset($_GET['search']) && trim($_GET['search']) !== '') {
        $searchQuery = trim($_GET['search']);
        $sql = "SELECT * FROM books WHERE title LIKE :query OR author LIKE :query ORDER BY title";
        $stmt = $pdo->prepare($sql);
        $stmt->execute(['query' => '%' . $searchQuery . '%']);
    } else {
        $sql = "SELECT * FROM books ORDER BY title";
        $stmt = $pdo->query($sql);
    }
    $books = $stmt->fetchAll();
} catch (PDOException $e) {
    error_log("Homepage Error: " . $e->getMessage());
    $error = 'There was a problem loading the books.';
}
```

3.7 Book list section:



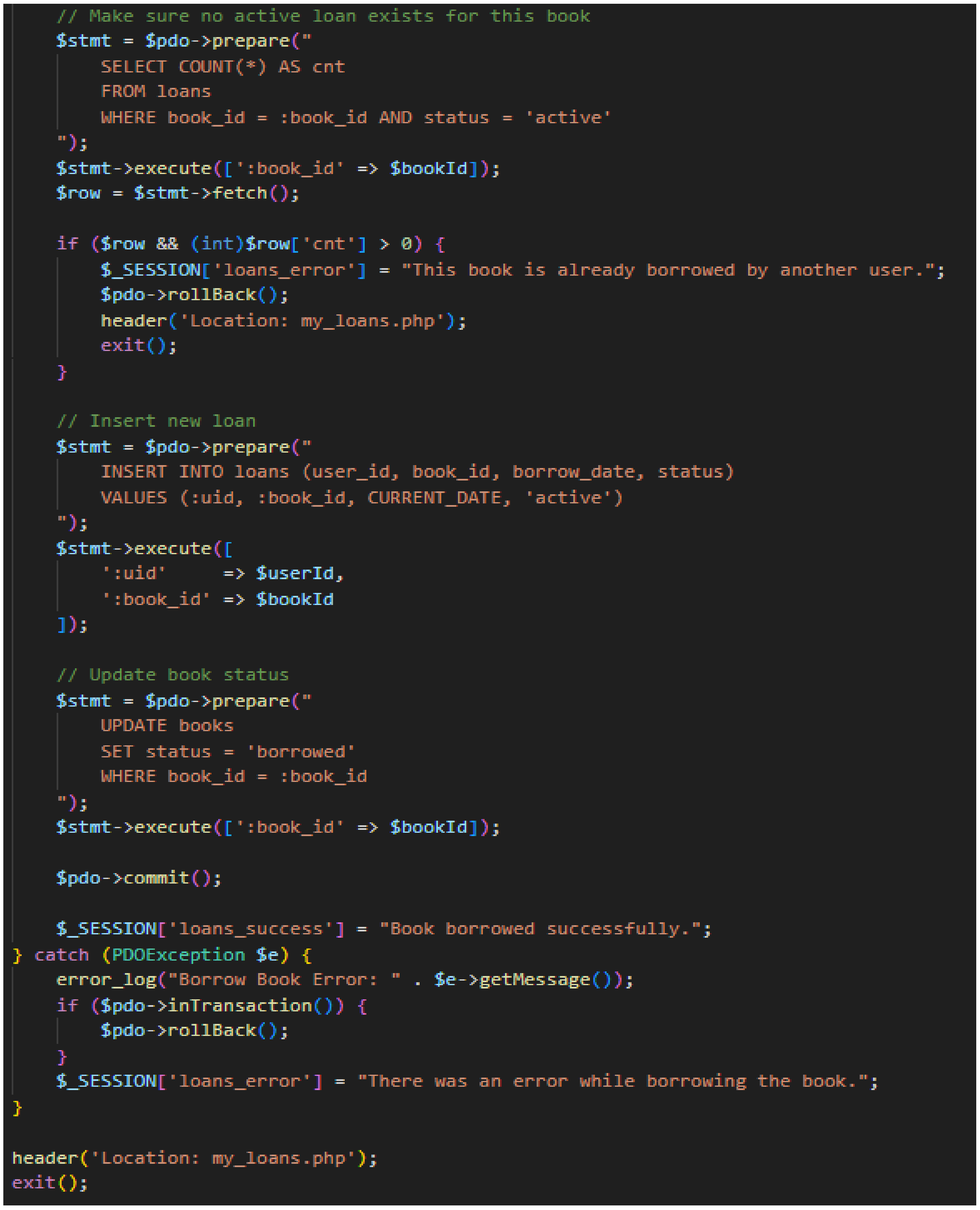
```html
<section id="books-list">
    <h2 style="border-bottom: 2px solid #3498db; padding-bottom: 10px; margin-bottom: 20px;">
        <?php echo $searchQuery !== '' ? 'Search Results' : 'All Books'; ?>
    </h2>

    <?php if (!empty($error)): ?>
        <p class="error" style="color: red;"><?php echo htmlspecialchars($error, ENT_QUOTES, 'UTF-8'); ?></p>
    <?php endif; ?>

    <div class="books-container" style="display: grid; grid-template-columns: repeat(auto-fill, minmax(250px, 1fr)); gap: 20px;">
        <?php if (!empty($books)): ?>
            <?php foreach ($books as $book): ?>
                <article class="book-card" style="border: 1px solid #ddd; padding: 20px; border-radius: 8px; background: white; box-shadow: 0 2px 4px rgba(0,0,0,0.1);">
                    <div style="height: 200px; background: #eee; margin-bottom: 15px; display: flex; align-items: center; justify-content: center; overflow: hidden;">
                        <?php if (!empty($book['cover_image'])): ?>
                            <img src="images/<?php echo htmlspecialchars($book['cover_image']); ?>" alt="Cover" style="max-width: 100%; max-height: 100%;">
                        <?php else: ?>
                            <span style="color: #aaa;">No Cover</span>
                        <?php endif; ?>
                    </div>

                    <h3 style="margin: 0 0 10px 0; font-size: 1.2em;"><?php echo htmlspecialchars($book['title'], ENT_QUOTES, 'UTF-8'); ?></h3>
                    <p style="margin: 0 0 10px 0; color: #666;"><strong>Author:</strong> <?php echo htmlspecialchars($book['author'], ENT_QUOTES, 'UTF-8'); ?></p>

                    <p style="margin-bottom: 15px;">
                        Status:
                        <span style="font-weight: bold; color: <?php echo $book['status'] === 'available' ? 'green' : 'red'; ?>">
                            <?php echo ucfirst($book['status']); ?>
                        </span>
                    </p>

                    <a href="BookDetails/BookDetails.php?id=<?php echo urlencode($book['book_id']); ?>"
                      style="display: inline-block; padding: 8px 15px; background-color: #3498db; color: white; text-decoration: none; border-radius: 4px;">
                        View Details
                    </a>
                </article>
            <?php endforeach; ?>
        <?php else: ?>
            <p>No books found.</p>
        <?php endif; ?>
    </div>
</section>
```

3.8 Showing a book details:



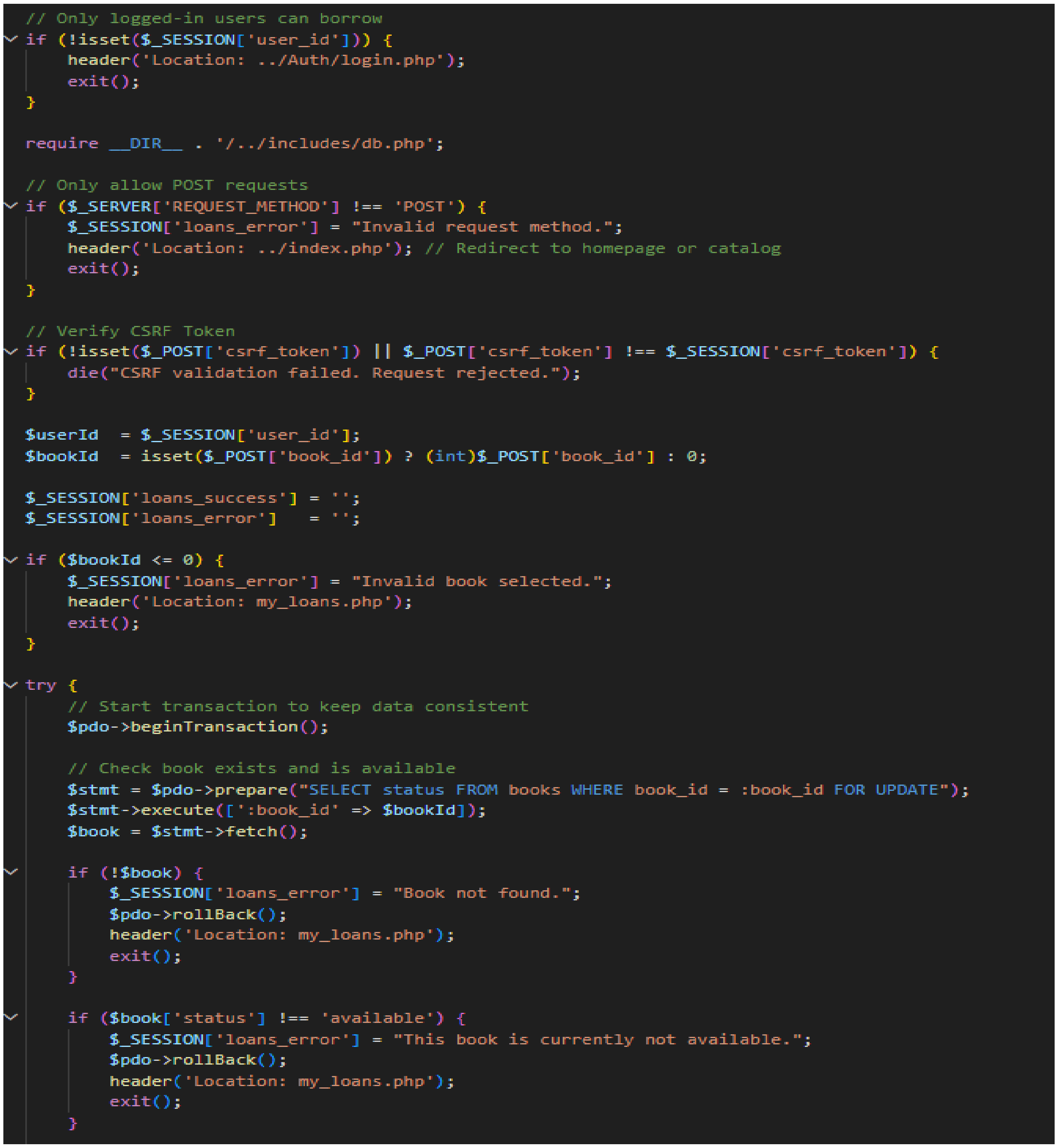
```php
if (isset($_GET['id']) && ctype_digit($_GET['id'])) {
    $bookId = (int) $_GET['id'];

    try {
        $sql = "SELECT * FROM books WHERE book_id = :id";
        $stmt = $pdo->prepare($sql);
        $stmt->execute(['id' => $bookId]);
        $book = $stmt->fetch();

        if (!$book) {
            $error = 'Book not found.';
        }
    } catch (PDOException $e) {
        error_log("BookDetails Error: " . $e->getMessage());
        $error = 'There was a problem loading this book.';
    }
} else {
    $error = 'No book specified.';
}
```

3.9 Create Loan & Update Book Status



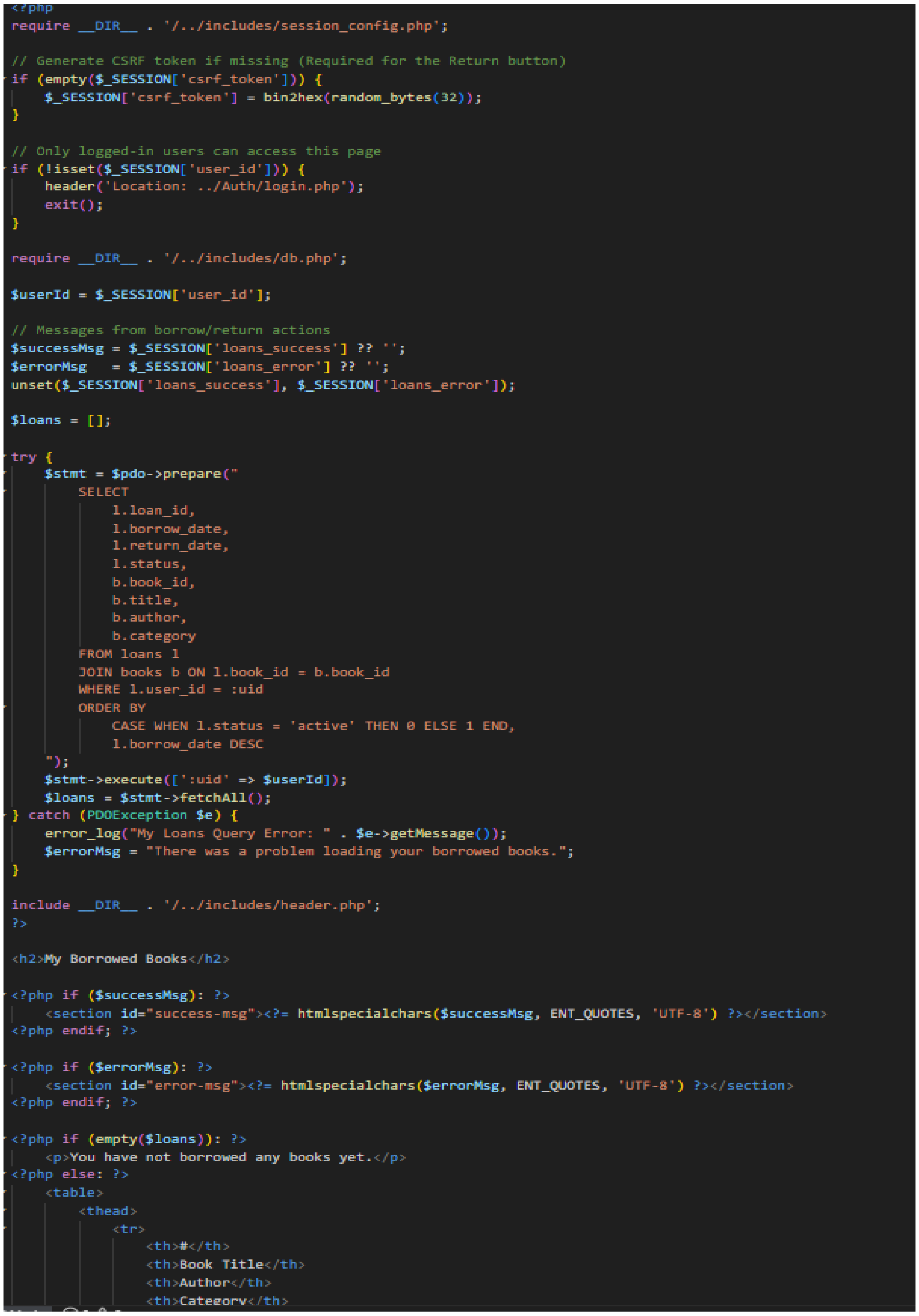
```php
    // Make sure no active loan exists for this book
    $stmt = $pdo->prepare("
        SELECT COUNT(*) AS cnt
        FROM loans
        WHERE book_id = :book_id AND status = 'active'
    ");
    $stmt->execute([':book_id' => $bookId]);
    $row = $stmt->fetch();

    if ($row && (int)$row['cnt'] > 0) {
        $_SESSION['loans_error'] = "This book is already borrowed by another user.";
        $pdo->rollBack();
        header('Location: my_loans.php');
        exit();
    }

    // Insert new loan
    $stmt = $pdo->prepare("
        INSERT INTO loans (user_id, book_id, borrow_date, status)
        VALUES (:uid, :book_id, CURRENT_DATE, 'active')
    ");
    $stmt->execute([
        ':uid'     => $userId,
        ':book_id' => $bookId
    ]);

    // Update book status
    $stmt = $pdo->prepare("
        UPDATE books
        SET status = 'borrowed'
        WHERE book_id = :book_id
    ");
    $stmt->execute([':book_id' => $bookId]);

    $pdo->commit();

    $_SESSION['loans_success'] = "Book borrowed successfully.";
} catch (PDOException $e) {
    error_log("Borrow Book Error: " . $e->getMessage());
    if ($pdo->inTransaction()) {
        $pdo->rollBack();
    }
    $_SESSION['loans_error'] = "There was an error while borrowing the book.";
}

header('Location: my_loans.php');
exit();
```

3.10 Borrow Request Security & Validation Checks



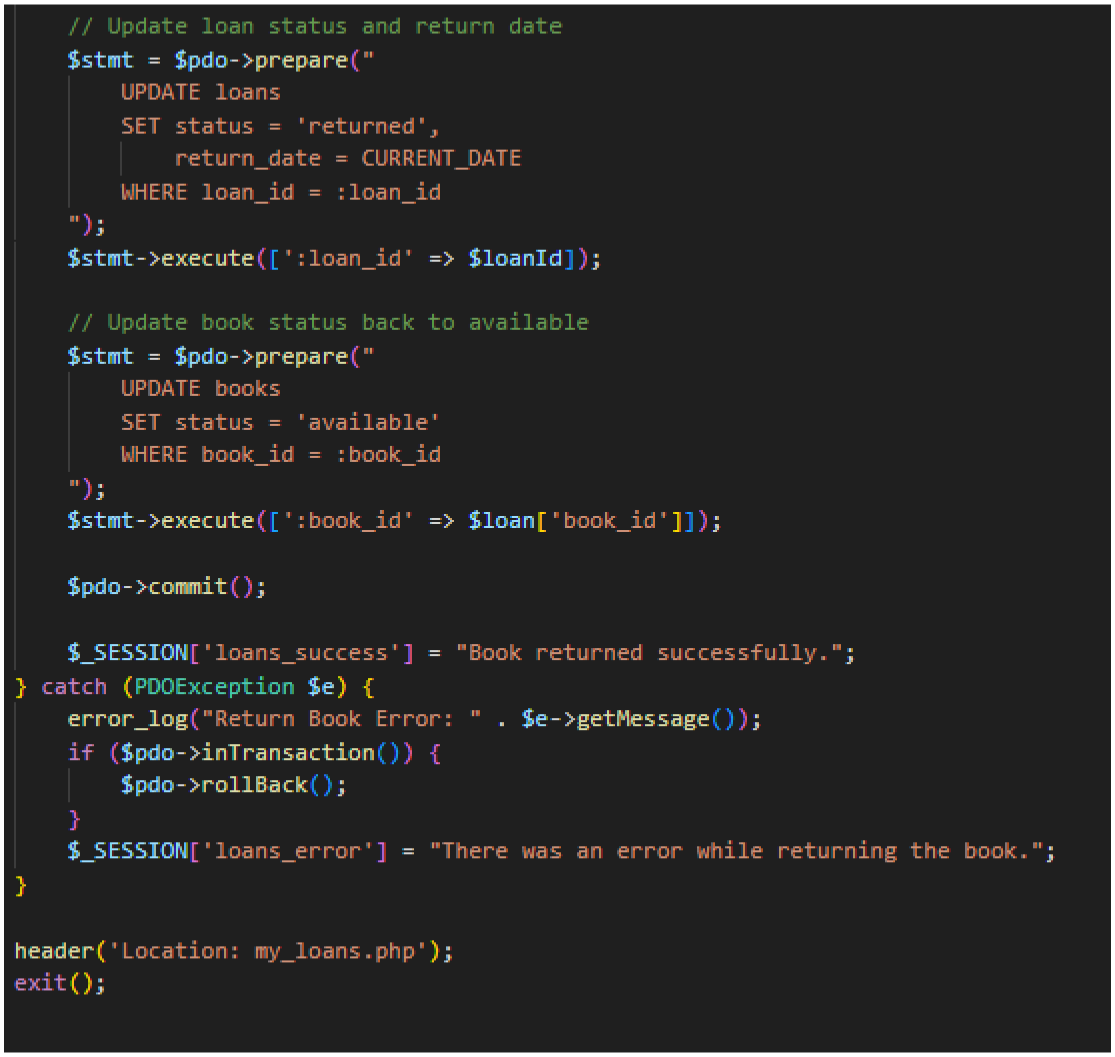
```php
// Only logged-in users can borrow
if (!isset($_SESSION['user_id'])) {
    header('Location: ../Auth/login.php');
    exit();
}

require __DIR__ . '/../includes/db.php';

// Only allow POST requests
if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
    $_SESSION['loans_error'] = "Invalid request method.";
    header('Location: ../index.php'); // Redirect to homepage or catalog
    exit();
}

// Verify CSRF Token
if (!isset($_POST['csrf_token']) || $_POST['csrf_token'] !== $_SESSION['csrf_token']) {
    die("CSRF validation failed. Request rejected.");
}

$userId  = $_SESSION['user_id'];
$bookId  = isset($_POST['book_id']) ? (int)$_POST['book_id'] : 0;

$_SESSION['loans_success'] = '';
$_SESSION['loans_error']   = '';

if ($bookId <= 0) {
    $_SESSION['loans_error'] = "Invalid book selected.";
    header('Location: my_loans.php');
    exit();
}

try {
    // Start transaction to keep data consistent
    $pdo->beginTransaction();

    // Check book exists and is available
    $stmt = $pdo->prepare("SELECT status FROM books WHERE book_id = :book_id FOR UPDATE");
    $stmt->execute([':book_id' => $bookId]);
    $book = $stmt->fetch();

    if (!$book) {
        $_SESSION['loans_error'] = "Book not found.";
        $pdo->rollBack();
        header('Location: my_loans.php');
        exit();
    }

    if ($book['status'] !== 'available') {
        $_SESSION['loans_error'] = "This book is currently not available.";
        $pdo->rollBack();
        header('Location: my_loans.php');
        exit();
    }
```

3.11 Shows borrowed books, their status, and return option



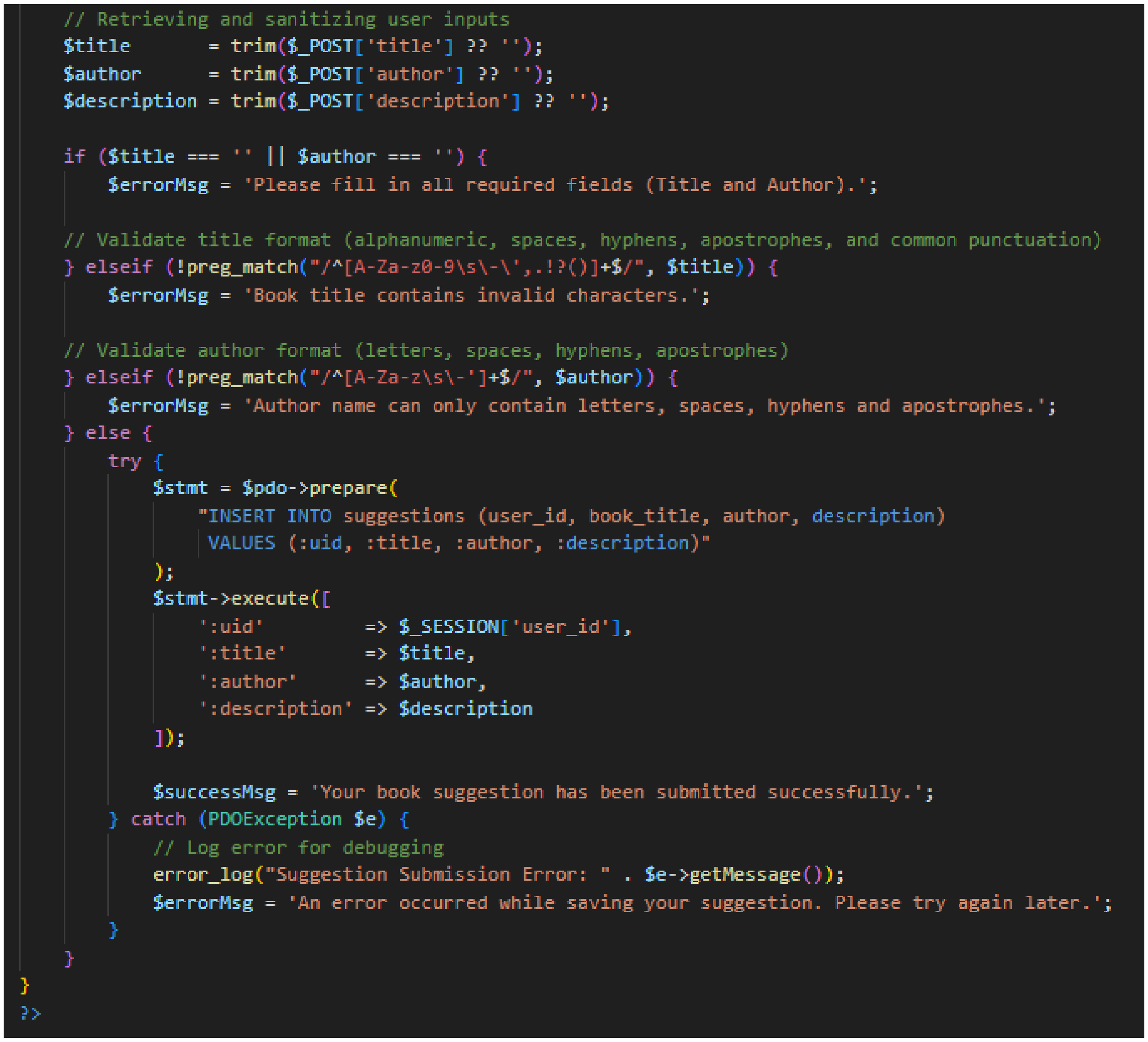
```
<?php
require __DIR__ . '/../includes/session_config.php';

// Generate CSRF token if missing (Required for the Return button)
if (empty($_SESSION['csrf_token'])) {
    $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
}

// Only logged-in users can access this page
if (!isset($_SESSION['user_id'])) {
    header('Location: ../Auth/login.php');
    exit();
}

require __DIR__ . '/../includes/db.php';

$userId = $_SESSION['user_id'];

// Messages from borrow/return actions
$successMsg = $_SESSION['loans_success'] ?? '';
$errorMsg   = $_SESSION['loans_error'] ?? '';
unset($_SESSION['loans_success'], $_SESSION['loans_error']);

$loans = [];

try {
    $stmt = $pdo->prepare("
        SELECT
            l.loan_id,
            l.borrow_date,
            l.return_date,
            l.status,
            b.book_id,
            b.title,
            b.author,
            b.category
        FROM loans l
        JOIN books b ON l.book_id = b.book_id
        WHERE l.user_id = :uid
        ORDER BY
            CASE WHEN l.status = 'active' THEN 0 ELSE 1 END,
            l.borrow_date DESC
    ");
    $stmt->execute([':uid' => $userId]);
    $loans = $stmt->fetchAll();
} catch (PDOException $e) {
    error_log("My Loans Query Error: " . $e->getMessage());
    $errorMsg = "There was a problem loading your borrowed books.";
}

include __DIR__ . '/../includes/header.php';
?>

<h2>My Borrowed Books</h2>

<?php if ($successMsg): ?>
    <section id="success-msg"><?= htmlspecialchars($successMsg, ENT_QUOTES, 'UTF-8') ?></section>
<?php endif; ?>

<?php if ($errorMsg): ?>
    <section id="error-msg"><?= htmlspecialchars($errorMsg, ENT_QUOTES, 'UTF-8') ?></section>
<?php endif; ?>

<?php if (empty($loans)): ?>
    <p>You have not borrowed any books yet.</p>
<?php else: ?>
    <table>
        <thead>
            <tr>
                <th>#</th>
                <th>Book Title</th>
                <th>Author</th>
                <th>Category</th>
```

3.12 Return Loan & Restore Book Availability



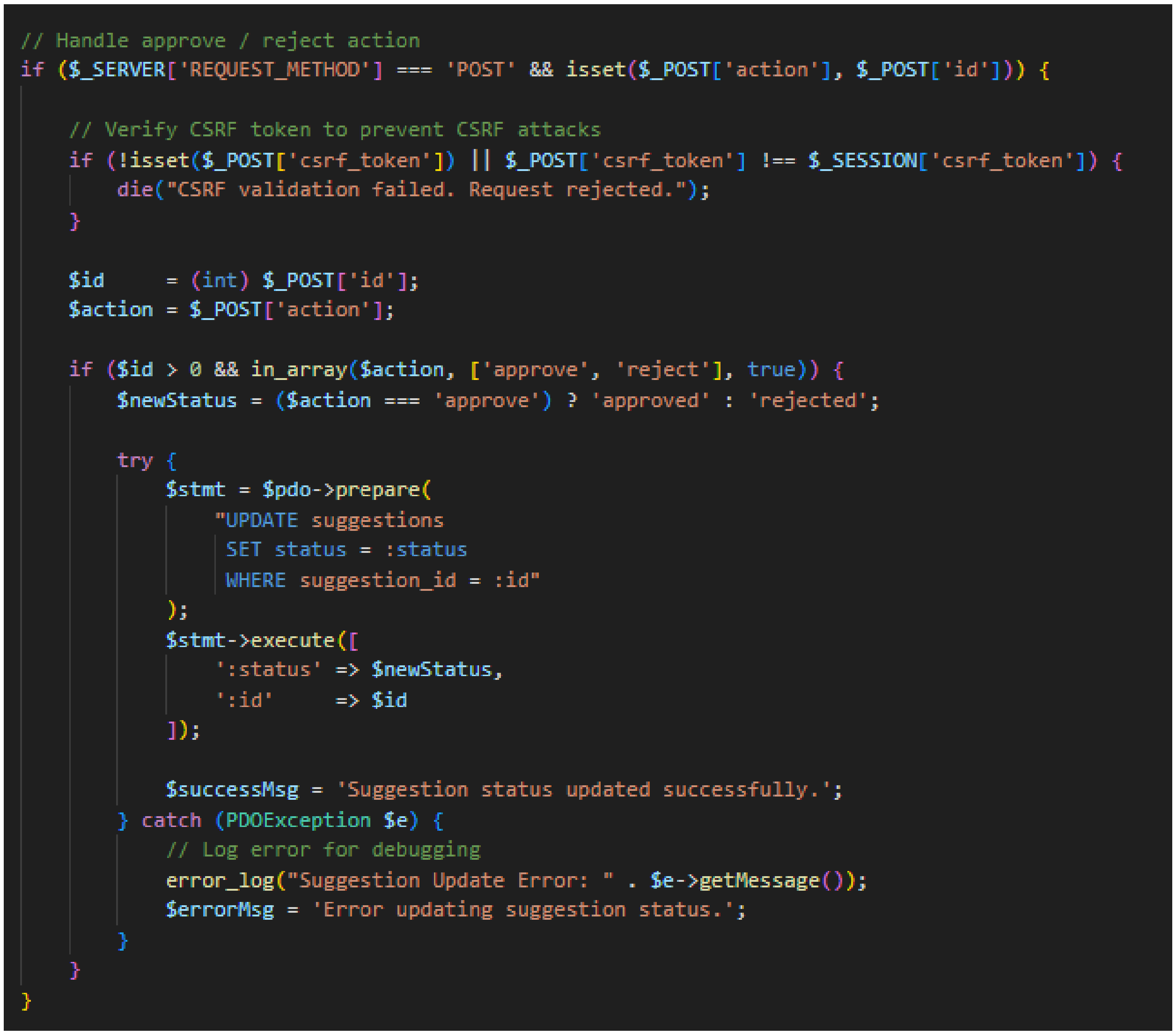
```
    // Update loan status and return date
    $stmt = $pdo->prepare("
        UPDATE loans
        SET status = 'returned',
            return_date = CURRENT_DATE
        WHERE loan_id = :loan_id
    ");
    $stmt->execute([':loan_id' => $loanId]);

    // Update book status back to available
    $stmt = $pdo->prepare("
        UPDATE books
        SET status = 'available'
        WHERE book_id = :book_id
    ");
    $stmt->execute([':book_id' => $loan['book_id']]);

    $pdo->commit();

    $_SESSION['loans_success'] = "Book returned successfully.";
} catch (PDOException $e) {
    error_log("Return Book Error: " . $e->getMessage());
    if ($pdo->inTransaction()) {
        $pdo->rollBack();
    }
    $_SESSION['loans_error'] = "There was an error while returning the book.";
}

header('Location: my_loans.php');
exit();
```

3.13 Submit New Book Suggestion (Validation + Insert Query)



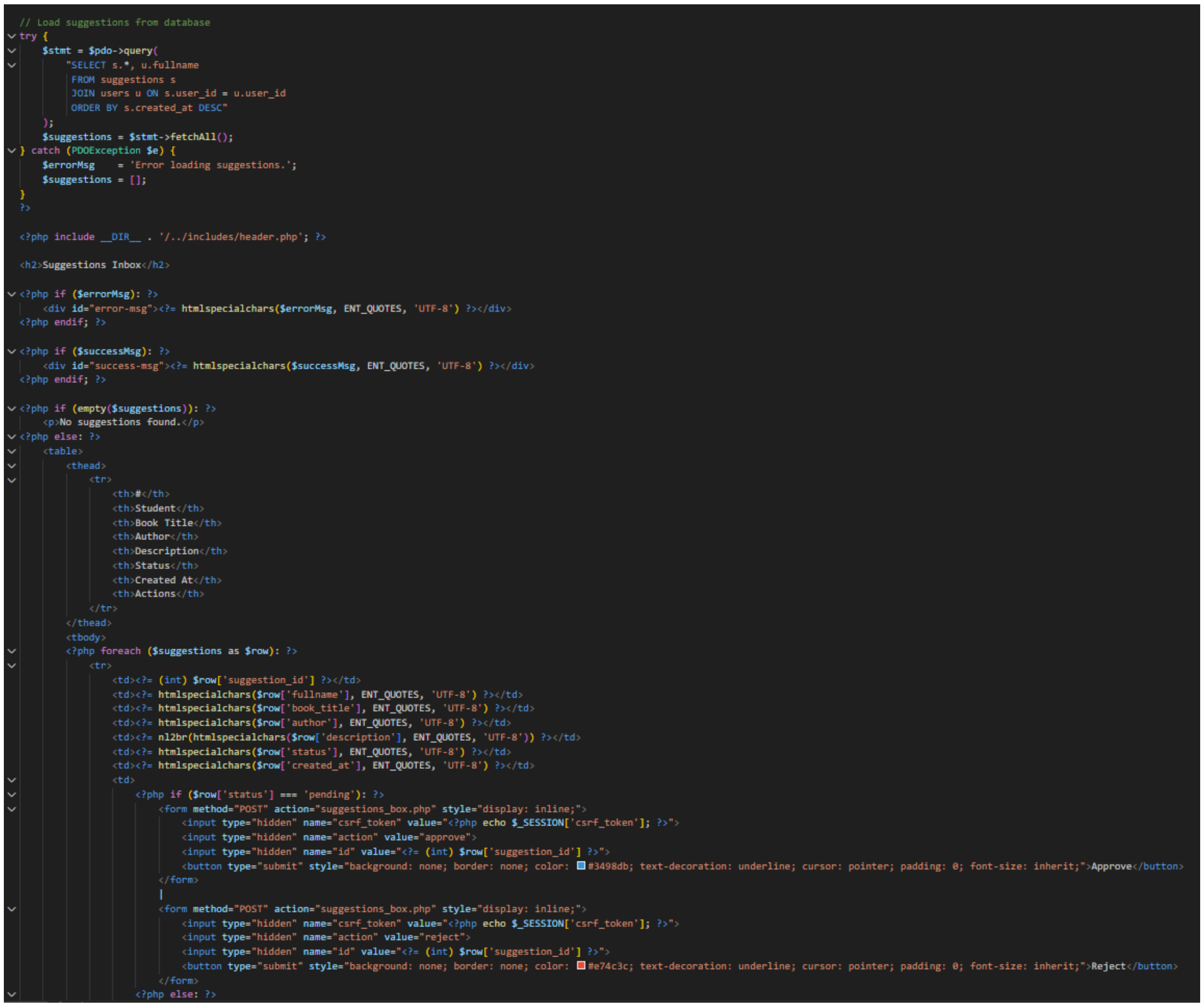
```php
    // Retrieving and sanitizing user inputs
    $title       = trim($_POST['title'] ?? '');
    $author      = trim($_POST['author'] ?? '');
    $description = trim($_POST['description'] ?? '');

    if ($title === '' || $author === '') {
        $errorMsg = 'Please fill in all required fields (Title and Author).';

    // Validate title format (alphanumeric, spaces, hyphens, apostrophes, and common punctuation)
    } elseif (!preg_match("/^[A-Za-z0-9\s\-\',.!?()]+$/", $title)) {
        $errorMsg = 'Book title contains invalid characters.';

    // Validate author format (letters, spaces, hyphens, apostrophes)
    } elseif (!preg_match("/^[A-Za-z\s\-']+$/", $author)) {
        $errorMsg = 'Author name can only contain letters, spaces, hyphens and apostrophes.';
    } else {
        try {
            $stmt = $pdo->prepare(
                "INSERT INTO suggestions (user_id, book_title, author, description)
                 VALUES (:uid, :title, :author, :description)"
            );
            $stmt->execute([
                ':uid'         => $_SESSION['user_id'],
                ':title'       => $title,
                ':author'      => $author,
                ':description' => $description
            ]);

            $successMsg = 'Your book suggestion has been submitted successfully.';
        } catch (PDOException $e) {
            // Log error for debugging
            error_log("Suggestion Submission Error: " . $e->getMessage());
            $errorMsg = 'An error occurred while saving your suggestion. Please try again later.';
        }
    }
}
?>
```

3.14 Approve / Reject Book Suggestion (Admin Action Handler)



```php
// Handle approve / reject action
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['action'], $_POST['id'])) {

    // Verify CSRF token to prevent CSRF attacks
    if (!isset($_POST['csrf_token']) || $_POST['csrf_token'] !== $_SESSION['csrf_token']) {
        die("CSRF validation failed. Request rejected.");
    }

    $id     = (int) $_POST['id'];
    $action = $_POST['action'];

    if ($id > 0 && in_array($action, ['approve', 'reject'], true)) {
        $newStatus = ($action === 'approve') ? 'approved' : 'rejected';

        try {
            $stmt = $pdo->prepare(
                "UPDATE suggestions
                 SET status = :status
                 WHERE suggestion_id = :id"
            );
            $stmt->execute([
                ':status' => $newStatus,
                ':id'     => $id
            ]);

            $successMsg = 'Suggestion status updated successfully.';
        } catch (PDOException $e) {
            // Log error for debugging
            error_log("Suggestion Update Error: " . $e->getMessage());
            $errorMsg = 'Error updating suggestion status.';
        }
    }
}
```

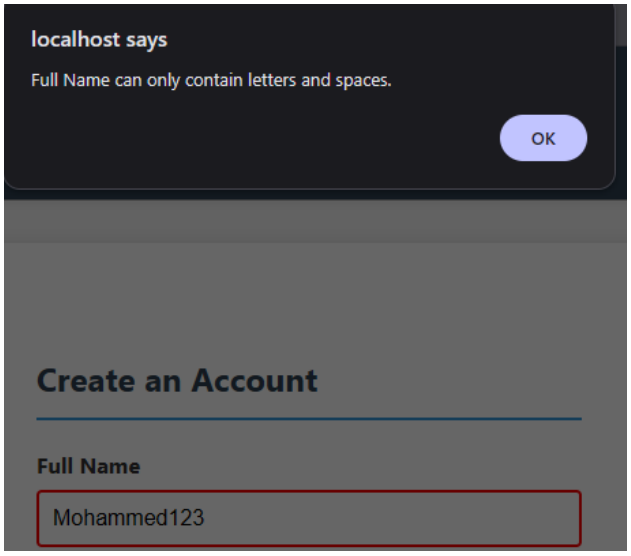
3.15 Loads all submitted suggestions from the database and renders them in a table



```php
// Load suggestions from database
try {
    $stmt = $pdo->query(
        "SELECT s.*, u.fullname
         FROM suggestions s
         JOIN users u ON s.user_id = u.user_id
         ORDER BY s.created_at DESC"
    );
    $suggestions = $stmt->fetchAll();
} catch (PDOException $e) {
    $errorMsg    = 'Error loading suggestions.';
    $suggestions = [];
}
?>

<?php include __DIR__ . '/../includes/header.php'; ?>

<h2>Suggestions Inbox</h2>

<?php if ($errorMsg): ?>
    <div id="error-msg"><?= htmlspecialchars($errorMsg, ENT_QUOTES, 'UTF-8') ?></div>
<?php endif; ?>

<?php if ($successMsg): ?>
    <div id="success-msg"><?= htmlspecialchars($successMsg, ENT_QUOTES, 'UTF-8') ?></div>
<?php endif; ?>

<?php if (empty($suggestions)): ?>
    <p>No suggestions found.</p>
<?php else: ?>
    <table>
        <thead>
            <tr>
                <th>#</th>
                <th>Student</th>
                <th>Book Title</th>
                <th>Author</th>
                <th>Description</th>
                <th>Status</th>
                <th>Created At</th>
                <th>Actions</th>
            </tr>
        </thead>
        <tbody>
        <?php foreach ($suggestions as $row): ?>
            <tr>
                <td><?= (int) $row['suggestion_id'] ?></td>
                <td><?= htmlspecialchars($row['fullname'], ENT_QUOTES, 'UTF-8') ?></td>
                <td><?= htmlspecialchars($row['book_title'], ENT_QUOTES, 'UTF-8') ?></td>
                <td><?= htmlspecialchars($row['author'], ENT_QUOTES, 'UTF-8') ?></td>
                <td><?= nl2br(htmlspecialchars($row['description'], ENT_QUOTES, 'UTF-8')) ?></td>
                <td><?= htmlspecialchars($row['status'], ENT_QUOTES, 'UTF-8') ?></td>
                <td><?= htmlspecialchars($row['created_at'], ENT_QUOTES, 'UTF-8') ?></td>
                <td>
                    <?php if ($row['status'] === 'pending'): ?>
                        <form method="POST" action="suggestions_box.php" style="display: inline;">
                            <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
                            <input type="hidden" name="action" value="approve">
                            <input type="hidden" name="id" value="<?= (int) $row['suggestion_id'] ?>">
                            <button type="submit" style="background: none; border: none; color: #3498db; text-decoration: underline; cursor: pointer; padding: 0; font-size: inherit;">Approve</button>
                        </form>
                        |
                        <form method="POST" action="suggestions_box.php" style="display: inline;">
                            <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
                            <input type="hidden" name="action" value="reject">
                            <input type="hidden" name="id" value="<?= (int) $row['suggestion_id'] ?>">
                            <button type="submit" style="background: none; border: none; color: #e74c3c; text-decoration: underline; cursor: pointer; padding: 0; font-size: inherit;">Reject</button>
                        </form>
                    <?php else: ?>
```

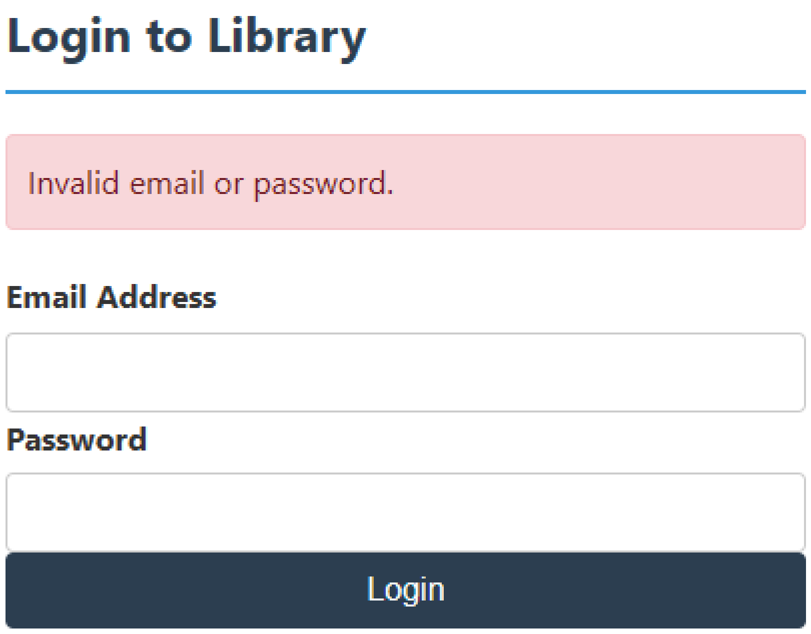
Section 4: Testing Results

4.1 Registration input validation: invalid Full Name


localhost says

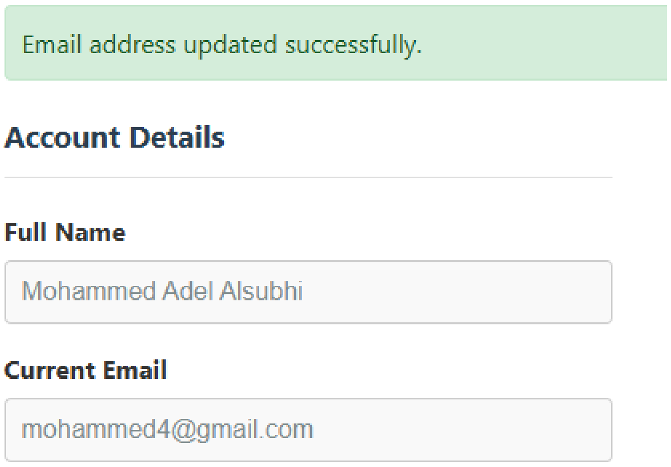
Full Name can only contain letters and spaces.

4.2 Authentication security: Email or password mismatch


Login to Library

Invalid email or password.

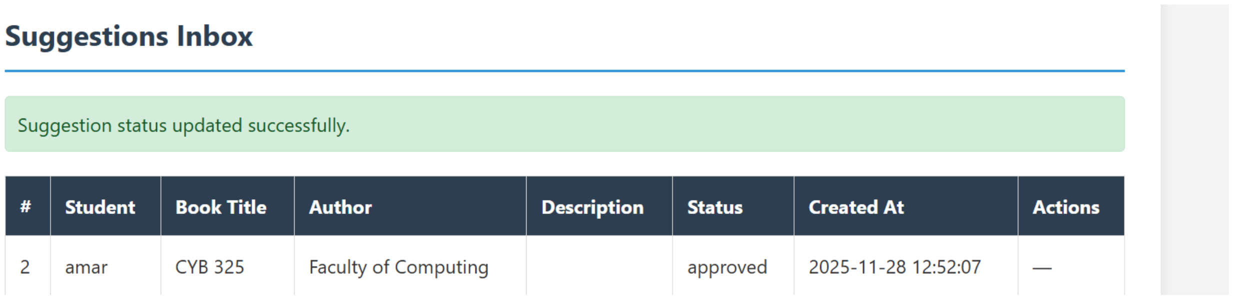
4.3 Profile update: updating Email address


Email address updated successfully.

Account Details

Full Name	Mohammed Adel Alsubhi
Current Email	mohammed4@gmail.com

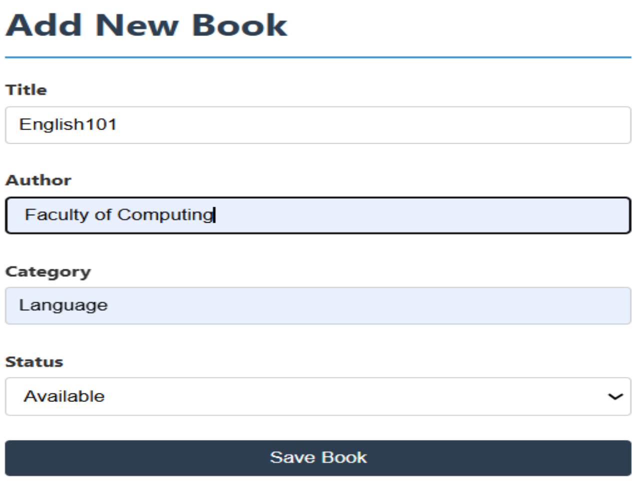
4.4 Accepting user suggestions


Suggestions Inbox

Suggestion status updated successfully.

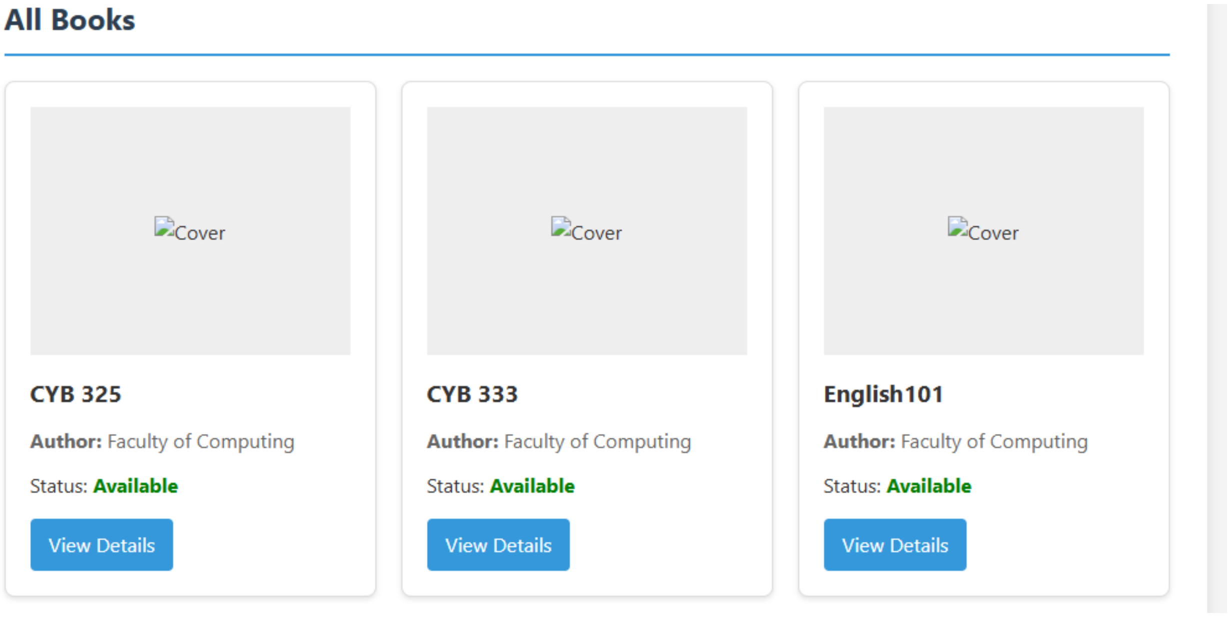
#	Student	Book Title	Author	Description	Status	Created At	Actions
2	amar	CYB 325	Faculty of Computing		approved	2025-11-28 12:52:07	—

4.5 Adding a new book


Add New Book

Title	English101
Author	Faculty of Computing
Category	Language
Status	Available

4.6 the new book is Available to everyone.



All Books

Cover

CYB 325

Author: Faculty of Computing

Status: **Available**

Cover

CYB 333

Author: Faculty of Computing

Status: **Available**

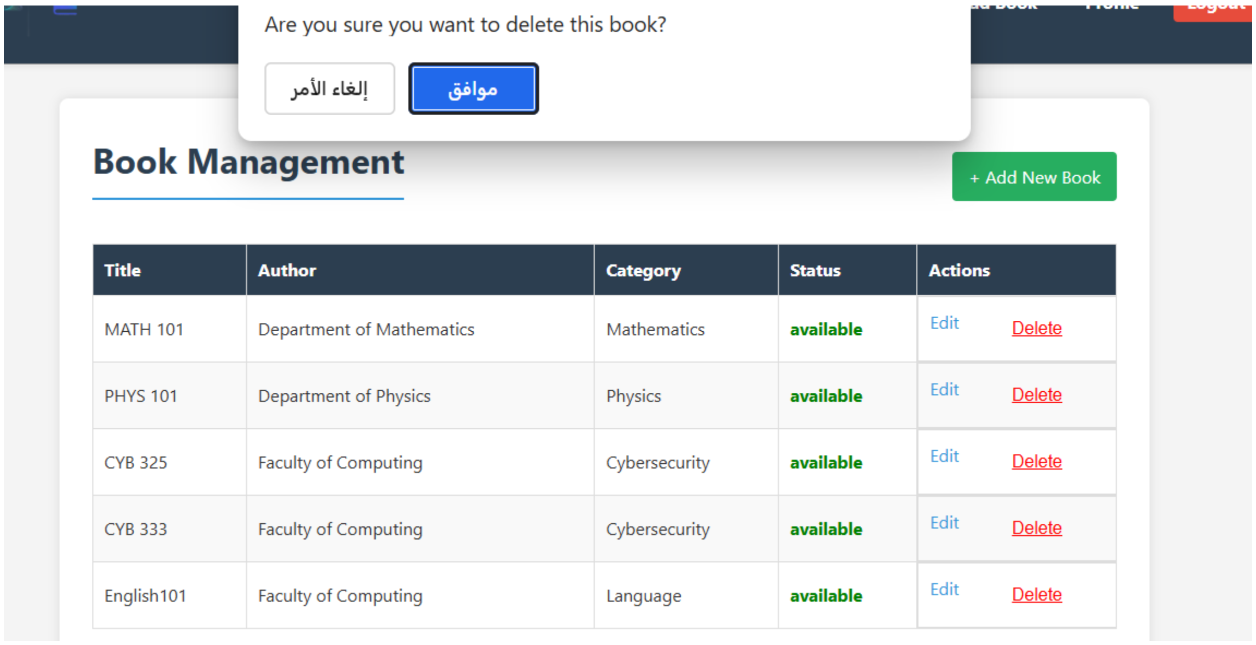
Cover

English101

Author: Faculty of Computing

Status: **Available**

4.7 Delete the book

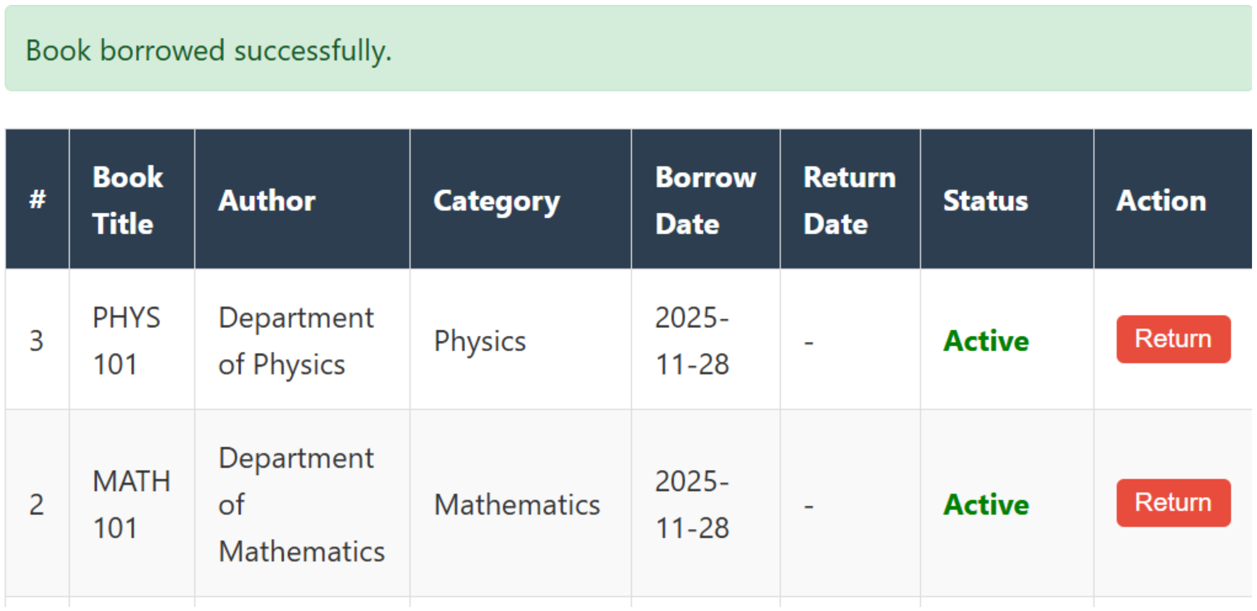


Are you sure you want to delete this book?

Book Management

Title	Author	Category	Status	Actions
MATH 101	Department of Mathematics	Mathematics	available	Edit Delete
PHYS 101	Department of Physics	Physics	available	Edit Delete
CYB 325	Faculty of Computing	Cybersecurity	available	Edit Delete
CYB 333	Faculty of Computing	Cybersecurity	available	Edit Delete
English101	Faculty of Computing	Language	available	Edit Delete

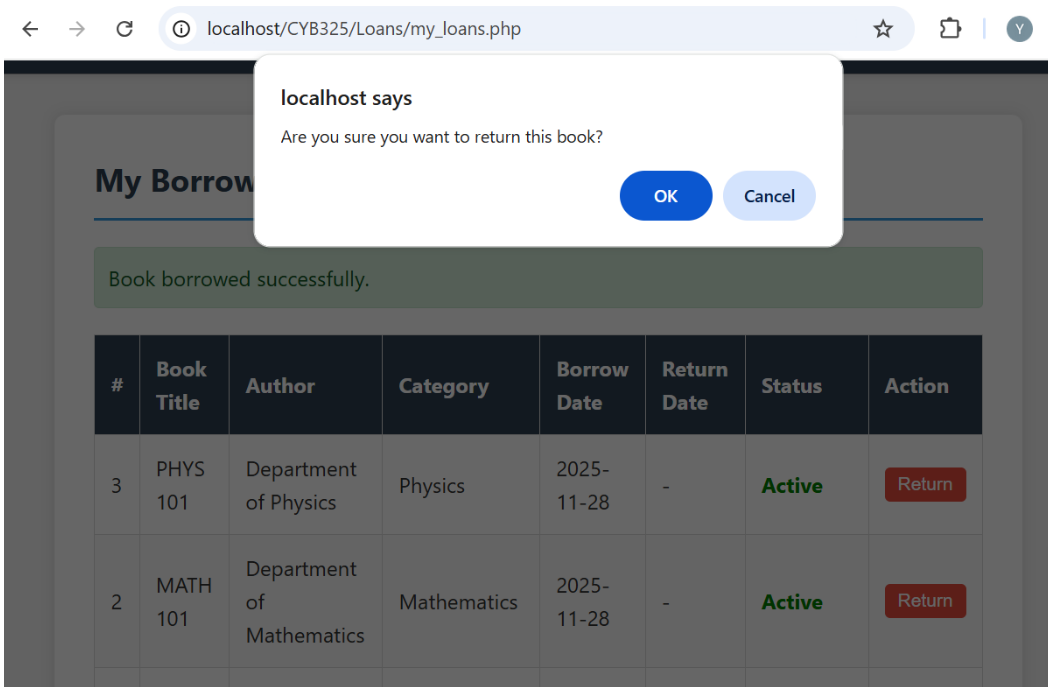
4.8 Book Borrowed



Book borrowed successfully.

#	Book Title	Author	Category	Borrow Date	Return Date	Status	Action
3	PHYS 101	Department of Physics	Physics	2025-11-28	-	Active	Return
2	MATH 101	Department of Mathematics	Mathematics	2025-11-28	-	Active	Return

4.9 Book Return



localhost/CYB325/Loans/my_loans.php

localhost says

Are you sure you want to return this book?

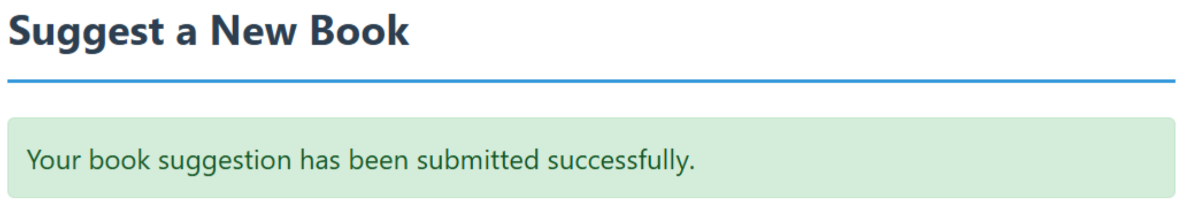
OK Cancel

My Borrow

Book borrowed successfully.

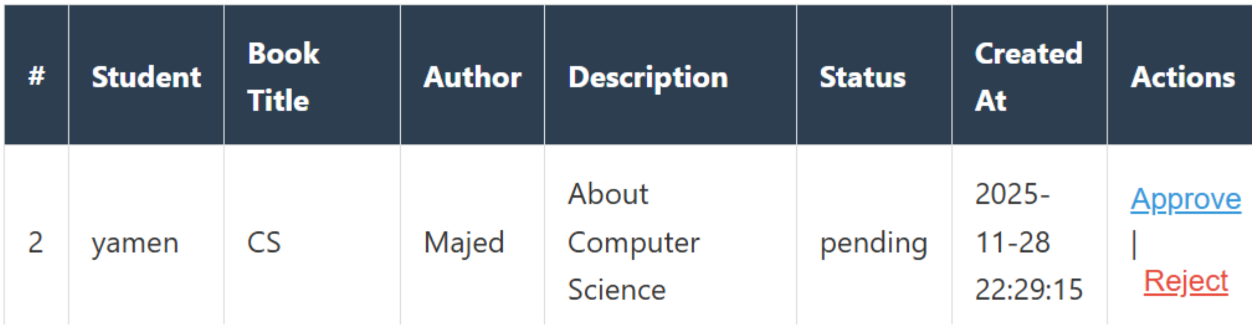
#	Book Title	Author	Category	Borrow Date	Return Date	Status	Action
3	PHYS 101	Department of Physics	Physics	2025-11-28	-	Active	Return
2	MATH 101	Department of Mathematics	Mathematics	2025-11-28	-	Active	Return

4.10 suggest a Book

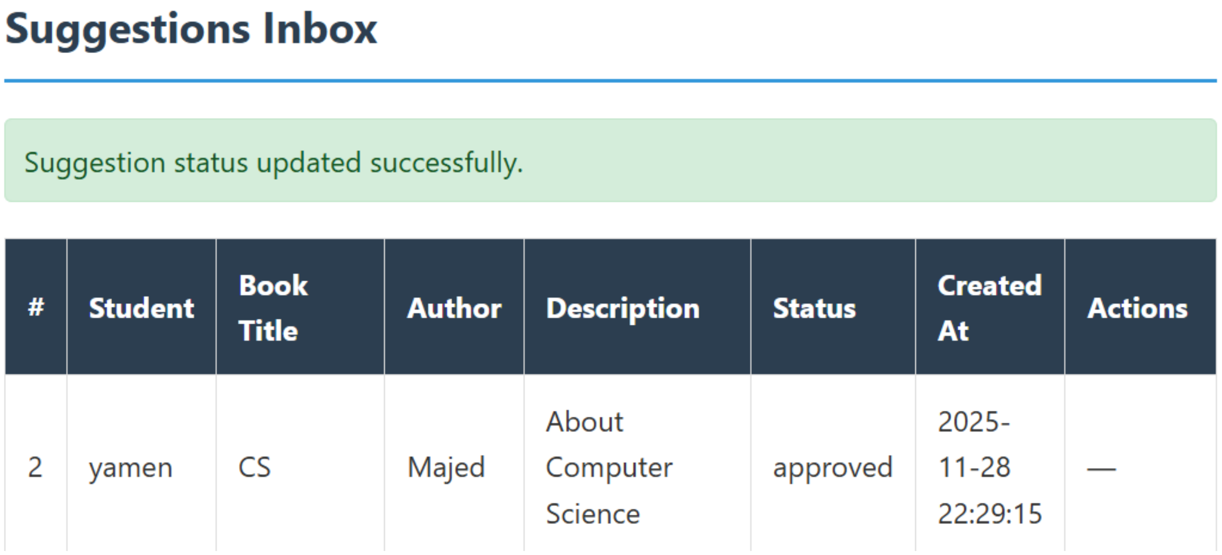

Suggest a New Book

Your book suggestion has been submitted successfully.

4.11 Approve or Reject a suggestion



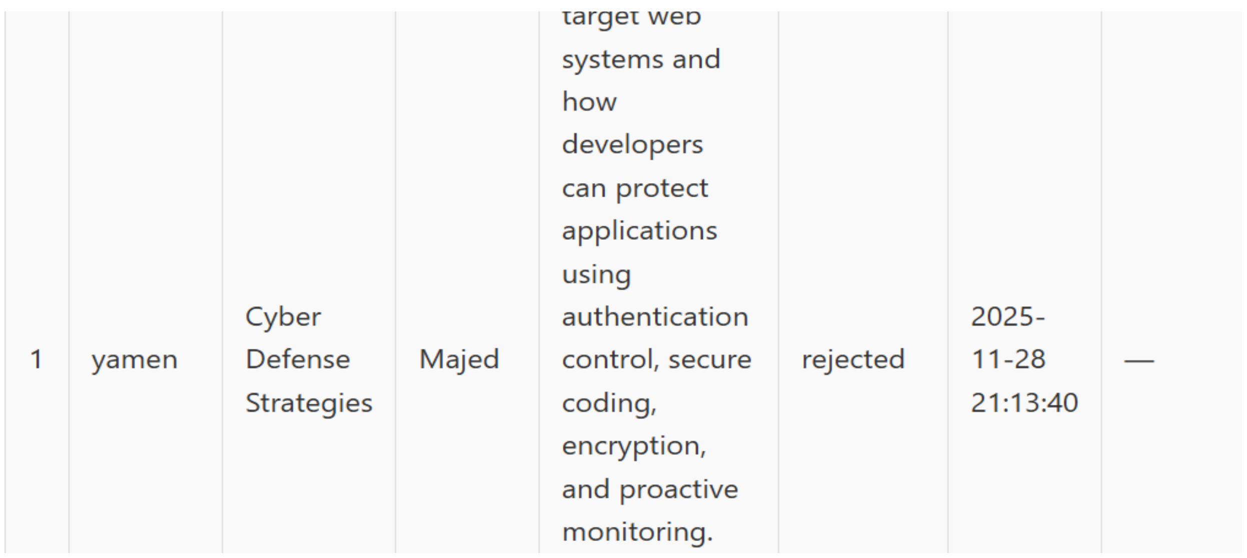
#	Student	Book Title	Author	Description	Status	Created At	Actions
2	yamen	CS	Majed	About Computer Science	pending	2025-11-28 22:29:15	Approve | Reject



Suggestions Inbox

Suggestion status updated successfully.

#	Student	Book Title	Author	Description	Status	Created At	Actions
2	yamen	CS	Majed	About Computer Science	approved	2025-11-28 22:29:15	—



1	yamen	Cyber Defense Strategies	Majed	target web systems and how developers can protect applications using authentication control, secure coding, encryption, and proactive monitoring.	rejected	2025-11-28 21:13:40	—