WEB SITE: LIBRARY

Project Report

TU856

Web Development 2

Aaron Baggot **Date: 8th December 2023**

C22716399 **TU856/2**

School of Computer Science

Technological University Dublin

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

Aaron Baggot

Aaron Baggot

C22716399

8th December 2023

Table of Contents

INTRODUCTION **4**

PROBLEM DESCRIPTION **5**

DESIGN CONSIDERATIONS **6**

USER AUTHENTICATION AND REGISTRATION **11**

FUNCTIONALITY OVERVIEW **16**

VISUAL PRESENTATION **19**

PAGINATION AND DATA DISPLAY **21**

WEBSITE TESTING AND OPTIMISATION REPORT 22

TESTING PLAN **23**

CONCLUSION **25**

PERSONAL LEARNING GAINED: 26

Appendix 1 ..... *27*

Appendix 2 -cancel_reservation *28*

Appendix 3 – connection *29*

Appendix 4 – footer *30*

Appendix 5 – header *31*

Appendix 6 – index *32*

Appendix 7 – login *33*

Appendix 8 – logout *34*

Appendix 9 – register *35*

Appendix 10 – register *36*

Appendix 11 – reservation_process *37*

Appendix 12 – search *38*

Appendix 13 – view_reservation *39*

BIBLIOGRAPHY **40**

Introduction

Building a website is a critical endeavour for any institution, and the development of a library website is no exception. This report outlines the process of scoping and designing a library website, which is a key component of the web development module within the TUD computer science program. The primary focus of the website is to create a user-friendly functional website that will enable the user to search for and reserve library books. Contained within Appendix 1 is an illustration of the design prototype used in designing the database and web pages.

Problem Description

In initial challenges encountered included the design and organisation of database tables. Key considerations encompassed the purpose of the website, required pages, content typology, and desired features. The development process involved the creation of a Low Fidelity Prototype, followed by iterative refinement through lecture notes and various research. Notable functionalities achieved include secure login with username and password verification, password restrictions, and the enforcement of unique usernames.

The overarching goal of this assignment being to showcase the development of a purpose-built website that is not only user-friendly but also fulfils the specific requirements outlined. The website is designed to provide users with seamless access to library resources, allowing for efficient book searches and reservations.

Design Considerations

When designing the website, several options and layouts were evaluated to ensure user-friendliness and optimal functionality. Prototyping played a crucial role in shaping the database structure, including the creation of tables using tools such as XAMPP and the formulation of SQL queries. The design prioritises simplicity and usability, adhering to the specified criteria of neat and straightforward presentation.

The website can perform actions such as updating reservation status and displaying book details by utilising SQL queries based on the table relationships.

Data Consistency:

Foreign key constraints ensure that relationships between tables are maintained, preventing inconsistencies in data.

The structure enables seamless interaction between the website and the database, supporting functionalities like user registration, book reservation, and category management.

The database design facilitates the functioning of the Book Reservation website. Key aspects include:

Book Reservations:

The books table (Image 1 & 2) stores book information, including reservation status, Category ID, ISBN number.

When a user reserves a book, a record is added to the Reservations table as illustrated in Image 3 and 4, linking the book's ISBN to the user's username.

Books

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**ISBN**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
2	**BookTitle**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
3	**Author**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
4	**Edition**	int(11)			Yes	*NULL*			Change Drop More
5	**Year**	int(11)			Yes	*NULL*			Change Drop More
6	**CategoryID**	int(11)			Yes	*NULL*			Change Drop More
7	**Reserved**	tinyint(1)			No	*None*			Change Drop More

Image 1

Show all | Number of rows: 50 Filter rows: Search this table Sort by key: None

Extra options

	ISBN	BookTitle	Author	Edition	Year	CategoryID	Reserved
Edit Copy Delete	012345678-9	Introduction to Machine Learning	Laura Johnson	2	2021	2	0
Edit Copy Delete	016886937-3	Funny People	Clair Jacketts	12	2008	7	0
Edit Copy Delete	037457042-6	Cup, The (Phôrpa)	Jdavie Dahler	15	2011	7	0
Edit Copy Delete	043705632-5	Advantages of attending Technology University Dubl...	Maurene Whitchurch	3	2011	4	0
Edit Copy Delete	080522135-2	Computer Science At TUD	Ashbey Verlinden	17	1991	4	1
Edit Copy Delete	093-403992	Computers in Business	Alicia Oneill	3	1997	3	1
Edit Copy Delete	123456789-0	Introduction to Computer Science	John Smith	1	2022	1	0
Edit Copy Delete	234567890-1	Data Structures and Algorithms	Jane Doe	2	2021	2	0
Edit Copy Delete	23472-8729	Exploring Peru	Stephanie Birching	4	2005	5	0
Edit Copy Delete	237-34823	Business Strategy	Joe Peppard	2	2002	2	0
Edit Copy Delete	23u8-923849	A guide to nutrition	John Thorpe	2	1997	1	0
Edit Copy Delete	2983-3494	Cooking for children	Anabelle Sharpe	1	2003	7	0
Edit Copy Delete	307727337-3	Trail of the Pink Panther	Ham Tither	5	1994	5	0
Edit Copy Delete	337423107-1	Michael Clayton	Trevar Codling	4	1999	6	0
Edit Copy Delete	345678901-2	Web Development Basics	Bob Johnson	1	2023	3	0
Edit Copy Delete	39892828-0	Common Man, A	Ginelle Thicking	13	2005	6	0
Edit Copy Delete	407521665-9	Don't Go Breaking My Heart (Daan gyun naam yu)	Gennifer Muckleston	4	2002	6	0
Edit Copy Delete	456789012-3	Artificial Intelligence: A Modern Approach	Alice Williams	3	2020	4	0

Console Bookmarks Options History Clear

Image 2

Book Reservations

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**ISBN**	varchar(20)	utf8mb4_general_ci		No	*None*			Change Drop More
2	**Username**	varchar(50)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
3	**ReservedDate**	date			Yes	*NULL*			Change Drop More

Image 3

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	ISBN	Username	ReservedDate
Edit Copy Delete	080522135-2	tommy100	2023-11-26
Edit Copy Delete	093-403992	tommy100	2023-11-26
Edit Copy Delete	80406632-4	tommy100	2023-11-26
Edit Copy Delete	9823-98345	tommy100	2008-10-11
Edit Copy Delete	98234-029384	joecrotty	2008-10-11

Check all *With selected:* Edit Copy Delete Export

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Image 4

Category Information:

The Category table provides details about different book categories, aiding in organising and presenting books on the website (Image 5 & 6).

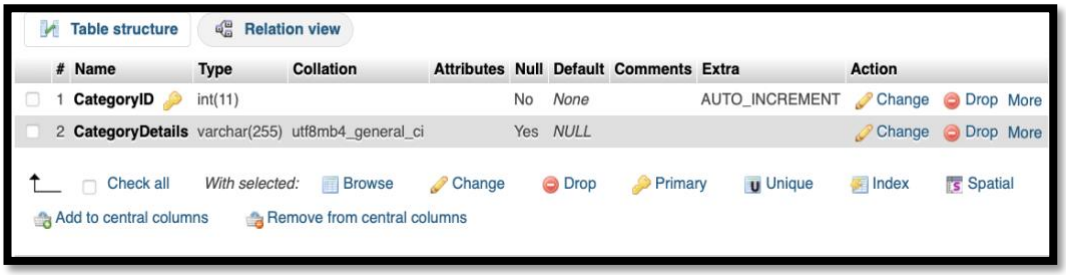


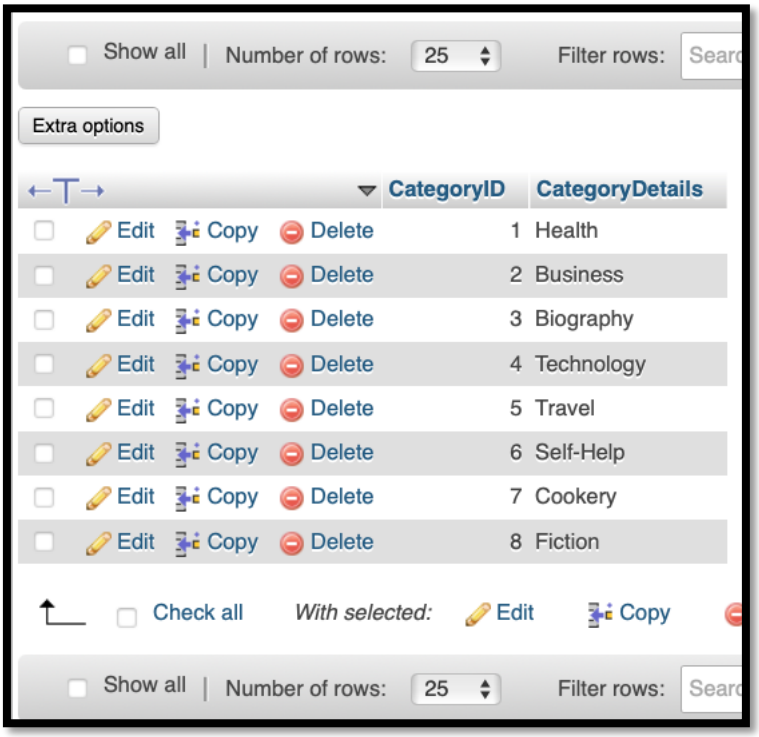
Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**CategoryID**	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
2	**CategoryDetails**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More

Check all *With selected:* Browse Change Drop Primary Unique Index Spatial

Add to central columns Remove from central columns

Image 5



Show all | Number of rows: 25 Filter rows: Searc

Extra options

	CategoryID	CategoryDetails
Edit Copy Delete	1	Health
Edit Copy Delete	2	Business
Edit Copy Delete	3	Biography
Edit Copy Delete	4	Technology
Edit Copy Delete	5	Travel
Edit Copy Delete	6	Self-Help
Edit Copy Delete	7	Cookery
Edit Copy Delete	8	Fiction

Check all *With selected:* Edit Copy

Show all | Number of rows: 25 Filter rows: Searc

Image 6

User Authentication:

The Users table stores user information, including usernames and passwords for authentication (Image 7 & 8).

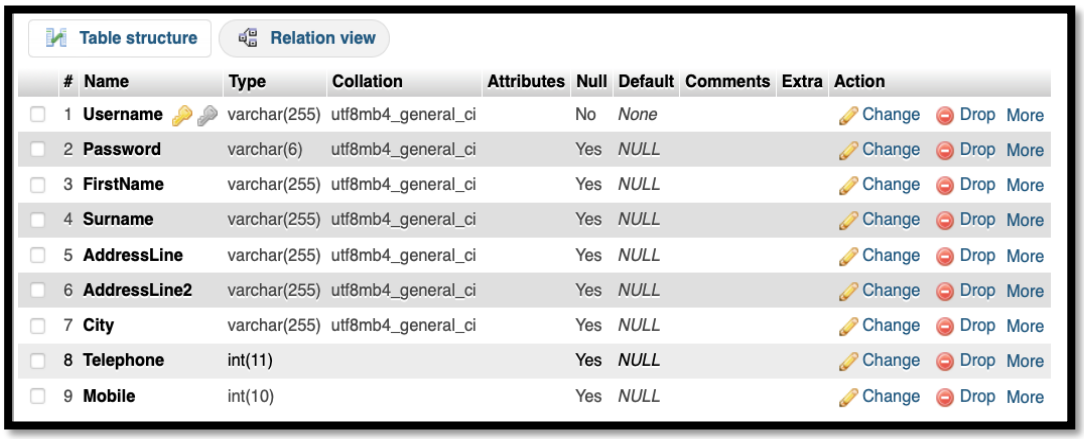
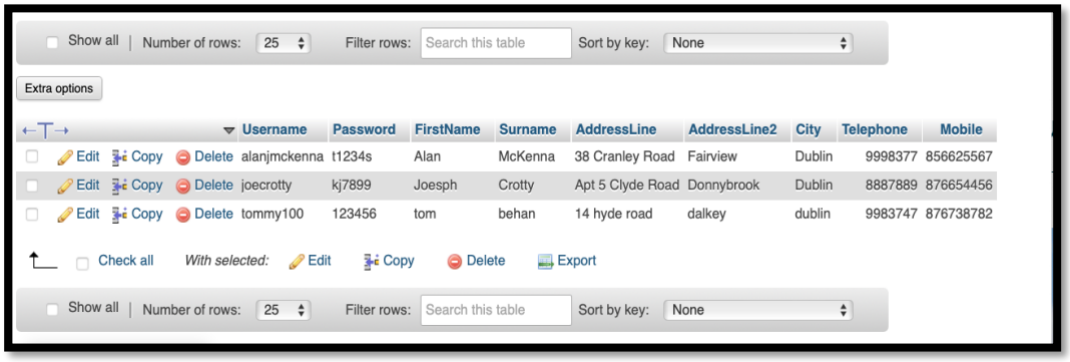


Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**Username**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
2	**Password**	varchar(6)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
3	**FirstName**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
4	**Surname**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
5	**AddressLine**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
6	**AddressLine2**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
7	**City**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
8	**Telephone**	int(11)			Yes	*NULL*			Change Drop More
9	**Mobile**	int(10)			Yes	*NULL*			Change Drop More

Image 7



Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	Username	Password	FirstName	Surname	AddressLine	AddressLine2	City	Telephone	Mobile
Edit Copy Delete	alanjmckenna	t1234s	Alan	McKenna	38 Cranley Road	Fairview	Dublin	9998377	856625567
Edit Copy Delete	joecrotty	kj7899	Joesph	Crotty	Apt 5 Clyde Road	Donnybrook	Dublin	8887889	876654456
Edit Copy Delete	tommy100	123456	tom	behan	14 hyde road	dalkey	dublin	9983747	876738782

Check all *With selected:* Edit Copy Delete Export

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Image 8

The BookReservationDB database design provides an organised and efficient structure for managing book reservations. The relationships between tables support the website's functionality, creating a robust system for users to interact with and reserve books. The integration ensures data consistency and facilitates a smooth user experience on the website.

User Authentication and Registration

The user authentication and registration functionality play a pivotal role in the Book Reservation website, aiming to provide a seamless experience for users. This section explores the key components of these features, implemented using PHP, HTML, and CSS.

User Authentication:

In the user authentication process, if the entered username is not recognised, an error message is displayed (Image 9), prompting the user to register by providing a link to the registration page; otherwise, if the username is recognised, a simple password comparison is performed, and if the password is incorrect, an appropriate error message is displayed.

Functionality:

The user is required to log in to access the website's features (Image 10).

- Upon submitting the login form, the provided credentials are validated against the stored information in the database.
- Successful login results in the creation of session variable ($_SESSION["Username"]) for the logged in user.
- Users are then redirected to the search page, ensuring a secure and authenticated session.

Image 9

```php
// Check if the user is found
if ($stmt->fetch()) {
    // Simple password comparison
    if ($password == $storedPassword) {
        // Set the session variable for the logged-in user
        $_SESSION["Username"] = $username;
        $stmt->close();

        // Redirect to the search page
        header("Location: search.php");
        exit(); // Make sure to exit after the header to prevent further execution
    } else {
        // Display an error message for incorrect credentials
        $error = "Incorrect Username or Password!";
    }
} else {
    // Display an error message for user not found
    $error = "User not found. You must Register";
}
```

Image 10

User Registration:

The registration process (Image 11) is robust, requiring users to enter valid details. Validation checks include numeric mobile phone numbers of 10 characters, a six-character password with confirmation, and the enforcement of unique usernames.

```
Project 2 > register.php
            exit();
        }

        $checkUsernameStmt->close();
    } else {
        echo "Error in username existence check preparation: " . $conn->error;
        exit();
    }

    // Check if password and confirm password match
    if ($ConfirmPassword !== $Password) {
        echo '<p>Password and Confirm Password do not match!! <a href="register.php">Go back to registration page</a></p>';
        ob_end_flush(); // Flush the output buffer and turn off output buffering
        exit();
    }

    // Validate password exactly 6 characters
    if (strlen($Password) !== 6) {
        echo '<p>Invalid Password!! Password must be exactly 6 characters in length. <a href="register.php">Go back to
        registration page</a></p>';
        ob_end_flush(); // Flush the output buffer and turn off output buffering
        exit();
    }

    // Validate mobile phone number
    $Mobile = $_POST['Mobile'];
    if (!is_numeric($Mobile) || strlen($Mobile) !== 10) {
        echo '<p>Invalid mobile phone number!! It should be numeric and 10 characters in length. <a href="register.php">Go
        back to registration page</a></p>';
        ob_end_flush(); // Flush the output buffer and turn off output buffering
        exit();
    }

    // Prepare and execute the SQL statement to insert user data
    $stmt = $conn->prepare("INSERT INTO Users (Username, Firstname, Surname, Password, AddressLine, AddressLine2, City,
    Telephone, Mobile) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)");
    $stmt->bind_param("sssssssss", $n, $f, $s, str_pad($Password, 6), $a, $b, $c, $t, $e);

    if ($stmt->execute()) {
        // Display a success message and a link to the login page
        echo '<p>Record added successfully. <a href="index.php">Go to login page</a></p>';
    } else {
        // Display an error message if there's an issue with the SQL execution
        echo "Error: " . $stmt->error;
    }

    // Close the statement
    $stmt->close();
}

// Flush the output buffer and turn off output buffering
ob_end_flush();
?>
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                    C Compiler Terminal
```

Image 11

Functionality:

- Users can register on the website by providing necessary details through the registration form (Image 12).
- Server-side validation ensures that all required details are entered correctly.
- Password validation ensures that the password is six characters long and matches the confirmation (Image 13).
- Mobile phone numbers are validated for numeric format and a length of 10 characters (Image 14).
- The system checks if the chosen username is unique, displaying an error message if it already exists (Image 15).
- Successful registration results in the insertion of user data into the Users table.

Image 12

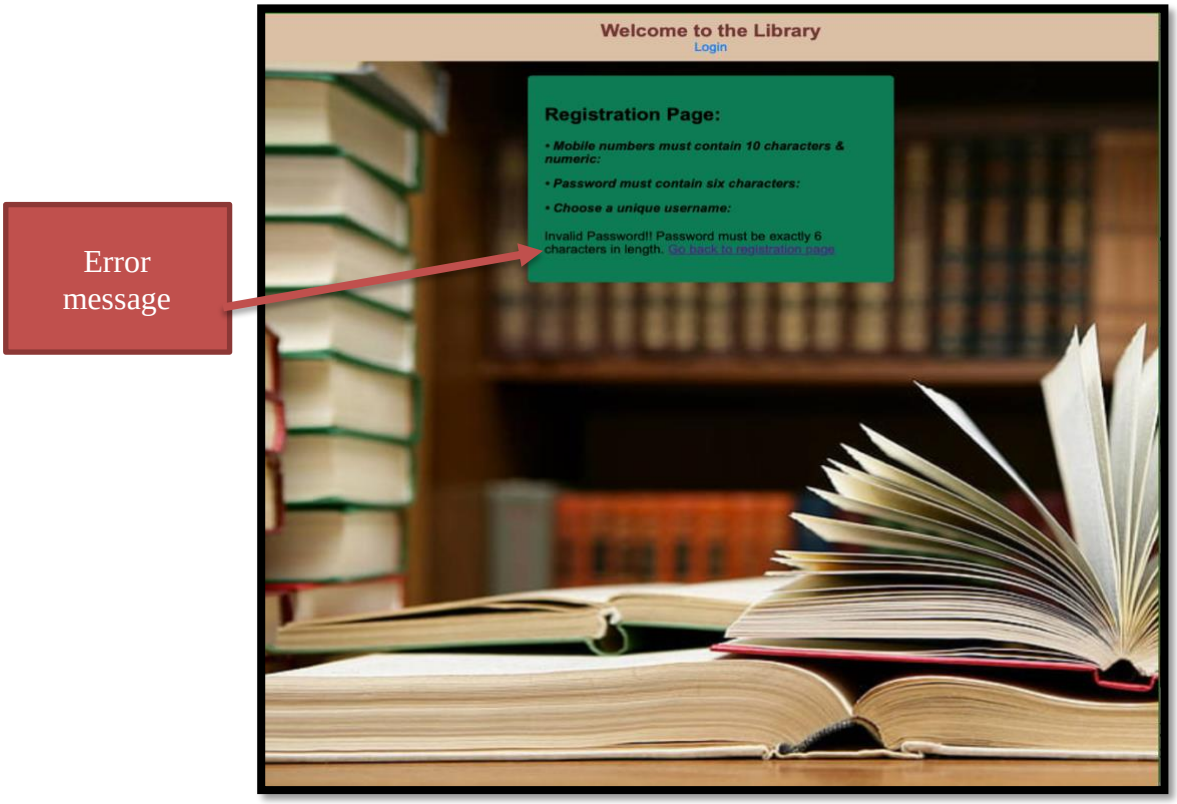


Image 13

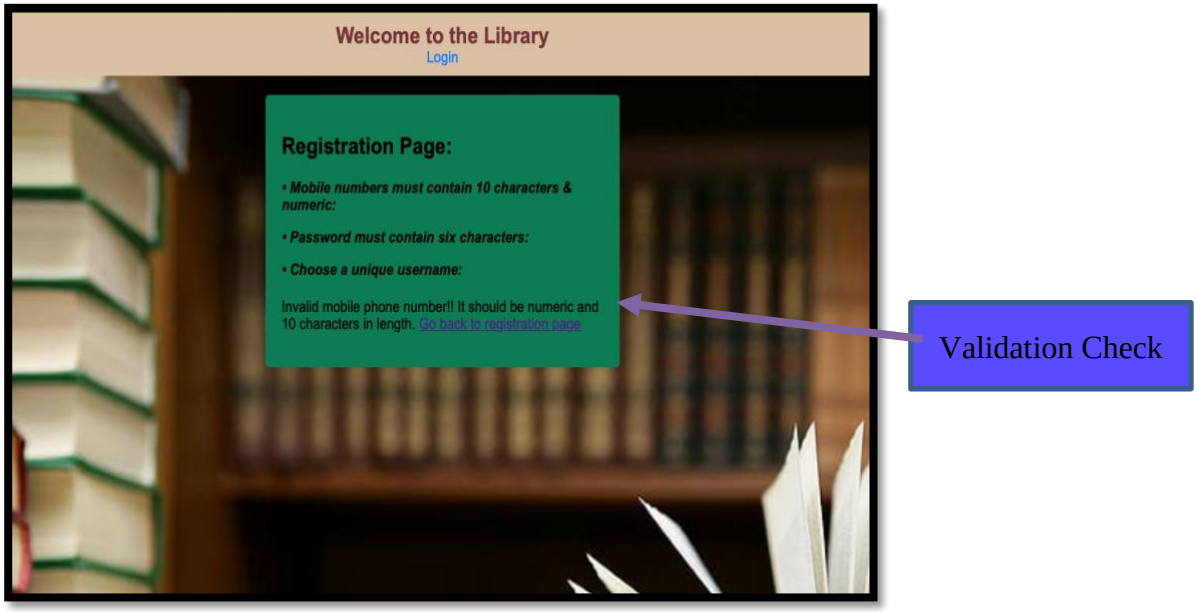


Image 14

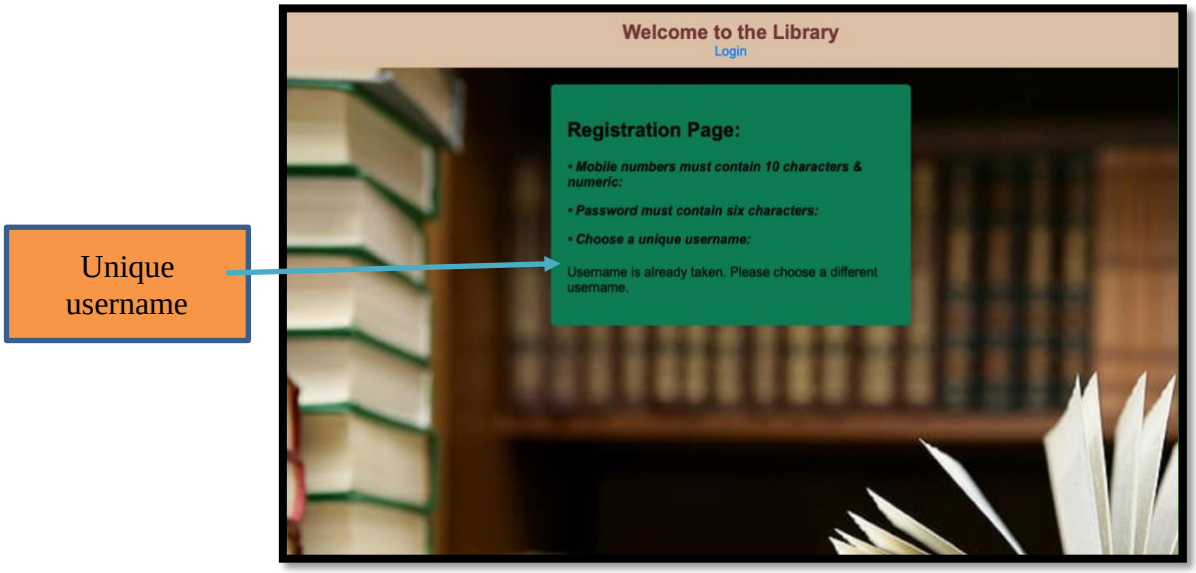

Unique username
Welcome to the Library
Login
Registration Page:
• Mobile numbers must contain 10 characters & numeric:
• Password must contain six characters:
• Choose a unique username:
Username is already taken. Please choose a different username.

Image 15

Functionality Overview

The website's functionality includes a comprehensive search feature that enables users to search for books by title, author, or category. The reservation system ensures that users can reserve available books, with the reservation date being recorded. Furthermore, users can view and manage their reserved books, including the option to remove reservations. Image 16 illustrates search by Title, Image 17 search by Author and Image 18, 18b search by Category in a dropdown menu.

Image 16

Image 17

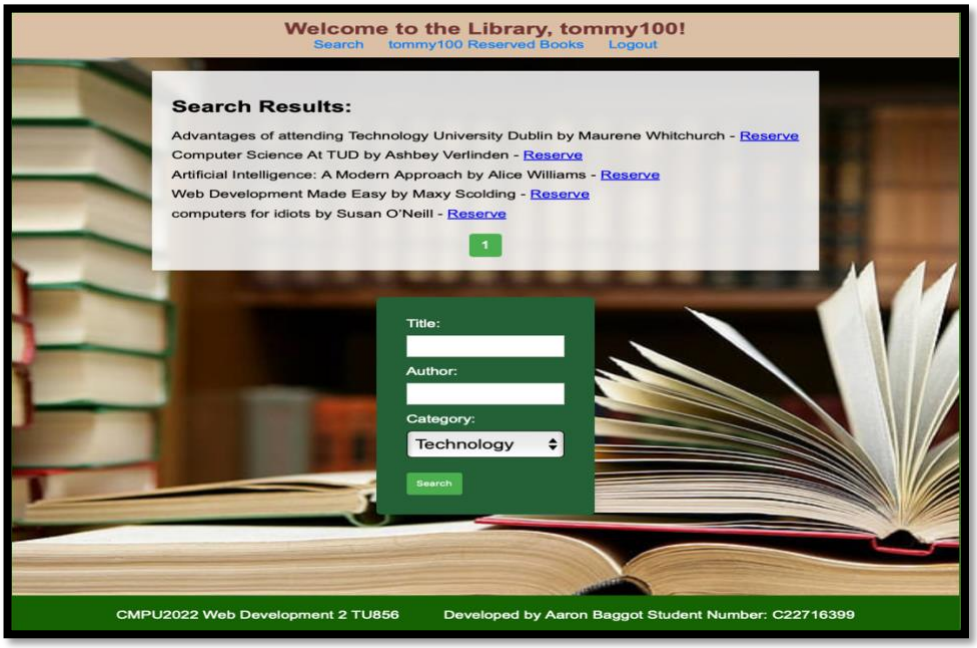


Image 18

Image 18b

Image 19 illustrates the Reserved book page whereby the current logged in user can view their books reserved by: Book title, Author, Edition, Year, Reserved date and an action to cancel reservation.

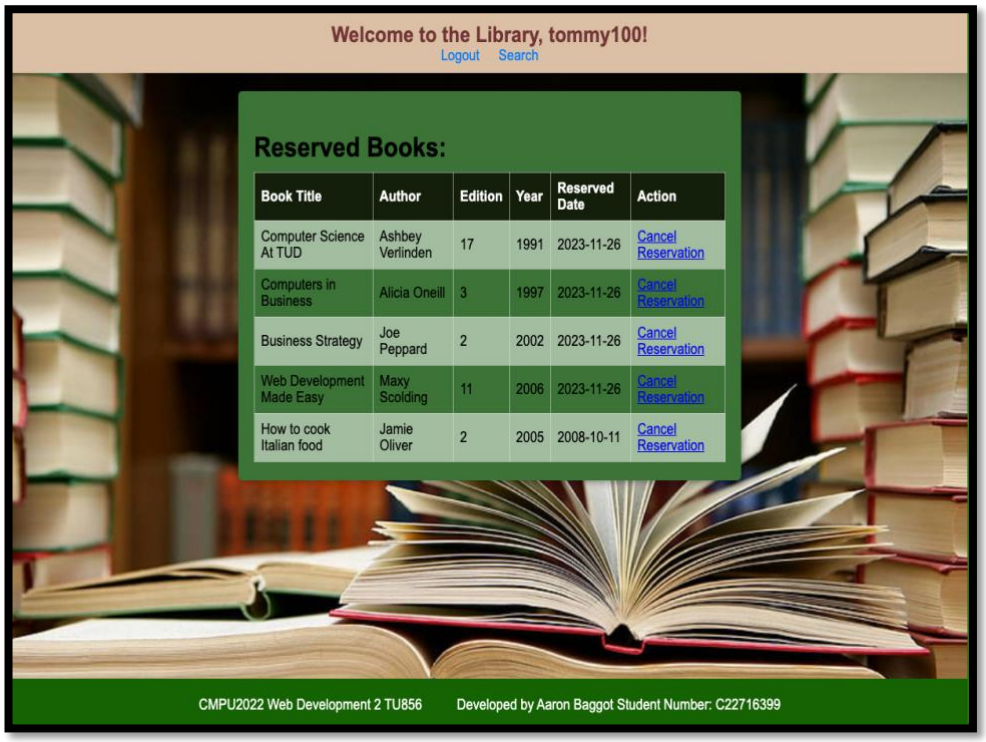


Image 19

Visual Presentation

In the comprehensive development plan for our library website, the use of CSS played a pivotal role in shaping the site's aesthetic appeal and user experience. The colour scheme, predominantly featuring calming greens and browns, along with a large, easily readable font, was chosen with the intention of creating a user-friendly interface accessible to individuals of all ages and abilities. Employing a simple and consistent CSS approach, each section had a designated heading, making it effortless to locate and modify fonts, styles, and positioning.

In line with the specified criteria, the website incorporates client-side technologies such as cascading style sheets (CSS) to enhance the visual appeal and organisation of pages. The design prioritises simplicity and functionality over-elaborate presentations.
cancel_reservation.php: This page handles the cancellation of book reservations. Upon accessing this page, users can view their currently reserved books and choose to cancel a reservation. The functionality is integrated with the database, updating the reservation status, and ensuring a smooth user experience.

connection.php: The connection.php file establishes a connection to the database using PHP. This essential file is included in all pages requiring database access, ensuring consistent and secure interactions between the website and the underlying database.

css/style.css: The style.css file, located in the css directory, defines the styling and layout for the entire website. It plays a crucial role in presenting a neat and user-friendly interface, aligning with the specified criteria of simplicity and usability.

footer.php: This PHP file contains the footer section that is included in each webpage. It contributes to a cohesive design by providing consistent footer elements throughout the site, enhancing the overall aesthetic appeal.

header.php: The header.php file encapsulates the header section present on every webpage. It includes essential elements like the site title, navigation links, and login/logout options, maintaining a standardised and organised layout across the website.

images: The images directory contains visual assets used throughout the website. The image contributes to the overall design, enhancing the visual appeal and potentially providing graphical representations of book categories or other relevant content.

index.php: The index.php file serves as the homepage of the website. It contains introductory information, login/register options, and a brief overview of the site's features. The homepage sets the tone for user engagement and navigation.
login.php: This page handles user login. Users are required to input their credentials, and upon submission, the system checks the information against the database for authentication. If successful, users are redirected to the search page; otherwise, appropriate error messages are displayed.

logout.php: The logout.php page facilitates user logout, ensuring a secure end to a user's session. Upon logging out, users are redirected to the login page, contributing to a smooth transition between different states of user interaction.

register.php: The register.php page allows new users to register by providing necessary details. The registration process includes validation checks for data integrity, such as ensuring unique usernames and specific criteria for mobile numbers and passwords.

reservation_process.php: This page handles the reservation process when a user decides to reserve a book. It interacts with the database, updating reservation statuses and capturing reservation dates, ensuring accurate and real-time information.

search.php: The search.php page enables users to search for books using various criteria such as title, author, or category. The results are displayed in a user-friendly format, allowing for efficient exploration and interaction.

view_reserved_books.php: This page displays a list of books currently reserved by the logged-in user. Users have the option to remove reservations, and the page interacts with the database to reflect changes in real-time.

Pagination and Data Display

To optimise user experience, the system implements pagination, limiting search results to a maximum of 5 rows per page. Lists are presented across multiple pages when necessary, ensuring a clean and efficient display of information. This PHP code initiates a loop to generate pagination links. It runs from page 1 to the total number of pages ($totalPages), which is 5 calculated based on the number of search results and the desired number of results per page. This array ($queryParams) holds the parameters for the pagination links. It includes the page number (‘page’) and any search criteria such as title, author, and category.

```
<div class='pagination'>
    <?php
    // Display pagination links
    for ($page = 1; $page <= $totalPages; $page++) {
        $queryParams = array(
            'page' => $page,
            'title' => $searchTitle,
            'author' => $searchAuthor,
            'category' => $searchCategory
        );
        $queryString = http_build_query($queryParams);
        $url = 'search.php?' . $queryString;

        echo "<a href='$url'>$page</a>";
    }
    ?>
</div>
```

Image 19

The code generates a set of pagination links, each linking to a specific page of search results, and allows users to navigate through the result pages. The URLs include the appropriate query parameters to maintain the search criteria during pagination.

Website Testing and Optimisation Report

Testing Across Browsers:

The website underwent comprehensive testing across popular browsers, including Chrome, Firefox, Safari, and Edge. The goal was to ensure consistent performance and appearance on different platforms. The website displayed uniformity in rendering across all browsers, with no notable issues identified. Compatibility checks were performed, and adjustments made where necessary to guarantee a seamless user experience regardless of the browser used.

Content Organisation Review:

An evaluation of the content organisation was conducted to optimise user navigation, especially considering potential future growth. The current structure ensures logical and intuitive access to information. Recommendations include periodically revisiting the site structure as content expands, potentially implementing subcategories or dynamic navigation features to accommodate a growing database of books.

Feedback Mechanism Implementation:

To enhance user engagement and address user concerns, family and friends of various ages gave feedback which aims to foster a collaborative environment, empowering users to contribute to the improvement of the website.

The testing and optimisation efforts undertaken contribute to the overall reliability and user-friendliness of the website. Regular reviews of content organisation and the incorporation of user feedback ensure adaptability to changing needs. As the website evolves, ongoing testing and optimisation practices will be maintained to uphold a high standard of performance and user satisfaction.

Testing Plan

To inspect the website the process of right-clicking on the page and selecting inspect displayed the compatibility with phones, iPad, laptop, and home pcs. Various browsers were also checked for compatibility the results showed a slight difference, but this did not impact significantly or a cause for concern. Throughout the development of the library website, a robust testing and debugging process was implemented to ensure the reliability and functionality of the PHP and HTML code. A systematic approach was adopted, encompassing various levels of testing, including unit testing, integration testing, and user acceptance testing.

At the unit testing level, individual components of both the PHP and HTML code were tested in isolation to verify their correctness and responsiveness. This involved scrutinising functions, forms, and other elements to ensure they operated as intended. PHP functions responsible for user authentication, registration, and reservation processing underwent thorough unit testing to identify and rectify any errors in the logic.

Integration testing played a crucial role in evaluating the interaction between different modules and components within the system. This involved examining how PHP scripts integrated with HTML pages, ensuring seamless communication and data flow. Additionally, the connection between the website and the underlying database was rigorously tested to validate data consistency and integrity.

The debugging process involved the meticulous identification and resolution of errors encountered during testing. PHP debugging tools, such as var_dump and echo statements, were strategically inserted into the code to trace the flow of variables and identify any unexpected behaviours. Similarly, browser developer tools were employed to inspect HTML elements and diagnose issues related to layout and styling.

User acceptance testing was conducted to gather feedback from potential end-users, with varying degrees of technological proficiency. This real-world testing helped

uncover usability issues, ensuring that the website met the needs and expectations of its intended audience.

Regular version control using tools like Git was employed throughout development, enabling me to roll back changes and track modifications, enhancing collaboration and reducing the risk of introducing new bugs. Overall, the testing and debugging strategies adopted were integral to the iterative development process, fostering a reliable and user-friendly library website.

Conclusion

Having completed this assignment it is evident that the successful design and implementation of the library website involves meticulous planning, with a specific focus on key features and user-friendliness. The project incorporated technologies such as XAMPP, PHP, HTML, and CSS, synergised through Visual Studio Code for streamlined code compilation. The development process commenced with a comprehensive description of the website and its content, considering the diverse user base.

Extensive research on existing websites provided valuable insights and inspired a sample layout characterised by a responsive design. To articulate the envisioned website, a Low Fidelity Prototype was crafted, serving as a guide. Subsequently, the project transitioned into the development and testing stage, encompassing rigorous validation checks and cross-browser compatibility assessments. This approach ensured the website's robustness and adaptability across various platforms.

The final phase involved evaluation by users with varying technical proficiencies, mirroring the diverse audience. This journey underscored the significance of strategic planning and consideration of user requirements in achieving a successful website design and development.

Having completed this assignment it is clear that an important aspect of designing a website is drafting a plan and making note of the most important features.

Personal Learning Gained:

The completion of the recent web development project has provided a valuable learning experience, encompassing various aspects such as research, compilation, and sourcing of project requirements. This endeavour has significantly enhanced my comprehension of web development principles, particularly in the realm of PHP and database queries. The project emphasised the critical role of adhering to best practices, including the implementation of secure password validation and the initiation of PHP sessions.

Throughout the project, understanding the synergy between core technologies, such as HTML and CSS, and incorporating PHP for server-side functionality, has become an integral part of my skill set. I learned to employ PHP for dynamic content generation, database interactions, and form processing, enabling a seamless integration of server-side logic into the website.

Lab exercises and comprehensive lecture notes provided valuable insights into the intricacies of developing a user-friendly website with robust backend functionality. Recognising the importance of creating an intuitive interface, I applied my knowledge to implement features like user authentication, session management, and database queries for efficient data retrieval and storage.

Moreover, the project equipped me with problem-solving skills, allowing me to analyse faults in PHP code, validate user inputs securely, and implement appropriate diagnostic tools and resources for thorough investigation. The incorporation of secure password validation, PHP sessions for user authentication, and effective database queries further enriched my understanding of web development practices.

In conclusion, this hands-on experience has not only broadened my technical capabilities but has also deepened my appreciation for the multifaceted aspects of effective web development, especially when incorporating PHP and database queries to create dynamic and secure web applications.

Appendix 1

Login Page:

- Username verification
- Password verification
- Link: Register Page
- Success Link: Search Page
- User logs in – Keep track with username

Header & Footer all Pages:
Footer:
Developed By
Header :
Links: Login, Logout, Search, Register.

Connection Page:
Server Connection
require_once "connection.php"

Reservation Page:

- Cancel Book
- Reserve Book
- Display date, name
- Display if already reserved
- Option cancel

View Reserved Books

- Author
- Title
- Date Reserved

Search Page:

- Dropdown Menu
- Health
- Business
- Biography
- Technology
- Travel
- Self-Help
- Cookery
- Fiction

Search By:
Author
Title
Category

Register Page:

- Details
- Name
- Address
- Password - 6 characters
- Mobile - 10 digits
- Username - unique

Database

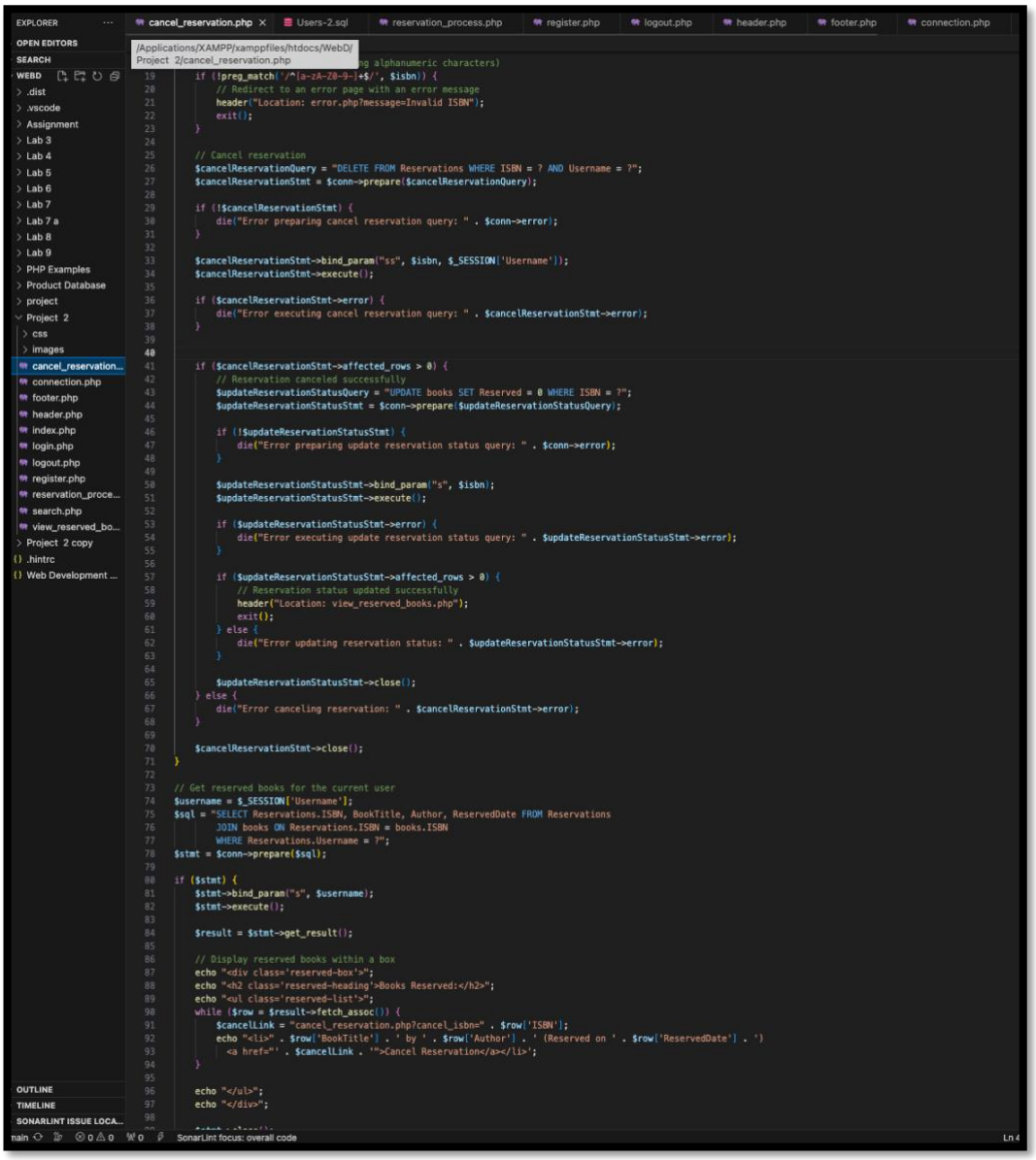
- Books Table
- *ISBN PK*
- *CategoryID FK*

- Categories Table
- *CategoryID PK*

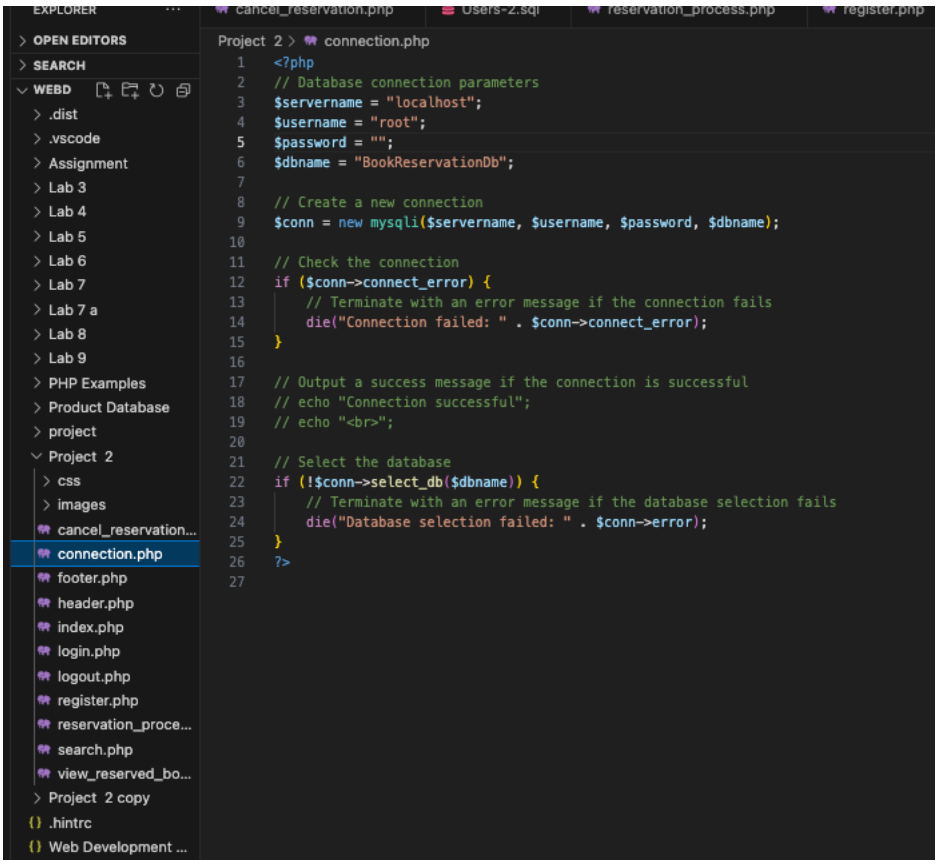
- Reservations
- *ISBN PK*
- *Username FK*

- Users Tables
- *Usernames PK*

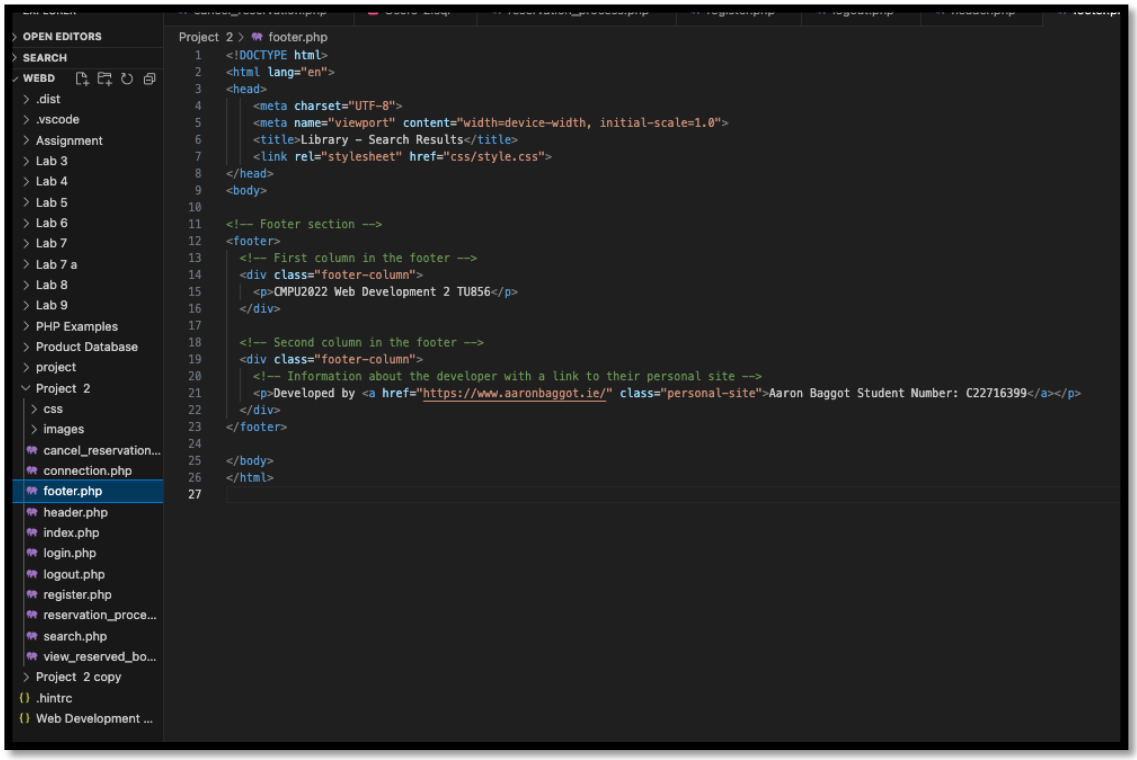
Appendix 2 -cancel_reservation



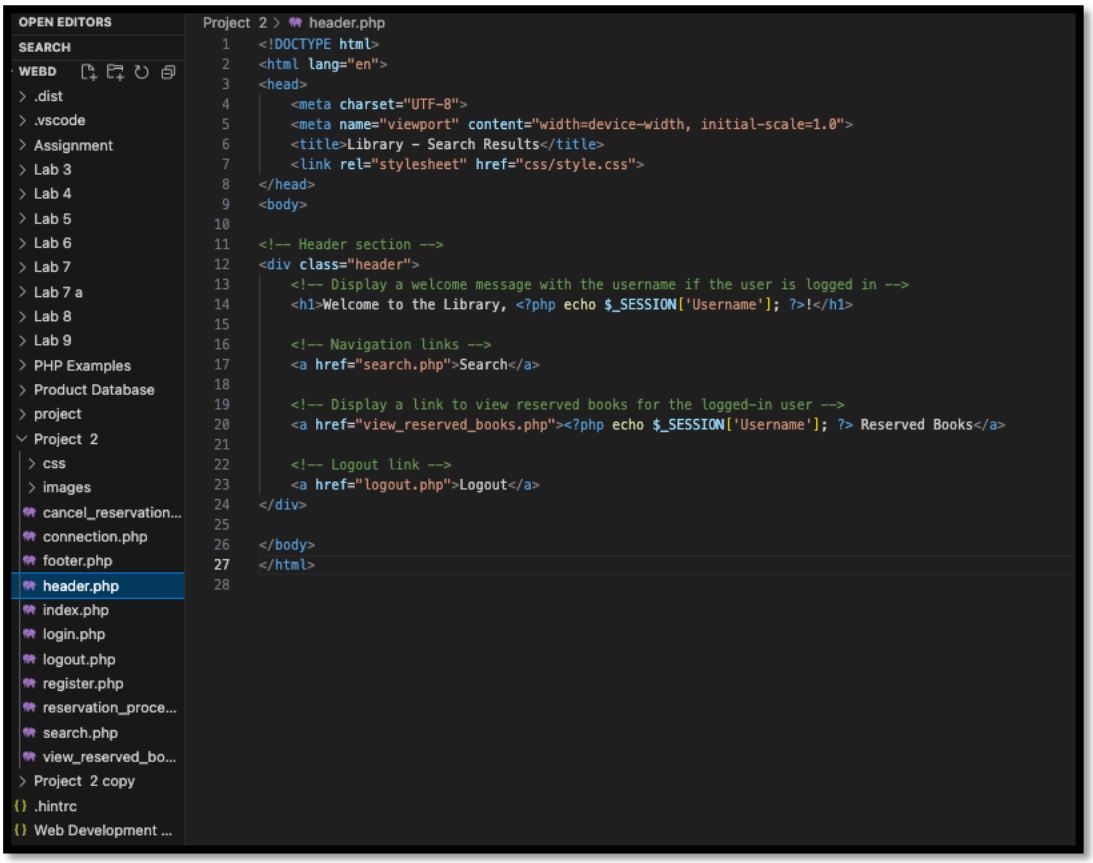
Appendix 3 – connection



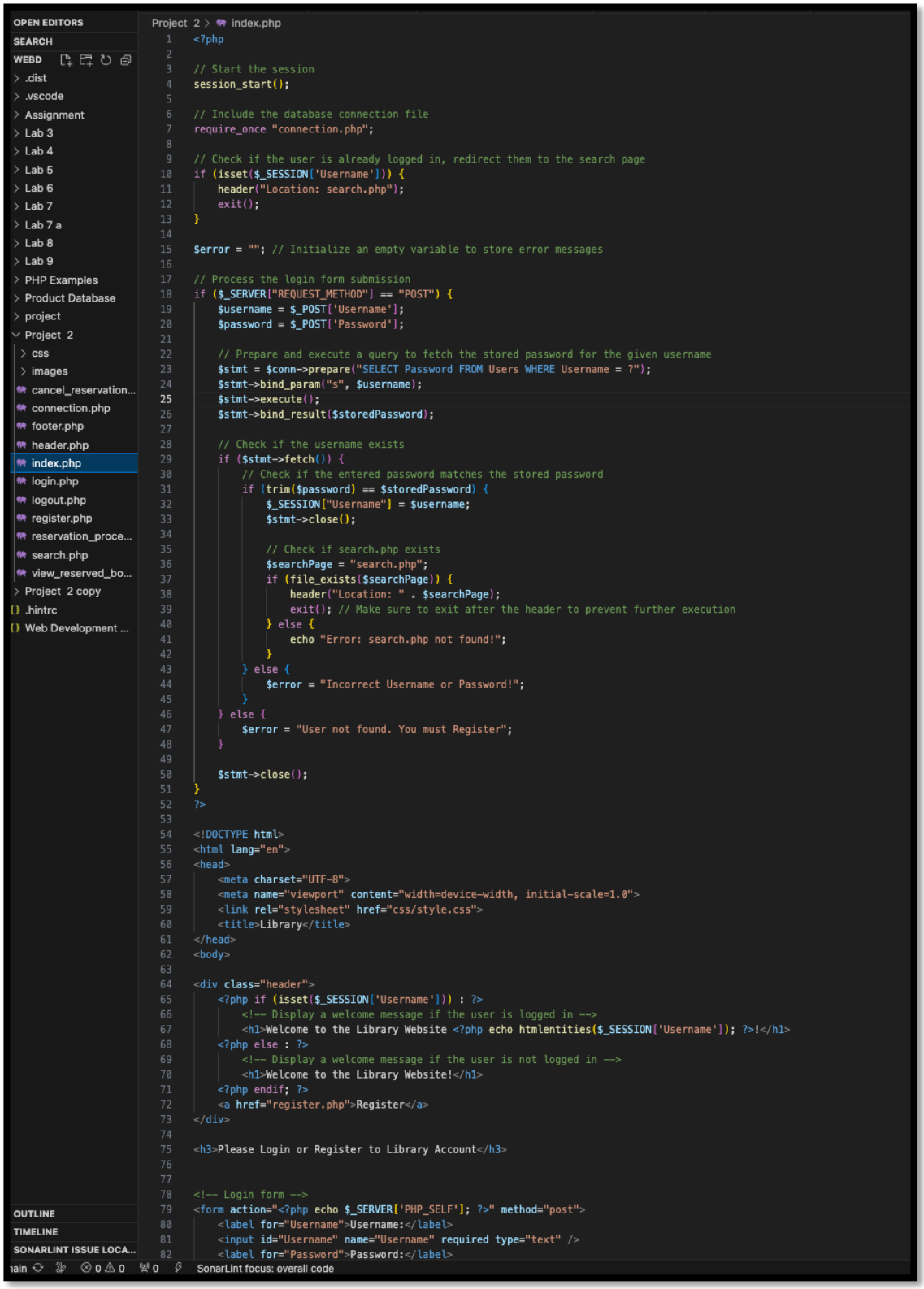
Appendix 4 – footer



Appendix 5 – header

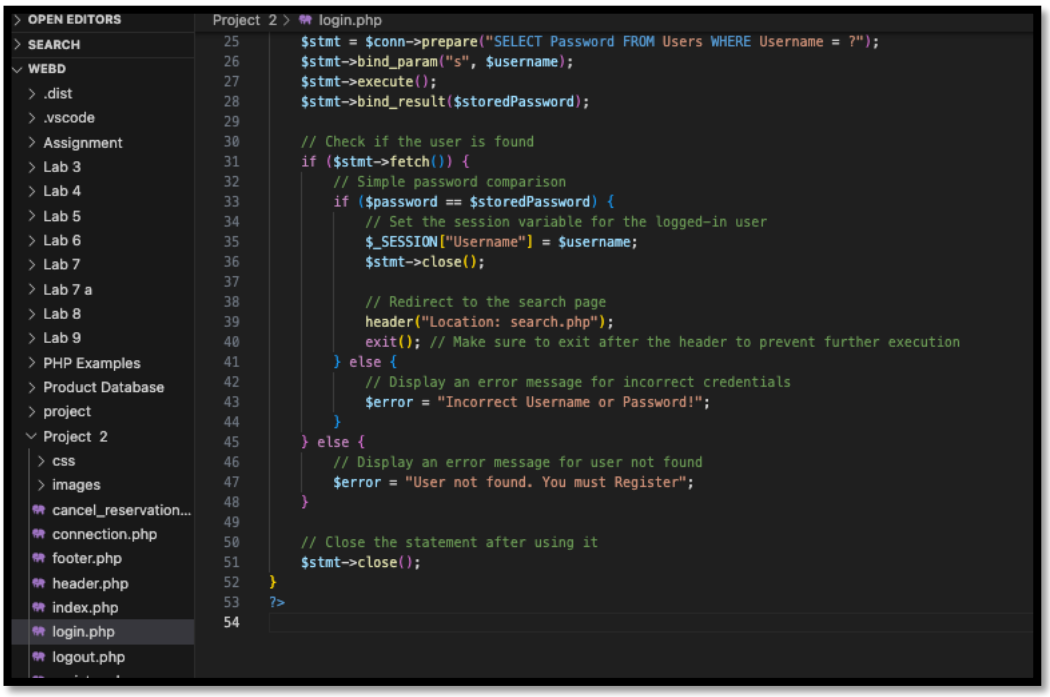


Appendix 6 – index

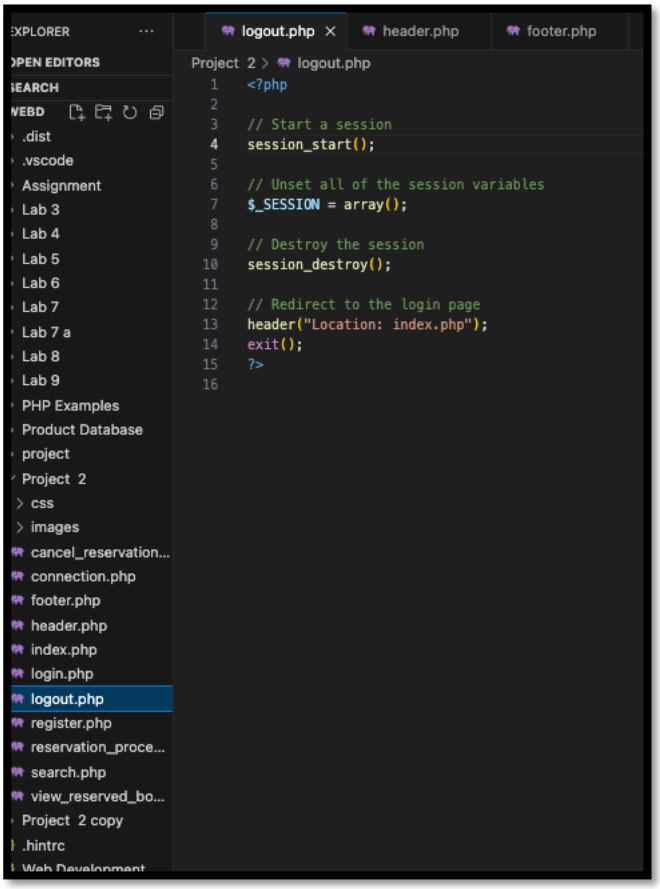



```php
<?php

// Start the session
session_start();

// Include the database connection file
require_once "connection.php";

// Check if the user is already logged in, redirect them to the search page
if (isset($_SESSION['Username'])) {
    header("Location: search.php");
    exit();
}

$error = ""; // Initialize an empty variable to store error messages

// Process the login form submission
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['Username'];
    $password = $_POST['Password'];

    // Prepare and execute a query to fetch the stored password for the given username
    $stmt = $conn->prepare("SELECT Password FROM Users WHERE Username = ?");
    $stmt->bind_param("s", $username);
    $stmt->execute();
    $stmt->bind_result($storedPassword);

    // Check if the username exists
    if ($stmt->fetch()) {
        // Check if the entered password matches the stored password
        if (trim($password) == $storedPassword) {
            $_SESSION["Username"] = $username;
            $stmt->close();

            // Check if search.php exists
            $searchPage = "search.php";
            if (file_exists($searchPage)) {
                header("Location: " . $searchPage);
                exit(); // Make sure to exit after the header to prevent further execution
            } else {
                echo "Error: search.php not found!";
            }
        } else {
            $error = "Incorrect Username or Password!";
        }
    } else {
        $error = "User not found. You must Register";
    }

    $stmt->close();
}
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/style.css">
    <title>Library</title>
</head>
<body>

<div class="header">
    <?php if (isset($_SESSION['Username'])) : ?>
        <!-- Display a welcome message if the user is logged in -->
        <h1>Welcome to the Library Website <?php echo htmlentities($_SESSION['Username']); ?>!</h1>
    <?php else : ?>
        <!-- Display a welcome message if the user is not logged in -->
        <h1>Welcome to the Library Website!</h1>
    <?php endif; ?>
    <a href="register.php">Register</a>
</div>

<h3>Please Login or Register to Library Account</h3>


<!-- Login form -->
<form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="post">
    <label for="Username">Username:</label>
    <input id="Username" name="Username" required type="text" />
    <label for="Password">Password:</label>
```

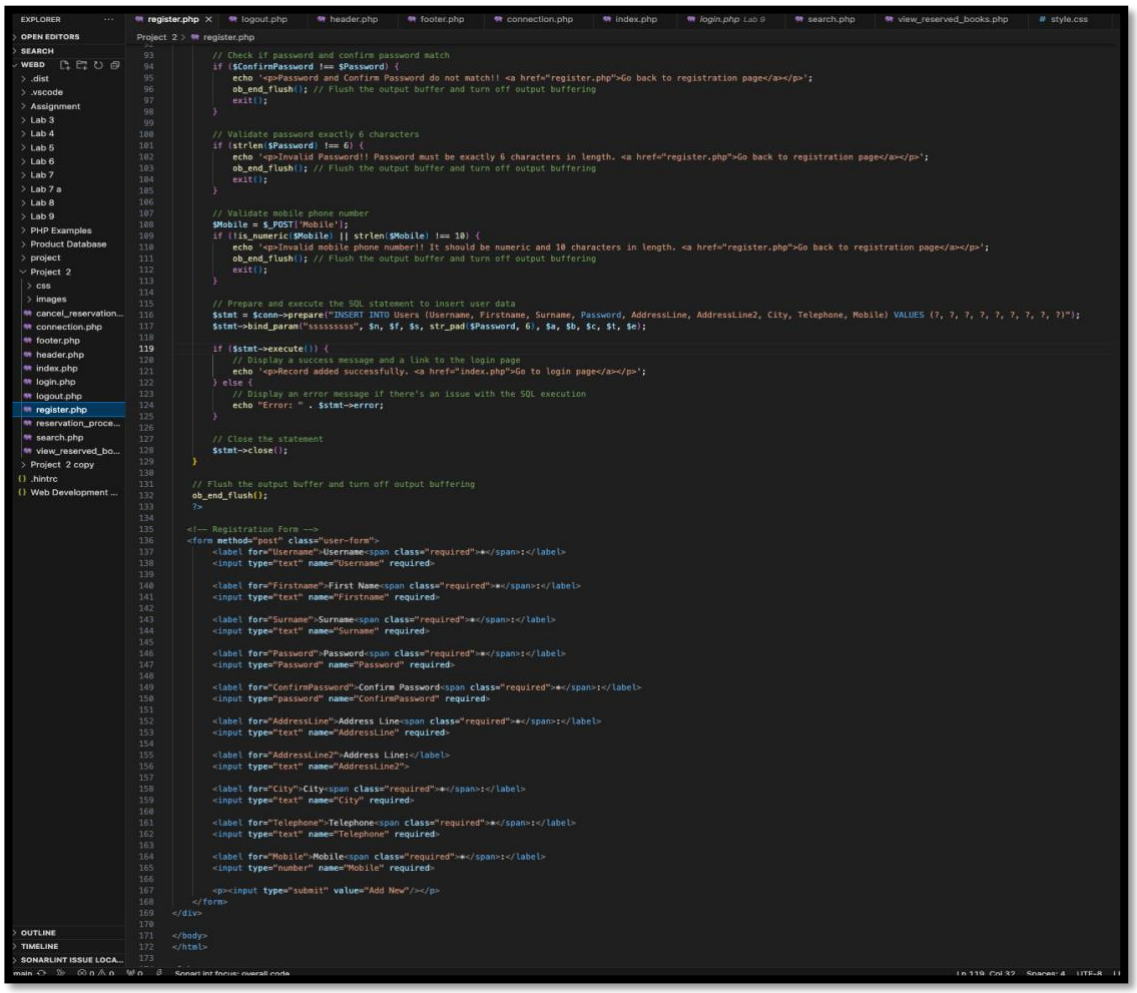
Appendix 7 – login



Appendix 8 – logout



Appendix 9 – register



```php
        // Check if password and confirm password match
        if ($ConfirmPassword !== $Password) {
            echo '<p>Password and Confirm Password do not match!! <a href="register.php">Go back to registration page</a></p>';
            ob_end_flush(); // Flush the output buffer and turn off output buffering
            exit();
        }

        // Validate password exactly 6 characters
        if (strlen($Password) !== 6) {
            echo '<p>Invalid Password!! Password must be exactly 6 characters in length. <a href="register.php">Go back to registration page</a></p>';
            ob_end_flush(); // Flush the output buffer and turn off output buffering
            exit();
        }

        // Validate mobile phone number
        $Mobile = $_POST['Mobile'];
        if (!is_numeric($Mobile) || strlen($Mobile) !== 10) {
            echo '<p>Invalid mobile phone number!! It should be numeric and 10 characters in length. <a href="register.php">Go back to registration page</a></p>';
            ob_end_flush(); // Flush the output buffer and turn off output buffering
            exit();
        }

        // Prepare and execute the SQL statement to insert user data
        $stmt = $conn->prepare("INSERT INTO Users (Username, Firstname, Surname, Password, AddressLine, AddressLine2, City, Telephone, Mobile) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)");
        $stmt->bind_param("sssssssss", $n, $f, $s, str_pad($Password, 6), $a, $b, $c, $t, $e);

        if ($stmt->execute()) {
            // Display a success message and a link to the login page
            echo '<p>Record added successfully. <a href="index.php">Go to login page</a></p>';
        } else {
            // Display an error message if there's an issue with the SQL execution
            echo "Error: " . $stmt->error;
        }

        // Close the statement
        $stmt->close();
    }

    // Flush the output buffer and turn off output buffering
    ob_end_flush();
    ?>

    <!-- Registration Form -->
    <form method="post" class="user-form">
        <label for="Username">Username<span class="required">*</span>:</label>
        <input type="text" name="Username" required>

        <label for="Firstname">First Name<span class="required">*</span>:</label>
        <input type="text" name="Firstname" required>

        <label for="Surname">Surname<span class="required">*</span>:</label>
        <input type="text" name="Surname" required>

        <label for="Password">Password<span class="required">*</span>:</label>
        <input type="Password" name="Password" required>

        <label for="ConfirmPassword">Confirm Password<span class="required">*</span>:</label>
        <input type="password" name="ConfirmPassword" required>

        <label for="AddressLine">Address Line<span class="required">*</span>:</label>
        <input type="text" name="AddressLine" required>

        <label for="AddressLine2">Address Line:</label>
        <input type="text" name="AddressLine2">

        <label for="City">City<span class="required">*</span>:</label>
        <input type="text" name="City" required>

        <label for="Telephone">Telephone<span class="required">*</span>:</label>
        <input type="text" name="Telephone" required>

        <label for="Mobile">Mobile<span class="required">*</span>:</label>
        <input type="number" name="Mobile" required>

        <p><input type="submit" value="Add New"/></p>
    </form>
</div>

</body>
</html>
```

Appendix 10 – register

EXPLORER

register.php | logout.php | header.php | footer.php | connection.php | index.php | logi

OPEN EDITORS
SEARCH
WEBD
- .dist
- .vscode
- Assignment
- Lab 3
- Lab 4
- Lab 5
- Lab 6
- Lab 7
- Lab 7 a
- Lab 8
- Lab 9
- PHP Examples
- Product Database
- project
- Project 2
  - css
  - images
  - cancel_reservation...
  - connection.php
  - footer.php
  - header.php
  - index.php
  - login.php
  - logout.php
  - register.php
  - reservation_proce...
  - search.php
  - view_reserved_bo...
- Project 2 copy
- .hintrc
- Web Development ...

OUTLINE
TIMELINE
SONARLINT ISSUE LOCA...

Project 2 > register.php

```php
<?php

// Start a session
session_start();

// Include the database connection file
require_once "connection.php";
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Library</title>
    <link rel="stylesheet" href="css/style.css">
</head>
<body>

<div class="header">
    <h1>Welcome to the Library</h1>
    <a href="index.php">Login</a>
</div>

<!-- Registration Form -->
<div class="form-container">
    <h2>Registration Page:</h2>

    <!-- Registration Instructions -->
    <h4>• Mobile numbers must contain 10 characters & numeric:
        <p>• Password must contain six characters:</p>
        • Choose a unique username:</h4>

    <?php
    // Use output buffering to prevent any output before the header function
    ob_start();

    // Check if the registration form is submitted
    if (
        isset($_POST['Username']) &&
        isset($_POST['Firstname']) &&
        isset($_POST['Surname']) &&
        isset($_POST['Password']) &&
        isset($_POST['ConfirmPassword']) &&
        isset($_POST['AddressLine']) &&
        isset($_POST['AddressLine2']) &&
        isset($_POST['City']) &&
        isset($_POST['Telephone']) &&
        isset($_POST['Mobile'])
    ) {
        // Sanitize and retrieve form data
        $n = htmlentities($_POST['Username']);
        $f = htmlentities($_POST['Firstname']);
        $s = htmlentities($_POST['Surname']);
        $Password = htmlentities($_POST['Password']);
        $ConfirmPassword = htmlentities($_POST['ConfirmPassword']);
        $a = htmlentities($_POST['AddressLine']);
        $b = htmlentities($_POST['AddressLine2']);
        $c = htmlentities($_POST['City']);
        $t = htmlentities($_POST['Telephone']);
        $e = htmlentities($_POST['Mobile']);

        // Check if the username is already taken
        $checkUsernameQuery = "SELECT COUNT(*) as count FROM Users WHERE Username = ?";
        $checkUsernameStmt = $conn->prepare($checkUsernameQuery);

        if ($checkUsernameStmt) {
            $checkUsernameStmt->bind_param("s", $n);
            $checkUsernameStmt->execute();
            $checkUsernameResult = $checkUsernameStmt->get_result();

            if ($checkUsernameResult) {
                $count = $checkUsernameResult->fetch_assoc()['count'];

                if ($count > 0) {
                    // Username already exists, display an error message
                    echo '<p>Username is already taken. Please choose a different username.</p>';
                    $checkUsernameStmt->close();
                    exit();
                }
            } else {
                echo "Error checking username existence: " . $checkUsernameStmt->error;
```

Appendix 11 – reservation_process

```php
<?php
// Start session
session_start();

// Include database connection file
require_once "connection.php";

// Check if the user is not logged in, redirect them to the login page
if (!isset($_SESSION['Username'])) {
    header("Location: login.php");
    exit();
}

// Logic for reserving a book
if (isset($_GET['isbn'])) {
    $isbn = $_GET['isbn'];

    // Validate ISBN as a sanitized URL
    $filteredIsbn = filter_var($isbn, FILTER_SANITIZE_URL);

    if ($filteredIsbn === false) {
        // Redirect to an error page with an error message
        header("Location: search.php?message=Invalid ISBN");
        exit();
    }

    // Check if the book is already reserved
    $checkReservationQuery = "SELECT Reserved FROM books WHERE ISBN = ?";
    $checkReservationStmt = $conn->prepare($checkReservationQuery);

    if ($checkReservationStmt) {
        $checkReservationStmt->bind_param("s", $filteredIsbn);
        $checkReservationStmt->execute();

        $currentReservation = $checkReservationStmt->get_result()->fetch_assoc();
        $checkReservationStmt->close();

        // Check if the book is already reserved
        if ($currentReservation['Reserved'] == 1) {
            header("Location: search.php?message=The book is already reserved by someone else.");
            exit();
        } else {
            // Insert a new reservation into the Reservations table
            $insertReservationQuery = "INSERT INTO Reservations (Username, ISBN, ReservedDate) VALUES (?, ?, NOW())";
            $insertReservationStmt = $conn->prepare($insertReservationQuery);

            if ($insertReservationStmt) {
                $insertReservationStmt->bind_param("ss", $_SESSION['Username'], $filteredIsbn);
                $insertReservationStmt->execute();

                if (!$insertReservationStmt->error) {
                    // Update the Reserved column in the books table to 1
                    $updateReservationStatusQuery = "UPDATE books SET Reserved = 1 WHERE ISBN = ?";
                    $updateReservationStatusStmt = $conn->prepare($updateReservationStatusQuery);

                    if ($updateReservationStatusStmt) {
                        $updateReservationStatusStmt->bind_param("s", $filteredIsbn);
                        $updateReservationStatusStmt->execute();

                        if (!$updateReservationStatusStmt->error) {
                            header("Location: search.php?message=The book has been successfully reserved!");
                            exit();
                        } else {
                            header("Location: search.php?message=Error updating books table: " . $updateReservationStatusStmt->error);
                            exit();
                        }

                        $updateReservationStatusStmt->close();
                    } else {
                        header("Location: search.php?message=Error preparing update statement: " . $conn->error);
                        exit();
                    }
                } else {
                    // Handle the error in inserting into the Reservations table
                    header("Location: search.php?message=Error in reservation: " . $insertReservationStmt->error);
                    exit();
                }

                $insertReservationStmt->close();
            } else {
                header("Location: search.php?message=Error preparing insert statement: " . $conn->error);
                exit();
```

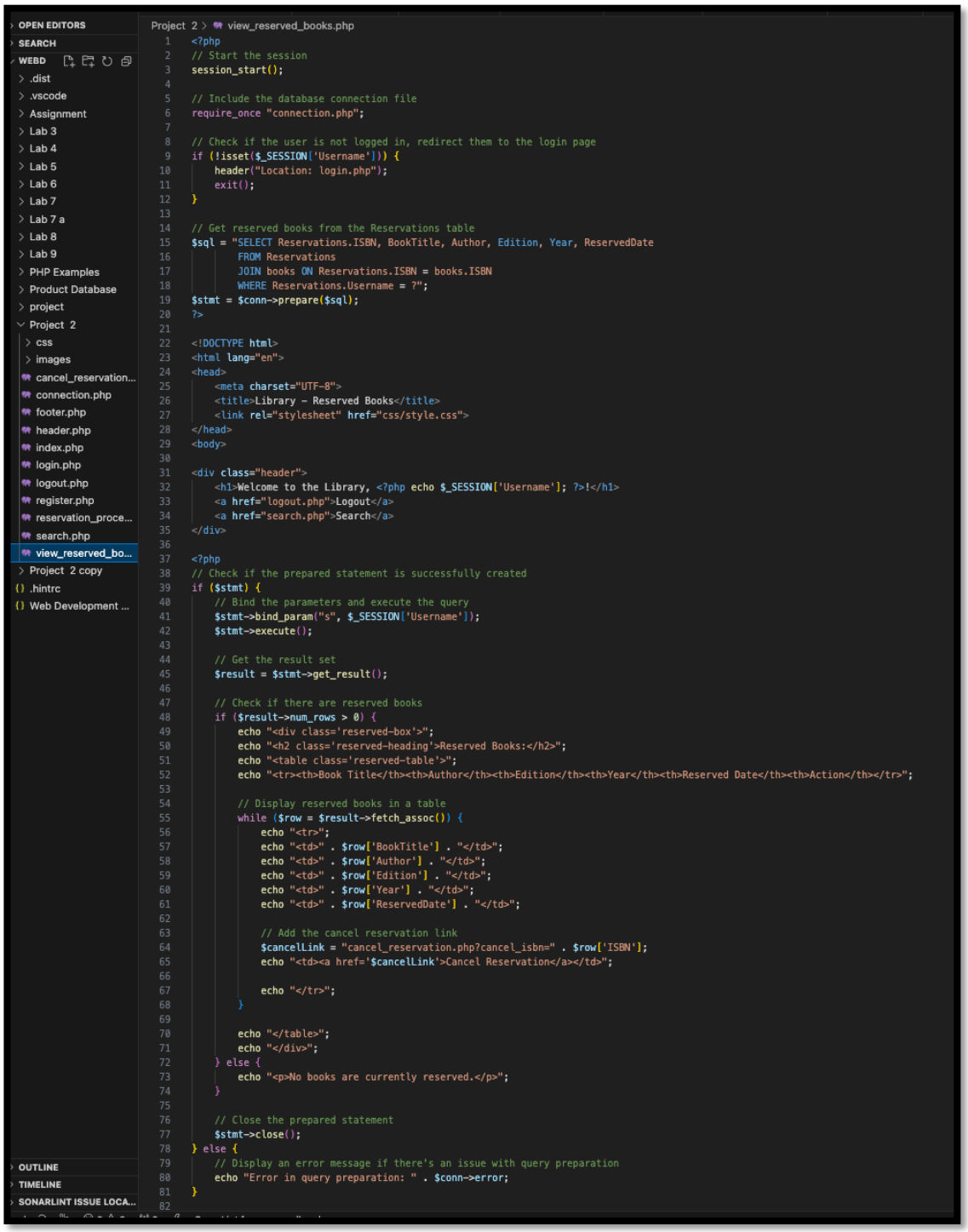
Appendix 12 – search

```
OPEN EDITORS
SEARCH
WEBD
  .dist
  .vscode
  Assignment
  Lab 3
  Lab 4
  Lab 5
  Lab 6
  Lab 7
  Lab 7 a
  Lab 8
  Lab 9
  PHP Examples
  Product Database
  project
  Project 2
    css
    images
    cancel_reservation...
    connection.php
    footer.php
    header.php
    index.php
    login.php
    logout.php
    register.php
    reservation_proce...
    search.php
    view_reserved_bo...
  Project 2 copy
  .hintrc
  Web Development ...
OUTLINE
TIMELINE
SONARLINT ISSUE LOCA...
```

Project 2 > search.php

```php

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Library - Search Results</title>
    <link rel="stylesheet" href="css/style.css">

</head>
<body>

<!-- Display reservation message if present -->
<?php if (!empty($reservationMessage)): ?>
    <div class="message-box"><?php echo $reservationMessage; ?></div>
<?php endif; ?>

<div class="books-container">
    <h2>Search Results:</h2>
    <ul>
        <?php
        // Display search results
        if ($result && $result->num_rows > 0) {
            while ($book = $result->fetch_assoc()) {
                echo "<li>" . $book['BookTitle'] . ' by ' . $book['Author'];

                // Create reservation link
                $reserveLink = 'reservation_process.php?isbn=' . $book['ISBN'] . '&reserved=true';
                echo ' - <a href="' . $reserveLink . '">Reserve</a>';

                echo "</li>";
            }
        } else {
            echo "No books found.";
        }
        ?>
    </ul>

    <div class='pagination'>
        <?php
        // Display pagination links
        for ($page = 1; $page <= $totalPages; $page++) {
            $queryParams = array(
                'page' => $page,
                'title' => $searchTitle,
                'author' => $searchAuthor,
                'category' => $searchCategory
            );
            $queryString = http_build_query($queryParams);
            $url = 'search.php?' . $queryString;

            echo "<a href='$url'>$page</a>";
        }
        ?>
    </div>
</div>

<!-- Search form -->
<form action="search.php" method="post">
    <label for="title">Title:</label>
    <input type="text" name="title" value="<?php echo $searchTitle; ?>">

    <label for="author">Author:</label>
    <input type="text" name="author" value="<?php echo $searchAuthor; ?>">

    <label for="category">Category:</label>
    <select name="category">
        <option value="" <?php echo ($searchCategory === '') ? 'selected' : ''; ?>>All Categories</option>
        <?php
        // Display category options
        $categoriesQuery = "SELECT CategoryID, CategoryDetails FROM Category";
        $categoriesStmt = $conn->prepare($categoriesQuery);
        $categoriesStmt->execute();
        $categoriesResult = $categoriesStmt->get_result();

        while ($row = $categoriesResult->fetch_assoc()) {
            // Format CategoryID with leading zeros
            $formattedCategoryID = sprintf('%03d', $row['CategoryID']);

            // Set the selected attribute based on the current search category
            $selected = ($formattedCategoryID == $searchCategory) ? 'selected' : '';
```

Appendix 13 – view_reservation




```php
<?php
// Start the session
session_start();

// Include the database connection file
require_once "connection.php";

// Check if the user is not logged in, redirect them to the login page
if (!isset($_SESSION['Username'])) {
    header("Location: login.php");
    exit();
}

// Get reserved books from the Reservations table
$sql = "SELECT Reservations.ISBN, BookTitle, Author, Edition, Year, ReservedDate
        FROM Reservations
        JOIN books ON Reservations.ISBN = books.ISBN
        WHERE Reservations.Username = ?";
$stmt = $conn->prepare($sql);
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Library - Reserved Books</title>
    <link rel="stylesheet" href="css/style.css">
</head>
<body>

<div class="header">
    <h1>Welcome to the Library, <?php echo $_SESSION['Username']; ?>!</h1>
    <a href="logout.php">Logout</a>
    <a href="search.php">Search</a>
</div>

<?php
// Check if the prepared statement is successfully created
if ($stmt) {
    // Bind the parameters and execute the query
    $stmt->bind_param("s", $_SESSION['Username']);
    $stmt->execute();

    // Get the result set
    $result = $stmt->get_result();

    // Check if there are reserved books
    if ($result->num_rows > 0) {
        echo "<div class='reserved-box'>";
        echo "<h2 class='reserved-heading'>Reserved Books:</h2>";
        echo "<table class='reserved-table'>";
        echo "<tr><th>Book Title</th><th>Author</th><th>Edition</th><th>Year</th><th>Reserved Date</th><th>Action</th></tr>";

        // Display reserved books in a table
        while ($row = $result->fetch_assoc()) {
            echo "<tr>";
            echo "<td>" . $row['BookTitle'] . "</td>";
            echo "<td>" . $row['Author'] . "</td>";
            echo "<td>" . $row['Edition'] . "</td>";
            echo "<td>" . $row['Year'] . "</td>";
            echo "<td>" . $row['ReservedDate'] . "</td>";

            // Add the cancel reservation link
            $cancelLink = "cancel_reservation.php?cancel_isbn=" . $row['ISBN'];
            echo "<td><a href='$cancelLink'>Cancel Reservation</a></td>";

            echo "</tr>";
        }

        echo "</table>";
        echo "</div>";
    } else {
        echo "<p>No books are currently reserved.</p>";
    }

    // Close the prepared statement
    $stmt->close();
} else {
    // Display an error message if there's an issue with query preparation
    echo "Error in query preparation: " . $conn->error;
}
```

Bibliography

Adams, D. (2023) *Secure registration system with PHP and mysql*, *CodeShack*. Available at: https://codeshack.io/secure-registration-system-php-mysql/ (Accessed: 19 November 2023).

Clinton, D. (2023) *How to create a simple HTML and PHP form*, *freeCodeCamp.org*. Available at: https://www.freecodecamp.org/news/creating-html-forms/ (Accessed: 11 November 2023).

Copes, F. (2022) The PHP handbook – learn PHP for Beginners, freeCodeCamp.org. Available at: https://www.freecodecamp.org/news/the-php-handbook/ (Accessed: 18 November 2023).

Learn PHP - free interactive PHP tutorial (no date) Learn PHP - Free Interactive PHP Tutorial. Available at: https://www.learn-php.org/ (Accessed: 11 November 2023).

Liu, C. (2023) ‘L7 -FormPHP’. ‘L6 – MYSQL’, ‘L8 -SessionPHP’, Available at: https://brightspace.tudublin.ie/d2l/(Accessed: 10 November 2023).

Moraes , F. (2020) Home, ". Available at: https://html.com/ (Accessed: 05 November 2023).

PHP examples. Available at: https://www.w3schools.com/php/php_examples.asp (Accessed: 18 November 2023).

PHP tutorial (no date) Online Tutorials, Courses, and eBooks Library. Available at: https://www.tutorialspoint.com/php/index.htm (Accessed: 18 November 2023).