# Web Development 2 CA

## Anmool Shah | TU856-2

**Project:** Library Book Reservation System
**TA/Lecturer:** Cindy Liu
**Github Link to Code:** Library System

For this assignment, I will be using PHP and a MYSQL database to develop a website which will function as a book reservation website for a library. The user should be able to sign up and create an account, or log into an already existing account. When registering, the process should validate all the details entered by the user e.g. a unique username is required.

The user should be able to search for books based on book title or author name. Alternatively, using a dropdown menu, the user will be able to find a book through different categories ie the book genre. The user will also be able to view all the books they have reserved already and remove any reservations through the same menu. Only one user may reserve a book at a time.

### Design Process

The main functions will be:

- Create Account
- Log in
- Log out
- Search for a book by book title
- Search for a book by author's name
- Search for a book by ISBN
- Search for a book by genre
- Reserve a book
- View reserved books

The tools that will be used:

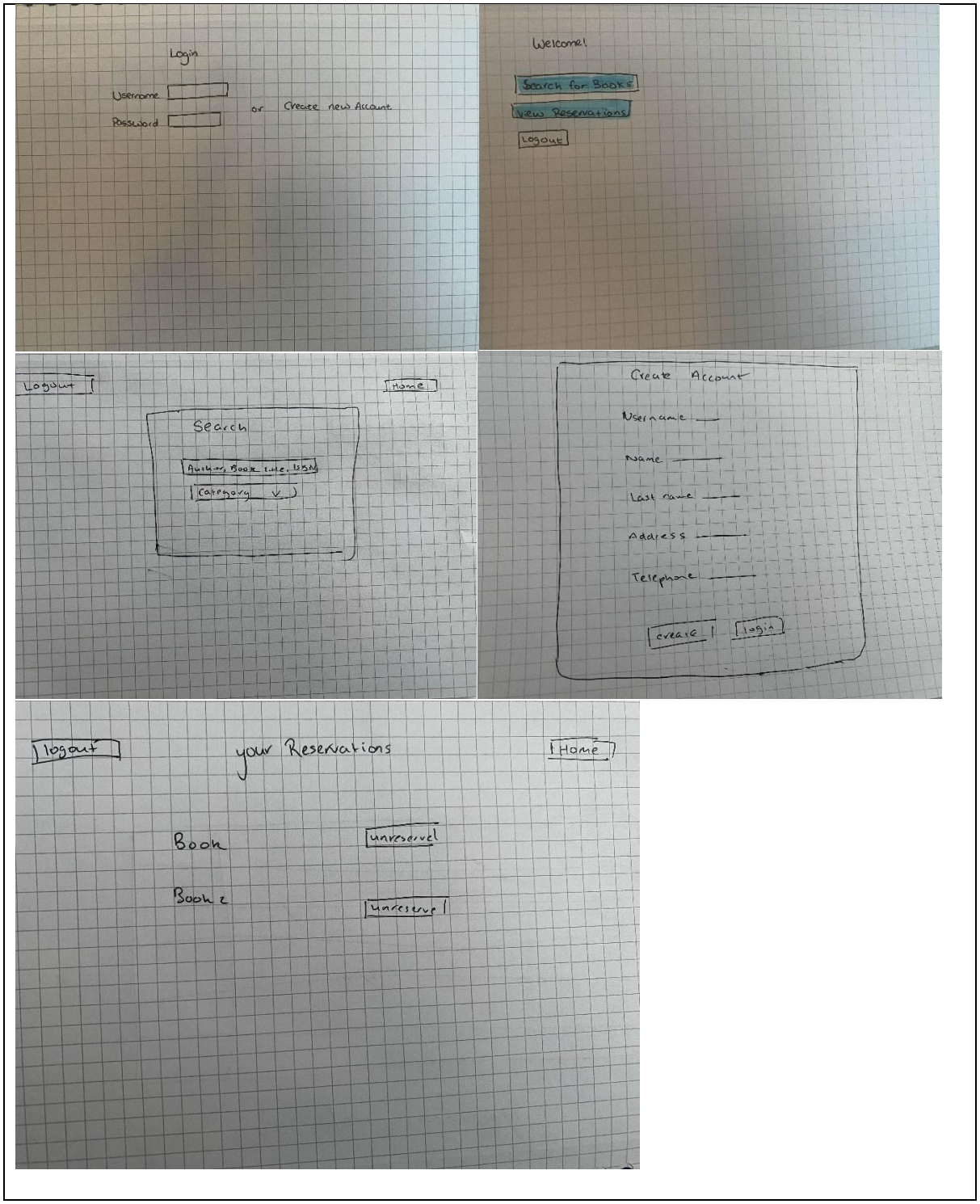
- Backend: Apache (Web server), PHP (Server side scripting)
- Frontend: HTML, CSS
- Database: MySQL

What should be noted:

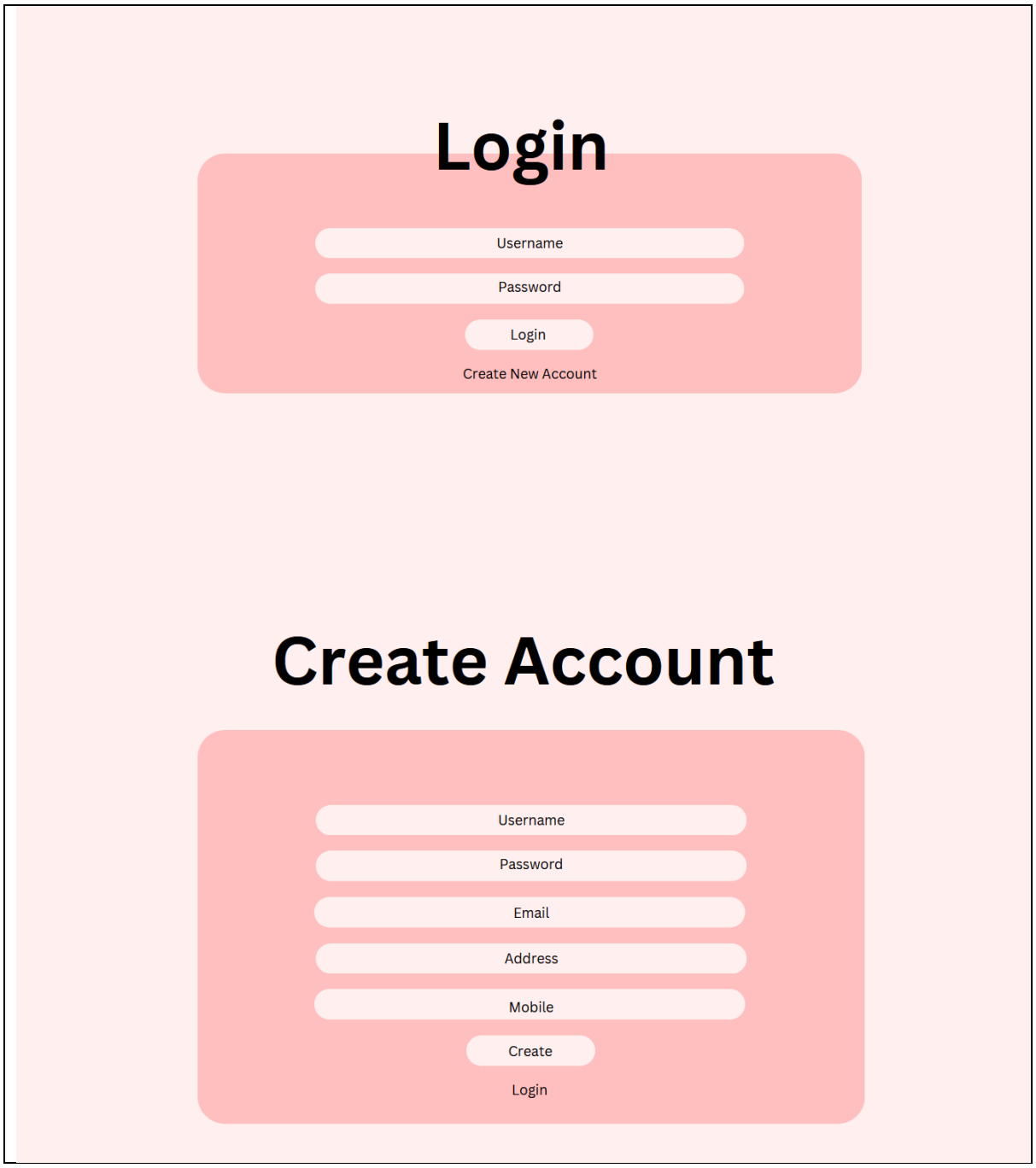
- Emphasise consistency to avoid errors between the PHP and database.
- Use the correct names from the tables to avoid errors.

- Use the correct file names to avoid errors.
- Make sure the user fills all the fields for account creation.
- Make sure the style is consistent.
- Provide visual feedback e.g. when the account is created the user should get a message.
- Use meaningful error messages.

Design Ideas on Paper:



After creating my design on paper, I used Canva to give myself a better idea of what to make the website look like.



# Welcome

Search Books

Your reservations

Logout

# Search

Search Books by Author, Title or ISBN

Select Genre v

Home

Logout

# Your Reservations

| ISBN | Title | Author | Edition | Year | Category | Action |
|---|---|---|---|---|---|---|
| 1 | A | V | 1 | 2024 | TECH | UNRESERVE |
| 2 | B | W | 1 | 2024 | COOKING | UNRESERVE |
| 3 | C | X | 1 | 2024 | BUSINESS | UNRESERVE |
| 4 | D | Y | 1 | 2024 | HEALTH | UNRESERVE |
| 5 | E | Z | 1 | 2024 | FICTION | UNRESERVE |

**<- PREV 1 2 3 NEXT ->**

# Search Results

| ISBN | Title | Author | Edition | Year | Category | Reserved | Action |
|---|---|---|---|---|---|---|---|
| 1 | A | V | 1 | 2024 | TECH | YES | |
| 2 | B | W | 1 | 2024 | COOKING | NO | RESERVE |
| 3 | C | X | 1 | 2024 | BUSINESS | YES | |
| 4 | D | Y | 1 | 2024 | HEALTH | NO | RESERVE |
| 5 | E | Z | 1 | 2024 | FICTION | YES | |

**<- PREV 1 2 3 NEXT ->**

After creating my design, I started on the database.

I named the database ‘calibrary’. This database contains 4 tables:

- books
- users
- category
- reservations

The books table contains the headings:

- ISBN
- BookTitle
- Author
- Edition
- Year
- Category
- Reserved
- ReservedBy

The users table contains the headings:

- Username
- Email
- Pass
- Firstname
- Surname
- AddressLine1
- AddressLine2
- City
- Telephone
- Mobile

The category table contains the headings:

- CategoryID
- CategoryDescription

The reservations table contains the headings:

- ISBN
- Username
- ReservedDate

I then populated the tables.

**Table: books**

| ISBN | BookTitle | Author | Edition | Year | Category | Reserved | ReservedBy |
|---|---|---|---|---|---|---|---|
| 093-403992 | Computers in Business | Alicia Oneill | 3 | 1997 | 4 | 0 | *NULL* |
| 23472-8729 | Exploring Peru | Stephanie Birch | 4 | 2005 | 3 | 0 | *NULL* |
| 237-34823 | Business Strategy | Joe Peppard | 2 | 2002 | 2 | 0 | *NULL* |
| 23u8-923849 | A guide to nutrition | John Thorpe | 2 | 1997 | 1 | 0 | *NULL* |
| 2983-3494 | Cooking for children | Anabelle Sharpe | 1 | 2003 | 7 | 0 | *NULL* |
| 82n8-308 | Computers for idiots | Susan O'Neill | 5 | 1998 | 4 | 0 | *NULL* |
| 9781785658709 | Gallant | V. E. Schwab | 1 | 2022 | 8 | 0 | *NULL* |
| 9823-23984 | My life in picture | Kevin Graham | 8 | 2004 | 1 | 0 | *NULL* |
| 9823-2403-0 | DaVinci Code | Dan Brown | 1 | 2003 | 8 | 0 | *NULL* |
| 9823-98345 | How to cook Italian food | Jamie Oliver | 2 | 2005 | 7 | 0 | *NULL* |
| 9823-98487 | Optimising your business | Cleo Blair | 1 | 2002 | 1 | 0 | *NULL* |
| 98234-029384 | My ranch in Texas | George Bush | 1 | 2005 | 3 | 0 | *NULL* |
| 988745-234 | Tara Road | Maeve Binchy | 4 | 2002 | 8 | 0 | *NULL* |
| 993-004-00 | My life in bits | John Smith | 1 | 2001 | 3 | 0 | *NULL* |
| 9987-0039882 | Shooting History | Jon Snow | 1 | 2003 | 1 | 0 | *NULL* |

**Table: category**

| CategoryID | CategoryDescription |
|---|---|
| 1 | Health |
| 2 | Business |
| 3 | Biography |
| 4 | Technology |
| 5 | Travel |
| 6 | Self-help |
| 7 | Cookery |
| 8 | Fiction |

The users and reservations tables haven't been populated before hand as the data will go into the tables when the user creates an account or reserves a book.

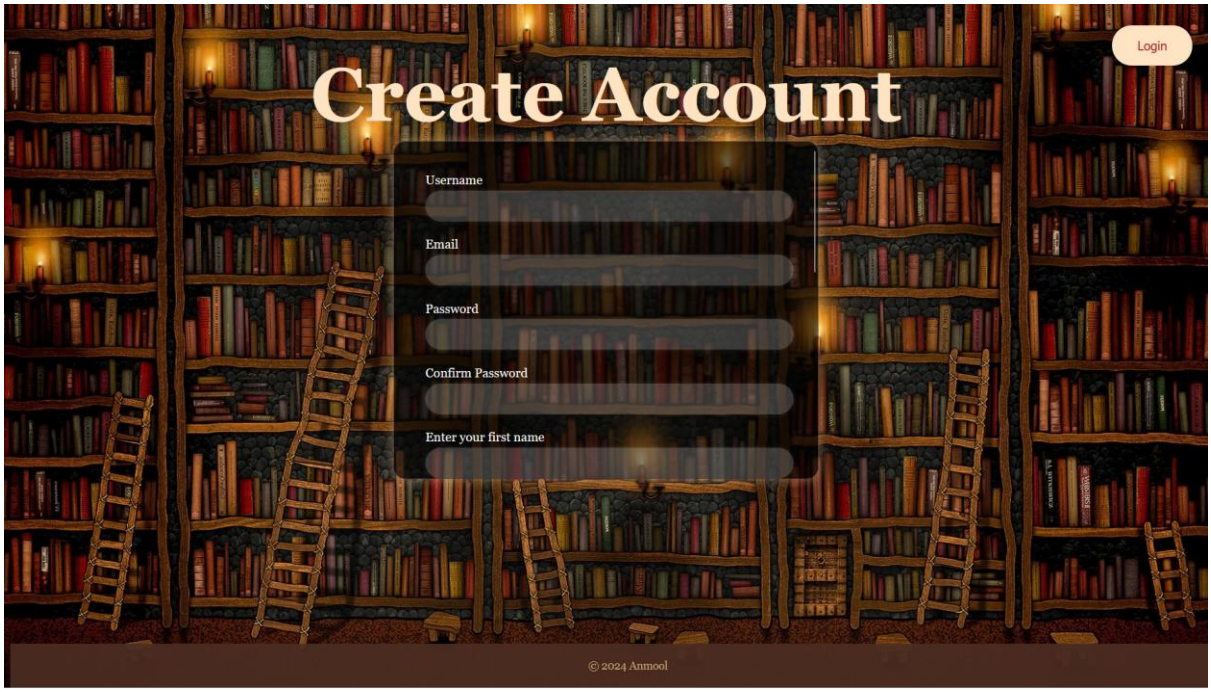
## Page Creation

After creating the database, I began to create the pages for the website.

## Create Account

The first page I created was the account creation page. I made sure to include:

- Fields for the user to enter their details.
- A way to make sure the username is unique.
- A way to make sure the user can confirm their password.
- A login button so the user isn't stuck in the account creation page.




**Below is the PHP and HTML for the account creation page**

**PHP:**

```
<?php
session_start();
require_once 'calibrary.php';

$error_message = '';
$success_message = '';

if (isset($_POST['create'])) {
    $username = $_POST['user'];
    $email = $_POST['email'];
    $password = $_POST['pass'];
    $confirm_password = $_POST['pass2'];
    $first_name = $_POST['fname'];
    $last_name = $_POST['lname'];
    $address1 = $_POST['Add1'];
    $address2 = $_POST['Add2'];
    $city = $_POST['city'];
    $telephone = $_POST['tp'];
    $mobile = $_POST['mb'];
    // Check if any required field is empty
    if (empty($username) || empty($email) || empty($password) || empty($confirm_password)
|| empty($first_name) || empty($last_name) || empty($address1) || empty($city) ||
empty($telephone) || empty($mobile)) {
        $error_message = 'Please fill in all the fields.';
    }
    // Check if the passwords match
    elseif ($password !== $confirm_password) {
        $error_message = 'Passwords do not match.';
    } else {
        // Check if the username already exists
        $sql_check_username = "SELECT * FROM users WHERE Username = '$username'";
        $result = $conn->query($sql_check_username);

        if ($result->num_rows > 0) {
            $error_message = 'Username is already taken, please choose another one.';
        } else {
            // Insert the new user into the database
            $sql = "INSERT INTO users (Username, Email, Pass, Firstname, Surname,
Addressline1, Addressline2, City, Telephone, Mobile)
                    VALUES ('$username','$email', '$password', '$first_name', '$last_name',
'$address1', '$address2', '$city', '$telephone', '$mobile')";

            if ($conn->query($sql) === TRUE) {
                $success_message = 'Account created successfully!';
```

```
            } else {
                $error_message = 'Error: ' . $conn->error;
            }
        }
        $conn->close();
    }
}
?>
```

This PHP script validates the data entered by the user and ensure that all the fields are filled out and the passwords match. It also makes sure that the username entered by the user is unique so that two users cannot have the same username. If all the fields are entered correctly, the user's details are securely inserted into the users table in the MySQL Database. After successfully creating the account, a success message is shown to the user.

## HTML:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Create Account</title>
    <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<form action="login.php" method="POST" class="logout-form">
            <button type="submit" name="login" class="logout-btn">Login</button>
        </form>
    <div id="title">
        <h1>Create Account</h1>
    </div>

    <div id="form">

        <?php if (!empty($error_message)) { ?>
            <div class="error-message" style="color: red; text-align: center;"><?=
$error_message; ?></div>
        <?php } ?>
        <?php if (!empty($success_message)) { ?>
            <div class="success-message" style="color: green; text-align: center;"><?=
$success_message; ?></div>
        <?php } ?>

        <form name="createAccountForm" method="POST">
            <div class="input-containerca">
                <label for="user">Username</label>
                <input type="text" id="user" name="user" required>
            </div>

            <div class="input-containerca">
```

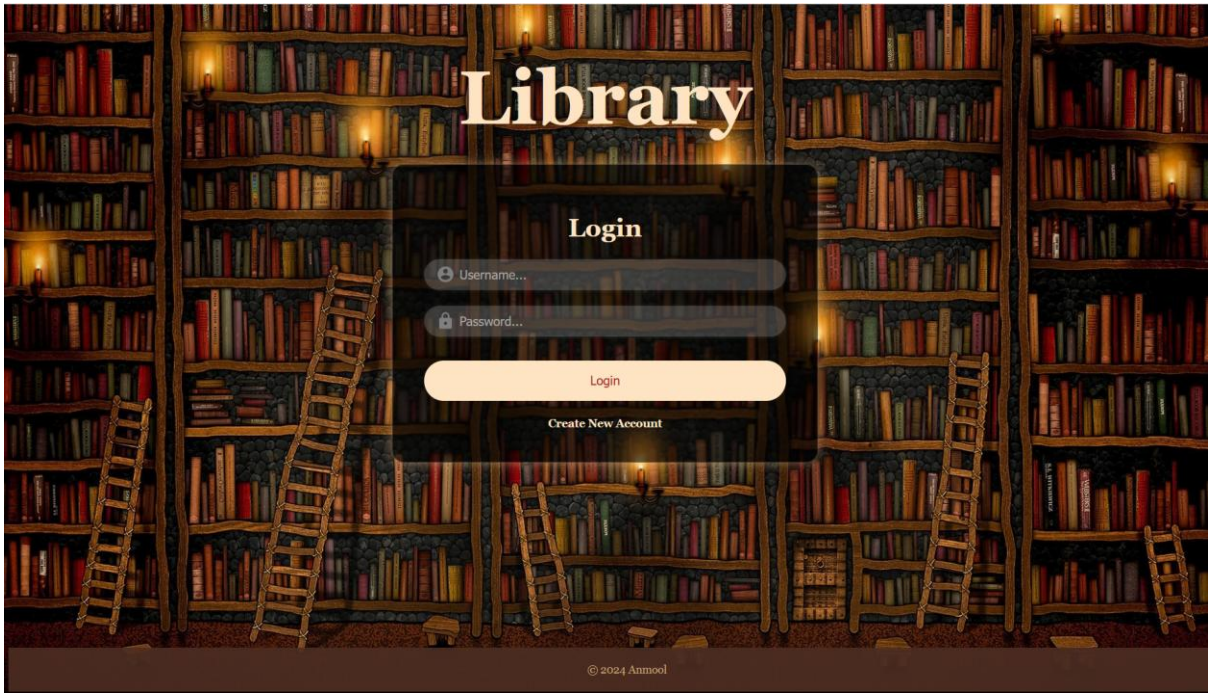
```
                <label for="user">Email</label>
                <input type="text" id="user" name="email" required>
     </div>

            <div class="input-containerca">
                <label for="pass">Password</label>
                <input type="password" id="pass" name="pass" required>
            </div>

            <div class="input-containerca">
                <label for="pass2">Confirm Password</label>
                <input type="password" id="pass2" name="pass2" required>
            </div>

            <div class="input-containerca">
                <label for="fname">Enter your first name</label>
                <input type="text" id="fname" name="fname" required>
            </div>

            <div class="input-containerca">
                <label for="lname">Enter your last name</label>
                <input type="text" id="lname" name="lname" required>
            </div>

            <div class="input-containerca">
                <label for="Add1">Address Line 1</label>
                <input type="text" id="Add1" name="Add1" required>
            </div>

            <div class="input-containerca">
                <label for="Add2">Address Line 2</label>
                <input type="text" id="Add2" name="Add2">
            </div>

            <div class="input-containerca">
                <label for="city">City</label>
                <input type="text" id="city" name="city" required>
            </div>

            <div class="input-containerca">
                <label for="tp">Telephone</label>
                <input type="text" id="tp" name="tp" required>
            </div>

            <div class="input-containerca">
                <label for="mb">Mobile</label>
                <input type="text" id="mb" name="mb" required>
            </div>

            <input type="submit" id="create" value="Create" name="create">
        </form>
    </div>
    <?php include('footer.php'); ?>
</body>
</html>
```

## Login

After creating the account creation page, I created the login page.

I made sure to include:

- Fields to enter username and password
- Login button
- Create Account button
- A way to make sure the user credentials match with the information stored in the database.




**Below is the HTML and PHP for the Log In page.**

## HTML:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>LogIn</title>
    <link rel="stylesheet" type="text/css" href="style.css">
    <link rel="stylesheet"
href="https://fonts.googleapis.com/icon?family=Material+Icons">
</head>
<body>
    <div id="title">
        <h1>Library</h1>
    </div>
    <div id="form">
        <h1>Login</h1>
```

```
        <?php if (isset($error_message)) {
            ?>
            <div class="error" style="color: red; text-align: center;">
                <?= $error_message; ?>
            </div><?php } ?>

        <form action="loginscript.php" method="POST">

            <div class="input-container">
                <span class="material-icons">account_circle</span>
                <input type="text" id="user" name="user"
placeholder="Username..." required>
            </div>

            <div class="input-container">
                <span class="material-icons">lock</span>
                <input type="password" id="pass" name="pass"
placeholder="Password..." required>
            </div>

            <input type="submit" id="sbt" value="Login" name="submit">

            <div class="new-account">
                <a href="createAccount.php" id="create-account">Create
New Account</a>
            </div>

        </form>

    </div>
    <?php include('footer.php'); ?>
</body>
</html>
```

This HTML is connected to the login script below

## PHP:

```
<?php
session_start();
require_once 'calibrary.php';
unset($_SESSION["user"]);

if (isset($_POST["user"]) && isset($_POST["pass"])) {
    $username = $_POST["user"];
    $password = $_POST["pass"];

    $stmt = $conn->prepare("SELECT * FROM users WHERE Username = ? AND
Pass = ?");
    $stmt->bind_param("ss", $username, $password);
    $stmt->execute();
    $result = $stmt->get_result();

    if ($result->num_rows > 0) {
        $_SESSION["user"] = $username;
        $_SESSION["success"] = "Logged in.";
        header('Location: index.php');
        return;
```

```
    } else {
        $_SESSION["error"] = "Incorrect username or password.";
        header('Location: login.php');
        return;
    }
} else if (count($_POST) > 0) {
    $_SESSION["error"] = "Missing Required Information.";
    header('Location: login.php');
    return;
}
?>
```

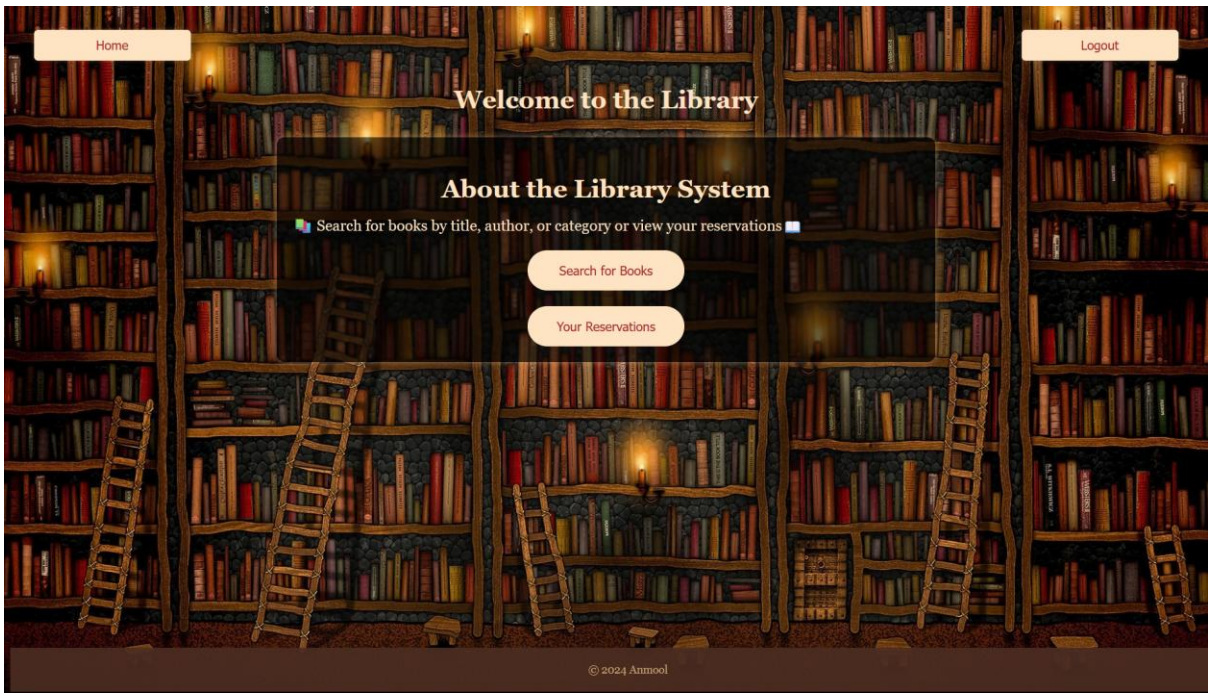
The PHP script handles the login process. It starts by checking if the user is already logged in by unsetting the session. When the user submits the login form it the script prepares a SQL query to check if the credentials match any data in the database. If matching credentials are found, the user is logged in. The user is then redirected to the home page. If the login details are incorrect, the user is redirected to the login page. If there's any missing information from the fields, an error message is displayed.

## Home

After creating the login page, I created a simple home page.

I made sure to include:

- A button that directs the user to the search page.
- A button that directs the user to their reservations.
- A header that includes the home and logout button.



Below is the HTML for this page.

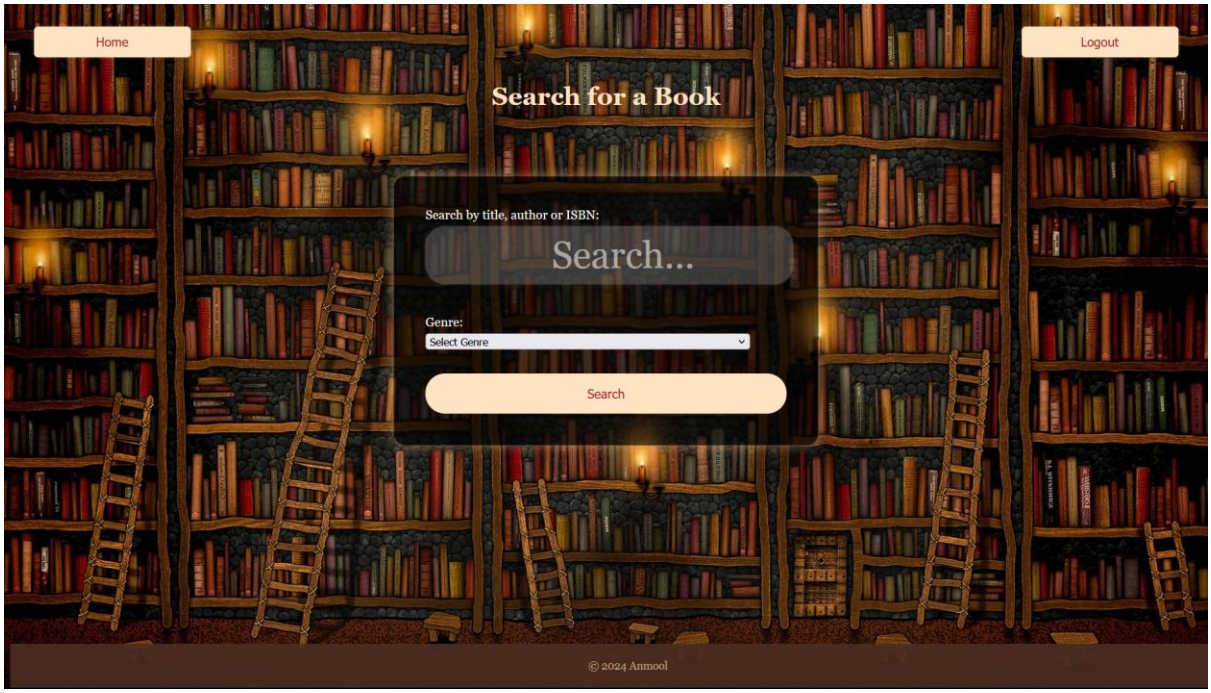
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Library Home</title>
    <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
    <header>
    <?php include('header.php'); ?>

        <h1>Welcome to the Library</h1>
    </header>

    <div id="homepage">
        <h2>About the Library System</h2>

        <ul style="list-style-type: none; padding: 0;">
            <li>📚 Search for books by title, author, or category or view
your reservations📖</li>
        </ul>

        <div class="button-container">
            <a href="search.php">
                <button>Search for Books</button>
            </a>
            <a href="reservations.php">
                <button>Your Reservations</button>
            </a>
        </div>
    </div>
    <?php include('footer.php'); ?>
</body>
</html>
```

## Search

I then created the search page.

This page includes:

- Search bar to search for books by ISBN, Author or Book Title.
- Drop down menu to search for books by category(genre).
- Header for home and logout button.




Below is the PHP and Html for this page.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Search Books</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
<?php include('header.php'); ?>

    <header>
        <h1>Search for a Book</h1>
    </header>

    <div id="form">
        <form action="books.php" method="GET">
            <div class="input-containerca">
                <label for="title">Search by title, author or
ISBN:</label>
                <input type="text" id="title" name="title"
placeholder="Search..." value="<?php echo isset($_GET['title']) ?
htmlspecialchars($_GET['title']) : ''; ?>">
            </div>

            <div class="input-containerca">
                <label for="category">Genre:</label>
                <select id="category" name="category">
                    <option value="">Select Genre</option>
                    <?php
                        session_start();
                        require_once 'calibrary.php';
```

```
                        $sql = "SELECT CategoryID, CategoryDescription
FROM category";
                        $select_result = $conn->query($sql);
                        if ($select_result->num_rows > 0) {
                            while ($row = $select_result-
>fetch_assoc()) {
                                $selected = (isset($_GET['category'])
&& $_GET['category'] == $row['CategoryDescription']) ? "selected" :
"";
                                echo "<option value=\"" .
htmlspecialchars($row['CategoryDescription']) . "\" $selected>" .
htmlspecialchars($row['CategoryDescription']) . "</option>";
                            }
                        }
                    ?>
                </select>
            </div>

            <button type="submit" id="sbt">Search</button>
        </form>
    </div>

    <?php include('footer.php'); ?>
</body>
</html>
```

This script allows the user to search for books based on the title, author or ISBN as well as filter books by genre.

The page includes a search bar where the user can search a term and it will filter through the database with the search term.

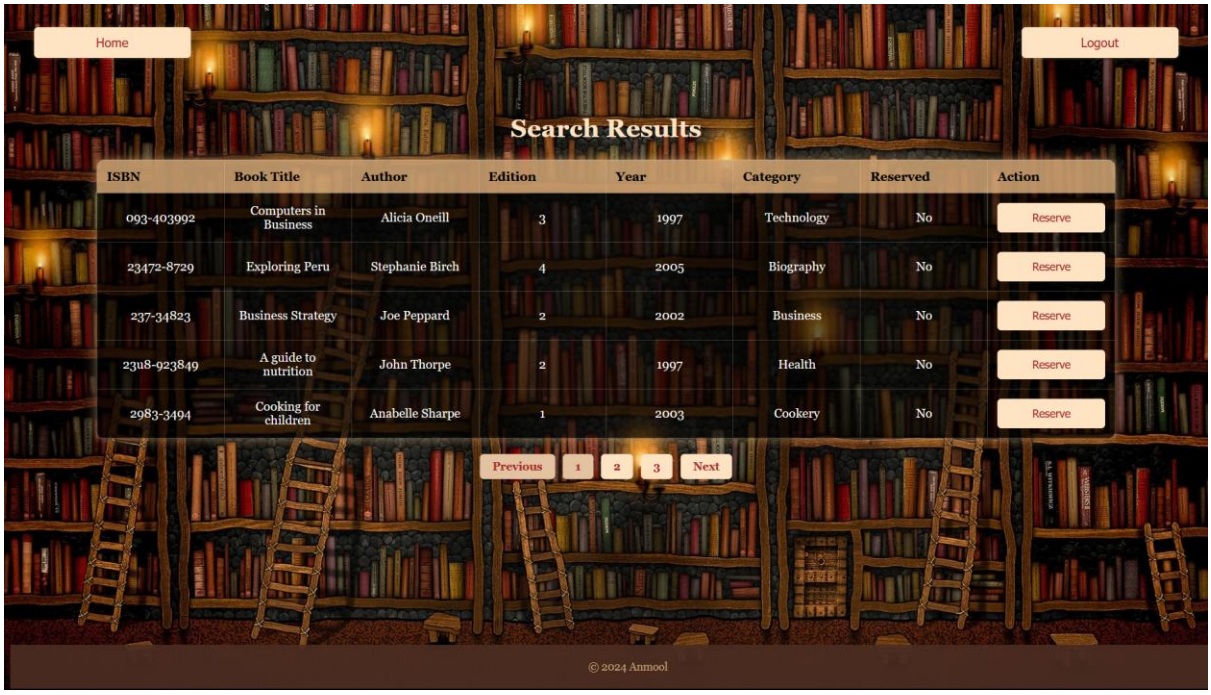
The form then sends the data to books.php using a GET method which will then display the results on a new page.

After searching, the user is lead to the books.php page which displays the search results so I created that page next.

## Search Results

I made sure this page includes:

- A table with all the books based off the search.
- Options to reserve books.
- Pagination to allow only 5 results per page.
- Header with home and logout button.




Below is the HTML and PHP for this page

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Books</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
<?php include('header.php'); ?>
<?php
require_once 'calibrary.php';

$user_id = isset($_SESSION['user_id']) ? $_SESSION['user_id'] :
$title = isset($_GET['title']) ? $_GET['title'] : '';
$category = isset($_GET['category']) ? $_GET['category'] : '';

$page = isset($_GET['page']) ? (int)$_GET['page'] : 1;
$records_per_page = 5;
$offset = ($page - 1) * $records_per_page;

$sql = "SELECT b.ISBN, b.BookTitle, b.Author, b.Edition, b.Year,
c.CategoryDescription, b.Reserved, b.ReservedBy FROM books b INNER JOIN
category c ON b.Category = c.CategoryID WHERE 1";

if (!empty($title)) {
    $sql .= " AND (b.BookTitle LIKE ? OR b.Author LIKE ? OR b.ISBN LIKE
?)";
}

if (!empty($category)) {
    $sql .= " AND c.CategoryDescription = ?";
```

```
}

$sql .= " LIMIT $records_per_page OFFSET $offset";

$stmt = $conn->prepare($sql);
if ($stmt === false) {
    die('MySQL prepare error: ' . $conn->error);
}

if (!empty($title) && !empty($category)) {
    $title_like = "%$title%";
    $stmt->bind_param('ssss', $title_like, $title_like, $title_like,
$category);
} elseif (!empty($title)) {
    $title_like = "%$title%";
    $stmt->bind_param('sss', $title_like, $title_like, $title_like);
} elseif (!empty($category)) {
    $stmt->bind_param('s', $category);
} else {
    $stmt->execute();
}

$stmt->execute();
$result = $stmt->get_result();

// Display results
echo '<div id="search-results">';
if ($result->num_rows > 0) {
    echo "<h1>Search Results</h1>";
    echo "<table id='booktable'>";
    echo "<thead>
            <tr>
                <th>ISBN</th>
                <th>Book Title</th>
                <th>Author</th>
                <th>Edition</th>
                <th>Year</th>
                <th>Category</th>
                <th>Reserved</th>
                <th>Action</th>
            </tr>
          </thead>";
    echo "<tbody>";

    while ($row = $result->fetch_assoc()) {
        echo "<tr>";
        echo "<td>" . htmlspecialchars($row['ISBN']) . "</td>";
        echo "<td>" . htmlspecialchars($row['BookTitle']) . "</td>";
        echo "<td>" . htmlspecialchars($row['Author']) . "</td>";
        echo "<td>" . htmlspecialchars($row['Edition']) . "</td>";
        echo "<td>" . htmlspecialchars($row['Year']) . "</td>";
        echo "<td>" . htmlspecialchars($row['CategoryDescription']) .
"</td>";
        echo "<td>" . ($row['Reserved'] ? 'Yes' : 'No') . "</td>";

        if ($row['Reserved'] == 0) {
            echo "<td>
                    <form method='POST' action='reserve.php'>
                        <input type='hidden' name='isbn' value='" .
htmlspecialchars($row['ISBN']) . "'>
```

```
                        <input type='submit' name='reserve'
class='reserve-button' value='Reserve'>
                    </form>
                  </td>";
        } else {
            echo "<td></td>";
        }
        echo "</tr>";
    }
    echo "</tbody>";
    echo "</table>";

    $total_sql = "SELECT COUNT(*) AS total FROM books b INNER JOIN
category c ON b.Category = c.CategoryID WHERE 1";
    if (!empty($title)) {
        $total_sql .= " AND (b.BookTitle LIKE '%$title%' OR b.Author LIKE
'%$title%' OR b.ISBN LIKE '%$title%')";
    }
    if (!empty($category)) {
        $total_sql .= " AND c.CategoryDescription = '$category'";
    }

    $total_result = $conn->query($total_sql);
    $total_row = $total_result->fetch_assoc();
    $total_records = $total_row['total'];
    $total_pages = ceil($total_records / $records_per_page);

    echo "<div class='pagination'>";
    $previous_disabled = ($page == 1) ? "disabled" : "";
    echo "<a href='?page=" . ($page - 1) . "' class='pagination-button
$previous_disabled'>Previous</a>";

    for ($i = 1; $i <= $total_pages; $i++) {
        $active = ($i === $page) ? "active" : "";
        echo "<a href='?page=$i' class='pagination-button
$active'>$i</a>";
    }

    $next_disabled = ($page == $total_pages) ? "disabled" : "";
    echo "<a href='?page=" . ($page + 1) . "' class='pagination-button
$next_disabled'>Next</a>";
    echo "</div>";
}
else
{
    echo "<p>No books found matching your criteria.</p>";
}
echo '</div>';
$conn->close();
?>

<?php include('footer.php'); ?>
</body>
</html>
```

This page fetches the information of the user logged in. I implemented a pagination method to limit the search results to only 5 per page. The current page is fetched from the GET

request while the first page is set to the default. Based on the page, the offset is calculated so only 5 records will be retrieved from the database.

The SQL query is to filter books based by search by using a LIKE clause while also using the INNER JOIN between books and category tables to make sure the genre matches with the book. I learned to use the bind parameter to ensure that the data securely inputs into the SQL query.

If both the category and search is filled, the query will bind data that has the appropriate values for both. Otherwise, if only one is filled it will search based by only that input. The data is then fetched using get_result.

To display the books, if the matching books are found the script loops through the results displaying the details and whether the book is reserved or not. If a book is unreserved then Reserved === 0. The reserve button will be visible and the user will be able to reserve using reserve.php. Otherwise, Reserved === 1 and no reserve button is shown.

For the pagination, the number of total pages are shown and the previous button is disabled on the first page while next is disabled on the last page.

If no books are available based on the search a message “No books found matching your criteria” is displayed. Finally in the end the database connection is closed.

Below is the code for reserve.php which allows the user to reserve books from the table.

```
<?php
session_start();
require_once 'calibrary.php';

if (!isset($_SESSION["user"])) {
    echo "You need to be logged in to reserve a book.";
    exit;
} else {
    $username = $_SESSION["user"];
}

if (isset($_POST['isbn'])) {
    $isbn = $_POST['isbn'];

    $check_query = "SELECT Reserved FROM books WHERE ISBN = ?";
    $stmt_check = $conn->prepare($check_query);
    $stmt_check->bind_param("s", $isbn);
    $stmt_check->execute();
    $stmt_check->store_result();
    $stmt_check->bind_result($reserved);
    $stmt_check->fetch();

    if ($reserved == 0) {
        $update_query = "UPDATE books SET Reserved = 1, ReservedBy = ?
WHERE ISBN = ? AND Reserved = 0";
        $stmt_update = $conn->prepare($update_query);
        $stmt_update->bind_param("ss", $username, $isbn);
        if ($stmt_update->execute()) {
            header("Location: books.php?reservation=success");
            exit;
        } else {
```

```
            echo "Error reserving the book.";
        }
        $stmt_update->close();
    } else {
        echo "This book is already reserved.";
    }
    $stmt_check->close();
} else {
    echo "Invalid request. ISBN not provided.";
}

$conn->close();
?>
```

This PHP allows the user to reserve a book only if they're logged in. If the user is logged in, it retrieves the book's ISBN from the POST request. It first checks if the book is reserved by checking if Reserved === 0, if so it then reserves the book for the user.

When the user reserves the book, the script updates Reserved === 1 and inputs the user's username into ReservedBy field in the database. This is easily done by using an UPDATE SQL query which also prevents SQL injection.

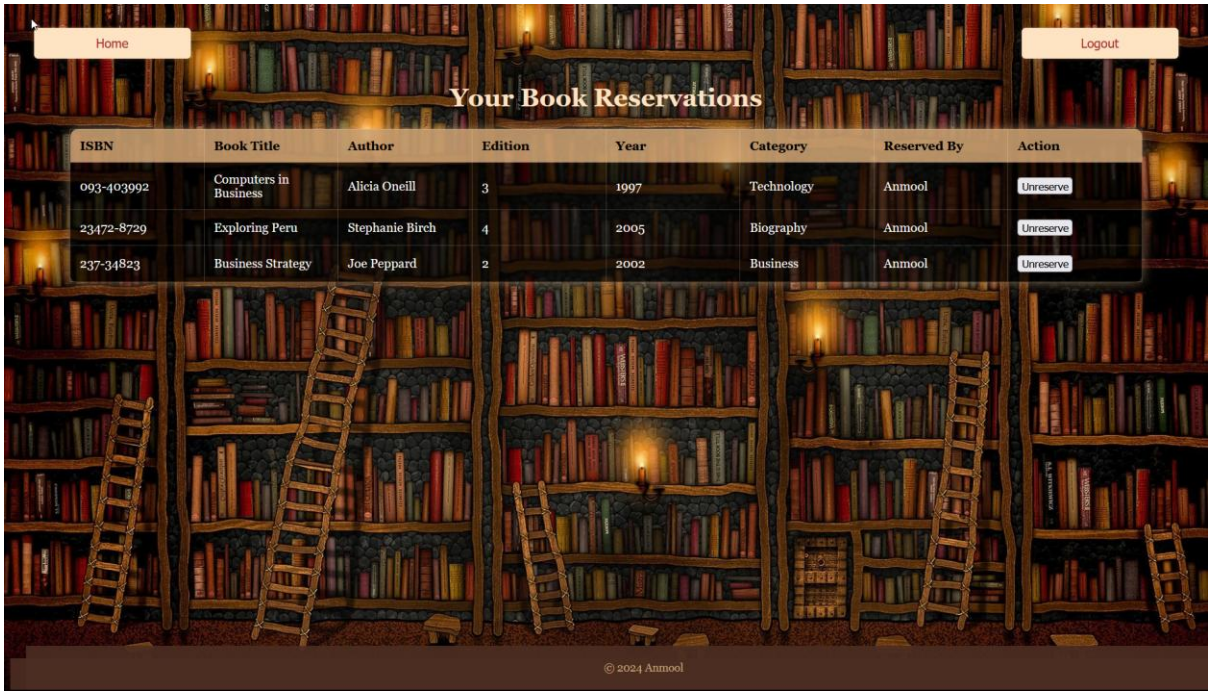
When successful, the user is redirected to the books.php page.

## Reservations

After the search results, I created a page where the user can view their reservations and unreserve books.

I made sure to include:

- A table with all the book's details.
- A button to unreserve books.
- Header with home and logout button.




Below is the PHP and HTML for this page.

## PHP

```
<?php include('header.php'); ?>
<?php

require_once 'calibrary.php';

if (!isset($_SESSION["user"])) {
    echo "You need to be logged in to reserve a book.";
    exit;
} else {
    $username = $_SESSION["user"];
}

//unreserve
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['unreserve']))
{
    $isbn = $_POST['isbn'];
    $unreserve_sql = "UPDATE books SET Reserved = 0, ReservedBy = NULL
WHERE ISBN = ? AND ReservedBy = ?";
    $stmt = $conn->prepare($unreserve_sql);
    if ($stmt === false) {
        die('MySQL prepare error: ' . $conn->error);
    }
    $stmt->bind_param('ss', $isbn, $username);
    $stmt->execute();
    $stmt->close();
}

//reserved books
$sql = "SELECT b.ISBN, b.BookTitle, b.Author, b.Edition, b.Year,
c.CategoryDescription, b.ReservedBy FROM books b INNER JOIN category c ON
b.Category = c.CategoryID WHERE b.ReservedBy = ?";
```

```
$stmt = $conn->prepare($sql);
if ($stmt === false) {
    die('MySQL prepare error: ' . $conn->error);
}
$stmt->bind_param('s', $username);
$stmt->execute();
$result = $stmt->get_result();
?>
```

This checks if the user is logged in. If not it won't display the list of reservations. If the user is logged in it uses the username to check what books have been reserved by the user. If the unreserve button is clicked, the script will tell by the POST method and retrieves the ISBN of the book unreserve the book. In the SQL, it will update to Reserved === 0 and the ReservedBy field will be set to NULL. This will only work if ReservedBy value is the same as the user's username.

An SQL query is then used to fetch the books the user has reserved. The query joins books with category to ensure that it also contains the genre of the book. It also used the bind parameter to make sure only the books reserved by the user are selected.

## HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Your Reservations</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <header>
        <h1>Your Book Reservations</h1>
    </header>

    <div id="reserved-books">
        <?php if ($result->num_rows > 0): ?>
            <table id="booktable">
                <thead>
                    <tr>
                        <th>ISBN</th>
                        <th>Book Title</th>
                        <th>Author</th>
                        <th>Edition</th>
                        <th>Year</th>
                        <th>Category</th>
                        <th>Reserved By</th>
                        <th>Action</th>
                    </tr>
                </thead>
                <tbody>
```

```
                    <?php while ($row = $result->fetch_assoc()): ?>
                        <tr>
                            <td><?= htmlspecialchars($row['ISBN'])
?></td>
                            <td><?= htmlspecialchars($row['BookTitle'])
?></td>
                            <td><?= htmlspecialchars($row['Author'])
?></td>
                            <td><?= htmlspecialchars($row['Edition'])
?></td>
                            <td><?= htmlspecialchars($row['Year'])
?></td>
                            <td><?=
htmlspecialchars($row['CategoryDescription']) ?></td>
                            <td><?= htmlspecialchars($row['ReservedBy'])
?></td>
                            <td>
                                <form method="POST" action="">
                                    <input type="hidden" name="isbn"
value="<?= htmlspecialchars($row['ISBN']) ?>">
                                    <input type="submit" name="unreserve"
class="unreserve-button" value="Unreserve">
                                </form>
                            </td>
                        </tr>
                    <?php endwhile; ?>
                </tbody>
            </table>
        <?php else: ?>
            <p class = "no-books">You have no reserved books at this
time.</p>
        <?php endif; ?>
    </div>

    <footer>
        <?php include('footer.php'); ?>
    </footer>
</body>
</html>
```

## Below are the additional PHP pages that aren't visible to the user.

### Header PHP

```
<?php
session_start();
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
```

```
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Library System</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
<header>
    <div id="Header">
    <a href="index.php" class="back-to-home">
            <button class="back-button">Home</button>
        </a>
        <?php if (isset($_SESSION['user'])): ?>
            <form method="POST" action="logout.php">
                <button type="submit" class="logout-
button">Logout</button>
            </form>
        <?php else: ?>
            <p><a href="login.php" style="color: #FFCB90; text-
decoration: none;">Login</a> to access your account.</p>
        <?php endif; ?>
    </div>
</header>
```

This is used to ensure there is a home and logout button on almost every page to make sure the user doesn't get stuck on any page.

## Footer PHP

```
<footer>
    <p>&copy; 2024  Anmool</p>
</footer>

<style>
    footer {
        text-align: center;
        padding: 20px;
        background-color: rgba(77, 46, 34, 0.9);
        position: fixed;
        bottom: 0;
        width: 100%;
    }

    footer p {
        margin: 0;
        font-size: 14px;
        color: #C5A678;
    }
</style>
```

Simple footer that includes author's name.

## CSS

```
body {
    background-image: url("library.jpg");
    background-size: cover;
    background-repeat: no-repeat;
    font-family: Georgia, 'Times New Roman', Times, serif;
}

#title, #main-header {
    text-align: center;
    margin-bottom: 20px;
    font-size: 48px;
    font-family: Georgia, 'Times New Roman', Times, serif;
}

h1 {
    text-align: center;
    color: bisque;
}

#form {
    background-color: rgba(0, 0, 0, 0.6);
    width: 30%;
    padding: 40px;
    border-radius: 15px;
    box-shadow: 0 0 15px rgba(255, 203, 144, 0.5);
    border: 1px solid rgba(255, 255, 255, 0.2);
    backdrop-filter: blur(1px);
    color: white;
    margin: auto;
    position: absolute;
    top: 45%;
    left: 50%;
    transform: translate(-50%, -50%);
    max-height: 40vh;
    overflow-y: auto;
}

.input-container {
    position: relative;
    display: flex;
    align-items: center;
    margin-bottom: 20px;
}

.input-containerca {
    position: relative;
    display: flex;
    flex-direction: column;
    margin-bottom: 20px;
    width: 90%;

}

.input-container span.material-icons, .input-containerca span.material-
icons {
    position: absolute;
    left: 15px;
```

```
    color: gray;
    pointer-events: none;
    top: 50%;
    transform: translateY(-50%);
}

.input-container input, .input-containerca input {
    width: 100%;
    padding: 10px 10px 10px 45px;
    border: none;
    border-radius: 25px;
    font-size: 16px;
    background-color: rgba(255, 255, 255, 0.2);
    color: white;
    outline: none;
    box-shadow: 0 0 5px rgba(255, 255, 255, 0.1);
    transition: all 0.3s ease-in-out;
}

.input-container input:focus, .input-containerca input:focus {
    background-color: rgba(255, 255, 255, 0.3);
    box-shadow: 0 0 10px rgba(255, 255, 255, 0.5);
}

.input-containerca label {
    display: block;
    margin-bottom: 5px;
    color: white;
    font-size: 16px;
}

.input-containerca span.material-icons {
    position: absolute;
    left: 15px;
    top: 50%;
    transform: translateY(-50%);
    color: gray;
}

#sbt, #create {
    background-color: bisque;
    border: none;
    border-radius: 25px;
    color: brown;
    padding: 16px 32px;
    text-decoration: none;
    cursor: pointer;
    width: 100%;
    font-size: 16px;
    margin-top: 10px;
    transition: background-color 0.3s;
}

#sbt:hover , #create:hover{
    background-color: #e3c3a1;
}

.new-account {
    text-align: center;
    margin-top: 20px;
}
```

```
.new-account a {
    color: bisque;
    text-decoration: none;
    font-size: 14px;
    font-weight: bold;
    transition: color 0.3s;
}

.new-account a:hover {
    color: #e3c3a1;
}

#homepage {
    background-color: rgba(0, 0, 0, 0.6);
    padding: 20px;
    border-radius: 10px;
    margin-top: 30px;
    width: 80%;
    max-width: 800px;
    margin-left: auto;
    margin-right: auto;
    color: bisque;
    box-shadow: 0 0 10px rgba(255, 203, 144, 0.5);
    backdrop-filter: blur(1px);
}

#homepage h2 {
    font-size: 2rem;
    text-align: center;
    margin-bottom: 15px;
    color: bisque;
}

#homepage ul {
    list-style-type: none;
    padding: 0;
    margin: 0;
    font-size: 1.2rem;
    color: bisque;
}

#homepage li {
    margin: 10px 0;
    display: flex;
    align-items: center;
}

#homepage p {
    font-size: 1.2rem;
    color: bisque;
    text-align: center;
    margin-top: 20px;
}

button {
    background-color: bisque;
    border: none;
    border-radius: 25px;
    color: brown;
    padding: 16px 32px;
```

```
    text-decoration: none;
    cursor: pointer;
    font-size: 16px;
    margin-top: 10px;
    transition: background-color 0.3s;
    width: 200px;
    display: inline-block;
    text-align: center;
}

.button-container {
    display: flex;
    gap: 10px;
    flex-direction: column;
    align-items: center;
}

button:hover {
    background-color: #e3c3a1;
}

header {
    text-align: center;
    margin-bottom: 20px;
}

#homepage {
    padding: 20px;
}

h2 {
    font-size: 24px;
    margin-bottom: 10px;
}

ul {
    padding-left: 20px;
    font-size: 18px;
}

.logout-btn {
    background-color: bisque;
    border: none;
    border-radius: 25px;
    color: brown;
    padding: 16px 32px;
    width: 100%;
    text-decoration: none;
    cursor: pointer;
    font-size: 16px;
    transition: background-color 0.3s;
}

.logout-btn:hover {
    background-color: #e3c3a1;
}

.logout-form {
    position: absolute;
    top: 20px;
    right: 20px;
```

```
}
#booktable {
    width: 90%;
    margin: 20px auto;
    border-collapse: collapse;
    background-color: rgba(0, 0, 0, 0.6);
    color: white;
    font-size: 16px;
    border: 1px solid rgba(255, 255, 255, 0.2);
    border-radius: 10px;
    box-shadow: 0 0 15px rgba(255, 203, 144, 0.5);
    backdrop-filter: blur(1px);
    overflow: hidden;
    table-layout: fixed;
}

#booktable th {
    background-color: rgba(255, 203, 144, 0.7);
    color: black;
    padding: 12px;
    border: 1px solid rgba(255, 255, 255, 0.2);
    text-align: left;
    font-weight: bold;
    width: 12%;
}

#booktable td {
    padding: 12px;
    border: 1px solid rgba(255, 255, 255, 0.2);
    word-wrap: break-word;
    width: 12%;
}

#booktable tr:hover {
    background-color: rgba(255, 255, 255, 0.1);
}

#search-results {
    padding: 20px;
    max-width: 95%;
    margin: auto;
    text-align: center;
}

.reserve-button {
    background-color: bisque;
    color: brown;
    padding: 10px 20px;
    border: none;
    border-radius: 5px;
    font-size: 14px;
    cursor: pointer;
    transition: background-color 0.3s;
    display: block;
    width: 100%;
    box-sizing: border-box;
}

.reserve-button:hover {
    background-color: #e3c3a1;
}
```

```
.pagination {
    display: flex;
    justify-content: center;
    align-items: center;
    margin-top: 20px;
    gap: 10px;
    position: relative;
}

.pagination a {
    text-decoration: none;
    background-color: bisque;
    color: brown;
    padding: 8px 16px;
    border-radius: 5px;
    transition: background-color 0.3s ease;
    font-size: 14px;
    font-weight: bold;
}

.pagination a.active {
    background-color: #e3c3a1;
    color: brown;
    cursor: default;
    pointer-events: none;
}
.pagination a.disabled {
    background-color: #e3c3a1;
    cursor: not-allowed;
    color: brown;
    pointer-events: none;
}

.pagination a:hover {
    background-color: #e3c3a1;
}

.pagination {
    margin-top: 20px;
    padding-bottom: 10px;
}
#Header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 10px 20px;
    border-radius: 15px;
    color: white;
    position: relative;
    margin: 10px;
    z-index: 10;
}
.back-to-home .back-button {
    padding: 10px 20px;
    background-color: bisque;
    color: brown;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    text-decoration: none;
```

```
    box-shadow: 0 0 5px rgba(255, 203, 144, 0.3);
    transition: background-color 0.3s ease, box-shadow 0.3s ease;
    font-size: 16px;
}

.back-to-home .back-button:hover {
    background-color: #e3c3a1;
    box-shadow: 0 0 10px rgba(255, 203, 144, 0.5);
}

a {
    text-decoration: none;
}

.logout-button {
    padding: 10px 20px;
    background-color: bisque;
    color: brown;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    box-shadow: 0 0 5px rgba(255, 203, 144, 0.3);
    transition: background-color 0.3s ease, box-shadow 0.3s ease;
    font-size: 16px;
}

.logout-button:hover {
    background-color: #e3c3a1;
    box-shadow: 0 0 10px rgba(255, 203, 144, 0.5);
}

.user-info p {
    margin: 0;
}

#Header .back-to-home,
#Header .user-info {
    display: flex;
    align-items: center;
}

.no-books {
    color: bisque;
}
```

The CSS makes the website look visually appealing and contain a consistent style.

## Conclusion

In this project, I developed a website for a library reservation system that allows the user to search, reserve and unreserve from the catalog. The system contains a database with functions including user authentication, book search, and manage reservations.

Throughout the creation process, I used PHP and MySQL to interact with the database. The UI was designed to be intuitive and easy to follow.

During the development process, I learned many new skills such as database management especially in querying the database. I learned how to write SQL queries to retrieve data and implement features such as user authentication and book reservations. I also learned how to implement a pagination system, learning how to display data across multiple pages instead of all on one page. I learned how to prepare binding parameters to ensure better security within the website but also allow the system to work more efficiently. Overall, this project enhanced my understanding of backend development and taught me how to integrate the backend with the front end interface.