

# औद्योगिक प्रशिक्षण के लिए राष्ट्रीय संस्थान

## National Institute for Industrial Training

One Premier Organization with Non Profit Status | Registered Under Govt. of WB

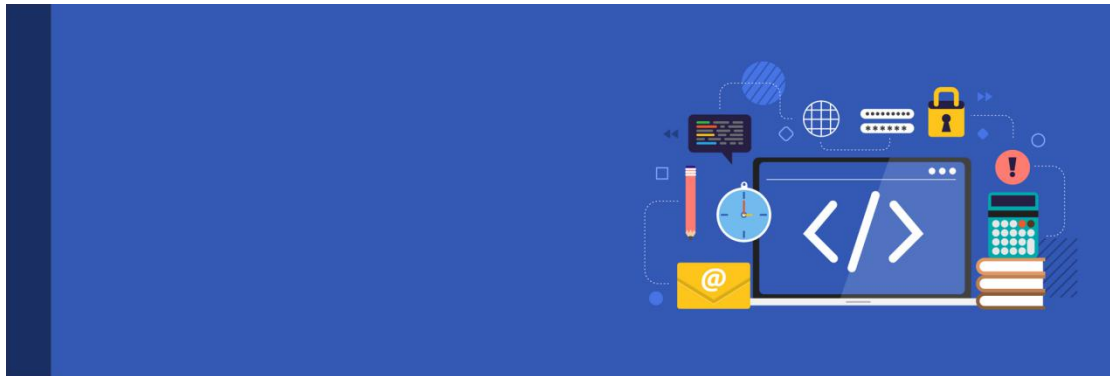
Empanelled Under Planning Commission Govt. of India

Inspired By: National Task Force on IT & SD Government of India

National Institute for Industrial Training- One Premier Organization with Non Profit Status Registered Under Govt. of West Bengal, Empanelled Under Planning Commission Govt. of India, Empanelled Under Central Social Welfare Board Govt. of India, Registered with National Career Services, Registered with National Employment Services.



## WEB DEVELOPMENT



**Subject:** Student Library Portal

**Submitted By:** Daisy Rabha

**Submitted to:** Pritam Mahapatra

# CONTENTS

1. ACKNOWLEDGEMENT
2. STUDENT PROFILE
3. INTRODUCTION
4. HARDWARE AND SOFTWARE REQUIREMENTS
5. OBJECTIVE AND PROCEDURE
6. CONCLUSION
7. REFERENCE

# ACKNOWLEDGEMENT

I would like to express my deepest appreciation to all those who provided me the possibility to complete this project. I give special gratitude to my project guider, Mr. Pritam Mahapatra, whose contribution in stimulating ideas and encouragement helped me coordinate my project. Furthermore, I would also like to acknowledge with much appreciation the crucial role of the National Institute for Industrial Training, which gave the permission to use all required equipment and the necessary materials to complete the course “Web Development”. I have put tremendous effort in this project. However, it would not have been possible without the kind support and help of above-mentioned individual and organization. I would like to extend my sincere thanks to all of them.

# STUDENT DETAILS

**Name:** Daisy Rabha

**University:** Kalinga Institute of Industrial Technology

**Branch:** B.Tech in Computer Science and Engineering

**Email:** [daisyrabha7576@gmail.com](mailto:daisyrabha7576@gmail.com)

**LinkedIn:** [linkedin.com/in/daisyrabha](https://www.linkedin.com/in/daisyrabha)

**Github:** [github.com/daisyrabha](https://github.com/daisyrabha)

**Project:** Student Library Portal

**Project Files:** <https://github.com/daisyrabha/Student-Library-Portal.git>

# INTRODUCTION



Web development is the work involved in developing a Web site for the Internet (World Wide Web) or an intranet (a private network). Web development can range from developing a simple single static page of plain text to complex web applications, electronic businesses, and social network services. A more comprehensive list of tasks to which Web development commonly refers, may include Web engineering, Web design, Web content development, client liaison, client-side/server-side scripting, Web server and network security configuration, and e-commerce development.

Among Web professionals, "Web development" usually refers to the main non-design aspects of building websites: writing markup and coding.

For larger organizations and businesses, Web development teams can consist of hundreds of people (Web developers) and follow standard methods like Agile methodologies while developing Web sites. Smaller organizations may only require a single permanent or contracting developer, or secondary assignment to related job positions such as a graphic designer or information systems technician. Web development may be a collaborative effort between departments rather than the domain of a designated department. There are three kinds of Web developer specialization: front-end developer, back-end developer, and full-stack developer. Front-end developers are responsible for behavior and visuals that run in the user browser, while back-end developers deal with the servers.

# HTML



**HTML5** is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and last major HTML version that is a World Wide Web Consortium (W3C) recommendation. The current specification is known as the HTML Living Standard.

"Hypertext" refers to links that connect web pages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web. By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web.

HTML uses "markup" to annotate text, images, and other content for display in a Web browser. HTML markup includes special "elements" such as `<head>`, `<title>`, `<body>`, `<header>`, `<footer>`, `<article>`, `<section>`, `<p>`, `<div>`, `<span>`, `<img>`, `<aside>`, `<audio>`, `<canvas>`, `<datalist>`, `<details>`, `<embed>`, `<nav>`, `<output>`, `<progress>`, `<video>`, `<ul>`, `<ol>`, `<li>` and many others.

# CASCADING STYLE SHEETS



**Cascading Style Sheets (CSS)** is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file which reduces complexity and repetition in the structural content as well as enabling the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

# JAVASCRIPT



**JavaScript (JS)** is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions. While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB and Adobe Acrobat. JavaScript is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles.

Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. Over 97% of websites use it client-side for web page behavior, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on the user's device.

As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).



# BOOTSTRAP



**Bootstrap** is an HTML, CSS & JS Library that focuses on simplifying the development of informative web pages (as opposed to web apps). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all HTML elements. The result is a uniform appearance for prose, tables and form elements across web browsers. In addition, developers can take advantage of CSS classes defined in Bootstrap to further customize the appearance of their contents. For example, Bootstrap has provisioned for light- and dark-colored tables, page headings, more prominent pull quotes, and text with a highlight.

Bootstrap also comes with several JavaScript components in the form of jQuery plugins. They provide additional user interface elements such as dialog boxes, tooltips, and carousels. Each Bootstrap component consists of an HTML structure, CSS declarations, and in some cases accompanying JavaScript code. They also extend the functionality of some existing interface elements, including for example an auto-complete function for input fields.

# PHP



**PHP** is a general-purpose scripting language geared towards web development. PHP originally stood for Personal Home Page, but it now stands for the recursive initialism PHP: Hypertext Preprocessor.

PHP code is usually processed on a web server by a PHP interpreter implemented as a module, a daemon or as a Common Gateway Interface (CGI) executable. On a web server, the result of the interpreted and executed PHP code – which may be any type of data, such as generated HTML or binary image data – would form the whole or part of an HTTP response. Various web template systems, web content management systems, and web frameworks exist which can be employed to orchestrate or facilitate the generation of that response. Additionally, PHP can be used for many programming tasks outside the web context, such as standalone graphical applications and robotic drone control. PHP code can also be directly executed from the command line.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on a variety of operating systems and platforms.

# MySQL



**MySQL** is a relational database management system based on **SQL** – Structured **Q**uery **L**anguage. The application is used for a wide range of purposes, including data warehousing, e-commerce, and logging applications.

The most common use for mySQL however, is for the purpose of a web database. It can be used to store anything from a single record of information to an entire inventory of available products for an online store.

In association with a scripting language such as **PHP** or **Perl** (both offered on our hosting accounts) it is possible to create websites which will interact in real-time with a mySQL database to rapidly display categorised and searchable information to a website user.

# HARDWARE & SOFTWARE REQUIREMENTS



## Software Requirements:

**Operating System:** Windows/Linux/Mac OS

**Programming Language:** HTML5, CSS, JavaScript and PHP

**Text Editor:** VSCode/Sublime Text/Atom/Notepad/etc.

**Web Browser:** Chrome/Safari/Mozilla Firefox/etc.

**Database:** MySQL

**Server:** XAMPP

## Hardware Requirements:

**Processor:** 1.9GHz x86- or x64-bit dual core processor with SSE2 instruction set/ 3.3GHz or faster 64-bit dual core processor with SSE2 instruction set (Recommended)

**Memory:** 2-GB RAM/4-GB RAM (Recommended)

**Display:** Super VGA with a resolution of 1024 x 768

# OBJECTIVE & PROCEDURE

My main objective of this project is to make a Student Library Portal where a student/user can sign-in to register themselves, and then issue any book they want ( just like a library management system). Alternatively, if the student is already registered, they can simply login to their account and issue their new book.

The importance of this project is that it can help save a lot of time, and is easy to use. If there were thousands of records and we need to find a particular one, we can simply use a query in the database to do so. Also, we can prevent duplicate entries and order them accordingly. Moreover, the issue process is very simple. All the user needs to do is to input the book details and its all done.

In this project, I have used HTML, CSS, JavaScript and PHP programming languages. Also, I have used XAMPP to test my website. XAMPP is an abbreviation for cross-platform, Apache, MySQL, PHP and Perl, and it allows us to build WordPress site offline, on a local web server on our computer. This simple and lightweight solution works on Windows, Linux, and Mac.

## **Procedure followed:**

- I. First of all, I did the front-end part. In this part, I have created the website using HTML, CSS, Bootstrap and JavaScript. This part is basically the appearance, how the website will appear to the student/user.

## **Sign Up Page**

1. Here, the first step is to create the structure of the website using HTML. Basically, the header, the body and the footer section.
2. In the header section I have added an appropriate heading; in the body, I have added the main content of the page, and finally in the footer section; I added the footer content (contact links, social links).

3. The main content consists of the Sign Up and Log In buttons. By clicking the Sign Up button, a popup form will be shown asking the user to register; and if the Log In button is clicked, the user will be redirected to another page specifically for logging in. ( used anchor tags <a> for redirecting to new page). I have used BootStrap modals for the popup forms.

### **Log In Page**

4. The structure of the Log In page is similar to the Sign Up page. Here, in the content part, I have included a login button and a link back to the sign up page.
5. By clicking the login button, a login popup will appear. Using this, the user can login to their account.

### **Books Issue page**

6. This page is for issuing new books by the user. I included a popup form which asks the user to issue a new book.
7. Also, here I have included a link to another page which shows the books that the student has already issued. Alternatively, if the student issues a new book he is redirected to the display page automatically.

### **Book display page**

8. This page is for viewing all the books that a student has issued till date, in the form of a table. I have also included a logout button, which will logout of the account and redirect the user to the login page.

### **Designing the Website**

9. Now, next step is to design the website. For this, CSS is used. Using classes and ids, I have targeted all necessary tags and elements for styling and making the website look beautiful.

10. I have also used Bootstrap wherever required for mobile responsiveness and easy coding ( buttons, forms, etc. ).

### **Validating Input fields**

11. Now, last part of the front-end part to validate the input fields of the forms. This is an important step which helps in correct entering of data, etc. I have validated the required input fields such as First Name & Last Name should be only be letters, phone number should only take input as 10 digits, and input for password and confirm password should be same.
  12. Also, I have added tooltips in the input fields, so that the user can enter their correct details.
- II. Now, after finishing the front-end part, I have done the back-end part using PHP and used XAMPP to test my website as a localhost server.

### **Creating a table using MySQL**

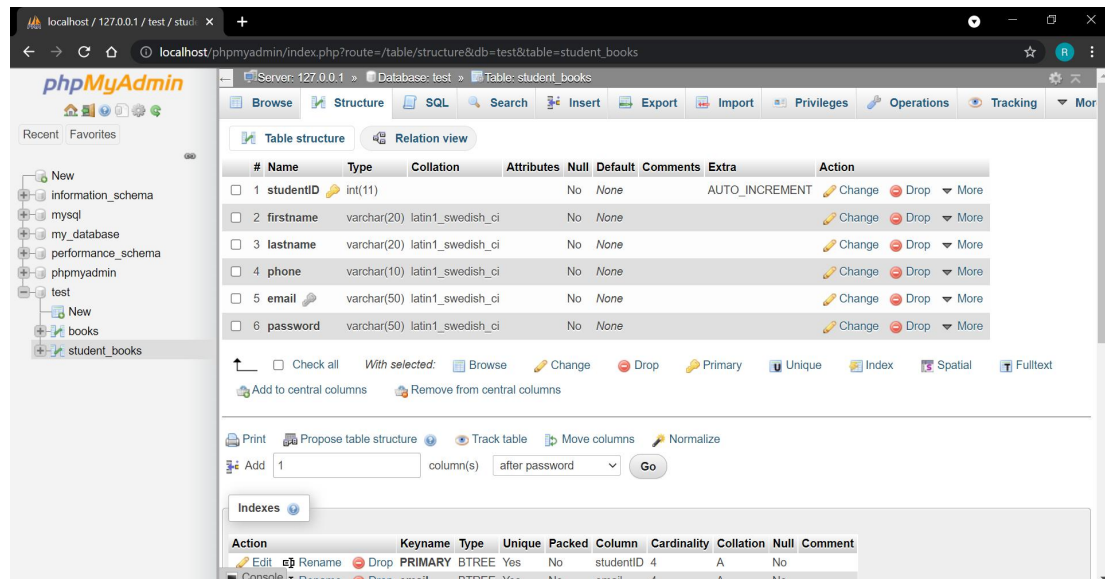
13. Now in the localhost server, I have first created a table 'student\_books' to store the student/user information, with studentID as the primary key and email as unique.
14. Next, I created another table 'books' to store the books information of every student/user, with studentID as the foreign key.

### **Using PHP to connect to the database**

15. Finally, I added all the necessary PHP code required (to insert a new record in the database, in tables 'student\_books' and 'books'; fetch data from the database, etc.) I have also used session variables for the login session of a student. The session starts when the student logs into their account and ends when the student logs out.

# OUTPUT

Structure of the tables ('student\_books' and 'books'):



The screenshot shows the phpMyAdmin interface for the 'test' database. The 'Table structure' tab is selected for the 'student\_books' table. The table has 6 columns: studentID (int(11), PRIMARY, AUTO\_INCREMENT), firstname (varchar(20)), lastname (varchar(20)), phone (varchar(10)), email (varchar(50)), and password (varchar(50)). All columns are NOT NULL and use the latin1\_swedish\_ci collation. The 'Indexes' section shows a PRIMARY index on studentID.

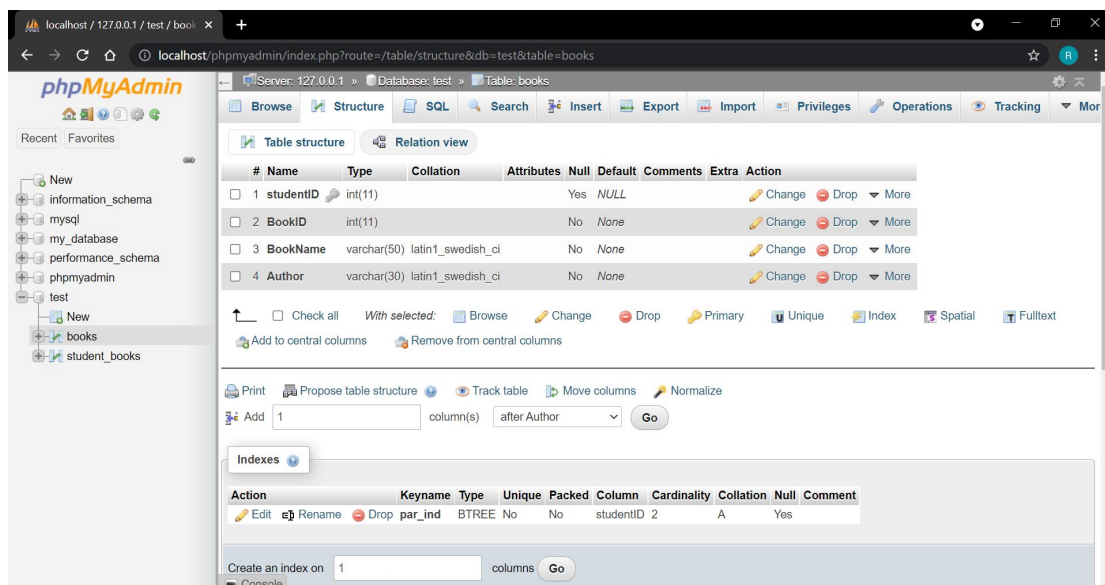
| # | Name      | Type        | Collation         | Attributes | Null | Default | Comments | Extra          | Action           |
|---|-----------|-------------|-------------------|------------|------|---------|----------|----------------|------------------|
| 1 | studentID | int(11)     |                   |            | No   | None    |          | AUTO_INCREMENT | Change Drop More |
| 2 | firstname | varchar(20) | latin1_swedish_ci |            | No   | None    |          |                | Change Drop More |
| 3 | lastname  | varchar(20) | latin1_swedish_ci |            | No   | None    |          |                | Change Drop More |
| 4 | phone     | varchar(10) | latin1_swedish_ci |            | No   | None    |          |                | Change Drop More |
| 5 | email     | varchar(50) | latin1_swedish_ci |            | No   | None    |          |                | Change Drop More |
| 6 | password  | varchar(50) | latin1_swedish_ci |            | No   | None    |          |                | Change Drop More |

| Action           | Keyname | Type  | Unique | Packed | Column    | Cardinality | Collation | Null | Comment |
|------------------|---------|-------|--------|--------|-----------|-------------|-----------|------|---------|
| Edit Rename Drop | PRIMARY | BTREE | Yes    | No     | studentID | 4           | A         | No   |         |
| Edit Rename Drop |         |       |        |        | email     | 4           | A         | No   |         |

Query to create table 'student\_books':

```
CREATE TABLE student_books (studentID int NOT NULL  
AUTO_INCREMENT, firstname varchar(20) NOT NULL, lastname  
varchar (20) NOT NULL, phone varchar(10) NOT NULL, email  
varchar(50) NOT NULL, password varchar(50) NOT NULL, PRIMARY KEY  
(studentID), UNIQUE (email));
```



The screenshot shows the phpMyAdmin interface for the 'test' database. The 'Table structure' tab is selected for the 'books' table. The table has 4 columns: studentID (int(11), PRIMARY, NOT NULL), bookID (int(11), NOT NULL), BookName (varchar(50)), and Author (varchar(30)). All columns use the latin1\_swedish\_ci collation. The 'Indexes' section shows a PRIMARY index on studentID and a UNIQUE index on Author.

| # | Name      | Type        | Collation         | Attributes | Null | Default | Comments | Extra | Action           |
|---|-----------|-------------|-------------------|------------|------|---------|----------|-------|------------------|
| 1 | studentID | int(11)     |                   |            | Yes  | NULL    |          |       | Change Drop More |
| 2 | bookID    | int(11)     |                   |            | No   | None    |          |       | Change Drop More |
| 3 | BookName  | varchar(50) | latin1_swedish_ci |            | No   | None    |          |       | Change Drop More |
| 4 | Author    | varchar(30) | latin1_swedish_ci |            | No   | None    |          |       | Change Drop More |

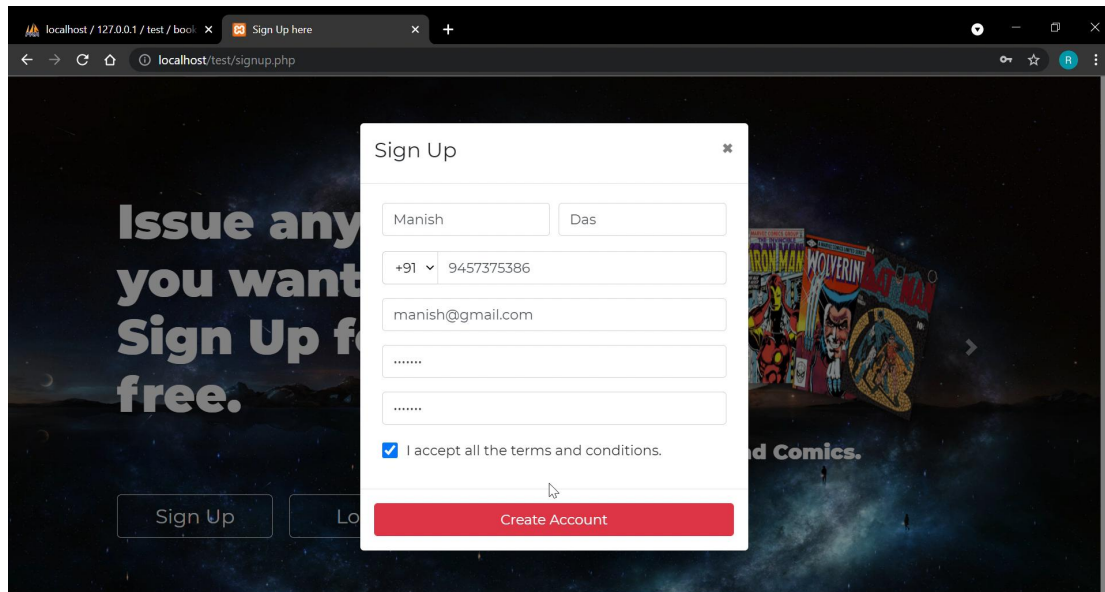
| Action           | Keyname | Type  | Unique | Packed | Column    | Cardinality | Collation | Null | Comment |
|------------------|---------|-------|--------|--------|-----------|-------------|-----------|------|---------|
| Edit Rename Drop | par_ind | BTREE | No     | No     | studentID | 2           | A         | Yes  |         |



Query to create table 'books':

```
CREATE TABLE books (studentID int NOT NULL, BookID int NOT NULL,  
BookName varchar(50) NOT NULL, Author varchar(30) NOT NULL,  
FOREIGN KEY (studentID) REFERENCES student_books(studentID));
```

The Sign up page: Registering a new user 'Manish Das'.



PHP code snippet for inserting a new record in the database:

```
if(isset($_POST['submit'])){  
    $fname=$_POST['f_name'];  
    $lname=$_POST['l_name'];  
    $phone=$_POST['phone'];  
    $email=$_POST['email'];  
    $password=$_POST['password'];  
    $cpassword=$_POST['cpassword'];  
  
    if ($password == $cpassword)  
    {  
        $sql = "SELECT * FROM student_books WHERE email='$email'";  
        $result = mysqli_query($conn, $sql);  
        if (!$result->num_rows > 0)  
        {  
            $sql = "INSERT INTO `student_books` (`studentID`, `firstname`,  
            `lastname`, `phone`, `email`, `password`) VALUES ('NULL', '$fname', '$lname', '$phone',  
            '$email', '$password')";  
            $result = mysqli_query($conn, $sql);  
            if ($result==1)  
            {  
                ?>  
                <script>alert('Successfully Signed in!')</script>  
                <?php
```

```

    }
  }
  else {
    ?>
    <script>alert('Email already exists!')</script>;
    <?php
  }
}
}

```

Here, I am checking if the password is the same as the one entered in the confirm password input field. If it is same then the above code is executed. Once this code is executed successfully, an alert will be shown saying that “Successfully Signed in!”.

MySQL query to insert a new record is:

```

INSERT INTO student_books (studentID, firstname, lastname, phone,
email, password) VALUES ('NULL', '$fname', '$lname', '$phone',
'$email', '$password');

```

While registering if it is found that an account is already present in the database with the same email then “Email already exists” alert is shown.

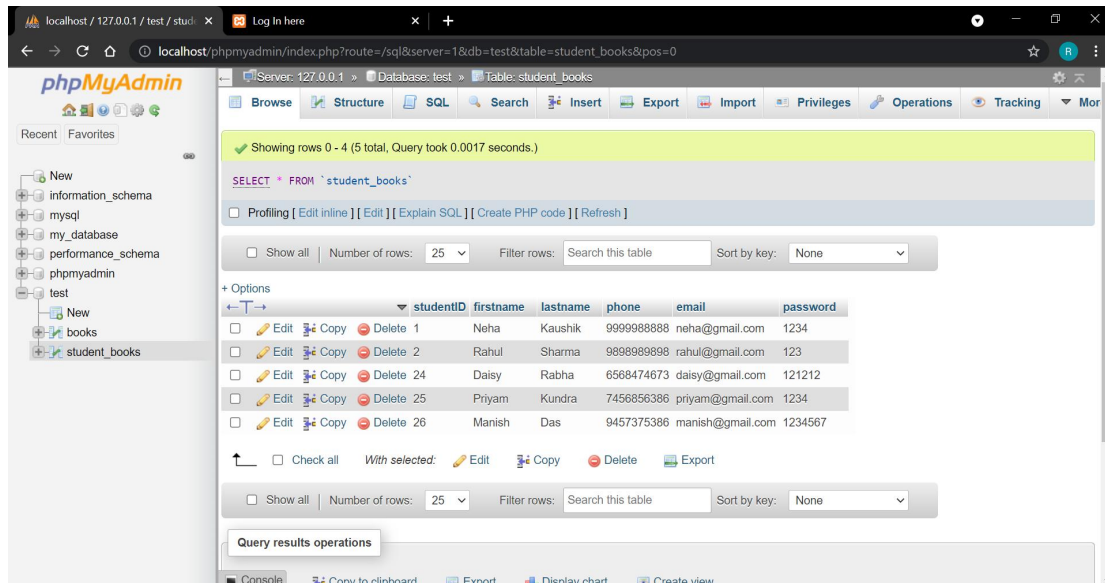
Now, If the password is not same as the one entered in the confirm password field, and alert will be shown saying “Password not matched”. I have done this using JavaScript by validating the required input fields. The code snippet is given below:

```

function passwordconfirm(){
  var pass = document.getElementById("password").value;
  var cpass = document.getElementById("cpassword").value;
  if(pass == cpass){
    return true;
  }
  else{
    alert("Password not matched!");
    return false;
  }
}

```

**When the account is created, data is stored in the database:**

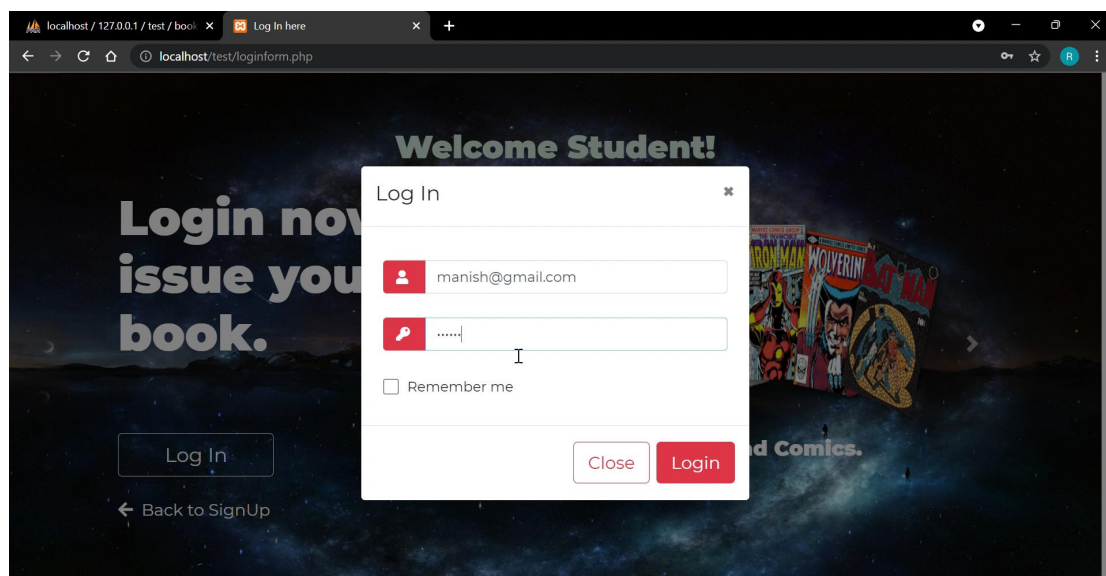


The screenshot shows the phpMyAdmin interface. The left sidebar displays the database structure with 'test' selected. The main area shows the 'student\_books' table with 5 rows. The table columns are studentID, firstname, lastname, phone, email, and password. The rows are as follows:

| studentID | firstname | lastname | phone      | email            | password |
|-----------|-----------|----------|------------|------------------|----------|
| 1         | Neha      | Kaushik  | 9999988888 | neha@gmail.com   | 1234     |
| 2         | Rahul     | Sharma   | 9898989898 | rahul@gmail.com  | 123      |
| 24        | Daisy     | Rabha    | 6568474673 | daisy@gmail.com  | 121212   |
| 25        | Priyam    | Kundra   | 7456856386 | priyam@gmail.com | 1234     |
| 26        | Manish    | Das      | 9457375386 | manish@gmail.com | 1234567  |

A new record is inserted with studentID=26, which is auto incremented.

**The Log In page: Logging into Manish's account.**



**PHP code snippet for logging into user account:**

```
session_start();
if(isset($_POST['login'])){

    $_SESSION['Logged']="No";
    $email = $_POST['email'];
    $password = $_POST['password'];

    include'dbconn.php';
```

```

    $query = "SELECT * FROM student_books WHERE email='$email' AND
password='$password'";
    $data = mysqli_query($conn, $query);

    if($info = mysqli_fetch_array($data))
    {
        $QueryPwd=$info['password'];
        if( $password ==$QueryPwd ) {
            $_SESSION['Logged_expert']="Yes";
            $_SESSION['ID']=$info['studentID'];
            $_SESSION['fname']=$info['firstname'];
            $_SESSION['lname']=$info['lastname'];
            ?>
            <script>
                alert('Successfully Logged in!');
            </script>
            <?php
            header("Location: books.php");
            die;
        }
    }
    else
    {
        ?>
        <script>alert('Wrong Email or Password')</script>
        <?php
    }
}

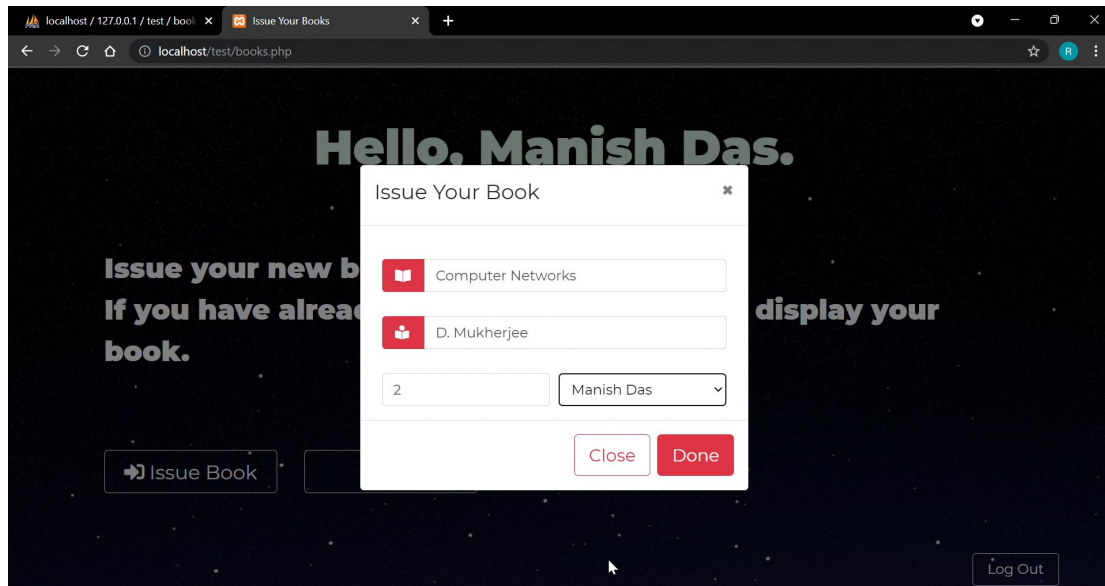
```

Here, I am using session variable for a user. Sessions are a simple way to store data for individual users against a unique session ID. This can be used to persist state information between page requests. Session IDs are normally sent to the browser via session cookies and the ID is used to retrieve existing session data. The absence of an ID or session cookie lets PHP know to create a new session, and generate a new session ID. Sessions can be started manually using the `session_start()` function.

First I am fetching the data from table 'student\_books' from the database by using the MySQL query **SELECT \* FROM student\_books WHERE email='\$email' AND password='\$password';**.

Now I am checking if the information the user enters is same as the data fetched from the database. If so, then the user will be successfully logged into their account. If not, then an alert saying "Wrong Email or Password" is shown.

## Issuing a new book:



## PHP code for inserting new book into the database:

```
session_start();

if (!isset($_SESSION['ID'])) {
    header("Location: loginform.php");
}
$cid=$_SESSION['ID'];

include'dbconn.php';

if(isset($_POST['submitbook'])){
    $query1="INSERT INTO `books` (`studentID`, `BookID`, `BookName`, `Author`)
VALUES ('$cid','" . $_POST['book_id'] . "','" . $_POST['bookname'] . "','" .
$_POST['authorname'] . "')";
    $run1=mysqli_query($conn, $query1);
    if($run1==1){

        ?>
        <script>
            alert('Book has been issued successfully');
        </script>

        <?php
        header("Location: showbooks.php");
        die;
    }
}
else{
    ?>
```

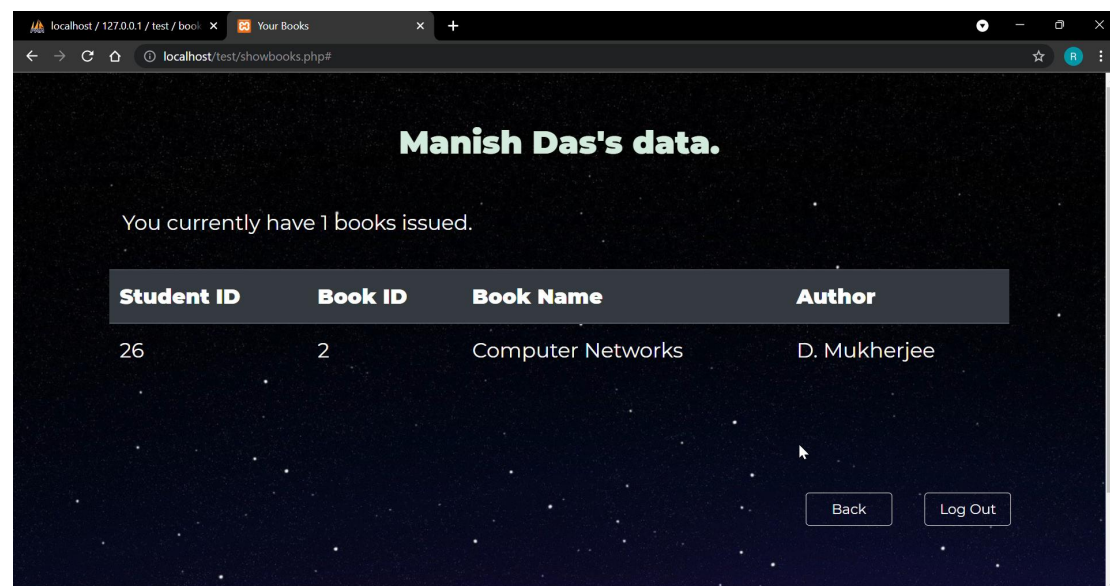
After the user logs into their account, they can either issue a new book or view their already issued books.

## MySQL query to insert new book:

```
INSERT INTO books (studentID, BookID, BookName, Author) VALUES
('$sid', ' ".$_POST['book_id']." ', ' ".$_POST['book_id']." ',
' ".$_POST['book_id']." ');
```

After the user finishes issuing a new book, the data is stored in the books table.

After issuing the book, a table where all books issued by Manish till date is shown:



The screenshot shows a web browser window with the URL `localhost/test/showbooks.php#`. The page has a dark background with a starry pattern. At the top, it says "Manish Das's data." Below that, it says "You currently have 1 books issued." There is a table with the following data:

| Student ID | Book ID | Book Name         | Author       |
|------------|---------|-------------------|--------------|
| 26         | 2       | Computer Networks | D. Mukherjee |

At the bottom right, there are two buttons: "Back" and "Log Out".

Code for counting number of books and displaying data:

```
<section id="main-content">
    <div class="container-fluid">
        <div class="row heading">
            <h2><?php echo $_SESSION['fname']." ".$_SESSION['lname']."'s
data."; ?></h2>
        </div>
        <div class="row sub-heading">
            <p class="book-count">You currently have
                <?php
                    $count_query=mysqli_query($conn,"SELECT COUNT(*) FROM books
where studentID='$sid'");
                    while($count=mysqli_fetch_array($count_query)){
                        echo $count['COUNT(*)'];
                    }
                ?>
                books issued.</p>
        </div>
        <div class="row display-table">
            <table class="table">
                <thead class="thead-dark">
                    <tr>
                        <th scope="col">Student ID</th>
                        <th scope="col">Book ID</th>
                        <th scope="col">Book Name</th>
```



```

        <th scope="col">Author</th>
    </tr>
</thead>
<tbody>
<tr>
<?php
    include 'dbconn.php';

    $sql = mysqli_query($conn, "SELECT * FROM books where
studentID='$sid'");

    while($row2 = mysqli_fetch_array($sql))
    {
        ?>
        <td><?php echo $sid ?></td>
        <td><?php echo $row2['BookID']; ?></td>
        <td><?php echo $row2['BookName']; ?></td>
        <td><?php echo $row2['Author']; ?></td>

    </tr>
    <?php } ?>
</tbody>
</table>
</div>
</div>
</section>

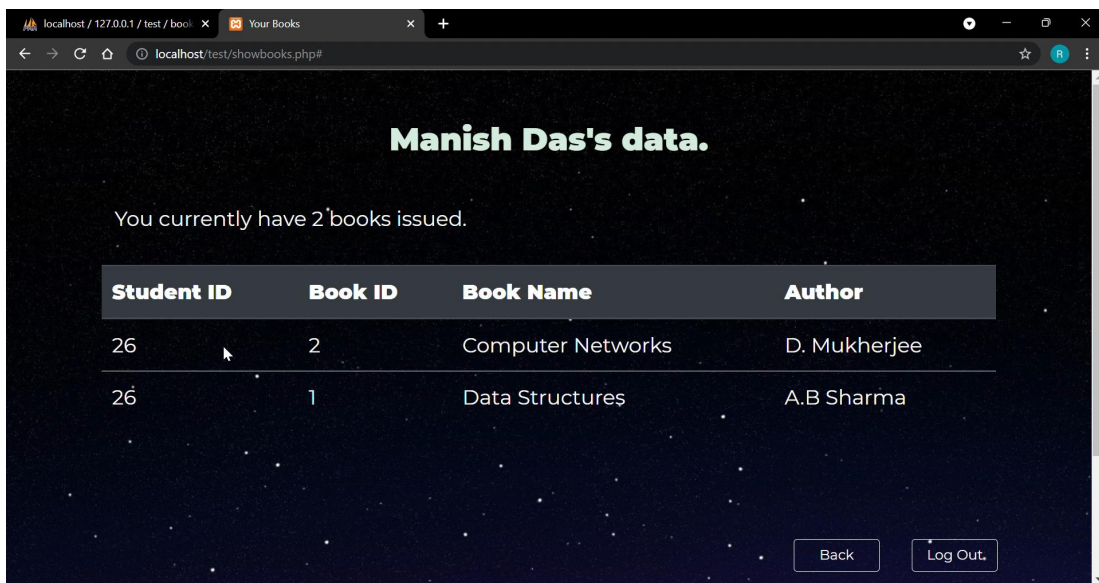
```

Here, I am counting the number of books the user has issued, and displaying it in the form of a table with all the book information.

MySQL query for counting total books issued by the user:  
**SELECT COUNT(\*) FROM books WHERE studentID= '\$sid';**

MySQL query for fetching all data of the user:  
**SELECT \* FROM books WHERE studentID='\$sid';**

After issuing another book, the result will be:



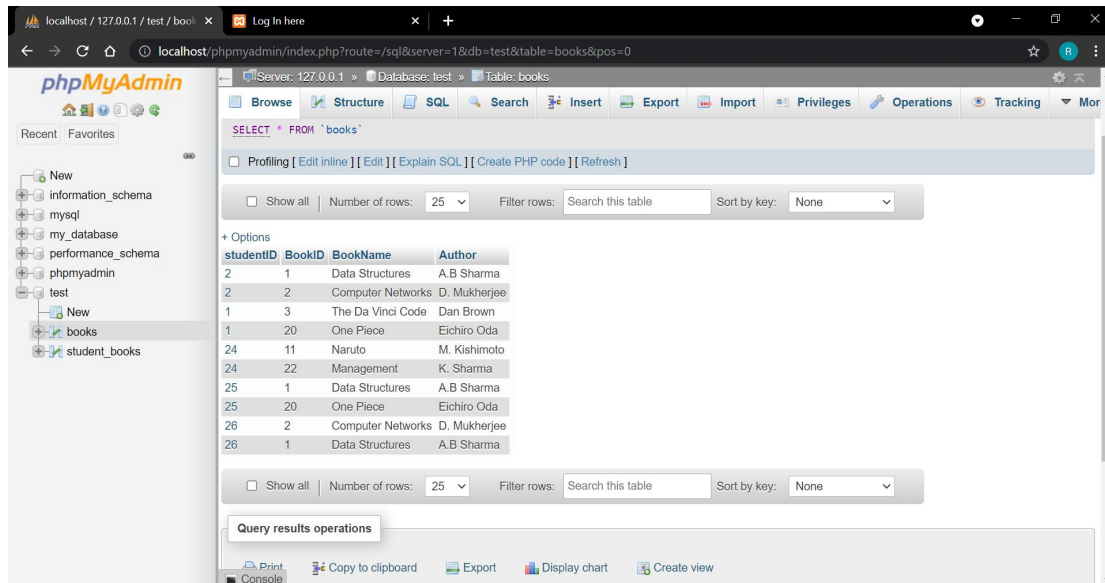
**Manish Das's data.**

You currently have 2 books issued.

| Student ID | Book ID | Book Name         | Author       |
|------------|---------|-------------------|--------------|
| 26         | 2       | Computer Networks | D. Mukherjee |
| 26         | 1       | Data Structures   | A.B Sharma   |

Back Log Out.

## Data in the database:



The screenshot shows the phpMyAdmin interface for a database named 'test'. The 'books' table is selected, and its data is displayed in a table format. The table has four columns: studentID, BookID, BookName, and Author. The data is as follows:

| studentID | BookID | BookName          | Author       |
|-----------|--------|-------------------|--------------|
| 2         | 1      | Data Structures   | A.B Sharma   |
| 2         | 2      | Computer Networks | D. Mukherjee |
| 1         | 3      | The Da Vinci Code | Dan Brown    |
| 1         | 20     | One Piece         | Eiichiro Oda |
| 24        | 11     | Naruto            | M. Kishimoto |
| 24        | 22     | Management        | K. Sharma    |
| 25        | 1      | Data Structures   | A.B Sharma   |
| 25        | 20     | One Piece         | Eiichiro Oda |
| 26        | 2      | Computer Networks | D. Mukherjee |
| 26        | 1      | Data Structures   | A.B Sharma   |

## Logging out of the session:

```
session_start();
session_destroy();

header("Location: loginform.php");
```

The above code is used to logout of a current session. The user is then redirected to the login page and can no longer go back to the previous page.

---

To see project demo video , [click here](#).

To see all codes related to this project, [click here](#).



# CONCLUSION

A proper orientation is a needful if we have to record thousands of data of students everyday of an institution or organization. So a database is necessary where the data can be stored and recorded for every student, which is unique and not identical. Student Library Portal is a must on such cases as it can help save a great deal of time and effort. The prime benefits of the system are to reduce overheads and increase productivity. All big organizations use such management systems to record data.

In this era of the Internet, almost every person has a connection to the World Wide Web. Hence, such systems make life easy, not only for the librarians but also for the user themselves. Users can easily access the cloud or web-based application from anywhere at any time by a PC or mobile. Also, this helps reduce manual processes to issue books and maintain records and reduces the overall library's operating cost.

Thus, a Student Library Portal is essential to run smart school functions, and maintain accurate data of a library.

# REFERENCES

The following references have been used for the completion of this project:

1. <https://www.wikipedia.org>
2. <https://developer.mozilla.org>
3. <https://www.youtube.com>
4. <https://www.w3schools.com>
5. <https://www.google.com>
6. <https://www.php.net>