**DEPARTMENT LIBRARY MANAGEMENT SYSTEM**

**A MINI PROJECT REPORT**

**Submitted in partial fulfillment of the requirements for the award of the degree of**

**MASTER OF COMPUTER APPLICATIONS**

**Submitted By**

**GOKUL.S**

**Register No. 22PCA242507**

**Under the Guidance of**

**Dr. C. SENTHAMARAI MCA, Ph.D.**

**Assistant Professor of Computer Applications**



**DEPARTMENT OF COMPUTER APPLICATIONS**

**GOVERNMENT ARTS COLLEGE (AUTONOMOUS)**

**(Re-Accredited by NAAC with B Status) SALEM 636007**

**OCTOBER 2023**

# CERTIFICATE

This is to certify that the project work entitled “**DEPARTMENT LIBRARY MANAGEMENT SYSTEM”** submitted to Government Arts College (Autonomous), Salem-7. In partial fulfillment requirements for the award of degree of **MASTER OF COMPUTER APPLICATIONS** is a record of project work done by [**GOKUL.S], (Register No: 22PCA242507)** between May 2022 under my supervision and guidance.

Date :  **Signature of the Guide**

Place : Salem-7

**Head of the Department**

Submitted for the Autonomous Viva-Voce Examination held on……………………..

## Internal Examiner External Examiner

**DECLARATION**

I hereby declare that the mini project work entitled “**DEPARTMENT LIBRARY MANGEMENT SYSTEM**” submitted to Government Arts College (Autonomous) Salem-7 in partial fulfillment of the requirements for the award of degree of **MASTER OF COMPUTER APPLICATIONS** is a record of original work done by me under the supervision and guidance of **Dr. C. SENTHAMARAI MCA., Ph.D.,** Assistant Professor in Department of Computer Applications, Government Arts College (Autonomous), Salem-7.

Signature of the Candidate

(**GOKUL.S**)

**REGISTER NUMBER:**

22PCA242507

Date:

Place: Salem-7.

# ACKNOWLEDGEMENT

At the outset, I would like to thank and honor God who gave me the wisdom and knowledge to complete this project.

I would like to profusely thank to our college principal **Dr. N. SHENBAGALAKSHIMI, MA.(Tamil) MA.(JMC), M.Sc.,(Yoga) B.Ed., M.Phil., Ph.D.,(NET),** Government Arts College (Autonomous), Salem-7 for her guidance and encouragement to successfully complete my project.

I express my gratitude **Dr. A. KANGAIAMMAL MCA, MPhil, ME, Ph.D.** Head of the Department of Computer Applications, Government Arts College (Autonomous), Salem-7 for being an unfailing source of inspiration and encouragement.

I would like to express my gratitude and sincere thanks to my guide of

**Dr. C. SENTHAMARAI MCA, Ph.D.** Assistant Professor**,** Department of Computer Applications, Government Arts College (Autonomous), Salem-7 for her source of guidance and suggestions throughout the project.

I express my sincere thanks to all other staff members of computer science department for this supporting help.

This acknowledgement will be incomplete without expressing my thanks to my parents

and friends for their constant support and encouragement.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER** | **CONTENTS** | **P.NO** |
|  | **ABSTRACT** |  |
| **1** | **INTRODUCTION** |  |
|  | 1.1 Project Description |  |
| **2** | **SYSTEM ANALYSIS** |  |
| 2.1 Existing System |  |
| 2.1.1 Drawbacks of Existing system |  |
| 2.2 Proposed System |  |
| 2.2.1 Advantages of Proposed System |  |
| 2.3 System Requirement and specification |  |
| 2.3.1 Hardware Requirements |  |
| 2.3.2 Software Requirements |  |
| 2.3.3 Software Description |  |
| **3** | **SYSTEM DESIGN AND DEVELOPMENT** |  |
| 3.1 Data Flow Diagram |  |
| 3.2 Module Description |  |
| 3.3 Table Design |  |
| 3.4 Input Design |  |
| 3.5 Output Design |  |
| **4** | **TESTING AND IMPLEMENTATION** |  |
| 4.1 System Testing |  |
| 4.2 System Implementation |  |
| **5** | **CONCLUSION** |  |
|  | **FUTURE ENHANCEMENT** |  |
|  | **BIBLIOGRAPHY** |  |
| Reference Books |  |
|  | **APPENDIX** |  |

**DEPARTMENT LIBRARY MANAGEMENT SYSTEM**

**Abstract:**

A “Department Library Management System” is a project that manages and stores books information electronically according to student’s needs.

The system helps both students and library manager to keep a constant track of all the books available in the library.

It allows both the admin and the student to search for the desired book. It becomes necessary for department to keep a continuous check on the books issued and returned and even calculate fine.

This task if carried out manually will be tedious and includes chances of mistakes.

These errors are avoided by allowing the system to keep track of information such as issue date, last date to return the book and even fine information and thus there is no need to keep manual track of this information which thereby avoids chances of mistakes.

Thus this system reduces manual work to a great extent allows smooth flow of library activities by removing chances of errors in the details Library Management System.

Students need access to authentic information.

The librarian can answer queries with ease regarding the availability of books. Adding, removing or editing the database is a simple process.

**FRONT END : PHP**

**BACK END : MYSQL**

**CHAPTER 1**

**Introduction**

The purpose of a Library Management System is to operate a library with efficiency and at reduced costs.

The system being entirely automated streamlines all the tasks involved in operations of the library.

The activities of book purchasing, cataloguing, indexing, circulation recording and stock checking are done by the software.

Such software eliminates the need for repetitive manual work and minimizes the chances of errors.

The library management system software helps in reducing operational costs

An automated system reduces the need for manpower and stationery.

The system saves time for both the user and the librarian.

With just a click the user can search for the books available in the library.

The librarian can answer queries with ease regarding the availability of books. Adding, removing or editing the database is a simple process.

Adding new members or cancelling existing memberships can be done with ease.

Stock checking and verification of books in the admin can be done within a few hours.

The automated system saves a considerable amount of time as opposed to the manual system.

Students need access to authentic information.

**MODULES:**

* Student Module
* staff Module
* Search Module
* Report Generation

**OBJECTIVES:**

**User Access Control:**

To restrict access to department-owned materials only to authorized department members, including faculty, researchers, students, and staff.

**Online Catalog and Search:**

To provide an accessible online catalog or search interface, allowing department members to easily discover and locate materials within the department's collection.

**Reporting and Analytics:**

To generate reports and gather insights into departmental library usage, circulation patterns, and collection management. This data informs decisions about resource acquisition and allocation.

**User-Friendly Interface:**

To offer a user-friendly interface that requires minimal training, ensuring that department members can navigate and utilize the system effortlessly.

**Security and Privacy:**

To ensure the security and privacy of departmental library records, safeguarding sensitive information about users and their borrowing history.

**CHAPTER 2**

**SYSTEM ANALYSIS**

**2.1 EXISTING SYSTEM:**

In the department Library Management System, librarian maintains the details of each book on the registers so to find out the number of books available in the library they need to go to check the entire entry which makes the process slow. For a student who wants to know about a book in the library need to search the entire book section. The student needs to check the status of the book means the last date of book, how many books has issued.

**2.1.1 DRAWBACKS OF EXISTING SYSTEM:**

* In paper method admin can calculate for each student’s library record.
* It takes more time and energy.
* Huge amount of data cannot be maintained manually.
* More Manpower is required.

**2.2 PROPOSED SYSTEM:**

The proposed Online Library Management System project will help the students and librarian to maintain the details of the library. It will assist the librarian before the shortage of books while they can know the details of the number of currently available in the library according to the author by accessing the system. A student can view the details of the book issued by them, and the system will notify the students about the last date of submission of books. At the time of issue of a book, the student will get assisted by the system about different authors of a similar book so that they can get the best available book from the library.

**2.2.1 ADVANTAGE:**

* Reduces Manual Work and Saves Time
* Record Maintenance
* Free from errors

**2.3 SYSTEM REQUIREMENTS**

**2.3.1 HARDWARE REQUIREMENTS:**

* System : Pentium IV 2.4 GHz.
* Hard Disk : 500 GB.
* Monitor : 15 VGA Colour.
* Mouse : Logitech.
* RAM : 4 GB.

**2.3.2 SOFTWARE REQUIREMENTS:**

* Operating System : Windows-7/8/10.
* Frontend : PHP 7.1.
* Backend : MYSQL 10.3
* Web Server : XAMPP 3.2.4

#### 2.3.3 SOFTWARE DESCRIPTION:

**PHP**

PHP is used for creating a department library management system helpful for students and staff easy to retrieve the desired book.

**SQL:**

SQL is used for connecting the database for books, students and staffs details.

**CHAPTER 3**

**DATA FLOW DIAGRAM**

Report Generation

Column wise

Global

Request book

Update Password

Update profile

Log In

Student

Search

Staff

Dashboard

Dashboard

View book

View book

Search book

Search book

Request book

Update profile

Log Out

Start

**3.2 MODULES DESCRIPTION**

* Student Module
* Staff Module
* Search Module
* Report Module

**STUDENT MODULES:**

In this module students can login the account though the valid login string which is provided by the admin, after login the student can view, search the desired books. Also student can update his/her profile like mob.no, password etc. The student module working is,

• View Books

• Search Books

• Request Book

• Update Profile

• Update Password

**STAFF MODULES:**

In this module staffs can login the account though the valid login string which is provided by the admin, after login the staff can view, search the desired books. Also staff can update his/her profile like mob.no, password etc. The staff module working is,

• View Books

• Search Books

• Request Book

• Update Profile

• Update Password

**SEARCH MODULES:**

Search module is used to search the desired book in the library, located to which position of the book. Global and Advanced search to search the desired book. In this module find the berrow, rack, position of the book etc.

* Column wise search
* Global search

**REPORT MODULES:**

Report module is used to annual of the year generating the report submits to HOD and Principal.

**Advantages of this Project:**

* It reduces the manual work.
* Students can find book easily.
* Easy to maintain the record.
* Save the time of students and management of the department.
* It enhances the security of library resources.

**REPORT GENERATION:**

All the above mentioned data are stored in the back end and can be retrieved as reports with filtering options. Admin only have an authority to modify the database only user can view the website. Finally if admin need report they can make a report of the whole databases.

**3.3 TABLE DESIGN**

In department library management system 3 tables are created for the proposed project. They are:

* + - Student table
    - Staff table
    - Book table

**Table 3.3.1 Student table:**

|  |  |
| --- | --- |
| **Books Table** | |
| **Column** | **Type** |
| St\_id | Int |
| Regno | varchar(20) |
| Sname | varchar(150) |
| Gender | enum(’male’,’female’) |
| Spass | varchar(150) |
| Stemail | varchar(150) |
| Contact | varchar(12) |
| Did | Int(11) |
| Year | varchar(15) |
| Shift | varchar(5) |
| Role | char(24) |

**st\_id :** unique student id

**reg\_no :** students register number

**sname :** name of the student

**gender :** gender of the student(male,female)

**spass :** students password(encrypted)

**stemail :** email of the students

**contact :** mobile number of the students

**shift :** shift I or Shift II

**Appendix student table 3.3.1**

**Table 3.3.2 Staff table:**

|  |  |
| --- | --- |
| **Books Table** | |
| **Column** | **Type** |
| Sid | Int |
| Regno | varchar(20) |
| Sname | varchar(150) |
| Gender | enum(‘boy’,’girl’,’male’,’female’) |
| Spass | varchar(150) |
| Semail | varchar(150) |
| Contact | varchar(12) |
| Did | Int(11) |
| Year | varchar(15) |
| Shift | varchar(5) |
| Role | char(24) |
| Designation | int(11) |

**sid :** staffs unique id

**regno :** staffs register number

**sname :** staffs name

**gender :** staffs gender(male,female)

**spass :** staffs password(encrypted)

**semail :** staffs email address

**contact :** staffs mobile number

**did :** staffs department id

**year :** year of joining

**shift :** staffs shift(I or II)

**role :** staff

**designation :** designation of the staff(assist.prof., guest lecture etc)

**Appendix staff table 3.3.2**

**Table 3.3.3 Book table:**

|  |  |
| --- | --- |
| **Books Table** | |
| **Column** | **Type** |
| Bid | Int |
| Bno | varchar(20) |
| Bcode | varchar(26) |
| Title | varchar(150) |
| Aname | aname varchar(150) |
| Publication | varchar(150) |
| Price | varchar(150) |
| year\_of\_publication | varchar(150) |
| Language | varchar(150) |
| Edition | varchar(24) |
| shelf\_id | Int |
| Remark | varchar(500) |
| Status | Int |

**Bid :** id of the book

**Bno :** book number

**Bcode :** barcode of the book

**Title :** title of the book

**Anme :** author name of the book

**Publication :** publication of the book

**Price :** price of the book

**y.o.p :** year of publication of the book

**language :** language of the book(tamil/English)

**edition :** year of edition of the book

**shelf\_id :** shelf id locate the book

**status :** status of the book(available/not available)

**Appendix book table 3.3.3**

**SYSTEM DESIGN:**

**3.4 Input Design**

The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy.

There are 3 input forms are proposed in this project. They are

* + - Login form
    - Update profile
    - Search books

**3.4.1 Login form:**

Login form is used to login with own details like reg.no and password and can access the department library both students and staffs can login and access the library.

**Appendix 3.4.1 login form.**

**3.4.2 Update profile:**

Update form is used to update the student/staff personal details.

**Appendix 3.4.2 update profile**

**3.4.3 Search book:**

Search book is used to search the desired book at book position, berrow, rack etc.

Two ways to search the book column wise and global search.

**Appendix 3.4.3 search book**

**3.5 OUTPUT DESIGN:**

* A quality output is one, which meets the requirements of the end user and presents the information clearly.
* Dashboard, book search, reports, borrow book list, profile, request book

**Appendix 3.5.1 dashboard, 3.5.2 search book, 3.5.3 book status, 3.5.4 request list**

## CHAPTER 4

## Testing and Implementation

### 4.1 SYSTEM TESTING

System testing is the state of implementation, which is aimed at ensuring that the System Works accurate and efficient as expect before, live operation, commences. It certifies that the whole set of programs hang together system testing requires a test plan, that consist of several key activities and step for run program, string, system and user acceptance testing. The implementation of newly design package is important in adapting a successful new system

Testing is important stage in software development system test is implementation should be as confirmation that all is correct and opportunity to show the user that the system works as they expected it accounts the largest percentage of technical effort in software development process

Testing phase development phase that validates the code against the functional specification. Testing is a vital at the achievement of the system goals the objective of testing is to discover errors, to fulfill this objective a series of test step such as the unit test, integration, validation and system test where planned and executed.

**UNIT TESTING**

Unit testing are called module testing. The following modules have been tested

* Login module - valid username and password.
* Search module - valid book

### INTEGRATION TESTING

Following modules have been integrated for evaluating the efficiency of interface

* Staff module - Login, update.
* Student module - login, update.
* Admin modules - insert, update, delete and retrieve.

**VAILIDATION TESTING**

Software validation is achieved through a series of test that demonstrates the conformity and requirement thus the proposed system under consideration has to be tested by validation and found to be working satisfactorily

### 4.2 SYSTEM IMPLEMENTATION

Implementation is the final and important phase, the most critical stage in achieving a successful new system and giving the user confidence. That the new system will work be effective. The system can be implemented only after through testing is done and if it found to working according to the specification

**CHAPTER 5**

**CONCLUSION**

* The Project “Department Library management system” is designed in order to reduce the burden of maintaining bulk of records of all the student reports.
* It supports encryption and authentication mechanisms, helping to prevent unauthorized access and data tampering.
* Maintaining the project is also easy which can be easily understandable.
* Maintaining the details in the database is manageable.

**FUTURE ENHANCEMENTS**

* Due to the lack of time, the design part is not done so attractive.
* Further enhancements can be made in designing the screens.
* Some more forms can also be added so as to better retrieve the book details.
* Various other options can also be added for the better usability of project.

**BIBLIOGRAPHY**

**PHP BOOK REFERENCE**

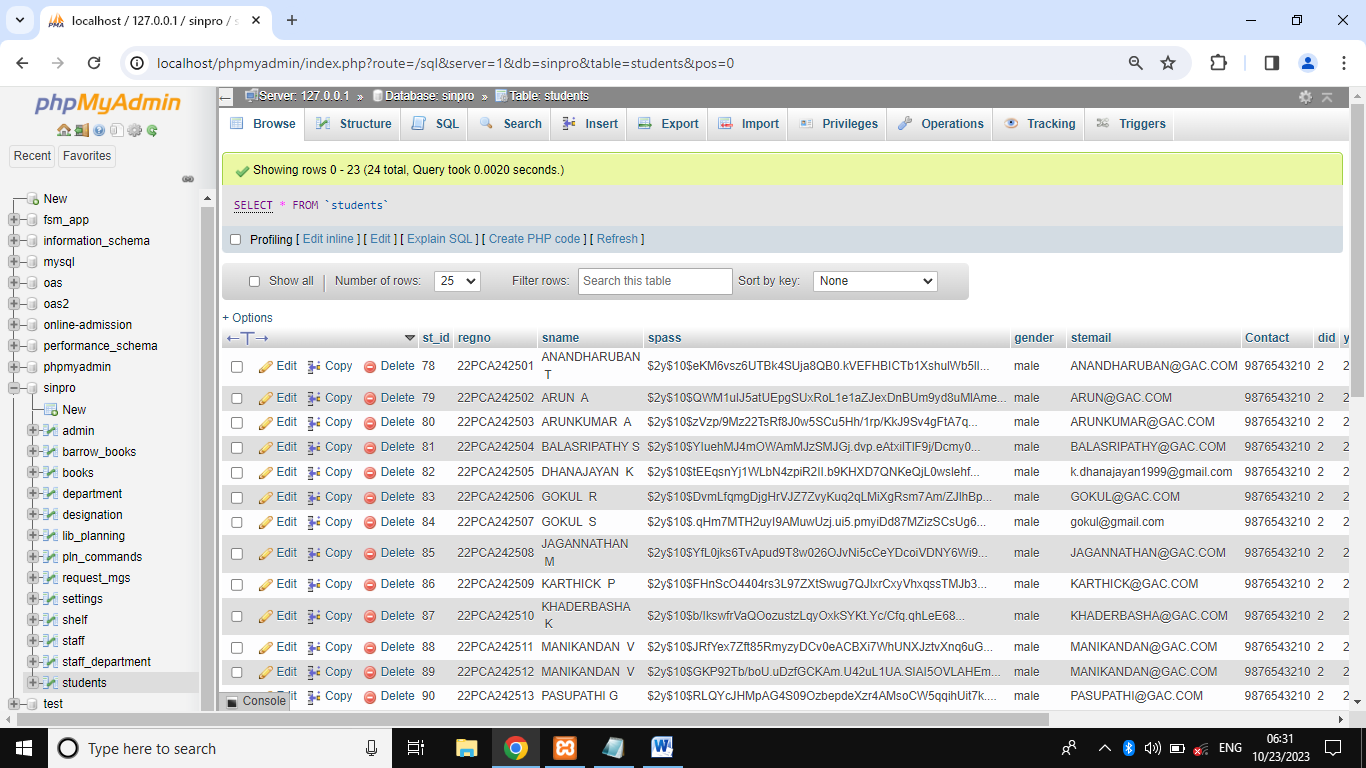
1. Doyle, Matt. Beginning PHP 5.3 (Wrox Programmer to Programmer), 2009.
2. Lerdorf, Rasmus. PHP Pocket Reference, O'Reilly, 2000.
3. Nixon, Robin. Robin Nixon's PHP Crash Course: Learn PHP in 14 easy lectures, O'Reilly, 2012.
4. Ullman, Larry. PHP Advanced and Object-Oriented Programming: Visual QuickPro Guide (3rd Edition), 2012.
5. Ullman, Larry. PHP for the Web: Visual QuickStart Guide, 2011.

**MY SQL BOOK REFERENCE**

1. Nixon, Robin. Learning PHP, MySQL, JavaScript, and CSS: A Step-by-Step Guide to Creating Dynamic Websites, O'Reilly, 2012.
2. Welling, Luke and Thomson, Laura. PHP and MySQL Web Development, Third Edition, Sams, 2008.
3. Database Programming with JDBC and Java. George Reese. O’Reilly Media. Sebastopol, CA. 2000. 1-565-92616-1.
4. JavaServer Pages. Hans Bergsten. O’Reilly Media. Sebastopol, CA. 2003. 0-596-00563-6

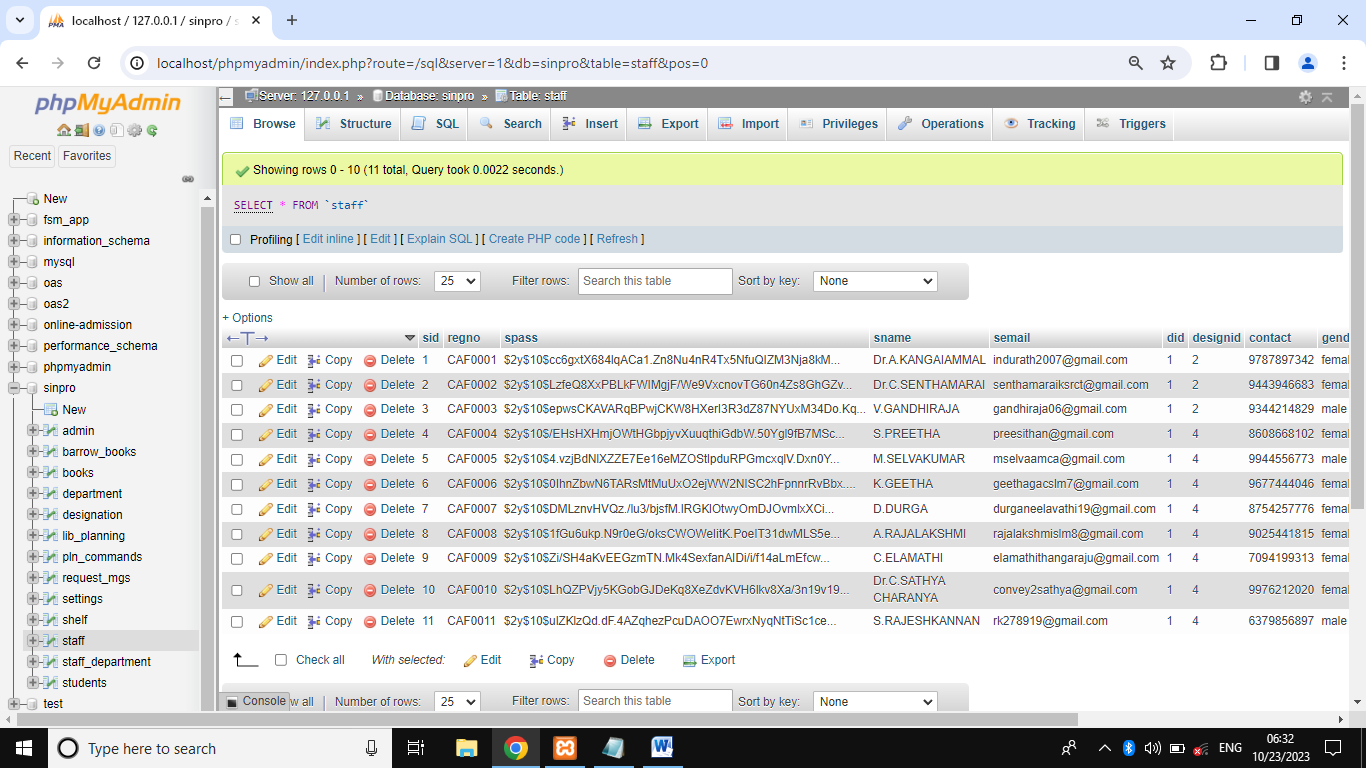
**APPENDIX**

**STUDENT TABLE:**

****

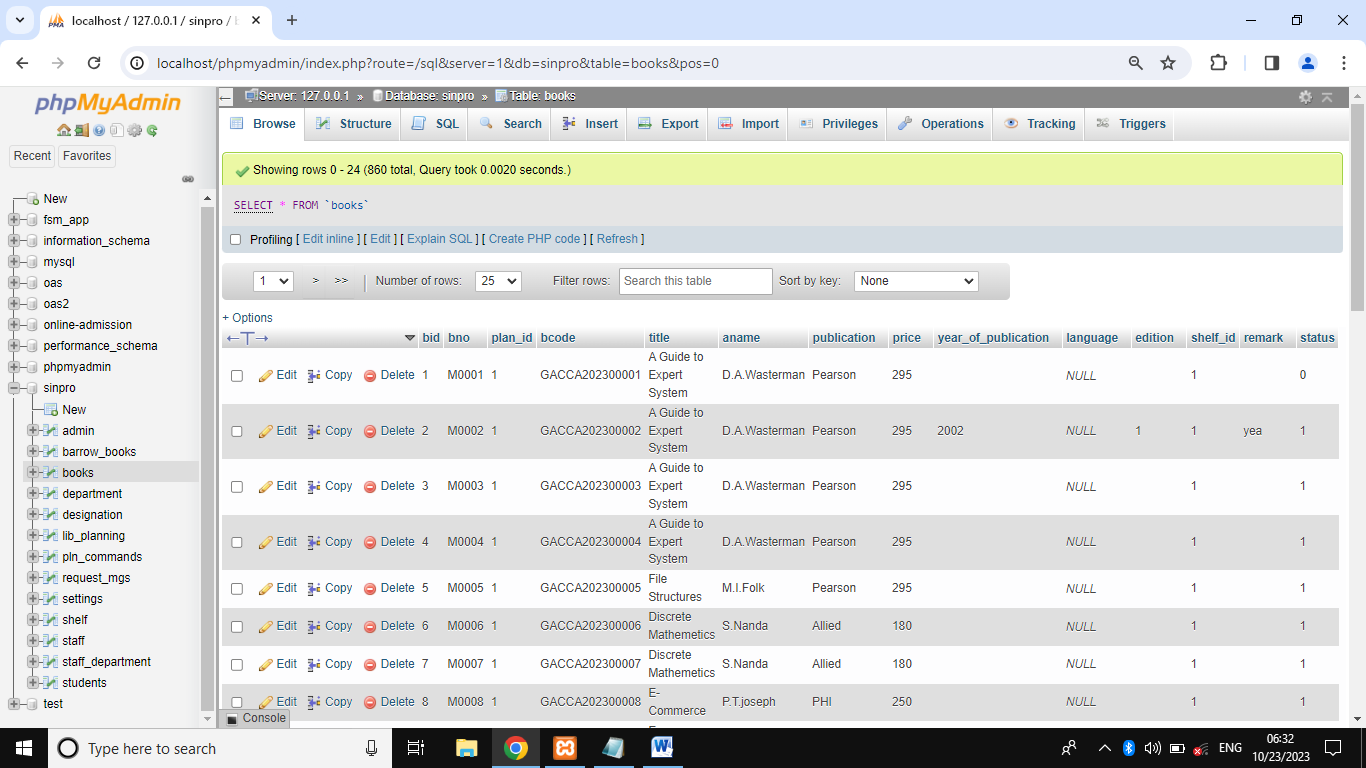
**Fig 3.3.1 student’s database table**

**STAFF TABLE:**

****

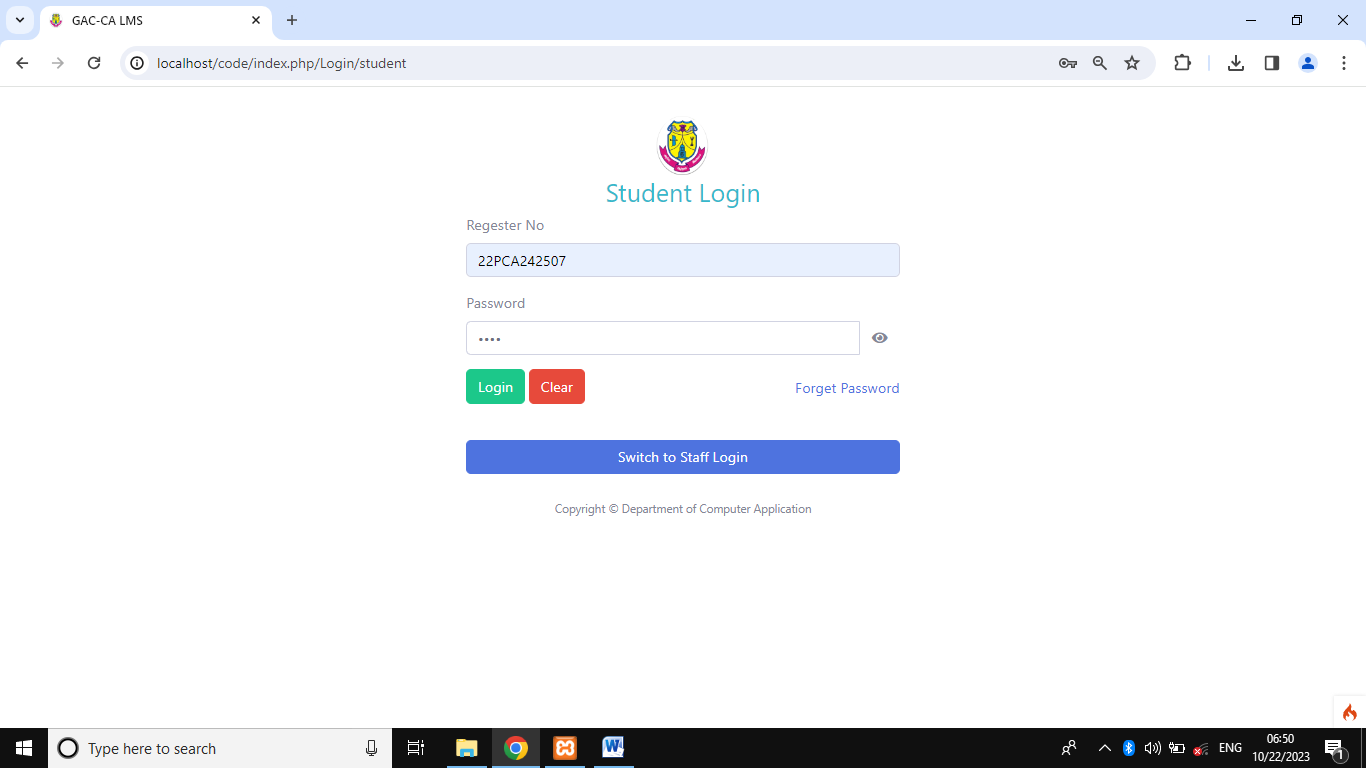
**Fig 3.3.2 staff’s database table**

**BOOK TABLE:**

****

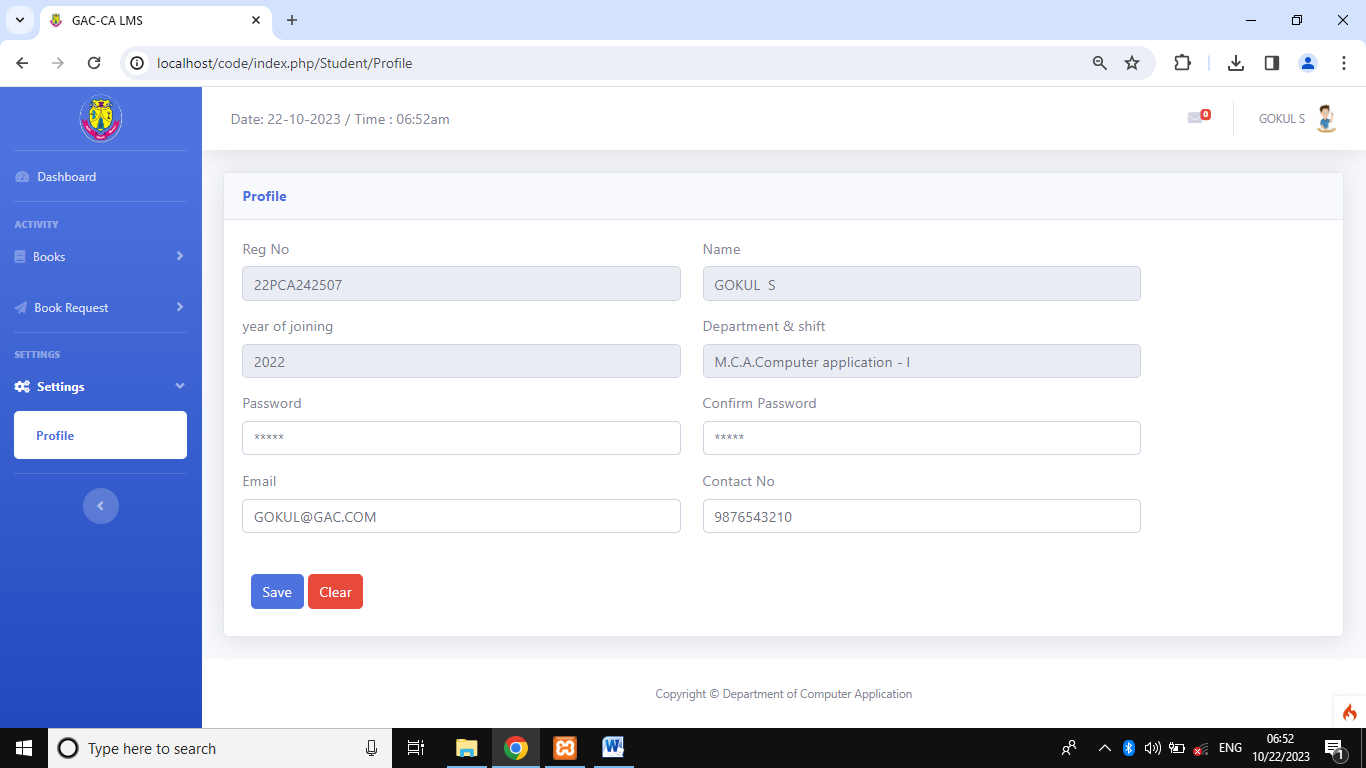
**Fig 3.3.3 book’s database table**

**STUDENT LOGIN FORM:**

****

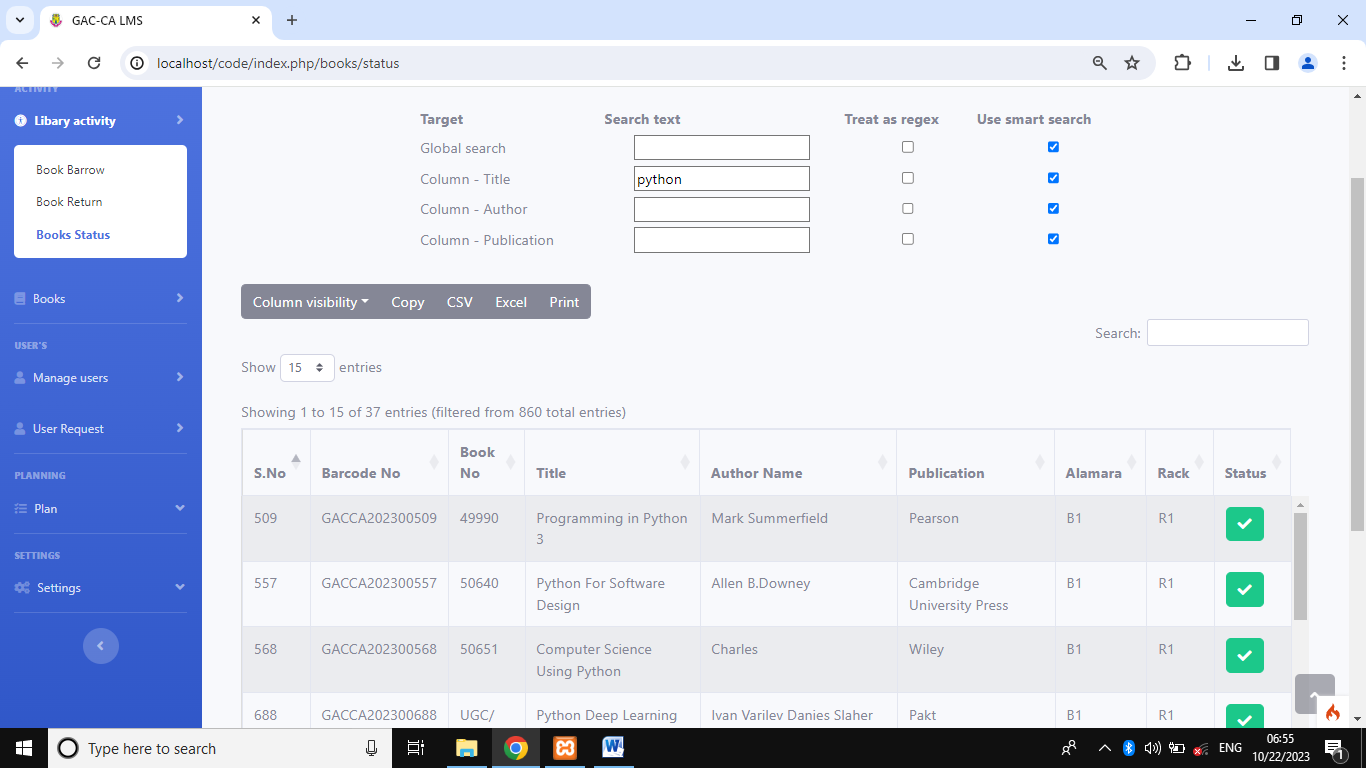
**Fig 3.4.1 (a) student login forms**

**UPDATE PROFILE:**

****

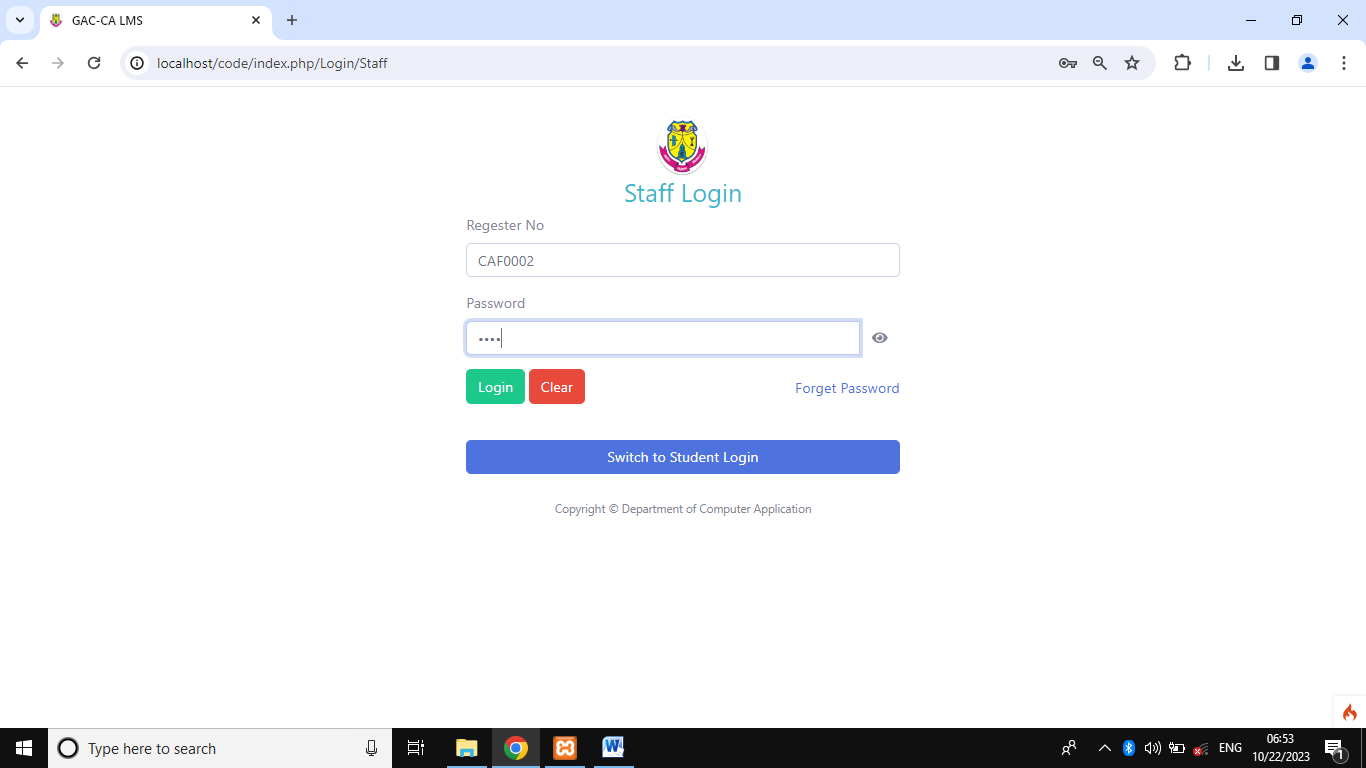
**Fig 3.4.2(b) student’s update profile**

**SEARCH BOOK:**

****

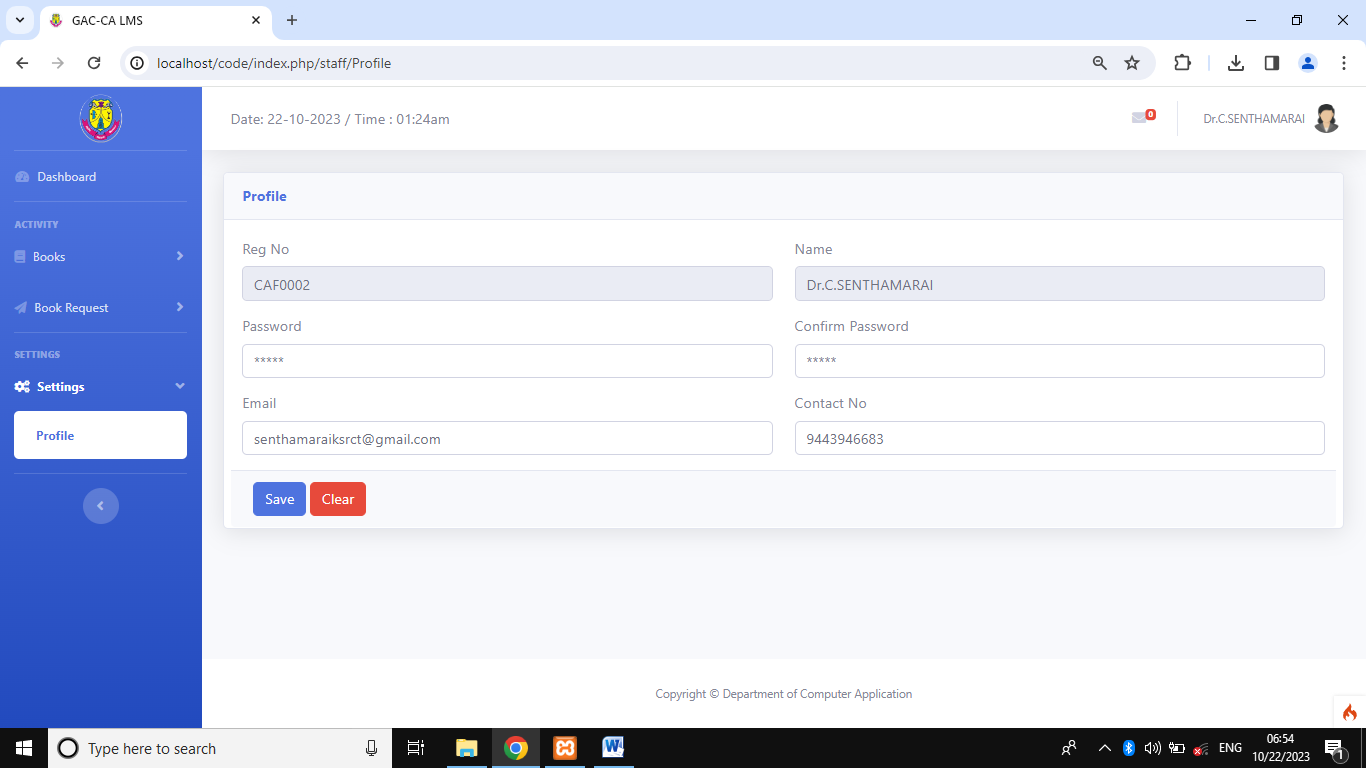
**Fig 3.4.3 (c) search the desired book**

**STAFF LOGIN:**

****

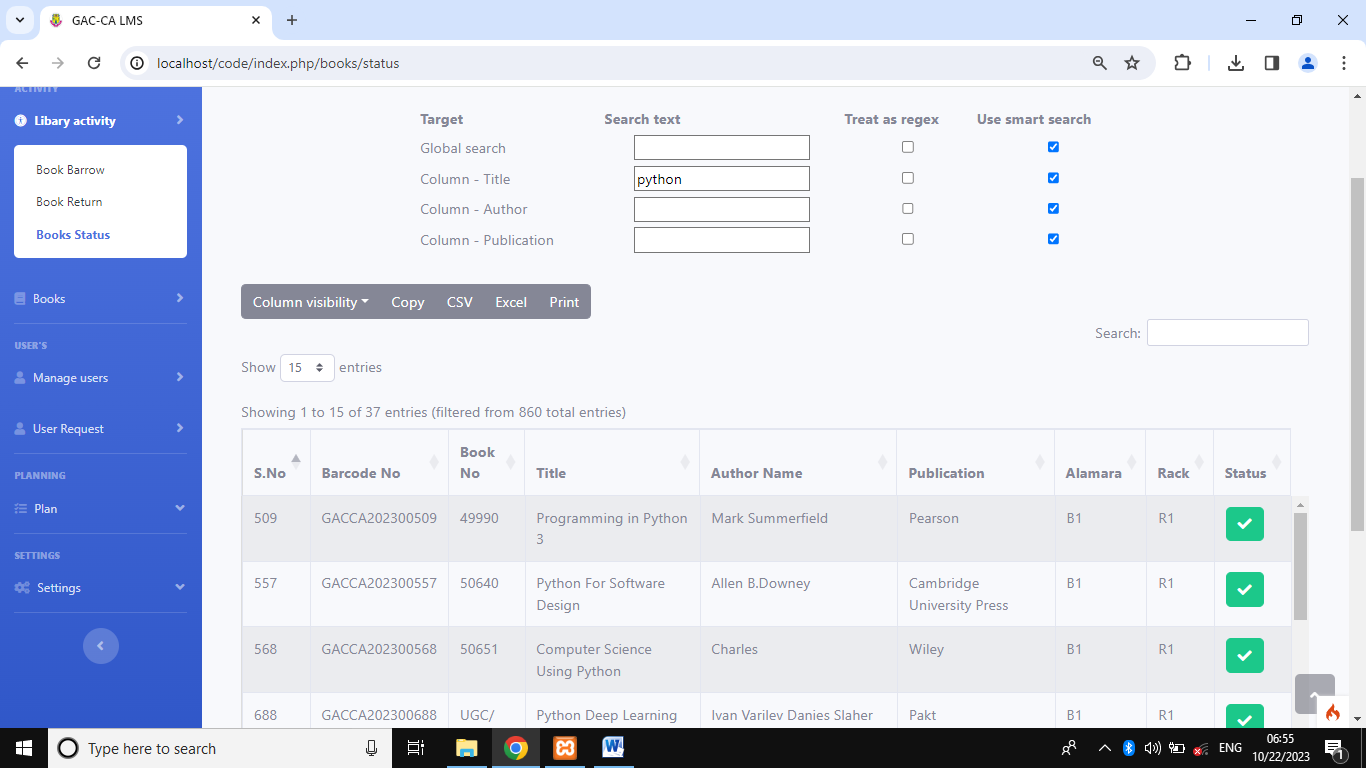
**Fig 3.4.1 (d) staff’s login form**

**UPDATE PROFILE:**

****

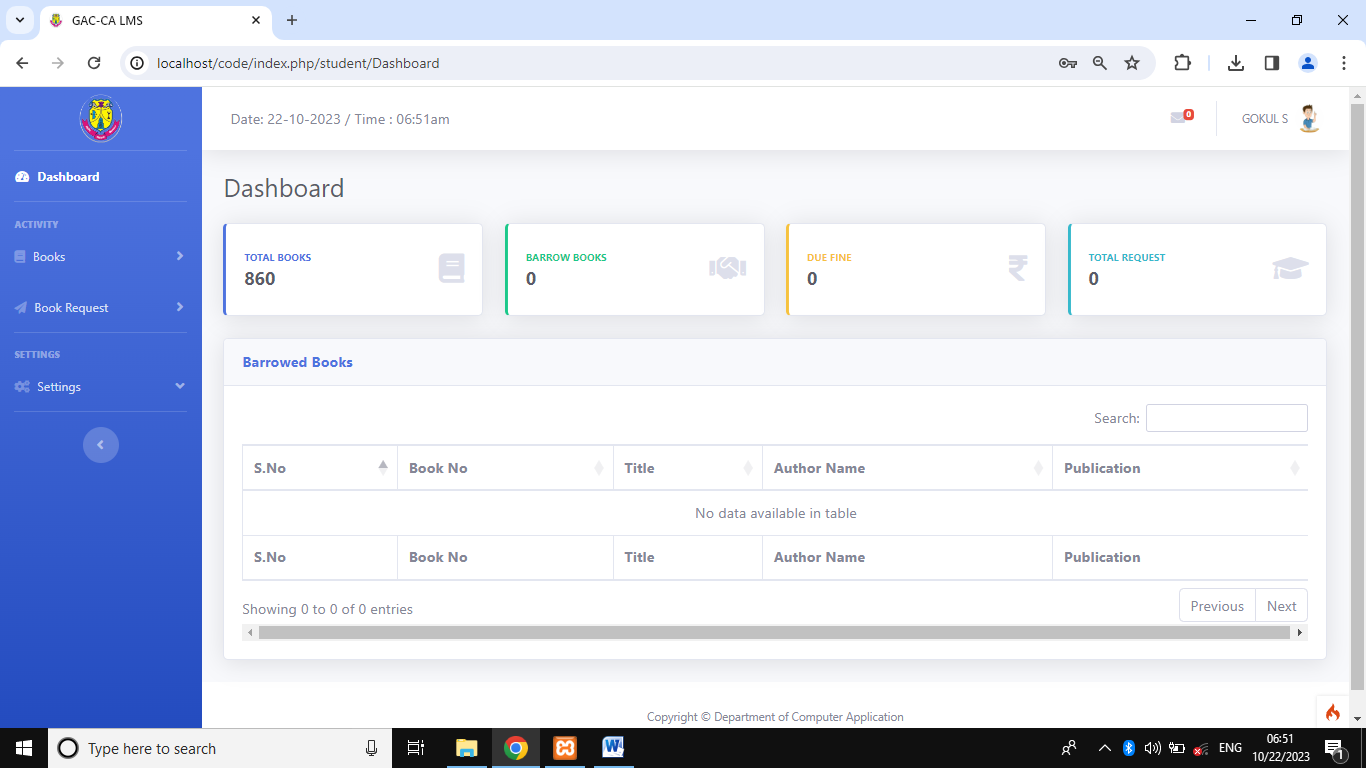
**Fig 3.4.2(e) staff’s update profile**

**SEARCH BOOK:**

****

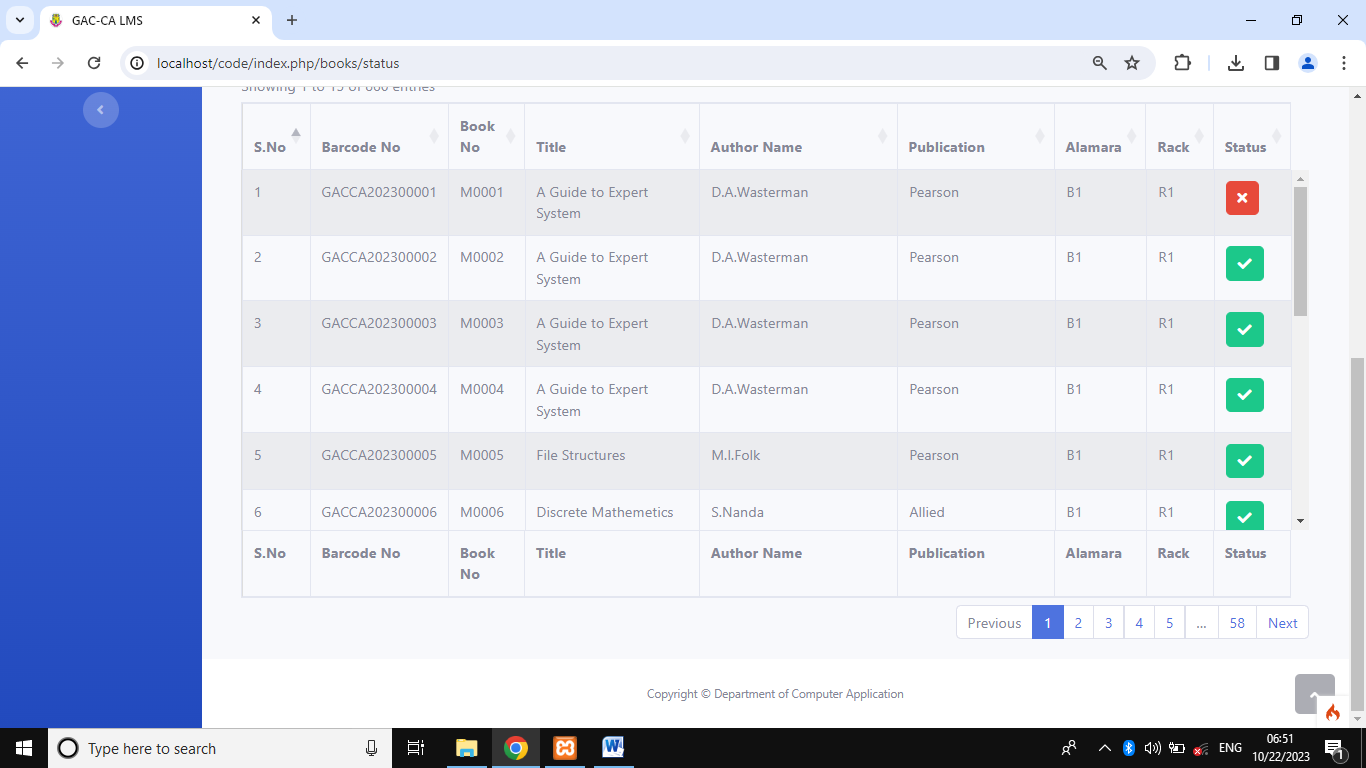
**Fig 3.4.3(f) search the books**

**STUDENT DASHBOARD:**

****

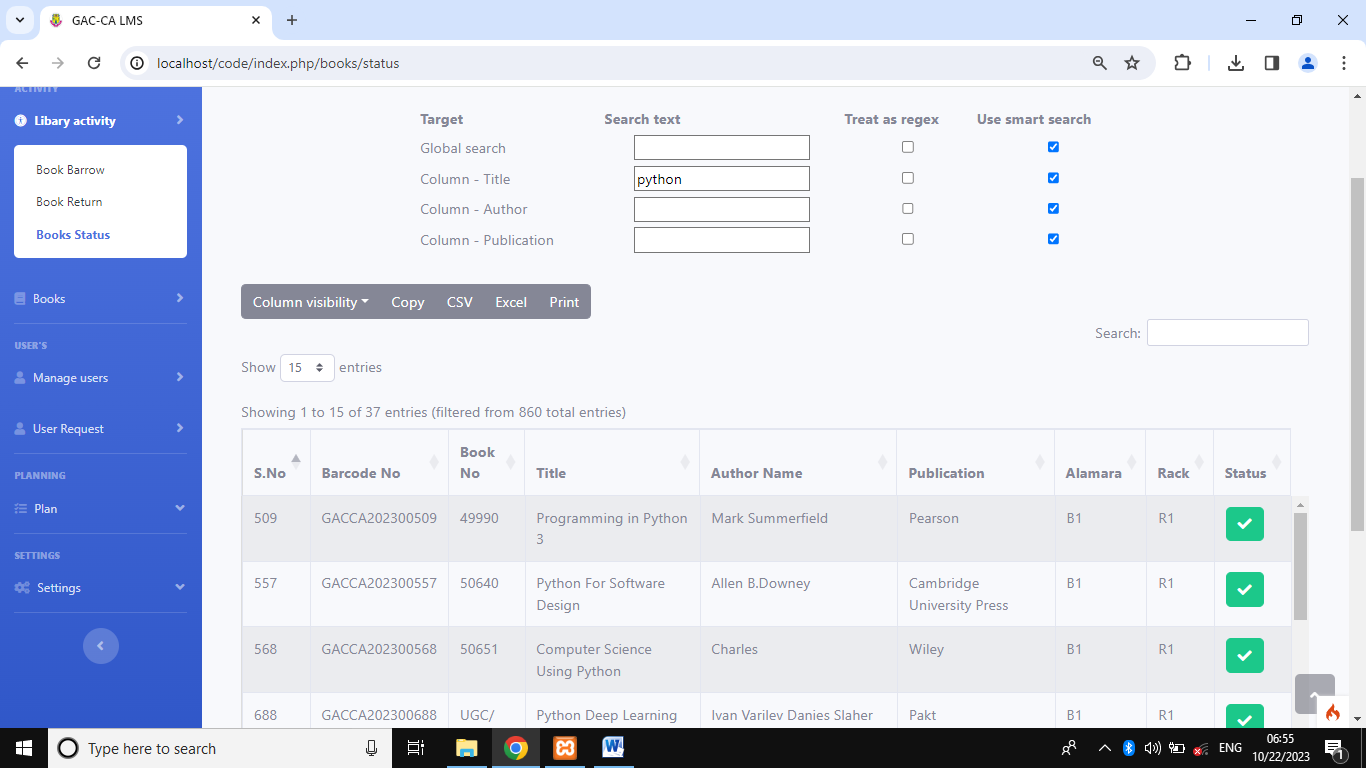
**Fig 3.5.1(a) student dashboard**

**BOOK STATUS:**

****

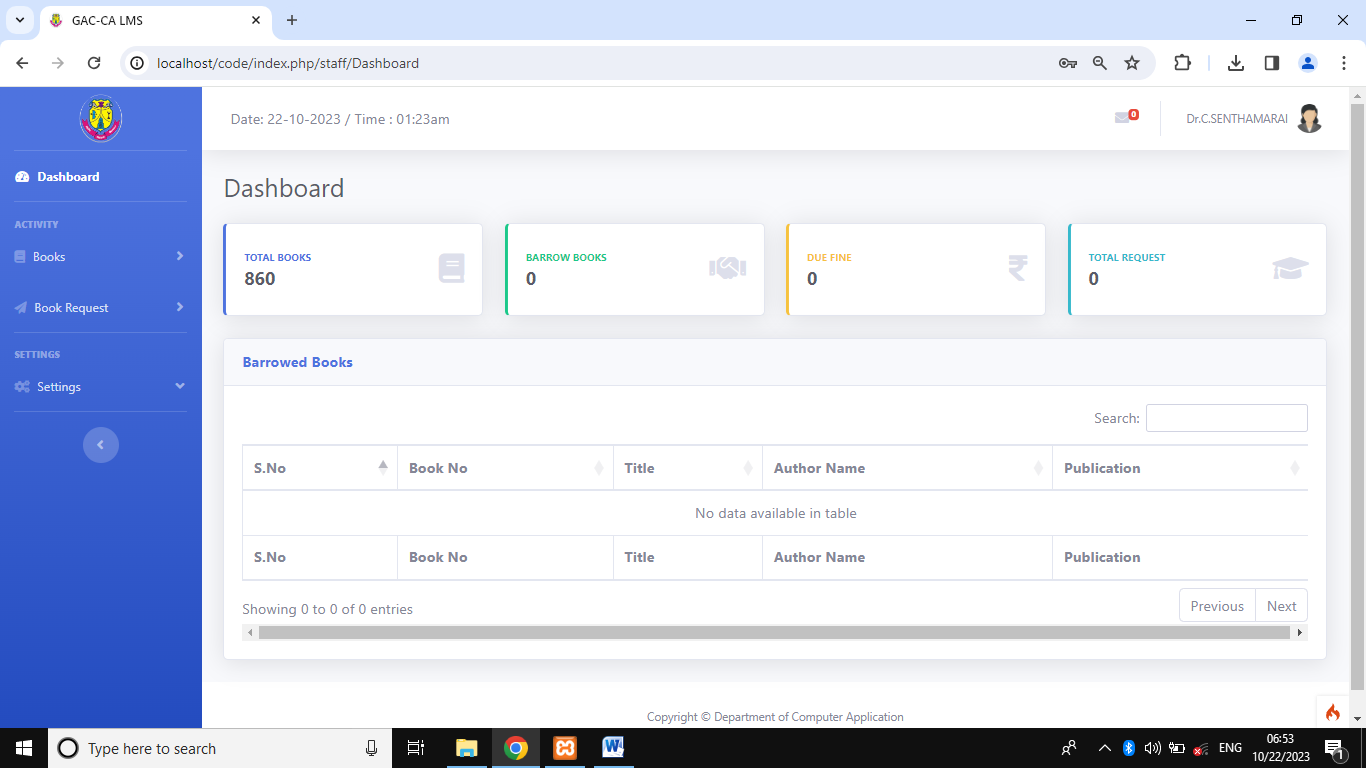
**Fig 3.5.2 (b) check the book status**

**SEARCH BOOK LIST:**

****

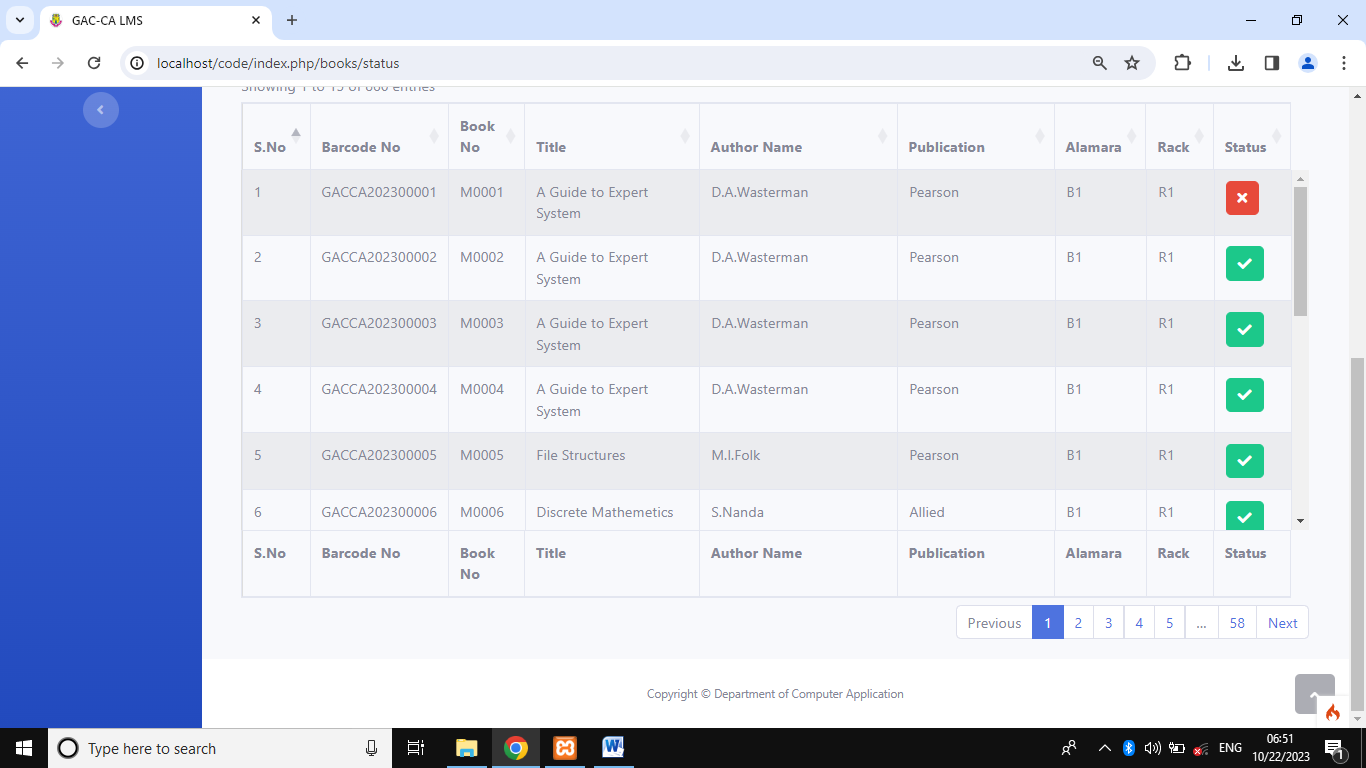
**Fig 3.5.3 (c) search book desired book results**

**STAFF DASHBOARD:**

****

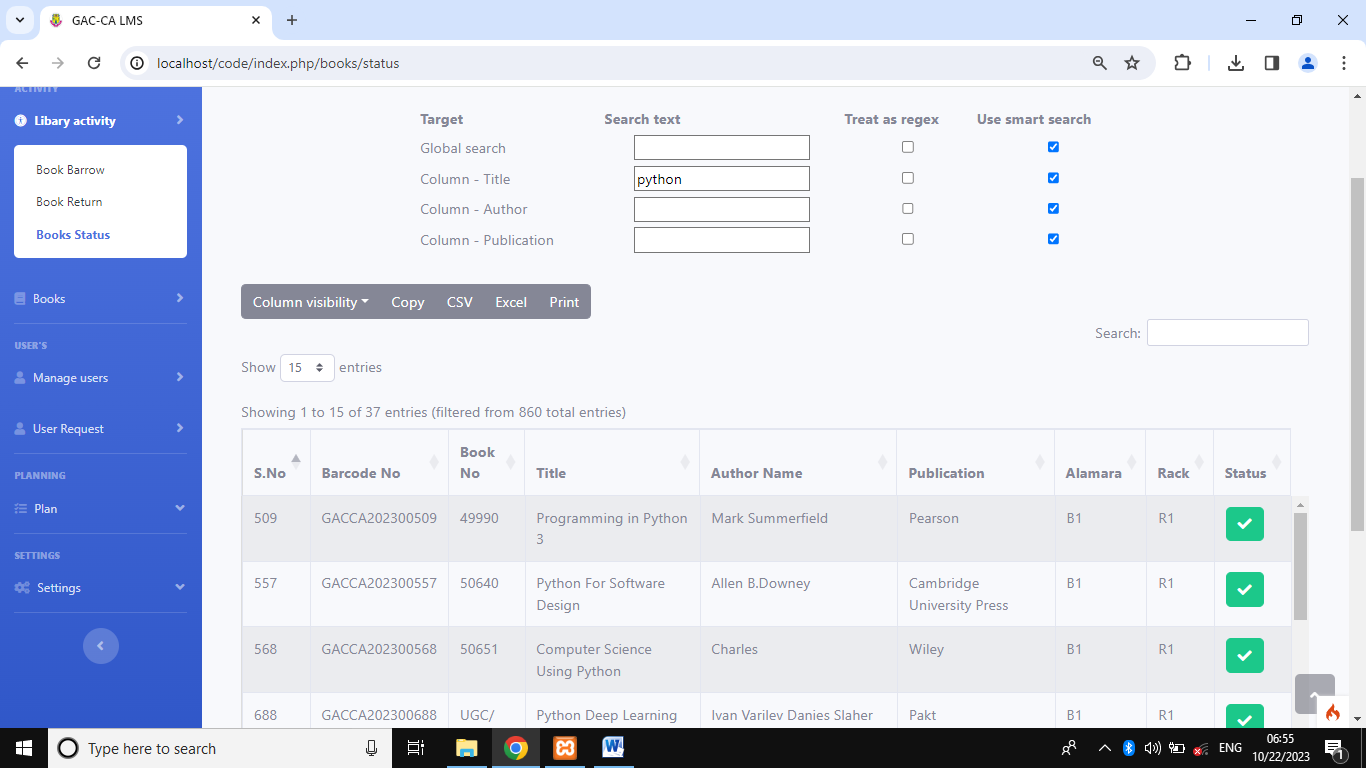
**Fig 3.5.1 (d) staff dashboard**

**BOOK STATUS:**

****

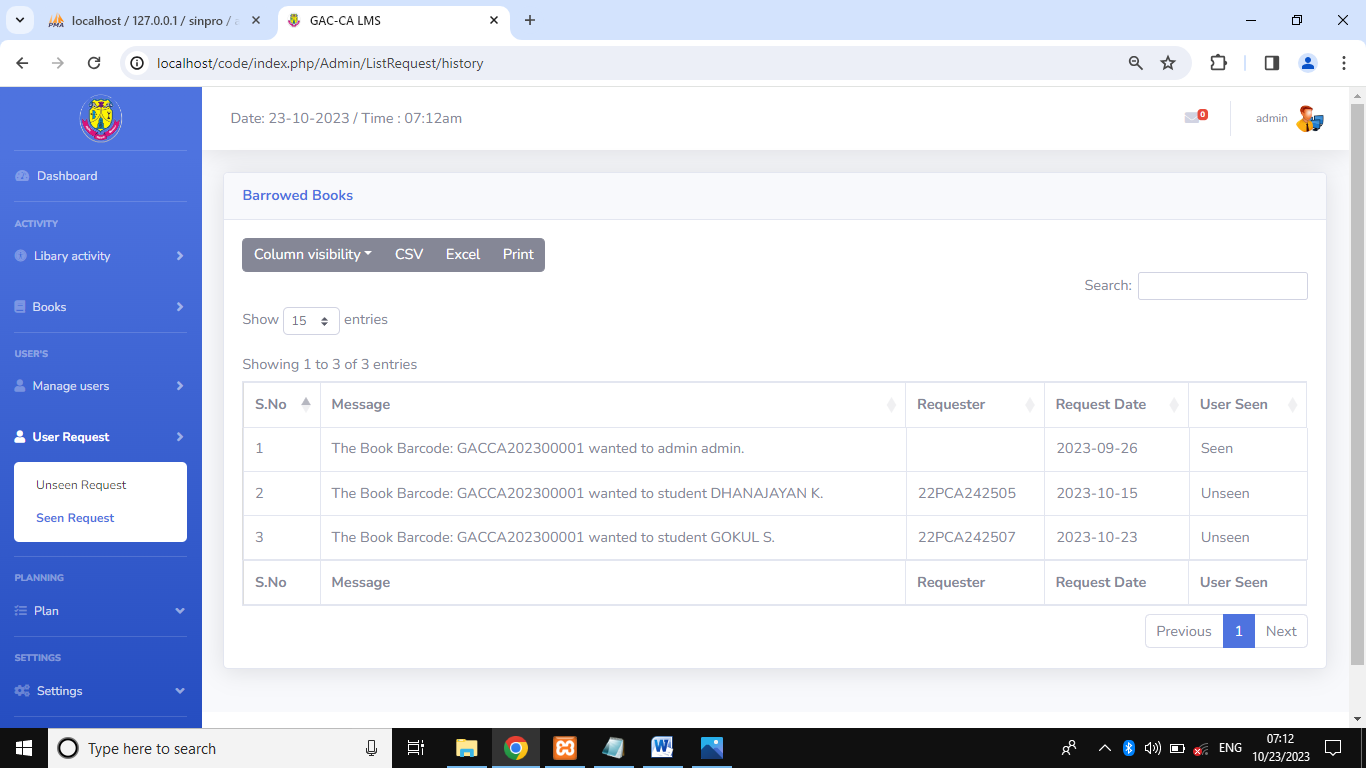
**Fig 3.5.2 (e) check the book status**

**SEARCH BOOKS:**

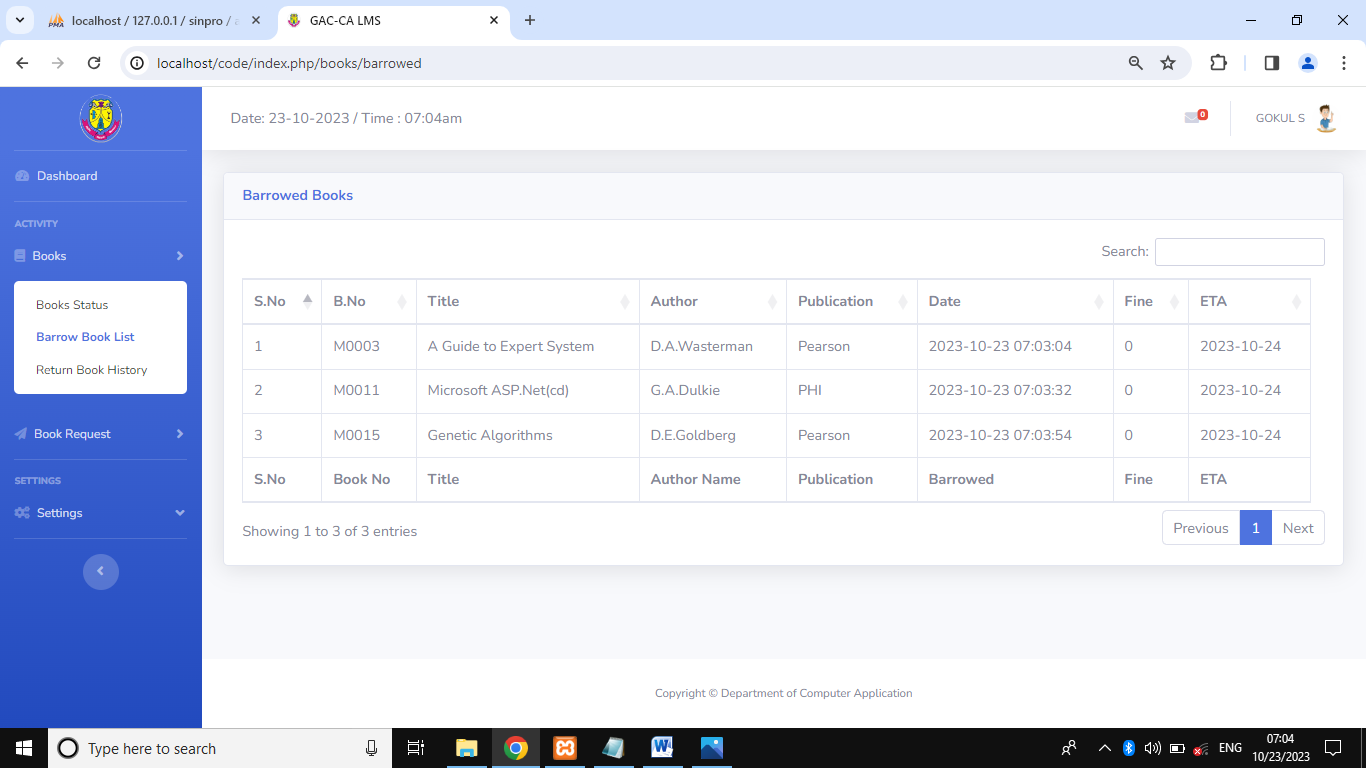
****

**Fig 3.5.3 (f) search the desired book results**

**REQUEST LIST:**

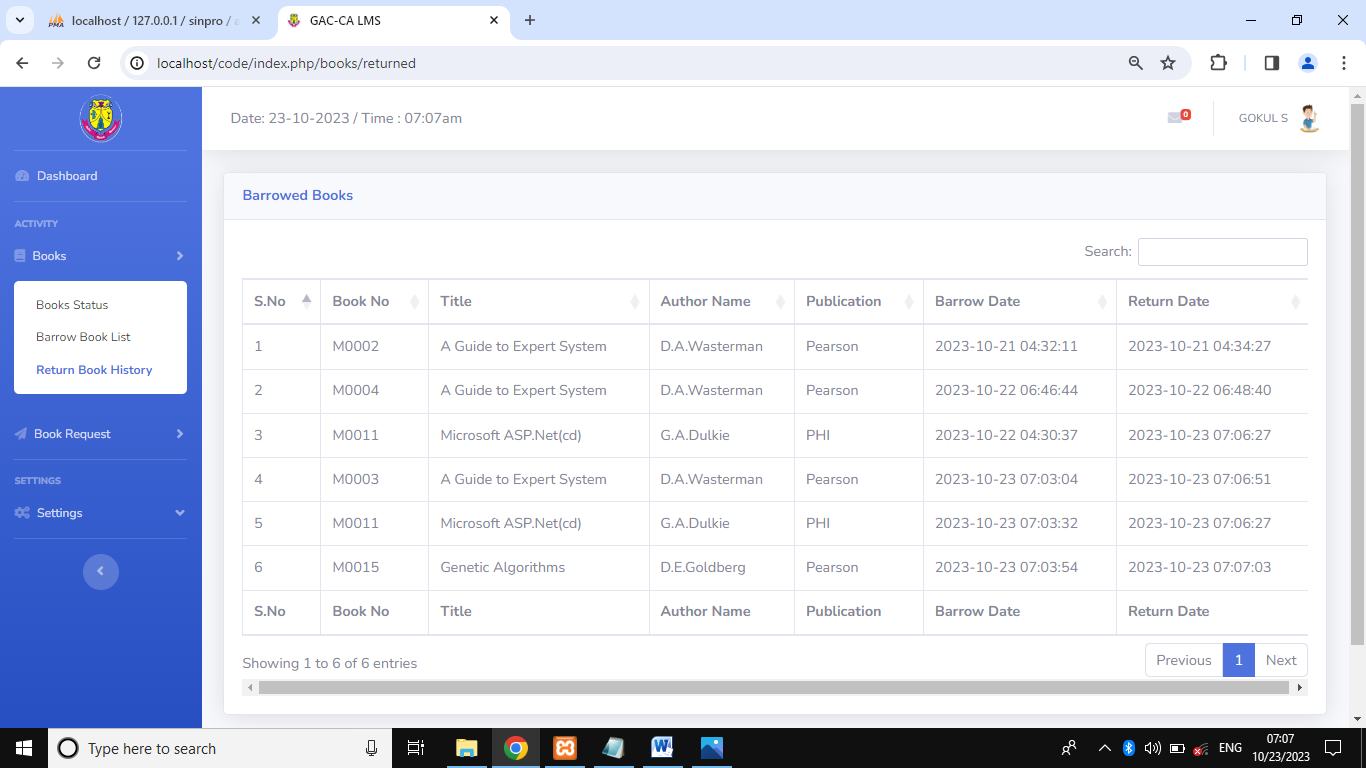
****

**Fig 3.5.4 (g) users can request a book**

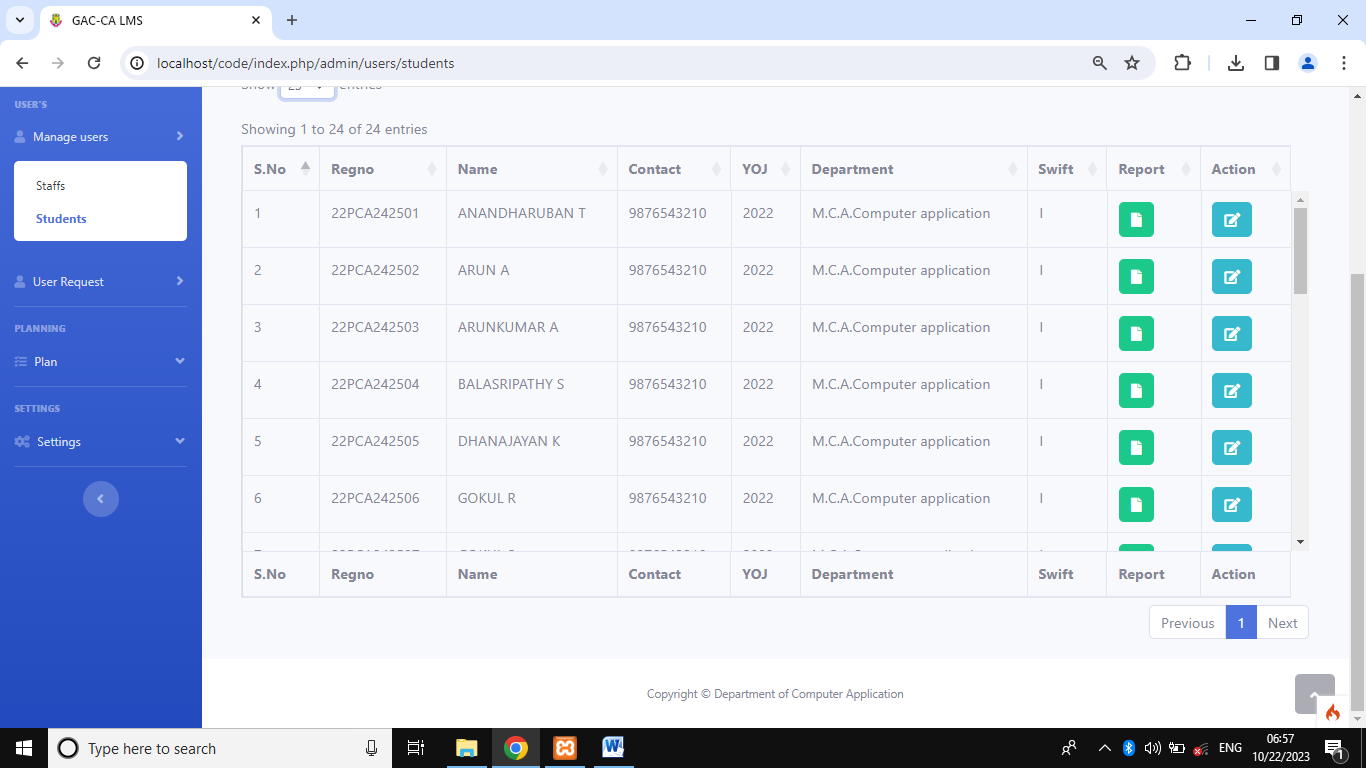
**BORROWED BOOK LIST: **

**Fig 3.5.5 (h) users borrowed book lists**

**RETURN BOOK LIST:**

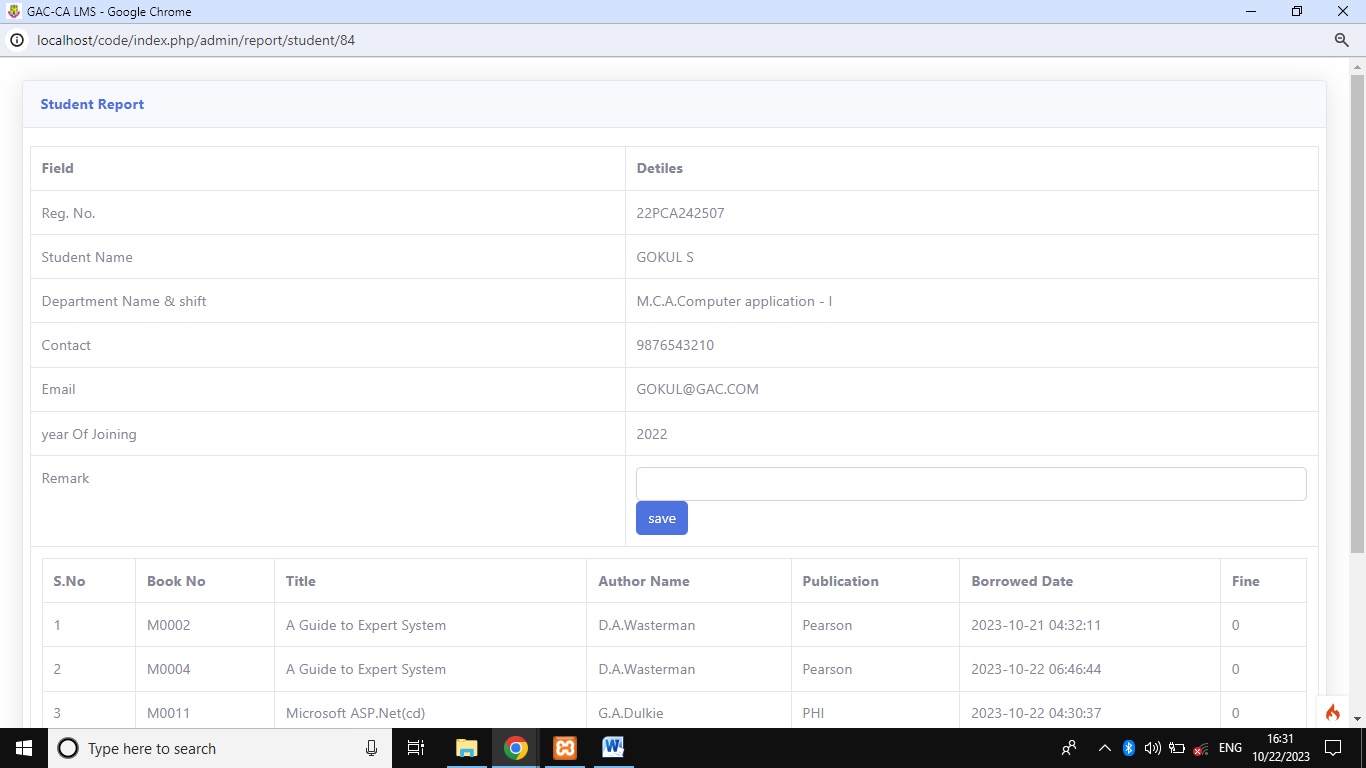
****

**Fig 3.6.1 users return book lists**

**REPORT LIST:**

**Fig 3.6.2 student’s final report lists**

**STUDENT REPORT:**

****

**Fig 3.6.3 student report**

**Sample Code**

**Login.php:**

<?php

namespace App\Controllers;

use App\Models\AdminModel;

use App\Models\StaffModel;

use App\Models\StudentModel;

use App\Models\OtherModel;

use App\Controllers\Mail;

class Login extends BaseController

{

protected $userModel;

protected $adminModel;

protected $studentModel;

public function \_\_construct()

{

// $this->preogin();

// $this->userModel = new UserModel();

// $this->adminModel = new AdminModel();

// $this->studentModel = new StudentModel();

}

public function index(){

$session = session();

switch($session->get("role")){

case "admin":

return redirect()->to('/admin/home');

case "staff":

return redirect()->to('/staff/Dashboard');

case "student":

return redirect()->to('/student/Dashboard');

default:

$session->destroy();

return redirect()->to('Login/student');

}

}

public function student(){

$session = session();

switch($session->get("role")){

case "admin":

return redirect()->to('/admin/home');

case "staff":

return redirect()->to('/staff/Dashboard');

case "student":

return redirect()->to('/student/Dashboard'); }

$validation = \Config\Services::validation();

$validation->setRules([

'username' => 'required',

'password' => 'required'

]);

if ($this->request->getMethod() === 'post') {

$this->studentModel = new StudentModel();

$regno = $this->request->getPost('regno');

$password = $this->request->getPost('password');

$data = $this->studentModel->where('regno', $regno)->first();

if ($data && password\_verify($password,$data['spass'])) {

// Successful login

$session->set([

'id' => $data['st\_id'],

'name' => $data['sname'],

'email' => $data['stemail'],

'role' => $data['role'],

'image' => $data['image'],

'isLoggedIn' => true

]);

return redirect()->to('/student/Dashboard');

}

// Invalid credentials

$session->setFlashdata('msg', 'Email or Password is incorrect.');

// return redirect()->to('/');

}

helper(['form']);

echo view('Others/header');

echo view('Login');

echo view('Others/fooder');

}

public function staff(){

$session = session();

switch($session->get("role")){

case "admin":

return redirect()->to('/admin/home');

case "staff":

return redirect()->to('/staff/Dashboard');

case "student":

return redirect()->to('/student/Dashboard'); }

$validation = \Config\Services::validation();

$validation->setRules([

'regno' => 'required',

'password' => 'required'

]);

if ($this->request->getMethod() === 'post') {

$this->userModel = new StaffModel();

$regno = $this->request->getPost('regno');

$password = $this->request->getPost('password');

$data = $this->userModel->where('regno', $regno)->first();

if ($data && password\_verify($password, $data['spass'])) {

// Successful login

$session->set([

'id' => $data['sid'],

'regno' => $data['regno'],

'name' => $data['sname'],

'email' => $data['semail'],

'role' => $data['role'],

'image' => $data['image'],

'isLoggedIn' => true

]);

return redirect()->to('/staff/Dashboard');

}

// Invalid credentials

$session->setFlashdata('msg', 'Email or Password is incorrect.');

// return redirect()->to('/staff');

}

helper(['form']);

echo view('Others/header');

echo view('Login');

echo view('Others/fooder');

}

public function Admin2Login()

{

$session = session();

switch($session->get("role")){

case "admin":

return redirect()->to('/admin/home');

case "staff":

return redirect()->to('/staff/Dashboard');

case "student":

return redirect()->to('/student/Dashboard');

}

$validation = \Config\Services::validation();

$validation->setRules([

'username' => 'required',

'password' => 'required'

]);

if ($this->request->getMethod() === 'post') {

$this->adminModel = new AdminModel();

$username = $this->request->getPost('username');

$password = $this->request->getPost('password');

$data = $this->adminModel->where('aname', $username)->first();

if ($data && password\_verify($password,$data['apass'])) {

// Successful login

$session->set([

'id' => $data['id'],

'name' => $data['aname'],

'email' => $data['a\_mail'],

'role' => $data['role'],

'isLoggedIn' => true

]);

return redirect()->to('/admin/home');

}

// Invalid credentials

$session->setFlashdata('msg', 'Email or Password is incorrect. ');

// return redirect()->to('Admin2Login');

}

helper(['form']);

echo view('Others/header');

echo view('Admin/Admin2login');

echo view('Others/fooder');

}

public function forgetpassword()

{

$session = session();

$this->mail = new Mail();

$this->otherModel = new OtherModel();

$this->studentModel = new StudentModel();

$this->userModel = new StaffModel();

if ($this->request->getMethod() === 'post') {

$regno = $this->request->getPost('regno');

$email = $this->request->getPost('email');

$data = $this->otherModel->forgetpassword($regno,$email);

if ($data) {

$sid = $data->sid;

$stemail = $data->semail;

$sname = $data->sname;

$role = $data->role;

try{

$characters = '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ';

$newpassword = '';

for ($i = 0; $i < 8; $i++) {

$index = rand(0, strlen($characters) - 1);

$newpassword .= $characters[$index];

}

$subject = '(TESTING) DLMS Password Reset request';

$body = str\_replace(

array('{cname}', '{cpassword}'),

array($sname, $newpassword),

file\_get\_contents(base\_url('assets/Template/resetpasswd.phtml')));

// mail trigger (calling send mail function)

if($this->mail->sendmail($stemail,$sname,$subject,$body)){

$data = [

'spass' => password\_hash($newpassword, PASSWORD\_DEFAULT),

];

if ($role == 'staff') {

$this->userModel->updateProfile($sid,$data);

} elseif ($role == 'student'){

$this->studentModel->updateProfile($sid,$data);

}

return redirect()->to('/');

}

} catch(Exception $e){

$session->setFlashdata('msg', 'Email is not being sent. Please contact a librarian for assistance');

}

} else{

$session->setFlashdata('msg', 'Incorrect Register Number or Email');

}

}

echo view('Others/header');

echo view('ForgetPassword');

echo view('Others/fooder');

}

}

**Dashboard.php:**

<?php

namespace App\Controllers\student;

use App\Controllers\BaseController;

use App\Models\BooksModel;

use App\Models\BarrowBooksModel;

use App\Models\RequestModel;

class Dashboard extends BaseController

{

public function \_\_construct()

{

$this->session = session();

$this->bookModel = new BooksModel();

$this->barrowBooksModel = new BarrowBooksModel();

$this->requestModel = new RequestModel();

}

public function index()

{

$session = session();

if ($session->get('role')!="student") {

return redirect()->to('/');

}

$data = [

'totalBooks' => $this->bookModel->getTotalBooks(),

'totalRequest' => count($this->requestModel->getBookRequest($session->get("id"),$session->get("role"))),

'totalFine' => $this->barrowBooksModel->getFineAmount($session->get("id"),$session->get("role")),

'books' => $this->barrowBooksModel->getBarrowedBookbyUser($session->get("id"),$session->get("role")),

];

echo view('Others/header');

echo view('Student/Dashboard',$data);

echo view('Others/fooder');

}

}

**profile.php:**

<?php

namespace App\Controllers\student;

use App\Controllers\BaseController;

use App\Models\StudentModel;

class Profile extends BaseController

{

public function \_\_construct()

{

$this->studentModel = new StudentModel();

}

public function index()

{

$session = session();

if ($session->get('role')!="student") {

return redirect()->to('/');

}

if ($this->request->getMethod() === 'post') {

$data = [

'profile' => $this->studentModel->getProfile($session->get('id')),

];

'contactno' => 'required',

'amail' => 'required | valid\_email'

]);

if (empty($apass)) {

$ahpass = $data['profile']['spass'];

}else{

$ahpass = password\_hash($apass, PASSWORD\_DEFAULT);

}

if (empty($contact)) {

$contact = $data['profile']['Contact'];

}

if (empty($amail)) {

$amail = $data['profile']['stemail'];

}

$data = [

'spass' => $ahpass,

'stemail' => $amail,

'Contact' => $contact

];

if($this->studentModel->updateProfile($id,$data)){

$session->setFlashdata('msg', 'Profile Update Successfully.');

} else{

$session->setFlashdata('msg', 'Profile Update Failed.');

}

}

$data = [

'profile' => $this->studentModel->getProfile($session->get('id')),

];

echo view('Others/header');

echo view('Student/StudentProfile',$data);

echo view('Others/fooder'); }}

**Book Status.php:**

<div class="container-fluid">

<h1 class="h3 mb-4 text-gray-800">Status Books</h1>

<!-- <ul class="nav nav-tabs" role="tablist" id="list-tab">

<li class="nav-item" role="presentation">

<a class="nav-link active" id="list-home-list" data-toggle="tab" href="#list-home" role="tab" aria-controls="home">All</a>

</li>

<li class="nav-item" role="presentation">

<a class="nav-link" id="list-profile-list" data-toggle="tab" href="#list-profile" role="tab" aria-controls="profile">Barrowed</a>

</li>

</ul> -->

<div class="card-body">

<div class="table-responsive">

<div class="tab-content" id="nav-tabContent">

<div class="tab-pane fade show active" id="list-home" role="tabpanel" aria-labelledby="list-home-list">

<table style="width: 67%; margin: 0 auto 2em auto;" cellspacing="0" cellpadding="3" border="0"> <thead> <tr>

<th>Target</th>

<th>Search text</th>

<th>Treat as regex</th>

<th>Use smart search</th>

</tr></thead><tbody>

<tr id="filter\_global">

<td>Global search</td>

<td align="center"><input type="text" class="global\_filter" id="global\_filter"></td>

<td align="center"><input type="checkbox" class="global\_filter" id="global\_regex"></td>

<td align="center"><input type="checkbox" class="global\_filter" id="global\_smart" checked="checked"></td></tr>

<tr id="filter\_col3" data-column="3">

<td>Column - Title</td>

<td align="center"><input type="text" class="column\_filter" id="col3\_filter"></td>

<td align="center"><input type="checkbox" class="column\_filter" id="col3\_regex"></td>

<td align="center"><input type="checkbox" class="column\_filter" id="col3\_smart" checked="checked"></td>

<tr id="filter\_col4" data-column="4">

<td>Column - Author</td>

<td align="center"><input type="text" class="column\_filter" id="col4\_filter"></td>

<td align="center"><input type="checkbox" class="column\_filter" id="col4\_regex"></td>

<td align="center"><input type="checkbox" class="column\_filter" id="col4\_smart" checked="checked"></td>

<tr id="filter\_col5" data-column="5">

<td>Column - Publication</td>

<td align="center"><input type="text" class="column\_filter" id="col5\_filter"></td>

<td align="center"><input type="checkbox" class="column\_filter" id="col5\_regex"></td>

<td align="center"><input type="checkbox" class="column\_filter" id="col5\_smart" checked="checked"></td>

</tbody>

<table class="table table-bordered table-striped" id="Status" data-show-columns="true" data-key-events="true" data-cookie="true" data-cookie-id-table="saveId" data-show-export="true">

<thead>

<tr>

<th>S.No</th>

<th>Barcode No</th>

<th>Book No</th>

<th>Title</th>

<th>Author Name</th>

<th>Publication</th>

<th>Alamara</th>

<th>Rack</th>

<th>Status</th>

</tr>

</thead>

<tfoot>

<tr>

<th>S.No</th>

<th>Barcode No</th>

<th>Book No</th>

<th>Title</th>

<th>Author Name</th>

<th>Publication</th>

<th>Alamara</th>

<th>Rack</th>

<th>Status</th>

</tr> </div>

<script type="text/javascript">

function request\_click(rowElement) {

var bcode = $(rowElement).closest('tr').find('.bcode').text();

$.ajax({

type: "POST",

url: "<?php echo site\_url(); ?>/books/bookrequest/",

data: {

bcode: bcode,

},

success: function(result) {

// alert('Request successful');

$('#popupSuccessModal').modal('show');

},

error: function(xhr, status, error) {

// var errorMessage = "An error occurred: " + status + " - " + error;

// if (xhr.responseText) {

// errorMessage += "\nAdditional details: " + xhr.responseText;

// }

// alert(errorMessage);

if (xhr.responseText) {

// Parse the server's JSON response

var errorResponse = JSON.parse(xhr.responseText);

// Check the message

if (errorResponse.message === "Already Requested") {

// Book already requested, show appropriate modal or handle the error

$('#popupAlreadyRequestedModal').modal('show');

} else {

// Some other error occurred, show a generic error message or handle the error

$('#popupFailedModal').modal('show');

}

} else {

// No valid JSON response received, show a generic error message or handle the error

$('#popupFailedModal').modal('show');

}

}

});

}

</script>

**Logout.php:**

<?php

namespace App\Controllers;

class Logout extends BaseController

{

public function index()

{

$session = Session();

$session->destroy();

// Get the session save path

$sessionSavePath = session\_save\_path();

// Get the maximum session lifetime (in seconds)

$maxLifetime = ini\_get('session.gc\_maxlifetime');

if (!empty($sessionSavePath) && is\_dir($sessionSavePath)) {

// Get the current timestamp

$currentTime = time();

// List session files in the save path

$sessionFiles = glob($sessionSavePath . '/ci\_\*');

foreach ($sessionFiles as $file) {

// Check if the session file is older than the maximum lifetime

if (filemtime($file) + $maxLifetime < $currentTime) {

unlink($file); // Delete the expired session file

}

}

}

return redirect()->to('/');

}

}