

***[Library Management System]***

***Version: [1.0]***

Group Members/Roll Numbers  
K17-3613 Sameer Khowaja  
K17-3808 Ali Rehman  
K17-2387 Kashif Hussain

Date of Submission  
December 20, 2020

Live Hosting URL  
<https://wp-lms.000webhostapp.com/index.php>

**Table of Contents**

[*1.* *Introduction 3*](about:blank)

[1.1. Purpose of Document](about:blank)

[1.2. Intended Audience](about:blank)

[1.3. Document Convention](about:blank)

[1.4. Project Background](about:blank)

[1.5. Project Scope](about:blank)

[1.6. Stakeholders](about:blank)

[1.7. Operating Environment](about:blank)

[1.8. System Constraints](about:blank)

[*2.* *External Interface Requirements 5*](about:blank)

[2.1. Hardware Interfaces](about:blank)

[2.2. Software Interfaces](about:blank)

[*3.* *Functional Requirements 6*](about:blank)

[3.1. Functional Hierarchy](about:blank)

[3.2. Use Cases Diagram](about:blank)

[*4.* *Non-functional Requirements 8*](about:blank)

[4.1. Performance Requirements -](about:blank)

[4.2. Safety Requirements -](about:blank)

[4.3. Security Requirements -](about:blank)

[*5.* *Feasibility Analysis 9*](about:blank)

[5.1. Economic Feasibility](about:blank)

[5.2. Operational Feasibility](about:blank)

[5.3. Technological Feasibility](about:blank)

[5.4. Legal Feasibility](about:blank)

[*6.* *System Architecture 11*](about:blank)

[6.1. System Level Architecture](about:blank)

[6.2. Software Architecture](about:blank)

[*7.* *Detailed System Design 12*](about:blank)

[7.1. ER Diagram](about:blank)

[7.2. Data Dictionary](about:blank)

[7.2.1. user](about:blank)

[7.2.2. member](about:blank)

[7.2.3. category](about:blank)

[7.2.4. borrowdetail](about:blank)

[7.2.5. borrow](about:blank)

[7.2.6. book](about:blank)

[7.3. Sequence Diagram](about:blank)

[7.4. State Diagram](about:blank)

[7.5. Deployment Diagram](about:blank)

[7.6. Activity Diagram](about:blank)

[7.7. Class Diagram](about:blank)

[*8.* *Project Details 3*](about:blank)

[8.1. Features List](about:blank)

[8.2. Future Work](about:blank)

[8.3. Gantt Chart Diagram](about:blank)

[8.4. Screenshot of Project](about:blank)

[*9.* *Project Testing 3*](about:blank)

[9.1. Unit Testing](about:blank)

[9.2. Integration Testing](about:blank)

[9.3. Load Testing](about:blank)

[9.4. Stress Testing](about:blank)

[*10.* *References 3*](about:blank)

1. ***Introduction***
   1. ***Purpose of Document***

The purpose of this document is to describe how the Library Management System would work. This document will describe all the functionalities of the system and how users will be able to fulfill their goals. It will also describe the requirements of the system.

* 1. ***Intended Audience***

*The intended audience of this document is Web Programming Course Teacher.*

* 1. ***Document Convention***

*Heading Font Size: 14 with Bold Arial*

*Subheading Font Size: 12 with Bold Arial*

*Paragraph Font Size: 12 Calibri (Body)*

* 1. ***Project Background***

Library Management System is an application which refers to library systems which are generally small or medium in size. It is used by librarians to manage the library using a computerized system where he/she can record various transactions like issue of books, return of books, addition of new books, addition of new students etc. In addition, the print module is also included in the Library Management System. The admin is able to print different kinds of reports like lists of students registered, list of books, issue and return reports.

* 1. ***Project Scope***

In addition, the report module is also included in the Library Management System. If the user's position is admin, the user is able to generate different kinds of reports like lists of students registered, list of books, issue and return reports. Admin can record various transactions like issue of books, return of books, addition of new books, addition of new students and print past transaction records etc.

* 1. ***Stakeholders***

Stakeholders include Librarian as admin and student/faculty as members.

* 1. ***Operating Environment***

*System: Client/Server system*   
*Operating System: Windows, Linux etc.*  
*Database: SQL*   
*Browser: Mozilla Firefox, Opera, and Google Chrome etc.*

* 1. ***System Constraints***

1. **Software constraints**

No additional software needs to be installed for the library system. Only internet connectivity is required.

1. **Hardware constraints**

All end systems with basic specifications should be able to use the library system.

1. **Cultural constraints**

The language of the health care system will be Basic English.

1. **Legal constraints**

The books and members' information needs to be accurate and valid.

1. **User constraints**

Library management system will be user friendly as users will be patient.

1. ***External Interface Requirements***
   1. ***Hardware Interfaces***

The Library Management System is web based so it does not have any direct hardware interface. Following are the hardware requirements: Any end system with basic specifications. Browsers with web connectivity are required.

* 1. ***Software Interfaces***

User system will require a browser that supports JavaScript. Server side will require following software: MYSQL, APACHE and PHP.

1. ***Functional Requirements***
   1. ***Functional Hierarchy***

Functions are divided into (Librarian) admin, (Student/Faculty) members.

* + 1. ***Admin***

*3.3.1.1. Login with credentials*

*3.3.1.2. Add new books to stock*

*3.3.1.3. Add new category for books*

*3.3.1.4. Update/Delete books information*

*3.3.1.5. Add new members*

*3.3.1.6. Update/Delete member’s information*

*3.3.1.7. Search records*

*3.1.1.8. Print records*

*3.1.1.9. Borrow books to member request*

*3.1.1.10. Add book when it is returned*

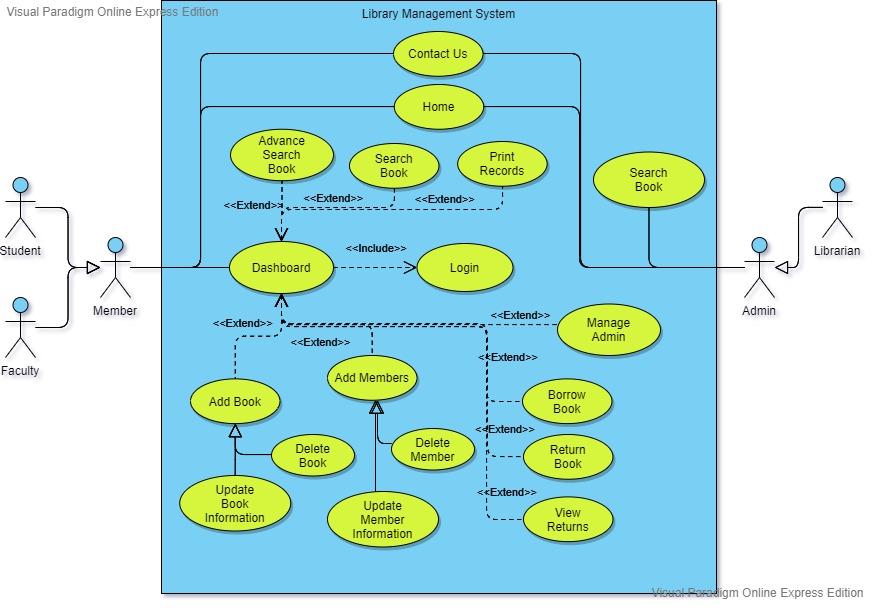
*3.1.1.11. Advance Search for books*

* + 1. ***Members***

*3.1.2.1. View available books*

*3.1.2.2. Search for books*

* 1. ***Use case Diagram***

******

1. ***Non-Functional Requirements***
   1. ***Performance Requirements***

The project will be web based and shall run from web server. So the loading time of a website will depend on the user 's internet connection’s strength and the end system. Hence Performance will depend upon hardware specification of the user's end system.

* 1. ***Safety Requirements***

A backup of all records will be made in mysql server so that in case of system crash data loss can be prevented.

* 1. ***Security Requirements***

User details kept secure using secure sockets. Proper login systems will be implemented and security would be considered.

1. ***Feasibility Analysis***
   1. ***Economic Feasibility***

The development expenditure of minimal viable products of our system is nothing much. Since we are developing a library system the cost of storing the details regarding the library will be incurred. So we will acquire monthly services of Oracle MYSQL database which will cost $0.04(Rupees 6.63) per GB. For the website domain and SSL certificate will incur a cost of around Rupees 2500.

Cost of features included in HMS is stated below:

1. **Wireframing and Requirement Elicitation**

Cost: 10,000

1. **Project Configuration**

Cost: 5000

1. **Development**

Cost: 50,000

1. **User authentication**

Cost = 7000

1. **Member Panel**

Cost = 10,000

1. **Book Panel**

Cost = 10,000

1. **Advance searches**

Cost = 10,000

1. **Member searches**

Cost = 8000

1. **Borrow Details panel**

Cost = 5000

* 1. ***Operational Feasibility***

It is mainly related to the human organizational and political aspect. The points to be considered are:

 • What changes will be brought with the system?

• What organizational structures are distributed?

• What new skills will be required?

The proposed system is feasible because of the following reasons: The system reduces the workload of the staff because on a mouse click he / she has the desired result, work can be done with the help of keyboard and mouse watching the computer screen not on the paper. The system will be built on the technology of GUI so that interaction to the system will not be boring as like writing / preparing / maintaining data into the form of the manual paper. Users that work into the GUI environment work more interestingly than the paper based. This result works more efficiently. The proposed system is better in use and user friendly as it generates proper messages at run time. The input from the user is much as the fields like supplier code. Customer code and current data are included by the system.

* 1. ***Technological Feasibility***

There are several Library systems present which implies that development of this system is technically feasible. We will be using PHP which is a free open source for the development of web applications. PHP contains Built-in support for user-authentication and it also keeps web applications secure by protecting it against cross-site request forgery, SQL injection and cross-site scripting. We will use SQL infrastructure to support server-side features and maintain our data. SQL has a better potential to control several databases within the same transaction using a two-phase commit protocol and it also provides support for online backup and recovery and excellent application fault tolerance to disk failure.

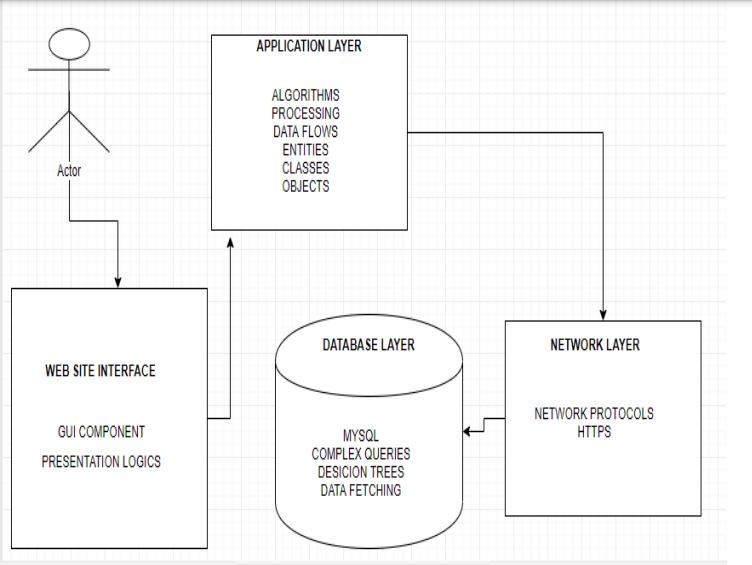
* 1. ***Legal Feasibility***

Our Library system is much similar to the existing legal LMS. We will keep records of members, books, admin and other records. These records will not be given to any other organization and the records will not be used for any purpose which can trouble our users.

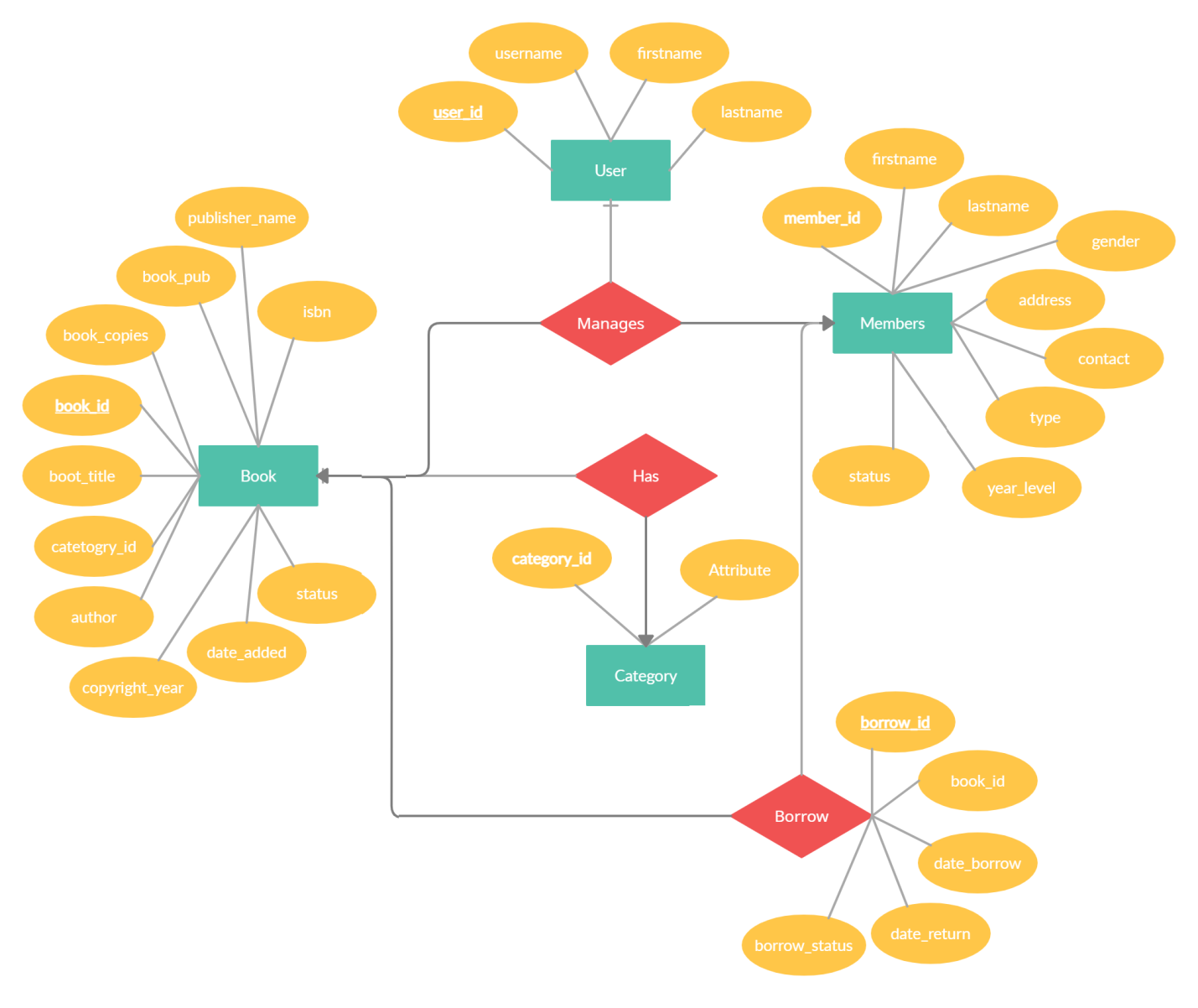
1. ***System Architecture***
   1. ***System Level Architecture***

******

* 1. ***Software Architecture***

******

1. ***Detailed System Design***
   1. ***ER Diagram***

******

* 1. ***Data Dictionary***
     1. ***User***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **User** | | | | | | | |
| **Name** | | | user | | | | |
| **Alias** | | |  | | | | |
| **Where-used/how-used** | | | Contain Admin login information. | | | | |
| **Column Name** | **Description** | **Type** | | **Length** | **Null able** | **Default Value** | **Key Type** |
| *User\_id* | *Id number* | *Integer* | | *11* | *No* | *AI* | *PK* |
| *Username* | *Name for login* | *String* | | *100* | *No* |  |  |
| *Password* | *Password for login* | *String* | | *100* | *No* |  |  |
| *Firstname* | *Admin first name* | *String* | | *100* | *No* |  |  |
| *Lastname* | *Admin last name* | *String* | | *100* | *No* |  |  |

* + 1. ***Member***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Member** | | | | | | | |
| **Name** | | | member | | | | |
| **Alias** | | |  | | | | |
| **Where-used/how-used** | | | Contain member (student/faculty) information. | | | | |
| **Column Name** | **Description** | **Type** | | **Length** | **Null able** | **Default Value** | **Key Type** |
| *Member\_id* | *Id number* | *Integer* | | *11* | *No* | *AI* | *PK* |
| *Firstname* | *Name for lmember* | *String* | | *100* | *No* |  |  |
| *Lastname* | *Surname for member* | *String* | | *100* | *No* |  |  |
| *Gender* | *gender* | *String* | | *10* | *No* |  |  |
| *address* | *Place of living* | *String* | | *100* | *No* |  |  |
| *Contact* | *Phone number* | *String* | | *100* | *No* |  |  |
| *Type* | *Teacher or Student* | *String* | | *100* | *No* |  |  |
| *Year\_level* | *Faculty or year i.e. 1st, 2nd etc.* | *String* | | *100* | *No* |  |  |
| *Status* | *Banned or active* | *String* | | *100* | *No* |  |  |

* + 1. ***Category***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Category** | | | | | | | |
| **Name** | | | category | | | | |
| **Alias** | | |  | | | | |
| **Where-used/how-used** | | | Contain book category list. | | | | |
| **Column Name** | **Description** | **Type** | | **Length** | **Null able** | **Default Value** | **Key Type** |
| *Category\_id* | *Id number* | *Integer* | | *11* | *No* | *AI* | *PK* |
| *Classname* | *Name for categories* | *String* | | *50* | *No* |  |  |

* + 1. ***Borrow Detail***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Borrow Details** | | | | | | | |
| **Name** | | | borrowdetails | | | | |
| **Alias** | | |  | | | | |
| **Where-used/how-used** | | | Contain information of books and member that borrowed books. | | | | |
| **Column Name** | **Description** | **Type** | | **Length** | **Null able** | **Default Value** | **Key Type** |
| *borrow\_details\_id* | *Id number* | *Integer* | | *11* | *No* | *AI* | *PK* |
| *Book\_id* | *Id of book* | *Integer* | | *11* | *No* |  |  |
| *Borrow\_id* | *Borrower member id* | *Integer* | | *11* | *No* |  |  |
| *Borrow\_status* | *Returned or pending* | *String* | | *50* | *No* |  |  |
| *Date\_return* | *Date of return* | *String* | | *50* | *No* |  |  |

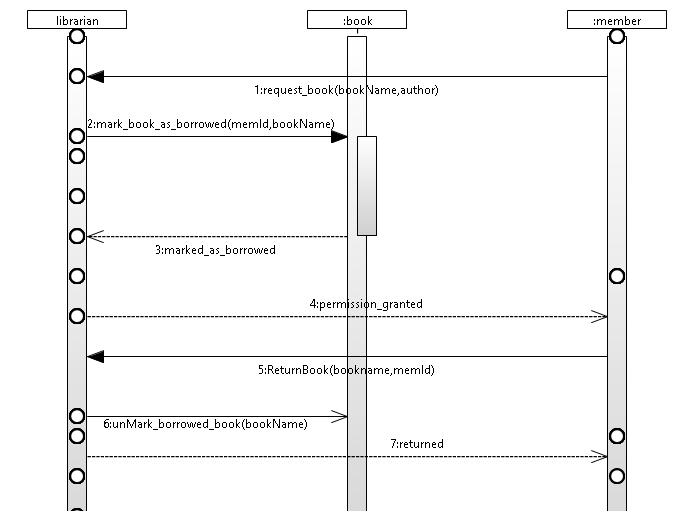
* + 1. ***Borrow***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Borrow** | | | | | | | |
| **Name** | | | borrow | | | | |
| **Alias** | | |  | | | | |
| **Where-used/how-used** | | | Contain information of books that borrowed. | | | | |
| **Column Name** | **Description** | **Type** | | **Length** | **Null able** | **Default Value** | **Key Type** |
| *borrow\_id* | *Borrower id* | *Integer* | | *11* | *No* | *AI* | *PK* |
| *Member\_id* | *Id of member* | *Integer* | | *50* | *No* |  |  |
| *Date\_borrow* | *Date which book borrow* | *String* | | *100* | *No* |  |  |
| *Due\_date* | *Return date of book* | *String* | | *100* | *Yes* |  |  |

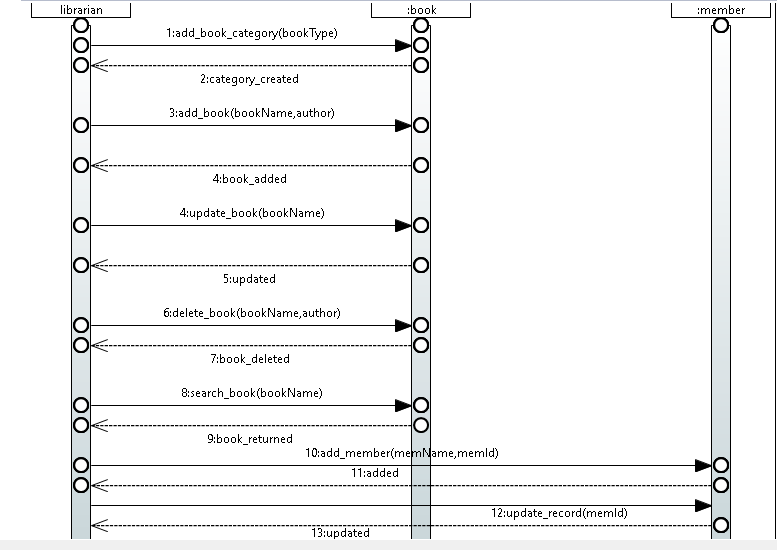
* + 1. ***Book***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Book** | | | | | | | |
| **Name** | | | book | | | | |
| **Alias** | | |  | | | | |
| **Where-used/how-used** | | | Contain information of books. | | | | |
| **Column Name** | **Description** | **Type** | | **Length** | **Null able** | **Default Value** | **Key Type** |
| *book\_id* | *Id of book* | *Integer* | | *11* | *No* | *AI* | *PK* |
| *Book\_title* | *Name of book* | *String* | | *100* | *No* |  |  |
| *Category\_id* | *Book category id* | *Integer* | | *50* | *No* |  |  |
| *Author* | *Author name* | *String* | | *50* | *No* |  |  |
| *Book\_copies* | *Copies number* | *Integer* | | *11* | *No* |  |  |
| *Book\_pub* | *Publication authority name* | *String* | | *100* | *No* |  |  |
| *Publisher\_name* | *Publisher name* | *String* | | *100* | *No* |  |  |
| *isbn* | *Isbn number* | *String* | | *50* | *No* |  |  |
| *Copyright\_year* | *Year of copyright* | *Integer* | | *11* | *No* |  |  |
| *Date\_recieve* | *Added to stock date* | *String* | | *20* | *No* |  |  |
| *Date\_added* | *Added to stock date* | *Datetime* | |  | *No* |  |  |
| *status* | *New, old or damaged* | *String* | | *30* | *No* |  |  |

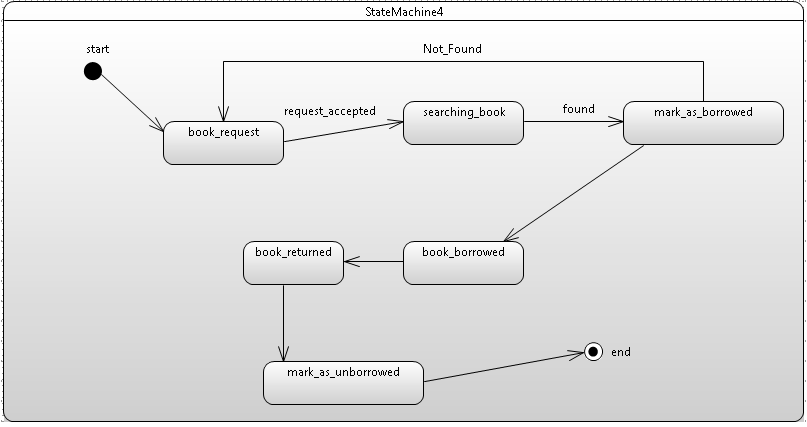
* 1. ***Sequence Diagram***
     1. ***Sequence diagram 1***

******

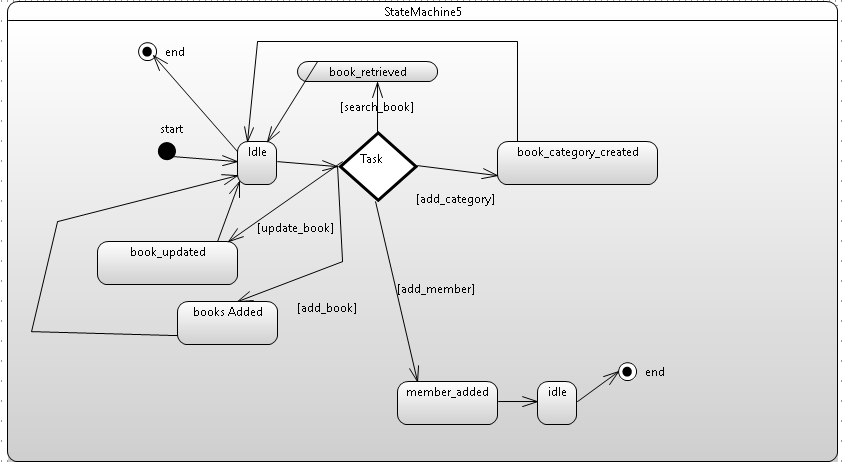
* + 1. ***Sequence diagram 2***

******

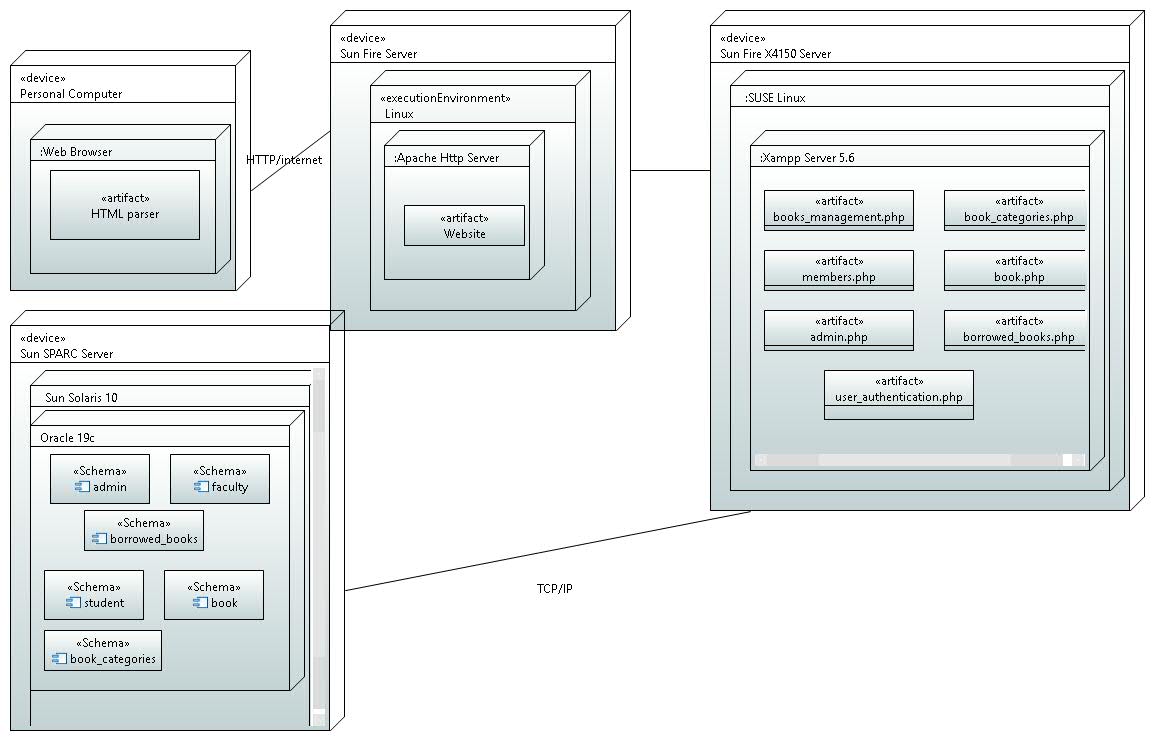
* 1. ***State Diagram***
     1. ***State Diagram 1***

******

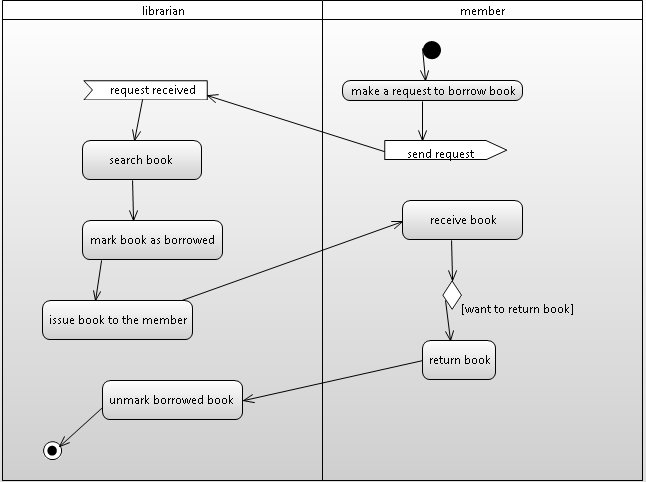
* + 1. ***State Diagram 2***

******

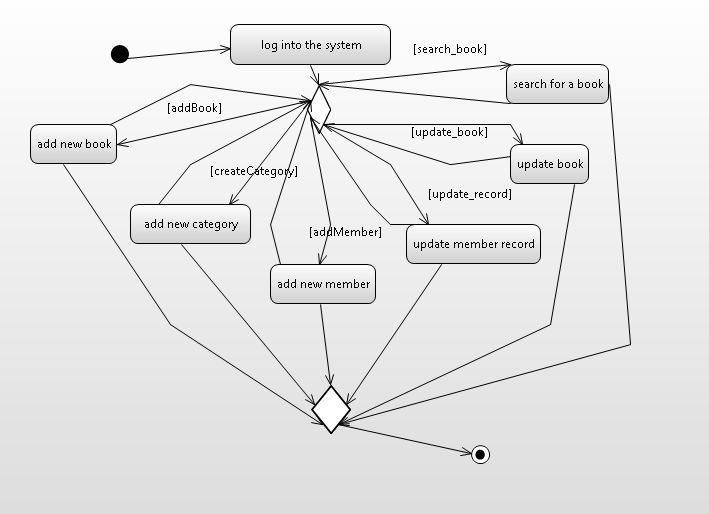
* 1. ***Deployment Diagram***

******

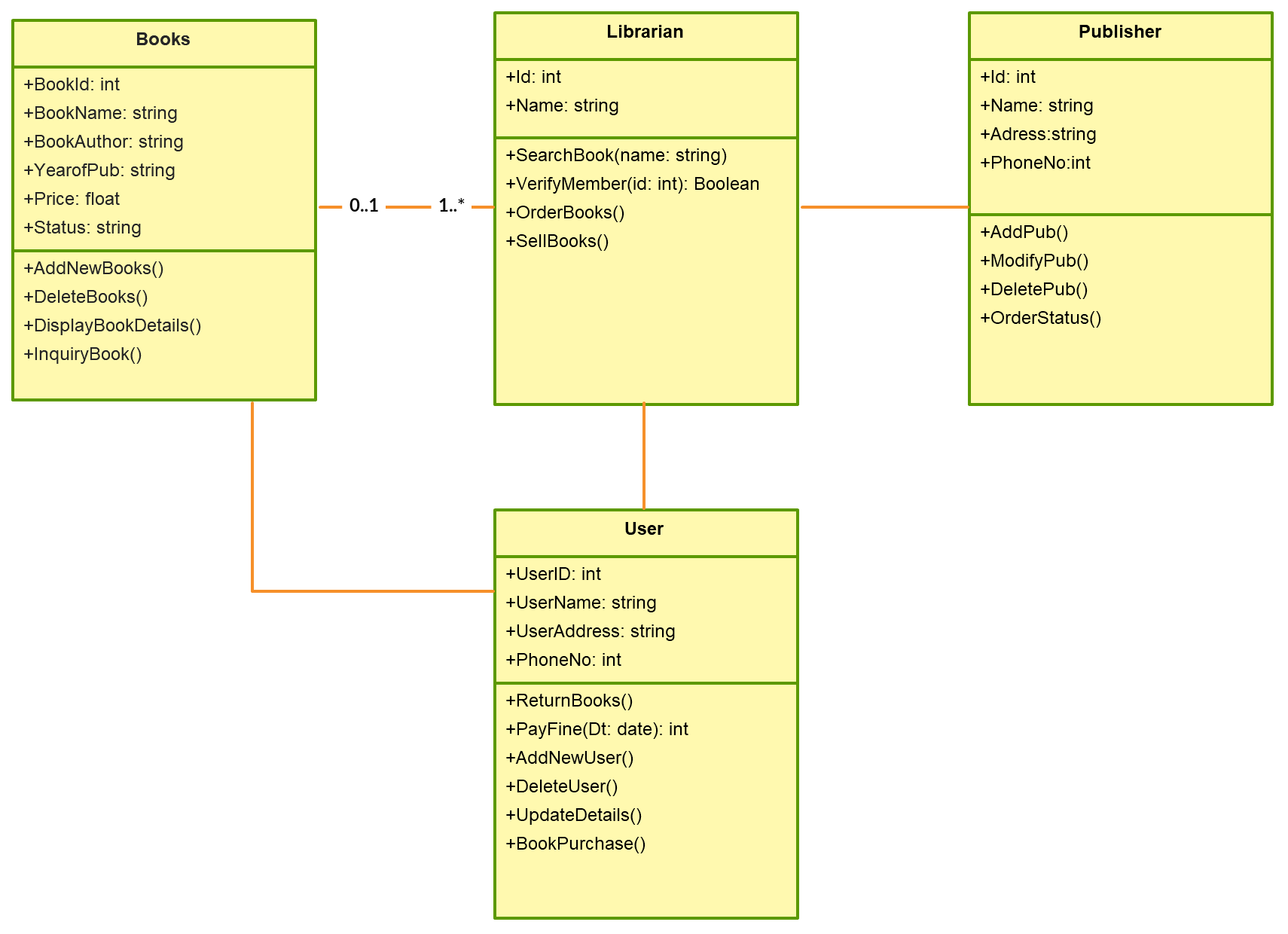
* 1. ***Activity Diagram***
     1. ***Activity Diagram 1***

******

* + 1. ***Activity Diagram 2***

******

* 1. ***Class Diagram***



1. ***Project Details***
   1. ***Features List***

Library Management System Features include:

* + 1. Manages library through the easy software interface.
    2. It removes the manual process of issuing/returning books which save time and effort.
    3. Admin manages books i.e. add new books, update or delete books.
    4. Librarian manages members i.e. add new members, update or delete members.
    5. Librarians can ban or activate members' membership.
    6. Librarians can search for records i.e. books, members etc.
    7. Librarians can advance search for books.
    8. Members can also check the availability of particular books.
    9. Members can borrow books from librarians.
    10. Members can return books to the librarian.
    11. Book purchase date and return date is viewed by the librarian.
    12. Librarians can print records if he/she wants.
    13. Category of books can be managed by Librarian.
    14. Books status can be managed by Librarian i.e., new, old or damaged.
    15. Keep records of complete info of books, members, and librarians.
    16. Login authentication required for Librarian.
    17. User friendly Software.
    18. Images and Slide presentation and nice UI for member attraction.
    19. It is cost effective.
    20. Fast, secure and easy way of transactions.
  1. ***Future Work***

There is a future scope of this facility that many more features such as online lectures video tutorials can be added by teachers as well as online assignments submission facility, a feature Of group chat where students can discuss various issues of engineering can be added to this project thus making it more interactive more user friendly and project which fulfills each users need in the best way possible.

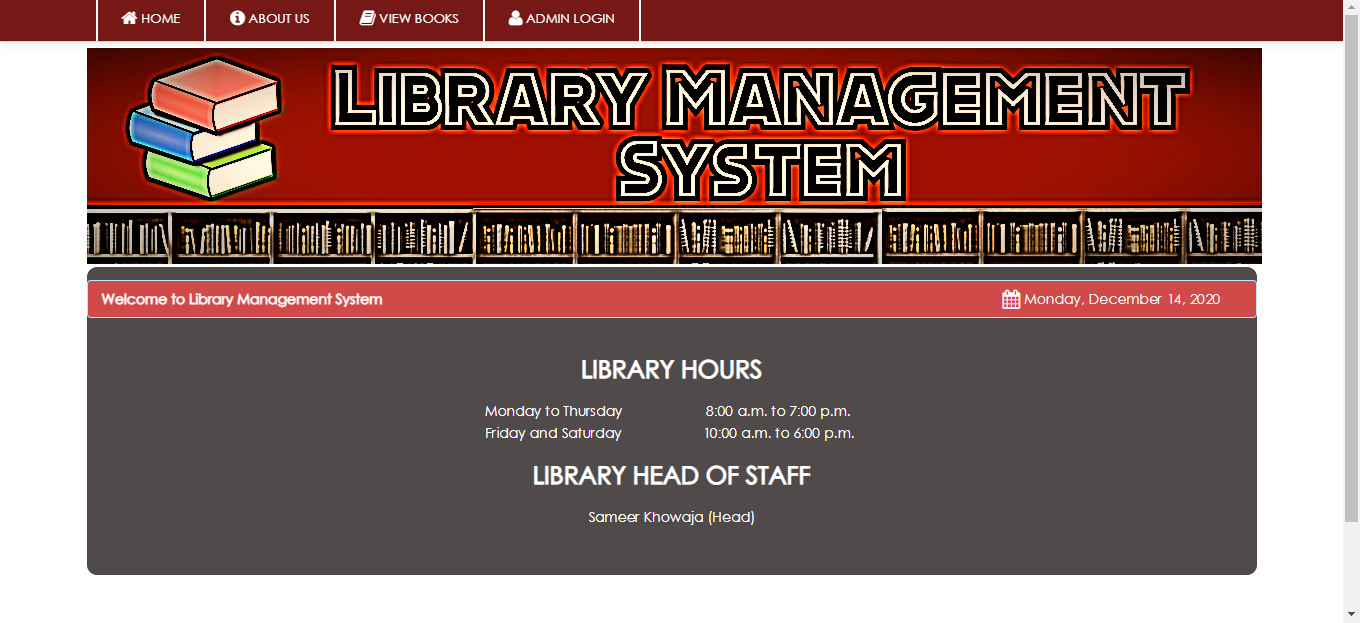
* 1. ***Gantt Chart Diagram***

******

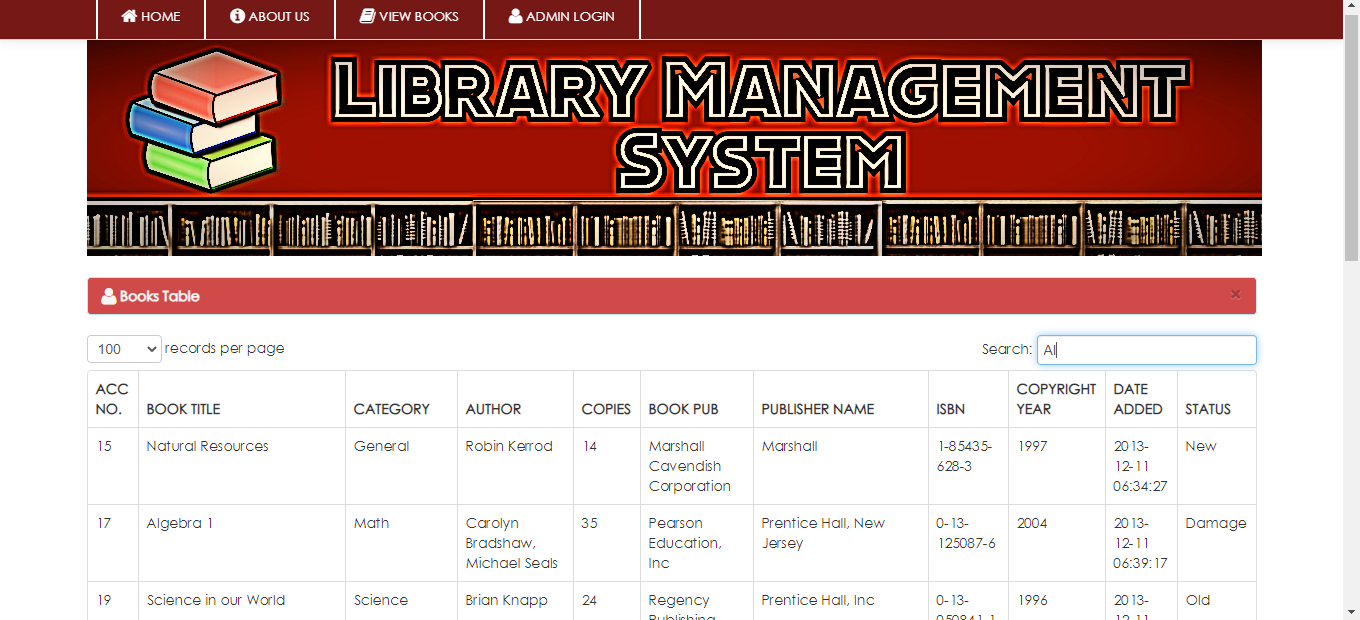
* 1. ***Screenshot of Project***
     1. ***Home Page***

******

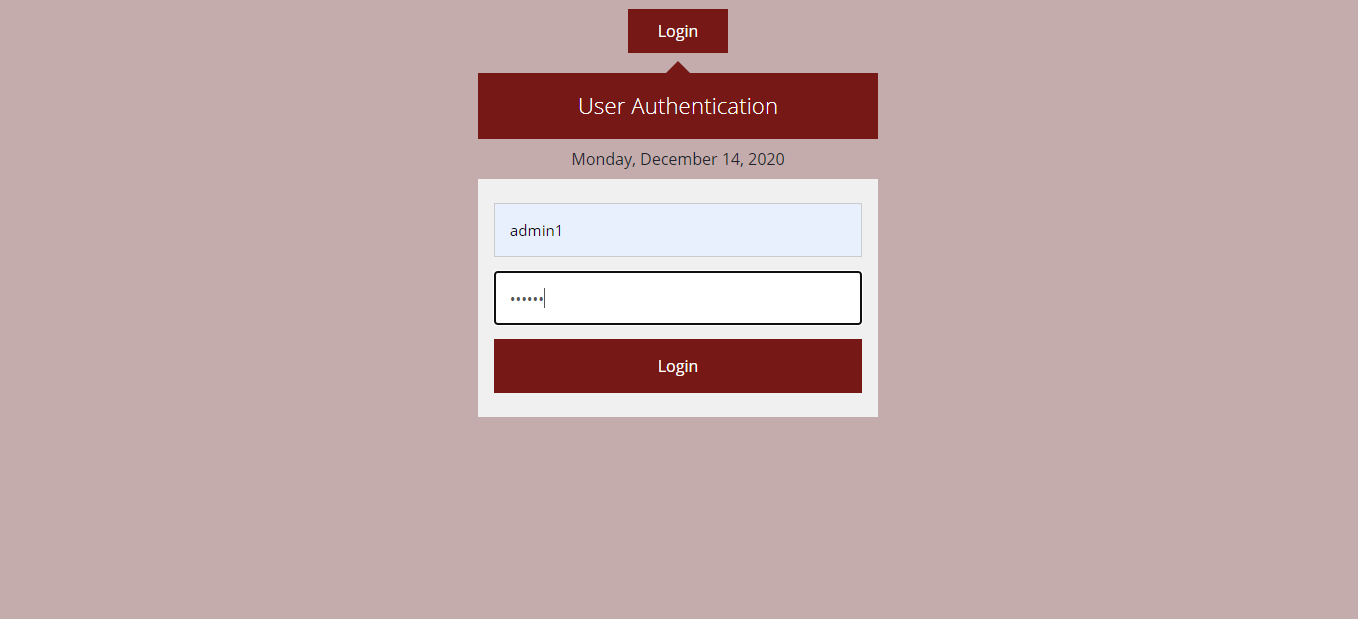
* + 1. ***About Us***

******

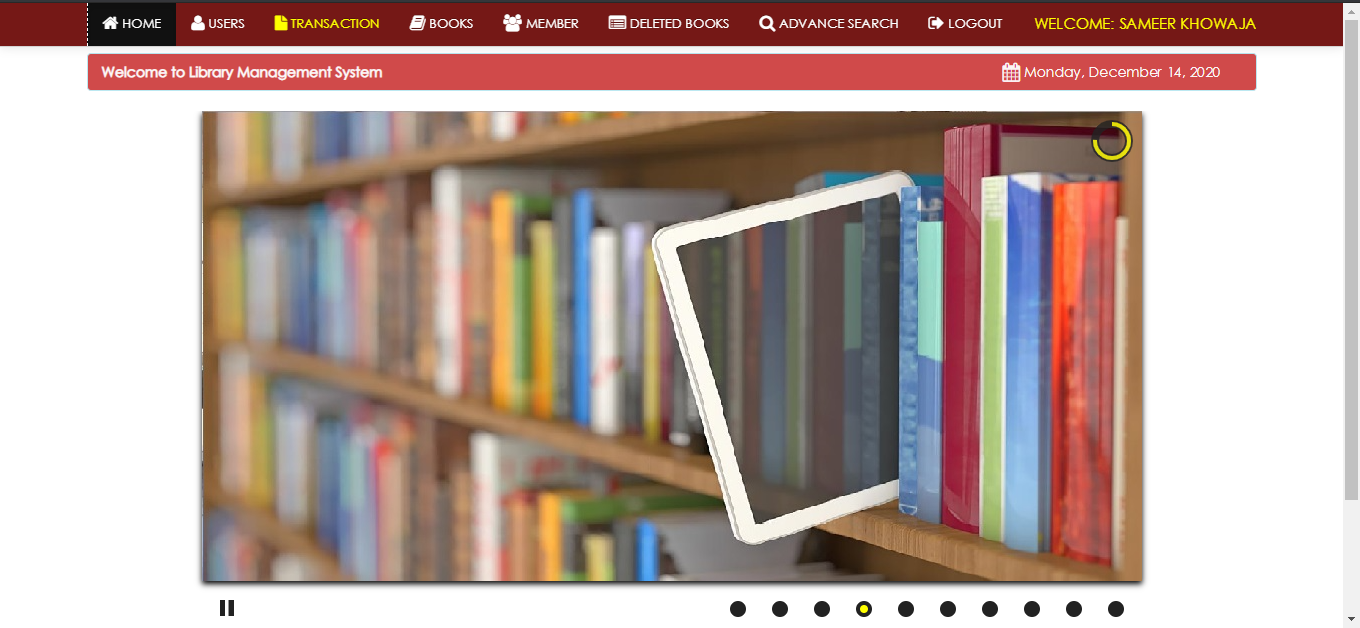
* + 1. ***View Books***

******

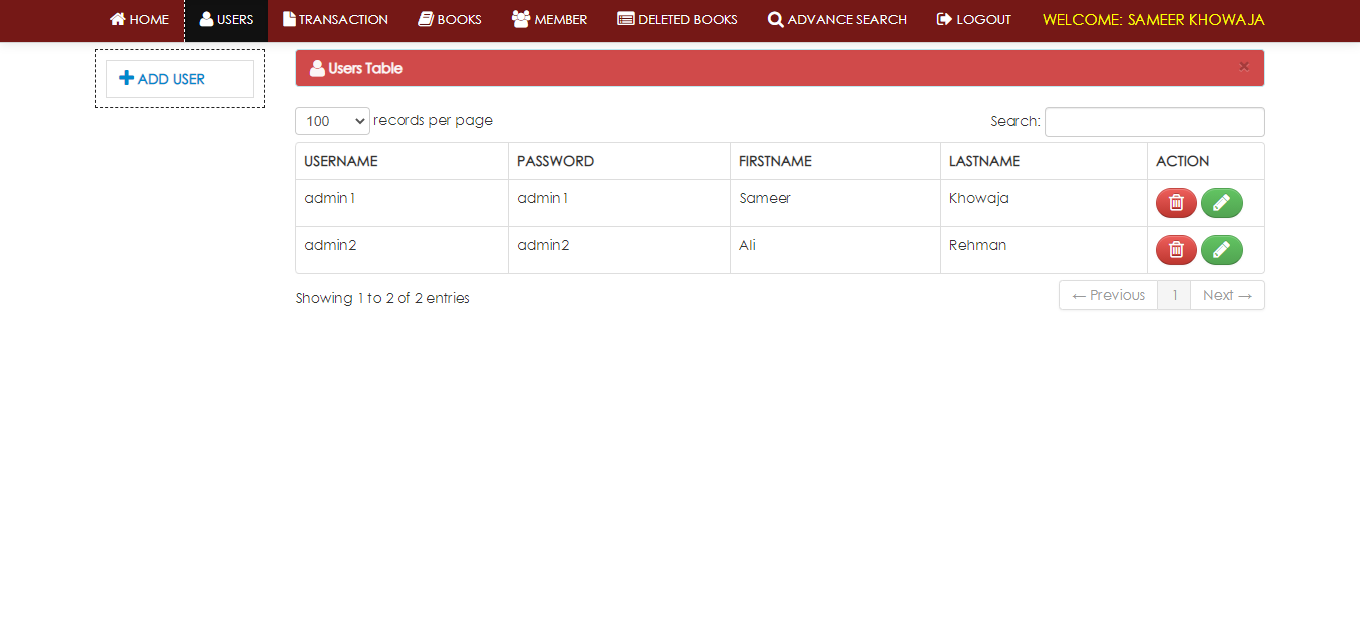
* + 1. ***Login***

******

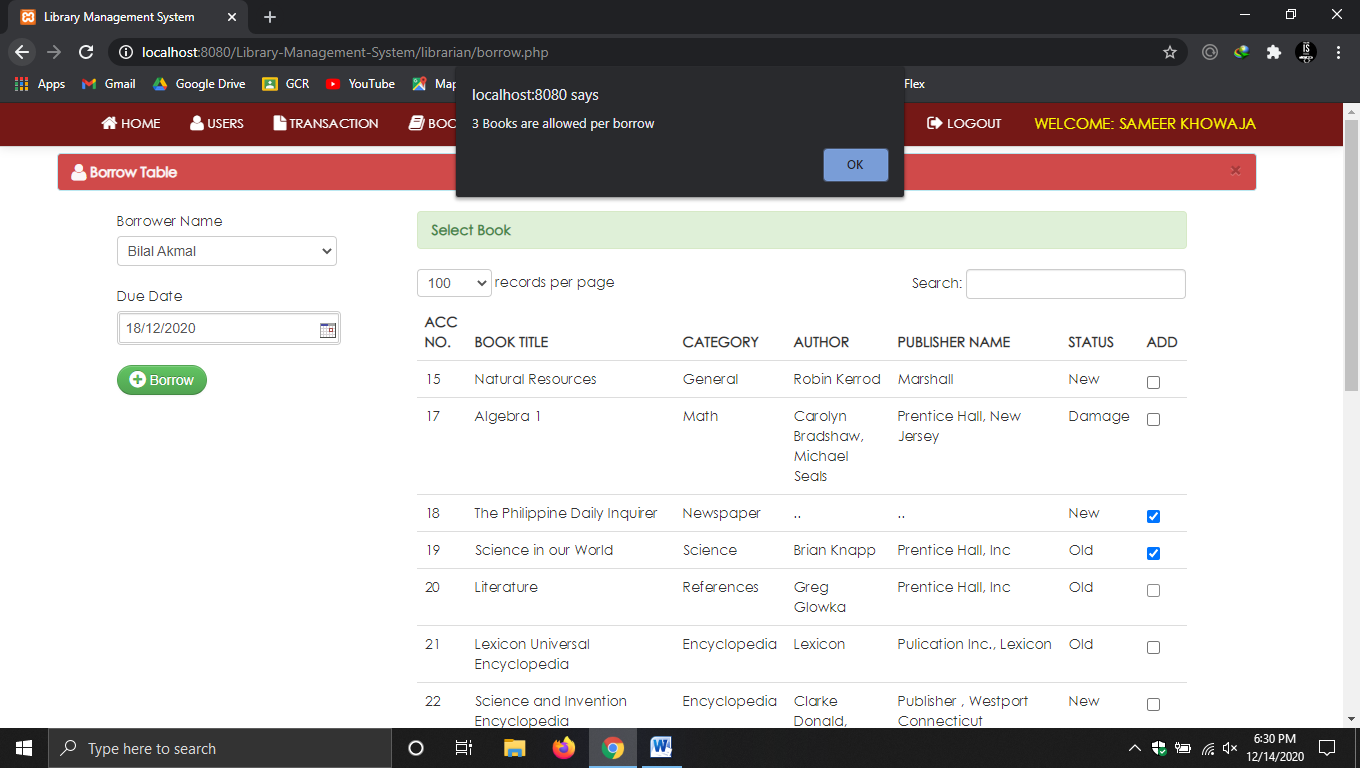
* + 1. ***Dashboard***

******

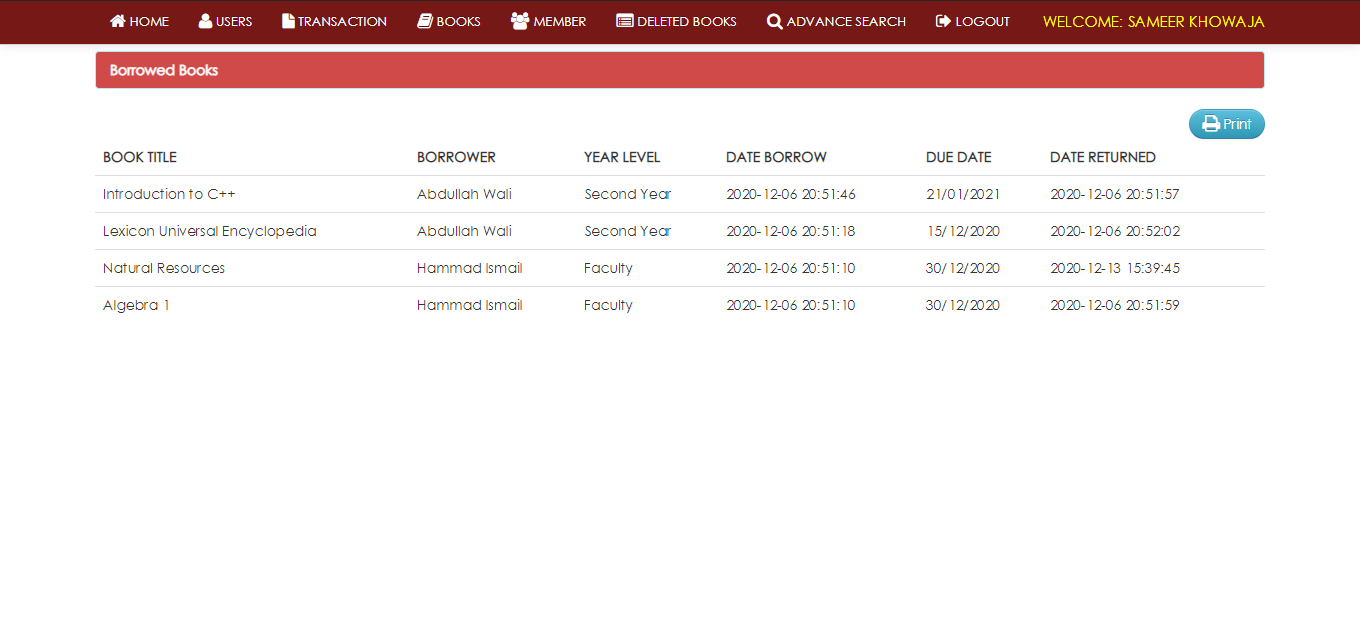
* + 1. ***Users Page***

******

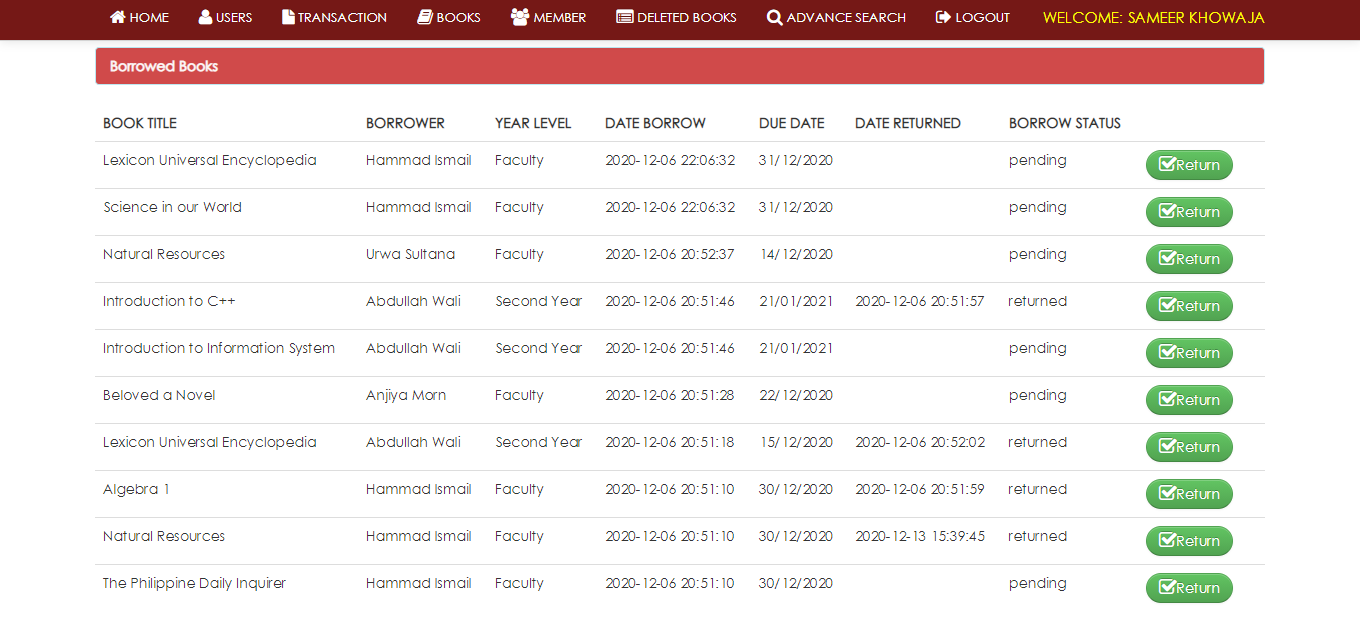
* + 1. ***Transactions***
       1. ***Borrow***

******

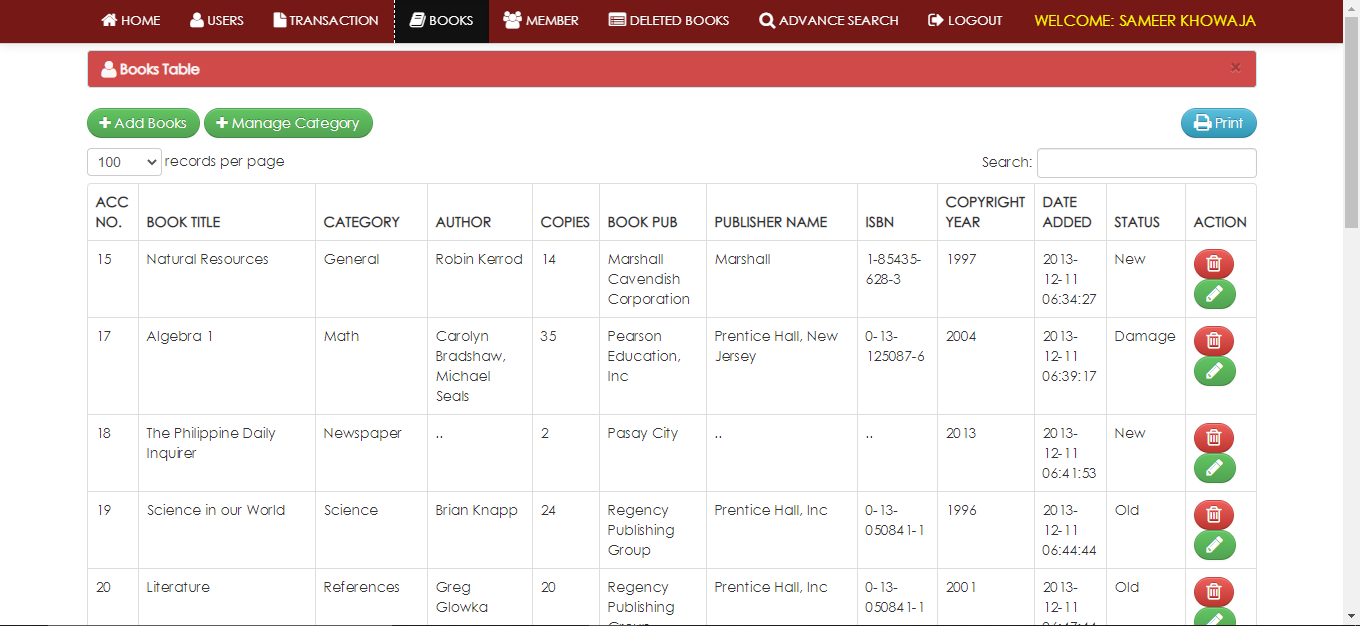
* + - 1. ***View Returns***

******

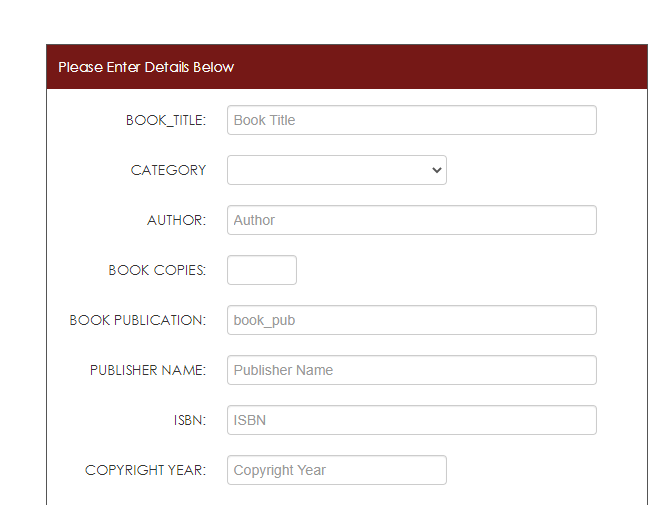
* + - 1. ***View Borrow books***

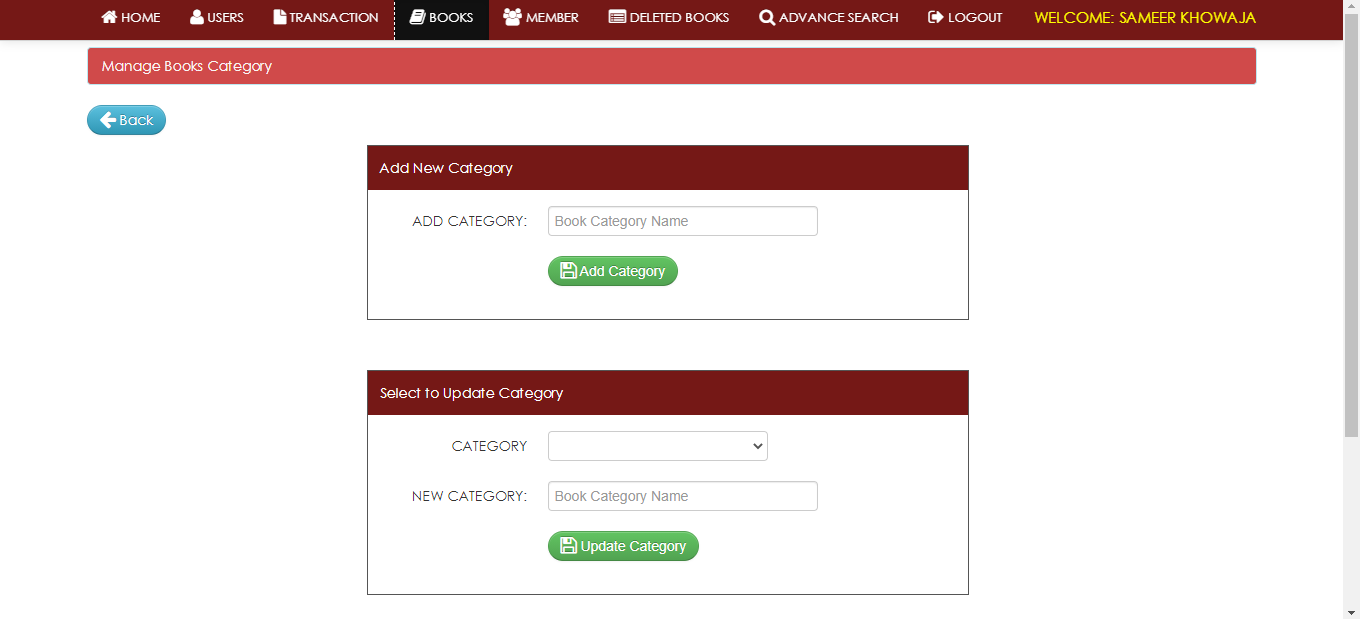
******

* + 1. ***Book***

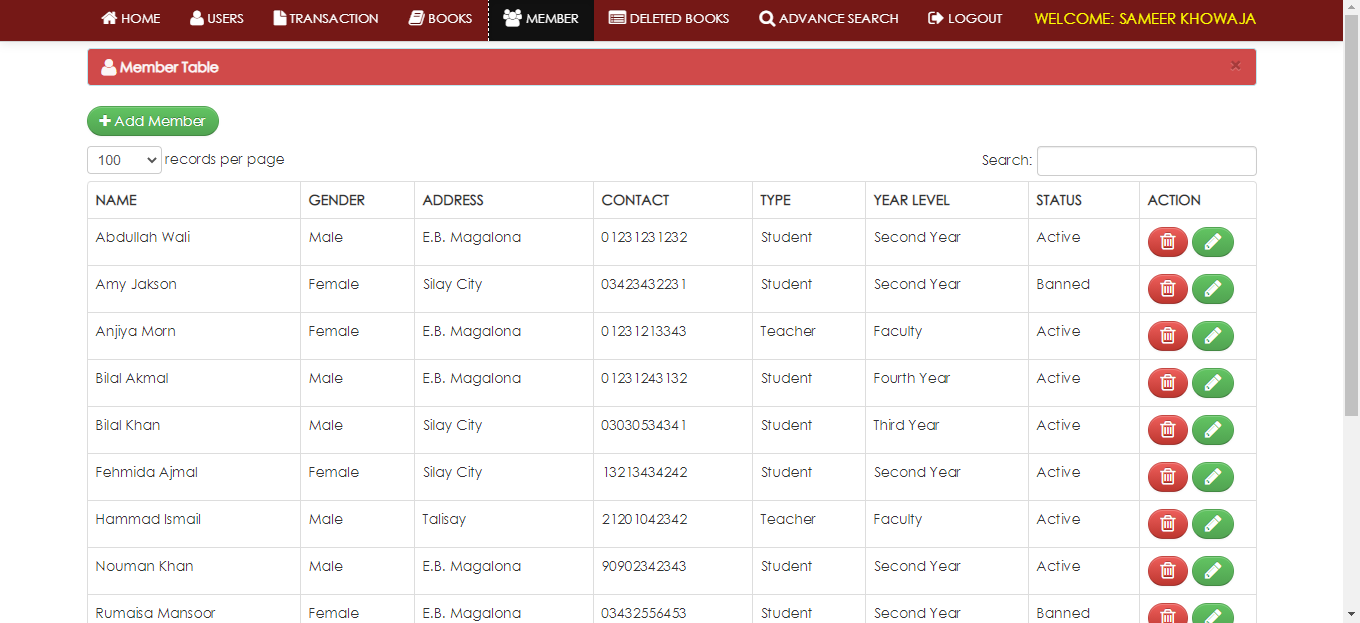
******

* + - 1. ***Manage books***

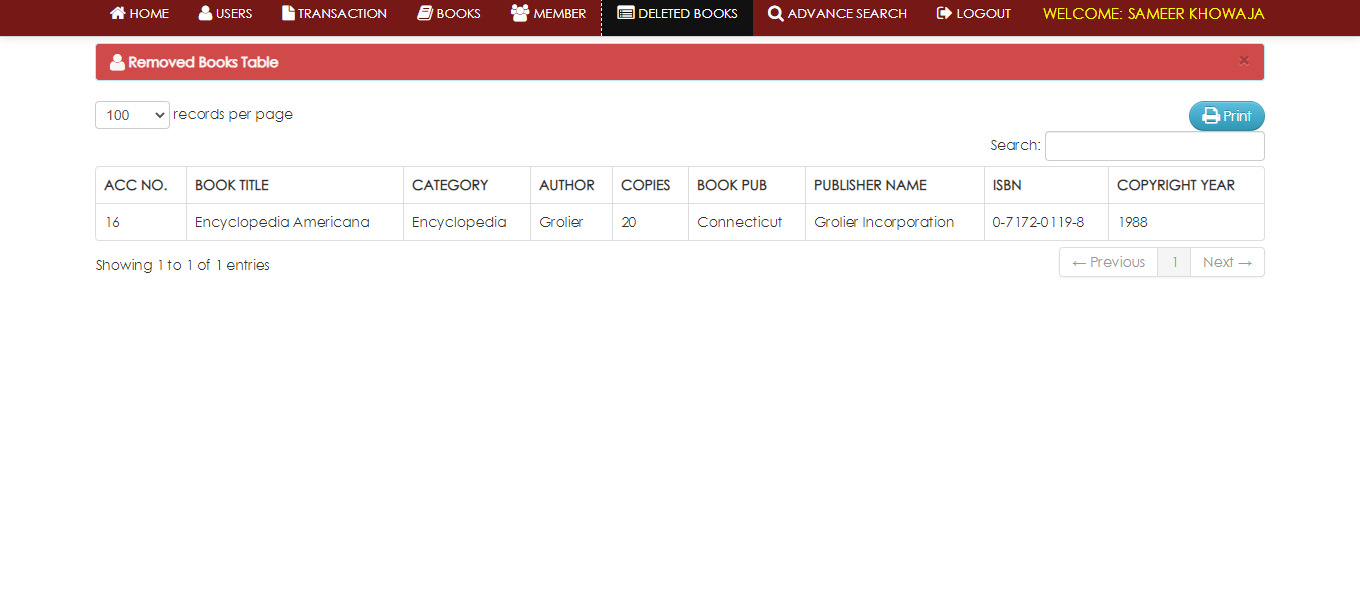
******

******

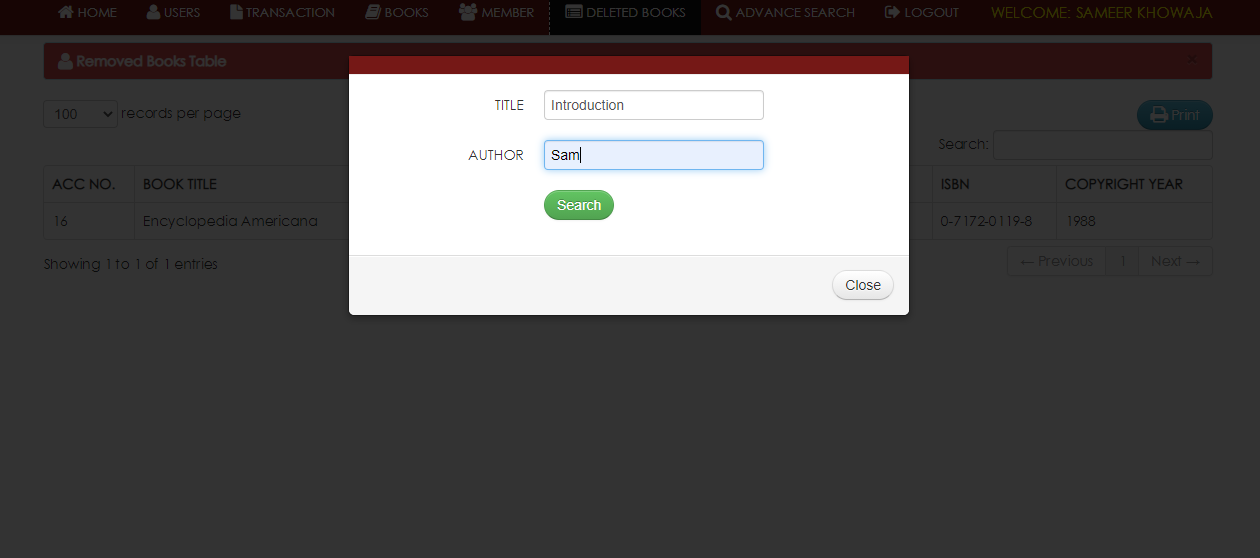
* + 1. ***Members Page***

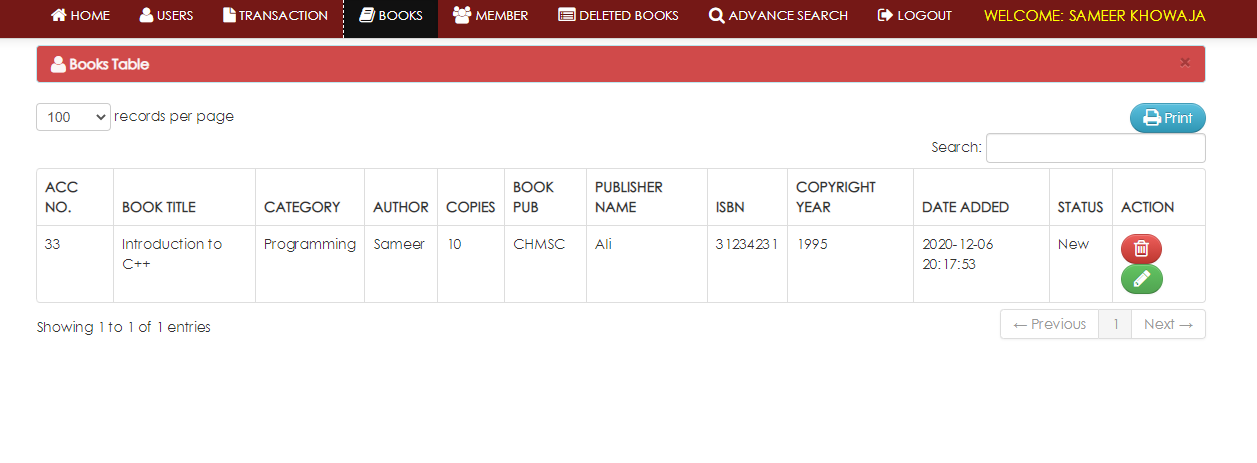
******

* + 1. ***Deleted books page***

******

* + 1. ***Advance Search***

******

******

1. ***Project Testing***

The aim of the system testing process was to determine all defects in our project .The program was subjected to a set of test inputs and various observations were made and based on these observations it will be decided whether the program behaves as expected or not. Our Project went through two levels of testing i.e. Unit testing and integration testing.

* 1. ***Unit Testing***

Unit testing is undertaken when a module has been created and successfully reviewed .In order to test a single module we need to provide a complete environment i.e. besides the module we would require.

∙ The procedures belonging to other modules that the module under test calls

∙ Non local data structures that module accesses

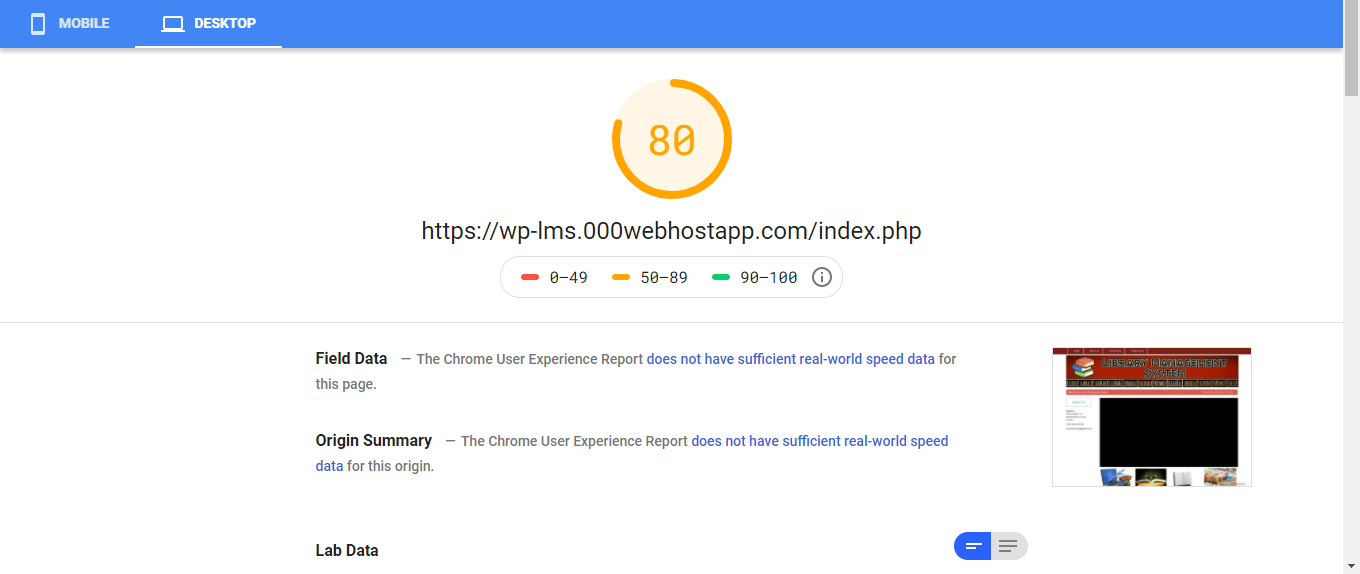
∙ A procedure to call the functions of the module under test with appropriate parameters Unit testing was done on each and every module

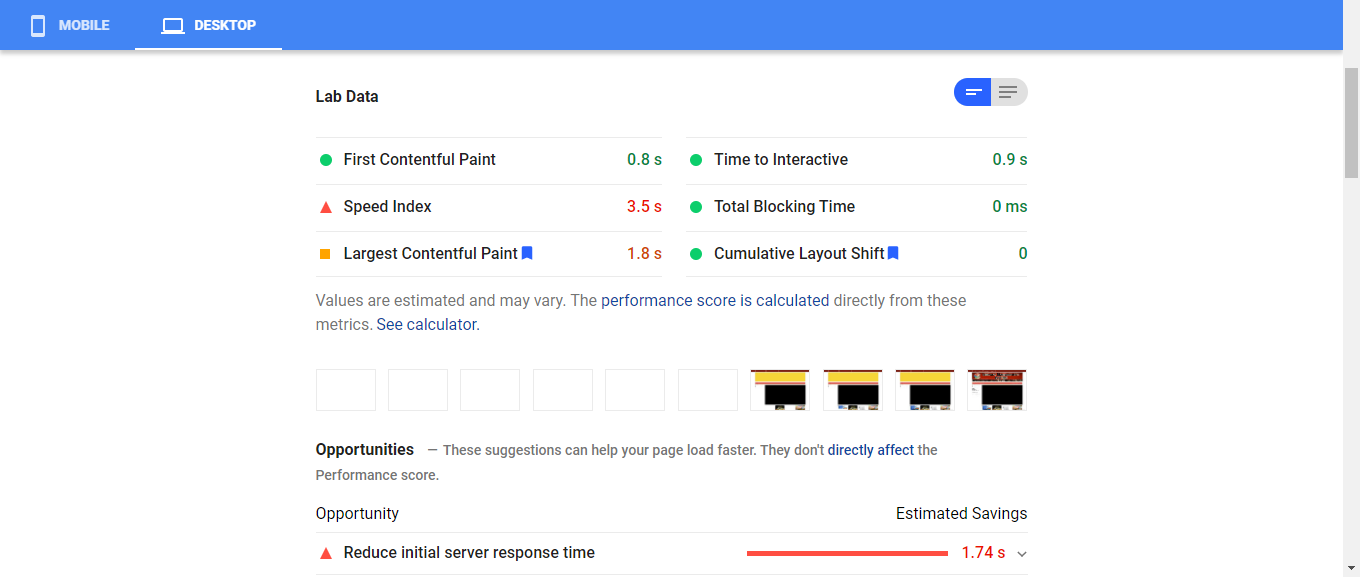
* + 1. **Test for the librarian admin module**
       1. This form is used for login of the administrator of the system. In this we enter the username and password if both are correct, the administration page will open otherwise if any of data is wrong it will get redirected back to the login page and again ask for username and password.
       2. After Successful login, the admin can perform various module tasks.
          1. Manage User i.e. add/update/delete admin information.
          2. Manage Books i.e. add/update/delete books information.
          3. Manage Books category.
          4. Manage Members i.e. add/update/delete member information.
          5. Ban or unban members from issuing books.
          6. Issue book to on request.
          7. Add book back on return by member.
          8. Searching records i.e. books, members, etc.
          9. Advance search for books.
          10. Can view deleted books.
          11. Logout from window.
    2. **Test for the member and other user**
       1. This form is used for users/members of the library. Who can perform various tasks on different modules
          1. View home page for details like social media accounts info
          2. View contact details.
          3. Search for book availability.
  1. ***Integration Testing***

In this type of testing we test various integration of the project module by providing the input. The primary objective is to test the module interfaces in order to ensure that no errors are occurring when one module invokes the other module.

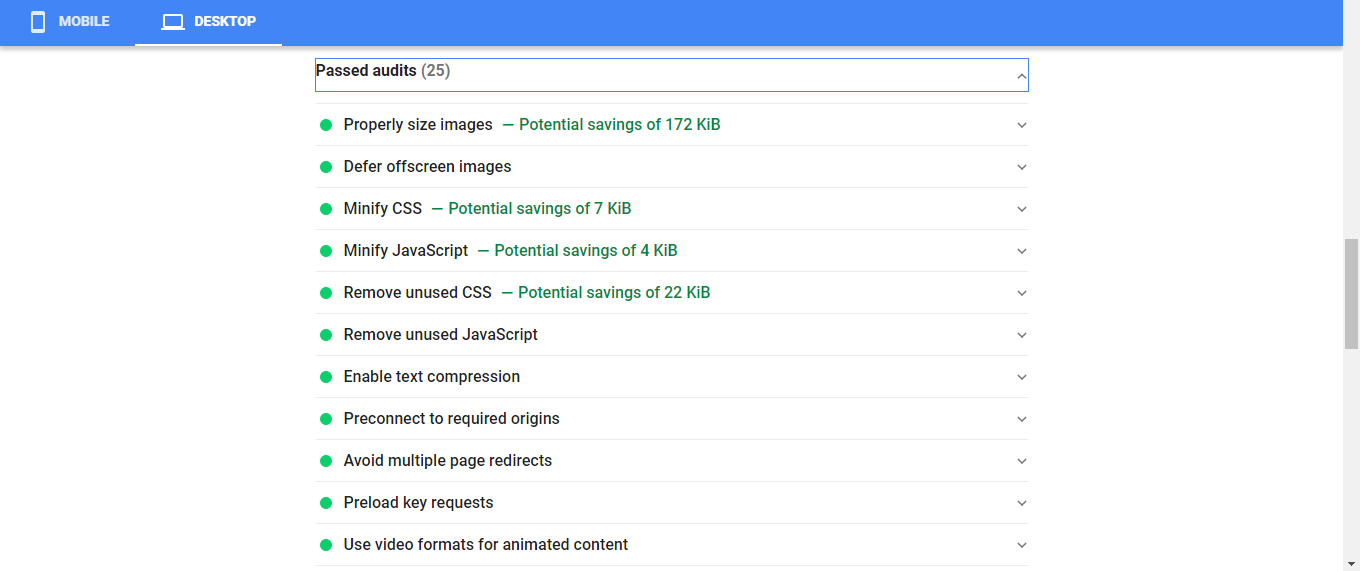
* 1. ***Load Testing***

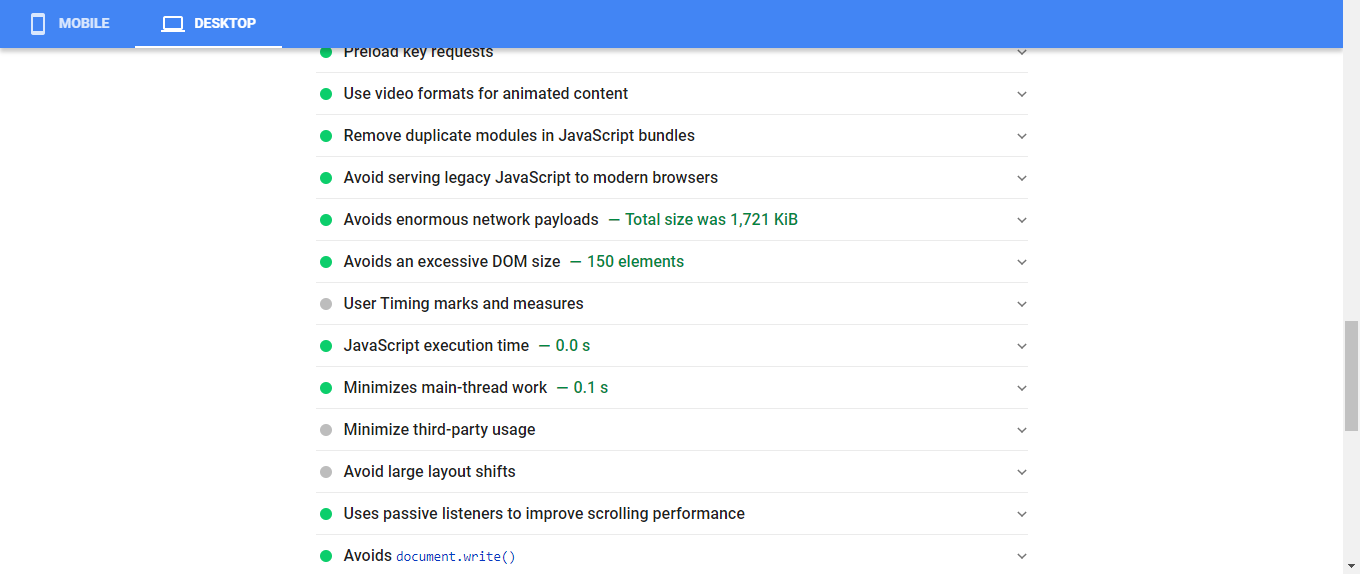
**Load Testing** is a non-functional software testing process in which the performance of software application is tested under a specific expected load. It determines how the software application behaves while being accessed by multiple users simultaneously. The goal of Load Testing is to improve performance bottlenecks and to ensure stability and smooth functioning of software application before deployment.





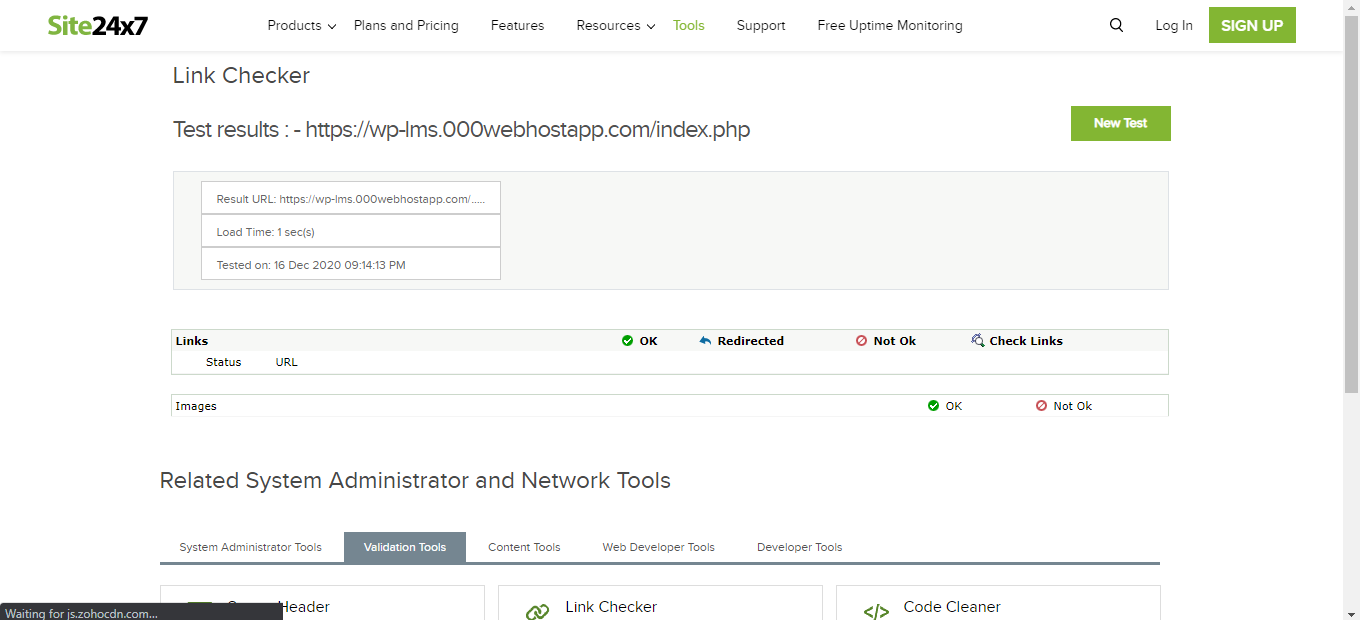


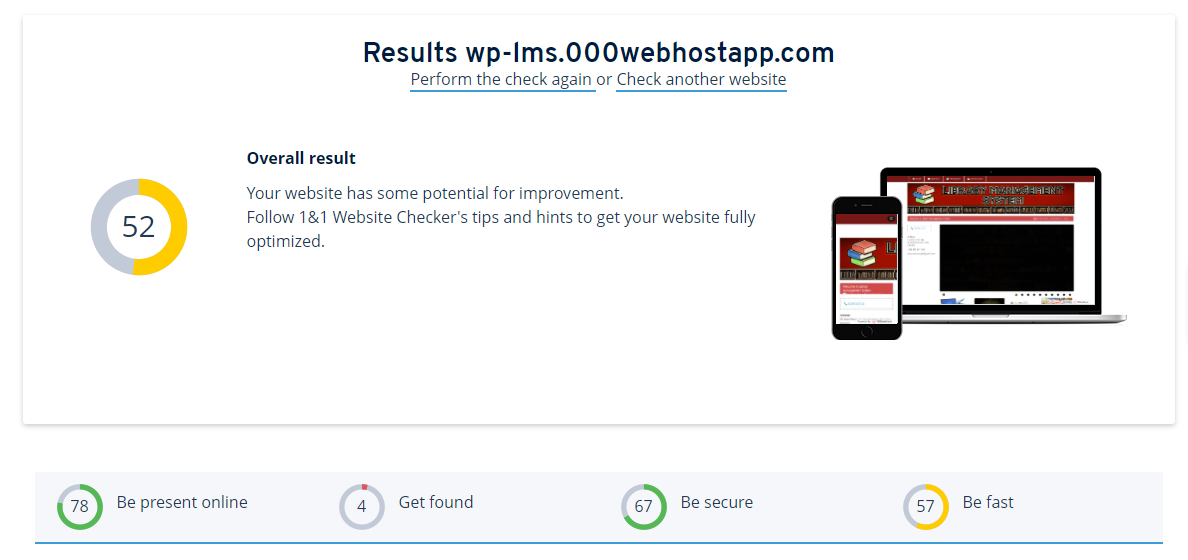




* 1. ***Stress Testing***

[Stress Testing](https://www.guru99.com/stress-testing-tutorial.html) determines the breaking point of the system to reveal the maximum point after which it breaks.





1. ***References***

[1] Elmarsi and Navathe, Fundamentals of Database System (Third Edition), Addision Wesley.

[2] Ivan Bayross, SQL, PL/SQL programming language of Oracle, Second Edition, BPB Publication.

[3] PHP MySQL Website Programming: Problem - Design – Solution byChris Lea, Mike Buzzard, Dilip Thomas , Jessey White-Cinis

[4] Theodore H. Tulchinsky MD, MPH, Elena A. Varavikova MD, MPH, PhD, in [The New Public Health (Third Edition)](https://www.sciencedirect.com/book/9780124157668/the-new-public-health), 2014