**ACKNOWLEDGEMENT**

The completion of project brings a sense of satisfaction, but it is never complete without thanking the persons responsible for its successful completion.

We extend our deep sense of sincere gratitude to **Dr. B R Lakshmikantha, Principal**, **DSATM,** Bangalore,for providing us an opportunity to finish the necessary work.

We express our heartfelt sincere gratitude to **Dr. Sumithra Devi K.A**, **Head of Department of Information Science and Engineering**, DSATM**,** Bangalore,for her valuable suggestions and support.

We extend a sincere gratitude to **Mrs. Bhavyashree**, **Assistant Professor**, **Department of Information Science and Engineering**, DSATM**,** Bangalore, for her constant support as the project guide for completion of the project work.

Finally, we would like to thank all the Teaching and Non-Teaching faculty members of Department of Information Science and Engineering, DSATM, Bangalore, for their support.

**ABSTRACT**

"**LIBRARY DATABASE MANAGEMENT SYSTEM**" is targeted at users who want an integrated experience. In this scholarly thesis pertinent to the setting up of a automated library performance record management system which enables the users of a university like library and faculty to access the important information with ease through a user friendly web application. This proposed system aims at eliminating the practice of time consuming and vulnerable tradition of manual maintenance of library information in paper at the very basic level. In a university there are many departments all these departments provide various records regarding library. Most of these track records need to maintain information about the librarys. Thus by proposing a computerizes library record management system will enable the users to access data at any time and any place. The library web portal enables huge storage of data and easy retrieval. There are many departments in a college thus but introducing a library web portal will centralize the administration and the entire system will work as one single entity.

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ACKNOWLEDGEMENT** |  |
|  | **ABSTRACT** |  |
|  | **TABLE OF CONTENTS** |  |
| **1** | **INTRODUCTION** | **1-2** |
| **2** | **REQUIREMENT ANALYSIS** | **3** |
| **3** | **DESIGN** | **4-6** |
| **4** | **IMPLEMENTATION** | **7-10** |
| **5** | **TESTING** | **11-12** |
| **6** | **SNAPSHOTS** | **13-15** |
| **7** | **CONCLUSION AND FUTURE**  **ENHANCEMENTS** | **16** |
|  | **REFERENCES** |  |

**CHAPTER 1**

**INTRODUCTION**

**OBJECTIVE**

**TO DESIGN LIBRARY MANAGEMENT SYSTEM :**

The Purpose of the project is to build a web application program to reduce the manual work for managing student’s Profile in a library, Result, User Logins, Admin logins, Issuing books, Co-curricular-activities.

* 1. **DATABASE MANAGEMENT** **SYSTEM**

A database management system (DBMS) refers to the technology for creating and managing databases. Basically DBMS is a software tool to organize (create, retrieve, update and manage) data in a database. The main aim of a DBMS is to supply a way to store and retrieve database information that is both convenient and efficient. By data, we mean known facts that can be recorded and that have embedded meaning. Normally people use software such as Microsoft ACCESS, or EXCEL to store data in the form of database.

Some among them which are widely used are:-

1) Library database management system

2) Restaurant database management system

3) Salary database management system

4) Whole shale database management system

Thus this project deals with a concept derived from DBMS, it is a unique attempt to aid the management of information of librarys in a university know as Library Database Management System.

**What is library database management system?**

A library database management system is automation of manual performance record management which enables the user to assess necessary data at any place and any time through internet. The user can access the library details once they run the application therefore the library details is appeared for the user where it shows important information and records in the library like, book name, book authors, book id, issue date, return date, student details, etc.

The admin of library can update all the book details along with the student information.

* 1. **OVER VIEW OF SYSTEM**
* **Signup**

Create student profiles in the library with categories and fields including email, name, usn, and custom password.

* **Feedback**

Shows the feedback given by students.

* **Book**

Contain details of all the books present in the library.

* **Issue book**

This feature includes the description of students who have issued book.

* **Fine**

Provides the data of fine generated for late submission.

**CHAPTER 2**

**REQUIREMENTS ANALYSIS**

The requirement analysis specifies the requirements needed to develop a graphic project. In this phase, we collect the requirements needed for designing the project. The requirements collected are then examined and carried to the next phase.

**2.1 SOFTWARE REQUIREMENTS:**

1. Operating System: Windows 10
2. Back-end Development: MYSQL
3. phpMyAdmin Version 5.0.4
4. VDDT: MySQL Workbench
5. Any python compatible IDE. (eg:- VSC,pycharm,VS)

**2.2 HARDWARE REQUIREMENTS**

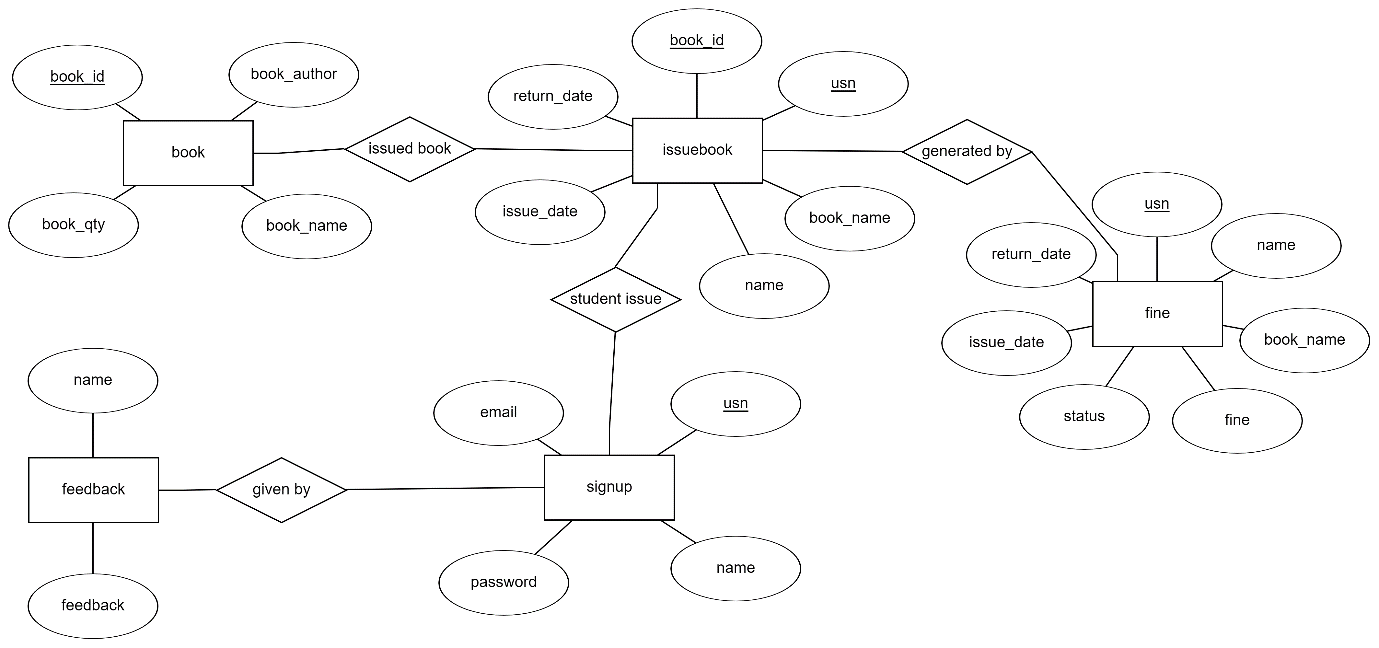
1. Processor – 1.8 GHz Dual-Core Intel Core i5 or Pentium V in windows or above
2. RAM – 8GB 1600 MHz DDR3 or 2 GB in Windows or more
3. Hard disk – Macintosh HD 2GB or 3 GB in Windows or more

**CHAPTER 3**

**DESIGN**

**3.1 ER DIAGRAM**

An **entity-relationship diagram** is a data modelling technique that graphically illustrates an information system's entities and the relationships between those entities.



**Figure 3.1: Entity – Relational diagram of Library Database Management System**

**The E-R diagram of our LIBRARY DATABASE MANAGEMENT SYSTEM contains 5 Entities:**

**1. ISSUE BOOK**

**2. SIGNUP**

**3. BOOK**

**4. FINE**

**5. FEEDBACK**

**3.2 SCHEMA DIAGRAM**

The term "schema" refers to the organization of data as a blueprint of how the database is constructed (divided into database tables in the case of **relational** databases).

Signup

|  |  |  |  |
| --- | --- | --- | --- |
| usn | email | name | password |

Book

|  |  |  |  |
| --- | --- | --- | --- |
| book\_id | book\_author | book\_name | book\_qty |

issuebook

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| book\_id | usn | book\_name | name | issue\_date | return\_date |

fine

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| usn | name | book\_name | fine | status | issue\_date | return\_date |

Feedback

|  |  |  |
| --- | --- | --- |
| usn | name | feedback |

**Figure 3.2: Schema Diagram**

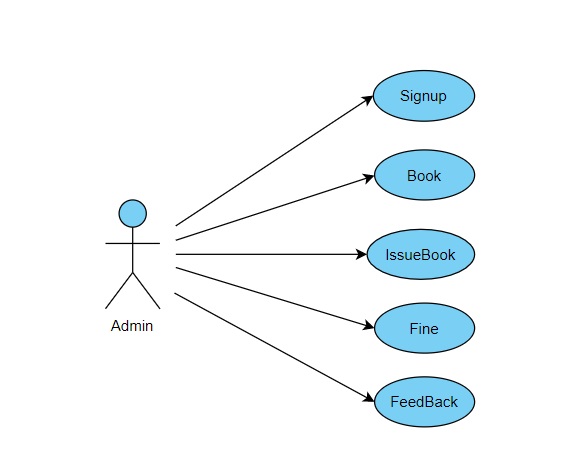
## 3.3 USE CASE DIAGRAM

The main purpose of a use case diagram is to show who interacts with your system, and the main goals they achieve with it.

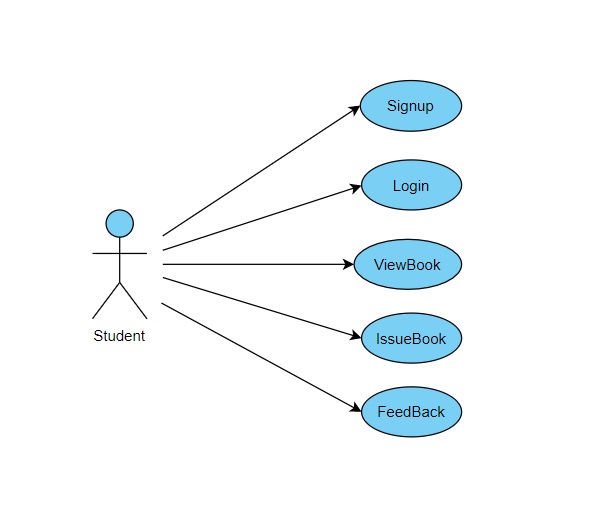
The boundary, which defines the system of interest in relation to the world around it. The actors, usually individuals involved with the system defined according to their roles. The use cases, which are the specific roles played by the actors within and around the system.

In software and systems engineering, a use case is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modeling Language (UML) as an actor) and a system to achieve a goal.

The actor can be a human or other external system. In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or stakeholder goals. The detailed requirements may then be captured in the Systems Modeling Language (SysML) or as contractual statements.



**Figure 3.3.1: Use Cases for Admin**

****

**Figure 3.4.2: Use Cases for Students**

**CHAPTER 4**

**IMPLEMENTATION**

**4.1 INTRODUCTION TO BACKEND END TOOL**

**4.1.1 MYSQL**

**MySQL**: MySQL is a **R**elational **D**ata-**B**ase **M**anagement **S**ystem (RDBMS).

RDBMS means R—DB--MS.

- DB stands for Database, a repository for the information store.

1. The data in a database is organized into tables, and each table is organized into rows and columns.
2. Each row in a table is called a record. A record may contains several pieces (called fields) of information, and each column in a table is known as a field.

-MS stands for Management System, the software that allows you to insert, retrieve, modify, or delete records.

-R stands for Relational, indicates a particular kind of DBMS that is good at relating information stored in one table to information stored in another table by looking for elements common to each of them. Relational DBMS has the advantage of efficient storage, and retrieval mechanisms for data, and uses normalization process during design of RDBMS. MySQL operates using client/server architecture in which the server runs on the machine containing the databases and clients connect to the server over a network. The server operating systems is usually a Linux or Windows 2000 operating system. Typically MySQL is supported on Windows XP, Windows Server 2003, Red Hat Fedora Linux, and Debian Linux, and others. As with any other client/server application, MySQL is a multiuser database system, meaning several users can access the database simultaneously.

**4.1.2 MYSQL Workbench**

**MySQL** Workbench is a visual database design tool that integrates SQL development, administration, database design, creation and maintenance into a single integrated development environment for the MySQL database system. It is the successor to DBDesigner 4 from fabFORCE.net, and replaces the previous package of software, MySQL GUI Tools Bundle.

It is [free and open-source software](https://en.wikipedia.org/wiki/Free_and_open-source_software) under the terms of the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License), and is also available under a variety of [proprietary](https://en.wikipedia.org/wiki/Proprietary_software) licenses. **MySQL** was owned and sponsored by the [Swedish](https://en.wikipedia.org/wiki/Sweden) company [MySQL AB](https://en.wikipedia.org/wiki/MySQL_AB), which was bought by [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems) (now [Oracle Corporation](https://en.wikipedia.org/wiki/Oracle_Corporation)). In 2010, when Oracle acquired Sun, Widenius [forked](https://en.wikipedia.org/wiki/Fork_(software_development)) the [open-source](https://en.wikipedia.org/wiki/Open-source) MySQL project to create [MariaDB](https://en.wikipedia.org/wiki/MariaDB).

MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the [LAMP](https://en.wikipedia.org/wiki/LAMP_(software_bundle)) [web application](https://en.wikipedia.org/wiki/Web_application) [software stack](https://en.wikipedia.org/wiki/Software_stack) (and [others](https://en.wikipedia.org/wiki/List_of_AMP_packages)), which is an acronym for [*Linux*](https://en.wikipedia.org/wiki/Linux)*,*[*Apache*](https://en.wikipedia.org/wiki/Apache_HTTP_Server)*, MySQL,*[*Perl*](https://en.wikipedia.org/wiki/Perl)*/*[*PHP*](https://en.wikipedia.org/wiki/PHP)*/*[*Python*](https://en.wikipedia.org/wiki/Python_(programming_language)). MySQL is used by many database-driven web applications, including [Drupal](https://en.wikipedia.org/wiki/Drupal), [Joomla](https://en.wikipedia.org/wiki/Joomla), [phpBB](https://en.wikipedia.org/wiki/PhpBB), and [WordPress](https://en.wikipedia.org/wiki/WordPress). MySQL is also used by many popular [websites](https://en.wikipedia.org/wiki/Website), including [Facebook](https://en.wikipedia.org/wiki/Facebook), [Flickr](https://en.wikipedia.org/wiki/Flickr), MediaWiki, Twitter and Youtube

A variety of MySQL [forks](https://en.wikipedia.org/wiki/Fork_(software_development)) exist, including the following.

### **Current :**

**MariaDB**

[MariaDB](https://en.wikipedia.org/wiki/MariaDB) is a community-developed fork of the MySQL relational database management system intended to remain free under the GNU GPL. The fork has been led by the original developers of MySQL, who forked it due to concerns over its acquisition by Oracle.[[33]](https://en.wikipedia.org/wiki/MySQL#cite_note-dead-33)

**Percona Server for MySQL**

[Percona Server for MySQL](https://en.wikipedia.org/wiki/Percona_Server_for_MySQL), forked by [Percona](https://en.wikipedia.org/wiki/Percona), aims to retain close compatibility to the official MySQL releases. Also included in Percona Server for MySQL is [XtraDB](https://en.wikipedia.org/wiki/XtraDB), Percona's fork of the [InnoDB Storage Engine](https://en.wikipedia.org/wiki/InnoDB).

### **Abandoned :**

**Drizzle**

[Drizzle](https://en.wikipedia.org/wiki/Drizzle_(database_server)) was a free software/open source relational database management system (DBMS) that was forked from the now-defunct 6.0 development branch of the MySQL DBMS.[[112]](https://en.wikipedia.org/wiki/MySQL#cite_note-Clark_2011-115) Like MySQL, Drizzle had a [client/server](https://en.wikipedia.org/wiki/Client/server) architecture and uses [SQL](https://en.wikipedia.org/wiki/SQL) as its primary [command language](https://en.wikipedia.org/wiki/Command_language). Drizzle was distributed under version 2 and 3 of the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License) (GPL) with portions, including the protocol drivers and [replication](https://en.wikipedia.org/wiki/Database_replication) messaging under the [BSD](https://en.wikipedia.org/wiki/BSD_licenses) license.

**WebScaleSQL**

[WebScaleSQL](https://en.wikipedia.org/wiki/WebScaleSQL) was a software branch of MySQL 5.6, and was announced on 27 March 2014 by Facebook, Google, LinkedIn and Twitter as a joint effort to provide a centralized development structure for extending MySQL with new features specific to its large-scale deployments, such as building large replicated databases running on server farms. Thus, WebScaleSQL opened a path toward deduplicating the efforts each company had been putting into maintaining its own branch of MySQL, and toward bringing together more developers. By combining the efforts of these companies and incorporating various changes and new features into MySQL, WebScaleSQL aimed at supporting the deployment of MySQL in large-scale environments. The project's source code is licensed under version 2 of the GNU General Public License, and is hosted on [GitHub](https://en.wikipedia.org/wiki/GitHub).

**OurDelta**

The OurDelta distribution, created by the Australian company Open Query (later acquired by Catalyst IT Australia), had two versions: 5.0, which was based on MySQL, and 5.1, which was based on MariaDB. It included patches developed by Open Query and by other notable members of the MySQL community including Jeremy Cole and Google. Once the patches were incorporated into the MariaDB mainline, OurDelta's objectives were achieved and OurDelta passed on its build and packaging toolchain to Monty Program (now MariaDB Corp).

**4.1.3 PhpMyAdmin**

PhpMyAdmin is a free software tool written in PHP, intended to handle the administration of MySQL over the Web. phpMyAdmin supports a wide range of operations on MySQL and MariaDB. Frequently used operations (managing databases, tables, columns, relations, indexes, users, permissions, etc) can be performed via the user interface, while you still have the ability to directly execute any SQL statement. phpMyAdmin comes with a wide range of documentation and users are welcome to update our wiki pages to share ideas and how to use for various operations. The phpMyAdmin team will try to help you if you face any problem; you can use a variety of support channels to get help . phpMyAdmin is also very deeply documented in a book written by one of the developers – Mastering phpMyAdmin for Effective MySQL Management, which is available in English and Spanish.

**4.1.3.1 Features**

* Intuitive web interface
* Support for most MySQL features:

1. browse and drop databases, tables, views, fields and indexes
2. create, copy, drop, rename and alter databases, tables, fields and indexes
3. maintenance server, databases and tables, with proposals on server configuration
4. execute, edit and bookmark any SQL-statement, even batch-queries
5. manage MySQL user accounts and privileges
6. manage stored procedures and triggers

* Import data from CSV and SQL
* Export data to various formats: CSV, SQL, XML, PDF, ISO/IEC 26300 - OpenDocument Text and Spreadsheet, Word, LATEX and others
* Administering multiple servers
* Creating graphics of your database layout in various formats
* Creating complex queries using Query-by-example (QBE)
* Searching globally in a database or a subset of it
* Transforming stored data into any format using a set of predefined functions, like displaying BLOB-data as image or download-link

And much more...

**CHAPTER 5**

**TESTING**

**5.1 TESTING**

Testing is the process of executing a program to find the errors. A good test has the high probability of finding a yet undiscovered error. A test is vital to the success of the system. System test makes a logical assumption that if all parts of the system are correct, then goal will be successfully achieved.

**5.2 TYPES OF TESTING**

5.2.1 Module Testing.

5.2.2 Integration Testing.

**5.2.1 Module Testing**

Module testing is the testing of complete code objects as produced by the compiler when built from source.

A library may be composed of a single complied object or several complied objects. There is only a slight difference between unit testing and module testing. Modules are fully formed chunks of coherent source code that can typically be tested by driving a few functions signatures with various stimuli. On the other hand, unit testing (which is considered as part of the implementation phase for this software development process) may involve testing one small part of a function that will never formally implement any function interface.

As a result of modules being more self-contained, module testing will likely require less testing infrastructure such as test harness and test stubs. The testing of modules could perhaps even be automated so that they can be included in regression test suites or a acceptance test suites.

**5.2.2 Integration Testing**

Integration testing the phase in [software testing](https://en.wikipedia.org/wiki/Software_testing) in which individual software modules are combined and tested as a group. It occurs after [unit testing](https://en.wikipedia.org/wiki/Unit_testing) and before [validation testing.](https://en.wikipedia.org/wiki/Software_verification_and_validation) Integration testing takes as its input [modules](https://en.wikipedia.org/wiki/Module_(programming)) that have been unit tested, groups them in larger aggregates, applies tests defined in an integration [test plan](https://en.wikipedia.org/wiki/Test_plan) to those aggregates, and delivers as its output the integrated system ready for [system testing.](https://en.wikipedia.org/wiki/System_testing)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Description** | **Input Data** | **Expected**  **Output** | **Actual Output** | **Status** |
| 1 | Signup | Usn  Email  Password  Name | Successfully  data inserted | Successfully  data inserted | Pass |
| 2 | Book | book\_id  book\_name  book\_author  book\_qty | Successfully  data inserted | Successfully  data inserted | Pass |
| 3 | Issuebook | Usn  Name  Book\_id  Book\_name  Issue\_date  Return\_date | Successfully  data inserted | Successfully  data inserted | Pass |
| 4 | Fine | Usn  Name  Book\_id  Book\_name  Issue\_date  Return\_date  Fine | Successfully  data inserted | Successfully  data inserted | Pass |
| 5 | Feedback | Name  Feedback | Successfully  data inserted | Successfully  data inserted | Pass |

**CHAPTER 6**

**SNAPSHOTS**

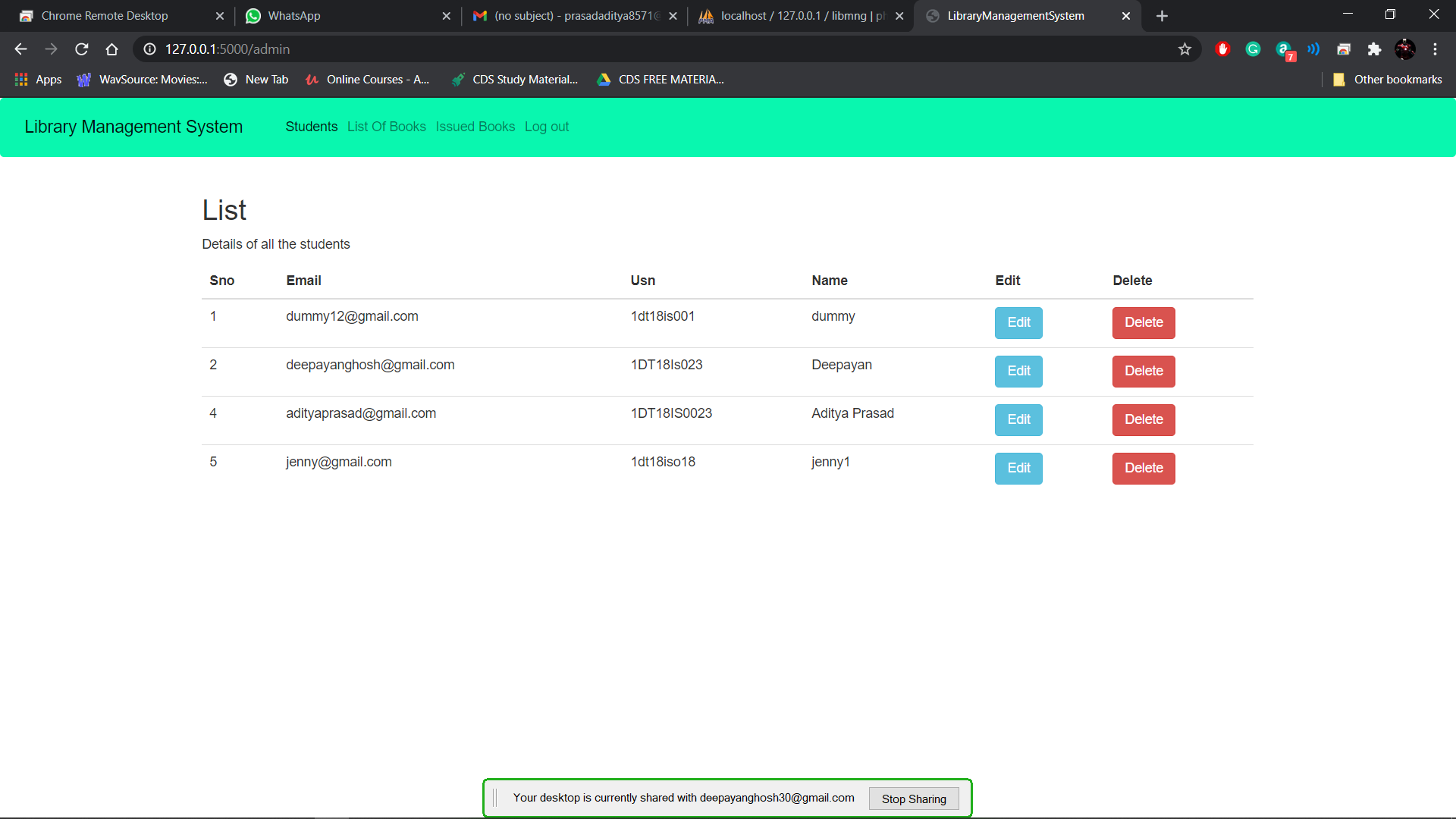
The user interface design was one of the core tasks of any application. The aim of UI is to provide a graphical-user-friendly interface to the end-users. UI makes it easy for the end users to access the application.

**6.1 USER INTERFACE (FRONT-END):**

This section will describe the Front-end Code design of the entire application

**6.1.1 Admin Portal:**

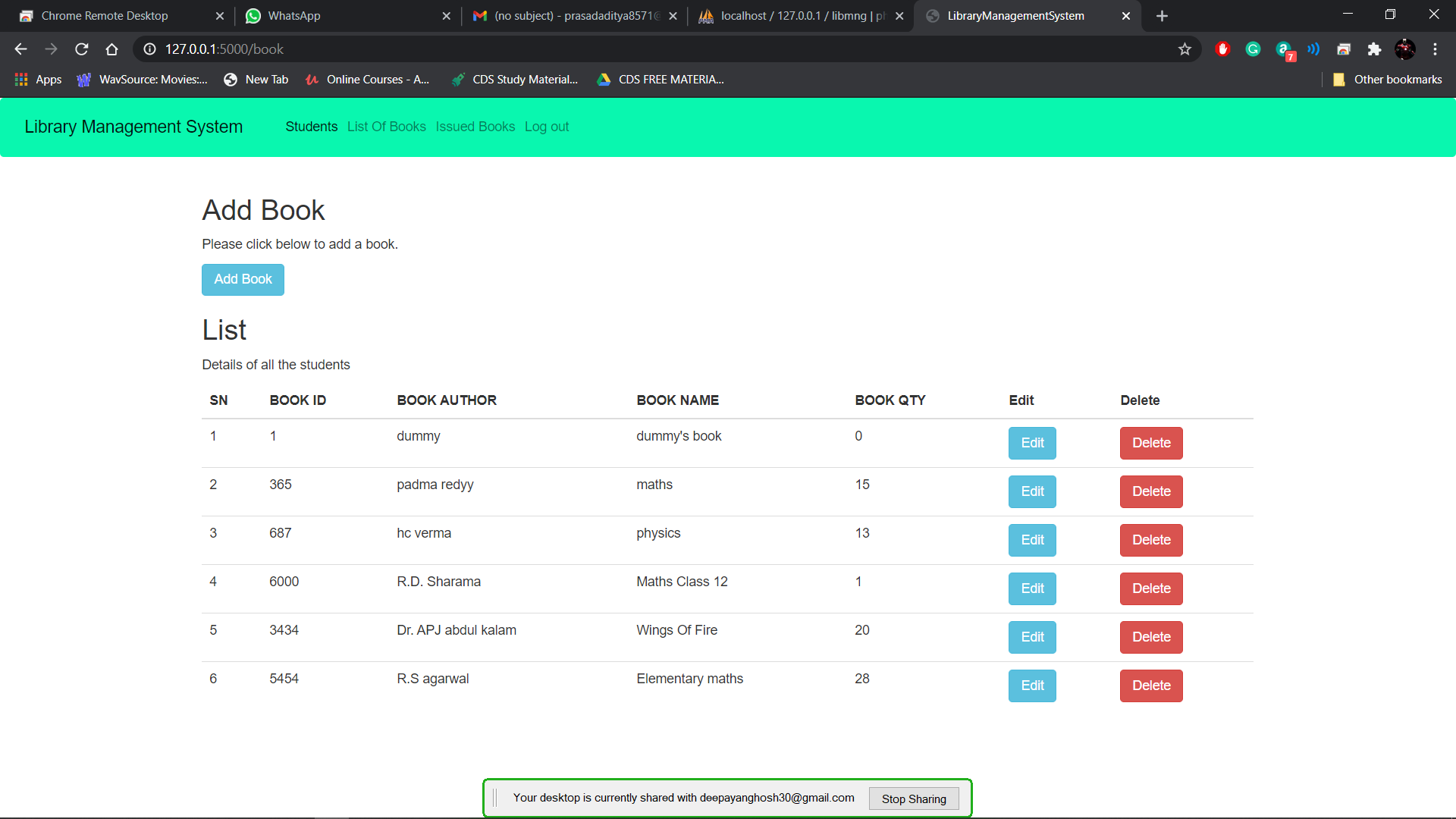
The Admin can select the library table using the respective SQL query commands to display the library usn, name, phone, address, sem, batch, gender,email, dept\_id.



**Figure 6.1.1: Admin Portal**

**6.1.2 Available Books:**

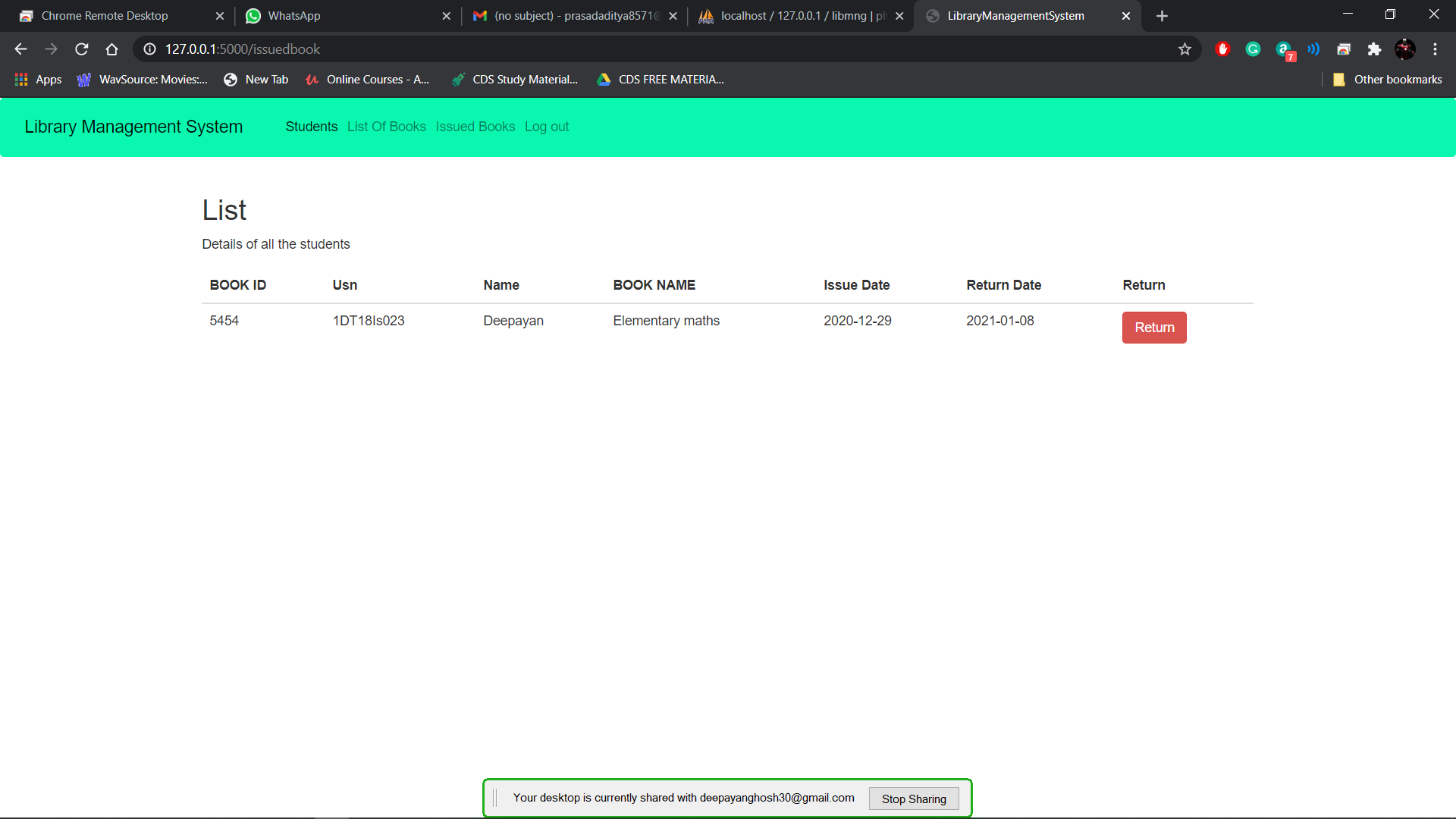
The Admin can view the book details such as name, author, book id, book qty. The admin can add or delete existing books.



**Figure 6.1.2:Book Details**

**6.1.3 Issued Books:**

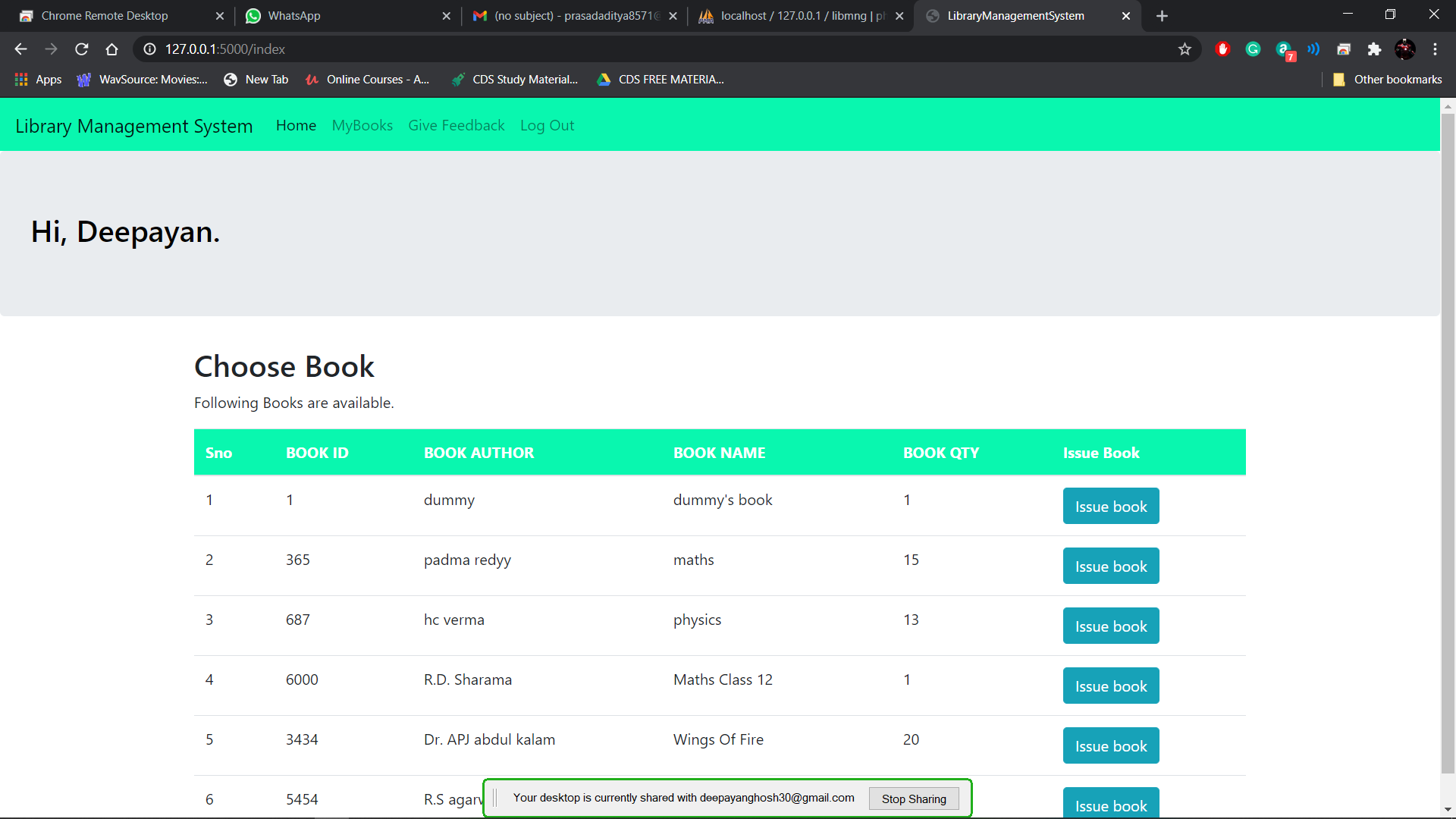
The admin can view the issued books along with student details.



**Figure 6.1.3: Issued Books**

**6.1.4 Student Portal**

The students will be able to issue books, view available books.



**Figure 6.1.4: Library Extra-curricular Activities**

**CHAPTER 7**

**CONCLUSIONS AND FUTURE ENHANCEMENTS**

**7.1 CONCLUSION**

The backend-portal is enables its users to access, manage and update library’s data effectively and efficiently. It allows for a centralized facility that can easily be modified and quickly shared among multiple users. The backend-portal eliminates the paper work which could lead to loss of data and data redundancy. It also allows the possibility of queries to obtain information for various surveys. Due to the many users reading and modifying library data in the department, hence it is an ideal use for such a system.

* 1. **FUTURE ENHANCEMENTS**

The following is just a sample of future opportunities that would help sustain the portal for undergraduates: -

1. By using artificial intelligence the web portal may track the usage of Wi-Fi based on the library’s registered device.
2. One can deploy this web application into mobile android application and be used in smaller devices like mobile phones, tablets and notepads.
3. In future web portal can be combined with the university internal and external web sites. So that all courses in the university will have single web app.
4. The present system may be further upgraded in future even maintain the activities in hostel like hostel mess bill and attendance system even in hostels

**REFERENCES**

1. Reference to MySQL documentation @ [**https://dev.mysql.com/doc/**](https://dev.mysql.com/doc/)
2. Reference to MySQL WorkBench @ [**https://dev.mysql.com/doc/workbench/en/**](https://dev.mysql.com/doc/workbench/en/)
3. Reference to DBMS @ Elmasri-Navathe-Fundamentals-of-Database-Systems-7th-Edition
4. <https://www.phpmyadmin.net/>
5. <https://www.w3schools.com/php/default.asp>
6. https://www.w3schools.com/html/default.asp
7. https://www.w3schools.com/css/default.asp
8. https://www.w3schools.com/sql/default.asp