Visvesvaraya Technological University Belagavi, Karnataka

R.T.E Society's

Rural Engineering College Hulkoti – 582205

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Mini Project Report On

"LIBRARY MANAGEMENT SYSTEM"

Under the Guidance of

Mrs. Geeta M. Patil

Project Associates:

POORNIMA BADIGER	**2RH23CS408**
VANITA JADHAV	**2RH23CS411**
TARANNUM DODDAMANI	**2RH22CS083**
KOUSTUBH KATTI	**2RH23CS404**

R.T.E Society's Rural Engineering College, Hulkoti – 582205
Department of Computer Science & Engineering

2023-2024

CERTIFICATE

Certified that mini project work entitled

"LIBRARY MANAGEMENT SYSTEM"

Carried out by

Miss. **POORNIMA BADIGER** bearing USN 2RH23CS408, Miss. **VANITA JADHAV** bearing USN 2RH23CS411, Miss. **TARANNUM DODDAMANI** bearing USN 2RH22CS083, Mr. **KOUSTUBH KATTI** bearing USN 2RH23CS404 are bonafide student of Department of Computer Science and Engineering, in partial fulfillment for the award of Degree of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belagavi during the year 2023-24. It is certified that all corrections/suggestions indicated for internal assessments have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of mini project work prescribed for the said degree.

Signature of the Guide	**Signature of the HOD**	**Signature of the Principal**
Mrs. Geeta M. Patil	Dr.S.H Angadi	Dr.V.M. Patil

ACKNOWLEDGEMENT

The knowledge and satisfaction that accompanies a successful completion of a project phase1 is hard to describe. Behind any successful project there are wise people guiding throughout. We thank them for guiding us, correcting our mistakes, and providing valuable feedback. We would consider it as our privilege to express our gratitude and respect to all those who guided and encouraged us in this project.

We extend our heartfelt gratitude to our beloved principal Dr. V. M. Patil, REC Hulkoti, for the success of this project.

We are grateful to Dr. S H Angadi, Head of CSE Department Rural Engineering College Hulkoti, for providing support and encouragement.

We convey our sincerest regards to our project guide, Mrs. Geeta M. Patil, Dept. of CSE, Rural Engineering College Hulkoti, for providing guidance and encouragement at all times needed.

PROJECT ASSOCIATES

POORNIMA BADIGER	**2RH23CS408**
VANITA JA DHAV	**2RH23CS411**
TARANNUM DODDAMANI	**2RH22CS083**
KOUSTUB KATTI	**2RH23CS404**

ABSTRACT

The Library Management System (LMS) is designed to manage and streamline the operations of a library. This system provides a comprehensive solution for maintaining and managing library resources, including books, journals, and digital media. The primary objective of the Library Management System is to facilitate the effective and efficient management of library operations, enhance user experience, and support administrative functions. The Library Management System (LMS) is a comprehensive solution designed to manage and streamline library operations using a Database Management System (DBMS). It encompasses user management, catalog management, search and retrieval, circulation management, inventory management, reporting, and notifications. By automating routine tasks and providing easy access to resources, the LMS improves efficiency, enhances user experience, and optimizes resource utilization. The system also offers data-driven insights for informed decision-making, making it an essential tool for modern library management.

CONTENTS

No	Chapters	Page No
1.	INTRODUCTION	1
2.	OBJECTIVES	2-3
3.	SOFTWARE TOOLS USED	4-7
4.	THE PROJECT DETAILS, MODULES	8
5.	ENTITY RELATIONSHIP DIAGRAM, SCHEMA DIAGRAM	9
6.	CREATING DATABSE USING MYSQL	10
7.	OUTPUT OF CREATED TABLES	11-13
8.	SOURCE CODE	14-20
9.	OUTPUT OF THE LIBRARY MANAGEMENT SYSTEM	21-24
10.	CONCLUSION	25
11.	REFERENCES	26

CHAPTER-1

INTRODUCTION

The Library Management System is a software solution that helps manage the daily operations of a library. It is designed to automate routine tasks, such as cataloguing, circulation, acquisitions, and inventory management, which enables librarians to focus on providing better services to their patrons.

Many formalities come under library management. A Library Management System can bring several benefits here. For growing knowledge and enhancing skills to solve real-life problems I developed a Library Management System Project in Python using Tkinter and MySQL.

It will help to manage the following tasks:

1. Add new books to the database.
2. Issue books to the users or students.
3. Take borrowed books from the users or students.
4. Issue the book again to the person who is already holding that book.
5. Displays all the book records presented in the library with the total available piece of each book (It shows all the records in a table view format).
6. Search a book by name. In this case, if the users don’t know the full name of the book, they can search by only a word or characters instead.
7. Update and delete the book records.
8. Displays the records of all the borrowers with the related information.

CHAPTER-2

OBJECTIVES OF LIBRARY MANAGEMENT SYSTEM

The main objective of the Project on Library Management System is to manage the details of Student, Books, Issues, Librarian, Member. It manages all the information about Student, Address of Member as well as Student. The project is totally built at administrative end and thus only the administrator is guaranteed the access. The purpose of the project is to build a web-based application program to reduce the manual work for managing the Student, Books, Address, Issues. It tracks all the details about the Issues, Librarian, Member, etc.

1. The objectives 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.
2. The activities of book purchasing, cataloging, 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
3. The library management system software helps in reducing operational costs. Managing a library manually is labor intensive and an immense amount of paperwork is involved. An automated system reduces the need for manpower and stationery. This leads to lower operational costs.
4. 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.
5. Adding, removing or editing the database is a simple process. Adding new members or cancelling existing memberships can be done with ease.
6. Stock checking and verification of books in the library can be done within a few hours. The automated system saves a considerable amount of time as opposed to the manual system.
7. The library management system software makes the library a smart one by organizing the books systematically by author, title and subject. This enables users to search for books quickly and effortlessly.
8. Students need access to authentic information. An advanced organized library is an integral part of any educational institution.
9. In this digital age a web-based library management system would be ideal for students who can access the library's database on their smartphones.

10. The main objective of the Project of Library Management System is to manage the details of users as well as books.
11. It also manages all the information about Student, Address of Member as well as Student

PURPOSE OF LIBRARY MANAGEMENT SYSTEM

The Library Database Management System is a comprehensive system that shows all available books and their count, as well as books taken by people, the date they borrowed a particular book, the expected return date, late fees, membership details, and more. Using a database for library management system ensures that everything is crystal clear, with no ambiguity. This clarity is beneficial for both students and librarians.

This system is very efficient and cost-effective. Implementing a database for library management system saves a lot of time for both librarians and students. With this system, manual work is significantly reduced, requiring less staff and maintenance. Additionally, the system is user-friendly and very easy to use.

CHAPTER-3

SOFTWARE TOOLS USED

The whole Project is divided in two parts the front end and the back end.

FRONT END

HTML

- HTML is an acronym which stands for Hyper Text Markup Language which is used for creating web pages and web applications. Let's see what is meant by Hypertext Markup Language, and Web page.
- Hyper Text: HyperText simply means "Text within Text." A text has a link within it, is a hypertext. Whenever you click on a link which brings you to a new webpage, you have clicked on a hypertext. HyperText is a way to link two or more web pages (HTML documents) with each other.
- Markup language: A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic. It can turn text into images, tables, links, etc.
- Web Page: A web page is a document which is commonly written in HTML and translated by a web browser. A web page can be identified by entering an URL. A Web page can be of the static or dynamic type. With the help of HTML only, we can create static web pages.

CSS

- CSS stands for Cascading style sheets. It describes to the user how to display HTML elements on the screen in a proper format. CSS is the language that is used to style HTML documents. In simple words, cascading style sheets are a language used to simplify the process of making a webpage.

- CSS is used to handle some parts of the webpage. With the help of CSS, we can control the colour of text and style of fonts, and we can control the spacing between the paragraph and many more things. CSS is easy to understand but provides strong control on the Html documents.CSS is combined with HTML.

BACK END

PYTHON

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python installers for the Windows platform usually include the entire standard library and often also include many additional components. For Unix-like operating systems Python is normally provided as a collection of packages, so it may be necessary to use the packaging tools provided with the operating system to obtain some or all of the optional components. Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features such as list comprehension, cycle detecting garbage collection, reference counting, and Unicode support. Python 3.0, 20 released in 2008, was a major revision that is not completely backward-compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020. Python consistently ranks as one of the most popular programming languages.

MYSQL

MySQL MySQL is a Relational Database Management System (RDBMS) developed by Oracle that is based on Structured Query Language (SQL). MySQL is one of the most recognizable technologies in the modern big data ecosystem. Often called the most popular database and currently enjoying widespread, effective use regardless of industry, it's clear that anyone involved with enterprise data or general IT should at least aim for a basic familiarity of MySQL. With MySQL, even those new to relational systems can immediately build fast, powerful, and secure data storage systems. MySQL's programmatic syntax and interfaces are also perfect gateways into the wide world of other popular query languages and structured data stores. A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or a place to hold the vast amounts of information in a corporate network. In particular, a relational database is a digital store collecting data and organizing it according to the relational model. In this model, tables consist of rows and columns, and relationships between data elements all follow a strict logical structure. An RDBMS is simply the set of software tools used to actually implement, manage, and query such a database. MySQL is integral to many of the most popular software stacks for building and maintaining everything from customer-facing web applications to powerful, data driven B2B services. Its open-source nature, stability, and rich feature set, paired with ongoing development and support from Oracle, have meant that internet-critical organizations such as Facebook, Twitter, Wikipedia, and YouTube all employ MySQL backends.

HARDWARE

Hardware is a term that refers to all the physical parts that make up a computer. The internal hardware devices that make up the computer. Various devices which are essentials to form a hardware is called as components. Following are the hardware specifications that is required to develop this project is as follows:

1. Computer components like Monitor, Keyboard, Mouse, CPU, Keyboard.
2. Minimum 1 GB ram for smooth working of application.
3. 250 GB Hard Disk or More. CD ROM Drive.

SOFTWARE

Computer software, or simply software, is a collection of data or computer instructions that tell the computer how to work. This is in contrast to physical hardware, from which the system is build and actually performs the work.

- Front End- HTML, CSS
- Back End- PYTHON, MySQL Workbench
- Text Editors- Python IDE
- Operating System-Windows11

CHAPTER-4

THE PROJECT DETAILS

The graphical interface of this project is managed by the Tkinter library. The PyMySQL package is used here to manage database operations using Python. The project folder contains three Python files, 'main.py', 'custom.py', and 'credentials.py'.

As the name of 'main.py', it handles all the tasks related to library management. 'custom.py' has several information about the colors and fonts used in the main program (main.py). 'credentials.py' contains credentials to log in to the MySQL Server. Here, users need to enter their own Username and Password for the MySQL server.

As we'll be interacting with the database, it's essential to have MySQL Server installed on your system (You can find instructions on how to install MySQL on Windows, Linux, and Mac). If you've already completed this step, there's no need to do it again.

NUMBER OF MODULES AND PROCESS LOGIC

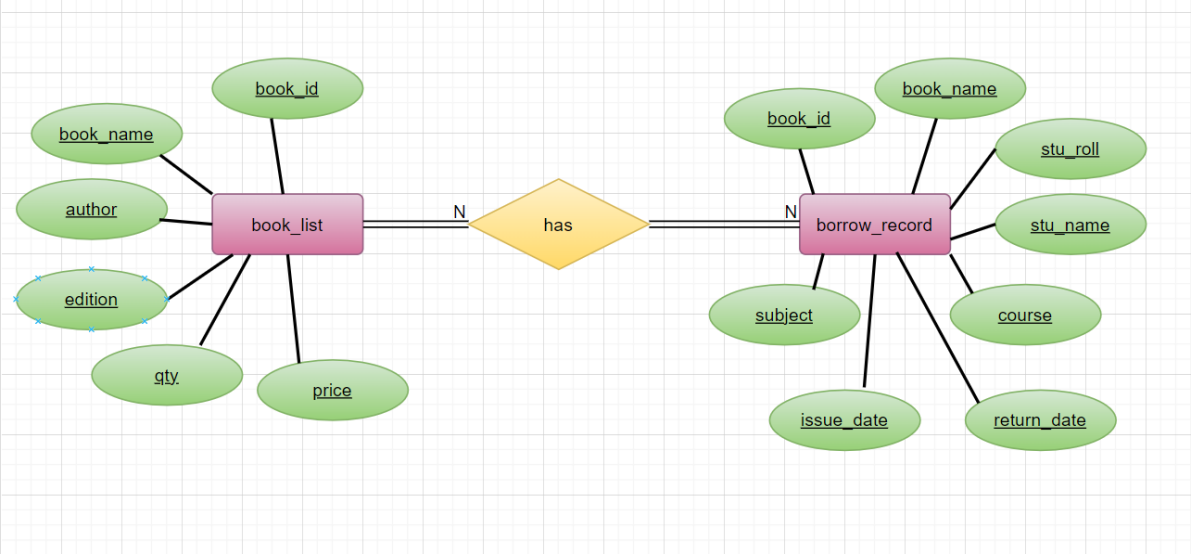
- **Modules used in Project:**

```
from tkinter import*
import tkinter
from tkinter import ttk
import customs as cs
import pymysql
from tkinter import messagebox
import random
import time
import datetime
import mysql.connector
```

CHAPTER-5

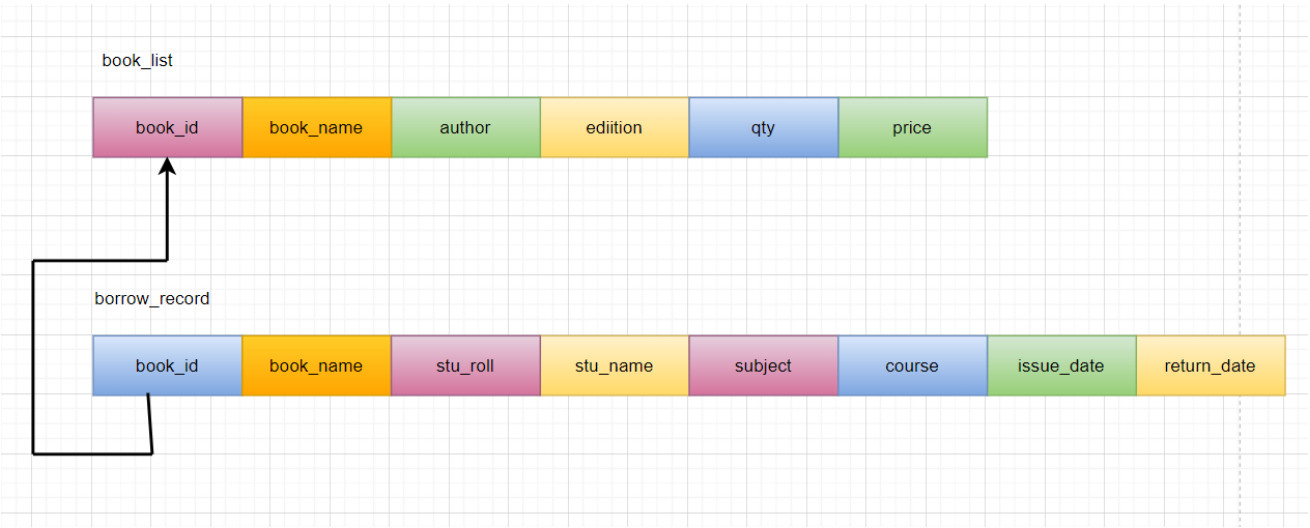
ENTITY RELATIONSHIP DIAGRAM

- ER model stands for an Entity-Relationship model. It is a high-level data model. This model is used to define the data elements and relationship for a specified system.
- It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.
- In ER modeling, the database structure is portrayed as a diagram called an entity-relationship diagram.



SCHEMA DIAGRAM

A database schema is a structure that represents the logical storage of the data in a database. It represents the organization of data and provides information about the relationships between the tables in a given database.



CHAPTER-6

CREATING DATABSE USING MYSQL

Create a Database and Two Tables

Create a Database with this name: library_management.

```
create database library_management;
```

Create a table book_list under the library_management database.

```
create table book_list(
        book_id VARCHAR(10) NOT NULL,
        book_name VARCHAR(50) NOT NULL,
        author VARCHAR(50) NOT NULL,
        edition VARCHAR(10) NOT NULL,
        price Int(6) NOT NULL,
        qty Int(4) NOT NULL,
        PRIMARY KEY ( book_id )
);
```

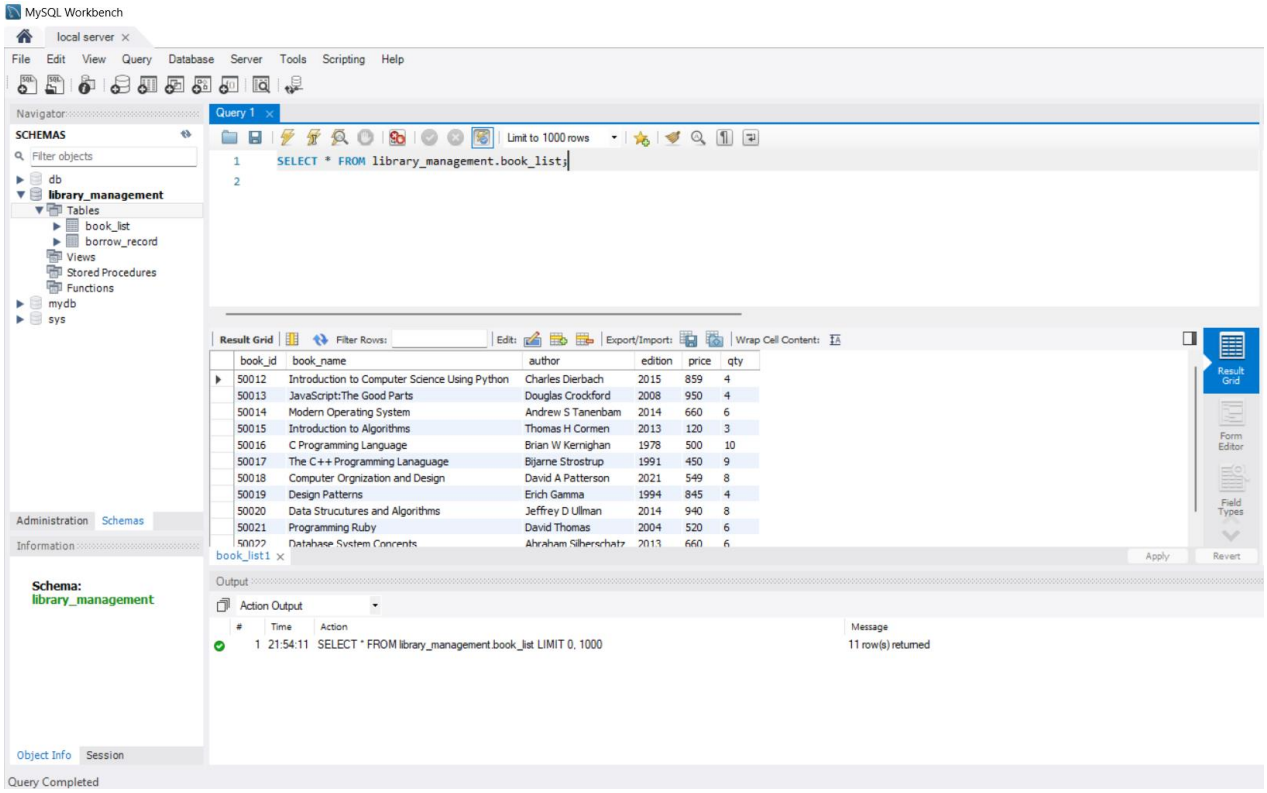
Create a table borrow_record under the same database.

```
create table borrow_record(
        book_id VARCHAR(10) NOT NULL,
        book_name VARCHAR(50) NOT NULL,
        stu_roll VARCHAR(15) NOT NULL,
        stu_name VARCHAR(50) NOT NULL,
        course VARCHAR(10) NOT NULL,
        subject VARCHAR(30) NOT NULL,
        issue_date date NOT NULL,
        return_date date NOT NULL
);
```

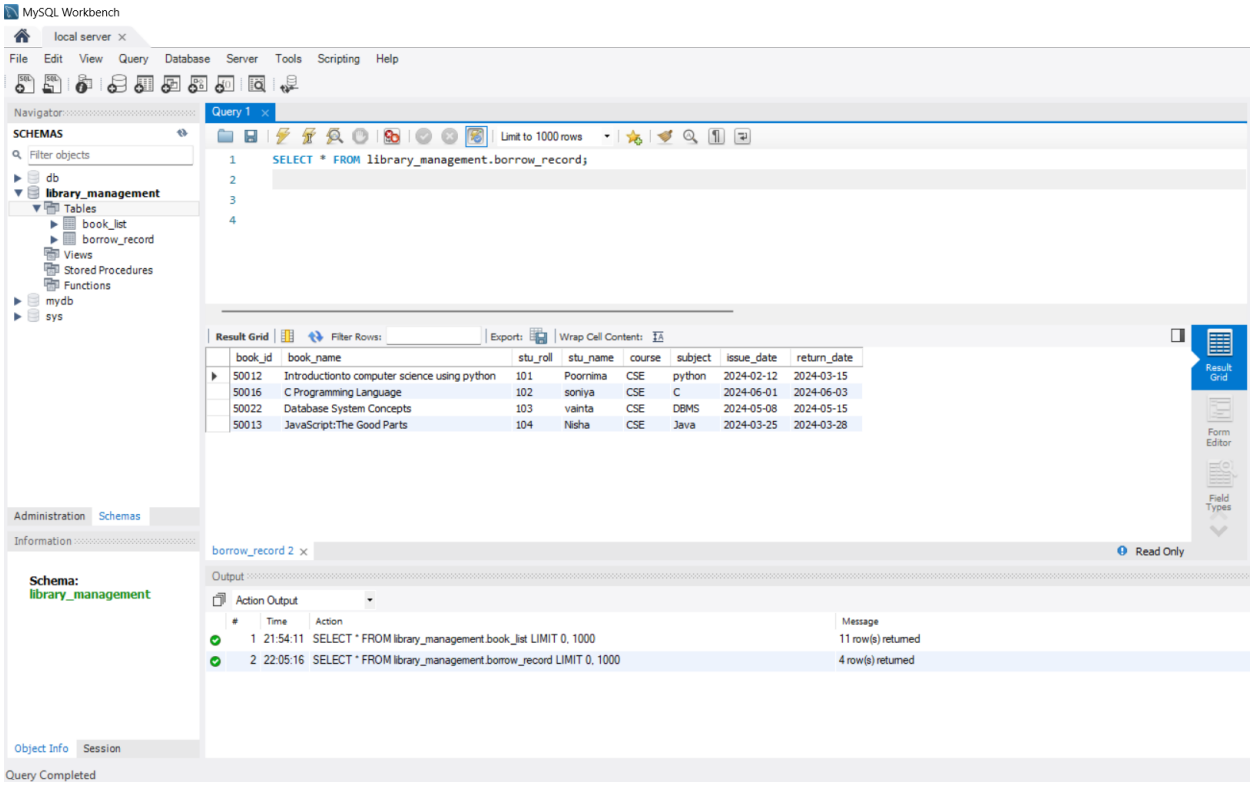
CHAPTER-7

OUTPUT OF CREATED TABLES

BOOK_LIST



BORROW_RECORD



CREATE A DEDICATED MYSQL USER AND GRANT PRIVILEGES

I suggest to create a specific MySQL user exclusively for this project and grant the necessary privileges. Please perform the following steps:

1. Log in to MySQL:

You'll need to have administrative access to your MySQL server. You can log in using the MySQL command-line client:

mysql -u root -p

Replace root with your MySQL username if it's different, and you'll be prompted for the MySQL root password.

2. Create a New User:

To create a new MySQL user, you can use the `CREATE USER` command. Replace `username` and `password` with your desired username and password:

CREATE USER 'username'@'localhost' IDENTIFIED BY 'password';

This creates a user called `username` that can only connect from the local machine. If you want to allow connections from any host, replace `localhost` with `%`.

3. Grant Privileges:

You can use the `GRANT` statement to assign privileges to the user. Here's how to grant various privileges:

Grant all privileges on a specific database:

GRANT ALL PRIVILEGES ON database_name.* TO 'username'@'localhost';

Grant specific privileges (e.g., GRANT CREATE, ALTER, DROP, INSERT, UPDATE, INDEX, DELETE, SELECT, REFERENCES, RELOAD) on a specific database:

GRANT SELECT, INSERT, UPDATE, DELETE ON database_name.* TO 'username'@'localhost';

Replace `database_name` with the name of the database you want to grant access to.

Grant all privileges on all databases (not recommended for security reasons):

GRANT ALL PRIVILEGES ON *.* TO 'username'@'localhost';

4. Apply Privileges:

After granting privileges, you need to apply the changes using the FLUSH PRIVILEGES command:

FLUSH PRIVILEGES;

5. Exit MySQL:

Now you can exit the MySQL Client:

exit;

REQUIREMENTS AND INSTALLATION

Make sure that you have Python installed on your system. You will also need to install PyMySQL library for connecting to and interacting with MySQL databases.

pip install PyMySQL

Note that, by default Tkinter comes with most Python installations, so you likely won't need to install it separately. However, if you don't have it, you can install it using pip:

pip install tk

CHAPTER-8

SOURCE CODE

#Delete the book

```
# delete the book
    def DeleteBook(self):
        x = self.tree.selection()
        row = self.tree.item(x)['values']
        try:
            status = messagebox.askokcancel('Delete Book', 'Are you want to proceed?')
            if status == True:
                connection = pymysql.connect(host=cr.host, user=cr.user, password=cr.password,
database=cr.database)
                curs = connection.cursor()

                curs.execute("select * from borrow_record where book_id=%s", row[0])
                var = curs.fetchall()

                if len(var) != 0:
                    messagebox.showwarning("Critical Warning!", "You can't delete this book
record")
                else:
                    curs.execute("delete from book_list where book_id=%s",
                    (
                        row[0]
                    ))
                    messagebox.showinfo("Success!", "The book record has been deleted")
                    connection.commit()
                    connection.close()
                    self.ClearScreen()
                    self.ShowBooks()
        except Exception as e:
            messagebox.showerror("Error!",f"Error due to {str(e)}",parent=self.window)
```

 # Return of Book

```
  def ReturningBook(self):
      x = self.tree_1.selection()
      row = self.tree_1.item(x)['values']

      try:
          status = messagebox.askokcancel('Returning Book', 'Are you want to proceed?')
          if status == True:
              connection = pymysql.connect(host=cr.host, user=cr.user, password=cr.password,
database=cr.database)
              curs = connection.cursor()
              curs.execute("delete from borrow_record where book_id=%s",
              (
                  row[0]
              ))
              curs.execute("select * from book_list where book_id=%s", row[0])
              var = curs.fetchone()

              book_count = var[5]
              book_count += 1

              curs.execute("update book_list set qty=%s where book_id=%s", (book_count,
row[0]))

              messagebox.showinfo("Success!", "Thanks for returning the book!")
              connection.commit()
              connection.close()
              self.ClearScreen()
      except Exception as e:
          messagebox.showerror("Error!",f"Error due to {str(e)}",parent=self.window)
```

#Issue of Book

```
def IssueAgain(self):
      x = self.tree_1.selection()
      row = self.tree_1.item(x)['values']
```

```
        self.ClearScreen()

        book_id = Label(self.frame_1, text="Book Id", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=130,y=30)
        id = Label(self.frame_1, text=row[0], font=(cs.font_1, 10))
        id.place(x=130,y=60, width=200)

        book_name = Label(self.frame_1, text="Book Name", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=400,y=30)
        bookname = Label(self.frame_1, text=row[1], font=(cs.font_1, 10))
        bookname.place(x=400,y=60, width=200)

        student_roll = Label(self.frame_1, text="Student Roll", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=130,y=100)
        sturoll = Label(self.frame_1, text=row[2], font=(cs.font_1, 10))
        sturoll.place(x=130,y=130, width=200)

        student_name = Label(self.frame_1, text="Student Name", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=400,y=100)
        stuname = Label(self.frame_1, text=row[3], font=(cs.font_1, 10))
        stuname.place(x=400,y=130, width=200)

        course = Label(self.frame_1, text="Course", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=130,y=170)
        Course = Label(self.frame_1, text=row[4], font=(cs.font_1, 10))
        Course.place(x=130,y=200, width=200)

        subject = Label(self.frame_1, text="Subject", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=400,y=170)
        Subject = Label(self.frame_1, text=row[5], font=(cs.font_1, 10))
        Subject.place(x=400,y=200, width=200)

        issue_date = Label(self.frame_1, text="Issue Date", font=(cs.font_2, 15, "bold"),
```

```
bg=cs.color_1).place(x=130,y=240)
        issuedate = Label(self.frame_1, text=row[6], font=(cs.font_1, 10))
        issuedate.place(x=130,y=270, width=200)

        return_date = Label(self.frame_1, text="Return Date", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=400,y=240)
        self.return_date_entry = Entry(self.frame_1, bg=cs.color_4, fg=cs.color_3)
        self.return_date_entry.insert(0, row[7])
        self.return_date_entry.place(x=400,y=270, width=200)

        self.submit_bt = Button(self.frame_1, text='Submit', font=(cs.font_1, 12), bd=2,
command=partial(self.BorrowBookAgain, row),cursor="hand2",
bg=cs.color_2,fg=cs.color_3).place(x=300,y=320,width=100)
```

 # Borrow of Book

```
def BorrowBookAgain(self, row):
        try:
            connection = pymysql.connect(host=cr.host, user=cr.user, password=cr.password,
database=cr.database)
            curs = connection.cursor()
            curs.execute("update borrow_record set return_date=%s where stu_roll=%s and
book_id=%s",
            (
                self.return_date_entry.get(),
                row[2],
                row[0]
            ))
            messagebox.showinfo("Success!", "The book is issued again")
            connection.commit()
            connection.close()
            self.ClearScreen()
        except Exception as e:
            messagebox.showerror("Error!",f"Error due to {str(e)}",parent=self.window)
```

 # Submit for Update Book

```
def SubmitforUpdateBook(self, row):
    try:
        connection = pymysql.connect(host=cr.host, user=cr.user, password=cr.password,
database=cr.database)
        curs = connection.cursor()
        curs.execute("update book_list set
book_name=%s,author=%s,edition=%s,price=%s,qty=%s where book_id=%s",
        (
            self.bookname_entry.get(),
            self.author_entry.get(),
            self.edition_entry.get(),
            self.price_entry.get(),
            self.qty_entry.get(),
            row[0]
        ))
        messagebox.showinfo("Success!", "The data has been updated")
        connection.commit()
        connection.close()
        self.ClearScreen()
    except Exception as e:
        messagebox.showerror("Error!",f"Error due to {str(e)}",parent=self.window)
```

 # Search of Book

```
def SearchBook(self):
    if self.book_entry.get() == "":
        messagebox.showerror("Error!", "Please Enter the Book Name")
    else:
        try:
            connection = pymysql.connect(host=cr.host, user=cr.user, password=cr.password,
database=cr.database)
            curs = connection.cursor()
            curs.execute("select * from book_list where book_name like %s", ("%" +
self.book_entry.get() + "%"))
            rows=curs.fetchall()
```

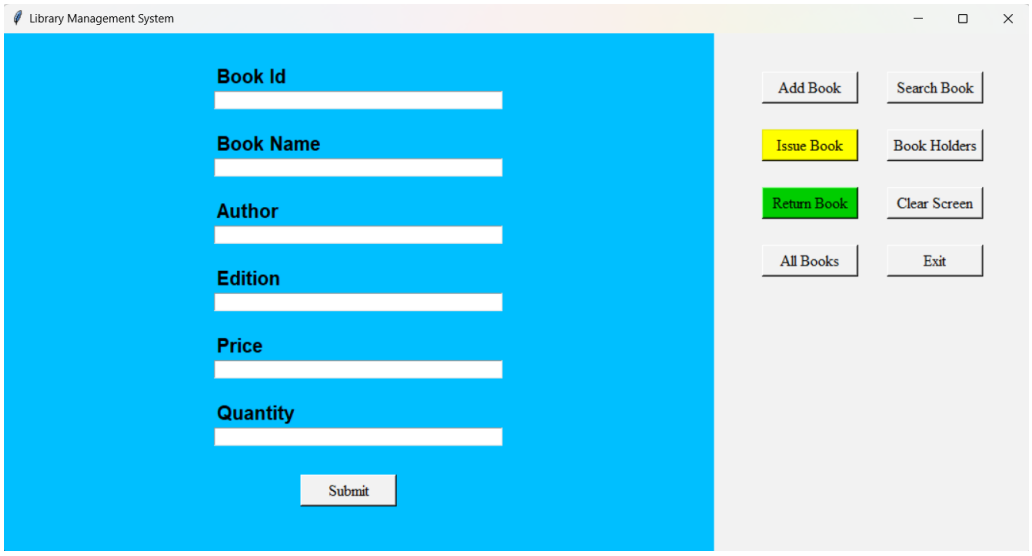
```
            if rows == None:
                messagebox.showinfo("Database Empty","There is no data to
show",parent=self.window)
                connection.close()
                self.ClearScreen()
            else:
                connection.close()
        except Exception as e:
            messagebox.showerror("Error!",f"Error due to {str(e)}",parent=self.window)
```

 # Add New Book

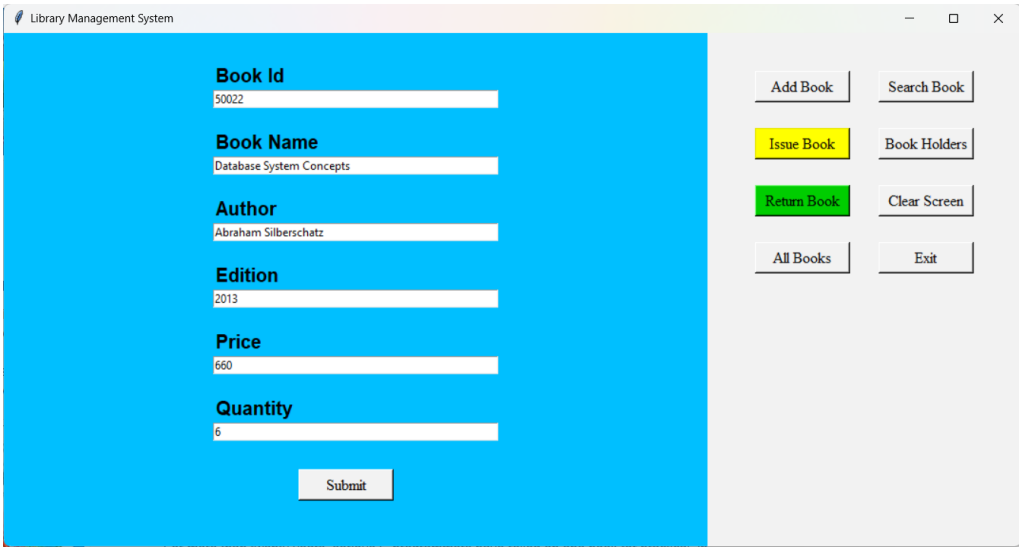
```
  def AddNewBook(self):
      self.ClearScreen()

      book_id = Label(self.frame_1, text="Book Id", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=220,y=30)
      self.id_entry = Entry(self.frame_1, bg=cs.color_4, fg=cs.color_3)
      self.id_entry.place(x=220,y=60, width=300)

      book_name = Label(self.frame_1, text="Book Name", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=220,y=100)
      self.bookname_entry = Entry(self.frame_1, bg=cs.color_4, fg=cs.color_3)
      self.bookname_entry.place(x=220,y=130, width=300)

      author = Label(self.frame_1, text="Author", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=220,y=170)
      self.author_entry = Entry(self.frame_1, bg=cs.color_4, fg=cs.color_3)
      self.author_entry.place(x=220,y=200, width=300)

      edition = Label(self.frame_1, text="Edition", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=220,y=240)
      self.edition_entry = Entry(self.frame_1, bg=cs.color_4, fg=cs.color_3)
      self.edition_entry.place(x=220,y=270, width=300)

      price = Label(self.frame_1, text="Price", font=(cs.font_2, 15, "bold"),
```

```
bg=cs.color_1).place(x=220,y=310)
        self.price_entry = Entry(self.frame_1, bg=cs.color_4, fg=cs.color_3)
        self.price_entry.place(x=220,y=340, width=300)

        quantity = Label(self.frame_1, text="Quantity", font=(cs.font_2, 15, "bold"),
bg=cs.color_1).place(x=220,y=380)
        self.qty_entry = Entry(self.frame_1, bg=cs.color_4, fg=cs.color_3)
        self.qty_entry.place(x=220,y=410, width=300)

        self.submit_bt_1 = Button(self.frame_1, text='Submit', font=(cs.font_1, 12), bd=2,
command=self.Submit,cursor="hand2",
bg=cs.color_2,fg=cs.color_3).place(x=310,y=459,width=100)
```

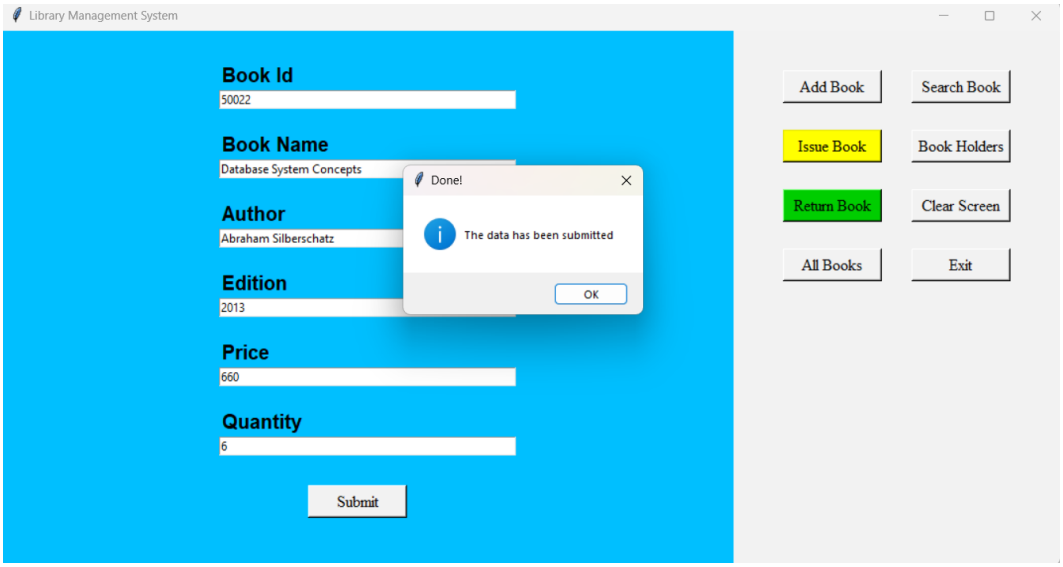
OUTPUT OF THE LIBRARY MANAGEMENT SYSTEM



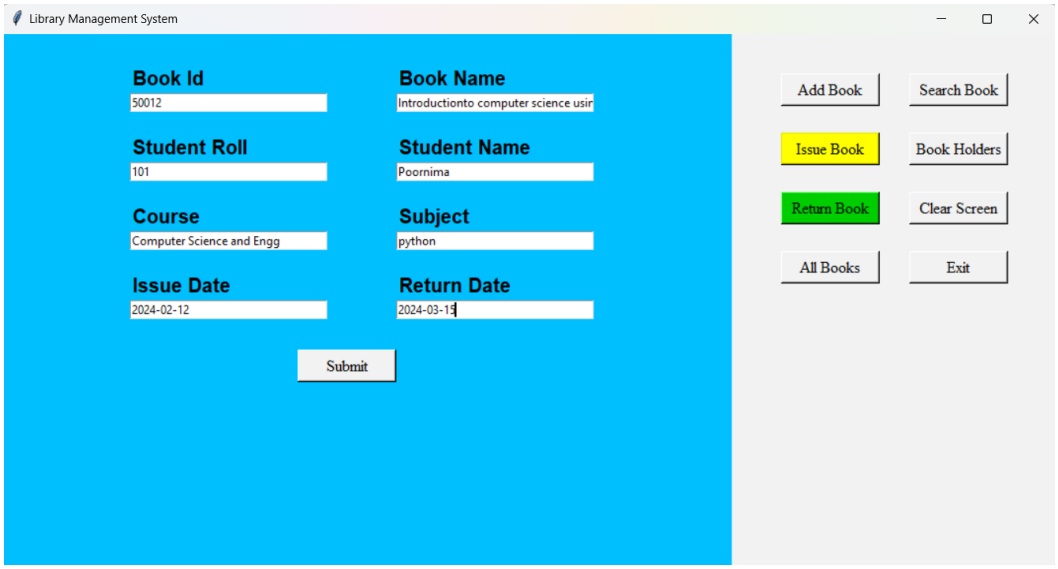
Adding Books recode in to the Database and Submit.



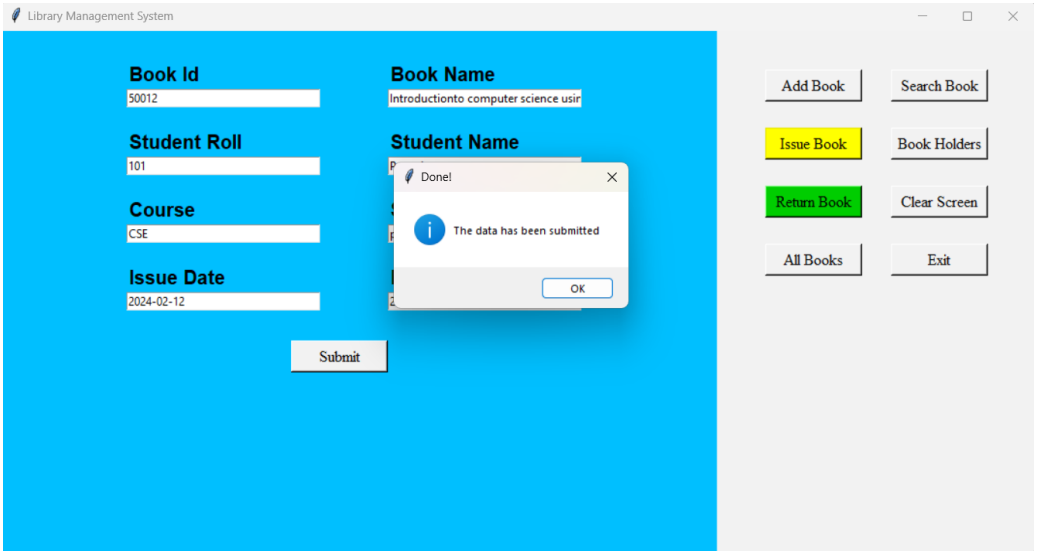
Here the data has been submitted.



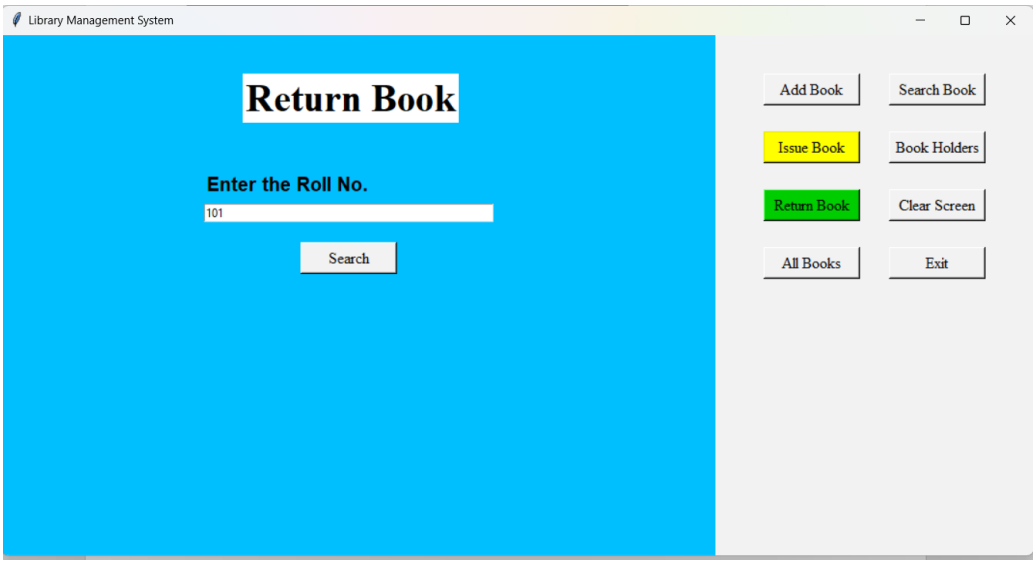
Inserting the recode of Issue Book to the Students.



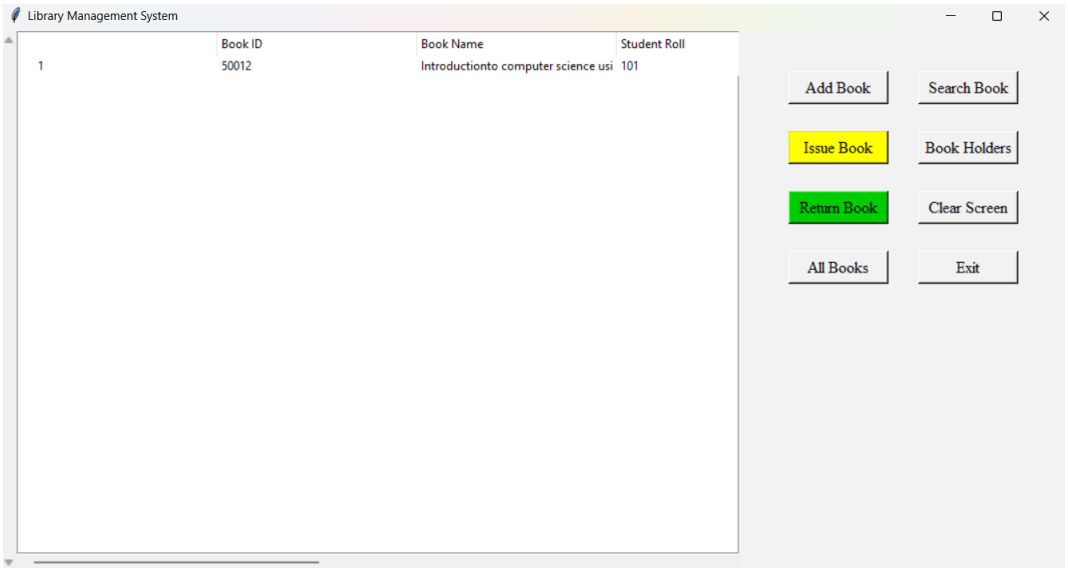
Here the data has been submitted.



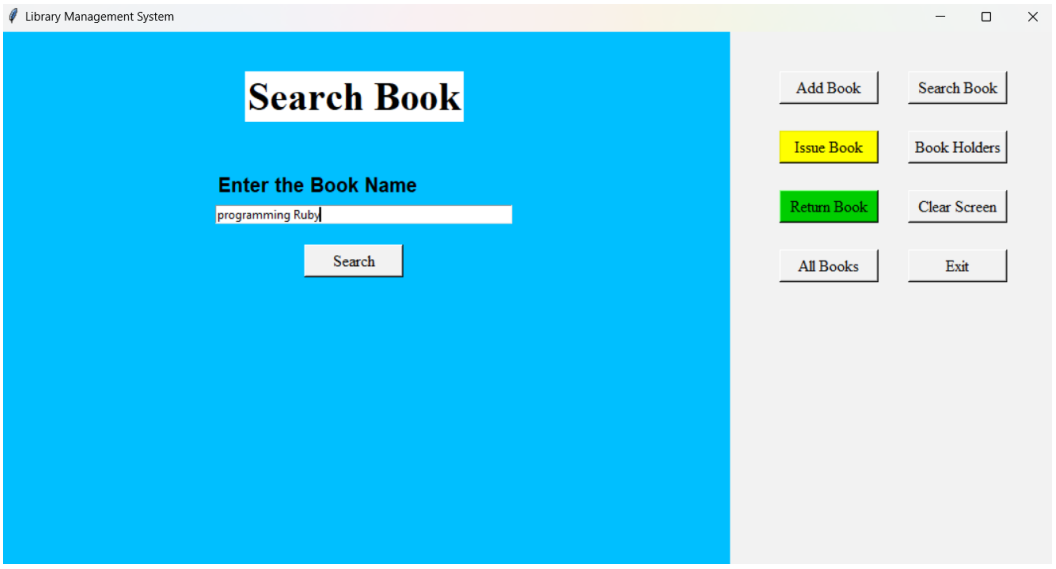
Searching for Return Book details of Students.



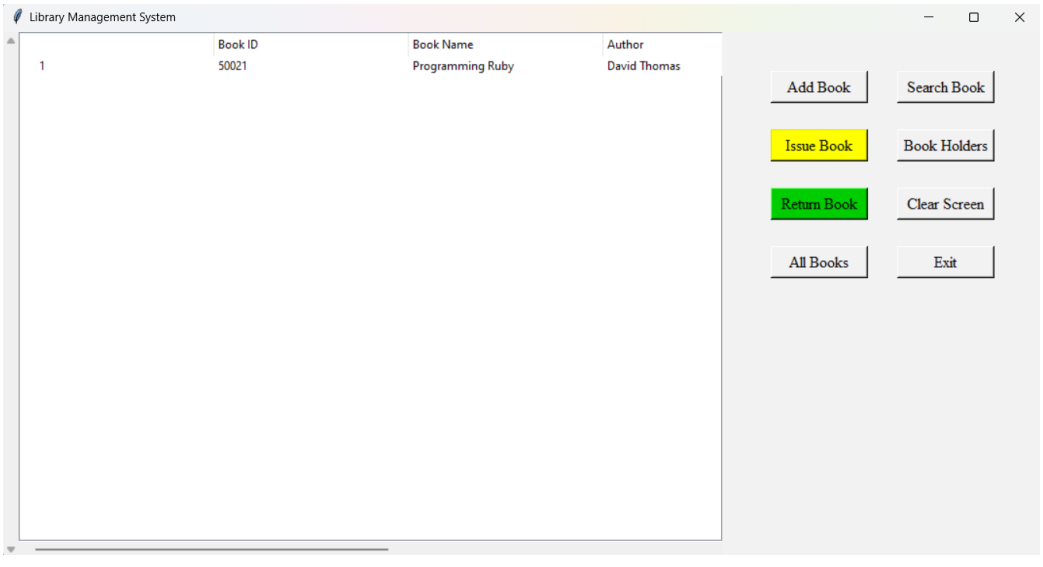
Here it's show whether the book is return or not.



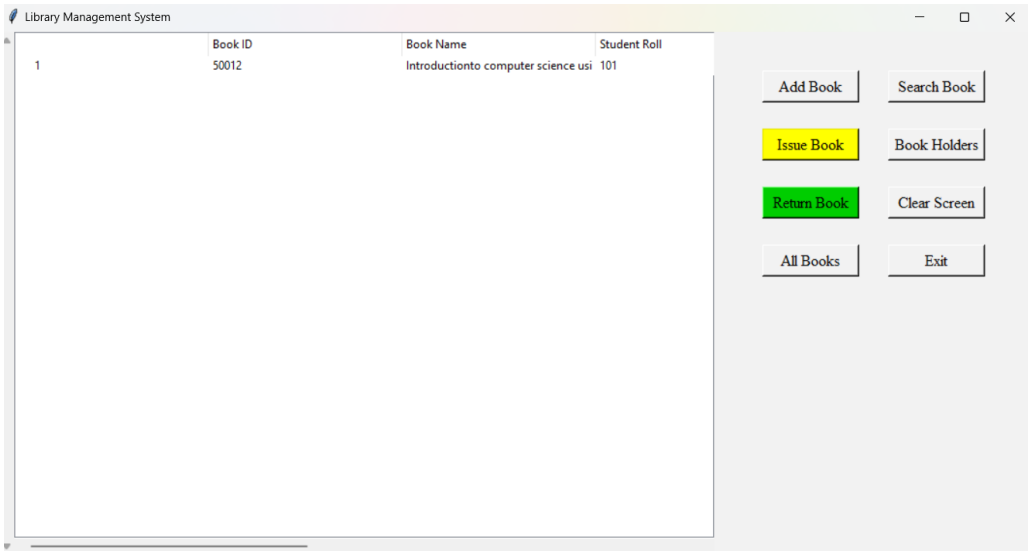
Searching book with book name



Here it's shown the searched book.



Here it's shown the Book Holders with (Student Roll).



Here we use to clear screen by (Clear Screen Button).

Here we can Exit by (Exit Button).

CONCLUSION

- we discussed a library management system project in Python. Here we have handled graphical interface and database-related tasks using Tkinter and MySQL respectively.
- In conclusion, a library management system can be a valuable tool for libraries looking to improve their operations, enhance their services, and meet the changing needs of their users
- The Library Management System project implemented using Python GUI provides a user-friendly interface for managing library tasks efficiently. With the use of the Tkinter GUI library and CSV module, this project provides an easy way to store and manage book data, borrow/return books, and generate reports.
- This project also serves as an excellent starting point for developers who want to learn more about building GUI applications with Python. By studying this project's source code and experimenting with its features, developers can gain valuable experience in GUI programming, data handling, and project development.
- Overall, the Library Management System project in Python using the GUI approach is an excellent example of how to leverage Python's powerful features to create robust and user-friendly applications.

REFERENCES

https://github.com/smita3199/Library-Management-System-using-Data-Structures

https://www.javatpoint.com/library-management-system-in-c

https://www.codewithc.com/mini-project-in-c-library-management-system/?amp=1

https://www.cppbuzz.com/projects/c/library-management-in-c

https://www.geeksforgeeks.org/library-management-system/