

SMART LIBRARY MANAGEMENT SYSTEM WITH BOOK RECOMMENDATION

A MINI PROJECT REPORT

Submitted by

ABISHEK KHANNA .M .S

ALAGAPPAN .V

ANANTHA KRISHNAN .N

ARJUN .S

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING



**K. RAMAKRISHNAN COLLEGE OF
ENGINEERING (AUTONOMOUS),
SAMAYAPURAM, TRICHY – 621112**



**ANNA UNIVERSITY
CHENNAI 600 025**

MAY 2024

SMART LIBRARY MANAGEMENT SYSTEM WITH BOOK RECOMMENDATION

UCS1413 UG MINI PROJECT WORK

Submitted by

ABISHEK KHANNA .M .S (8115U22CS003)

ALAGAPPAN .V (8115U22CS005)

ANANTHA KRISHNAN .N (8115U22CS007)

ARJUN .S (8115U22CS012)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Under the Guidance of

Mr. R. Sasikumar, ME., (Ph.D)

Department of Computer Science and Engineering

K. RAMAKRISHNAN COLLEGE OF ENGINEERING

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF ENGINEERING

(AUTONOMOUS)

Under

ANNA UNIVERSITY, CHENNAI





**K. RAMAKRISHNAN COLLEGE OF ENGINEERING
(AUTONOMOUS)
Under
ANNA UNIVERSITY, CHENNAI**



BONAFIDE CERTIFICATE

Certified that this project report titled “**SMART LIBRARY MANAGEMENT SYSTEM WITH BOOK RECOMMENDATION**” is the bonafide work of **ABISHEK KHANNA M S (8115U22CS003), ALAGAPPAN V (8115U22CS005), ANANTHA KRISHNAN N (8115U22CS007)** and **ARJUN S (8115U22CS012)** who carried out the work under my supervision.

Dr. T. M. NITHYA, ME., Ph.D
HEAD OF THE DEPARTMENT
ASSOCIATE PROFESSOR,
Department of Computer Science
& Engineering,
K. Ramakrishnan College of
Engineering, (Autonomous)
Samayapuram, Trichy.

Mr. R. SASIKUMAR, ME., (Ph.D)
SUPERVISOR
ASSISTANT PROFESSOR,
Department of Computer Science
& Engineering,
K. Ramakrishnan College of
Engineering, (Autonomous)
Samayapuram, Trichy.

SIGNATURE OF INTERNAL EXAMINER

NAME:

DATE:

SIGNATURE OF EXTERNAL EXAMINER

NAME:

DATE:

ACKNOWLEDGEMENT

We thank the almighty GOD, without whom it would not have been possible for us to complete our project.

We wish to address our profound gratitude to **Dr.K.RAMAKRISHNAN**, Chairman, K.Ramakrishnan College of Engineering (Autonomous), who encouraged and gave us all help throughout the course.

We express our hearty gratitude and thanks to our honorable and grateful Executive Director **Dr.S.KUPPUSAMY, B.Sc., MBA., Ph.D.**, K.Ramakrishnan College of Engineering (Autonomous).

We are glad to thank our principal **Dr.D.SRINIVASAN, M.E., Ph.D., FIE., MIIW., MISTE., MISAE., C.Engg**, for giving us permission to carry out this project.

We wish to convey our sincere thanks to **Dr.T.M.NITHYA, M.E., Ph.D.**, Head of the Department, Computer Science and Engineering for giving us constants encouragement and advice throughout the course.

We are grateful to **Mr.R.SASIKUMAR, M.E., (Ph.D.)**, Assistant Professor in the Department of Computer Science and Engineering, K.Ramakrishnan College of Engineering (Autonomous), for his guidance and valuable suggestions during the course of study.

Finally, we sincerely acknowledge in no less term for all our faculty members, colleagues, our parents and friends for their cooperation and help at various stages in this project work.



**K. RAMAKRISHNAN COLLEGE OF ENGINEERING
(AUTONOMOUS)
Under
ANNA UNIVERSITY, CHENNAI**



DECLARATION BY THE CANDIDATE

I declare that to the best of my knowledge the work reported here in has been composed solely by myself and that it has not been in whole or in part in any previous application for a degree.

Submitted for the project Viva- Voce held at K. Ramakrishnan College of Engineering
on _____

SIGNATURE OF THE CANDIDATE

ABISHEK KHANNA M S

(8115U22CS003)



**K. RAMAKRISHNAN COLLEGE OF ENGINEERING
(AUTONOMOUS)
Under
ANNA UNIVERSITY, CHENNAI**



DECLARATION BY THE CANDIDATE

I declare that to the best of my knowledge the work reported here in has been composed solely by myself and that it has not been in whole or in part in any previous application for a degree.

Submitted for the project Viva- Voce held at K. Ramakrishnan College of Engineering
on _____

SIGNATURE OF THE CANDIDATE

ALAGAPPAN V

(8115U22CS005)



**K. RAMAKRISHNAN COLLEGE OF ENGINEERING
(AUTONOMOUS)
Under
ANNA UNIVERSITY, CHENNAI**



DECLARATION BY THE CANDIDATE

I declare that to the best of my knowledge the work reported here in has been composed solely by myself and that it has not been in whole or in part in any previous application for a degree.

Submitted for the project Viva- Voce held at K. Ramakrishnan College of Engineering
on _____

SIGNATURE OF THE CANDIDATE

ANANTHA KRISHNAN

(8115U22CS007)



**K. RAMAKRISHNAN COLLEGE OF ENGINEERING
(AUTONOMOUS)
Under
ANNA UNIVERSITY, CHENNAI**



DECLARATION BY THE CANDIDATE

I declare that to the best of my knowledge the work reported here in has been composed solely by myself and that it has not been in whole or in part in any previous application for a degree.

Submitted for the project Viva- Voce held at K. Ramakrishnan College of Engineering
on _____

SIGNATURE OF THE CANDIDATE

ARJUN S

(8115U22CS012)

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	x
	LIST OF ABBREVIATIONS	xi
	LIST OF FIGURES	xii
1.	INTRODUCTION	1
	1.1 INTRODUCTION TO DBMS	1
	1.1.1 Scope of DBMS	2
	1.1.2 Overview of DBMS	3
	1.2 AIM AND OBJECTIVE	4
	1.2.1 Objectives	5
2.	LITERATURE REVIEW	6
3.	SYSTEM ANALYSIS	13
	3.1 EXISTING SYSTEM	13
	3.1.1 Limitations	14
	3.2 PROPOSED SYSTEM	14
	3.2.1 Algorithm	16
	3.2.2 Advantages	19
4.	SYSTEM REQUIREMENTS	20
	4.1 HARDWARE REQUIREMENTS	20
	4.2 SOFTWARE REQUIREMENTS	22
5.	MODULE DESCRIPTION	24
	5.1 SYSTEM INTEGRATION	24
	5.2 DATABASE	25
	5.3 PYTHON	27

	5.4 RECOMMENDATION INTERFACE	28
6.	SYSTEM DESIGN	29
	6.1 ARCHITECTURE DIAGRAM	29
	6.2 COMMUNICATION DIAGRAM	30
	6.3 USE CASE DIAGRAM	31
7.	SYSTEM TESTING	32
	7.1 UNIT TESTING	32
	7.1.1 Recommendation Engine	32
	7.1.2 User Interface	33
	7.1.3 Database Module	33
	7.2 INTEGRATION TESTING	34
	7.2.1 Database and Python Script	34
	7.2.2 Recommendation Engine and User Interface	34
8.	CONCLUSION & FUTURE ENHANCEMENT	35
	8.1 CONCLUSION	35
	8.2 FUTURE ENHANCEMENTS	35
	APPENDIX	38
	APPENDIX A (SOURCE CODES)	37
	APPENDIX B (SCREENSHOTS)	43
	REFERENCE	46
	CERTIFICATES	48

ABSTRACT

In today's digital era, libraries are evolving to better serve their patrons. Our project, Library Management System with Book Recommendation System, integrates modern software technologies to enhance traditional library operations. This innovative system automates book management processes, ensures accurate book placement, and offers personalized recommendations to readers. The primary objective is to streamline the book return process and provide book suggestions based on interests. When a book is scanned using a QR code reader, the system verifies its placement in the database. If correct, it confirms with a positive message; otherwise, it provides the correct rack details for adjustment. Additionally, the system recommends similar books of interest. Our system enhances user experience by automating tasks and offering intelligent recommendations, making library usage more efficient and enjoyable. In conclusion, our Library Management System with Book Recommendation System is a valuable tool for modern libraries, improving accuracy and enriching the reading experience.

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
CPU	Central processing unit
CRUD	Create, Read, Update, and Delete
CRM	Customer relationship management
DDL	Data Definition Language
DML	Data Manipulation Language
DQL	Data Query Language
DBMS	Database Management System
ERP	Enterprise resource planning
GUI	Graphical user interface
IDE	Integrated development environment
LMS	Learning management system
OLTP	Online Transaction Processing
Open CV	Open-Source Computer Vision Library
QR	Quick Response code
RAM	Random-access memory
RFID	Radio-Frequency Identification
SQL	Structured Query Language
UAT	User Acceptance Testing

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
Figure 6.1	Architecture Diagram	29
Figure 6.2	Communication Diagram	30
Figure 6.3	Use Case Diagram	31

CHAPTER 1

INTRODUCTION

In this section, we will provide a foundational understanding of the "Library Management System with Book Recommendation System" project. We will begin with an introduction to Database Management Systems (DBMS), explaining their essential role and functionalities. Next, we will explore the scope of DBMS, highlighting its applications and potential impact on library management.

1.1 INTRODUCTION TO DATABASE MANAGEMENT SYSTEMS

A Database Management System (DBMS) is a software system that facilitates the creation, organization, management, and access of data in a structured format. It serves as an interface between users, applications, and the database itself, providing efficient methods for storing, retrieving, and manipulating data. DBMSs play a crucial role in modern information systems, enabling businesses, organizations, and individuals to effectively manage large volumes of data and derive valuable insights from it.

One of the primary functions of a DBMS is to provide a mechanism for defining the structure of the data, known as a schema, which specifies the organization of data elements and their relationships. This allows users to interact with the database using high-level query languages, such as SQL (Structured Query Language), without needing to understand the underlying storage details. By providing a centralized and structured approach to data management, DBMSs enable organizations to streamline operations, improve efficiency, and gain valuable insights from their data assets, driving innovation and informed decision-making.

1.1.1 SCOPE OF DATABASE MANAGEMENT SYSTEMS

The scope of Database Management Systems (DBMS) is vast and multifaceted, encompassing several key functionalities essential for efficient data management within organizations. At its core, DBMSs provide a centralized platform for storing, organizing, and retrieving data in a structured manner. This includes defining data schemas, creating relationships between different data entities, and enforcing data integrity constraints to ensure the accuracy and consistency of stored information.

Beyond basic data storage and retrieval, DBMSs play a crucial role in ensuring the security and confidentiality of sensitive information. They provide mechanisms for controlling access to data, implementing user authentication and authorization, and encrypting data to prevent unauthorized access or tampering. Additionally, DBMSs support transaction management features to maintain data consistency in the face of concurrent access and update operations. By adhering to the principles of ACID (Atomicity, Consistency, Isolation, Durability), DBMSs ensure that transactions are executed reliably and that data remains in a consistent state even in the event of system failures or interruptions.

Furthermore, the scope of DBMS extends to scalability, performance optimization, and support for advanced data analysis and reporting functionalities. They also integrate with analytical tools and business intelligence platforms, enabling organizations to derive actionable insights from their data through data mining, statistical analysis, and visualization. Overall, the scope of DBMS is integral to the effective management and utilization of data assets within modern organizations, driving efficiency, security, and informed decision-making.

1.1.2 OVERVIEW OF DATABASE MANAGEMENT SYSTEMS

Definition - A Database Management System (DBMS) is a software system designed to facilitate the creation, organization, management, and retrieval of data in a database. It provides a systematic and efficient way to handle data by offering functionalities for defining data structures, storing data, and performing operations such as querying, updating, and deleting data. DBMSs ensure data integrity, security, and consistency while enabling multiple users to access and manipulate the data concurrently.

Key Components are Data Definition Language (DDL), Data Manipulation Language (DML), Data Query Language (DQL), Transaction Management. Data Definition Language (DDL)- DDL is used to define the structure and organization of the data stored in the database. It includes commands for creating, modifying, and deleting database objects such as tables, views, indexes, and constraints. Data Manipulation Language (DML)- DML is used to perform operations on the data stored in the database. It includes commands for inserting, updating, deleting, and querying data within tables. Data Query Language (DQL)-DQL is used to retrieve data from the database based on specific criteria.

The most common DQL is SQL (Structured Query Language), which allows users to formulate complex queries to extract relevant information from the database. Transaction Management- DBMSs support transactions, which are sequences of database operations that are treated as a single unit of work. Transaction management ensures that transactions are executed reliably, adhering to the ACID (Atomicity, Consistency, Isolation, Durability) properties.

Some of the applications are, Enterprise Resource Planning (ERP) Systems- DBMSs are extensively used in ERP systems to store and manage enterprise-wide data related to various business functions such as finance, human resources, supply chain management, and customer relationship management. ERP systems integrate data from multiple departments into a centralized database, facilitating efficient resource planning, decision-making, and business process automation ;

Customer Relationship Management (CRM) Systems - CRM systems leverage DBMSs to store and manage customer-related data, including contact information, purchase history, interactions, and preferences. By centralizing customer data in a database, CRM systems enable organizations to analyze customer behavior, personalize marketing campaigns, and enhance customer satisfaction and loyalty ; Online Transaction Processing (OLTP) Systems - DBMSs power OLTP systems, which are used for processing high volumes of online transactions in real-time. DBMSs ensure data integrity, concurrency control, and transaction management to support efficient and reliable transaction processing

1.2 AIM AND OBJECTIVE

The aim of our project is to develop a software solution for efficient library management and personalized book recommendations. By leveraging QR code scanning and database management, we aim to automate book tracking and ensure accurate placement. Additionally, we plan to implement a recommendation system based on user preferences. This integration of traditional library methods with modern software aims to enhance efficiency and user satisfaction within the library environment.

1.2.1 Objectives

Automated Book Tracking - Develop a system using Python and image processing techniques to automatically track book placement in the library shelves. This involves capturing images of books using cameras or scanners, processing them to identify book titles, and determining shelf locations.

Database Integration - Integrate a database system using SQLite to store comprehensive information about the library's book collection, including titles, authors, barcodes, and shelf locations. This database will serve as the backend for efficient book data management.

Barcode Detection and Verification - Implement barcode detection algorithms using Python libraries such as OpenCV to identify books based on their barcodes. Verify these barcodes against the database to ensure accurate book identification and shelf assignment.

Recommendation System - Develop a recommendation system using Python-based machine learning libraries such as scikit-learn or TensorFlow that suggests books to users based on their preferences and current selections. Utilize data analysis techniques to personalize recommendations and increase user engagement.

User Interface Design - Design an intuitive user interface (UI) for library patrons to interact with the system using Python-based GUI frameworks such as Tkinter or PyQt. The UI should facilitate book searches, provide feedback on book placements, display recommendations, and support other relevant functionalities.

Evaluation and Testing - Conduct comprehensive testing and evaluation using Python testing frameworks such as pytest or unittest to assess system performance, accuracy, and user satisfaction. Gather feedback from library staff and patrons to identify areas for improvement and refinement.

CHAPTER 2

LITERATURE REVIEW

Previous studies have improved library systems by using barcodes and RFID for better book tracking. These systems help reduce errors and speed up cataloging. Recent advancements include AI recommendations to suggest books based on users' reading habits.

[1] IoT-BASED SMART LIBRARY SYSTEM FOR BOOK MANAGEMENT. [15]

Xu and Zhu (2023) present an exploration into the development and implementation of an IoT-based smart library system designed to enhance book management. Xu and Zhu's research focuses on leveraging Internet of Things (IoT) technologies to streamline and improve various library operations related to book management, such as tracking, inventory, and user interaction.

The study highlights several key components of the smart library system, including RFID-enabled smart shelves, automated checkout and return kiosks, and environmental monitoring tools. The smart shelves equipped with RFID readers can automatically update the library's inventory database whenever a book is moved, ensuring accurate and up-to-date records. Automated kiosks allow users to check out and return books without librarian assistance.

Xu and Zhu also discuss the broader implications of implementing an IoT-based smart library system. Xu and Zhu argue that such systems can significantly enhance user experience by providing real-time information on book availability and location, personalized recommendations, and faster service.

[2] RFID AND IoT IN LIBRARY MANAGEMENT AN OVERVIEW. [12]

Smith (2019) provides a comprehensive overview of the integration of RFID (Radio Frequency Identification) and IoT (Internet of Things) technologies in modern library management. The review outlines how these technologies work together to streamline library operations, enhance security, and improve user services. RFID technology, which involves embedding books with RFID tags, allows for efficient tracking, inventory management, and self-service checkouts.

The review delves into specific applications of RFID and IoT within the library context. One significant application is in inventory management, where RFID tags and IoT sensors automate the tracking of books, reducing the need for manual stock-taking and minimizing human errors. Another application is in security, where RFID gates at library exits can detect unauthorized removal of books, thereby enhancing theft prevention. Additionally, IoT-enabled smart shelves can monitor the location and status of books, making it easier for librarians to locate misplaced items and maintain organized collections.

Smith also discusses the broader benefits and potential challenges associated with adopting RFID and IoT in libraries. The benefits include increased operational efficiency, cost savings through reduced labor and resource wastage, and improved user satisfaction through faster and more reliable services. However, the review also highlights challenges such as the initial cost of implementation, the need for technical expertise, and concerns about data privacy and security.

[3] ENVIRONMENTAL MONITORING IN LIBRARIES USING IoT. [9]

Lee and Park (2021) examine the role of IoT (Internet of Things) in enhancing environmental monitoring within libraries. Lee and Park's review emphasizes how IoT technologies can help maintain optimal conditions for preserving library collections. By using sensors and smart devices, libraries can continuously monitor environmental parameters such as temperature, humidity, light levels, and air quality. These real-time data collection capabilities ensure that library materials are kept in conditions that prevent deterioration and extend their lifespan.

The review highlights several practical implementations of IoT-based environmental monitoring systems in libraries. For instance, IoT sensors placed throughout the library can detect fluctuations in temperature and humidity, which are critical for the preservation of books and other materials. This automated approach reduces the reliance on manual checks and interventions, ensuring more consistent and precise environmental control.

Lee and Park also discuss the broader implications of adopting IoT for environmental monitoring in libraries. By optimizing the use of HVAC (heating, ventilation, and air conditioning) systems, libraries can reduce energy consumption and operational costs. Furthermore, the data collected by IoT sensors can be analyzed to identify trends and potential issues, enabling proactive maintenance and better resource management. Lee and Park conclude that IoT-based environmental monitoring represents a significant advancement for libraries, providing both preservation benefits and operational efficiencies.

[4] AI CHATBOTS IN LIBRARIES ENHANCING USER INTERACTION. [6]

Johnson (2018) explores the increasing adoption of AI chatbots in libraries and their impact on enhancing user interaction. The review highlights how AI chatbots, which use natural language processing and machine learning, provide automated, real-time assistance to library users. These chatbots can handle a variety of tasks such as answering common questions, guiding users in locating resources, and providing recommendations based on user queries. By automating these interactions, libraries can offer consistent, 24/7 support to their patrons, significantly improving the user experience.

The review details specific functionalities of AI chatbots that benefit both users and library staff. For users, chatbots offer quick responses to frequently asked questions, such as those related to library hours, locations of specific books, and how to access digital resources. This immediate assistance helps reduce wait times and enhances user satisfaction. For library staff, chatbots can handle routine inquiries, freeing up staff to focus on more complex tasks and personalized user support. This division of labor not only improves operational efficiency but also allows staff to engage in more meaningful interactions with patrons.

Johnson also addresses the challenges and future potential of AI chatbots in libraries. One challenge is ensuring that chatbots provide accurate and contextually relevant responses, which requires continuous updates and improvements in their algorithms. Additionally, there are concerns about privacy and data security, as chatbots collect and process user information. Despite these challenges, the review suggests that the potential benefits of AI chatbots are substantial.

[5] AUTOMATING METADATA GENERATION WITH AI. [8]

Kim (2019) examines the application of artificial intelligence (AI) in automating the generation of metadata for library resources. The review discusses how AI technologies, particularly machine learning and natural language processing, can large volumes of text and extract relevant information to create metadata. This process significantly reduces the time and effort required for manual cataloging, ensuring that library resources are more quickly and accurately indexed and made discoverable to users. By leveraging AI, libraries can keep up with the increasing volume of digital and physical materials needing metadata.

The review highlights several advantages of using AI for metadata generation. One key benefit is the consistency and precision that AI systems offer, as they can process and standardize data uniformly, minimizing human error and subjective inconsistencies.. Furthermore, AI can continuously learn and improve from feedback, enhancing its metadata generation capabilities over time and adapting to new types of content and formats.

Kim also addresses potential challenges and future directions for AI in metadata generation. Challenges include the initial setup cost and the need for high-quality training data to ensure the accuracy and reliability of AI-generated metadata. There are also concerns about the loss of professional expertise in cataloging as more tasks become automated. However, Kim suggests that AI should be viewed as a tool that complements rather than replaces human expertise, allowing librarians to focus on more complex and strategic tasks.

[6] RFID-ENABLED SELF-CHECKOUT SYSTEMS IN LIBRARIES. [10]

Patel (2020) explores the implementation and benefits of RFID-enabled self-checkout systems in libraries, highlighting how this technology streamlines the borrowing process and enhances user convenience. RFID (Radio Frequency Identification) tags, attached to library materials, interact with self-checkout kiosks equipped with RFID readers. This interaction allows patrons to check out and return items independently, without the need for librarian assistance.

The review delves into the operational advantages of RFID-enabled self-checkout systems. These systems not only facilitate a quicker and more efficient checkout experience for users but also improve the accuracy of inventory management. The RFID tags can be read simultaneously, enabling multiple items to be processed at once, unlike traditional barcode systems that require individual scanning.

Patel also discusses the challenges and considerations associated with the adoption of RFID-enabled self-checkout systems. One significant challenge is the initial investment required for RFID technology, which includes costs for tags, readers, and software infrastructure. Despite these challenges, Patel concludes that the benefits of RFID-enabled self-checkout systems, such as increased efficiency, improved user satisfaction, and enhanced security, make them a valuable addition to modern libraries. With careful planning and implementation, libraries can successfully leverage this technology to enhance their services and operations.

[7] ENHANCING LIBRARY SECURITY WITH RFID. [14]

Wang (2019) investigates the use of RFID (Radio Frequency Identification) technology to enhance library security. The review explores how RFID can significantly improve security measures by automating and enhancing the monitoring and control of library materials. RFID tags, embedded in library items, allow for efficient detection of unauthorized removal and can trigger alarms if items are taken out without being properly checked out.

The review discusses several key components of RFID-based security systems in libraries. One of the main features is the RFID gate, which is placed at library exits and detects RFID tags in library items as patrons pass through. If an item has not been properly checked out, the gate triggers an alarm, alerting library staff to the potential theft. This system not only enhances security but also provides a deterrent effect, discouraging individuals from attempting to remove library materials without authorization.

Wang also highlights the operational benefits of RFID for library security. The technology enables libraries to automate inventory management and reduce the time and effort required for manual security checks. RFID systems can conduct inventory audits much faster and more accurately than traditional methods, ensuring that library collections are accounted for and reducing the likelihood of misplaced or lost items. Overall, Wang concludes that RFID technology is a valuable tool for enhancing library security, offering both preventive measures against theft and operational efficiencies that benefit library staff and patrons alike.

CHAPTER 3

SYSTEM ANALYSIS

System analysis is the process of examining and understanding a system to identify its components, relationships, and functionality. This analytical phase involves breaking down the system into smaller parts to study how they interact, determining requirements, and uncovering potential problems or inefficiencies.

3.1 EXISTING SYSTEM

In the existing system, library staff manage book tracking and organization manually. This involves a lot of physical effort, as staff must frequently check the shelves, organize returned books, and ensure each book is placed in its correct location. Some libraries use barcode systems for checking out and returning books, which helps speed up the process and reduce errors. However, these systems typically do not include automated tracking of book placements, meaning staff must still manually verify and adjust the arrangement of books on the shelves.

For users, interaction with the library system is limited to basic functions like checking out books, returning them, and searching for titles in the catalog. Manual book management also involves various challenges that affect both library staff and patrons. The lack of automation in tracking book placements means that significant time and effort are spent on ensuring books are in their proper locations. This manual process can be cumbersome and prone to errors, leading to misplaced books and difficulty in locating them.

Additionally, the current system does not facilitate personalized services such as book recommendations based on user preferences, which could enhance the user experience and encourage more frequent library visits. Consequently, the manual nature of this system results in inefficiencies and a less satisfying experience for library patrons.

3.1.1 Limitations

The manual process requires a lot of time and effort from library staff to organize books and ensure they are correctly shelved. This labor-intensive work can be particularly challenging during busy periods when many books are returned or checked out at once. Without real-time monitoring, it can be difficult for patrons to find books if they are misplaced or not returned to their proper locations, leading to frustration and inefficiency.

Additionally, the existing system does not offer personalized book recommendations based on users' reading preferences and choices, which could enhance the user experience by helping patrons discover new books that match their interests. This lack of personalization means that users miss out on potential new reads that could be of great interest to them.

3.2 PROPOSED SYSTEM

Our proposed "Library Management System with Book Recommendation System" is a comprehensive solution designed to modernize library operations and enhance user experience. At the core of our system is the utilization of Python-based technologies to automate various tasks and functionalities.

Firstly, the system employs image processing techniques implemented in Python to track the placement of books within library shelves. Cameras or scanners capture images of books, and Python scripts analyze these images to identify book titles and shelf locations. This information is then stored in a centralized database, allowing for efficient inventory management.

Database integration plays a crucial role in our system, with a SQLite database utilized to store comprehensive information about the library's book collection. Python scripts handle database interactions, facilitating seamless retrieval and management of book data. Furthermore, barcode detection algorithms implemented in Python, using libraries like OpenCV, identify books based on their barcodes and verify them against the database to ensure accurate book identification and shelf assignment.

The system also incorporates a sophisticated recommendation engine, developed using Python-based machine learning libraries, to suggest relevant books to users based on their preferences and borrowing history. A user-friendly interface, designed using Python GUI frameworks such as Tkinter or PyQt, allows library patrons to interact seamlessly with the system, facilitating book searches, providing feedback on book placements, displaying personalized recommendations, and supporting other relevant functionalities.

Lastly, the system is built to be scalable and reliable, capable of accommodating a growing library collection and user base. Python-based scalable architectures, such as microservices or cloud-based solutions, ensure efficient handling of increased demand, while robust error handling mechanisms guarantee uninterrupted service and graceful handling of system failures or errors. Overall, our proposed system aims to revolutionize library management, optimize book organization, and enhance the overall library experience for patrons.

3.2.1 Algorithm

Algorithm name: “ Greedy Algorithm ”

A Greedy Algorithm is a problem-solving approach that makes the best choice at each step to achieve the best overall outcome. In the context of our project, the Greedy Algorithm can be applied to optimize book placements and recommendations by always making the best immediate decision based on the current data.

START Program

INITIALIZE database connection

INITIALIZE camera for QR code scanning

INITIALIZE user interface

DISPLAY Main Menu

OPTIONS: Scan Book, Search Book, View Recommendations, Exit

WAIT for user input

IF user selects 'Scan Book' THEN

 OPEN camera

 SCAN QR code

 EXTRACT book_id from QR code

 QUERY database with book_id

IF book_id is invalid THEN

 DISPLAY "Invalid QR code"

```
GOTO Main Menu
ENDIF

COMPARE book's current location with recorded location

IF location matches THEN
    DISPLAY "Book is correctly placed"
ELSE
    DISPLAY "Book is misplaced, should be at location: correct_location"
    GOTO Main Menu
ENDIF

CALL RecommendBooks(book_id)
DISPLAY recommended books
CLOSE camera
GOTO Main Menu
ENDIF

IF user selects 'Search Book' THEN
    PROMPT user for search query
    QUERY database for matching books
    DISPLAY search results

    WAIT for user to select a book
    IF no book is selected THEN
        GOTO Main Menu
    ENDIF
```

```

    DISPLAY details of selected book
    CALL RecommendBooks(selected_book_id)
    DISPLAY recommended books
    GOTO Main Menu
ENDIF

IF user selects 'View Recommendations' THEN
    PROMPT user for user ID or book ID
    READ input

    IF input is user ID THEN
        QUERY database for user's borrowing history
        GENERATE recommendations based on borrowing history
    ELSE
        QUERY database for similar books
        GENERATE recommendations based on book ID
    ENDIF

    DISPLAY recommended books
    GOTO Main Menu
ENDIF

IF user selects 'Exit' THEN
    CLOSE database connection
    END Program
ENDIF

```

MAIN MENU:

DISPLAY Main Menu

OPTIONS: Scan Book, Search Book, View Recommendations, Exit

WAIT for user input

RECOMMEND_BOOKS(book_id or user_id):

QUERY database for recommendations based on book_id or user_id

RETURN list of recommended books

3.2.2 Advantages

Ensures books are always in their correct locations, making it easier for users to find books and reducing the need for staff to reorganize misplaced items. Enhances user satisfaction by providing immediate feedback on book placement and personalized book recommendations. Automates book placement checks and recommendations, minimizing human error and ensuring accurate library management. Saves time for both library staff and users, allowing staff to focus on more critical tasks and users to quickly find books.

CHAPTER 4

SYSTEM REQUIREMENTS

The successful implementation of the Smart Library Management System relies on a combination of hardware and software components. These elements work together to ensure seamless operation, accurate book tracking, and efficient user interaction. Below are the key system requirements needed for this project.

4.1 HARDWARE REQUIREMENTS

Laptop: Any modern laptop with a built-in camera can be used to run the system.

CPU: A standard dual-core or quad-core processor

Memory (RAM): At least 4GB of RAM

Storage: 300GB

Camera: A built-in camera or an external webcam

Display: The laptop's built-in display or an external monitor

Internet Connectivity: accessing online resources and updates.

The hardware requirements for the "Library Management System with Book Recommendation System" project are carefully selected to ensure the smooth functioning and effectiveness of the system. Here's why each hardware component is essential: Firstly, a laptop serves as the primary hardware platform for deploying the system. Laptops are portable and versatile, making them ideal for use in various library settings. They provide the necessary computing power to run the system's software components and interact with users effectively.

The CPU, or central processing unit, is the brain of the laptop and is responsible for executing instructions and performing computations. A modern dual-core or quad-core processor ensures that the system can handle multiple tasks simultaneously without experiencing performance bottlenecks. Sufficient memory (RAM) is crucial for ensuring smooth system operation. With at least 4GB of RAM, the system can efficiently handle data processing, storage, and retrieval tasks, providing a responsive user experience.

Adequate storage space is required for storing the application files, database, and any additional resources needed for the system to function correctly. Sufficient storage ensures that the system can store a large volume of book data and user information without running out of space. The built-in camera or external webcam is essential for capturing images of books for book tracking purposes. By leveraging camera technology, the system can automatically identify books and track their placement within the library shelves, enhancing accuracy and efficiency in book management.

The display, whether it's the laptop's built-in screen or an external monitor, provides the user interface for interacting with the system. It allows library staff to view feedback notifications, perform book searches, and access other system functionalities easily. While internet connectivity is optional, it is recommended for accessing online resources. Overall, the hardware requirements ensure that the system can be deployed effectively on a laptop, providing library staff with the tools they need to manage books efficiently and enhance user experience within the library environment.

4.2 SOFTWARE REQUIREMENTS

- Python – 3.12.0
- OpenCV Library – 4.9.0
- SQLite – 3.0.0.1

The software requirements for the "Library Management System with Book Recommendation System" project play a crucial role in enabling the development, deployment, and functionality of the system. Here's how each software component contributes to the system: Firstly, Python serves as the primary programming language for developing the system. Python's simplicity, readability, and extensive library support make it an ideal choice for implementing various system functionalities, such as image processing, database management, and user interface development.

Python libraries, such as OpenCV for image processing and SQLite for database management, provide essential functionalities required for the system's operation. OpenCV allows the system to analyze images captured by cameras or scanners to identify book titles and shelf locations, facilitating automated book tracking. SQLite, on the other hand, provides a lightweight and efficient database solution for storing and managing book information, ensuring seamless data retrieval and manipulation.

An integrated development environment (IDE) like PyCharm, Visual Studio Code, or Jupyter Notebook is essential for code development and debugging. These IDEs offer features such as syntax highlighting, code completion, and debugging tools, enhancing developer productivity and facilitating the development process.

Additionally, the choice of a graphical user interface (GUI) framework, such as Tkinter, PyQt, or Kivy, enables the development of an intuitive and user-friendly interface for interacting with the system. A well-designed GUI enhances user experience, allowing library staff to navigate the system effortlessly and perform tasks efficiently.

While optional, the inclusion of a web server like Apache or Nginx may be necessary if the system includes web-based components. A web server enables hosting of the web application, providing users with access to system functionalities through a web browser interface.

Overall, the software requirements provide the necessary tools and frameworks for developing, deploying, and maintaining the "Library Management System with Book Recommendation System." By leveraging Python and its associated libraries, along with appropriate development and deployment tools, the system can be effectively implemented to streamline library operations and enhance user experience within the library environment.

CHAPTER 5

MODULE DESCRIPTION

The Module Description section in a project documentation provides detailed information about each functional component or module of the system. This section breaks down the project into its core elements, describing the purpose, functionality, and interactions of each module. By doing so, it helps readers understand how the system operates as a whole and how each part contributes to the overall objectives of the project.

5.1 SYSTEM INTEGRATION

The system integration process for our "Library Management System with Book Recommendation System" project involves seamlessly combining various modules and components to ensure cohesive functionality and optimal performance. Firstly, integration begins with the coordination of hardware and software components, ensuring that the system can effectively leverage resources such as the laptop's camera for book tracking and the necessary software libraries for image processing and database management. This integration ensures that hardware devices and software modules work together harmoniously to achieve the system's objectives.

Secondly, the integration process involves linking the different software modules to facilitate data exchange and communication within the system. For instance, the Book Tracking Module interacts with the Barcode Detection and Verification Module to identify books based on their barcodes and update their locations in the database accordingly.

Similarly, the Recommendation System Module utilizes data from the Database Integration Module to generate personalized book recommendations for users. This seamless integration of software modules enables the system to perform complex tasks efficiently and accurately.

Finally, system integration encompasses testing and validation to ensure that all integrated components work together as intended. Comprehensive testing, including unit testing, integration testing, and user acceptance testing, is conducted to verify the functionality, reliability, and usability of the integrated system. Any inconsistencies or issues identified during testing are addressed through iterative refinement and optimization of the integration process. Ultimately, successful system integration results in a robust and cohesive "Library Management System with Book Recommendation System" that enhances library operations and user experience effectively.

5.2 DATABASE

Creating the database for our "Library Management System with Book Recommendation System" using SQLite involves several essential steps to ensure its effectiveness and efficiency. Initially, the database schema must be meticulously designed to accurately represent the various entities and relationships within the library management system. This entails identifying the fundamental components such as books, users, borrowing history, and book genres, and determining their respective attributes. Each table's structure is carefully crafted, specifying data types, constraints, and relationships to maintain data integrity and facilitate efficient storage and retrieval of information. Once the tables are created and populated with initial data, the database undergoes thorough testing and validation to ensure its functionality and reliability.

Various queries and operations are executed to retrieve, update, and delete data from the database, ensuring that the system behaves as expected and that data integrity is maintained. Testing also includes validating constraints and relationships to verify that they are enforced correctly. Any inconsistencies or issues identified during testing are addressed promptly through iterative refinement and optimization of the database schema and data.

Furthermore, documentation plays a crucial role in documenting the database schema, including table structures, relationships, constraints, and sample data. This documentation serves as a reference guide for developers, administrators, and users interacting with the database, providing valuable insights into its design and functionality. It also aids in troubleshooting and maintenance tasks, facilitating the smooth operation of the database over time.

In conclusion, creating the database using SQLite for our "Library Management System with Book Recommendation System" involves meticulous planning, execution, and validation to ensure its effectiveness and reliability. By following a structured approach and adhering to best practices in database design and management, we can develop a robust database that supports the system's functionalities and enhances the overall efficiency and user experience of the library management system.

5.3 PYTHON

Connecting the SQLite database with Python involves using the SQLite3 module, a built-in Python library that facilitates interaction with SQLite databases. Initially, you import the SQLite3 module into your Python script, enabling database connectivity and manipulation.

Once imported, you establish a connection to the SQLite database file using the `connect()` function, specifying the path to the database file. This connection serves as a gateway for executing SQL commands and interacting with the database.

After establishing the connection, you create a cursor object using the `cursor()` method of the connection object. This cursor acts as a pointer to facilitate database operations such as executing SQL queries and modifying database contents. With the cursor in hand, you can execute SQL commands to perform various tasks, such as creating tables, inserting data, querying data, updating records, or deleting records.

One of the primary tasks when connecting the database with Python is executing SQL commands to create the necessary database structures. For instance, you can create tables to store QR code information, book details, user information, or borrowing history. By executing SQL commands through the cursor object, you define the database schema and establish the foundation for storing and managing data within the library management system.

Once the database structures are created and modified as needed, you commit the changes to save them permanently to the database file using the `commit()` method. Finally, you close the connection to the database using the `close()` method, releasing any associated resources and ensuring proper termination of the database connection.

5.4 RECOMMENDATION INTERFACE

Creating the recommendation interface for our "Library Management System with Book Recommendation System" involves designing and implementing a user-friendly interface that provides personalized book recommendations to users based on their preferences and borrowing history. Here are the steps and key considerations for creating the recommendation interface:

User Interface Design - Begin by designing the user interface (UI) layout and components using a GUI framework such as Tkinter, PyQt, or Kivy in Python. The recommendation interface should be intuitive and easy to navigate, allowing users to input their preferences, view recommended books, and explore additional options. Consider including elements such as text inputs for entering preferences (e.g., favorite genres, authors), dropdown menus for selecting criteria (e.g., new releases, popular titles), and buttons for submitting queries and displaying recommendations.

Integration with Recommendation Engine - Integrate the recommendation interface with the recommendation engine module of the system, which generates personalized book recommendations based on user input and historical data. Implement functionality to capture user preferences from the interface and pass them as input to the recommendation engine. Retrieve the recommended books from the recommendation engine and display them on the interface in a visually appealing and informative manner. Ensure that the interface seamlessly communicates with the recommendation engine to provide real-time and accurate recommendations to users.

CHAPTER 6

SYSTEM DESIGN

The system design of the Smart Library Management System outlines the architecture and workflow to achieve its functionality. This design includes the integration of hardware components, software modules, and their interactions.

6.1 ARCHITECTURE DIAGRAM

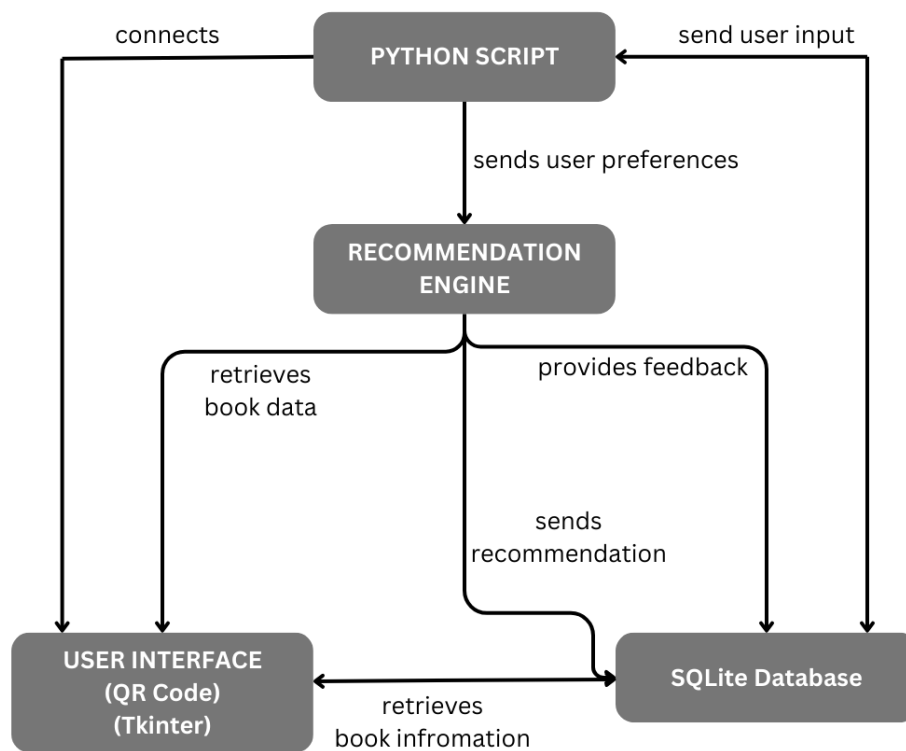


Figure 6.1 Architecture Diagram

In our system architecture, the Python script acts as the mediator, establishing connections between the SQLite database, recommendation engine, and user interface.

Through these connections, the script coordinates the flow of data and functionalities, retrieving book information from the database, sending user preferences to the recommendation engine, and displaying personalized recommendations on the user interface.

6.2 COMMUNICATION DIAGRAM

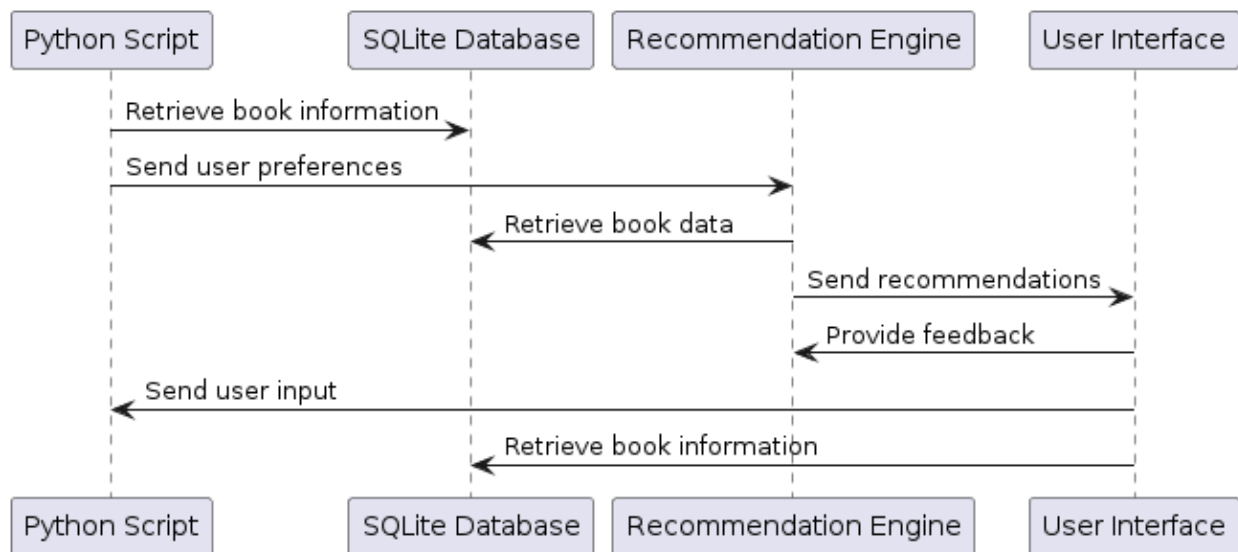


Figure 6.2 Communication Diagram

In our system architecture, communication flows between components to facilitate seamless interaction and data exchange. The Python script initiates communication by retrieving book information from the SQLite database and sending user preferences to the recommendation engine. The recommendation engine then communicates with the database to retrieve relevant book data and generates personalized recommendations, which are sent to the user interface for display.

6.3 USE CASE DIAGRAM

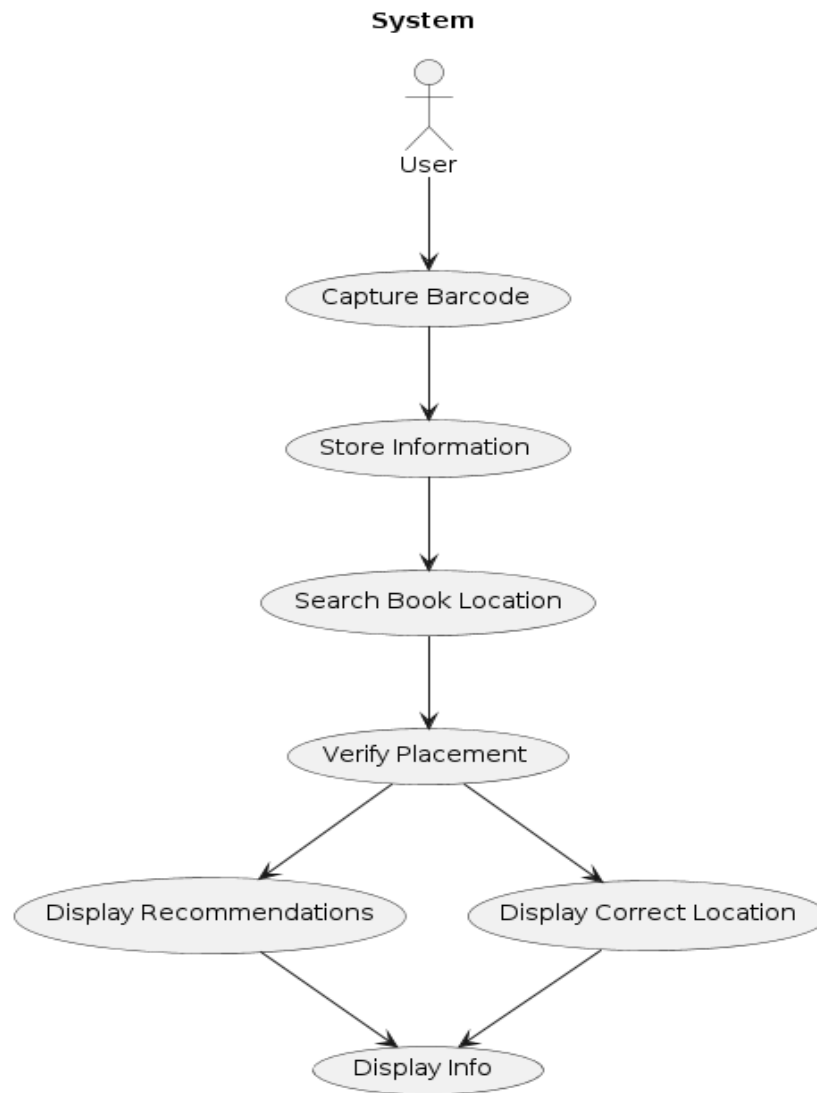


Figure 6.3 Use Case Diagram

The user scans the QR code of the book and the system fetches the book id from the SQLite database. If it is the right rack to place the book then it shows similar recommendations of books and if not then it displays that right rack number of the respective book.

CHAPTER 7

SYSTEM TESTING

System testing is a phase in the software testing lifecycle where the complete and integrated software is tested to ensure that the system meets its specified requirements. It involves testing the system as a whole, including the interaction between its components, to validate the end-to-end functionalities.

7.1 UNIT TESTING

Unit testing for our "Smart Library Management with Recommendation system" project involves testing individual components or units of the system in isolation to ensure they function correctly and meet their specifications. For the unit testing of database operations, several key aspects need validation to ensure the correct functioning of the SQLite database integration within the system. Initially, the connectivity between the Python script and the SQLite database requires verification.

7.1.1 Recommendation Engine

Unit testing of the recommendation engine focuses on validating its core functionality, accuracy, and effectiveness in generating personalized book recommendations. Firstly, the recommendation generation process undergoes scrutiny to ensure that the engine can effectively analyze user preferences and historical data to generate relevant recommendations. Additionally, tests are conducted to verify the accuracy of the recommendations provided by the engine. This involves comparing the recommended books against the user's preferences and assessing whether the suggestions align with the user's interests.

7.1.2 User Interface

Unit testing of the user interface encompasses the verification of various UI components and user interactions within the system. Firstly, the functionality of UI elements such as buttons, text inputs, dropdown menus, and feedback forms is tested to ensure they perform their intended actions accurately. Tests are conducted to validate the handling of user input by the interface, including the capture and processing of user preferences, feedback, and search queries. Additionally, the display of recommended books on the interface undergoes scrutiny to confirm that the suggestions are presented accurately and attractively to the user. Through thorough unit testing of the user interface, the system can ensure a seamless and intuitive user experience, enhancing user satisfaction and engagement.

7.1.3 Database Module

Unit testing for the database module involves validating the functionality and reliability of database operations within the system. Initially, the connectivity between the Python script and the SQLite database is tested to ensure seamless communication. This entails verifying that the script can establish a connection to the database and handle potential connection errors effectively.

Subsequently, CRUD operations (Create, Read, Update, Delete) are subjected to rigorous testing to guarantee the correctness of data manipulation. Unit tests are conducted to ascertain that functions responsible for creating, reading, updating, and deleting records in the database perform their tasks accurately. These tests validate the integrity of data insertion, retrieval, modification, and deletion, ensuring the consistency and reliability of the database operations.

7.2 INTEGRATION TESTING

Integration testing for our project involves testing the interaction and integration between different components of the system to ensure that they work together seamlessly and fulfill the system requirements. Here's how integration testing can be conducted for key components of the system:

7.2.1 Database and Python Script

The integration between the Python script and the SQLite database is crucial for ensuring seamless data management within the system. Integration testing verifies that the Python script can establish a connection to the database and perform CRUD operations accurately. Tests are conducted to validate data insertion, retrieval, modification, and deletion operations, ensuring that data is managed effectively within the database. Additionally, integration tests assess error handling mechanisms to confirm that the system responds appropriately to database connection errors or exceptions, maintaining reliability and stability in database interactions.

7.2.2 Recommendation Engine and User Interface

Integration testing also focuses on validating the interaction between the recommendation engine and the user interface to ensure the accurate generation and display of personalized book recommendations. Tests are conducted to verify that the recommendation engine can effectively analyze user preferences and historical data to generate relevant recommendations. By validating the integration between the recommendation engine and the user interface, the system can provide users with personalized book recommendations that align with their interests and preferences.

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENT

8.1 CONCLUSION

The smart library management system is a modern solution that incorporates QR code technology and intelligent hardware integration to streamline operations and enhance user experience. By seamlessly integrating the ESP32CAM module and FTDI232 connector, the system ensures efficient book scanning and placement verification. Real-time feedback is provided to users through visual and auditory cues, ensuring accurate book placement and improving overall usability. The system's ability to detect correct book placement and issue warnings for incorrect placement ensures that library shelves remain organized and books are readily accessible to users. This project aims to revolutionize traditional library management systems by leveraging IoT technology to automate book tracking, enhance organization, and provide personalized recommendations to users. .

8.2 FUTURE ENHANCEMENTS:

Future enhancements for the "Library Management System with Book Recommendation System" project could significantly expand its functionality and improve user experience. One potential enhancement is integrating machine learning algorithms for more sophisticated and personalized book recommendations. By analyzing user behavior, borrowing history, and preferences, the system could provide more accurate and diverse book suggestions. Additionally, implementing natural language processing (NLP) capabilities could allow users to interact with the system through voice commands or conversational interfaces, making the system more accessible and user-friendly.

Another future enhancement could involve expanding the system's capabilities to support mobile devices and remote access. Developing a mobile application would enable users to search for books, receive recommendations, and check book availability from their smartphones or tablets. Moreover, integrating cloud-based services could ensure data synchronization across multiple devices and locations, allowing library staff and patrons to access the system anytime, anywhere. These enhancements would not only improve the system's flexibility and convenience but also enhance user engagement and satisfaction.

APPENDIX A

(SOURCE CODE)

```
import tkinter as tk
from tkinter import messagebox, ttk
import cv2
from pyzbar.pyzbar import decode
import sqlite3

class LibraryManagementApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Smart Library Management System")
        self.root.geometry("600x400")
        self.root.configure(bg="#f9f9f9")
        self.style = ttk.Style()
        self.style.configure("TLabel", font=("Arial", 12), background="#f9f9f9")
        self.style.configure("TButton", font=("Arial", 12))
        self.header = tk.Label(root, text="Smart Library Management System with
Recommendation", font=("Arial", 18, "bold"), bg="#f9f9f9", fg="#333333")
        self.header.pack(pady=20)
        self.label = tk.Label(root, text="Enter the current rack number:",
font=("Arial",14),background="#f9f9f9")
        self.label.pack(pady=10)
        self.rack_entry = tk.Entry(root, width=20, font=("Arial", 12))
        self.rack_entry.pack(pady=10)
        self.scan_button = tk.Button(root, text="Scan Barcode",
command=self.start_scanning, style="Custom.TButton")
```

```

self.scan_button.pack(pady=10)
self.total_books_button = tk.Button(root, text="View Total Books",
command=self.view_total_books, style="Custom.TButton")
self.total_books_button.pack(pady=5)
self.list_books_button = tk.Button(root, text="List Books",
command=self.list_books, style="Custom.TButton")
self.list_books_button.pack(pady=5)
self.close_button = tk.Button(root, text="Close",
command=self.close_window, style="Custom.TButton")
self.close_button.pack(pady=5)
self.result_label = tk.Label(root, text="", font=("Arial", 12), bg="#f9f9f9",
fg="#333333")
self.result_label.pack(pady=10)
def start_scanning(self):
    current_rack = self.rack_entry.get()
    if not current_rack:
        messagebox.showerror("Input Error", "Please enter a rack number.")
        return
    self.result_label.config(text="Scanning for barcode...")
    # Initialize camera
    cap = cv2.VideoCapture(0)
    output_shown = False
    while not output_shown:
        # Read frame from camera
        ret, frame = cap.read()
        # Decode barcodes
        barcodes = decode(frame)

```

```

# Loop over detected barcodes
for barcode in barcodes:
    # Extract barcode data
    barcode_data = barcode.data.decode('utf-8')
    # Check if the book is placed in the correct rack
    book_title, correct_rack, genre =
self.check_book_placement(barcode_data)
    if book_title:
        if current_rack == correct_rack:
            result_text = f"The book '{book_title}' is placed in the correct rack."
            recommendations = self.recommend_books(genre, book_title)
            if recommendations:
                result_text += "\nYou might also like these books from the same
genre:"

                for rec in recommendations:
                    result_text += f"\n- {rec[0]} (Rack {rec[1]})"
            else:
                result_text += "\nNo other recommendations available in the
same genre."
        else:
            result_text = f"The book '{book_title}' is placed in the wrong rack.
Please place it in {correct_rack}."
            self.result_label.config(text=result_text)
            output_shown = True
            break
    # Exit loop if 'q' is pressed
    if cv2.waitKey(1) & 0xFF == ord('q'):

```

```

        break

    # Release camera and close OpenCV windows
    cap.release()
    cv2.destroyAllWindows()

def check_book_placement(self, barcode_data):
    conn = sqlite3.connect('library.db')
    cursor = conn.cursor()

    # Query the database for the book with the given barcode
    cursor.execute('SELECT title, rack_no, genre FROM books WHERE barcode
= ?', (barcode_data,))
    book = cursor.fetchone()
    conn.close()

    if book:
        return book[0], book[1], book[2]
    else:
        return None, None, None

def recommend_books(self, genre, current_book_title):
    conn = sqlite3.connect('library.db')
    cursor = conn.cursor()

    # Query the database for books of the same genre, excluding the current book
    cursor.execute('SELECT title, rack_no FROM books WHERE genre = ?
AND title != ? LIMIT 3', (genre, current_book_title))
    recommendations = cursor.fetchall()
    conn.close()

    return recommendations

def view_total_books(self):
    conn = sqlite3.connect('library.db')

```

```

    cursor = conn.cursor()
    # Query the database for the total number of books
    cursor.execute('SELECT COUNT(*) FROM books')
    total_books = cursor.fetchone()[0]
    conn.close()
    messagebox.showinfo("Total Books", f"Total number of books in the library:
{total_books}")
def list_books(self):
    conn = sqlite3.connect('library.db')
    cursor = conn.cursor()
    # Query the database for all book titles
    cursor.execute('SELECT title FROM books')
    book_titles = [book[0] for book in cursor.fetchall()]
    conn.close()
    book_list = "\n".join(book_titles)
    messagebox.showinfo("Book List", f"List of books:\n{book_list}")
def close_window(self):
    self.root.destroy()
if __name__ == "__main__":
    root = tk.Tk()
    app = LibraryManagementApp(root)
    # Customizing button style
    root.style = ttk.Style()
    root.style.theme_use('clam')
    root.style.configure('Custom.TButton', background='#007bff',
foreground='#ffffff', font=('Arial', 12, 'bold'))
    root.mainloop()

```

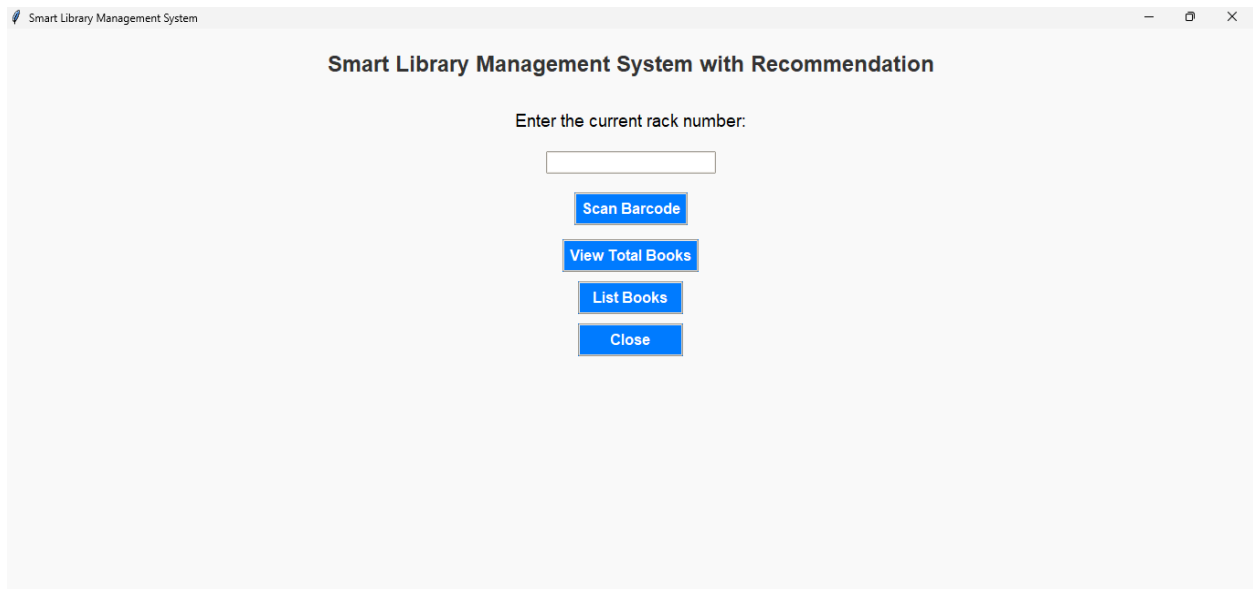
```

import sqlite3
conn = sqlite3.connect('library.db')
cursor = conn.cursor()
cursor.execute("""
    CREATE TABLE IF NOT EXISTS books (
        id INTEGER PRIMARY KEY,
        title TEXT NOT NULL,
        barcode TEXT NOT NULL UNIQUE,
        rack_no TEXT NOT NULL,
        genre TEXT NOT NULL
    )
""")
# Insert multiple book records into the table with genres
books = [
    ('Mobile Code 1', '0051111407592', '1', 'Mobile'),
    ('Mobile Code 2', '0512345000107', '1', 'Mobile'),
    ('Transformer', '9788184720099', '2', 'EEE'),
    ('Coil', '987654567786', '2', 'EEE'),
    ('DBMS', '9789333221290', '3', 'Database'),
    ('RDBMS', '9789333221287', '3', 'Database'),
    ('OS', '9789333221153', '4', 'CS'),
    ('SE', '9789333221345', '4', 'CS')
]
cursor.executemany('INSERT OR IGNORE INTO books (title, barcode, rack_no,
genre) VALUES (?, ?, ?, ?)', books)
conn.commit()
conn.close()

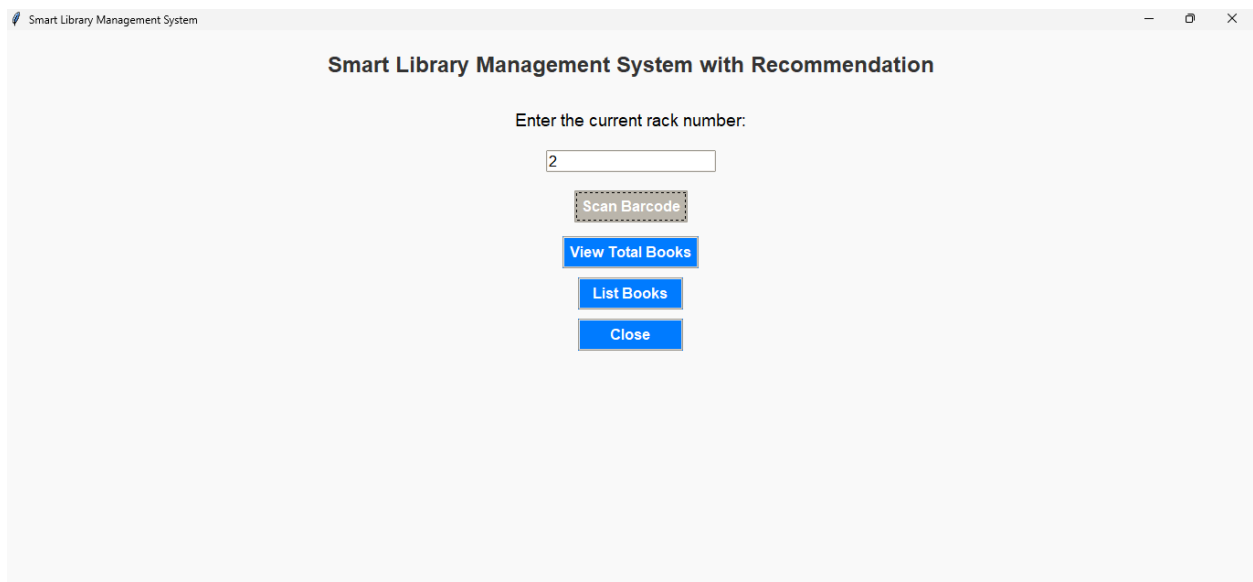
```

APPENDIX B

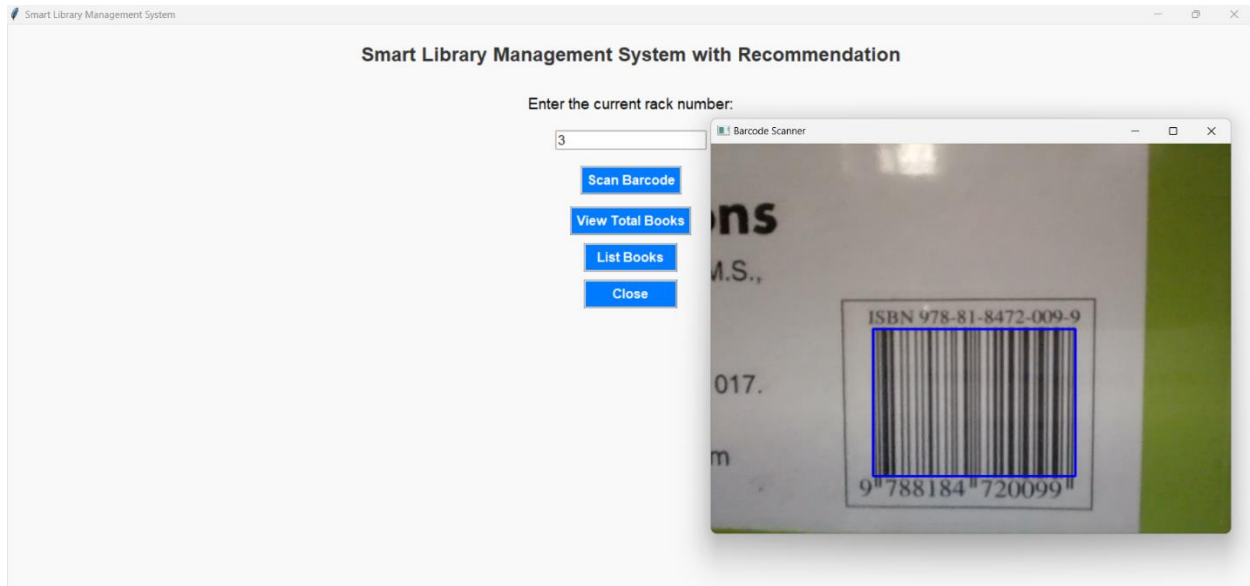
(SCREENSHOTS)



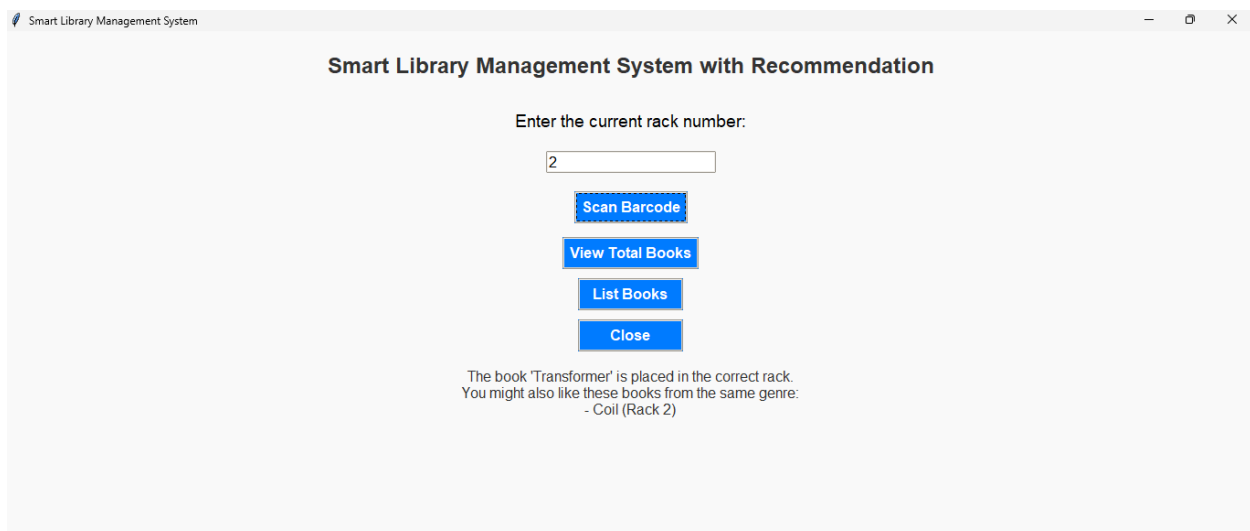
The main UI integrates the database with Tkinter. It includes an input field for the rack number and four buttons: barcode scanning, total books, list of books, and close.



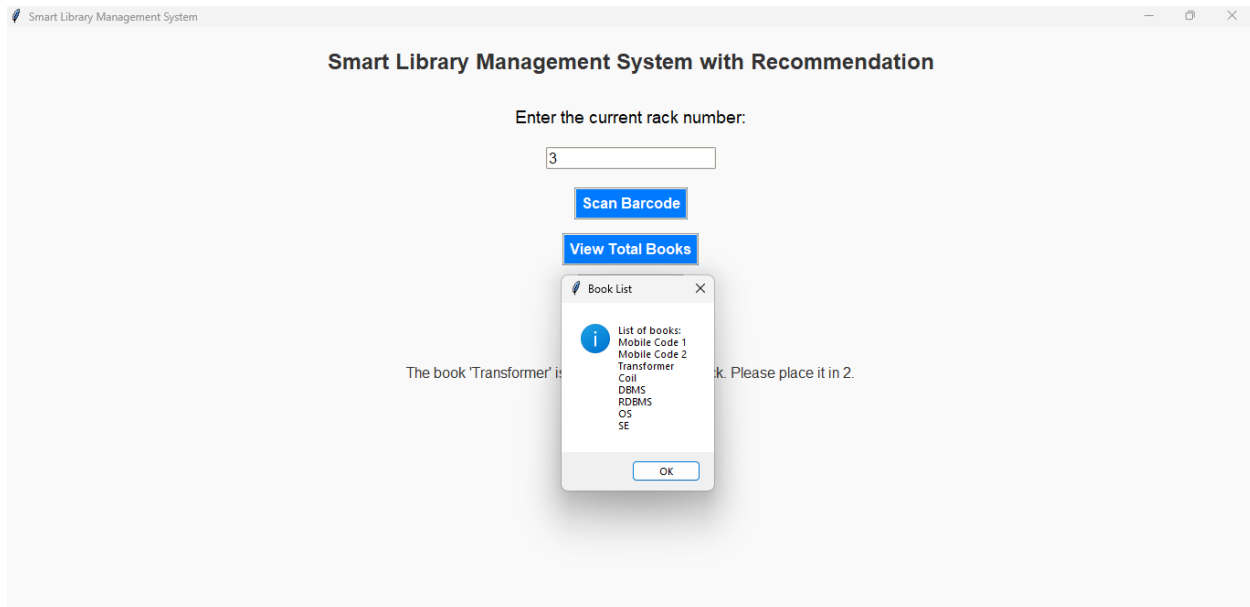
This screenshot shows the display when the user is entering the rack number .This rack number is fixed as the default rack to the program.



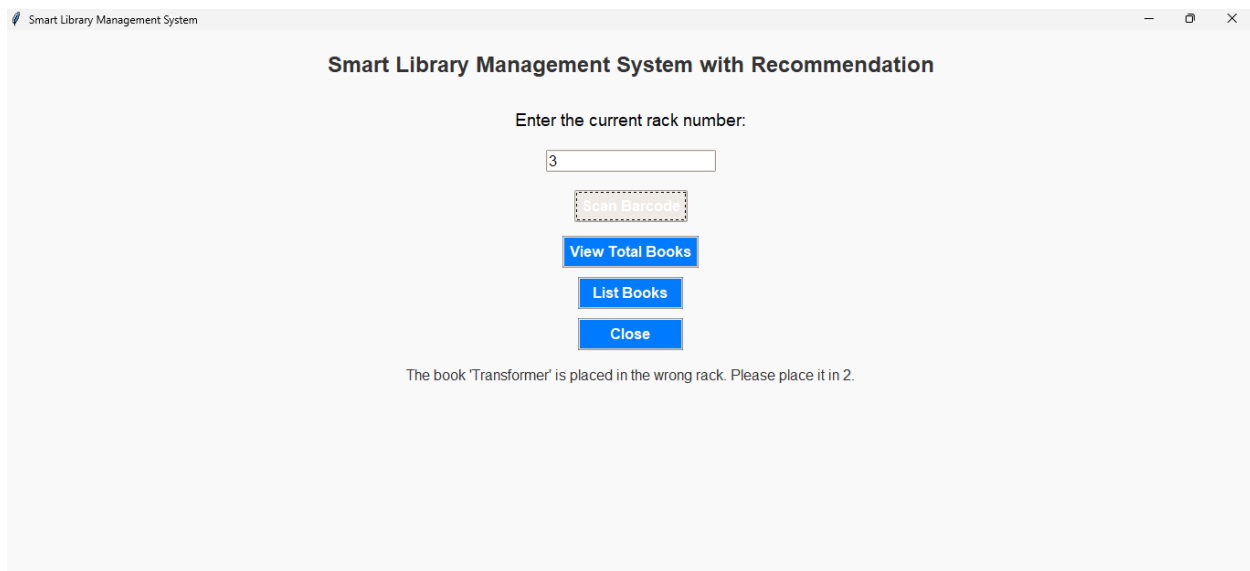
Then the barcode of a book is scanned by the barcode scanner. This scanned code is searched in the SQLite database.



This Scanned information is scanned in the database of the particular rack number and displays the correctness of the rack. If the book is placed in the right place then it shows that it is placed in the right place and recommends similar books. If not then it shows that it is not the right rack to place and displays the correct rack to place.



The Project also displays the list of books in the respective rack if we press the "View Total Books" button.



If a book which is not supposed to be placed in the particular rack is scanned then it shows the error message as displayed in the above screenshot. It also shows the correct location of the book.

REFERENCES

- [1] Anderson, M. (2020). “Big Data Analytics in Libraries: Opportunities and Challenges.” Information Research.
- [2] Clark, E. (2021). “Addressing the Digital Divide in Smart Library Initiatives.” Information Technology and Libraries.
- [3] Davis, A. (2018). “Personalized Library Services through Smart Technologies.” Public Library Quarterly.
- [4] Green, C. (2021).”Improving Library Accessibility with Digital Solutions.” Library & Information Science Research.
- [5] Hill, M. (2019). “Overcoming Barriers to Implementing Smart Library Technologies.” Library Technology Reports.
- [6] Johnson, L. (2018). “AI Chatbots in Libraries: Enhancing User Interaction.” Library Trends.
- [7] Jones, P. (2020). “Cost Savings through Smart Library Management.” Library Resources & Technical Services.
- [8] Kim, H. (2019).” Automating Metadata Generation with AI.” Journal of Information Science.
- [9] Lee, S., & Park, K. (2021).”Environmental Monitoring in Libraries Using IoT”. Journal of Library and Information Science..

- [10] Patel, R. (2020). "RFID-Enabled Self-Checkout Systems in Libraries." Library Hi Tech.
- [11] Roberts, K. (2019)." Optimizing Library Resources with Data Analytics." Journal of Academic Librarianship.
- [12] Smith, J. (2019). "RFID and IoT in Library Management An Overview." Library Technology Reports.
- [13] Taylor, L. (2021)." Sustainability Practices in Smart Libraries". Journal of Librarianship and Information Science.
- [14] Wang, Y. (2019)."Enhancing Library Security with RFID." Library Management.
- [15] Xu, H., & Zhu, Y. (2023). "IoT-Based Smart Library System for Book Management". International Journal of Emerging Technologies.

CERTIFICATES



K.RAMAKRISHNAN
COLLEGE OF TECHNOLOGY
Autonomous

Affiliated to Anna University Chennai, Approved by AICTE New Delhi,
ISO 9001:2015, 14001:2015 Certified Institution, Accredited with A grade by NAAC
Samayapuram, Tiruchirappalli - 621 112, Tamilnadu, India.



**International Conference on Newer
Engineering Concepts and Technology
(ICONNECT - 2024)**
28th MARCH 2024



TNSCST
TAMILNADU STATE COUNCIL FOR
SCIENCE AND TECHNOLOGY



CERTIFICATE

This is to certify that ABISHEK KHANNA M.S
has presented a paper entitled SMART LIBRARY
MANAGEMENT WITH RECOMMENDATION
SYSTEM
in the International Conference on Newer Engineering
Concepts and Technology (ICONNECT-2024), held at
K.Ramakrishnan College of Technology (Autonomous),
Samayapuram, Tiruchirappalli, India on 28.03.2024.


Coordinator


Conference Chair



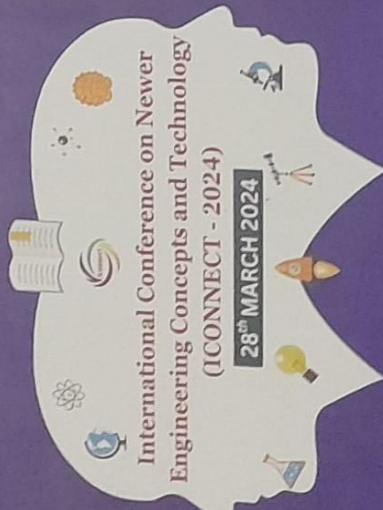

HoD


Principal



K. RAMAKRISHNAN
COLLEGE OF TECHNOLOGY
Autonomous

Alligned to Anna University Chennai, Approved by AICTE, New Delhi,
ISO 9001:2015, 14001:2015 Certified Institution. Accredited with A grade by NAAC
Samayapuram, Tiruchirappalli - 621 112, Tamilnadu, India.



CERTIFICATE




This is to certify that ALAGAPPAN . V
has presented a paper entitled SMART LIBRARY
..... MANAGEMENT WITH RECOMMENDATION
..... SYSTEM

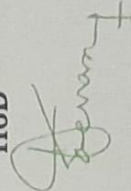
in the International Conference on Newer Engineering
Concepts and Technology (ICONNECT-2024), held at
K.Ramakrishnan College of Technology (Autonomous),
Samayapuram, Tiruchirappalli, India on 28.03.2024.


Coordinator




HoD

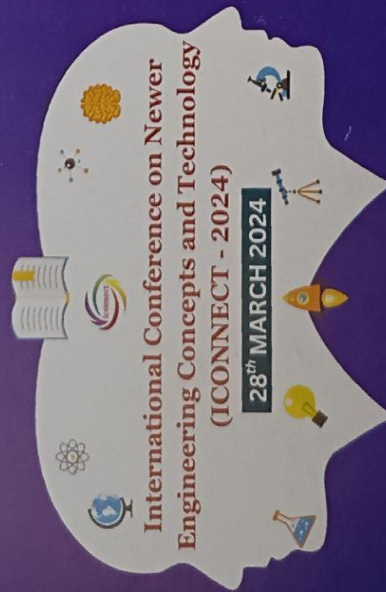

Conference Chair


Principal




K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY Autonomous

Affiliated to Anna University Chennai, Approved by AICTE New Delhi,
ISO 9001:2015, 14001:2015 Certified Institution, Accredited with A grade by NAAC
Samayapuram, Tiruchirappalli - 621 112, Tamilnadu, India.




CERTIFICATE


This is to certify that ANANTHA KRISHNAN . N
has presented a paper entitled SMART
..... LIBRARY MANAGEMENT WITH
..... RECOMMENDATION SYSTEM
in the International Conference on Newer Engineering
Concepts and Technology (ICONNECT-2024), held at
K.Ramakrishnan College of Technology (Autonomous),
Samayapuram, Tiruchirappalli, India on 28.03.2024.

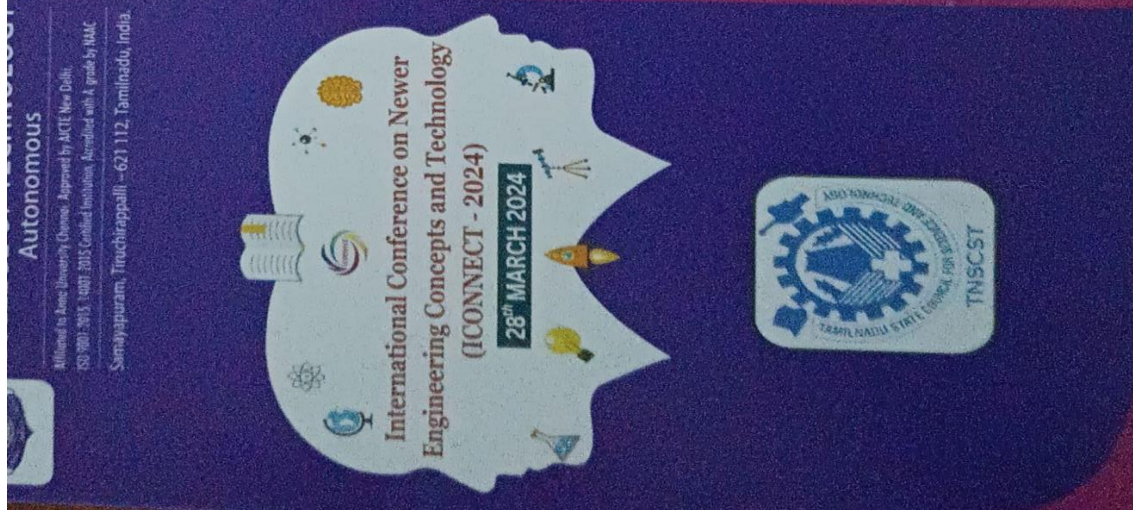

Coordinator




HoD


Principal

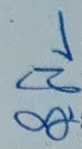

Conference Chair



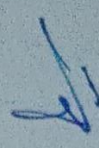
CERTIFICATE

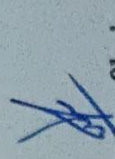
This is to certify that ARJUN . S
has presented a paper entitled SMART
..... LIBRARY MANAGEMENT WITH
..... RECOMMENDATION SYSTEM


in the International Conference on Newer Engineering
Concepts and Technology (ICONNECT-2024), held at
K.Ramakrishnan College of Technology (Autonomous),
Samayapuram, Tiruchirappalli, India on 28.03.2024.


Coordinator




HoD


Conference Chair


Principal