



PROJECT LIBRARY MANAGEMENT

BY:

GROUP - 1

S. M. Intisham Bin Hossain
021 222 010

Md. Rownak Hasan
021 222 003

Jannatul Nayme Saroar
021 222 017

SUBMITTED TO

RAHMAN, KALED MASUKUR, Ph.D.
Professor

INTRODUCTION:

The mini project 'Library Management' is the sample project for managing library. The project aims at developing library management system using the C language that enables an organization to maintain its library. The project demonstrates the creation of a user interface of a system, without the use of C Graphics library. The application uses basic C function to generate menus, show message boxes and print text on the screen. It also effectively applies the various C concepts such as file operations, looping and branching constructs and string manipulation functions.

EXECUTIVE SUMMARY:

The Library Management System project is a comprehensive solution designed to facilitate efficient organization and administration of a library's resources. Developed using the C programming language, the project serves as a functional model for libraries seeking to streamline their operations. This initiative is focused on creating a user-friendly interface without relying on the C Graphics library, showcasing the application of fundamental C functions and concepts.

PROJECT OBJECT:

The key features of library management system are:

- Add books.
- Return books.
- Issue books.
- Delete record of books issued.
- View all issued books records all together and individually.
- Password for administrator to use the system.
- Books are categorized into different departments.
- Search book if it's stored in the library or not.
- Student registration in the library.
- Students can see his/her issued books and also can see the data of books stored in the library.

KEY FEATURES:

The Library Management System project offers essential features for efficient library operations. It simplifies book management by enabling the addition and return of books, as well as issuing and deleting records. The system facilitates easy viewing of issued book records, enhancing monitoring capabilities. Security measures include an administrator password for protected access. Books are categorized by department, and a robust search function allows users to check book availability. Student interaction features include registration, viewing of issued books, and access to the library catalog, promoting transparency and resource awareness. Overall, the project provides a user-friendly solution for streamlined library administration.

USER IMPACT:

The Library Management System project significantly improves the efficiency of library administration. Administrators benefit from streamlined book management, including the addition, return, issuing, and deletion of records. Easy access to issued book records enhances monitoring capabilities. The administrator password ensures secure access, safeguarding sensitive data. The system also simplifies student registration and provides a transparent view of issued books, promoting accountability. Overall, the project enhances the user experience for administrators, offering a user-friendly platform for effective library administration.

FUTURE ENDEAVORS:

Future enhancements for the Library Management System may include advanced search algorithms, data analytics for user insights, and features for digital resource management. Integration with cloud services and mobile accessibility can enhance flexibility. Regular updates to security protocols and user interfaces will ensure ongoing reliability. Collaboration with educational institutions can facilitate broader adaptation. These endeavors aim to align the system with emerging technological trends and evolving administrative needs, ensuring a comprehensive and adaptable solution for the future.

PROJECT GOALS:

- **Efficient Administration:** Streamline library administrative tasks, including book addition, return, issuing, and deletion of records, to enhance overall efficiency.
- **User-Friendly Interface:** Develop a user-friendly system that employs basic C functions for menus, message boxes, and text display, ensuring ease of use for administrators.
- **Functional Diversity:** Provide a range of functionalities, such as secure administrator login, book categorization, and a robust search feature, to cater to diverse library management needs.
- **Data Integrity:** Ensure the integrity of library records by implementing secure file operations, looping, branching constructs, and string manipulation functions for accurate data handling.
- **Student Interaction:** Enable student registration, offer transparent access to issued books, and provide visibility into the library catalog, fostering a seamless interaction between students and the library system.
- **Security Measures:** Implement a robust administrator password system to safeguard sensitive information and maintain the confidentiality of the library's data.
- **Adaptability for Future Trends:** Lay the groundwork for future adaptability by considering potential enhancements, such as advanced search algorithms, data analytics, cloud integration, and mobile accessibility.
- **Collaborative Potential:** Create a platform with the potential for collaboration with educational institutions and libraries, ensuring the system's relevance and applicability across diverse administrative contexts.

CODE:

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
// Function prototype
```

```
void student(); //View Student Portfolio
```

```
void admin(); //View Admin Portfolio
```

```
void addbook(); // Adding Book Function
```

```
void returnbook(); // Function for Return Book
```

```
void issue(); // Funtion for Issued Book
```

```
void view(); //Funtion for see all book list
```

```
void registr(); // Register Student New
```

```
void search(); // Search Book
```

```
void vieweeee();
```

```
void viewcse();
```

```
void datadelet(); // Delete Data
```

```
void record();
```

```
void alldata(); // View Individual Issued Book
```

```
void viewissue(); // Funtion for Issued Book
```

```
void viewallissue(); // View all Issed Book
```

```
struct ST{ // Student Registration Structure
```

```
    char stdname[50];
```

```
    int stdid;
```

```
};
```

```
struct ISS{ // Issued Book Structure
```

```
    int id;
```

```
    char name[50];
```

```
    char bk[50];
```

```

    char isdate[50];
};

struct SI{ //Add Book Structure

    int id;
    char book[100];
    char author[100];
    char dept[20];
    char self[10];
    char date[40];
};

void viewallissue() {
    system("cls");
    struct ISS s; // Use the correct structure name here
    FILE *fPtr;

    fPtr = fopen("issue_bin.dat", "rb");
    if (fPtr == NULL) {
        printf("Error opening file for reading.\n");
        return;
    }

    while (fread(&s, sizeof(struct ISS), 1, fPtr) == 1) {
        if (s.id != 0) { // Assuming '0' represents an empty record
            printf("\nID=%d, Name=%s, Book=%s, Date=%s\n", s.id, s.name, s.bk, s.isdate);
        }
    }

    fclose(fPtr);
}

```

```
}
```

```
void viewissue(){  
    system("cls");  
    int n;  
    printf("\n 1| Show All Issued Book ");  
    printf("\n 2| Show Individual Data ");  
    printf("\n");  
    scanf("%d",&n);  
    if(n==2) alldata();  
    if(n==1) viewallissue();  
}
```

```
int main() {  
    struct ST newStudent;  
    FILE *fle;  
    fle = fopen("student_bin.dat", "rb");  
    int n;  
    char id[10];  
    char pass[] = "weq123Admin";  
    char user[20];  
  
    printf("||||||||||||||||||||| Welcome to RIN G1 Library  
|||||||||||||||||||||\n");  
    printf("          1) Admin\n");  
    printf("          2) Student\n");  
    printf("          3) Student Registration\n");  
    scanf("%d", &n);  
  
    if (n == 2) {
```

```

printf("\nEnter Student ID :");
scanf("%s", id);

int found = 0;

// Loop until the end of the file or until the Student is found
while (fread(&newStudent, sizeof(struct ST), 1, fle) == 1) {
    // Check if the student ID matches
    if (newStudent.stdid == atoi(id)) {
        student();
        found = 1;
        break;
    }
}

fclose(fle);

if (found == 0) {
    printf("Student not found!\n");
}
} else if (n == 1) {
    printf("Enter Password : ");
    scanf("%s", user);

    if (strcmp(user, pass) == 0) {
        admin();
    } else {
        printf("Incorrect Password !!!!!");
    }
}

```

```

    } else if (n == 3) {
        registr();
    }

    return 0;
}

void student(){
    int dec;
    while(1){
        printf("\n");
        printf(" (1) View Book \n");
        printf(" (2) Issued Book \n");
        printf(" (3) Exist \n");
        scanf("%d",&dec);
        if(dec==1) view();
        else if(dec==2) record();
        else if(dec==3) break;
    }
}

void registr() {
    struct ST new = {0, ""};
    FILE *newstd;

    newstd = fopen("student_bin.dat", "rb+");

    if (newstd == NULL) {
        newstd = fopen("student_bin.dat", "wb+");
        for (int i = 0; i < 1000; i++) { // Corrected the comment

```



```
        fwrite(&new, sizeof(struct ST), 1, newstd);  
    }  
}
```

```
printf("\nEnter Student ID : ");  
scanf("%d", &new.stdid);  
printf("\nEnter Student Name : ");  
scanf("%s", new.stdname);
```

```
fseek(newstd, (new.stdid % 1000 - 1) * sizeof(struct ST), SEEK_SET);  
fwrite(&new, sizeof(struct ST), 1, newstd);  
printf("\nRegistration Successful!\n");
```

```
fclose(newstd);  
}
```

```
void admin() {
```

```
    int n;
```

```
    system("cls");
```

//This code uses preprocessor directives to include different commands based on the operating system.
Let's break it down://

```
    while(1){  
        printf("\n");  
        printf(" 1 | Add Book\n");  
        printf(" 2 | View Book List\n");  
        printf(" 3 | Return Book\n");  
        printf(" 4 | Issue Book\n");
```

```

printf(" 5 | Search Book\n");
printf(" 6 | View record of issued books\n");
printf(" 7 | Delete record from Book List\n");
printf(" 8 | Exit\n");
scanf("%d",&n);
if(n==8) break;
else if(n==1) addbook();
else if(n==2) view();
else if(n==3) returnbook();
else if(n==4) issue();
else if(n==7) datadelet();
else if(n==6) viewissue();
else if(n==5) search();
}
}
void alldata(){
system("cls");
int nm;
struct ISS man;
FILE *isst;

isst = fopen("issue_bin.dat", "rb");
if (isst == NULL) {
    printf("Error opening file for reading.\n");
    return;
}

printf("Enter Student ID : ");
scanf("%d", &nm);

```

```

int found = 0;

// Loop until the end of the file or until the book is found
while (!feof(isst)) {
    fread(&man, sizeof(struct ISS), 1, isst);

    // Check if the book name matches
    if (man.id==nm) {
        printf("Book Found !!!\n");

        printf("=====
=====\\n");

        printf("| %-10s | %-30s | %-20s | %-15s \\n", "Student ID", "Student Name", "Book Name", "Issued
Date");

        printf("=====
=====\\n");

        printf("| %-10d | %-30s | %-20s | %-15s \\n",
            man.id, man.name, man.bk, man.isdate);
        found = 1;
        break;
    }

}

fclose(isst);

if (found==0) {

```

```
    printf("Book not found!\n");  
}  
}
```

```
void record() {  
    system("cls");  
    int nm;  
    struct ISS man;  
    FILE *isst;  
  
    isst = fopen("issue_bin.dat", "rb");  
    if (isst == NULL) {  
        printf("Error opening file for reading.\n");  
        return;  
    }  
}
```

```
printf("Enter Student ID : ");  
scanf("%d", &nm);
```

```
int found = 0;
```

```
// Loop until the end of the file or until the book is found
```

```
while (!feof(isst)) {  
    fread(&man, sizeof(struct ISS), 1, isst);
```

```
// Check if the book name matches
```

```
if (man.id==nm) {  
    printf("Book Found !!!!\n");
```

```
printf("=====
=====\\n");
```

```
printf("| %-10s | %-30s | %-20s | %-15s \\n", "Student ID", "Student Name", "Book Name", "Issued
Date");
```

```
printf("=====
=====\\n");
```

```
printf("| %-10d | %-30s | %-20s | %-15s \\n",
        man.id, man.name, man.bk, man.isdate);
```

```
found = 1;
```

```
break;
```

```
}
```

```
}
```

```
fclose(isst);
```

```
if (found==0) {
```

```
printf("Book not found!\\n");
```

```
}
```

```
}
```

```
void returnbook() {
```

```
system("cls");
```

```
struct ISS book;
```

```
FILE *dlt;
```

```
int n;
```

```
printf("Enter Student ID: ");
scanf("%d", &n);

dlt = fopen("issue_bin.dat", "rb+");
if (dlt == NULL) {
    printf("Error opening file for reading.\n");
    return;
}

int found = 0;

// Loop until the end of the file
while (fread(&book, sizeof(struct ISS), 1, dlt) == 1) {
    // Check if the student ID matches
    if (book.id == n) {
        fseek(dlt, -sizeof(struct ISS), SEEK_CUR);
        struct ISS emptyBook = {0, "", "", "", ""};
        fwrite(&emptyBook, sizeof(struct ISS), 1, dlt);
        found = 1;
        break;
    }
}

fclose(dlt);

if (found == 0) {
    printf("Student not issued any book!\n");
} else {
```

```
        printf("Return Successful!\n");
    }
}
```

```
void search() {
    system("cls");
    char nm[100];
    struct SI book;
    FILE *src;

    src = fopen("book_bin.dat", "rb");
    if (src == NULL) {
        printf("Error opening file for reading.\n");
        return;
    }
```

```
    printf("Enter Book Name : ");
    scanf("%s", nm);
```

```
    int found = 0;
```

```
    // Loop until the end of the file or until the book is found
```

```
    while (!feof(src)) {
        fread(&book, sizeof(struct SI), 1, src);
```

```
        // Check if the book name matches
```

```
        if (strcmp(book.book, nm) == 0) {
            printf("Book Found !!!!\n");
```

```
printf("=====
=====\\n");
```

```
printf("| %-10s | %-30s | %-20s | %-15s | %-10s | %-15s |\\n", "Book ID", "Book Name", "Author",
"Department", "Self No", "Date");
```

```
printf("=====
=====\\n");
```

```
printf("| %-10d | %-30s | %-20s | %-15s | %-10s | %-15s |\\n",
        book.id, book.book, book.author, book.dept, book.self, book.date);
```

```
found = 1;
```

```
break;
```

```
}
```

```
}
```

```
fclose(src);
```

```
if (found==0) {
```

```
printf("Book not found!\\n");
```

```
}
```

```
}
```

```
void view(){
```

```
system("cls");
```

```
int ans;
```

```
printf(" 1] Electrical and Electronic Engineering\\n");
```

```
printf(" 2] Computer Science and Engineering\\n");
```



```

printf(" 3] Bachelor of Business Administration\n");
printf(" 4] Civil Engineering\n");
printf(" 5] Pharmacy\n");
scanf("%d",&ans);
if(ans==1)vieweeee();
else if(ans==2)viewcse();
else if(ans==3)viewbba();
else if(ans==4)viewce();
else if(ans==5)viewpharm();
}

void addbook() {
    struct SI book = {0, "", "", ""};
    FILE *fptr;

    if (fptr == NULL) {
        fptr = fopen("book_bin.dat", "wb+");
        //Assume Varsity has 5 department and 400 book can store each department//
        for (int i = 0; i < 2000; i++) {
            fwrite(&book, sizeof(struct SI), 1, fptr);
        }
    }
    if(fptr != NULL){
        system("cls");
        fptr=fopen("book_bin.dat","rb+");
        printf("\nEnter Book ID :");
        scanf("%d",&book.id);
        printf("\nEnter Book Name :");
        scanf("%s",book.book);
    }
}

```

```

    printf("\nEnter Author Name :");
    scanf("%s",book.author);
    printf("\nEnter Department :");
    scanf("%s",book.dept);
    printf("\nEnter Self No :");
    scanf("%s",book.self);
    printf("\nEnter Date :");
    scanf("%s",book.date);
    fseek(fptr,(book.id-1)*sizeof(struct SI),SEEK_SET);
    fwrite(&book,sizeof(struct SI),1,fptr);
}

fclose(fptr);
}

void vieweeee() {
    system("cls");
    FILE *viu;
    struct SI book;
    viu = fopen("book_bin.dat", "rb");

    printf("=====
    =====\n");

    printf("| %-10s | %-30s | %-20s | %-15s | %-10s | %-15s |\n", "Book ID", "Book Name", "Author",
    "Department", "Self No", "Date");

    printf("=====
    =====\n");

    int startRecord = 0; // Adjust the start record as needed

```

```

int endRecord = 399; // Adjust the end record as needed

// Move the file pointer to the start record
fseek(viu, (startRecord - 1) * sizeof(struct SI), SEEK_SET);

// Read and display records within the specified range
for (int i = startRecord; i <= endRecord && fread(&book, sizeof(struct SI), 1, viu) == 1; i++) {
    if (book.id != 0) {
        printf("| %-10d | %-30s | %-20s | %-15s | %-10s | %-15s |\n",
               book.id, book.book, book.author, book.dept, book.self, book.date);
    }
}

printf("=====
=====\\n");

fclose(viu);
}

void viewwcse() {
    system("cls");
    FILE *viu;
    struct SI book;
    viu = fopen("book_bin.dat", "rb");

    printf("=====
=====\\n");

    printf("| %-10s | %-30s | %-20s | %-15s | %-10s | %-15s |\n", "Book ID", "Book Name", "Author",
"Department", "Self No", "Date");

```

```
printf("=====
=====\\n");
```

```
int startRecord = 400; // Adjust the start record as needed
```

```
int endRecord = 799; // Adjust the end record as needed
```

```
// Move the file pointer to the start record
```

```
fseek(viu, (startRecord - 1) * sizeof(struct SI), SEEK_SET);
```

```
// Read and display records within the specified range
```

```
for (int i = startRecord; i <= endRecord && fread(&book, sizeof(struct SI), 1, viu) == 1; i++) {
```

```
    if (book.id != 0) {
```

```
        printf(" | %-10d | %-30s | %-20s | %-15s | %-10s | %-15s |\\n",
```

```
               book.id, book.book, book.author, book.dept, book.self, book.date);
```

```
    }
```

```
}
```

```
printf("=====
=====\\n");
```

```
fclose(viu);
```

```
}
```

```
void viewpharm() {
```

```
    system("cls");
```

```
    FILE *viu;
```

```
    struct SI book;
```

```
    viu = fopen("book_bin.dat", "rb");
```

```
printf("=====  
=====\\n");
```

```
    printf("| %-10s | %-30s | %-20s | %-15s | %-10s | %-15s |\\n", "Book ID", "Book Name", "Author",  
"Department", "Self No", "Date");
```

```
printf("=====  
=====\\n");
```

```
int startRecord = 1600; // Adjust the start record as needed
```

```
int endRecord = 2000; // Adjust the end record as needed
```

```
// Move the file pointer to the start record
```

```
fseek(viu, (startRecord - 1) * sizeof(struct SI), SEEK_SET);
```

```
// Read and display records within the specified range
```

```
for (int i = startRecord; i <= endRecord && fread(&book, sizeof(struct SI), 1, viu) == 1; i++) {
```

```
    if (book.id != 0) {
```

```
        printf("| %-10d | %-30s | %-20s | %-15s | %-10s | %-15s |\\n",
```

```
            book.id, book.book, book.author, book.dept, book.self, book.date);
```

```
    }
```

```
}
```

```
printf("=====  
=====\\n");
```

```
fclose(viu);
```

```
}
```

```
void viewce() {
```

```
    system("cls");
```

```

FILE *viu;

struct SI book;

viu = fopen("book_bin.dat", "rb");


printf("=====
=====\\n");

    printf("| %-10s | %-30s | %-20s | %-15s | %-10s | %-15s |\\n", "Book ID", "Book Name", "Author",
"Department", "Self No", "Date");

printf("=====
=====\\n");


int startRecord = 1200; // Adjust the start record as needed
int endRecord = 1599;  // Adjust the end record as needed


// Move the file pointer to the start record
fseek(viu, (startRecord - 1) * sizeof(struct SI), SEEK_SET);


// Read and display records within the specified range
for (int i = startRecord; i <= endRecord && fread(&book, sizeof(struct SI), 1, viu) == 1; i++) {
    if (book.id != 0) {
        printf("| %-10d | %-30s | %-20s | %-15s | %-10s | %-15s |\\n",
            book.id, book.book, book.author, book.dept, book.self, book.date);
    }
}

printf("=====
=====\\n");

```

```

    fclose(viu);
}

void viewbba() {
    system("cls");

    FILE *viu;

    struct SI book;

    viu = fopen("book_bin.dat", "rb");

    printf("=====
=====\\n");

    printf("| %-10s | %-30s | %-20s | %-15s | %-10s | %-15s |\\n", "Book ID", "Book Name", "Author",
    "Department", "Self No", "Date");

    printf("=====
=====\\n");

    int startRecord = 800; // Adjust the start record as needed
    int endRecord = 1199; // Adjust the end record as needed

    // Move the file pointer to the start record
    fseek(viu, (startRecord - 1) * sizeof(struct SI), SEEK_SET);

    // Read and display records within the specified range
    for (int i = startRecord; i <= endRecord && fread(&book, sizeof(struct SI), 1, viu) == 1; i++) {
        if (book.id != 0) {
            printf("| %-10d | %-30s | %-20s | %-15s | %-10s | %-15s |\\n",
                book.id, book.book, book.author, book.dept, book.self, book.date);
        }
    }
}

```

```
printf("=====  
=====\\n");
```

```
fclose(viu);  
}
```

```
void datadelet(){
```

```
    system("cls");
```

```
    char nm[100];
```

```
    struct SI book;
```

```
    FILE *dlt;
```

```
    dlt = fopen("book_bin.dat", "rb+");
```

```
    if (dlt == NULL) {
```

```
        printf("Error opening file for reading.\\n");
```

```
        return;
```

```
    }
```

```
    printf("Enter Book Name : ");
```

```
    scanf("%s", nm);
```

```
    int found = 0;
```

```
    // Loop until the end of the file or until the book is found
```

```
    while (!feof(dlt)) {
```

```
        fread(&book, sizeof(struct SI), 1, dlt);
```

```
        // Check if the book name matches
```



```

if (strcmp(book.book, nm) == 0) {
    fseek(dlt, -sizeof(struct SI), SEEK_CUR);
    struct SI book={0, "", "", "", "", ""};
    fwrite(&book,sizeof(struct SI),1,dlt);
    found = 1;
    break;
}

```

```

}

```

```

fclose(dlt);

```

```

if (found==0) {
    printf("Issued Book not found!\n");
}
else{
    printf("Book Data Remove Succesfully!!!!");
}
}

```

```

void issue() {
    system("cls");
    char nm[100];
    struct SI book;
    struct ISS isbook;
    FILE *src;

    src = fopen("book_bin.dat", "rb");

```

```

if (src == NULL) {
    printf("Error opening file for reading.\n");
    return;
}

printf("Enter Book Name : ");
scanf("%s", nm);

int found = 0;

// Loop until the end of the file or until the book is found
while (!feof(src)) {
    fread(&book, sizeof(struct SI), 1, src);

    // Check if the book name matches
    if (strcmp(book.book, nm) == 0) {
        // issue book file
        struct ISS isbook = {0, "", "", "", ""};
        FILE *fptr;

        fptr = fopen("issue_bin.dat", "rb+");
        if (fptr == NULL) {
            fptr = fopen("issue_bin.dat", "wb+");
            // do it for 2000 student
            for (int i = 0; i < 2000; i++) {
                fwrite(&isbook, sizeof(struct ISS), 1, fptr);
            }
        }
        if (fptr != NULL) {

```

```

        system("cls");

        fptr = fopen("issue_bin.dat", "rb+");

        printf("\nEnter Student ID :");

        scanf("%d", &isbook.id);

        printf("\nEnter Student Name :");

        scanf("%s", isbook.name);

        printf("\nEnter Book Name :");

        scanf("%s", isbook.bk);

        printf("\nEnter Issue Date :");

        scanf("%s", isbook.isdate);

        fseek(fptr, ((isbook.id%1000) - 1) * sizeof(struct ISS), SEEK_SET);

        fwrite(&isbook, sizeof(struct ISS), 1, fptr);
    }

    fclose(fptr);

    found = 1;

    break;
}

}

fclose(src);

if (found == 0) {
    printf("Book not found!\n");
}

}

```

OUTPUT:

- Home page:

```
||||| Welcome to RIN G1 Library |||||
1) Admin
2) Student
3) Student Registration
|
```

- Password for administrator to use the system.

```
||||| Welcome to RIN G1 Library |||||
1) Admin
2) Student
3) Student Registration
1
Enter Password : weq123Admin|
```

- Only Administration can use this site and to enter this site, admin needs password:

```
1 | Add Book
2 | View Book List
3 | Return Book
4 | Issue Book
5 | Search Book
6 | View record of issued books
7 | Delete record from Book List
8 | Exit
|
```

- Select 1 to store new books in the library

```
Enter Book ID :123

Enter Book Name :Zoology

Enter Author Name :Nayeem

Enter Department :Pharmacy

Enter Self No :s-322-r

Enter Date :12/12/2023
```

- **Select 2 to see the stored book list**

```
1] Electrical and Electronic Engineering
2] Computer Science and Engineering
3] Bachelor of Business Administration
4] Civil Engineering
5] Pharmacy
```

Here, needs to select the department to see the departmental books.

After select the department

Book ID	Book Name	Author	Department	Self No	Date
1717	Zoology	Nayeem	Pharmacy	s-12-l	12/12/12
1899	Microbiology	Alexender	Pharmacy	S-6-R	13/12/2023

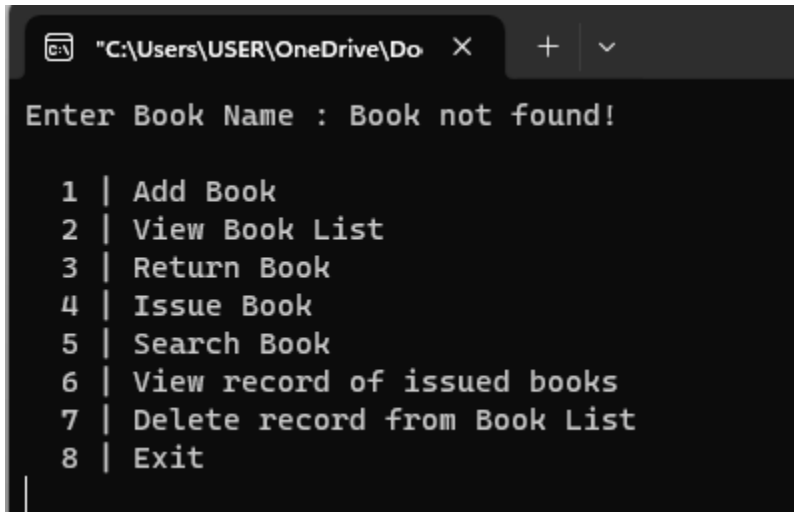
We can see that the departmental books and the writer's name of the book and what is self-number of these books.

- **Select 5 to search the desired book and it shows the information of the particular selected book**

```
Enter Book Name : Physics
Book Found !!!!
```

Book ID	Book Name	Author	Department	Self No	Date
101	Physics	J.Smith	EEE	S-1-L	02/02/2019

If any book is not founded then showing screen pops-out



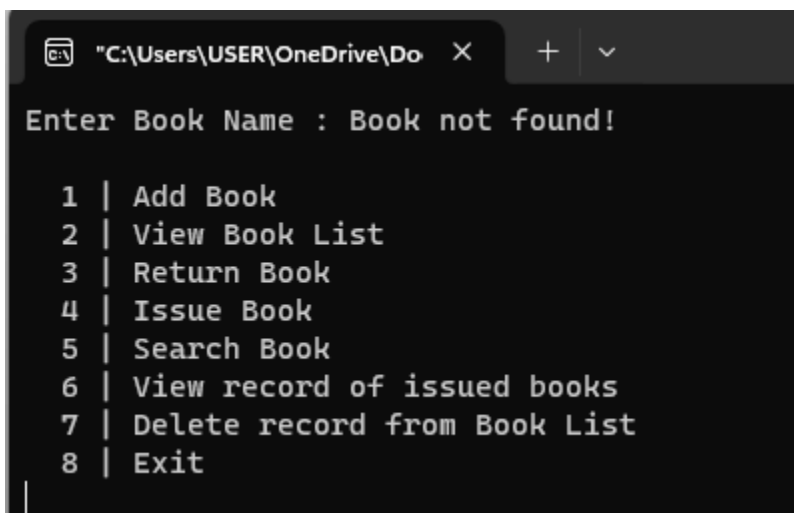
```
"C:\Users\USER\OneDrive\Do  X  +  v
Enter Book Name : Book not found!

1 | Add Book
2 | View Book List
3 | Return Book
4 | Issue Book
5 | Search Book
6 | View record of issued books
7 | Delete record from Book List
8 | Exit
```

- To issue any book, administration needs to select 4

If any student's desired books are already stored in the library, then it will ask for students ID and name but the student needs to be registered before in the library.

If the student has already been registered.



```
"C:\Users\USER\OneDrive\Do  X  +  v
Enter Book Name : Book not found!

1 | Add Book
2 | View Book List
3 | Return Book
4 | Issue Book
5 | Search Book
6 | View record of issued books
7 | Delete record from Book List
8 | Exit
```

- Select option 3, if any student return the issued book and it will remove the records of that student

```
Enter Student ID: 222010
Return Successful!

1 | Add Book
2 | View Book List
3 | Return Book
4 | Issue Book
5 | Search Book
6 | View record of issued books
7 | Delete record from Book List
8 | Exit
```

- Select 7 to delete book from book list

```
Enter Book Name : Physics
Book Data Remove Succesfully!!!!

1 | Add Book
2 | View Book List
3 | Return Book
4 | Issue Book
5 | Search Book
6 | View record of issued books
7 | Delete record from Book List
8 | Exit
```

- Select 6, to see the records of issued books. Administration can see all the records all together or individually

```
1| Show All Issued Book
2| Show Individual Data
```

Now if administer select 1:

```
ID=222003, Name=Rownak, Book=Physics, Date=12/12/2023
ID=222017, Name=Nayeem, Book=Circuit_1, Date=19/12/2023
ID=212100, Name=Khairul, Book=Physics, Date=12/12/2020
```

Here all the records together.

If administer select 2:

```
Enter Student ID : 222003
Book Found !!!!
=====
| Student ID | Student Name           | Book Name           | Issued Date
=====
| 222003     | Rownak                 | Physics             | 12/12/2023
```

Here administer can see the records of individual student.

Back to home page and 2 for students means they can access this site. Mainly it's for students and they can see and moderate their activity from this site.

```
||||| Welcome to RIN G1 Library |||||
1) Admin
2) Student
3) Student Registration
|
```

- Students need to enter their ID to get entry in this site

```
||||| Welcome to RIN G1 Library |||||
1) Admin
2) Student
3) Student Registration
2
Enter Student ID :222003|
```


Then 3 options will pop-up:

```
||||| Welcome to RIN G1 Library |||||
1) Admin
2) Student
3) Student Registration

2

Enter Student ID :222003

(1) View Book
(2) Issued Book
(3) Exist
```

- Option 1, to see the existing books in the library

```
1] Electrical and Electronic Engineering
2] Computer Science and Engineering
3] Bachelor of Business Administration
4] Civil Engineering
5] Pharmacy
```

Here student need to choose their department.

Suppose a student chose EEE department

=====					
Book ID	Book Name	Author	Department	Self No	Date
=====					
9	Circuit_1	Sadiku	EEE	S-2-R	12/12/2022
89	Circuit_2	Boylstate	EEE	S-2-L	12/12/2021
=====					

From here, he/she can see the existing books from his/her department.

- Option 2 is to see the issued book the student has.

```
Enter Student ID : 222003
Book Found !!!!

=====
| Student ID | Student Name | Book Name | Issued Date |
=====
| 222003 | Rownak | Physics | 12/12/2023 |
=====
```

Back to home page. Option 3 is to register new student in the library

```
||||| Welcome to RIN G1 Library |||||
1) Admin
2) Student
3) Student Registration
3|
```

- After selecting option 3, it will ask for new student's name and id. After the input these requirements the student will be registered

```
Enter Student ID : 222100
Enter Student Name : Zarir Adib
Registration Successful!
```

CONCLUSION:

The Library Management System project stands as a testament to the successful realization of its goals in streamlining library administration. With a focus on efficiency, the system facilitates seamless book management, user-friendly interfaces, and robust security measures. The incorporation of diverse functionalities, including student interaction features, contributes to an enriched user experience. As the project lays the foundation for future adaptability, considering advancements like advanced search algorithms and data analytics, it positions itself as a dynamic solution. The collaborative potential ensures its relevance across various administrative contexts. In conclusion, the Library Management System not only meets current needs but also anticipates and prepares for the evolving landscape of library administration.

