Project library management

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| SNO | FIGURE NAME | PAGE NO |
| 1 | Fig 3.1 Stack |  |
| 2 | Fig 6.1:Output 1 |  |
| 3 | Fig 5.2:Output 2 |  |
| 4 | Fig 5.3:Output 3 |  |
| 5 | Fig 5.4:Output 4 |  |
| 6 | Fig 5.5:Output 5 |  |
| 7 | Fig 5.6:Output 6 |  |
| 8 | Fig 5.7:Output 7 |  |
| 9 | Fig 5.8:Output 8 |  |

# PROBLEM DEFINITION

**Library Management System**

## About Library Management System Project in C:

For library management, this project considers six departments – Computer, Electrical, Civil, Electronics, Mechanical and Architecture. These departments work simultaneously with the operations mentioned above. You can add a book to the Civil section, delete a book from the Electrical section or view issued book details of the Mechanical department.

These are the **functions used** in this mini project:

**voidmainmenu(void)** – This function is used to display the main menu of this project. Scroll down to Output to view the main menu.

**voidreturnfunc(void)**– Inside this function, the main menu function (mentioned above) is called when the user presses a key. With this, the user can return back to the main menu.

**voidaddbooks(void)**– This function adds books in a file. For that, you need to mention the department to which you want to add the book. The record is saved in a file. And, it is similar for the following functions as well.

**void deletebooks(void)**  
**void editbooks(void)**  
**void searchbooks(void)**  
**void issuebooks(void)**  
**void viewbooks(void)**

**voidissuerecord()** – With this function, you can keep record of the [student](https://www.codewithc.com/mini-project-in-c-student-record-system/) to whom the book has been issued.

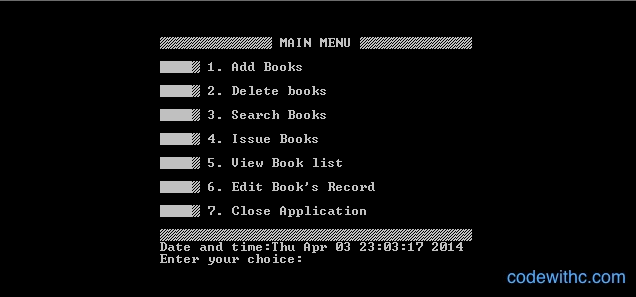
**voidcloseapplication(void)** – This function is for closing the application.

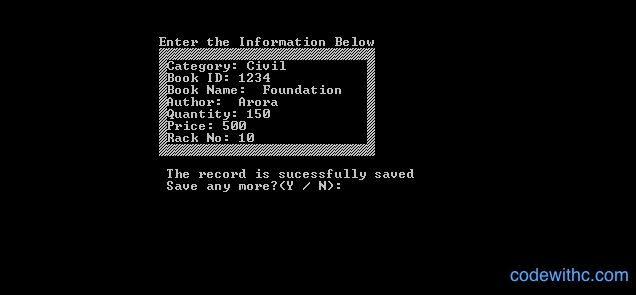
**int  getdata()** – This function asks for data input from the user.

**intcheckid(int)** – This function is used to check whether the ID of a book entered by a user exists in file or not.

**void Password()** – Due to this function, the user is asked to input password to run the application after it is opened. You can’t change the password upon running the application.

***Sample Window:***





Use appropriate data structure to implement this.

# REQUIREMENT SPECIFICATION

**2.1 HARDWARE REQUIREMENTS**

Processor : Core i3/ i5/ i7

RAM : 128 MB

Hard Disk : 20GB

Monitor : 15” Color monitor

Key Board : 104 Keys

**2.2 SOFTWARE REQUIREMENTS**

Operating System : Windows 7/8.

Language : C ++

Software : Turbo Cpp compiler

# ALGORITHM DESIGN

4.1 ALGORITHM

void addBookInDataBase()

fp = fopen(FILE\_NAME,"ab+");

if(fp == NULL)

print(File is not opened)

exit(1)

addBookInfoInDataBase.books\_id

do

addBookInfoInDataBase.bookName

if (!status)

print(Name contain invalid character. Please enter again)

while(!status)

do

addBookInfoInDataBase.authorName

if (!status)

print(Name contain invalid character. Please enter again)

while(!status)

do

addBookInfoInDataBase.studentName

if (!status)

print(Name contain invalid character. Please enter again)

while(!status)

do

addBookInfoInDataBase.bookIssueDate

if (!status)

print(Name contain invalid character. Please enter again)

while(!status)

void searchBooks()

fp = fopen(FILE\_NAME,"rb")

if(fp == NULL)

printf(File is not opened)

exit(1)

if (fseek(fp,FILE\_HEADER\_SIZE,SEEK\_SET) != 0)

fclose(fp)

print(Facing issue while reading file)

exit(1)

printf(Enter Book Name to search)

fflush(stdin)

fgets(bookName,MAX\_BOOK\_NAME,stdin)

while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))

if(!strcmp(addBookInfoInDataBase.bookName, bookName))

found = 1

break

if(found)

printf(addBookInfoInDataBase.books\_id)

printf(addBookInfoInDataBase.bookName)

printf(addBookInfoInDataBase.authorName)

printf(addBookInfoInDataBase.bookIssueDate.dd,

addBookInfoInDataBase.bookIssueDate.mm,

addBookInfoInDataBase.bookIssueDate.yyyy)

void viewBooks()

fp = fopen(FILE\_NAME,"rb")

if(fp == NULL)

printf(File is not opened)

exit(1)

if (fseek(fp,FILE\_HEADER\_SIZE,SEEK\_SET) != 0)

fclose(fp)

printf(Facing issue while reading file)

exit(1)

while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))

printf(countBook)

printf(addBookInfoInDataBase.books\_id)

printf(addBookInfoInDataBase.bookName)

printf(addBookInfoInDataBase.authorName)

printf(addBookInfoInDataBase.bookIssueDate.dd,

addBookInfoInDataBase.bookIssueDate.mm, addBookInfoInDataBase.bookIssueDate.yyyy)

if(!found)

printf(No Record)

printf(Press any key to go to main menu)

void deleteBooks()

fp = fopen(FILE\_NAME,"rb")

if(fp == NULL)

printf(File is not opened)

exit(1)

tmpFp = fopen("tmp.bin","wb")

if(tmpFp == NULL)

fclose(fp)

printf("File is not opened)

exit(1)

fread (fileHeaderInfo)

fwrite(fileHeaderInfo)

while (fread(addBookInfoInDataBase, sizeof(addBookInfoInDataBase))

if(addBookInfoInDataBase.books\_id != bookDelete)

fwrite(&addBookInfoInDataBase,sizeof(addBookInfoInDataBase))

else

found = 1

(found)? print(Record deleted successfully):print(Record not found)

remove(FILE\_NAME)

rename(FILE\_NAME)

void updateCredential(void)

fp = fopen(FILE\_NAME,"rb+");

if(fp == NULL)

printf(File is not opened)

exit(1)

fread (fileHeaderInfo)

if (fseek(fp,0,SEEK\_SET) != 0)

fclose(fp)

printf(Facing issue while updating password)

exit(1)

printf(New Username)

fflush(stdin)

fgets(userName)

printf(New Password)

fflush(stdin)

fgets(password)

strncpy(fileHeaderInfo.username)

strncpy(fileHeaderInfo.password)

fwrite(fileHeaderInfo)

fclose(fp)

printf(Your Password has been changed successfully)

printf(Login Again)

fflush(stdin)

getchar()

exit(1)

}

# SOURCE CODE

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#define MAX\_YR 9999

#define MIN\_YR 1900

#define MAX\_SIZE\_USER\_NAME 30

#define MAX\_SIZE\_PASSWORD 20

#define FILE\_NAME "KCEbooks.bin"

#define MAX\_BOOK\_NAME 50

#define MAX\_AUTHOR\_NAME 50

#define MAX\_STUDENT\_NAME 50

#define MAX\_STUDENT\_ADDRESS 300

#define FILE\_HEADER\_SIZE sizeof(sFileHeader)

//structure to store date

typedef struct

{

int yyyy;

int mm;

int dd;

} Date;

typedef struct

{

char username[MAX\_SIZE\_USER\_NAME];

char password[MAX\_SIZE\_PASSWORD];

} sFileHeader;

typedef struct// to call in program

{

unsigned int books\_id; // declare the integer data type

char bookName[MAX\_BOOK\_NAME];// declare the character data type

char authorName[MAX\_AUTHOR\_NAME];// declare the charecter data type

char studentName[MAX\_STUDENT\_NAME];// declare the character data type

char studentAddr[MAX\_STUDENT\_ADDRESS];// declare the character data type

Date bookIssueDate;// declare the integer data type

} s\_BooksInfo;

void printMessageCenter(const char\* message)

{

int len =0;

int pos = 0;

//calculate how many space need to print

len = (78 - strlen(message))/2;

printf("\t\t\t");

for(pos =0 ; pos < len ; pos++)

{

//print space

printf(" ");

}

//print message

printf("%s",message);

}

void headMessage(const char \*message)

{

printf("\n\t\t\t###########################################################################");

printf("\n\t\t\t############ ############");

printf("\n\t\t\t############ Library management System Mini Project ############");

printf("\n\t\t\t############ ############");

printf("\n\t\t\t###########################################################################");

printf("\n\t\t\t---------------------------------------------------------------------------\n");

printMessageCenter(message);

printf("\n\t\t\t----------------------------------------------------------------------------");

}

void welcomeMessage()

{

headMessage("KCE Coimbatore");

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

printf("\n\t\t\t \*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*\n");

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

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

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

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

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

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

printf("\n\t\t\t = of KCE =");

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

printf("\n\t\t\t \*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*-\*\*\n");

printf("\n\n\n\t\t\t Enter any key to continue.....");

getch();

}

int isNameValid(const char \*name)

{

int validName = 1;

int len = 0;

int index = 0;

len = strlen(name);

for(index =0; index <len ; ++index)

{

if(!(isalpha(name[index])) && (name[index] != '\n') && (name[index] != ' '))

{

validName = 0;

break;

}

}

return validName;

}

// Function to check leap year.

//Function returns 1 if leap year

int IsLeapYear(int year)

{

return (((year % 4 == 0) &&

(year % 100 != 0)) ||

(year % 400 == 0));

}

// returns 1 if given date is valid.

int isValidDate(Date \*validDate)

{

//check range of year,month and day

if (validDate->yyyy > MAX\_YR ||

validDate->yyyy < MIN\_YR)

return 0;

if (validDate->mm < 1 || validDate->mm > 12)

return 0;

if (validDate->dd < 1 || validDate->dd > 31)

return 0;

//Handle feb days in leap year

if (validDate->mm == 2)

{

if (IsLeapYear(validDate->yyyy))

return (validDate->dd <= 29);

else

return (validDate->dd <= 28);

}

//handle months which has only 30 days

if (validDate->mm == 4 || validDate->mm == 6 ||

validDate->mm == 9 || validDate->mm == 11)

return (validDate->dd <= 30);

return 1;

}

// Add books in list

void addBookInDataBase()

{

int days;

s\_BooksInfo addBookInfoInDataBase = {0};

FILE \*fp = NULL;

int status = 0;

fp = fopen(FILE\_NAME,"ab+");

if(fp == NULL)

{

printf("File is not opened\n");

exit(1);

}

headMessage("ADD NEW BOOKS");

printf("\n\n\t\t\tENTER YOUR DETAILS BELOW:");

printf("\n\t\t\t---------------------------------------------------------------------------\n");

printf("\n\t\t\tBook ID NO = ");

fflush(stdin);

scanf("%u",&addBookInfoInDataBase.books\_id);

do

{

printf("\n\t\t\tBook Name = ");

fflush(stdin);

fgets(addBookInfoInDataBase.bookName,MAX\_BOOK\_NAME,stdin);

status = isNameValid(addBookInfoInDataBase.bookName);

if (!status)

{

printf("\n\t\t\tName contain invalid character. Please enter again.");

}

}

while(!status);

do

{

printf("\n\t\t\tAuthor Name = ");

fflush(stdin);

fgets(addBookInfoInDataBase.authorName,MAX\_AUTHOR\_NAME,stdin);

status = isNameValid(addBookInfoInDataBase.authorName);

if (!status)

{

printf("\n\t\t\tName contain invalid character. Please enter again.");

}

}

while(!status);

do

{

printf("\n\t\t\tStudent Name = ");

fflush(stdin);

fgets(addBookInfoInDataBase.studentName,MAX\_STUDENT\_NAME,stdin);

status = isNameValid(addBookInfoInDataBase.studentName);

if (!status)

{

printf("\n\t\t\tName contain invalid character. Please enter again.");

}

}

while(!status);

do

{

//get date year,month and day from user

printf("\n\t\t\tEnter date in format (day/month/year): ");

scanf("%d/%d/%d",&addBookInfoInDataBase.bookIssueDate.dd,&addBookInfoInDataBase.bookIssueDate.mm,&addBookInfoInDataBase.bookIssueDate.yyyy);

//check date validity

status = isValidDate(&addBookInfoInDataBase.bookIssueDate);

if (!status)

{

printf("\n\t\t\tPlease enter a valid date.\n");

}

}

while(!status);

fwrite(&addBookInfoInDataBase,sizeof(addBookInfoInDataBase), 1, fp);

fclose(fp);

}

// search books

void searchBooks()

{

int found = 0;

char bookName[MAX\_BOOK\_NAME] = {0};

s\_BooksInfo addBookInfoInDataBase = {0};

FILE \*fp = NULL;

int status = 0;

fp = fopen(FILE\_NAME,"rb");

if(fp == NULL)

{

printf("\n\t\t\tFile is not opened\n");

exit(1);

}

headMessage("SEARCH BOOKS");

//put the control on books detail

if (fseek(fp,FILE\_HEADER\_SIZE,SEEK\_SET) != 0)

{

fclose(fp);

printf("\n\t\t\tFacing issue while reading file\n");

exit(1);

}

printf("\n\n\t\t\tEnter Book Name to search:");

fflush(stdin);

fgets(bookName,MAX\_BOOK\_NAME,stdin);

while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))

{

if(!strcmp(addBookInfoInDataBase.bookName, bookName))

{

found = 1;

break;

}

}

if(found)

{

printf("\n\t\t\tBook id = %u\n",addBookInfoInDataBase.books\_id);

printf("\t\t\tBook name = %s",addBookInfoInDataBase.bookName);

printf("\t\t\tBook authorName = %s",addBookInfoInDataBase.authorName);

printf("\t\t\tBook issue date(day/month/year) = (%d/%d/%d)",addBookInfoInDataBase.bookIssueDate.dd,

addBookInfoInDataBase.bookIssueDate.mm, addBookInfoInDataBase.bookIssueDate.yyyy);

}

else

{

printf("\n\t\t\tNo Record");

}

fclose(fp);

printf("\n\n\n\t\t\tPress any key to go to main menu.....");

getchar();

}

// v books function

void viewBooks()

{

int found = 0;

char bookName[MAX\_BOOK\_NAME] = {0};

s\_BooksInfo addBookInfoInDataBase = {0};

FILE \*fp = NULL;

int status = 0;

unsigned int countBook = 1;

headMessage("VIEW BOOKS DETAILS");

fp = fopen(FILE\_NAME,"rb");

if(fp == NULL)

{

printf("File is not opened\n");

exit(1);

}

if (fseek(fp,FILE\_HEADER\_SIZE,SEEK\_SET) != 0)

{

fclose(fp);

printf("Facing issue while reading file\n");

exit(1);

}

while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))

{

printf("\n\t\t\tBook Count = %d\n\n",countBook);

printf("\t\t\tBook id = %u",addBookInfoInDataBase.books\_id);

printf("\n\t\t\tBook name = %s",addBookInfoInDataBase.bookName);

printf("\t\t\tBook authorName = %s",addBookInfoInDataBase.authorName);

printf("\t\t\tBook issue date(day/month/year) = (%d/%d/%d)\n\n",addBookInfoInDataBase.bookIssueDate.dd,

addBookInfoInDataBase.bookIssueDate.mm, addBookInfoInDataBase.bookIssueDate.yyyy);

found = 1;

++countBook;

}

fclose(fp);

if(!found)

{

printf("\n\t\t\tNo Record");

}

printf("\n\n\t\t\tPress any key to go to main menu.....");

fflush(stdin);

getchar();

}

// delete function

void deleteBooks()

{

int found = 0;

int bookDelete = 0;

sFileHeader fileHeaderInfo = {0};

char bookName[MAX\_BOOK\_NAME] = {0};

s\_BooksInfo addBookInfoInDataBase = {0};

FILE \*fp = NULL;

FILE \*tmpFp = NULL;

int status = 0;

headMessage("Delete Books Details");

fp = fopen(FILE\_NAME,"rb");

if(fp == NULL)

{

printf("File is not opened\n");

exit(1);

}

tmpFp = fopen("tmp.bin","wb");

if(tmpFp == NULL)

{

fclose(fp);

printf("File is not opened\n");

exit(1);

}

fread (&fileHeaderInfo,FILE\_HEADER\_SIZE, 1, fp);

fwrite(&fileHeaderInfo,FILE\_HEADER\_SIZE, 1, tmpFp);

printf("\n\t\t\tEnter Book ID NO. for delete:");

scanf("%d",&bookDelete);

while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))

{

if(addBookInfoInDataBase.books\_id != bookDelete)

{

fwrite(&addBookInfoInDataBase,sizeof(addBookInfoInDataBase), 1, tmpFp);

}

else

{

found = 1;

}

}

(found)? printf("\n\t\t\tRecord deleted successfully....."):printf("\n\t\t\tRecord not found");

fclose(fp);

fclose(tmpFp);

remove(FILE\_NAME);

rename("tmp.bin",FILE\_NAME);

}

void updateCredential(void)

{

sFileHeader fileHeaderInfo = {0};

FILE \*fp = NULL;

unsigned char userName[MAX\_SIZE\_USER\_NAME] = {0};

unsigned char password[MAX\_SIZE\_PASSWORD] = {0};

headMessage("Update Credential");

fp = fopen(FILE\_NAME,"rb+");

if(fp == NULL)

{

printf("File is not opened\n");

exit(1);

}

fread (&fileHeaderInfo,FILE\_HEADER\_SIZE, 1, fp);

if (fseek(fp,0,SEEK\_SET) != 0)

{

fclose(fp);

printf("\n\t\t\tFacing issue while updating password\n");

exit(1);

}

printf("\n\n\t\t\tNew Username:");

fflush(stdin);

fgets(userName,MAX\_SIZE\_USER\_NAME,stdin);

printf("\n\n\t\t\tNew Password:");

fflush(stdin);

fgets(password,MAX\_SIZE\_PASSWORD,stdin);

strncpy(fileHeaderInfo.username,userName,sizeof(userName));

strncpy(fileHeaderInfo.password,password,sizeof(password));

fwrite(&fileHeaderInfo,FILE\_HEADER\_SIZE, 1, fp);

fclose(fp);

printf("\n\t\t\tYour Password has been changed successfully");

printf("\n\t\t\ttLogin Again:");

fflush(stdin);

getchar();

exit(1);

}

void menu()

{

int choice = 0;

do

{

headMessage("MAIN MENU");

printf("\n\n\n\t\t\t1.Add Books");

printf("\n\t\t\t2.Search Books");

printf("\n\t\t\t3.View Books");

printf("\n\t\t\t4.Delete Book");

printf("\n\t\t\t5.Update Password");

printf("\n\t\t\t0.Exit");

printf("\n\n\n\t\t\tEnter choice => ");

scanf("%d",&choice);

switch(choice)

{

case 1:

addBookInDataBase();

break;

case 2:

searchBooks();

break;

case 3:

viewBooks();

break;

case 4:

deleteBooks();

break;

case 5:

updateCredential();

break;

case 0:

printf("\n\n\n\t\t\t\tTHANK YOU!!!\n\n\n\n\n");

exit(1);

break;

default:

printf("\n\n\n\t\t\tINVALID INPUT!!! Try again...");

} //Switch Ended

}

while(choice!=0); //Loop Ended

}

//login password

void login()

{

unsigned char userName[MAX\_SIZE\_USER\_NAME] = {0};

unsigned char password[MAX\_SIZE\_PASSWORD] = {0};

int L=0;

sFileHeader fileHeaderInfo = {0};

FILE \*fp = NULL;

headMessage("Login");

fp = fopen(FILE\_NAME,"rb");

if(fp == NULL)

{

printf("File is not opened\n");

exit(1);

}

fread (&fileHeaderInfo,FILE\_HEADER\_SIZE, 1, fp);

fclose(fp);

do

{

printf("\n\n\n\t\t\t\tUsername:");

fgets(userName,MAX\_SIZE\_USER\_NAME,stdin);

printf("\n\t\t\t\tPassword:");

fgets(password,MAX\_SIZE\_PASSWORD,stdin);

if((!strcmp(userName,fileHeaderInfo.username)) && (!strcmp(password,fileHeaderInfo.password)))

{

menu();

}

else

{

printf("\t\t\t\tLogin Failed Enter Again Username & Password\n\n");

L++;

}

}

while(L<=3);

if(L>3)

{

headMessage("Login Failed");

printf("\t\t\t\tSorry,Unknown User.");

getch();

system("cls");

}

}

int isFileExists(const char \*path)

{

// Try to open file

FILE \*fp = fopen(path, "rb");

int status = 0;

// If file does not exists

if (fp != NULL)

{

status = 1;

// File exists hence close file

fclose(fp);

}

return status;

}

void init()

{

FILE \*fp = NULL;

int status = 0;

const char defaultUsername[] ="KCELIBMANAGER\n";

const char defaultPassword[] ="KCEIAN\n";

sFileHeader fileHeaderInfo = {0};

status = isFileExists(FILE\_NAME);

if(!status)

{

//create the binary file

fp = fopen(FILE\_NAME,"wb");

if(fp != NULL)

{

//Copy default password

strncpy(fileHeaderInfo.password,defaultPassword,sizeof(defaultPassword));

strncpy(fileHeaderInfo.username,defaultUsername,sizeof(defaultUsername));

fwrite(&fileHeaderInfo,FILE\_HEADER\_SIZE, 1, fp);

fclose(fp);

}

}

}

int main()

{

init();

welcomeMessage();

login();

return 0;

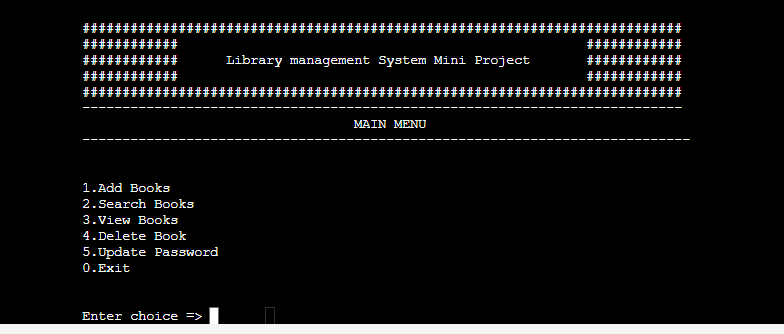
}

# RESULT AND DISCUSSION

# 5.1 HEAD MESSAGE AND LOGIN



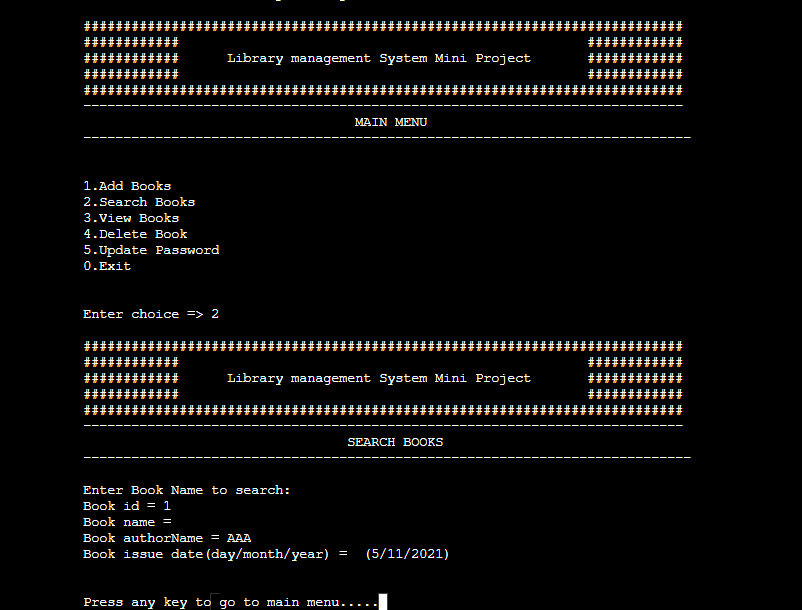
# 5.2 MAIN MENU



# 5.3 ADD NEW BOOK



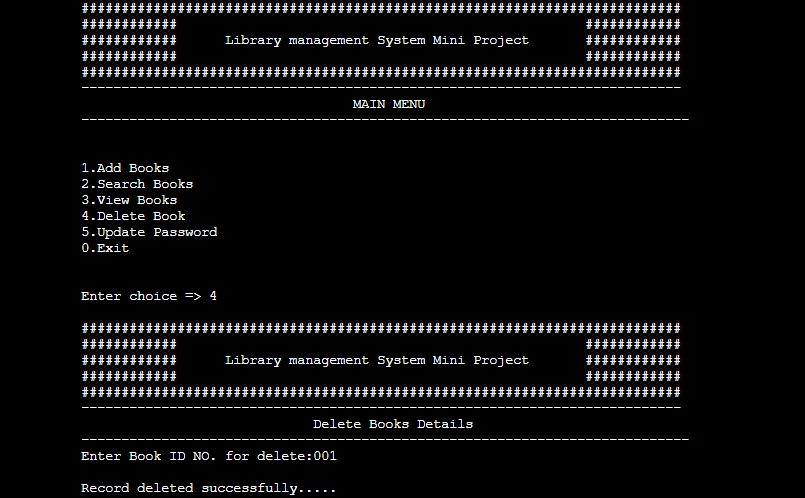
# 5.4 SEARCH BOOK



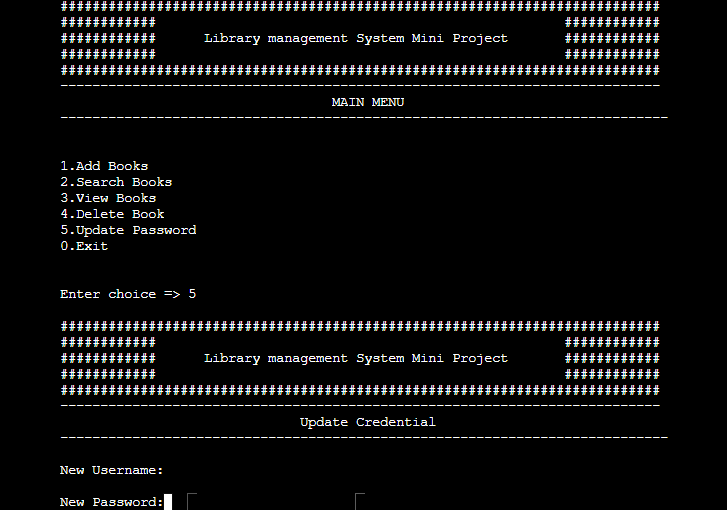
# 5.5 VIEW BOOK DETAILS



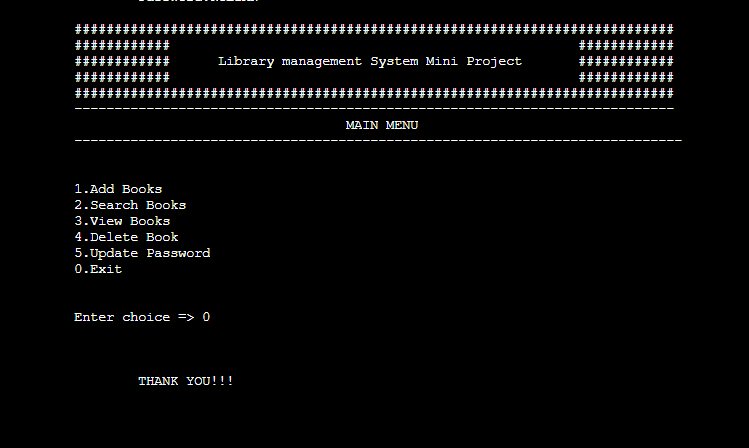
# 5.6 DELETE BOOK DETAILS



# 5.7 UPDATE CREDENTIALS



# 5.8 EXIT



**Description:**

|  |  |
| --- | --- |
| addBookInDataBase()  searchBooks()  viewBooks()  deleteBooks()  updateCredential() | To add books into the library  To search books available  To view books in the library  To delete books from the library  To update user credentials |

# CONCLUSION

Our project is a humble venture to satisfy the need of a librarian who maintains the library in a management. Several user friendly coding have also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the librarian as well as organization.

This system helps both students and library manger to keep constant track of all the books available in the library. It also allows us to search for the desired book in the library. This system reduce manual work to a great extent allows smooth flow of library activities by removing chances of error in details.

# BIBLIOGRAPHY

1. <https://www.tutorialspoint.com/cprogramming>
2. <https://www.programiz.com/c-programming>
3. <https://www.geeksforgeeks.org/library-management-system>