LIBRARY MANAGEMENT SYSTEM

Submitted by: Isha Singh Chauhan,1911596
Khushi Babbar,1911605
Kismat ,1911610
Lavanya Soni,1911625

For the award of degree of
Bachelor of Science
In
Mathematics
Under the supervision of Dr. Anoop K. Bhola
AIM and ACT
Department of Computer Science
Banasthali Vidyapith

Department of Mathematics and Computing Faculty of Mathematics and Computing

BANASTHALI VIDYAPEETH

BANASTHALI-304022

SESSION-2021-2022

CERTIFICATE

Certified that Isha Singh Chauhan, Khushi Babbar,Kismat and Lavanya soni has carried out the project work titled “Library Management System” from 03/01/2022 to 20/04/2022 for the award of the BSc (Mathematics) from Banasthali Vidyapith under my supervision. The thesis embodies result of original work and studies carried out by Students herself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else.

Name of Supervisor Dr. Anoop K. Bhola
Designation: AIM and ACT
Place: Banasthali Vidyapith
Date: 20-04-2022

ACKNOWLEDGEMENT

We take this opportunity to express our gratitude towards all those people who in various ways have helped in successful completion of our project.
We express gratitude to our project guide Dr. Anoop K. Bhola whose inspiration, suggestion and invaluable guidance enabled us to develop the present software. We are also thankful to our technical assistant of AIM &ACT Department for their cooperation throughout the project work. We hereby offer our sincere compliment to all our friends for their useful suggestion and cooperation. Last but not the least; we owe our debtness toward our revered parents for their moral support and constant encouragement that has made it possible for us to attain this usage this stage of academic achievement in our life.

Team Members: -

Isha Singh Chauhan, 1911596

Khushi Babbar, 1911605

Kismat , 1911610

Lavanya Soni, 1911625

Table of contents

2) System Design Specification

3) Coding
4) Screen images
5) Testing
6) References

SYSTEM REQUIRED SPECIFICATION

INTRODUCTION

1.1 Purpose

Library management system is a software application to upload the records related to Book purchase, Book search ,Catalog, Book issue, Book returns, Fine collection and all necessary requirements for the library to manage day to day operations.The purpose of this project is to provide a friendly environment to maintain the details of books which also help improving the security ,reliability, user service through greater access to accurate information also provide a faster and efficient method and an easier interface between user and software. This approach helps to maintain an easy circulation system using computers and to provide different reports.Due to computerized information,it reduces the risk of paperwork such as file loss ,file damage and time consuming.It can help users to manage the transaction or record more effectively and timesaving.

1.2 Scope

The software product “Library Management System” will be an application that will be used for maintaining the records in an organized manner and to replace old paper work systems. This project aimed at automating the books, newspaper, magazine etc. management for smooth working of the Library, by automating almost all the activities. Updating and modification will be easily achievable and all the calculations and accounting work would be accurate.

1.3 Definition ,Acronyms and Abbreviations

OS: Operating system

HDD: Hard Disk Drive

RAM: Random Access Memory

SQL: Structured Query Language

DFD: Data flow diagram

1.4 Overview

Library management is a project that manages and stores books information electronically according to students' needs. The system helps both administration and librarian to keep a constant track of all the books available in the library. It allows both the administration and the librarian to search for the desired book. It becomes necessary for colleges to keep a continuous check on the books issued and returned and even calculate fine. This task if carried out manually will be tedious and includes chances of mistakes. These errors are avoided by allowing the system to keep track of information such as issue date, last date to return the book and even fine information and thus there is no need to keep manual track of this information which thereby avoids chances of mistakes.

Thus, this system reduces manual work to a great extent allows smooth flow of library activities by removing chances of errors in detail.

THE OVERALL DESCRIPTION

2.1 Product Perspective

College library management system is a product which does not intent to have any relation with any other product. It is a complete system in itself. It is an exclusive product which is to be concerned with optimization.

The proposed library management system will take care of the current book detail at any point of time. The book issue, book return will update the current book details automatically so that the user will get the updated current book details.

2.1.1 Product Function

1. Login to account.
2. Provides the library general guidelines, search and contact facilities.
3. Provides the searching of general library information.
4. Allow login as admin, librarian.
5. Allow the admin to delete, update and add information about librarians and books.
6. Admin add and remove librarians account.
7. Librarians can create, update and delete information about books.
8. Increase efficiency of managing information.
9. Integration of all records of the library.

2.1.2 Hardware Interface

Processor: Pentium III and above

Operating System: Windows 2000 and above

RAM: Minimum RAM 2 GB

2.1.3 Software Interface

This project is an application that is developed using high level language.

The following technologies are used:

- Front End - c,c++
- Back End- Files
- Operating System: Windows 2000 and above

2.1.4 communication interface

2.2 User Characteristics:

Administrator: Administrator receives the request and permits the user to grant access. Administrator has all the rights to access all the information.

Librarian: Librarian contain all the information regarding books and students who rent books or who didn't. it can do modification,add or delete books records and also calculate fine for the book delay or which exceed their time limit with the help of student id and barcode method ,it also include data or information of returning books even without student's presence just by scanning the code of the book,also provide transaction report of book using searching of book name , publisher.

2.3 General Constraints:

Criticality of website: The server application will be available 24*7.

Safety and Security Considerations: The password and a valid username are the Security issue. External users and students will not be able to gain full Functionality of the website.

Any substantial enhancement in the website will require approval of the Administrator.

The information of all users, books and libraries must be stored in a database that is accessible by the website.

MS SQL server will be used as SQL engine and database.

Users may access LMS from any computer that has internet browsing capabilities and an internet connection.

Users must have their correct usernames and passwords to enter into their online accounts and do actions.

2.4 Technologies Used:

Front End: C, C++

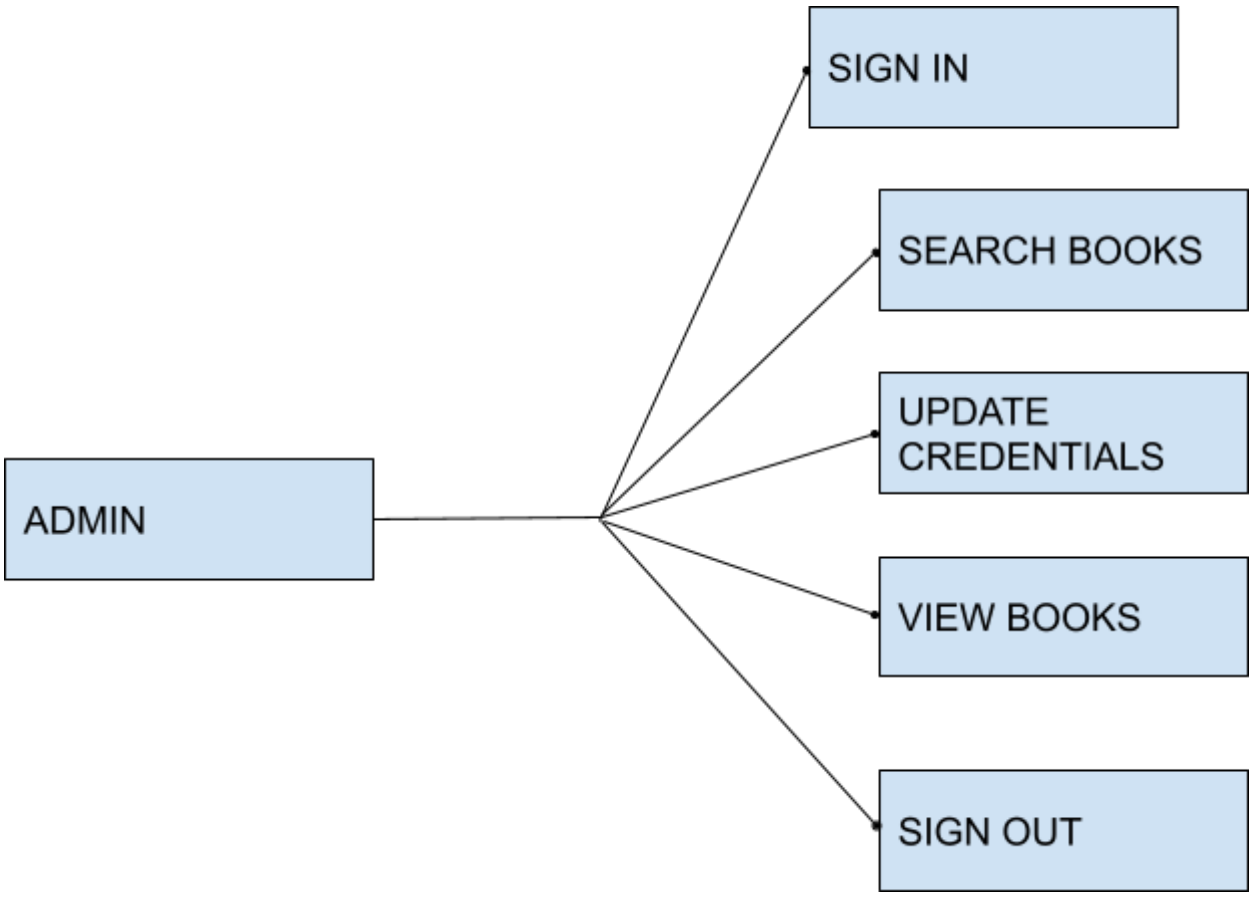
Design Tool: Windows 2000

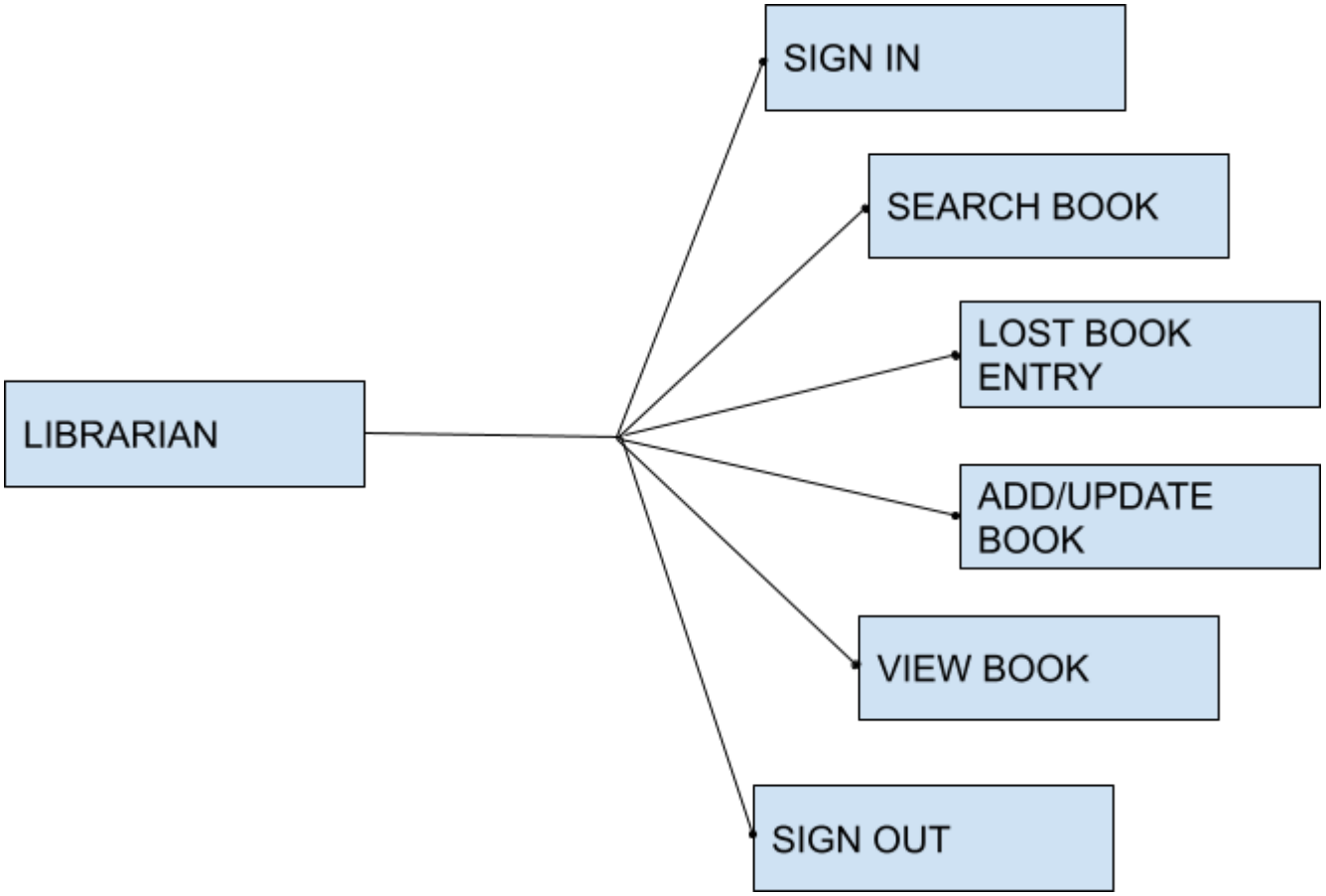
Back End: Files

SPECIFIC REQUIREMENTS

3.1 Functional Requirements:

1. View Home screen.
2. Login to system through the home page
   (if the account already exists otherwise send the request to admin to permit permission of new account).
3. View homepage of respective user.
4. Perform functionality as per user given below.
5. Logout the account/page





3.2 Non functional requirements:

Availability requirements:

The system is available 100% for the user and is used 24 hours a day and 365 days a year.The system shall be operational 24 hours a day and 7 days a week.

Efficiency requirements:

With the library management system,librarian should be able to process faster when they process book transaction.

Reliability requirements:

The system must perform accurately towards member request. A backup file is maintained so that in case of system clash, the data will not be affected.

Usability requirements:

This system must be designed user friendly and easy to use by staff so that user can perform their job nicely.

Security requirements:

This system must be highly secure in the login part. It is because the report can only view by admin level. Staff can perform most of the process except viewing reports and log files.

Maintainability Portability requirements:

Changes (new patrons addition,password changes,database changes)must be verified once per day.This system should provide automatic notifications to patrons by email about books overdue,reservation results,availability of reserved books.

SYSTEM DESIGN SPECIFICATION

OVERVIEW

Library management is a project that manages and stores books information electronically according to students' needs. The system helps both administration and librarian to keep a constant track of all the books available in the library. It allows both the administration and the librarian to search for the desired book. It becomes necessary for colleges to keep a continuous check on the books issued and returned and even calculate fine. This task if carried out manually will be tedious and includes chances of mistakes. These errors are avoided by allowing the system to keep track of information such as issue date, last date to return the book and even fine information and thus there is no need to keep manual track of this information which thereby avoids chances of mistakes.

Thus, this system reduces manual work to a great extent allows smooth flow of library activities by removing chances of errors in detail.

Definition ,Acronyms and Abbreviations

OS: Operating system

HDD: Hard Disk Drive

RAM: Random Access Memory

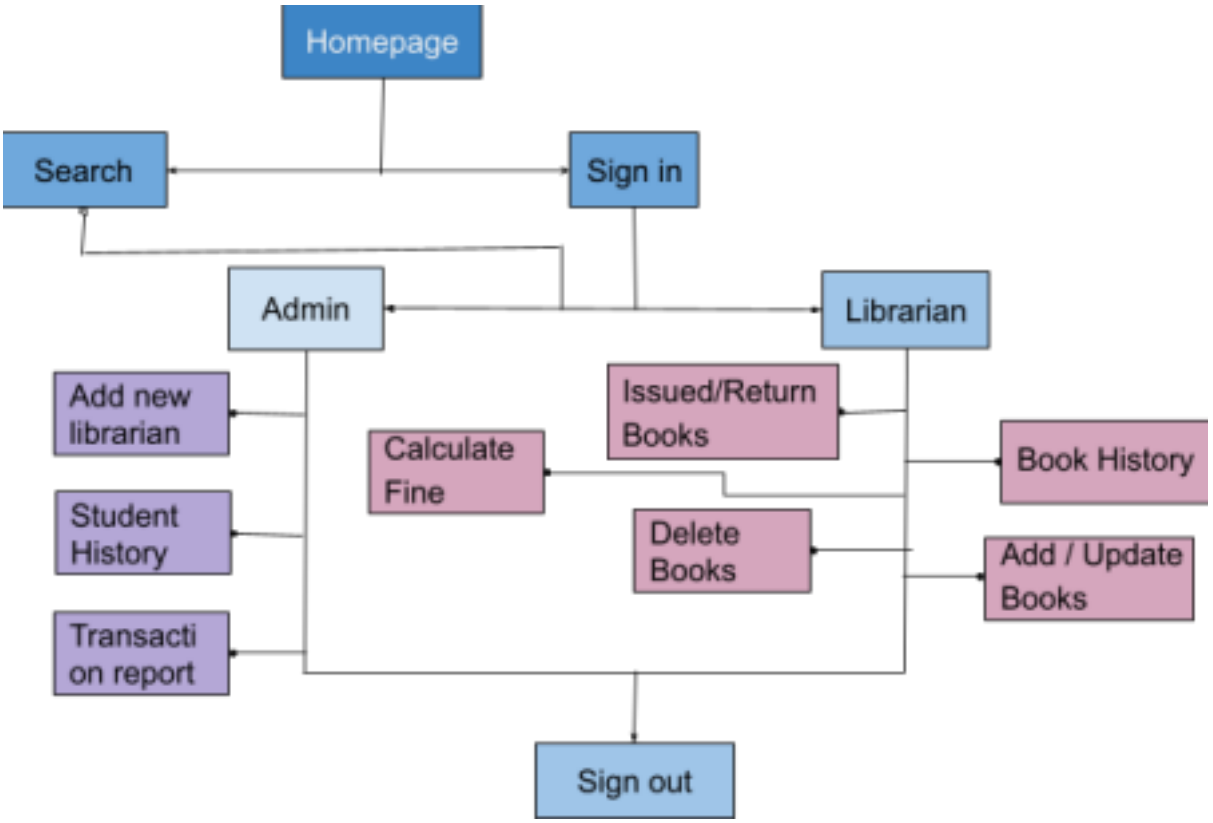
SQL: Structured Query Language

DFD: Data flow diagram

SYSTEM ARCHITECTURAL DESIGN

2.1 Detailed Description of Components

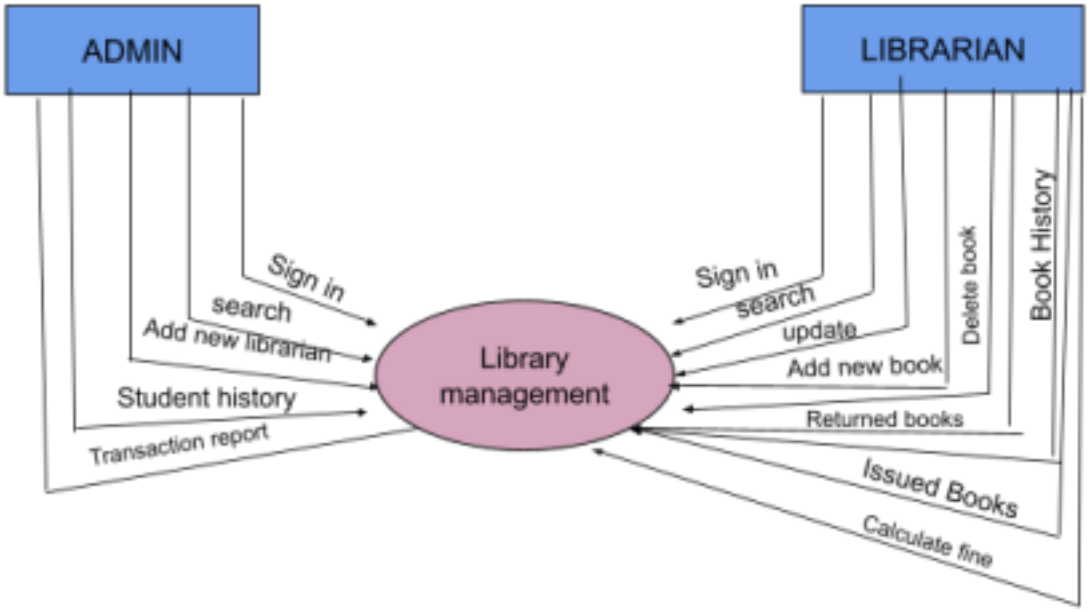
2.1.1 Structure chart



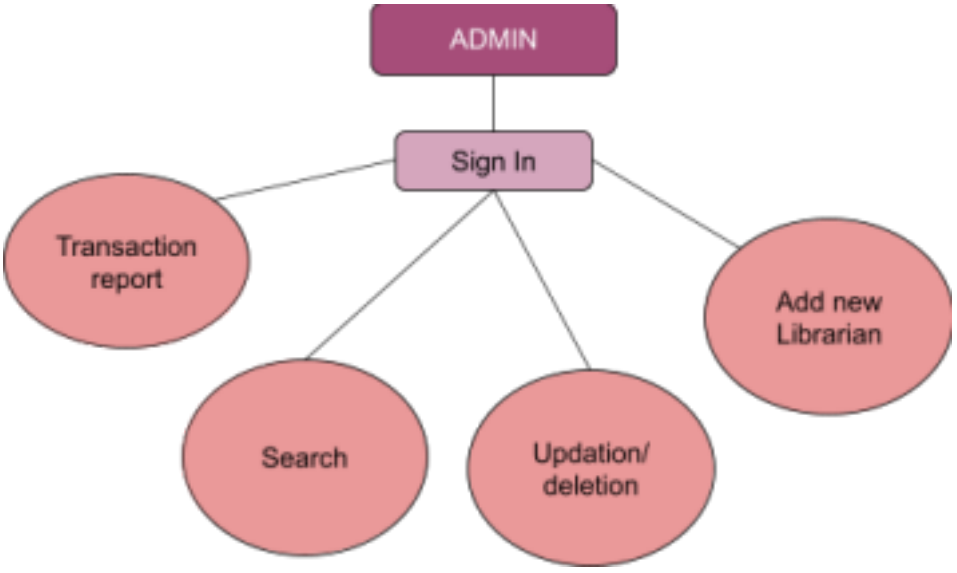
2.2.2 Data flow diagram

- DFD is a graphical representation of a system that shows data flow to,from and within the system.
- These are used to depict specific data flows from both physical and logical. ● the DFD's are divided into different level starting from 0- level until we get the final description of system.

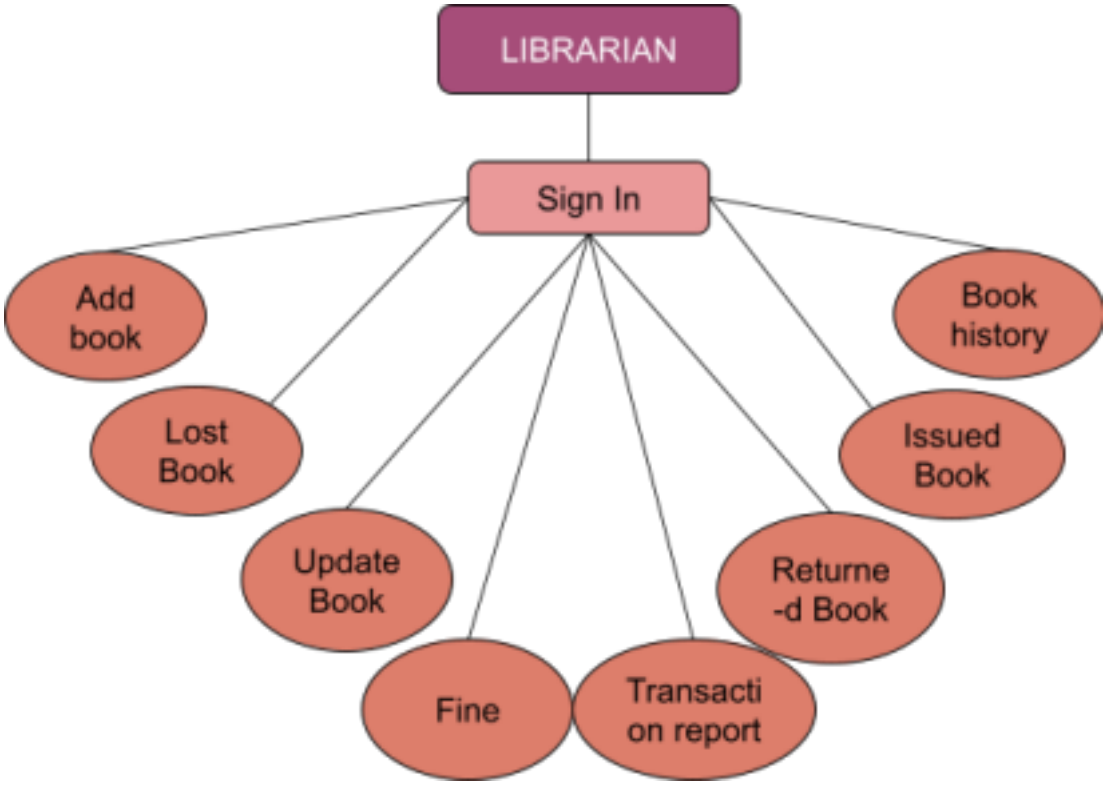
0-LEVEL DIAGRAM



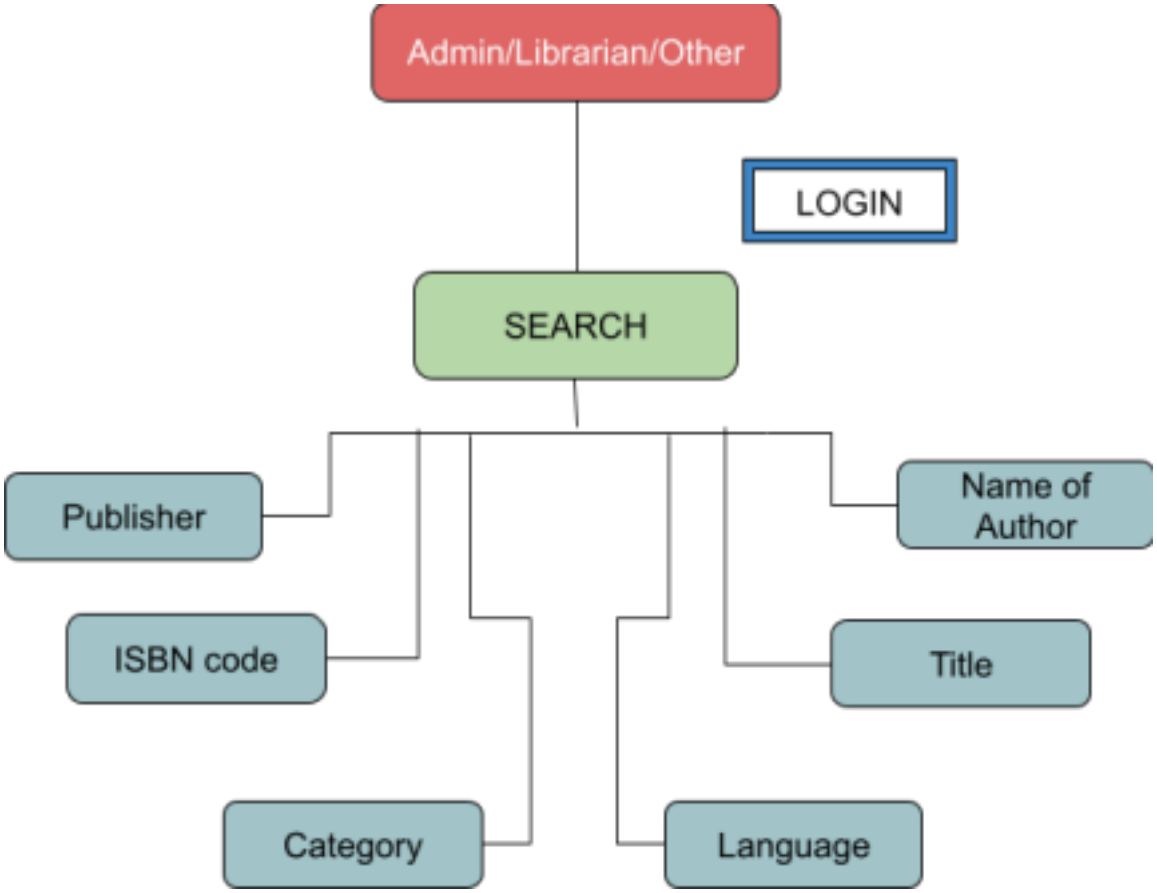
1-LEVEL DFD FOR ADMIN



1-LEVEL DFD FOR LIBRARIAN



2-LEVEL DFD FOR SEARCH MODULE



2.2 External Interface Requirements

User Interfaces

The Library Management System is used by librarians and admin, hence it is simply designed to sit users of all ages. We have aimed to make the interface user friendly and easy to use. This will be consistently tested when the software is being developed. The layout is made keeping in mind to position the buttons in a manner that its function is comprehensible and graspable when they look at the button. The button's are made plain with easy to read text for convenience. For example when they want to delete a record,we have used 'Delete' as the button's text rather than 'Destroy' or 'Expunge'.

Hardware Interfaces

For Library Management System, an additional device needed is a barcode scanner. Barcode scanner is needed to scan the book barcode, which would work as a unique id for each book and save time for the librarian.

Software Interfaces

The Library Management System requires users to install … .The OS required by the Library System is ….

Communication Interfaces

Although the Library System is not a web application, some functions require internet connection. Thus, HyperText Transfer Protocol (HTTP) shall be used to provide internet connection to the system. HTTP is a standard method of transferring data between web server and web client. Our library system will serve as a web client and send the request to the targeted web server. On the other hand, a USB port is also required to plug in the bar code reader.

DATA DESIGN

3.1 Database description

- In this we include ,maintain and format databases and its tables.
- the tables corresponding to each of the entities, holding the information.

The tables have the fields , their description, and their data type as well as integrity constraints.

TABLE 1: LIBRARIAN TABLE

Field	Type	Constraints	Description
user_id	varchar	primary key	smart id of the librarian
password	varchar	unique	password of librarian

TABLE 2: ADMIN TABLE

Field	Type	Constraints	Description
user_id	varchar	primary key	smart id of the admin
password	varchar	unique	password of admin

TABLE 3: BOOK TABLE

Field	Type	Constraints	Description
ISBN	varchar	primary key	Unique id of the book
Title	varchar	not null	Name of the book
Author	varchar	not null	Name of author(s) of the book
Language	varchar	not null	Language of the book
Description	varchar	not null	Description of the book
Category	varchar	not null	Category of book
Pubisher ID	varchar	Not null	Unique ID assigned to publisher

TABLE 4: STUDENT INFO TABLE

Field	Type	Constraints	Description
name	varchar	not null	name of the student
student_id	varchar	primary key	smart id of the student
ISBN	varchar	not null	unique id of the book
book issued	varchar	not null	ISBN code of books issued
date_of_issue	datetime	not null	date on which book was issued by student
date_of_return	datetime	not null	date on which book was returned by student

TABLE 5: BOOK HISTORY TABLE

Field	Type	Constraints	Description
ISBN	varchar	primary key	unique id of the book
BarcodeID	varchar	not null	barcode of the book
Title	varchar	not null	name of the book
student_id	varchar	foreign key	smart id of the student
date_of_issue	datetime	not null	date on which book was issued by student
date_of_return	datetime	not null	date on which book was returned by student

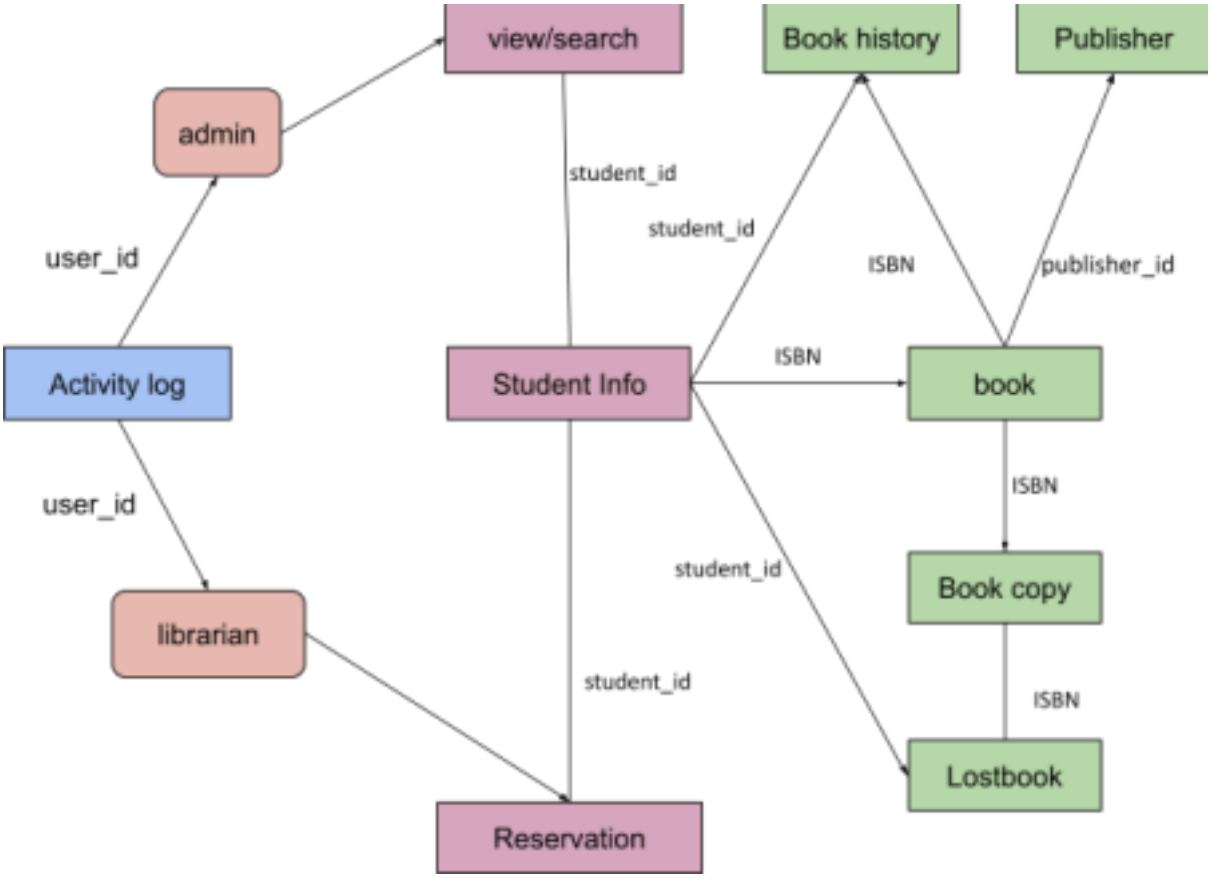
TABLE 6: PUBLISHER TABLE

FIeld	Type	Constraints	Description
publishername	varchar	not null	name of the publisher
publisher_id	varchar	primary key	assign id of publisher

TABLE 7: LOSTBOOK TABLE

Field	Type	Constraints	Description
ISBN	varchar	primary key	unique id of the book

ER DIAGRAM



USER INTERFACE DESIGN

4.1 DESCRIPTION OF THE USER INTERFACE

4.1.1 Login Module

Description of component

In order to make the system safe from unauthorized access, it is protected with user name and password. so that only valid and legitimate users can access the system.

Interface description

inputs: user type ,user ID and password.

outputs: The homepage will open up.

Processing details

- display the page with the user type ,username and password field.
- mark all the fields as compulsory.
- check whether the user name is valid or not ,if yes check whether the password matches or not.
- open the main page after clicking the login button after validation checking.

4.1.2 Search Module

Description of component

This module provides admin, librarian and non-users to search book ISBN code, Title, Author Publisher, Category or Language.

Interface description

inputs: book ISBN code, Title, Author Publisher, Category or Language.

outputs: searched details.

Processing details

- First user is to fill in the details, after choosing an option, if verified, then results are displayed.
- checks whether the search is an authorized search for the user or not.

4.1.3 Admin Update Credentials Module

Description of component

This module is used to update the login information which the admin wants.

Interface description

inputs:user can change login information like password and username.

outputs: updated information.

Processing details

- Admin enters the required new information.
- Verification is done on input data.

- If it is invalid , an error message will pop up.
- otherwise information is updated.

4.1.4 ADD New Book

Description of component

This module is used to add a new book; the librarian will have to add various book information to add the book to system data.

Interface description

inputs: book details.

outputs: new book added.

Processing details

- Only the librarian can add new books in the existing library.

4.1.5 View Book

Description of component

This module is used to view all the book's in the system.

Interface description

inputs: choice.

outputs: all the books displayed.

Processing details

- This option can be used by all the users.
- The details of all the books along with a counter variable is displayed.

4.1.6 Lost Book Entry

Description of component

This module is used to add books in the Lost Book Record ; the librarian either adds the book or checks for its existence using this module via its ISBN code.

Interface description

inputs: ISBN code.

output: new book added to the Lost Book Entry.

Processing details

- The librarian deletes the book from the records.

```
#include <iostream>
#include <conio.h>
#include <stdio.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 MAX_Language 15
#define MAX_Category_Type  20
#define MAX_ISBNcode 15
#define MAX_TITLE 100
#define MAX_PUBLISHER_NAME 50
#define FILE_NAME  "AticleworldLibBookS.bin"
#define MAX_BOOK_NAME   50
#define MAX_AUTHOR_NAME 50
#define MAX_LIBRARIAN_NAME 10
#define MAX_ADMIN_NAME 10
#define FILE_HEADER_SIZE  sizeof(sFileHeader)
//structure to store date
//defined various

char description[2048]; //description and publisher ID global variables
described
char publisherID[30];
int found;
int option;
int choice;

//structure datatype to store date about :
typedef struct
    {
        int yyyy;
        int mm;
        int dd;
    } Date;


typedef struct //user and password
    {
        char libuser[MAX_SIZE_USER_NAME];
        char libpass[MAX_SIZE_PASSWORD];
        char adminuser[MAX_SIZE_USER_NAME];
        char adminpass[MAX_SIZE_PASSWORD];
    } sFileHeader;
```

```
typedef struct // to store data about books
    {
        char language[MAX_Language];
        char category[MAX_Category_Type];
        char ISBNCode[MAX_ISBNcode];
        char publisherName[MAX_PUBLISHER_NAME];
        char bookName[MAX_BOOK_NAME];
        char authorName[MAX_AUTHOR_NAME];
        Date bookIssueDate;
    } s_BooksInfo;


//main program functions


void centermsg(const char* message)
{
        int l =0;  // length
        int p = 0; // position
        //calculate how many space need to print
        l = (78 - strlen(message))/2;
        printf("\t\t\t");
        for(p = 0; p < l ; p++)
          printf(" ");
        printf("%s",message);
}



void mainmsg(const char *message)
{

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

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



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++)
          printf(" ");
```

```
        printf("%s",message);
}



void headMessage(const char *message)
{
    //system("cls");

printf("\t\t\t#########################################################################
#########");
    printf("\n\t\t\t############
############");
    printf("\n\t\t\t############      Library management System Project in C
############");
    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()
{
    char message[100]="Welcome Screen";
    headMessage(message);
    printf("\n\n");
    printf("\n\t\t\t
*-------------------------------------------------------------------------*\n");
    printf("\n\t\t\t
*-------------------------------------------------------------------------*\n");
    printf("\n\t\t\t\t\t\t        =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=");
    printf("\n\t\t\t\t\t\t        =                  WELCOME                    =");
    printf("\n\t\t\t\t\t\t        =                    TO                       =");
    printf("\n\t\t\t\t\t\t        =                  LIBRARY                    =");
    printf("\n\t\t\t\t\t\t        =                MANAGEMENT                   =");
    printf("\n\t\t\t\t\t\t        =                  SYSTEM                     =");
    printf("\n\t\t\t\t\t\t        =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=");
    printf("\n\t\t\t\t
*-------------------------------------------------------------------------*\n");
    printf("\n\t\t\t\t
*-------------------------------------------------------------------------*\n");
    printf("\n\t\t\t\t  Final Year Project : GROUP A5\n");
    printf("\n\t\t\t\t Isha Singh Chauhan, Khushi Babbar, Kismat, Lavanya Soni");
    printf("\n\n\n\t\t\t\t Enter any key to continue.....");
```

```
    getch();
}



int isNameValid(const char *name)
{
    int validName = 1;
    int l = 0;
    int index = 0;
    l = strlen(name);
    for(; index < l ; ++index)
    {          //changing the space caution
        if(!(isalnum(name[index])) && (name[index] != '\n') && (name[index] != '
') && (name[index] != '-'))
        {
            validName = 0;
            break;
        }
    }
    return validName;
}



int  IsLeapYear(int year)
{
    return (((year % 4 == 0) &&
             (year % 100 != 0)) ||
            (year % 400 == 0));
}



int isValidDate(Date *validDate) // returns 1 if given date is valid.
{
    //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;
}



void searchTitle()
{
    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("SEARCHING BOOK BY TITLE");
    //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 :\t");
    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)// IF VALUE of found is 1, it is considered true value , hence the
if statement would work
    {
        printf("\t\t\tBook Name =\t %s\n",addBookInfoInDataBase.bookName);
        printf("\t\t\tAuthor Name =\t %s\n",addBookInfoInDataBase.authorName);
        printf("\t\t\tLanguage =\t %s\n",addBookInfoInDataBase.language);
         printf("\t\t\tISBN Code =\t %s\n",addBookInfoInDataBase.ISBNCode);
   }
    else
    {
        printf("\n\t\t\tNo Record Found");
    }
```

```
    fclose(fp);
    printf("\n\n\t\t\tPress any key to go to main menu.....");
    fflush(stdin);
    getchar();
}



void searchISBN() // not sure of the retuen type
{
    char ISBNCode[MAX_ISBNcode] = {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("SEARCHING BOOK BY ISBN CODE");
    //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 The Code to search : \t");
    fflush(stdin);
    fgets(ISBNCode,MAX_ISBNcode,stdin);
    while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))
    {  // might have to make changes in this function :addBookInfoInDataBase: or
make anew one
        if(!strcmp(addBookInfoInDataBase.ISBNCode, ISBNCode))
        {
            found = 1;
            break;
        }
    }
    if(found)// IF VALUE of found is 1, it is considered true value , hence the
if statement would work
    {
        printf("\t\t\tBook Name =\t %s\n",addBookInfoInDataBase.bookName);
        printf("\t\t\tAuthor Name =\t %s\n",addBookInfoInDataBase.authorName);
        printf("\t\t\tLanguage =\t %s\n",addBookInfoInDataBase.language);
         printf("\t\t\tISBN Code =\t %s\n",addBookInfoInDataBase.ISBNCode);
    }
    else
    {
        printf("\n\t\t\tNo Such Record Found");
    }
```

```
    fclose(fp);
    printf("\n\n\t\t\tPress any key to go to main menu.....");
    fflush(stdin);
    getchar();
}



void searchAuthor()// not sure of the retuen type
{
    char authorName[MAX_AUTHOR_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("SEARCHING BOOK BY AUTHOR NAME");
    //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 the name of Author : \t");
    fflush(stdin);
    fgets(authorName,MAX_AUTHOR_NAME,stdin);
    while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))
    {  // might have to make changes in this function :addBookInfoInDataBase: or
make anew one
        if(!strcmp(addBookInfoInDataBase.authorName, authorName))
        {
            found = 1;
            break;
        }
    }
    if(found)// IF VALUE of found is 1, it is considered true value , hence the
if statement would work
    {
        printf("\t\t\tBook Name =\t %s\n",addBookInfoInDataBase.bookName);
        printf("\t\t\tAuthor Name =\t %s\n",addBookInfoInDataBase.authorName);
        printf("\t\t\tLanguage =\t %s\n",addBookInfoInDataBase.language);
         printf("\t\t\tISBN Code =\t %s\n",addBookInfoInDataBase.ISBNCode);
    }
    else
    {
        printf("\n\t\t\tNo Record Found");
    }
```

```
    fclose(fp);
    printf("\n\n\t\t\tPress any key to go to main menu.....");
    fflush(stdin);
    getchar();
}



void searchPublisher()// not sure of the retuen type
{
    char publisherName[MAX_PUBLISHER_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("SEARCHING BOOK BY PUBLISHER");
    //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 the nmae of Publisher : \t");
    fflush(stdin);
    fgets(publisherName,MAX_PUBLISHER_NAME,stdin);
    while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))
    {  // might have to make changes in this function :addBookInfoInDataBase: or
make anew one
        if(!strcmp(addBookInfoInDataBase.publisherName, publisherName))
        {
            found = 1;
            break;
        }
    }
    if(found)// IF VALUE of found is 1, it is considered true value , hence the
if statement would work
    {
        printf("\t\t\tBook Name =\t %s\n",addBookInfoInDataBase.bookName);
        printf("\t\t\tAuthor Name =\t %s\n",addBookInfoInDataBase.authorName);
        printf("\t\t\tLanguage =\t %s\n",addBookInfoInDataBase.language);
         printf("\t\t\tISBN Code =\t %s\n",addBookInfoInDataBase.ISBNCode);
    }
    else
    {
        printf("\n\t\t\tNo Record Found");
    }
```

```
    fclose(fp);
    printf("\n\n\t\t\tPress any key to go to main menu.....");
    fflush(stdin);
    getchar();
}



void searchCategory()// not sure of the retuen type
{
    char category[MAX_Category_Type] = {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("SEARCHING BOOK BY CATEGORY");
    //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 the category of book : \t");
    fflush(stdin);
    fgets(category,MAX_Category_Type,stdin);
    while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))
    {  // might have to make changes in this function :addBookInfoInDataBase: or
make anew one
        if(!strcmp(addBookInfoInDataBase.category, category))
        {
            found = 1;
            break;
        }
    }
    if(found)// IF VALUE of found is 1, it is considered true value , hence the
if statement would work
    {
        printf("\t\t\tBook Name =\t %s\n",addBookInfoInDataBase.bookName);
        printf("\t\t\tAuthor Name =\t %s\n",addBookInfoInDataBase.authorName);
        printf("\t\t\tLanguage =\t %s\n",addBookInfoInDataBase.language);
         printf("\t\t\tISBN Code =\t %s\n",addBookInfoInDataBase.ISBNCode);
    }
    else
    {
        printf("\n\t\t\tNo Record Found");
    }
```

```
    fclose(fp);
    printf("\n\n\t\t\tPress any key to go to main menu.....");
    fflush(stdin);
    getchar();
}



void searchLanguage()// not sure of the retuen type
{
    char language[MAX_Language] = {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("SEARCHING BOOK BY LANGUAGE");
    //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 The Language of the book : \t");
    fflush(stdin);
    fgets(language,MAX_Language,stdin);
    while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))
    {  // might have to make changes in this function :addBookInfoInDataBase: or
make anew one
        if(!strcmp(addBookInfoInDataBase.language, language))
        {
            found = 1;
            break;
        }
    }
    if(found)// IF VALUE of found is 1, it is considered true value , hence the
if statement would work
    {
        printf("\t\t\tBook Name =\t %s\n",addBookInfoInDataBase.bookName);
        printf("\t\t\tAuthor Name =\t %s\n",addBookInfoInDataBase.authorName);
        printf("\t\t\tLanguage =\t %s\n",addBookInfoInDataBase.language);
         printf("\t\t\tISBN Code =\t %s\n",addBookInfoInDataBase.ISBNCode);
    }
    else
    {
        printf("\n\t\t\tNo Record Found");
    }
```

```
    fclose(fp);
    printf("\n\n\t\t\tPress any key to go to main menu.....");
    fflush(stdin);
    getchar();
}



void addInDataBase() // Add books in list
{
    int days;  //days ko store karne ka struct
    int status = 0; // structure type ka variable toh esmein saare ceeze hongi
structure wali
    s_BooksInfo addBookInfoInDataBase = {0}; // variable i value is zero for
now, we are going to fill in sll the other declared variables in the struct
    FILE *fp = NULL; // fp is a pointer

    fp = fopen(FILE_NAME,"ab+");   // file_name is already defined
    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 ISBN Code = ");
    fflush(stdin);
    gets(addBookInfoInDataBase.ISBNCode);
    do // make this valid comment for all the things mentioned
    {
        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\tBook Name contains 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\tAuthor name contains invalid character. Please enter
```

```
again.");
        }
    }
    while(!status);
    printf("\n\t\t\tPublisher ID = ");
    fflush(stdin);
    gets(addBookInfoInDataBase.publisherName);
    do
    {
        printf("\n\t\t\tCategory of book  = ");
        fflush(stdin);
        fgets(addBookInfoInDataBase.category,MAX_Category_Type,stdin);
        status = isNameValid(addBookInfoInDataBase.category);
        if (!status)
        {
            printf("\n\t\t\tCategory contain invalid character. Please enter
again.");
        }
    }
    while(!status);
    do
    {
        printf("\n\t\t\tLanguage  = ");
        fflush(stdin);
        fgets(addBookInfoInDataBase.language,MAX_Language,stdin);
        status = isNameValid(addBookInfoInDataBase.language);
        if (!status)
        {
            printf("\n\t\t\tLanguage name 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);
    printf("\n\t\t\tDescription = ");
    fflush(stdin);
    scanf("%s",&description);
    //taking a global variable for desription, and calling it here without
structure
```

```
    fwrite(&addBookInfoInDataBase,sizeof(addBookInfoInDataBase), 1, fp);
    fclose(fp);
    printf("\n\n\t\t\tPress any key to go to main menu.....");
    fflush(stdin);
    getchar();
}



void searchBooks()
{
    found = 0;
    printf("\n\n\t\t\tSearch book by... choose one.\n\n\n");
    printf("\t\t\t1. ISBN code\n");
    printf("\t\t\t2. Title\n");
    printf("\t\t\t3. Name of Author\n");
    printf("\t\t\t4. Name of Publisher\n");
    printf("\t\t\t5. Category\n");
    printf("\t\t\t6. Language\n");
    printf("\t\t\t0. To exit\n");
    scanf("%d",&option);
    switch(option)
        {
        case 1:
            searchISBN();
            break;
        case 2:
            searchTitle();
            break;
        case 3:
            searchAuthor();
            break;
        case 4:
            searchPublisher();
            break;
        case 5:
            searchCategory();
            break;
         case 6:
            searchLanguage();
            break;
        case 0:
            printf("\n\n\n\t\t\t\tThank you !\n\n\n\n\n");
        default:
            printf("\n\n\n\t\t\tINVALID INPUT!!! Try again...");
        }
}



void viewBooks()
{
```

```
    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 Name = %s",addBookInfoInDataBase.bookName);
        printf("\n\t\t\tBook ISBN Code = %s",addBookInfoInDataBase.ISBNCode);
        printf("\n\t\t\tBook Author Name =
%s",addBookInfoInDataBase.authorName);
        printf("\n\t\t\tBook Publisher Name =
%s",addBookInfoInDataBase.publisherName);
        printf("\n\t\t\tBook Language = %s",addBookInfoInDataBase.language);
        printf("\n\t\t\tBook Category = %s",addBookInfoInDataBase.category);
        printf("\n\t\t\tBook Count = %d\n\n",countBook);
        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();
}



void updateCredential(void)
{
    sFileHeader fileHeaderInfo = {0};
    FILE *fp = NULL;
    char userName[MAX_SIZE_USER_NAME] = {0};
    char password[MAX_SIZE_PASSWORD] = {0};
    headMessage("Update Credentials");
```

```
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\t\t\tPress 1 to Update Credentials for Librarian\n");
printf("\n\t\t\tPress 2 to Update Credentials for Admin\n");
printf("\n\t\t\tPress 2 to Exit");
printf("\n\n\n\t\t\tEnter choice => \t");
scanf("%d",&choice);
    switch(choice)
    {
    case 1:
         printf("\n\n\t\t\tNew Username:\t");
               fflush(stdin);
               fgets(userName,MAX_SIZE_USER_NAME,stdin);
               printf("\n\n\t\t\tNew Password:\t");
               fflush(stdin);
               fgets(password,MAX_SIZE_PASSWORD,stdin);
               strncpy(fileHeaderInfo.libuser,userName,sizeof(userName));
               strncpy(fileHeaderInfo.libpass,password,sizeof(password));
        break;
    case 2:
         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.adminuser,userName,sizeof(userName));
               strncpy(fileHeaderInfo.adminpass,password,sizeof(password));
        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);
fwrite(&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
fclose(fp);
printf("\n\t\t\tPassword has been changed successfully");
printf("\n\t\t\ttLogin Again:");
```

```
    fflush(stdin);
    getchar();
    exit(1);
}



void LostBooks()  //lost books - to keep record of all the lost books
{
    found = 0;
    char booklost[15];
    sFileHeader fileHeaderInfo = {0};
    char bookName[MAX_BOOK_NAME] = {0};
    s_BooksInfo addBookInfoInDataBase = {0};
    FILE *fp = NULL;
    FILE *tmpFp = NULL;
    int status = 0;
   headMessage("Lost Book Details");
    fp = fopen(FILE_NAME,"rb");
    if(fp == NULL)
    {
        printf("File is not opened\n");
        exit(1);
    }
    tmpFp = fopen("tmp.bin","wb");  //check this tmpFp = fopen("tmp","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 ISBN Code ");
    scanf("%s",&booklost);
    while (fread (&addBookInfoInDataBase, sizeof(addBookInfoInDataBase), 1, fp))
    {
        if(addBookInfoInDataBase.ISBNCode != booklost)
        {
            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 menulibrarian()
{
//add more options depening upon "how data to add"
    choice = 0;
    do
    {
        printf("\n\n\t logged in as Libraian\n\n");
         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.Lost Book Entry");
        printf("\n\t\t\t0.Exit");
        printf("\n\n\n\t\t\tEnter choice => ");
        scanf("%d",&choice);
        switch(choice)
        {
        case 1:
               addInDataBase();
            break;
        case 2:
            searchBooks();
            break;
        case 3:
               viewBooks();
            break;
        case 4:
            LostBooks();
            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
}



void menuadmin()
{
    int choice = 0;
    do
```

```
    {
        printf("\n\n\t logged in as Admin\n\n");
         headMessage("MAIN MENU");
        printf("\n\n\n\t\t\t1.View Books");
        printf("\n\t\t\t2.Search Books");
        printf("\n\t\t\t3.Update Credential");
        printf("\n\t\t\t0.Exit");
        printf("\n\n\n\t\t\tEnter choice => ");
        scanf("%d",&choice);
        switch(choice)
        {
        case 1:
         viewBooks();
            break;
        case 2:
            searchBooks();
            break;
        case 3:
            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
}



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 login()//add exit n all statements
{
```

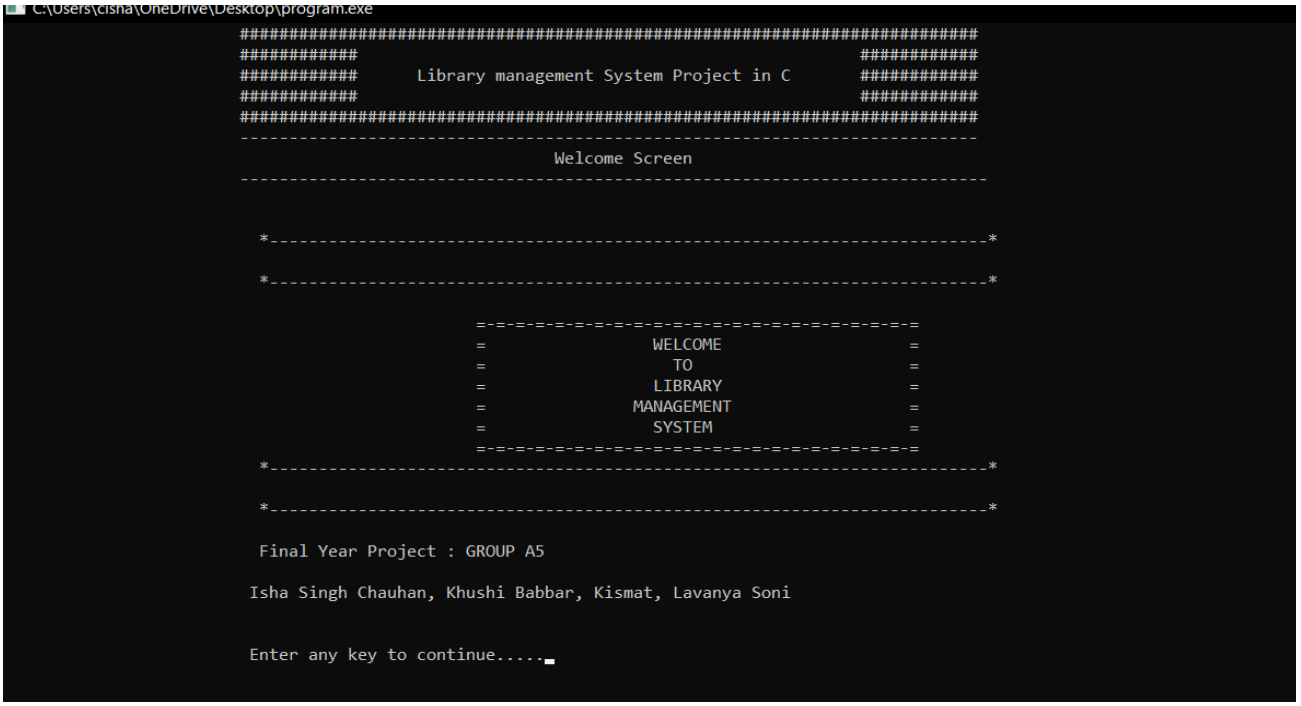
```
    char userName[MAX_SIZE_USER_NAME] = {0}; //821     48
C:\Users\cisha\OneDrive\Documents\full.cpp  [Error] invalid conversion from
'unsigned char*' to 'char*' [-fpermissive]
    char password[MAX_SIZE_PASSWORD] = {0};
    int L=0;
    int n;
    sFileHeader fileHeaderInfo = {0};
    FILE *fp = 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.adminuser)) ^
(!strcmp(userName,fileHeaderInfo.libuser)))
        {
            n=strcmp(userName,fileHeaderInfo.adminuser);
               if(n==0)
               {
                     n=strcmp(password,fileHeaderInfo.adminpass);// compare
password
                     if(n==0)
                           menuadmin();
                     else
                           printf("\n\t\t\t\tLogin Failed! Password
Incorrect\n\n");
                           //exit(0);
            }
            else
                     n = strcmp(password,fileHeaderInfo.libpass);// saved space
                     if(n==0)
                           menulibrarian();
                     else
                           printf("\n\t\t\t\tLogin Failed! Password
Incorrect\n\n");
            }
        else
        {
            printf("n\t\t\t\tLogin Failed! Invalid Username\n\n");
            L++;
        }
    }
    while(L<=3);
```

```
    if(L>3)
    {
        headMessage("Login Failed");
        printf("\t\t\t\tSorry,Unknown User.");
        getch();
        system("cls");
    }
}



void init()
{
    FILE *fp = NULL;
    int status = 0;
    const char defaultadminUser[] ="admin\n";
    const char defaultadminPass[] ="pass123\n";
    const char defaultlibUser[] ="library\n";
    const char defaultlibPass[] ="pass123\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.adminpass,defaultadminPass,sizeof(defaultadminPass));

strncpy(fileHeaderInfo.adminuser,defaultadminUser,sizeof(defaultadminUser));

strncpy(fileHeaderInfo.libpass,defaultlibPass,sizeof(defaultlibPass));

strncpy(fileHeaderInfo.libuser,defaultlibUser,sizeof(defaultlibUser));
            fwrite(&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
            fclose(fp);
        }
    }
}



int main()
{
    int ch;
    init();
    welcomeMessage();
    //we are in main menu now
    do
    {
```

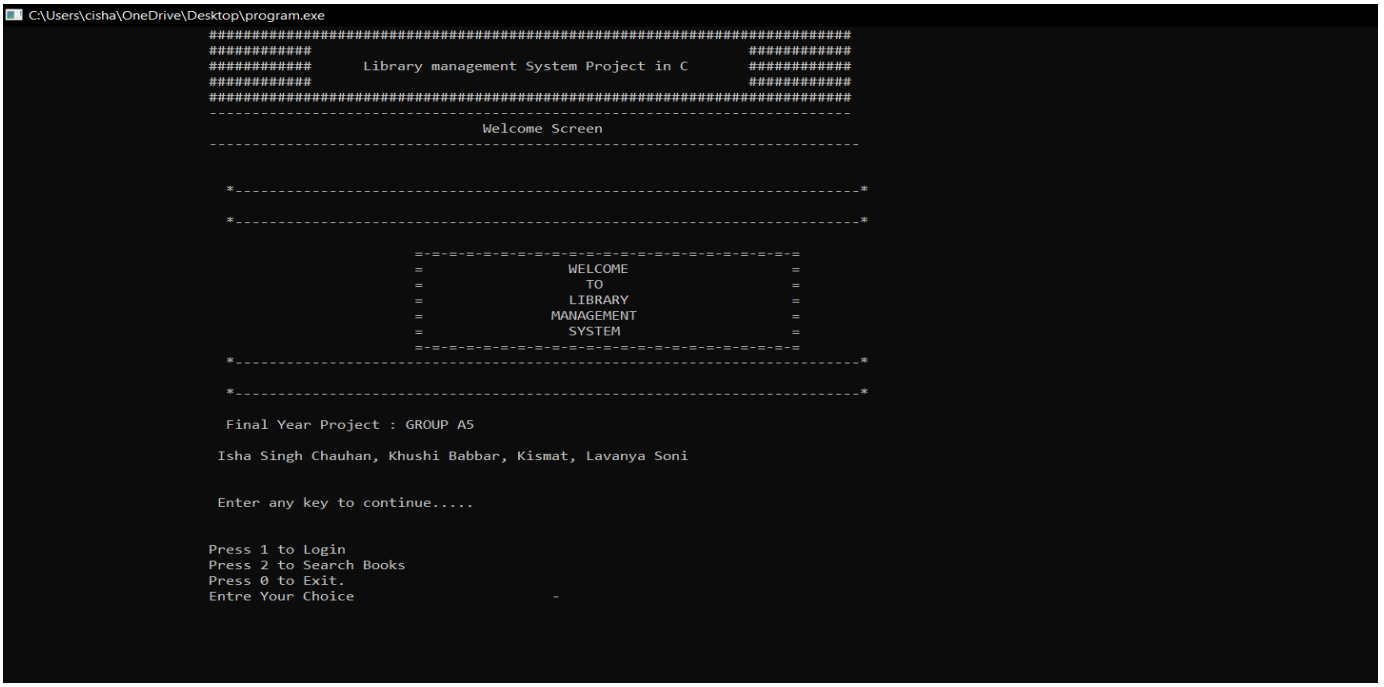
```
          printf("\n\n\n\t\t\tPress 1 to Login");
          printf("\n\t\t\tPress 2 to Search Books");
          printf("\n\t\t\tPress 0 to Exit.");
          printf("\n\t\t\tEntre Your Choice\t\t\t- ");
         scanf("%d",&ch);
        switch(ch)
         {
         case 1:
          login();
             break;
         case 2:
             searchBooks();
             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(ch!=0);
}



/*                              End of
                                Prog
```

SCREEN IMAGES

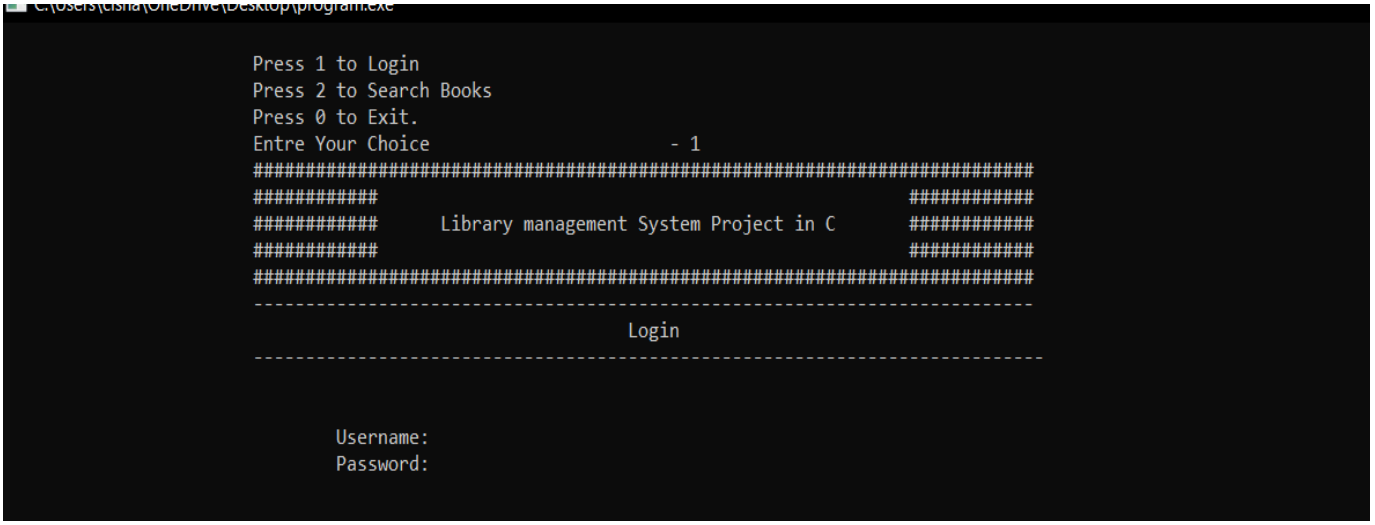
WELCOME SCREEN



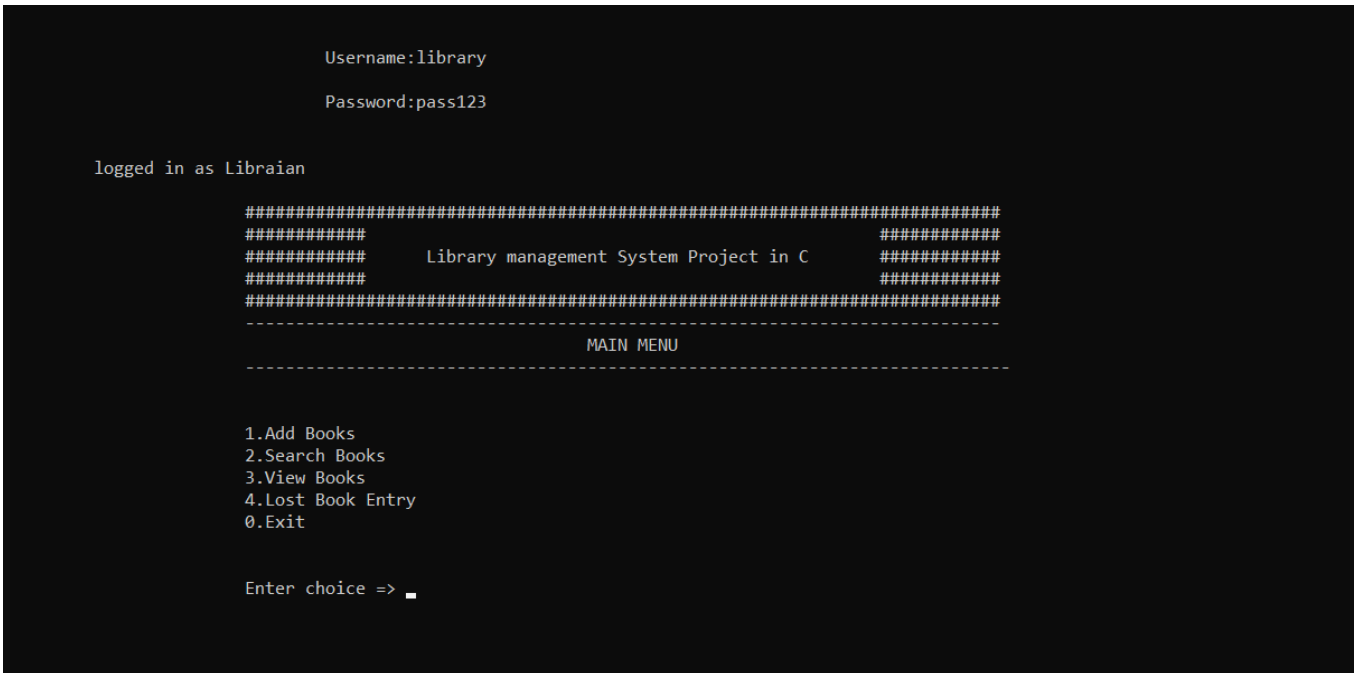
MAIN MENU



LOGIN SCREEN



LOGGED IN AS LIBRARIAN



LOGGED IN AS ADMIN

```
                        Username:admin

                        Password:pass123


logged in as Admin

                ################################################################################
                #############                                                    #############
                #############      Library management System Project in C       #############
                #############                                                    #############
                ################################################################################
                --------------------------------------------------------------------------------
                                              MAIN MENU
                --------------------------------------------------------------------------------


                1.View Books
                2.Search Books
                3.Update Credential
                0.Exit


                Enter choice => _
```

SEARCHING BOOKS WITHOUT LOGIN

```
C:\Users\cisha\OneDrive\Desktop\program.exe
                ########################################################################
                ############                                                ############
                ############      Library management System Project in C    ############
                ############                                                ############
                ########################################################################
                --------------------------------------------------------------------------
                                          Welcome Screen
                --------------------------------------------------------------------------


                  *----------------------------------------------------------------------*

                  *----------------------------------------------------------------------*

                                      =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
                                      =                  WELCOME                  =
                                      =                    TO                     =
                                      =                  LIBRARY                  =
                                      =                MANAGEMENT                 =
                                      =                  SYSTEM                   =
                                      =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
                  *----------------------------------------------------------------------*

                  *----------------------------------------------------------------------*

                   Final Year Project : GROUP A5

                  Isha Singh Chauhan, Khushi Babbar, Kismat, Lavanya Soni


                   Enter any key to continue.....


                Press 1 to Login
                Press 2 to Search Books
                Press 0 to Exit.
                Entre Your Choice                       - 2


                Search book by... choose one.


                1. ISBN code
                2. Title
                3. Name of Author
                4. Name of Publisher
                5. Category
                6. Language
                0. To exit
```

TESTING ISSUE

Software testing can be stated as the process of validating and verifying that a software program/application product:

- Meet the requirements that guided its design and development.
- works as expected; and
- can be implemented with the same characteristics.

Testing the process of making sure that the program performs the intended tasks.it include:

- unit testing: each component or part of the system is tested individually.

- module testing: a collection of dependent components such as an object class ,procedures and functions.
- integration testing: in this, many unit tested modules are combined into sub systems, which are then tested.
- system testing: entire system software is tested .it is a testing of a system against its initial objectives.

REFERENCES

1) www.google.com
2) A.K. Sharma, Data Structures Using C
3) Yashavant K. Let Us C
4) Sample studied- LMS6
https://drive.google.com/file/d/1Y5Fl_AMRyzgktNru9HD-52TVkFLoFici/view?usp=drivesdk.
5) articleworld.com