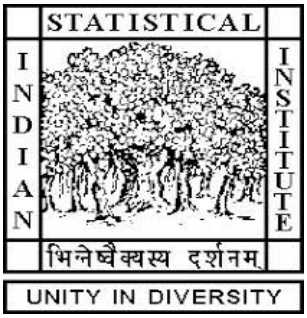
LIBRARY MANAGEMENT SYSTEM

Final Project Report of PGDCA 2nd Semester Examination, 2018

Project Guide: Dipankar Chatterjee



Submitted By:

Name: - Subhankar Karmakar

Roll: - DCA – 1702

Department of Computer Science

Indian Statistical Institute, Giridih

New Barganda, Rose Villa, Giridih-815301

INDIAN STATISTICAL INSTITUTE

CERTIFICATE

This is to certify that the final project report for ***"Library Management Software",*** submitted towards the partial fulfilment of 2nd semester of ***PGDCA (Post Graduate Diploma in Computer Application)*** at ***ISI (Indian Statistical Institute) Giridih,*** was carried out by *SUBHANKAR KARMAKAR, Roll No. DCA-1702,* under the supervision of *DIPANKAR CHATTERJEE, Teacher* ***PGDCA ISI, Giridih.***

Signature of Teacher/Professor

Dipankar Chatterjee

Teacher (PGDCA)

Indian Statistical Institute, Giridih

ACKNOWLEDGEMENT

Apart from the efforts of me, the success of completion of this project depends largely on the encouragement and guidance of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the completion of this project.

I would like to show my greatest appreciation to my project guide ***Dipankar Chatterjee, Teacher (PGDCA), ISI Giridih***. I feel encouraged and motivated I attend his meeting. Without his encouragement and guidance this would not have materialized.

I shall like to pay the deepest of my gratitude to the ***Dr. H.C.Behera*** in charge of ***ISI Giridih*** for providing me such a nice atmosphere and allow me to use all kind of resources I wanted.

The guidance and support received from all those who contributed and who are contributing to this project, was vital for the success of this project. I am grateful for their constant support and help of all the other members, and staff of ***ISI***

Signature of the Student

Subhankar Karmakar

ROLL No: - DCA – 1702

TABLE OF CONTENTS

Chapter 1

Chapter 2

Chapter 1

INTRODUCTION

This chapter gives an overview about the aim, objectives, background and operation environment of the system.

Topics Covered:

1.1 PROJECT AIMS AND OBJECTIVES

1.2 BACKGROUND OF PROJECTS

1.3 OPERATION ENVIRONMENT

1.1 PROJECT AIMS AND OBJECTIVES

The project aims and objectives that will be achieved after completion of this project are discussed in this subchapter. The aims and objectives are as follows:

- Admin Login Page where administrator or the Librarian can login with his user name password.
- A separate portion for the system settings where admin or the librarian will be able to change the system settings i.e. User name and Password for the admin,the maximum number of book can be lend to the Student or The Staff members separately, the maximum number of day that a member can keep books .
- Members are able to put request for the new books which are currently not in the library.
- In normal user section members are able to see books issued to him/her with fine details for each book and total fine for the member up to the date, books having fine by providing registered member id, book catalogue with the availability, members brief information Department wise.
- Various types of search options for both the members Admin or normal user.

1.2 BACKGROUND OF THE PROJECT

Library Management System is an application which refers to library systems which are generally small or medium in size. It is used by librarian to manage the library using a computerized system where he/she can record various transactions like issue of books, return of books, addition of new books, addition of new students etc.

Books and student maintenance modules are also included in this system which would keep track of the students using the library and also a detailed description about the books a library contains. With this computerized system there will be no loss of book record or member record which generally happens when a non-computerized system is used.

In addition, report module is also included in Library Management System. If user's position is admin, the user is able to generate different kinds of reports like lists of students registered, list of books, issue and return reports.

All these modules are able to help librarian to manage the library with more convenience and in a more efficient way as compared to library systems which are not computerized.

Chapter 2

SYSTEM ANALYSIS

In this chapter, we will discuss and analyse about the developing process of Library Management System including software requirement specification (SRS) and comparison between existing and proposed system.
The functional and non-functional requirements are included in SRS part to provide complete description and overview of system requirement before the developing process is carried out. Besides that, existing vs. proposed provides a view of how the proposed system will be more efficient than the existing one.

Topics Covered:

2.1 SOFTWARE REQUIREMENT AND SPECIFICATION
2.2 EXISTEING VS. PROPOSED SYSTEM
2.3 SOFTWARE TOOLS USED

2 . 1 SOFTWARE REQUIREMENT AND SPECIFICATION

2.1.1 GENERAL DESCRIPTION

2.1.1.1 PRODUCT DESCRIPTION:

Library Management System is a computerized system which helps user (librarian) to manage the library daily activity in electronic format. It reduces the risk of paper work such as file lost, file damaged and time consuming. It can help user to manage the transaction or record more effectively and time-saving.

2.1.1.2 PROBLEM STATEMENT:

The problem occurred before having computerized system includes:

- FILE LOST

When computerized system is not implemented file is always lost because of human environment. Sometimes due to some human error there may be a loss of records.

- FILE DAMAGED

When a computerized system is not there file is always lost due to some accident like spilling of water by some member on file accidentally. Besides some natural disaster like floods or fires may also damage the files.

- DIFFICULT TO SEARCH RECORD

When there is no computerized system there is always a difficulty in searching of records if the records are large in number.

- SPACE CONSUMING

After the number of records become large the space for physical storage of file and records also increases if no computerized system is implemented.

- COST CONSUMING

As there is no computerized system to add each record paper will be needed which will increase the cost for the management of library.

2.1.2 SYSTEM OBJECTIVES

- IMPROVEMENT IN CONTROL AND PERFORMANCE

The system is developed to cope up with the current issues and problems of library .The system can add user, validate user and is also bug free.

- SAVE COST

After computerized system is implemented less human force will be required to maintain the library thus reducing the overall cost.

- SAVE TIME

Librarian is able to search record by using few clicks of mouse and few search keywords thus saving his valuable time.

- OPTION FOR SUBMIT AND VIEW INFORMATION ABOUT PROPOSED BOOKS

Librarian will be able to provide a detailed description of workshops going in the college as well as in nearby colleges.

- LECTURE NOTES

Teacher have a facility to upload lectures notes in a pdf file having size not more than 10mb.

2.1.3 SYSTEM REQUIREMENTS

2.1.3.1 NON FUNCTIONAL REQUIREMENTS

2.1.3.1.1 Product Requirements

EFFICIENCY REQUIREMENT

When a library management system will be implemented librarian and user will easily access library as searching and book transaction will be very faster.

RELIABILITY REQUIREMENT

The system should accurately performs member registration, member validation, report generation, book transaction and search.

USABILITY REQUIREMENT

The system is designed for a user friendly environment so that student and staff of library can perform the various tasks easily and in an effective way.

INTEROPERABILITY REQUIREMENT

The system is not interoperable with external devices.

PORTABILITY REQUIREMENT

The system is portable to any system having windows or Linux.

2.1.3.1.2 Organizational Requirements

IMPLEMENTATION REQUIREMNTS

In implementing whole system it uses C/C++ Programming Language and for record keeping C++ file system are used.

DELIVERY REQUIREMENTS

The whole system is expected to be delivered in 5 months of time with a weekly evaluation by the project guide.

2.1.3.2 FUNCTIONAL REQUIREMENTS

2.1.3.2.1 ADMIN LOGIN

Description of feature

This feature used by the ADMINISTRTOR OR LIBRARIAN to login into system. They are required to enter user id and password before they are allowed to enter the system. The user id and password will be verified and if invalid id is there user is allowed to not enter the system.

Input

Correct user name, password and confirm password.

Output

Prompt to the administrator section menu.

2.1.3.2.2 MAINTAIN STUDENT MEMBER RECORDS

2.1.3.2.2.1 REGISTER NEW STUDENT

Description of feature

This feature used by admin/librarian to register new student in the library, the details of the student members, such as member id, name of the member, roll number, email id, address, sex, department, pin code, contact number.

Input

Student member details.

Output

Student successfully will be registered and proper message will be displayed.

2.1.3.2.2.2 MODIFY STUDENT RECORD

Description of feature

This feature used by admin/librarian to update student information who are registered in the library, such name of the member, email id, address, pin code, contact number.

Input

Choose which you want to update and the put student member id of the member whose record we want to modify.

Output

Corresponding student record will be successfully registered and proper message will be displayed.

2.1.3.2.2.3 DEREGISTER STUDENT

Description of feature

This feature used by admin/librarian to deregister a student member from the library for a specified member id.

Input

Student member id.

Output

Corresponding student record will be successfully deregistered and proper message will be displayed.

2.1.3.2.3 MAINTAIN STAFF MEMBER RECORDS

2.1.3.2.3.1 REGISTER NEW STAFF

Description of feature

This feature used by admin/librarian to register new staff in the library, the details of the staff members, such as member id, name of the member, email id, address, sex, department, designation, qualification, pin code, contact number.

Input

Staff member details.

Output

Staff successfully will be registered and proper message will be displayed.

2.1.3.2.3.2 MODIFY STAFF RECORD

Description of feature

This feature used by admin/librarian to update staff information who are registered in the library, such name of the member, email id, address, pin code, contact number.

Input

Choose which you want to update and the put staff member id of the member whose record we want to modify.

Output

Corresponding staff record will be successfully updated and proper message will be displayed.

2.1.3.2.3.3 DEREGISTER STAFF

Description of feature

This feature used by admin/librarian to deregister a staff member from the library for a specified member id.

Input

Staff member id.

Output

Corresponding staff record will be successfully deregistered and proper message will be displayed.

2.1.3.2.4 MAINTAIN BOOK RECORDS

2.1.3.2.4.1 REGISTER NEW BOOK

Description of feature

This feature used by admin/librarian to register new book in the library, the details of the books, such as book id, title of the book, author name(s) of the book, edition, publishers, price, and ISBN-13 number.

Input

Book details.

Output

Book will be successfully registered and proper message will be displayed.

2.1.3.2.4.2 MODIFY BOOK RECORD

Description of feature

This feature used by admin/librarian to update book information which are registered in the library, such as author's name, publisher's name, price, book's title.

Input

Choose which you want to update and the put book id of the book whose record we want to modify.

Output

Corresponding book record will be successfully updated and proper message will be displayed.

2.1.3.2.4.3 DEREGISTER BOOK

Description of feature

This feature used by admin/librarian to deregister a book from the library for a specified book id.

Input

Book id.

Output

Corresponding book record will be successfully deregistered and proper message will be displayed.

2.1.3.2.5 MANAGE TRANSACTIONS

2.1.3.2.5.1 BOOK ISSUE

Description of feature

This feature allows to issue book to a registered library member. If he/she has/have some book(s) with fine so he can't issue any book further until unless he/she return that book and pay fine. List of the book having fines issued to that member will be displayed then.

Input

Book ID of the book to be issued and member id of the book to whom the book would be issued.

Output

Book will be successfully issued to the member and proper message will be displayed. And also display the return date of the book.

2.1.3.2.5.2 Book Return

Description of feature

This feature used by admin/librarian this feature allows to return issued books.

Input

Book Id

Output

Corresponding book will be successfully deposited to the library and proper message will be displayed.

2.1.3.2.5.3 Book Return

Description of feature

This feature used by admin/librarian this feature allows to renew issued books.

Input

Book id.

Output

Corresponding book will be successfully renewed and proper message will be displayed.

2.1.3.2.6 Report Generation as per various query

Description of feature

This feature allows to generate various reports ay per query. Admin any library member will choose options like

1> complete list of books in the library
2> complete list of registered library members
3> complete issued books with corresponding members
4> list of books issued to a specified member
5> list of books issued to a member having fine
6> total fine up to the date for a specified member
7> fine for a specified book
And the system will generate corresponding lists from the files required.

2.1.3.2.7 SEARCH OPTIONS

2.1.3.2.7.1 BOOK SEARCH OPTIONS

Description of feature

This feature allows both admin and library members to search book catalogues based on the book ID, book title, author's name, and publishers.

Input

By choosing proper options and giving book ID or book title or author's name or publishers.

Output

Corresponding list/details of the book(s) will be displayed.

2.1.3.2.7.2 MEMBER SEARCH OPTIONS

Description of feature

This feature allows both admin and library members to search member register based on the member ID, email address and department.

Input

By choosing proper options and giving member ID, email address and department.

Output

Corresponding list/details of the member(s) will be displayed.

2.1.3.2.8 MANAGE PROPOSED ITEMS

2.1.3.2.8.1 ADD NEW PROPOSAL

Description of feature

This feature used by the normal users to add proposals for the books with minimal book details and member details

Input

Minimal of Book details i.e. book title, author's name, publishers name and Minimal of member details i.e. member Id, member name, and email ID

Output

Item will be added successfully and proper message will be displayed.

2.1.3.2.8.2 VIEW PROPOSED ITEMS

Description of feature

This feature used by both admin/librarian and normal users to see proposed book list.

Input

Choose the correct option.

Output

A list of proposed item with corresponding members will be displayed.

2.1.3.2.8.3 REMOVE ITEM RECORD

Description of feature

This feature used by admin/librarian to remove a proposed item record from the library for a specified book name.

Input

Book title.

Output

Corresponding book item record will be successfully removed and proper message will be displayed.

2.1.3.2.9. SYSTEM SETTINGS

Description of feature

This feature used only by the administrator or librarian to change the system settings like

1> Setting user name or password providing old and new password.
2> Setting book limits for staff/student members.
3> Setting no of book keeping days for staff/student members.
4> Setting fine amount for the student/staff members.

Input

Minimal of Book details i.e. book title, author's name, publishers name and
Minimal of member details i.e. member Id, member name, and email ID

Output

Item will be added successfully and proper message will be displayed.

2.1.3.3. SOFTWARE AND HARDWARE REQUIREMENTS

This section describes the software and hardware requirements of the system

2.1.3.3.1. SOFTWARE REQUIREMENTS

- *Operating system:* - Windows 10 is used as the operating system as it is stable and supports more features and is more user friendly.

- *Development tools and Programming language:* - C/C++ programming language is used to develop the entire project

- *For record saving:* - C/C++ file system are used.

- *IDE used:* - Code Blocks, version 17.2

- *Documentation:* - Star UML, Microsoft Office word 2013, Foxit Pro pdf reader.

2.1.3.3.2. HARDWARE REQUIREMENTS

- *Processor:* AMD A10-9600P RADEON R5, 10 COMPUTE CORES 4C+6Generation 2.40 GHz(recommended) is used as a processor because it is fast than other processors an provide reliable and stable and we can run our pc for long time. By using this processor we can keep on developing our project without any worries. Minimum Intel Dual Core Processor is required for good performance.

- *RAM:* - Ram 4 GB (recommended) is used as it will provide fast reading and writing capabilities and will in turn support in processing. Minimum 1 GB RAM is required for good performance.

2 .2 EXISTING VS. PROPOSED SYSTEMS

EXISTING SYSTEM:

- At present all the information maintained in manually.
- It is difficult to add new feature.
- So manually system is very slow.

Disadvantage:-

- The manual system required more time for processing.
- System is difficult and time consuming.

PROPOSED SYSTEM:

- Record are easy to maintain using computerized system.
- There are any types of report such as book detail member info,
- Book issue, member info, book issue, book return and stock.
- Information is easily updatable.
- Saving time and lowest cost.
- Reduce the paper work.
- Easy to searching the record.

2 .3 SOFTWARE TOOLS USED

2.3.1. C++ Programming Language: C++ is a general-purpose programming language. It has imperative, object-oriented and generic programming features, while also providing facilities for low-level memory manipulation.

It was designed with a bias toward system programming and embedded, resource-constrained and large systems, with performance, efficiency and flexibility of use as its design highlights++ has also been found useful in many other contexts, with key strengths being software infrastructure and resource-constrained applications,[6] including desktop applications, servers (e.g. e-commerce, web search or SQL servers), and performance-critical applications (e.g. telephone switches or space probes).C++ is a compiled language, with implementations of it available on many platforms. Many vendors provide C++ compilers, including the Free Software Foundation, Microsoft, Intel, and IBM.

2.3.2. Star UML: StarUML is an open source software modelling tool that supports UML (Unified Modelling Language). It is based on UML version 1.4, provides eleven different types of diagram and it accepts UML 2.0 notation. It actively supports the MDA (Model Driven Architecture) approach by supporting the UML profile concept and allowing to generate code for multiple languages.

Windows 2000, Windows XP, or higher; Microsoft Internet Explorer 5.0 or higher; 128 MB RAM (256MB recommended); 110 MB hard disc space (150MB space recommended)

Chapter 3

DESIGINING

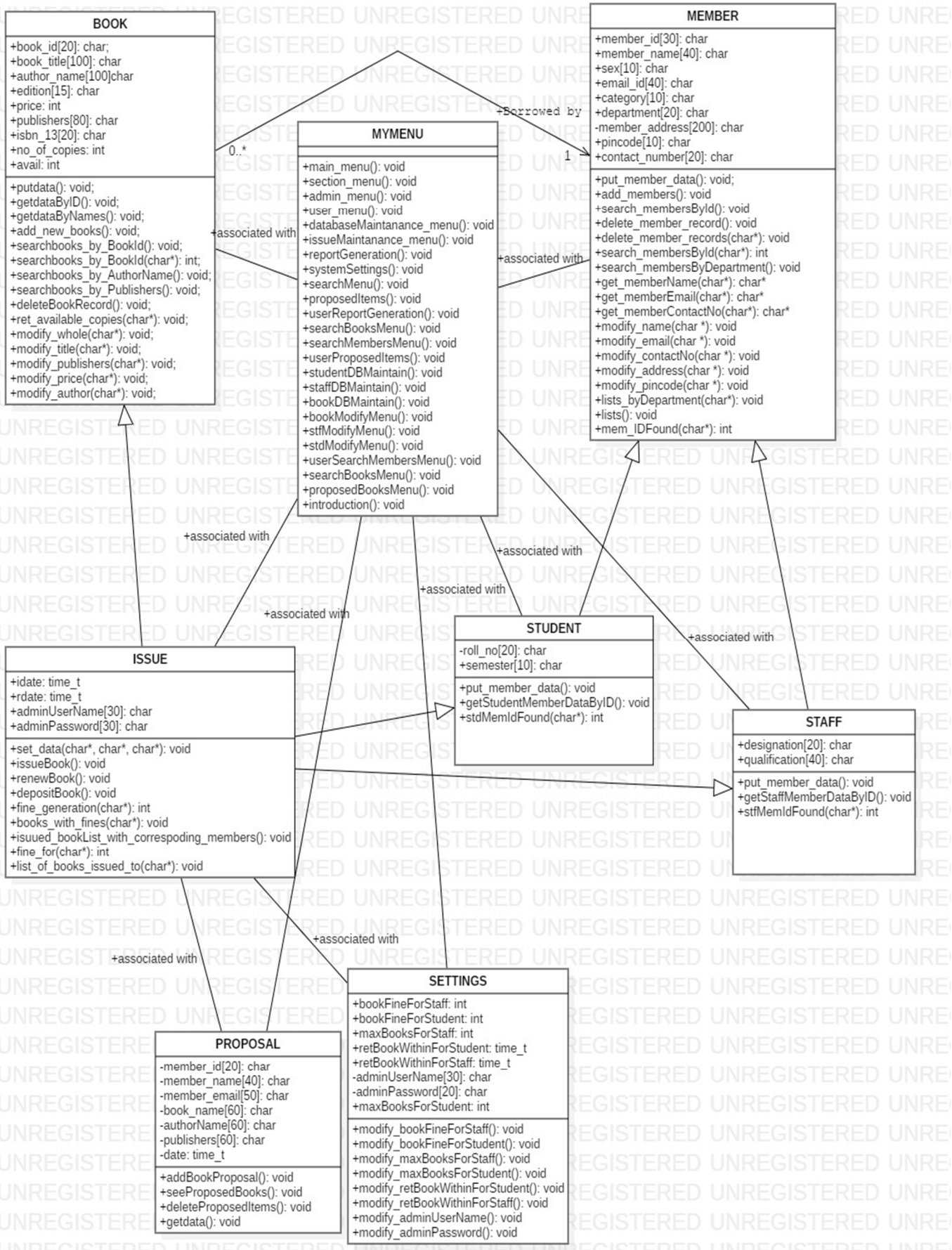
Topics Covered:

3.1 CLASS DIAGRAM

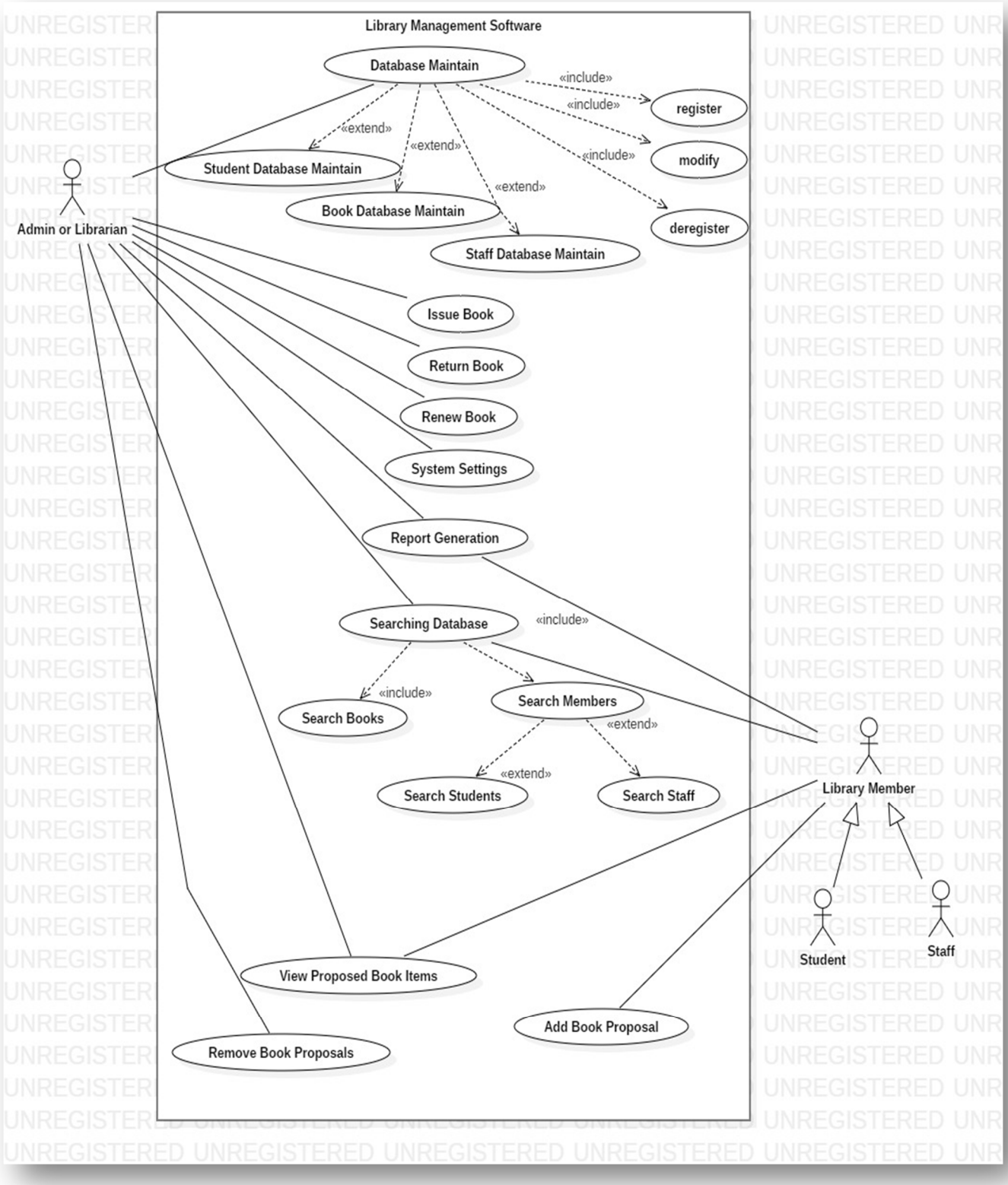
3.2 USE CASE DIAGRAM

3.3 GNATT CHART

3.1 CLASS DIAGRAM



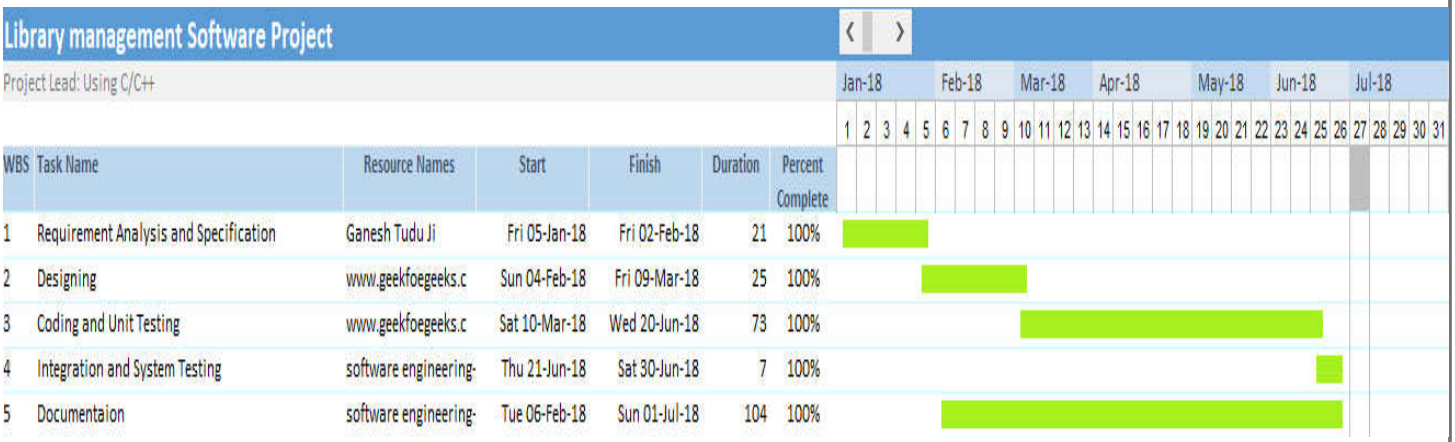
3.1 USE CASE DIAGRAM



3.2 GANTT CHART

Library Management Project Report Gantt Chart.xlsm

Click on the link for the further details.



WBS	Task Name	Resource Names	Start	Finish	Duration	Percent Complete
1	Requirement Analysis and Specification	Ganesh Tudu Ji	Fri 05-Jan-18	Fri 02-Feb-18	21	100%
2	Designing	www.geekfoegeeks.c	Sun 04-Feb-18	Fri 09-Mar-18	25	100%
3	Coding and Unit Testing	www.geekfoegeeks.c	Sat 10-Mar-18	Wed 20-Jun-18	73	100%
4	Integration and System Testing	software engineering-	Thu 21-Jun-18	Sat 30-Jun-18	7	100%
5	Documentaion	software engineering-	Tue 06-Feb-18	Sun 01-Jul-18	104	100%

Chapter 4

CODINGING

```
#include<iostream>
#include<conio.h>
#include<fstream>
#include<time.h>
#include<string.h>
#include<dos.h>
#include<process.h>
#include<windows.h>
#include<dir.h>
#include<stdio.h>
#include<stdlib.h>
/*******************************************************
          Some SPECIAL FUNCTION Body Definition
*******************************************************/

/********* For DELAY() Function*********/
void delay(unsigned int mseconds)
{
    clock_t goal = mseconds + clock();
```

```
    while (goal > clock());
}

/*********For GOTOXY( ) Function**********/

  COORD coord={0,0}; // this is global variable
                                    //center of axis is set to the top left corner of the screen
 void gotoxy(int x,int y)
 {
   coord.X=x;
 coord.Y=y;
 SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),coord);
 }
 /***********For Clear Screen Function()*********/
 void clrscr()
 {
     system("cls");
 }
using namespace std;
/**********************/
/*********************************
This Function will help to set text  colors
*********************************/

void SetColor(int ForgC)
 {
 WORD wColor;

  HANDLE hStdOut = GetStdHandle(STD_OUTPUT_HANDLE);
  CONSOLE_SCREEN_BUFFER_INFO csbi;

                       //We use csbi for the wAttributes word.
 if(GetConsoleScreenBufferInfo(hStdOut, &csbi))
 {
                 //Mask out all but the background attribute, and add in the forgournd color
      wColor = (csbi.wAttributes & 0xF0) + (ForgC & 0x0F);
      SetConsoleTextAttribute(hStdOut, wColor);
 }
 return;
}
/************************
```

```
The MMENU Class Definition
*************************/



class MYMENU
{
public :
void main_menu(void) ;
void section_menu(void);
void admin_menu(void);
void user_menu(void);
void databaseMaintanance_menu(void);
void issueMaintanance_menu(void);
void reportGeneration(void);
void systemSettings(void);
void searchMenu(void);
void proposedItems(void);
void userProposedItems(void);
void introduction(void) ;
void studentDBmaintain(void);
void bookDBmaintain(void);
void staffDBmaintain(void);
void bookModifyMenu(void);
void stdModifyMenu(void);
void stfModifyMenu(void);
void searchMembersMenu(void);
void userSearchMembersMenu(void);
void userReportGeneration(void);
void userSearchMenu(void);
void searchBooksMenu(void);
void proposedBooksMenu(void);
} ;



/*****************************
     Book Class Definition
*****************************/
class BOOK
{
    public:
```

```
char book_id[20];
char book_title[100];
char author_name[100];
char edition[15];
int price;
char publishers[80];
char isbn_13[20];
int no_of_copies;
int avail;
    void putdata(void)
{
    label:
    int flag=1;
    cout<<"\t\t\tEnter  Book details................"<<endl;
    cout<<"\n\t\tEnter id: ";
    cout<<"\n\t\tEnter title: ";
    cout<<"\n\t\tEnter aname: ";
    cout<<"\n\t\tEnter edition: ";
    cout<<"\n\t\tEnter price: ";
    cout<<"\n\t\tEnter publishers: ";
    cout<<"\n\t\tEnter ISBN-13: ";
    gotoxy(35,2);
    while(flag){
        gotoxy(35,2);
        cin>>book_id;
        if(flag=bookIdFound(book_id))
        {gotoxy(35,2);
            cout<<"Book Id Not Available";
            _getch();
                clrscr();
                goto label;
        }
    }
    cin.ignore();
    gotoxy(35,3);
    cin.getline(book_title,100);
    gotoxy(35,4);
    cin.getline(author_name,100);
    gotoxy(35,5);
    cin.getline(edition,15);
    gotoxy(35,6);
```

```
        cin>>price;
        cin.ignore();
        gotoxy(35,7);
        cin.getline(publishers,80);
        gotoxy(35,8);
        cin.getline(isbn_13,20);
        avail=1;
    }
    void getdataByID()
    {
        CONSOLE_SCREEN_BUFFER_INFO csbi;
        GetConsoleScreenBufferInfo(GetStdHandle( STD_OUTPUT_HANDLE ),&csbi);
        int m=csbi.dwCursorPosition.X;
        int n=csbi.dwCursorPosition.Y;
        gotoxy(m+12,n+1);
        cout<<"*******BOOK  DETAILS******"<<endl;
        cout<<"\n\tBook ID: "<<book_id;delay(1500);
        cout<<"\n\tBook Title: "<<book_title;delay(1400);
        cout<<"\n\tAuthor's Name: "<<author_name;delay(1400);
        cout<<"\n\tEdition: "<<edition;delay(1400);
        cout<<"\n\tPrice: "<<price;delay(1400);
        cout<<"\n\tPublishers: "<<publishers;delay(1400);
        cout<<"\n\tISBN-13 number: "<<isbn_13;delay(1400);
        if(avail==1)
        {cout<<"\n\t*****This Book is Available*****\n";delay(1400);}
        else
            {cout<<"\n\t*****This Book is NOT Available*****\n";delay(1400);}

    }
    void getdataByNames()
    {
        cout<<"\nID: "<<book_id;delay(1400);
        cout<<"   TITLE: "<<book_title;delay(1400);
        cout<<"   AUTHOR(s): "<<author_name;delay(1400);
        cout<<"   PUBLISHERS: "<<publishers;delay(1400);
    }
    void add_new_books();
    void searchbooks_by_BookId();
    int searchbooks_by_BookId(char*);
    void searchbooks_by_BookName();
    void searchbooks_by_AuthorName();
```

```
    void searchbooks_by_Publishers();
    void delete_BookRecord();
    int ret_available_copies(char*,char*);
    int ret_available_copies(char *);
    char* getBookName(char*);
    int ret_total_copies(char *);
    void searchbks_by_BookId(char *);
    int bookIdFound(char*);
    void modify_whole(char*);
    void modify_title(char*);
    void modify_author(char*);
    void modify_price(char*);
    void modify_publishers(char*);
    void modify_avail(char *);
    void lists(void);
    void lists_byAuthorName(char*);
    void lists_byPublishers(char*);
    void lists_byTitle(char*);
    int check_avail(char*);
};

   /************************
   MEMBER CLASS DEFINITION
   ************************/
   class MEMBER{

public:
   char member_id[30];
   char category[20];
   char member_name[40];
   char department[20];
   char member_address[200];
   char email_id[40];
   char sex[10];
   char pincode[10];
   char contact_no[15];
   public:
       MEMBER(){
       }
       MEMBER(char *x){
       strcpy(category,x);
```

```
        }
       void put_member_data();
         void add_members();
        void search_membersById();
        void delete_member_records();
         void delete_member_records(char*);
        int search_membersById(char *);
        void search_membersByEmail(void);
        void search_membersByDepartment(void);
        char* get_memberName(char*);
        char* get_memberEmail(char*);
        char* get_memberContactNo(char*);
        void modify_name(char *);
        void modify_email(char *);
        void modify_cntctNo(char *);
        void modify_address(char *);
        void modify_pincode(char*);
        void lists(void);
        void lists_byDepartment(char*);
        int memIDFound(char*);
    };


/*************************
STUDENT  CLASS  DEFINITION
*************************/
class STUDENT:public virtual MEMBER
{
public:
     char roll_no[20];
    char semester[10];

    STUDENT():MEMBER()
    {

    }
    STUDENT(char *s):MEMBER(s)
    {

    }
```

```
void put_member_data()
{
    label3:;
    MEMBER member;
    int flag=1;
    cout<<"\t\t\tEnter  Member details................."<<endl;
    cout<<"\n\t\tEnter Student Member Id: ";
    cout<<"\n\t\tEnter Member Name: ";
    cout<<"\n\t\tEnter Roll Number: ";
    cout<<"\n\t\tEnter  Sex: ";
    cout<<"\n\t\tEnter Semester: ";
    cout<<"\n\t\tEnter Department: ";
    cout<<"\n\t\tEnter Member Address: ";
    cout<<"\n\t\tEnter  PinCode Number: ";
    cout<<"\n\t\tEnter  Contact Number: ";
    cout<<"\n\t\tEnter Email Id: ";
    gotoxy(43,2);
                    while(flag){
        gotoxy(43,2);
     cin.getline(member_id,30);
        if( member.memIDFound(member_id))
        {
            flag=1;
            gotoxy(43,2);
            cout<<"Student Member Id Not Available";
            _getch();
            clrscr();
            goto label3;
            }
        else
        {
            flag=0;
        }
    }

    cin.ignore();
    gotoxy(43,3);
    cin.getline(member_name,40);
    gotoxy(43,4);
    cin.getline(roll_no,20);
    gotoxy(43,5);
```

```
        cin.getline(sex,10);
        gotoxy(43,6);
        cin.getline(semester,10);
        gotoxy(43,7);
        cin.getline(department,20);
        gotoxy(43,8);
        cin.getline(member_address,100);
        gotoxy(43,9);
        cin.getline(pincode,10);
        gotoxy(43,10);
        cin.getline(contact_no,15);
        gotoxy(43,11);
        cin.getline(email_id,40);
    }
    void getStudentMemberDataByID();
     int stdMemIdFound(char*);
};



/*************************
STAFF  CLASS  DEFINITION
*************************/
class STAFF:public virtual MEMBER
{
    public:
    char designation[20];
    char qualification[40];

public:
    STAFF():MEMBER()
    {
strcpy(category,"STAFF");
    }
    STAFF(char *s):MEMBER(s)
    {


    }

    void put_member_data()
    {
        label2:
```

```
MEMBER member;
int flag=1;
cout<<"\t\t\tEnter  Member details................."<<endl;
cout<<"\n\t\tEnter Staff Member Id: ";
cout<<"\n\t\tEnter Member Name: ";
cout<<"\n\t\tEnter Sex: ";
cout<<"\n\t\tEnter  Designation: ";
cout<<"\n\t\tEnter Qualification: ";
cout<<"\n\t\tEnter Department: ";
cout<<"\n\t\tEnter Member Address: ";
cout<<"\n\t\tEnter  PinCode Number: ";
cout<<"\n\t\tEnter  Contact Number: ";
cout<<"\n\t\tEnter Email Id: ";
gotoxy(43,2);
        while(flag){
    gotoxy(43,2);
 cin.getline(member_id,30);
    if(member.memIDFound(member_id))
    {
        flag=1;
        gotoxy(43,2);
        cout<<"Staff Member Id Not Available";
        _getch();
             clrscr();
             goto label2;
    }
    else
        {
        flag=0;
    }
}
cin.ignore();
gotoxy(43,3);
cin.getline(member_name,40);
gotoxy(43,4);
cin.getline(sex,10);
gotoxy(43,5);
cin.getline(designation,20);
gotoxy(43,6);
cin.getline(qualification,40);
gotoxy(43,7);
```

```
        cin.getline(department,20);
        gotoxy(43,8);
        cin.getline(member_address,100);
        gotoxy(43,9);
        cin.getline(pincode,10);
        gotoxy(43,10);
        cin.getline(contact_no,15);
        gotoxy(43,11);
        cin.getline(email_id,40);
    }

    void getStaffMemberDataByID();

    int stfMemIdFound(char *s);
};



  /*
  /**************************************
        TRANSACTION CLASS DEFINITION
  ***************************************/
  /*
  class TRANSACTIONS:public ISSUE
{
    time_t transaction_date;
    char status[10];

}  ;*/



  int MEMBER::memIDFound(char *memID)
  {
      int flag=0,flag2=0;
      STUDENT student("STUDENT");
      STAFF staff("STAFF");
      flag=student.stdMemIdFound(memID);
      flag2=staff.stfMemIdFound(memID);
      if(flag==1 || flag2==1)
      return 1;
      else
        return 0;
```

```
  }
```

```
/*****************************************
        ISSUE   CLASS    DEFINITION
*****************************************/

class ISSUE:public STAFF,public BOOK,public STUDENT{

public:
    time_t  idate;
    time_t  rdate;
    time_t  date;
    char status[10];
void Disp(void){
    cout<<idate<<endl;
    cout<<rdate<<endl;
    cout<<member_name<<endl;
    cout<<book_title<<endl;
}
ISSUE(){
    time_t t=time(NULL);
    idate=t;
    date=t;
    rdate=idate+1296000;
}
    void set_data(char*,char*,char*,char*);
    void issueBook(void);
    void depositBook(void);
    void renewBook(void);
    int fine_generation(char*);
    void books_with_fines(char*);
    void isuued_bookList_with_correspoding_members(void);
    int fine_for(char*);
    void list_of_books_issued_to(char*);
};
```

```
/************************************************
This function will return Book Tile for the corresponding Book_Id
************************************************/




char* BOOK::getBookName(char *bkID)
{
        int flag=0;
        ifstream ifs("BOOK.DAT",ios::binary);
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
        ifs.seekg(0,ios::beg);
   while(ifs.read((char*)this,sizeof(BOOK)))
    {
        if(!strcmp(bkID,book_id))
        {
         flag=1;
         break;
        }
    }
    ifs.close();
        }
            if(flag==1)
      {
        return book_title;
      }
    else
        return "Member Record Not found";
}


/*************************************************************************************************
    This function will check for the availability of the book for the corresponding BOOK_ID
*************************************************************************************************/
    int BOOK::check_avail(char *s){


    BOOK book;
   int  flag=0;
    char ch;
    ifstream ifs;
```

```
    ifs.open("BOOK.DAT",ios::in);
    while(ifs.read((char*)&book,sizeof(BOOK)))
    {
        if(book.avail==1  && !strcmp(s,book.book_id)){
            flag =1;
            break;
        }
    }
    if(flag==1){
        return 1;
    }
    else
    {return 0;
}
    }




/*******************************************************************************
            This function will check for the existence of the BOOK for the corresponding BOOK_ID
*******************************************************************************/



int BOOK::bookIdFound(char *s)
{
    BOOK book;
   int  flag=0;
    char ch;
    ifstream ifs;
    ifs.open("BOOK.DAT",ios::in);
    while(ifs.read((char*)&book,sizeof(BOOK)))
    {
        if(!strcmp(book.book_id,s)){
            flag =1;
            break;
        }
    }
    if(flag==1){
        return 1;
    }
    else
```

```
    {return 0;
}
}



/******************************************************************************************
            This function will generate the complete list of the BOOKs in the Library
******************************************************************************************/
void BOOK::lists()
{
BOOK bk;
clrscr() ;
int row = 6 , found=0, flag=0 ;
char ch ;
gotoxy(45,2) ;
cout <<"LIST OF BOOKS  ";
gotoxy(45,3) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
cout <<"CODE                     BOOK NAME                    AUTHOR  NAME             PUBLISHERS   COPIES";
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~" ;
fstream file ;
file.open("BOOK.DAT", ios::in) ;
file.seekg(0,ios::beg);
while (file.read((char *)&bk, sizeof(BOOK)))
{
    flag = 0 ;
delay(20) ;
        found = 1 ;
gotoxy(2,row) ;
cout <<bk.book_id;
gotoxy(18,row) ;
cout<<bk.book_title ;
gotoxy(55,row) ;
        cout<<bk.author_name ;
gotoxy(88,row) ;
cout<<bk.publishers;
gotoxy(109,row) ;
cout <<bk.ret_total_copies(bk.book_title);
gotoxy(18,row+1) ;
```

```
cout<<"STATUS: " ;
if(bk.avail==1)
cout<<"  Available" ;
else
   cout<<"  Not Available";
if ( row == 25 )
{
flag = 1 ;
row = 6 ;
gotoxy(1,27) ;
cout <<"Press any key to continue or Press <ESC> to exit" ;
ch = getch() ;
if (ch == 27)
break ;
clrscr() ;
gotoxy(33,2) ;
cout <<"LIST OF BOOKS" ;
gotoxy(32,3) ;
cout <<"~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
cout <<"CODE                 BOOK NAME                  AUTHOR  NAME               PUBLISHERS       COPIES" ;
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~" ;
}
else
row = row + 3 ;
}
if (!found)
{
gotoxy(5,10) ;
cout <<"Records not found" ;
gotoxy(5,12);
cout<<"Please enter valid Author Name";
_getch();}
if (!flag)
{
gotoxy(1,26) ;
cout <<"Press any key to continue..." ;
getche() ;
}
file.close () ;
```

```
}


/*******************************************************************************************************************
*
          This function will generate the list of the BOOK(s) in the Library with specified Author Name
*******************************************************************************************************************/



void BOOK::lists_byAuthorName(char *s)

{

BOOK bk;

clrscr() ;

int row = 6 , found=0, flag=0 ;

char ch ;

gotoxy(38,2) ;

cout <<"LIST OF BOOKS  WRITTEN  BY ";

gotoxy(37,3) ;

cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;

gotoxy(1,4) ;

cout <<"CODE                BOOK NAME                AUTHOR  NAME              PUBLISHERS      COPIES" ;

gotoxy(1,5) ;

cout
<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~" ;

fstream file ;

file.open("BOOK.DAT", ios::in) ;

file.seekg(0,ios::beg);

while (file.read((char *)&bk, sizeof(BOOK)))

{

    flag = 0 ;

delay(20) ;

        if(!strcmp(s,bk.author_name))

    {

        gotoxy(67,2);

        cout<<bk.author_name ;

        found = 1 ;

gotoxy(2,row) ;

cout<<bk.book_id;

gotoxy(18,row) ;

cout<<bk.book_title ;

gotoxy(55,row) ;

cout<<bk.author_name ;
```

```
gotoxy(88,row) ;
cout<<bk.publishers;
gotoxy(109,row) ;
cout<<bk.ret_total_copies(bk.book_title);
gotoxy(18,row+1) ;
cout<<"STATUS: " ;
if(bk.avail==1)
cout<<"   Available" ;
else
            cout<<"   Not Available";
if ( row == 25 )
{
flag = 1 ;
row = 6 ;
gotoxy(1,27) ;
cout <<"Press any key to continue or Press <ESC> to exit" ;
ch = getch() ;
if (ch == 27)
break ;
clrscr() ;
gotoxy(33,2) ;
cout <<"LIST OF BOOKS" ;
gotoxy(32,3) ;
cout <<"~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
cout <<"CODE            BOOK NAME                 AUTHOR  NAME               PUBLISHERS          COPIES" ;
gotoxy(1,5) ;
 cout
<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~" ;
}
else
row = row + 3 ;
}
}
if(!found)
{
gotoxy(5,10) ;
cout <<"Records not found" ;
gotoxy(5,12);
cout<<"Please enter valid Author Name";
_getch();
```

```
}if(!flag)
{
gotoxy(1,26) ;
cout <<"Press any key to continue..." ;
getche();}
file.close();
}




/***************************************************************************************************************
        This function will generate the list of the BOOK(s) in the Library with specified Author Name
***************************************************************************************************************/
void BOOK::lists_byPublishers(char *s)
{
BOOK bk;
clrscr() ;int row = 6 , found=0, flag=0 ;
char ch ;
gotoxy(38,2) ;
cout<<"LIST OF BOOKS  PUBLISHED  BY ";
gotoxy(35,3) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
cout<<"CODE                 BOOK NAME                  AUTHOR  NAME              PUBLISHERS   COPIES" ;
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~" ;
fstream file ;
file.open("BOOK.DAT", ios::in) ;
file.seekg(0,ios::beg);
while (file.read((char *)&bk, sizeof(BOOK)))
{
   flag = 0 ;
delay(20) ;
        if(!strcmp(s,bk.publishers))
    {
        gotoxy(67,2);
        cout<<bk.publishers ;
        found = 1 ;
gotoxy(2,row) ;
cout<<bk.book_id;
```

```
gotoxy(18,row) ;
cout<<bk.book_title ;
gotoxy(55,row) ;
cout<<bk.author_name ;
gotoxy(88,row) ;
cout<<bk.publishers;
gotoxy(109,row) ;
cout<<bk.ret_total_copies(bk.book_title);
gotoxy(18,row+1) ;
cout<<"STATUS: " ;
if(bk.avail==1)
cout<<"   Available" ;
else
            cout<<"   Not Available";
if ( row == 25 )
{
flag = 1 ;
row = 6 ;
gotoxy(1,27) ;
cout<<"Press any key to continue or Press <ESC> to exit" ;
ch = getch() ;
if (ch == 27)
break ;
clrscr() ;
gotoxy(33,2) ;
cout<<"LIST OF BOOKS" ;
gotoxy(32,3) ;
cout<<"~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
cout<<"CODE                  BOOK NAME                 AUTHOR  NAME               PUBLISHERS      COPIES" ;
gotoxy(1,5) ;

cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
}
else
row = row + 3 ;
}
}
if (!found)
{
gotoxy(5,10) ;
```

```
cout<<"Records not found" ;

gotoxy(5,12);

cout<<"Please enter valid Valid Name";

_getch();

}

if (!flag)

{

gotoxy(1,26) ;

cout <<"Press any key to continue..." ;

getche() ;

}

file.close () ;

}
```

```
/*************************************************************************

This function will generate the list of the BOOK(s) in the Library with specified Book Title

*************************************************************************/

void BOOK::lists_byTitle(char *s)

{

BOOK bk;

clrscr() ;

int row = 6 , found=0, flag=0 ;

char ch ;

gotoxy(34,2) ;

cout<<"LIST OF BOOKS  ENTITLED  WITH ";

gotoxy(33,3) ;

cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;

gotoxy(1,4) ;

cout<<"CODE                 BOOK NAME                  AUTHOR  NAME              PUBLISHERS        COPIES" ;

gotoxy(1,5) ;

cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~" ;

fstream file;

file.open("BOOK.DAT", ios::in) ;

file.seekg(0,ios::beg);

while(file.read((char *)&bk, sizeof(BOOK)))

{

   flag = 0 ;

delay(20) ;
```

```
        if(!strcmp(s,bk.book_title))
    {
        gotoxy(64,2);
        cout<<bk.book_title ;
        found = 1 ;
gotoxy(2,row) ;
cout<<bk.book_id;

gotoxy(18,row) ;
cout<<bk.book_title ;
gotoxy(55,row) ;
        cout<<bk.author_name ;
gotoxy(88,row) ;
cout<<bk.publishers;
gotoxy(109,row) ;
cout<<bk.ret_total_copies(bk.book_title);
gotoxy(18,row+1) ;
cout<<"STATUS: " ;
if(bk.avail==1)
cout<<"   Available" ;
else
           cout<<"   Not Available";
if ( row == 25 )
{
flag = 1 ;
row = 6 ;
gotoxy(1,27) ;
cout<<"Press any key to continue or Press <ESC> to exit" ;
ch = getch() ;
if (ch == 27)
break ;
clrscr() ;
gotoxy(33,2) ;
cout<<"LIST OF BOOKS" ;
gotoxy(32,3) ;
cout<<"~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
cout<<"CODE                 BOOK NAME                  AUTHOR  NAME                PUBLISHERS       COPIES" ;
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~" ;
}
```

```
else
row = row + 3 ;
}}
if (!found)
{
gotoxy(5,10) ;
cout<<"Records not found" ;
gotoxy(5,12);
cout<<"Please enter valid Valid Name";
_getch();
}
if (!flag)
{
gotoxy(1,26) ;
cout<<"Press any key to continue..." ;
getche() ;
}
file.close () ;
}
```

```
/*****************************************************
        This member function is to ADD NEW BOOKS
*****************************************************/
void BOOK::add_new_books()
{
        char ch,ch2;
        ofstream ofs("BOOK.DAT",ios::app|ios::binary);//|ios::out);
        if(!ofs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
   do
    {
        clrscr();
        putdata();
        cout<<"\n\n\t\t!!!!Do you want to save this BOOK record ??!!!!";
        cout<<"\n\n\t\tPress Enter to save Record \n             or              \n\t\tPress  ESC to Exit";
        ch2=_getch();
        if(ch2==13){
```

```
                cout<<"\n\n\t\tRecord is Saving ";
            int x=12;
           while(x>0)
           {cout<<". ";
             delay(400);
             x--;
           }
        cout<<"\n\n\t\t!!!!!*****Data successfully saved*****!!!!!!";

        ofs.write((char*)this,sizeof(BOOK));
        }
        else if(ch2==27)
        {
            break;
        }
        else
            cout<<" ";
               // ofs.write((char*)&book,sizeof(book));
                cout<<"\n\n\t\tDo you want to enter more records? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
    }while(ch=='y'||ch=='Y');
        }
        ofs.close();
    }




 /*************************************************************************************************
   This member function is to SEARCH books by its ID and print the book details of the corresponding
   *************************************************************************************************/
void BOOK::searchbooks_by_BookId()
    {
        int flag;
        char ch;
        char s[20];
        ifstream ifs("BOOK.DAT",ios::binary);
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
```

```
   do{
       clrscr();
        flag=0;
        ifs.seekg(0,ios::beg);
   cout<<"\nEnter Book ID : ";
        //cin.ignore();
    cin.getline(s,20);
   while(ifs.read((char*)this,sizeof(BOOK)))
    {
        if(!strcmp(s,book_id))
        {cout<<"\n*****Book Entry found****\n";
            getdataByID();
            flag=1;
            cout<<"\nPress any key to continue*********\n";
            _getch();
            fflush(stdin);
            break;
        }
    }
        if(flag==0)
        {cout<<"\n!!!!!!!!!!!!Book record is not FOUND!!!!!!!!!!!!\n";
            cout<<"\n****Please check the BOOK ID you have entered and try again****\n";
        ifs.clear();
        }
           cout<<"\nDo you want to search more books? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
    }while(ch=='y'||ch=='Y');
        }
   ifs.close();
   }

/*****************************************************************
This function will print the Book Title of he corresponding Book ID
*****************************************************************/


   void BOOK::searchbks_by_BookId(char *s)
    {
        ifstream ifs("BOOK.DAT",ios::binary);
```

```
        if(!ifs)
           cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
       else{
       ifs.seekg(0,ios::beg);
  while(ifs.read((char*)this,sizeof(BOOK)))
   {
       if(!strcmp(s,book_id))
       {
           cout<<book_title;
           break;
       }
   }
  ifs.close();
  }
   }



  int BOOK::searchbooks_by_BookId(char *s)
   {
       int flag=0;
       ifstream ifs("BOOK.DAT",ios::binary);
        if(!ifs)
       cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
       else{
       ifs.seekg(0,ios::beg);
  while(ifs.read((char*)this,sizeof(BOOK)))
   {
       if(!strcmp(s,book_id))
       {
        flag=1;
        break;
       }
   }
   ifs.close();
       }
           if(flag==1)
       return 1;
   else
       return 0;
   }
```

```
/*****************************************************
This member function is to SEARCH books by its TITLE
*****************************************************/
void BOOK::searchbooks_by_BookName()
 {
     int flag;
     char ch;
     char s[100];
     ifstream ifs("BOOK.DAT",ios::binary);
      if(!ifs)
         cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
     else{
do{
    clrscr();
     flag=0;
     ifs.seekg(0,ios::beg);
cout<<"\nEnter the NAME of the BOOK : ";
     //cin.ignore();
     cin.getline(s,100);
while(ifs.read((char*)this,sizeof(BOOK)))
 {
     if(!strcmp(s,book_title))
     {
         if(flag==0)
         cout<<"\n*****Book Entry found****\n";
         //getdataByNames();
         BOOK book;
         book.lists_byTitle(s);
         flag=1;
         break;
     }
 }
 if(flag==1)
     cout<<"\n\nThe No of AVAILABLE copies :  "<<ret_available_copies(book_title);
     if(flag==0)
     {
         cout<<"\n!!!!!!!!!!!Book record is not FOUND!!!!!!!!!!!!\n";
         cout<<"\n****Please check the Name of the BOOK you have entered and try again****\n";
         ifs.clear();
```

```
        }
        cout<<"\nDo you wan to SEARCH books' list again? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
    }while(ch=='y'||ch=='Y');
        }
   ifs.close();
   }




   /************************************************************
   This member function is to SEARCH books by its AUTHER'S   NAME
   ************************************************************/
void BOOK::searchbooks_by_AuthorName()
    {
        int flag;
        char ch;
        char s[80];
        ifstream ifs("BOOK.DAT",ios::binary);
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
   do{
       clrscr();
        flag=0;
        ifs.seekg(0,ios::beg);
        cout<<"Enter the AUTHOR's name : ";
        cin.getline(s,100);
   while(ifs.read((char*)this,sizeof(BOOK)))
    {
        if(!strcmp(s,author_name))
        {
            if(flag==0)
            cout<<"\n*****Book Entry found****\n";
            BOOK book;
            book.lists_byAuthorName(s);
            flag=1;
            break;
        }
```

```
    }
        if(flag==0)
        {cout<<"\n!!!!!!!!!!!Book record is not FOUND!!!!!!!!!!!!\n";
            cout<<"\n****Please check the Name of he AUTHOR you have entered and try again****\n";
        ifs.clear();
        }
        cout<<"\nDo you wan to SEARCH books' list again? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
    }while(ch=='y'||ch=='Y');
        }
   ifs.close();
   }



   /***********************************************************
    This member function is to SEARCH books by its PUBLISHERS
   ************************************************************/
void BOOK::searchbooks_by_Publishers()
    {
        int flag;
        char ch;
        char s[80];
        ifstream ifs("BOOK.DAT",ios::binary);
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
   do{
       clrscr();
        flag=0;
        ifs.seekg(0,ios::beg);
   cout<<"\nEnter PUBLISHER's name : ";
        //cin.ignore();
        cin.getline(s,80);
   while(ifs.read((char*)this,sizeof(BOOK)))
    {
        if(!strcmp(s,publishers))
        {
            if(flag==0)
```

```
            cout<<"\n*****Book Entry found****\n";
            //getdataByNames();
            BOOK book;
            book.lists_byPublishers(s);
            flag=1;
            break;
        }
    }
        if(flag==0)
        {cout<<"\n!!!!!!!!!!!Book record is not FOUND!!!!!!!!!!!!\n";
            cout<<"\n****Please check the PUBLISHER'S name you have entered and try again****\n";
        ifs.clear();
        }
    cout<<"\nDo you wan to SEARCH  books' list again? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
    }while(ch=='y'||ch=='Y');
        }
   ifs.close();
   }









   /*****************************************************
   This member function is to DELETE book record by its ID
  ******** ***********************************************/
   void BOOK::delete_BookRecord()
   {
       clrscr();
       BOOK bk,bk2;
       char s[30],ch;
        ifstream ifs("BOOK.DAT",ios::binary|ios::in);
        ofstream ofs;
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
                do{
```

```
                cout<<"\nEnter BOOK ID: ";
                cin>>s;
if(searchbooks_by_BookId(s)==1)
        {
                   getdataByID();
        cout<<"\nDo you want to DELETE this record from file? (y/n)";
ch=_getch();
if(ch=='y' || ch=='Y')
   {
       cout<<"\nProcessing ";
                  int x=12;
   while(x>0)
   {
       cout<<". ";
     delay(400);
     x--;
   }
    cout<<"\n";
         ofs.open("temp.DAT",ios::binary|ios::out);
        ifs.read((char*)&bk2,sizeof(BOOK));

    while(!ifs.eof())
    {
            if(strcmp(s,bk2.book_id))
            {
                ofs.write((char*)&bk2,sizeof(BOOK));
            }
            ifs.read((char*)&bk2,sizeof(BOOK));
    }
    ifs.close();
    ofs.close();
    remove("BOOK.DAT");
    rename("temp.DAT","BOOK.DAT");
    cout<<"\nThe Book Record has successfully deleted from the Database"<<endl;
   }
    }
    else
        {
        cout<<"\nBook Record Not Found\n";
    }
    cout<<"\nPress any key to continue*********\n";
```

```
        _getch();
        fflush(stdin);
           cout<<"\nDo you want to DELETE more records? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
                }while(ch=='y'||ch=='Y');
        }
   }
```

```
/*****************************************************************************************************
This function will return available copies in the Library for the corresponding Book_id and Book_title
*****************************************************************************************************/

   int BOOK::ret_available_copies(char *s,char *r)
   {
       BOOK bk;
       int availcount=0;
       ifstream ifs;
       ifs.open("BOOK.DAT",ios::binary|ios::in);
       if(!ifs)
       {
           cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
       }
       else
       {
           while(ifs.read((char*)&bk,sizeof(BOOK)))
           {
               if(strcmp(s,bk.book_title)==0  && strcmp(r,bk.book_id)==0 && bk.avail==1)
                availcount++;
           }
       }
       ifs.close();
       return availcount;
   }
```

```
   /*******************************************************************************
This function will return available copies in the Library for the corresponding Book_title
*******************************************************************************/
   int BOOK::ret_available_copies(char *s)
   {
        int availcount=0;
        ifstream ifs("BOOK.DAT",ios::binary);
        if(!ifs)
        {
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        }
        else
        {
            while(ifs.read((char*)this,sizeof(BOOK)))
            {
                if(strcmp(s,book_title)==0 && avail==1)
                 availcount++;
            }
        }
        ifs.close();
        return (ret_total_copies(s)- availcount);
   }
```

```
/*******************************************************************************
This function will return the total number of copies in the library for a same book
*******************************************************************************/
   int BOOK::ret_total_copies(char *s)
   {BOOK bk;
        int mycount=0;
        ifstream ifs("BOOK.DAT",ios::binary);
        if(!ifs)
        {
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        }
```

```
        else
        {
            while(ifs.read((char*)&bk,sizeof(BOOK)))
            {
                if(strcmp(s,bk.book_title)==0)
                 mycount++;
            }
        }
        ifs.close();
        return mycount;
    }
```

```
/*************************************************************
This function helps to modify whole book details for the corresponding Book ID
*************************************************************/
    void BOOK::modify_whole(char *BookId)
    {
        clrscr();
        BOOK bk;
        char ch,ch2;
        do{
        if(searchbooks_by_BookId(BookId)==1)
        {
                    cout<<"\n******Book record is found******";
                    getdataByID();
                    bk=*this;
                    cout<<"\nDo you realy want to update this record? (y/n)";
                    ch=_getch();
                    if(ch=='y'||ch=='Y')
                    {
                        cout<<"\nEnter New  Id: ";
                        cin.getline(bk.book_id,20);
                        cout<<"\nEnter New  Title: ";
                        cin.getline(bk.book_title,100);
                        cout<<"\nEnter New  Author's name: ";
                        cin.getline(bk.author_name,100);
                        cout<<"\nEnter New  Price: ";
```

```
                    cin>>bk.price;cin.ignore();
                    cout<<"\nEnter New  Publisher's Name: ";
                    cin.getline(bk.publishers,80);

                                cout<<"Processing ";
                     int x=12;
                     while(x>0)
                     {cout<<". ";
                     delay(400);
                     x--;
                     }
                    fstream fs("BOOK.DAT",ios::in|ios::out|ios::binary|ios::ate);
                    fs.seekg(0,ios::beg);
                    while(fs.read((char*)this,sizeof(BOOK)))
                    {
                        if(!strcmp(BookId,book_id))
                        {
                            fs.seekp(fs.tellp()-sizeof(BOOK));
                            fs.write((char*)&bk,sizeof(bk));
                        }
                    }
                    fs.close();
                    cout<<"\n*****Your BOOK list is successfully modified*****";
                }
      }
      else
      {cout<<"\n!!!!!BOOK record not found!!!!!";}
           cout<<"\nDo you wan to MODIFY more records? (y/n)";

       ch2=_getch();
       if(ch2=='n' || ch2=='N')
           break;
               }while(ch2=='y'||ch2=='Y');
   }
```

```
/************************************************************
```

```
This function helps to modify Author Name of a book  for the corresponding Book ID
************************************************************/
    void BOOK::modify_author(char *BookId)
   {
       clrscr();
       BOOK bk;
       char ch,ch2;
       do{
       if(searchbooks_by_BookId(BookId)==1)
       {
                   cout<<"\n******Book record is found******";
                   getdataByID();
                   bk=*this;
                   cout<<"\nDo you realy want to update this record? (y/n)";
                   ch=_getch();
                   if(ch=='y'||ch=='Y')
                   {

                       cout<<"\nEnter New  Author's name: ";
                       cin.getline(bk.author_name,100);
                      cout<<"\nPress any key to continue****";
                       _getch();
                       cout<<"\nProcessing ";
                                  int x=12;
                    while(x>0)
                    {cout<<". ";
                    delay(400);
                    x--;
                    }

                       fstream fs("BOOK.DAT",ios::in|ios::out|ios::binary|ios::ate);
                       fs.seekg(0);
                       while(fs.read((char*)this,sizeof(BOOK)))
                       {
                           if(!strcmp(BookId,book_id))
                           {
                               fs.seekp(fs.tellp()-sizeof(BOOK));
                               fs.write((char*)&bk,sizeof(bk));
                           }
                       }
                       fs.close();
```

```
                    cout<<"\n*****Author's name  is successfully modified*****";
                }
    }
    else
    {cout<<"\n!!!!!BOOK record not found!!!!!";}
         cout<<"\nDo you wan to MODIFY more records? (y/n)";

     ch2=_getch();
     if(ch2=='n' || ch2=='N')
         break;
             }while(ch2=='y'||ch2=='Y');
}
```

```
/*************************************************************************************
     This function helps to modify Title  of a book  for the corresponding Book ID
 *************************************************************************************/
   void BOOK::modify_title(char *BookId)
{
    clrscr();
    BOOK bk;
    char ch,ch2;
    do{
    if(searchbooks_by_BookId(BookId)==1)
    {
                cout<<"\n******Book record is found******";
                getdataByID();
                bk=*this;
                cout<<"\nDo you realy want to update this record? (y/n)";
                ch=_getch();
                if(ch=='y'||ch=='Y')
                {
                    cout<<"\nEnter New  Title: ";
                    cin.getline(bk.book_title,100);
                    cout<<"\nPress any key to continue****";
                    _getch();
                    cout<<"\nProcessing ";
                              int x=12;
```

```
            while(x>0)
            {cout<<". ";
              delay(400);
              x--;
            }
                        fstream fs("BOOK.DAT",ios::in|ios::out|ios::binary|ios::ate);
                        fs.seekg(0);
                        while(fs.read((char*)this,sizeof(BOOK)))
                        {
                            if(!strcmp(BookId,book_id))
                            {
                                fs.seekp(fs.tellp()-sizeof(BOOK));
                                fs.write((char*)&bk,sizeof(bk));
                            }
                        }
                        fs.close();
                        cout<<"\n*****Book's Title is successfuly modified*****";
                    }
        }
        else
        {cout<<"\n!!!!!BOOK record not found!!!!!";}
             cout<<"\nDo you wan to MODIFY more records? (y/n)";

         ch2=_getch();
         if(ch2=='n' || ch2=='N')
             break;
                 }while(ch2=='y'||ch2=='Y');
    }
```

```
/*****************************************************************************
This function helps to modify PUBLISHERS  of a book  for the corresponding Book ID
*****************************************************************************/

       void BOOK::modify_publishers(char *BookId)
    {
         clrscr();
         BOOK bk;
```

```
char ch,ch2;
do{
if(searchbooks_by_BookId(BookId)==1)
{
            cout<<"\n******Book record is found******";
            getdataByID();
            bk=*this;
            cout<<"\nDo you realy want to update this record? (y/n)";
            ch=_getch();
            if(ch=='y'||ch=='Y')
            {
                cout<<"\nEnter New  Publisher Name: ";
                cin.getline(bk.publishers,100);
                cout<<"\nPress any key to continue****";
                _getch();
                cout<<"\nProcessing ";
                           int x=12;
             while(x>0)
             {cout<<". ";
             delay(400);
             x--;
             }
                fstream fs("BOOK.DAT",ios::in|ios::out|ios::binary|ios::ate);
                fs.seekg(0);
                while(fs.read((char*)this,sizeof(BOOK)))
                {
                    if(!strcmp(BookId,book_id))
                    {
                        fs.seekp(fs.tellp()-sizeof(BOOK));
                        fs.write((char*)&bk,sizeof(bk));
                    }
                }
                fs.close();
                cout<<"\n*****Book's Publishers Name is successfuly modified*****";
            }
}
else
{cout<<"\n!!!!!BOOK record not found!!!!!";}
     cout<<"\nDo you wan to MODIFY more records? (y/n)";

 ch2=_getch();
```

```
        if(ch2=='n' || ch2=='N')
            break;
                }while(ch2=='y'||ch2=='Y');
   }









   /************************************************************
This function helps to modify Price  of a book  for the corresponding Book ID
************************************************************/


      void BOOK::modify_price(char *BookId)
   {
       clrscr();
       BOOK bk;
       char ch,ch2;
       do{
       if(searchbooks_by_BookId(BookId)==1)
       {
                   cout<<"\n******Book record is found******";
                   getdataByID();
                   bk=*this;
                   cout<<"\nDo you realy want to update this record? (y/n)";
                   ch=_getch();
                   if(ch=='y'||ch=='Y')
                   {
                       cout<<"\nEnter New  Price: ";
                       cin>>bk.price;
                       cout<<"\nPress any key to continue****";
                       _getch();
                       cout<<"\nProcessing ";
                                  int x=12;
                    while(x>0)
                    {cout<<". ";
                    delay(400);
                    x--;
                    }

                       fstream fs("BOOK.DAT",ios::in|ios::out|ios::binary|ios::ate);
```

```
                        fs.seekg(0);
                        while(fs.read((char*)this,sizeof(BOOK)))
                        {
                            if(!strcmp(BookId,book_id))
                            {
                                fs.seekp(fs.tellp()-sizeof(BOOK));
                                fs.write((char*)&bk,sizeof(bk));
                            }
                        }
                        fs.close();
                        cout<<"\n*****Book's Price is successfuly modified*****";
                    }
        }
        else
        {cout<<"\n!!!!!BOOK record not found!!!!!";}
             cout<<"\nDo you wan to MODIFY more records? (y/n)";

         ch2=_getch();
         if(ch2=='n' || ch2=='N')
             break;
                 }while(ch2=='y'||ch2=='Y');
    }




void STUDENT::getStudentMemberDataByID()
     {
         CONSOLE_SCREEN_BUFFER_INFO csbi;
         GetConsoleScreenBufferInfo(GetStdHandle( STD_OUTPUT_HANDLE ),&csbi);
         int m=csbi.dwCursorPosition.X;
         int n=csbi.dwCursorPosition.Y;
         gotoxy(m+12,n+1);
         MEMBER mem;
         mem=*this;
         cout<<"\n*******MEMBER  DETAILS******\n"<<endl;
         cout<<"\n\t\t Student Member Id: "<<member_id<<endl;delay(500);
         cout<<"\t\tMember Category: "<<category<<endl;delay(500);
         cout<<"\t\tMember Name: "<<member_name<<endl;delay(500);
         cout<<"\t\tRoll Number: "<<roll_no<<endl;delay(500);
         cout<<"\t\t Sex: "<<sex<<endl;delay(500);
         cout<<"\t\tSemester: "<<semester<<endl;delay(500);
```

```
        cout<<"\t\tDepartment: "<<department<<endl;delay(500);
        cout<<"\t\tMember Address: "<<member_address<<endl;delay(500);
        cout<<"\t\t PinCode Number: "<<pincode<<endl;delay(500);
        cout<<"\t\t Contact Number: "<<contact_no<<endl;delay(500);
        cout<<"\t\tEmail Id: "<<email_id<<endl;delay(500);
    }

    int STUDENT::stdMemIdFound(char *s)
{
    STUDENT stud;
    int  flag=0;
    char ch;
    ifstream ifs;
    ifs.open("STUDENT.DAT",ios::in);
    while(ifs.read((char*)&stud,sizeof(STUDENT)))
    {
        if(!strcmp(stud.member_id,s)){
            flag =1;
            break;
        }
    }
    if(flag==1){
        return 1;
    }
    else
    {return 0;
}
}



void STAFF::getStaffMemberDataByID()
{
        CONSOLE_SCREEN_BUFFER_INFO csbi;
        GetConsoleScreenBufferInfo(GetStdHandle( STD_OUTPUT_HANDLE ),&csbi);
        int m=csbi.dwCursorPosition.X;
        int n=csbi.dwCursorPosition.Y;
        gotoxy(m+12,n+1);
        cout<<"\t\t*******MEMBER  DETAILS******\n"<<endl;
        cout<<"\n\t\tStaff Member Id: "<<member_id<<endl;delay(500);
        cout<<"\t\tMember Category: "<<category<<endl;delay(500);
```

```
        cout<<"\t\tMember Name: "<<member_name<<endl;delay(500);
        cout<<"\t\tSex: "<<sex<<endl;delay(500);
        cout<<"\t\tDesignation: "<<designation<<endl;delay(500);
        cout<<"\t\tQualification: "<<qualification<<endl;delay(500);
        cout<<"\t\tDepartment: "<<department<<endl;delay(500);
        cout<<"\t\tMember Address: "<<member_address<<endl;delay(500);
        cout<<"\t\tPinCode Number: "<<pincode<<endl;delay(500);
        cout<<"\t\tContact Number: "<<contact_no<<endl;delay(500);
        cout<<"\t\tEmail Id: "<<email_id<<endl;delay(500);
    }




/************************************************************************
        This function is to generate the list of all members
************************************************************************/

void MEMBER::lists(void){
   if(!strcmp(category,"STUDENT"))
        {
            STUDENT student("STUDENT");
    clrscr() ;
int row = 6 , found=0, flag=0 ;
char ch ;
    gotoxy(45,2) ;
SetColor(11);
cout <<"    LIST   OF   STUDENT    MEMBERS" ;
gotoxy(45,3) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
SetColor(13);
cout <<"MEMBER ID     ROLL NO       NAME        SEMESTER  DEPARTMENT          EMAIL           CONTACT  NO" ;
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
fstream file ;
file.open("STUDENT.DAT", ios::in) ;
file.seekg(0,ios::beg);
while (file.read((char *) &student, sizeof(STUDENT)))
{
```

```
flag = 0 ;
delay(20) ;
found = 1 ;
gotoxy(2,row) ;
cout<<student.member_id;
gotoxy(14,row) ;
cout<<student.roll_no;
gotoxy(25,row) ;
cout<<student.member_name ;
gotoxy(50,row) ;
cout<<student.semester ;
gotoxy(61,row) ;
cout<<student.department ;
gotoxy(72,row) ;
cout<<student.email_id ;
gotoxy(106,row) ;
cout<<student.contact_no;
if ( row == 50 )
{
flag = 1 ;
row = 6 ;
gotoxy(1,65) ;
cout <<"Press any key to continue or Press <ESC> to exit" ;
ch = _getch() ;
if (ch == 27)
break ;
clrscr() ;
gotoxy(33,2) ;
cout <<"LIST  OF  STUDENT   MEMBERS " ;
gotoxy(32,3) ;
cout <<"~~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
cout <<"MEMBER ID     ROLL NO          NAME          SEMESTER  DEPARTMENT         EMAIL        CONTACT  NO" ;
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~" ;
}
else
row = row + 2 ;
}
if (!found)
{
```

```
gotoxy(5,10) ;
cout <<"\7Records not found" ;
}
if (!flag)
{
gotoxy(1,25) ;
cout <<"Press any key to continue..." ;
getche() ;
}
file.close () ;
        }

        if(!strcmp(category,"STAFF"))
        {
            STAFF staff("STAFF");
            clrscr();
    int row = 6 , found=0, flag=0 ;
char ch ;
gotoxy(45,2) ;
cout <<"  LIST   OF   STAFF    MEMBERS" ;
gotoxy(45,3) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
cout <<"MEMBER ID      NAME     QUALIFICATION   DESIGNATION   DEPARTMENT       EMAIL          CONTACT  NO" ;
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~" ;
fstream file ;
file.open("STAFF.DAT", ios::in) ;
file.seekg(0,ios::beg);
while (file.read((char *) &staff, sizeof(STAFF)))
{
flag = 0 ;
delay(20) ;
found = 1 ;
gotoxy(2,row) ;
cout <<staff.member_id;
gotoxy(12,row) ;
cout<<staff.member_name ;
gotoxy(32,row) ;
cout<<staff.qualification ;
gotoxy(49,row) ;
```

```
cout<<staff.designation ;
gotoxy(65,row) ;
cou<<staff.department ;
gotoxy(74,row) ;
cou<<staff.email_id ;
gotoxy(105,row) ;
cout <<staff.contact_no;
gotoxy(18,row+1) ;
        if ( row == 22 )
{
flag = 1 ;
row = 6 ;
gotoxy(1,60) ;
cout <<"Press any key to continue or Press <ESC> to exit" ;
ch = getch() ;
if (ch == 27)
break ;
clrscr() ;
gotoxy(33,2) ;
cout <<"LIST  OF  STUDENT   MEMBERS " ;
gotoxy(32,3) ;
cout <<"~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
cout <<"MEMBER ID                   NAME            DEPARTMENT          DESIGNATION        EMAIL
CONTACT  NO" ;
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~" ;
}
else
row = row + 3 ;}
if (!found)
{
gotoxy(5,10) ;
cout <<"\7Records not found" ;
}
if (!flag)
{
gotoxy(1,25) ;
cout <<"Press any key to continue..." ;
getche() ;}file.close () ;
        }
```

```
}



/*********************************************************************************
       This function will generate the list of members with specified member Id
**********************************************************************************/



void MEMBER::lists_byDepartment(char *dept){
   if(!strcmp(category,"STUDENT"))
        {
            STUDENT student("STUDENT");
clrscr() ;
int row = 6 , found=0, flag=0 ;
char ch ;
//system("COLOR  0D");
gotoxy(35,2) ;
SetColor(11);
cout <<" LIST  OF  STUDENT  MEMBERS  IN THE DEPARTMENT OF  " ;
gotoxy(34,3) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
SetColor(13);
cout <<"MEMBER ID     ROLL NO          NAME            SEMESTER   DEPARTMENT              EMAIL
CONTACT  NO" ;
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~" ;
fstream file ;
file.open("STUDENT.DAT", ios::in) ;
file.seekg(0,ios::beg);
while (file.read((char *) &student, sizeof(STUDENT)))
{
flag = 0 ;
delay(20) ;
if(!strcmp(dept,student.department))
        {
            gotoxy(86,2);
            cout<<dept;
found = 1 ;
gotoxy(2,row) ;
cout <<student.member_id;
```

```
gotoxy(14,row) ;
cout <<student.roll_no;
gotoxy(25,row) ;
cout<<student.member_name ;
gotoxy(50,row) ;
cout<<student.semester ;
gotoxy(61,row) ;
cout<<student.department ;
gotoxy(72,row) ;
cout<<student.email_id ;
gotoxy(106,row) ;
cout <<student.contact_no;
if ( row == 50 )
{
flag = 1 ;
row = 6 ;
gotoxy(1,65) ;
cout <<"Press any key to continue or Press <ESC> to exit" ;
ch = getch() ;
if (ch == 27)
break ;
clrscr() ;
gotoxy(33,2) ;
SetColor(11);
cout <<"LIST  OF  STUDENT   MEMBERS " ;
gotoxy(32,3) ;
cout <<"~~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
SetColor(13);
cout <<"MEMBER ID     ROLL NO          NAME            SEMESTER  DEPARTMENT              EMAIL
CONTACT  NO" ;
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~" ;
}
else
row = row + 2 ;
        }
}
if (!found)
{
gotoxy(5,10) ;
```

```
cout <<"\7Records not found" ;
}
if (!flag)
{
gotoxy(1,25) ;
cout <<"Press any key to continue..." ;
getche() ;
}file.close () ;
        }


        if(!strcmp(category,"STAFF"))
        {
            STAFF staff("STAFF");
            clrscr();
    int row = 6 , found=0, flag=0 ;char ch ;
gotoxy(33,2) ;
cout <<" LIST  OF  STAFF  MEMBERS  IN  THE  DEPARTMENT  OF " ;
gotoxy(32,3) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(1,4) ;
cout <<"MEMBER ID        NAME          QUALIFICATION   DESIGNATION   DEPARTMENT            EMAIL
CONTACT  NO" ;
gotoxy(1,5) ;
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~" ;
fstream file ;
file.open("STAFF.DAT", ios::in) ;
file.seekg(0,ios::beg);
while (file.read((char *) &staff, sizeof(STAFF)))
{
flag = 0 ;
delay(20) ;
if(!strcmp(dept,staff.department))
        {
            gotoxy(85,2);
            cout<<dept;
found = 1 ;
gotoxy(2,row) ;
cout <<staff.member_id;
gotoxy(12,row) ;
cout<<staff.member_name ;
gotoxy(32,row) ;
```

```
cout<<staff.qualification ;

gotoxy(49,row) ;

cout<<staff.designation ;

gotoxy(65,row) ;

cout<<staff.department ;

gotoxy(74,row) ;

cout<<staff.email_id ;

gotoxy(105,row) ;

cout <<staff.contact_no;

gotoxy(18,row+1) ;

        if ( row == 22 )

{

flag = 1 ;

row = 6 ;

gotoxy(1,60) ;

cout <<"Press any key to continue or Press <ESC> to exit" ;

ch = getch() ;

if (ch == 27)

break ;

clrscr() ;

gotoxy(33,2) ;

cout <<"LIST  OF  STUDENT   MEMBERS " ;

gotoxy(32,3) ;

cout <<"~~~~~~~~~~~~~~~" ;

gotoxy(1,4) ;

cout <<"MEMBER ID                   NAME             DEPARTMENT          DESIGNATION        EMAIL
CONTACT  NO" ;

gotoxy(1,5) ;

cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~" ;

}

else

row = row + 3 ;

        }

}

if (!found)

{

gotoxy(5,10) ;

cout <<"Records not found" ;}

if (!flag)

{

gotoxy(1,25) ;
```

```
cout <<"Press any key to continue..." ;
getche() ;}
file.close () ;
        }
}
```

```
    int STAFF::stfMemIdFound(char *s)
{
    STAFF stf;
   int  flag=0;
    char ch;
    ifstream ifs;
    ifs.open("STAFF.DAT",ios::in);
    while(ifs.read((char*)&stf,sizeof(STAFF)))
    {
        if(!strcmp(stf.member_id,s)){
            flag =1;
            break;
        }
    }
    if(flag==1){
        return 1;
    }
    else
    {return 0;
}
}
```

```
/*****************************************************************************************************
        This is the member function of CLASS   MEMBER used for adding a member(Student or Staff) record
*****************************************************************************************************
/
void MEMBER::add_members(){

    if(!strcmp(category,"STUDENT"))
```

```
        {
    STUDENT student("STUDENT");
char ch;
        ofstream ofs("STUDENT.DAT",ios::app|ios::binary);
        if(!ofs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
   do
    {
        clrscr();
        student.put_member_data();
                    cout<<"\nProcessing ";
           int x=12;
           while(x>0)
           {cout<<". ";
             delay(400);
             x--;
           }
        ofs.write((char*)&student,sizeof(STUDENT));
                cout<<"\nDo you wan to enter more records? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
    }while(ch=='y'||ch=='Y');
        }
        ofs.close();

}

    if(!strcmp(category,"STAFF"))
    {
    STAFF staff("STAFF");
    char ch;
        ofstream ofs("STAFF.DAT",ios::app|ios::binary);
        if(!ofs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
   do
    {
        clrscr();
        staff.put_member_data();
```

```
            cout<<"\nProcessing ";
            int x=12;
            while(x>0)
            {cout<<". ";
              delay(400);
              x--;
            }
        ofs.write((char*)&staff,sizeof(STAFF));
                cout<<"\nDo you wan to enter more records? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
    }while(ch=='y'||ch=='Y');
        }
        ofs.close();
}

}
```

```
 /**************************************************************************
   This member function of the CLASS   MEMBER is to SEARCH members by its ID
   **************************************************************************/
void MEMBER::search_membersById()
    {
        if(!strcmp(category,"STUDENT")){
        STUDENT student("STUDENT");
        int flag;
        char ch;
        char s[30];
        ifstream ifs("STUDENT.DAT",ios::binary);
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!\n";
        else{
   do{
       clrscr();
```

```
        flag=0;
        ifs.seekg(0,ios::beg);
    cout<<"\nEnter Member ID : ";
        //cin.ignore();
     cin.getline(s,30);
    while(ifs.read((char*)&student,sizeof(STUDENT)))
     {
        if(!strcmp(s,student.member_id))
        {cout<<"\n\t\t*****Member Record found****\n";
        //student.getStudentMemberDataByID();

           cout<<"\n\t\t*******MEMBER  DETAILS******\n"<<endl;

     cout<<"\t\t Student Member Id: "<<student.member_id<<endl;delay(500);
     cout<<"\t\tMember Category: "<<student.category<<endl;delay(500);
        cout<<"\t\tMember Name: "<<student.member_name<<endl;delay(500);
        cout<<"\t\tRoll Number: "<<student.roll_no<<endl;delay(500);
        cout<<"\t\t Sex: "<<student.sex<<endl;delay(500);
        cout<<"\t\tSemester: "<<student.semester<<endl;delay(500);
        cout<<"\t\tDepartment: "<<student.department<<endl;delay(500);
        cout<<"\t\tMember Address: "<<student.member_address<<endl;delay(500);
        cout<<"\t\t PinCode Number: "<<student.pincode<<endl;delay(500);
        cout<<"\t\t Contact Number: "<<student.contact_no<<endl;delay(500);
        cout<<"\t\tEmail Id: "<<student.email_id<<endl;delay(500);


            flag=1;
            cout<<"\nPress any key to continue*********\n";
            _getch();
            fflush(stdin);
            break;
        }
     }
        if(flag==0)
        {cout<<"\n!!!!!!!!!!!Member record is not FOUND!!!!!!!!!!!!\n";
            cout<<"\n****Please check the MEMBER ID you have entered and try again****\n";
        ifs.clear();
        }
           cout<<"\nDo you want to search more members? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
```

```
          break;
 }while(ch=='y'||ch=='Y');
      }
ifs.close();
}


  if(!strcmp(category,"STAFF")){
      STAFF staff("STAFF");
      int flag;
      char ch;
      char s[30];
      ifstream ifs("STAFF.DAT",ios::binary);
       if(!ifs)
          cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
      else{
do{
    clrscr();
      flag=0;
      ifs.seekg(0,ios::beg);
cout<<"\nEnter Member ID : ";
      //cin.ignore();
 cin.getline(s,30);
while(ifs.read((char*)&staff,sizeof(STAFF)))
 {
      if(!strcmp(s,staff.member_id))
      {cout<<"\n\t\t*****Mmber Record found****\n";
     // staff.getStaffMemberDataByID();
           cout<<"\n\t\t*******MEMBER  DETAILS******\n"<<endl;


 cout<<"\n\t\t Staff Member Id: "<<staff.member_id<<endl;delay(500);
 cout<<"\t\tMember Category: "<<staff.category<<endl;delay(500);
      cout<<"\t\tMember Name: "<<staff.member_name<<endl;delay(500);
      cout<<"\t\tSex: "<<staff.sex<<endl;delay(500);
       cout<<"\t\tDesignation: "<<staff.designation<<endl;delay(500);
      cout<<"\t\tQualification: "<<staff.qualification<<endl;delay(500);
      cout<<"\t\tDepartment: "<<staff.department<<endl;delay(500);
      cout<<"\t\tMember Address: "<<staff.member_address<<endl;delay(500);
      cout<<"\t\tPinCode Number: "<<staff.pincode<<endl;delay(500);
      cout<<"\t\tContact Number: "<<staff.contact_no<<endl;delay(500);
      cout<<"\t\tEmail Id: "<<staff.email_id<<endl;delay(500);
```

```
            flag=1;
            cout<<"\nPress any key to continue*********\n";
            _getch();
            fflush(stdin);
            break;
        }
    }
        if(flag==0)
        {cout<<"\n!!!!!!!!!!!Member record is not FOUND!!!!!!!!!!!!\n";
            cout<<"\n****Please check the MEMBER ID you have entered and try again****\n";
        ifs.clear();
        }
           cout<<"\nDo you want to search more books? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
    }while(ch=='y'||ch=='Y');
        }
   ifs.close();
   }
    }
```

```
/*************************************************************************************************************
******
This is the member function of CLASS   MEMBER used for searching a member(Student or Staff) record by its
Member ID
*************************************************************************************************************
******/
    int MEMBER::search_membersById(char *s)
    {
        if(!strcmp(category,"STUDENT")){
                STUDENT student("STUDENT");
        int flag=0;
        ifstream ifs("STUDENT.DAT",ios::binary);
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
        ifs.seekg(0,ios::beg);
   while(ifs.read((char*)&student,sizeof(STUDENT)))
```

```
    {
        if(!strcmp(s,student.member_id))
        {
         flag=1;
         break;
        }
    }
    ifs.close();
        }
            if(flag==1)
        return 1;
    else
        return 0;
        }


        if(!strcmp(category,"STAFF")){
                STAFF staff("STAFF");
        int flag=0;
        ifstream ifs("STAFF.DAT",ios::binary);
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
        ifs.seekg(0,ios::beg);
    while(ifs.read((char*)&staff,sizeof(STAFF)))
     {
        if(!strcmp(s,staff.member_id))
        {
         flag=1;
         break;
        }
     }
     ifs.close();
        }
            if(flag==1)
        return 1;
     else
        return 0;
        }
    }
```

```
 /*************************************************************************
     This member function of CLASS   MEMBER  is to SEARCH members by Email ID
*************************************************************************/
 void MEMBER::search_membersByEmail()
 {
        if(!strcmp(category,"STUDENT")){
     STUDENT student("STUDENT");
     int flag;
     char ch;
     char s[40],memID[30];
     ifstream ifs("STUDENT.DAT",ios::binary);
      if(!ifs)
         cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!\n";
     else{
do{
    clrscr();
     flag=0;
     ifs.seekg(0,ios::beg);
cout<<"\nEnter Email ID : ";
 cin.getline(s,40);
while(ifs.read((char*)&student,sizeof(STUDENT)))
 {
     if(!strcmp(s,student.email_id))
     {cout<<"\n\t\t*****Member Record found****\n";
     cout<<"\n\t\t*******MEMBER  DETAILS******\n"<<endl;
     strcpy(memID,student.member_id);
     cout<<"\t\tStudent Member Id: "<<student.member_id<<endl;delay(500);
     cout<<"\t\tMember Category: "<<student.category<<endl;delay(500);
     cout<<"\t\tMember Name: "<<student.member_name<<endl;delay(500);
     cout<<"\t\tRoll Number: "<<student.roll_no<<endl;delay(500);
     cout<<"\t\tSex: "<<student.sex<<endl;delay(500);
     cout<<"\t\tSemester: "<<student.semester<<endl;delay(500);
     cout<<"\t\tDepartment: "<<student.department<<endl;delay(500);
     cout<<"\t\tMember Address: "<<student.member_address<<endl;delay(500);
     cout<<"\t\tPinCode Number: "<<student.pincode<<endl;delay(500);
     cout<<"\t\tContact Number: "<<student.contact_no<<endl;delay(500);
     cout<<"\t\tEmail Id: "<<student.email_id<<endl;delay(500);
         flag=1;
         cout<<"\nPress any key to continue*********\n";
```

```
            cout<<"                          or                              ";
            cout<<"\nPress Enter to see the list of books issued to this member";
            char ch;
            ch=_getch();
            if(ch==13)
            {
                ISSUE isu;
                clrscr();
                isu.list_of_books_issued_to(memID);
            }
            else
            break;
        }
     }
        if(flag==0)
        {cout<<"\n!!!!!!!!!!!Member record is not FOUND!!!!!!!!!!!!\n";
            cout<<"\n****Please check the Email ID you have entered and try again****\n";
        ifs.clear();
        }
           cout<<"\nDo you want to search more Members? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
     }while(ch=='y'||ch=='Y');
        }
    ifs.close();
    }


        if(!strcmp(category,"STAFF")){
        STAFF staff("STAFF");
        int flag;
        char ch;
        char s[40],memID[30];
        ifstream ifs("STAFF.DAT",ios::binary);
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
    do{
        clrscr();
         flag=0;
         ifs.seekg(0,ios::beg);
```

```
cout<<"\nEnter Email ID : ";
     //cin.ignore();
 cin.getline(s,40);
while(ifs.read((char*)&staff,sizeof(STAFF)))
 {
     if(!strcmp(s,staff.email_id))
     {cout<<"\n\t\t*****Member Record found****\n";
         //staff.getStaffMemberDataByID();
            cout<<"\n\t\t*******MEMBER  DETAILS******\n"<<endl;
             strcpy(memID,staff.member_id);
     cout<<"\t\t Staff Member Id: "<<staff.member_id<<endl;delay(500);
     cout<<"\t\tMember Category: "<<staff.category<<endl;delay(500);
     cout<<"\t\tMember Name: "<<staff.member_name<<endl;delay(500);
     cout<<"\t\tSex: "<<staff.sex<<endl;delay(500);
     cout<<"\t\tDesignation: "<<staff.designation<<endl;delay(500);
     cout<<"\t\tQualification: "<<staff.qualification<<endl;delay(500);
     cout<<"\t\tDepartment: "<<staff.department<<endl;delay(500);
     cout<<"\t\tMember Address: "<<staff.member_address<<endl;delay(500);
     cout<<"\t\tPinCode Number: "<<staff.pincode<<endl;delay(500);
     cout<<"\t\tContact Number: "<<staff.contact_no<<endl;delay(500);
     cout<<"\t\tEmail Id: "<<staff.email_id<<endl;delay(500);
         flag=1;
         cout<<"\nPress any key to continue*********\n";
         cout<<"                          or                              ";
         cout<<"\nPress Enter to see the list of books issued to this member";
         char ch;
         ch=_getch();
         if(ch==13)
         {
             ISSUE isu;
             clrscr();
             isu.list_of_books_issued_to(memID);
         }
         else
         break;
     }
 }
     if(flag==0)
     {cout<<"\n!!!!!!!!!!!Member record is not FOUND!!!!!!!!!!!!\n";
         cout<<"\n****Please check the Email ID you have entered and try again****\n";
     ifs.clear();
```

```
        }
           cout<<"\nDo you want to search more books? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
    }while(ch=='y'||ch=='Y');
        }
   ifs.close();
   }
}
```

```
/*************************************************************************
   This member function of   MEMBER  CLASS is to SEARCH members by DEPARTMENT
   *************************************************************************/
void MEMBER::search_membersByDepartment(void)
    {
           if(!strcmp(category,"STUDENT")){
        STUDENT student("STUDENT");
        STUDENT student2("STUDENT");
        int flag;
        char ch;
        char s[40];
   do{
        if(flag==1)
       clrscr();
        flag=0;
   cout<<"\nEnter Department Name: ";
    cin.getline(s,20);
    ifstream ifs("STUDENT.DAT",ios::binary);
   while(ifs.read((char*)&student,sizeof(STUDENT)))
    {
        if(!strcmp(s,student.department))
            {
                flag=1;
            break;
            }
    }
       ifs.close();
```

```
    if(flag==1)
     {
         student2.lists_byDepartment(s);
             }
     if(flag==0)
     {
         cout<<"\n****Please check the DEPARTMENT NAME you have entered and try again****\n";
     }
        cout<<"\nDo you want to search more ? (y/n)";
     ch=_getch();
     if(ch=='n' || ch=='N')
         break;
 }while(ch=='y'||ch=='Y');
}


        if(!strcmp(category,"STAFF")){
     STAFF staff("STAFF");
      STAFF staff2("STAFF");
     int flag;
     char ch;
     char s[40];
do{
     if(flag==1)
    clrscr();
     flag=0;
cout<<"\nEnter Department Name: ";
 cin>>s;
     ifstream ifs("STAFF.DAT",ios::binary);
while(ifs.read((char*)&staff,sizeof(STAFF)))
 {
     if(!strcmp(s,staff.department))
         {
             flag=1;
         break;
         }
 }
    ifs.close();
    if(flag==1)
     {
         staff2.lists_byDepartment(s);
     }
```

```
        if(flag==0)
        {
            cout<<"\n****Please check the DEPARTMENT NAME you have entered and try again****\n";
        }
           cout<<"\nDo you want to search more ? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
    }while(ch=='y'||ch=='Y');

}

    }
```

```
   /*********************************************************
   This member function is to DELETE member record by its ID
   *********************************************************/
  void MEMBER::delete_member_records()
   {
       int flag=0,flag2=0;
       clrscr();
       if(!strcmp(category,"STUDENT")){
       STUDENT student("STUDENT");
       STUDENT student2("STUDENT");
       char s[30],ch;
       ifstream ifs;

                do{
                        cout<<"\nEnter Member ID: ";
                        cin.getline(s,30);

                                //student.getStudentMemberDataByID();
ifstream ifs2("STUDENT.DAT");
while(ifs2.read((char*)&student,sizeof(STUDENT)))
{
    if(!strcmp(s,student.member_id))
        {
            flag=1;
```

```
        break;
        }
}
ifs2.close();
if(flag==1){
               cout<<"\n*******MEMBER  DETAILS******\n"<<endl;

        cout<<"\n\t\t Student Member Id: "<<student.member_id<<endl;delay(500);
        cout<<"\t\tMember Category: "<<student.category<<endl;delay(500);
        cout<<"\t\tMember Name: "<<student.member_name<<endl;delay(500);
        cout<<"\t\tRoll Number: "<<student.roll_no<<endl;delay(500);
        cout<<"\t\t Sex: "<<student.sex<<endl;delay(500);
        cout<<"\t\tSemester: "<<student.semester<<endl;delay(500);
        cout<<"\t\tDepartment: "<<student.department<<endl;delay(500);
        cout<<"\t\tMember Address: "<<student.member_address<<endl;delay(500);
        cout<<"\t\t PinCode Number: "<<student.pincode<<endl;delay(500);
        cout<<"\t\t Contact Number: "<<student.contact_no<<endl;delay(500);
        cout<<"\t\tEmail Id: "<<student.email_id<<endl;delay(500);


               cout<<"\nDo you want to DELETE this record from file? (y/n)";
        ch=_getch();
        if(ch=='y' || ch=='Y')
           {
               cout<<"\nProcessing ";
           int x=12;
           while(x>0)
           {cout<<". ";
             delay(400);
             x--;
           }
                   ifs.open("STUDENT.DAT",ios::binary|ios::in);
                    ofstream ofs;
                  ofs.open("temp.DAT",ios::binary|ios::out);
            while(ifs.read((char*)&student2,sizeof(STUDENT)))
            {
                    if(strcmp(s,student2.member_id))
                    {
                        ofs.write((char*)&student2,sizeof(STUDENT));
                    }
                  //   ifs.read((char*)&student2,sizeof(STUDENT));
```

```
            }
            ifs.close();
            ofs.close();
            remove("STUDENT.DAT");
            rename("temp.DAT","STUDENT.DAT");
            cout<<"\nThe MEMBER Record has successfully deleted from the Database"<<endl;
           }
                        }
            else{
                cout<<"\nMember Record Not Found\n";
            }
            cout<<"\nPress any key to continue*********\n";
        _getch();
        fflush(stdin);
           cout<<"\nDo you want to DELETE more records? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
                }while(ch=='y'||ch=='Y');
        }



         if(!strcmp(category,"STAFF")){
       STAFF staff("STAFF");
       STAFF staff2("STAFF");

       char s[30],ch;
                do{
                        cout<<"\nEnter Member ID: ";
                        cin.getline(s,30);

ifstream ifs2("STAFF.DAT",ios::binary|ios::in);
while(ifs2.read((char*)&staff,sizeof(STAFF)))
{
    if(!strcmp(s,staff.member_id))
        {break;flag2=1;}
}
ifs2.close();
        if(flag2==1)
            {
                cout<<"\t\t*******MEMBER  DETAILS******"<<endl;
```

```
     cout<<"\t\tStaff Member Id: "<<staff.member_id<<endl;delay(500);
     cout<<"\t\tMember Category: "<<staff.category<<endl;delay(500);
     cout<<"\t\tMember Name: "<<staff.member_name<<endl;delay(500);
     cout<<"\t\tSex: "<<staff.sex<<endl;delay(500);
     cout<<"\t\tDesignation: "<<staff.designation<<endl;delay(500);
     cout<<"\t\tQualification: "<<staff.qualification<<endl;delay(500);
     cout<<"\t\tDepartment: "<<staff.department<<endl;delay(500);
     cout<<"\t\tMember Address: "<<staff.member_address<<endl;delay(500);
     cout<<"\t\tPinCode Number: "<<staff.pincode<<endl;delay(500);
     cout<<"\t\tContact Number: "<<staff.contact_no<<endl;delay(500);
     cout<<"\t\tEmail Id: "<<staff.email_id<<endl;delay(500);


                    //  staff.getStaffMemberDataByID();
     cout<<"\nDo you want to DELETE this record from file? (y/n)";
 ch=_getch();
 if(ch=='y' || ch=='Y')
{
 cout<<"\nProcessing ";
int x=12;
while(x>0)
{cout<<". ";
  delay(400);
  x--;
}
        ifstream ifs("STAFF.DAT",ios::binary|ios::in);
         ofstream ofs;
      ofs.open("temp.DAT",ios::binary|ios::out);


 while(ifs.read((char*)&staff2,sizeof(STAFF)))
 {
         if(strcmp(s,staff2.member_id))
         {
             ofs.write((char*)&staff2,sizeof(STAFF));
         }
 }
 ifs.close();
 ofs.close();
 remove("STAFF.DAT");
```

```
            rename("temp.DAT","STAFF.DAT");
            cout<<"\nThe MEMBER Record has successfully deleted from the Database"<<endl;
           }
                       }
            else{
                cout<<"\nMember Record Not Found\n";
            }
            cout<<"\nPress any key to continue*********\n";
      _getch();
      fflush(stdin);
         cout<<"\nDo you want to DELETE more records? (y/n)";
      ch=_getch();
      if(ch=='n' || ch=='N')
          break;
              }while(ch=='y'||ch=='Y');
      }

}
char* MEMBER::get_memberName(char *s)
 {
      if(!strcmp(category,"STUDENT")){

      MEMBER member;
      STUDENT student("STUDENT");
       char mname[40];
      int flag=0;
      ifstream ifs("STUDENT.DAT",ios::binary);
       if(!ifs)
          cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
      else{
      ifs.seekg(0,ios::beg);
while(ifs.read((char*)&student,sizeof(STUDENT)))
 {
      if(!strcmp(s,student.member_id))
      {
       flag=1;
       break;
      }
 }
 ifs.close();
      }
```

```
            if(flag==1)
        return student.member_name;
    else
        return " Student Member Record Not found";
        }

        if(!strcmp(category,"STAFF")){
                STAFF staff("STAFF");
                MEMBER member;
        int flag=0;
         char mname[40];
        ifstream ifs("STAFF.DAT",ios::binary);
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
        ifs.seekg(0,ios::beg);
   while(ifs.read((char*)&staff,sizeof(STAFF)))
    {
        if(!strcmp(s,staff.member_id))
        {
            strcpy(mname,staff.member_name);
         flag=1;
         break;
        }
    }
    ifs.close();
        }
            if(flag==1)
        return mname;
    else
        return "Staff Member Record Not found";
    }
   }


       char* MEMBER::get_memberEmail(char*s)
       {
        if(!strcmp(category,"STUDENT")){
                STUDENT student("STUDENT");
        int flag=0;
        ifstream ifs("STUDENT.DAT",ios::binary);
```

```
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
        ifs.seekg(0,ios::beg);
   while(ifs.read((char*)&student,sizeof(STUDENT)))
    {
        if(!strcmp(s,student.member_id))
        {
         flag=1;
         break;
        }
    }
    ifs.close();
        }
            if(flag==1)
        return student.email_id;
    else
        return "Member Record Not found";
        }


        if(!strcmp(category,"STAFF")){
                STAFF staff("STAFF");
        int flag=0;
        ifstream ifs("STAFF.DAT",ios::binary);
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
        ifs.seekg(0,ios::beg);
   while(ifs.read((char*)&staff,sizeof(STAFF)))
    {
        if(!strcmp(s,staff.member_id))
        {
         flag=1;
         break;
        }
    }
    ifs.close();
        }
            if(flag==1)
        return staff.email_id;
    else
```

```
        return "Member Record Not found";
        }
    }

     /*************************************************************
     This member function will return the member's contact number
     *************************************************************/

        char* MEMBER::get_memberContactNo(char*s)
        {
         if(!strcmp(category,"STUDENT")){
                 STUDENT student("STUDENT");
         int flag=0;
         ifstream ifs("STUDENT.DAT",ios::binary);
          if(!ifs)
             cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
         else{
         ifs.seekg(0,ios::beg);
    while(ifs.read((char*)&student,sizeof(STUDENT)))
     {
         if(!strcmp(s,student.member_id))
         {
          flag=1;
          break;
         }
     }
     ifs.close();
         }
             if(flag==1)
         return student.contact_no;
     else
         return "Member Record Not found";
         }

         if(!strcmp(category,"STAFF")){
                 STAFF staff("STAFF");
         int flag=0;
         ifstream ifs("STAFF.DAT",ios::binary);
          if(!ifs)
             cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
         else{
```

```
        ifs.seekg(0,ios::beg);
   while(ifs.read((char*)&staff,sizeof(STAFF)))
    {
        if(!strcmp(s,staff.member_id))
        {
         flag=1;
         break;
        }
    }
    ifs.close();
        }
            if(flag==1)
        return staff.contact_no;
    else
        return "Member Record Not found";
        }
   }



/************************************************************************
  This member function is used to modify name for specified member_id
************************************************************************/


   void MEMBER::modify_name(char *MemberId)
   {
       clrscr();
        if(!strcmp(category,"STUDENT")){
       STUDENT student("STUDENT");
       STUDENT student2;
       char ch,ch2;
       //do{
       if(student.search_membersById(MemberId))
       {
                   cout<<"\n******Member record is found******";


      ifstream ifs;
      ifs.open("STUDENT.DAT",ios::binary);


       ifs.seekg(0,ios::beg);
   while(ifs.read((char*)&student,sizeof(STUDENT)))
    {
```

```
        if(!strcmp(MemberId,student.member_id))
        {

           cout<<"\n*******MEMBER  DETAILS******\n"<<endl;

        cout<<"\n\t\t Student Member Id: "<<student.member_id<<endl;delay(500);
        cout<<"\t\tMember Category: "<<student.category<<endl;delay(500);
        cout<<"\t\tMember Name: "<<student.member_name<<endl;delay(500);
        cout<<"\t\tRoll Number: "<<student.roll_no<<endl;delay(500);
        cout<<"\t\t Sex: "<<student.sex<<endl;delay(500);
        cout<<"\t\tSemester: "<<student.semester<<endl;delay(500);
        cout<<"\t\tDepartment: "<<student.department<<endl;delay(500);
        cout<<"\t\tMember Address: "<<student.member_address<<endl;delay(500);
        cout<<"\t\t PinCode Number: "<<student.pincode<<endl;delay(500);
        cout<<"\t\t Contact Number: "<<student.contact_no<<endl;delay(500);
        cout<<"\t\tEmail Id: "<<student.email_id<<endl;delay(500);

            cout<<"\nPress any key to continue*********\n";
            _getch();
            fflush(stdin);
            break;
        }
    }
    ifs.close();
                        cout<<"\nDo you realy want to update this record? (y/n)";
                  ch=_getch();
                  if(ch=='y'||ch=='Y')
                  {
                      cout<<"\nEnter New  Student Members' name: ";
                      cin.getline(student.member_name,40);
                     cout<<"\nPress any key to continue****";
                      _getch();
                                  cout<<"\nProcessing ";
           int x=12;
           while(x>0)
           {cout<<". ";
             delay(400);
             x--;
           }
                      fstream fs("STUDENT.DAT",ios::in|ios::out|ios::binary|ios::ate);
                      fs.seekg(0);
```

```
                    while(fs.read((char*)&student2,sizeof(STUDENT)))
                    {
                        if(!strcmp(MemberId,student2.member_id))
                        {
                            fs.seekp(fs.tellp()-sizeof(STUDENT));
                            fs.write((char*)&student,sizeof(STUDENT));
                        }
                    }
                    fs.close();
                    cout<<"\n*****Member's name  is successfuly modified*****";
                }
    }
    else
    {cout<<"\n!!!!!MEMBER record not found!!!!!";}

}


     if(!strcmp(category,"STAFF")){
    STAFF staff("STAFF");
    STAFF staff2;
    char ch,ch2;
    //do{
    if(staff.search_membersById(MemberId))
    {
                cout<<"\n******Member record is found******";

   ifstream ifs;
   ifs.open("STAFF.DAT",ios::binary);
     ifs.seekg(0,ios::beg);
while(ifs.read((char*)&staff,sizeof(STAFF)))
 {
     if(!strcmp(MemberId,staff.member_id))
     {

         cout<<"\t\t*******MEMBER  DETAILS******\n"<<endl;

      cout<<"\n\t\t Staff Member Id: "<<staff.member_id<<endl;delay(500);
      cout<<"\t\tMember Category: "<<staff.category<<endl;delay(500);
      cout<<"\t\tMember Name: "<<staff.member_name<<endl;delay(500);
      cout<<"\t\tSex: "<<staff.sex<<endl;delay(500);
```

```
        cout<<"\t\tDesignation: "<<staff.designation<<endl;delay(500);
        cout<<"\t\tQualification: "<<staff.qualification<<endl;delay(500);
        cout<<"\t\tDepartment: "<<staff.department<<endl;delay(500);
        cout<<"\t\tMember Address: "<<staff.member_address<<endl;delay(500);
        cout<<"\t\tPinCode Number: "<<staff.pincode<<endl;delay(500);
        cout<<"\t\tContact Number: "<<staff.contact_no<<endl;delay(500);
        cout<<"\t\tEmail Id: "<<staff.email_id<<endl;delay(500);



            cout<<"\nPress any key to continue*********\n";
            _getch();
            fflush(stdin);
            break;
        }
    }
    ifs.close();
                        cout<<"\nDo you realy want to update this record? (y/n)";
                   ch=_getch();
                   if(ch=='y'||ch=='Y')
                   {
                        cout<<"\nEnter New  Staff Members' name: ";
                        cin.getline(staff.member_name,40);
                       cout<<"\nPress any key to continue****";
                        _getch();
            cout<<"\nProcessing ";
           int x=12;
           while(x>0)
           {cout<<". ";
             delay(400);
             x--;
           }
                        fstream fs("STAFF.DAT",ios::in|ios::out|ios::binary|ios::ate);
                        fs.seekg(0);
                        while(fs.read((char*)&staff2,sizeof(STAFF)))
                        {
                            if(!strcmp(MemberId,staff2.member_id))
                            {
                                fs.seekp(fs.tellp()-sizeof(STAFF));
                                fs.write((char*)&staff,sizeof(STAFF));
                            }
                        }
```

```
                        fs.close();
                        cout<<"\n*****Member's name  is successfuly modified*****";
                    }
        }
        else
        {
            cout<<"\n!!!!!MEMBER record not found!!!!!";}

    }

    }
```

```
/*************************************************************************
  This member function is used to modify email ID for specified member_id
*************************************************************************/

    void MEMBER::modify_email(char *MemberId)
    {
        clrscr();
         if(!strcmp(category,"STUDENT")){
        STUDENT student("STUDENT");
        STUDENT student2;
        char ch,ch2;
        if(student.search_membersById(MemberId))
        {
         cout<<"\n******Member record is found******";


       ifstream ifs;
       ifs.open("STUDENT.DAT",ios::binary);
        ifs.seekg(0,ios::beg);
    while(ifs.read((char*)&student,sizeof(STUDENT)))
     {
         if(!strcmp(MemberId,student.member_id))
         {
```

```
        cout<<"\n*******MEMBER  DETAILS******\n"<<endl;


    cout<<"\n\t\t Student Member Id: "<<student.member_id<<endl;delay(500);
    cout<<"\t\tMember Category: "<<student.category<<endl;delay(500);
    cout<<"\t\tMember Name: "<<student.member_name<<endl;delay(500);
    cout<<"\t\tRoll Number: "<<student.roll_no<<endl;delay(500);
    cout<<"\t\t Sex: "<<student.sex<<endl;delay(500);
    cout<<"\t\tSemester: "<<student.semester<<endl;delay(500);
    cout<<"\t\tDepartment: "<<student.department<<endl;delay(500);
    cout<<"\t\tMember Address: "<<student.member_address<<endl;delay(500);
    cout<<"\t\t PinCode Number: "<<student.pincode<<endl;delay(500);
    cout<<"\t\t Contact Number: "<<student.contact_no<<endl;delay(500);
    cout<<"\t\tEmail Id: "<<student.email_id<<endl;delay(500);


        cout<<"\nPress any key to continue*********\n";
        _getch();
        fflush(stdin);
        break;
    }
}
ifs.close();
                    cout<<"\nDo you realy want to update this record? (y/n)";
               ch=_getch();
               if(ch=='y'||ch=='Y')
               {
                    cout<<"\nEnter New  Student Member's name: ";
                    cin.getline(student.email_id,40);
                   cout<<"\nPress any key to continue****";
                    _getch();
                       cout<<"\nProcessing ";
        int x=12;
        while(x>0)
        {cout<<". ";
          delay(400);
          x--;
        }
        fstream fs("STUDENT.DAT",ios::in|ios::out|ios::binary|ios::ate);
                    fs.seekg(0);
                    while(fs.read((char*)&student2,sizeof(STUDENT)))
                    {
                        if(!strcmp(MemberId,student2.member_id))
```

```
                        {
                            fs.seekp(fs.tellp()-sizeof(STUDENT));
                            fs.write((char*)&student,sizeof(STUDENT));
                        }
                    }
                    fs.close();
                    cout<<"\n*****Member's name  is successfuly modified*****";
                }
    }

    else
    {
        cout<<"\n!!!!!MEMBER record not found!!!!!";}

}


     if(!strcmp(category,"STAFF")){
    STAFF staff("STAFF");
    STAFF staff2;
    char ch,ch2;
    //do{
    if(staff.search_membersById(MemberId))
    {
                cout<<"\n******Member record is found******";


   ifstream ifs;
   ifs.open("STAFF.DAT",ios::binary);
    //clrscr();
     ifs.seekg(0,ios::beg);
while(ifs.read((char*)&staff,sizeof(STAFF)))
 {
     if(!strcmp(MemberId,staff.member_id))
     {//cout<<"\n*****Book Entry found****\n";
 //    staff.getStaffMemberDataByID();

       cout<<"\t\t*******MEMBER  DETAILS******\n"<<endl;

 cout<<"\n\t\t Staff Member Id: "<<staff.member_id<<endl;delay(500);
 cout<<"\t\tMember Category: "<<staff.category<<endl;delay(500);
```

```
        cout<<"\t\tMember Name: "<<staff.member_name<<endl;delay(500);
        cout<<"\t\tSex: "<<staff.sex<<endl;delay(500);
         cout<<"\t\tDesignation: "<<staff.designation<<endl;delay(500);
        cout<<"\t\tQualification: "<<staff.qualification<<endl;delay(500);
        cout<<"\t\tDepartment: "<<staff.department<<endl;delay(500);
        cout<<"\t\tMember Address: "<<staff.member_address<<endl;delay(500);
        cout<<"\t\tPinCode Number: "<<staff.pincode<<endl;delay(500);
        cout<<"\t\tContact Number: "<<staff.contact_no<<endl;delay(500);
        cout<<"\t\tEmail Id: "<<staff.email_id<<endl;delay(500);


            cout<<"\nPress any key to continue*********\n";
            _getch();
            fflush(stdin);
            break;
        }
    }
    ifs.close();
                        cout<<"\nDo you realy want to update this record? (y/n)";
                  ch=_getch();
                  if(ch=='y'||ch=='Y')
                  {
                      cout<<"\nEnter New  Staff Members' name: ";
                      cin.getline(staff.email_id,40);
                     cout<<"\nPress any key to continue****";
                      _getch();
   cout<<"\nProcessing ";
           int x=12;
           while(x>0)
           {cout<<". ";
             delay(400);
             x--;
           }
                      fstream fs("STAFF.DAT",ios::in|ios::out|ios::binary|ios::ate);
                      fs.seekg(0);
                      while(fs.read((char*)&staff2,sizeof(STAFF)))
                      {
                          if(!strcmp(MemberId,staff2.member_id))
                          {
                              fs.seekp(fs.tellp()-sizeof(STAFF));
                              fs.write((char*)&staff,sizeof(STAFF));
```

```
                        }
                    }
                    fs.close();
                    cout<<"\n*****Member's name  is successfuly modified*****";
                }
        }
        else
        {
            cout<<"\n!!!!!MEMBER record not found!!!!!";}

    }

    }

    /************************************************************************
  This member function is used to modify contact number for specified member_id
************************************************************************/

    void MEMBER::modify_cntctNo(char *MemberId)
    {
        clrscr();
         if(!strcmp(category,"STUDENT")){
        STUDENT student("STUDENT");
        STUDENT student2;
        char ch,ch2;

        if(student.search_membersById(MemberId))
        {
                    cout<<"\n\t******Member record is found******\n"<<endl;

       ifstream ifs;
       ifs.open("STUDENT.DAT",ios::binary);
        ifs.seekg(0,ios::beg);
    while(ifs.read((char*)&student,sizeof(STUDENT)))
     {
```

```
        if(!strcmp(MemberId,student.member_id))
        {

           cout<<"\n*******MEMBER  DETAILS******";

        cout<<"\n\t\t Student Member Id: "<<student.member_id<<endl;delay(500);
        cout<<"\t\tMember Category: "<<student.category<<endl;delay(500);
        cout<<"\t\tMember Name: "<<student.member_name<<endl;delay(500);
        cout<<"\t\tRoll Number: "<<student.roll_no<<endl;delay(500);
        cout<<"\t\t Sex: "<<student.sex<<endl;delay(500);
        cout<<"\t\tSemester: "<<student.semester<<endl;delay(500);
        cout<<"\t\tDepartment: "<<student.department<<endl;delay(500);
        cout<<"\t\tMember Address: "<<student.member_address<<endl;delay(500);
        cout<<"\t\t PinCode Number: "<<student.pincode<<endl;delay(500);
        cout<<"\t\t Contact Number: "<<student.contact_no<<endl;delay(500);
        cout<<"\t\tEmail Id: "<<student.email_id<<endl;delay(500);

            cout<<"\nPress any key to continue*********\n";
            _getch();
            fflush(stdin);
            break;
        }
    }
    ifs.close();
                        cout<<"\nDo you realy want to update this record? (y/n)";
                  ch=_getch();
                  if(ch=='y'||ch=='Y')
                  {
                      cout<<"\nEnter New contact no the Student Member: ";
                      cin.getline(student.contact_no,15);
                     cout<<"\nPress any key to continue****";
                      _getch();
    cout<<"\nProcessing ";
           int x=12;
           while(x>0)
           {cout<<". ";
             delay(400);
             x--;
           }
                      fstream fs("STUDENT.DAT",ios::in|ios::out|ios::binary|ios::ate);
                      fs.seekg(0);
```

```
                    while(fs.read((char*)&student2,sizeof(STUDENT)))
                    {
                        if(!strcmp(MemberId,student2.member_id))
                        {
                            fs.seekp(fs.tellp()-sizeof(STUDENT));
                            fs.write((char*)&student,sizeof(STUDENT));
                        }
                    }
                    fs.close();
                    cout<<"\n*****Member's name  is successfuly modified*****";
                }
    }
    else
    {
        cout<<"\n!!!!!MEMBER record not found!!!!!";}

}


     if(!strcmp(category,"STAFF")){
    STAFF staff("STAFF");
    STAFF staff2;
    char ch,ch2;

    if(staff.search_membersById(MemberId))
    {
                cout<<"\n******Member record is found******\n\n";

   ifstream ifs;
   ifs.open("STAFF.DAT",ios::binary);
     ifs.seekg(0,ios::beg);
while(ifs.read((char*)&staff,sizeof(STAFF)))
 {
     if(!strcmp(MemberId,staff.member_id))
     {

               cout<<"\t\t*******MEMBER  DETAILS******"<<endl;

     cout<<"\n\t\t Staff Member Id: "<<staff.member_id<<endl;delay(500);
     cout<<"\t\tMember Category: "<<staff.category<<endl;delay(500);
     cout<<"\t\tMember Name: "<<staff.member_name<<endl;delay(500);
```

```
        cout<<"\t\tSex: "<<staff.sex<<endl;delay(500);
        cout<<"\t\tDesignation: "<<staff.designation<<endl;delay(500);
        cout<<"\t\tQualification: "<<staff.qualification<<endl;delay(500);
        cout<<"\t\tDepartment: "<<staff.department<<endl;delay(500);
        cout<<"\t\tMember Address: "<<staff.member_address<<endl;delay(500);
        cout<<"\t\tPinCode Number: "<<staff.pincode<<endl;delay(500);
        cout<<"\t\tContact Number: "<<staff.contact_no<<endl;delay(500);
        cout<<"\t\tEmail Id: "<<staff.email_id<<endl;delay(500);


            cout<<"\nPress any key to continue*********\n";
            _getch();
            fflush(stdin);
            break;
        }
    }
    ifs.close();
                        cout<<"\nDo you realy want to update this record? (y/n)";
                  ch=_getch();
                  if(ch=='y'||ch=='Y')
                  {
                      cout<<"\nEnter New contact no the Staff Member: ";
                      cin.getline(staff.contact_no,15);
                     cout<<"\nPress any key to continue****";
                      _getch();
   cout<<"\nProcessing ";
           int x=12;
           while(x>0)
           {cout<<". ";
             delay(400);
             x--;
           }
                      fstream fs("STAFF.DAT",ios::in|ios::out|ios::binary|ios::ate);
                      fs.seekg(0);
                      while(fs.read((char*)&staff2,sizeof(STAFF)))
                      {
                          if(!strcmp(MemberId,staff2.member_id))
                          {
                              fs.seekp(fs.tellp()-sizeof(STAFF));
                              fs.write((char*)&staff,sizeof(STAFF));
                          }
                      }
```

```
                    fs.close();
                    cout<<"\n*****Member's name  is successfuly modified*****";
                }
    }
    else
    {
        cout<<"\n!!!!!MEMBER record not found!!!!!";}
}

}



void MEMBER::modify_address(char *MemberId)
{
    clrscr();
     if(!strcmp(category,"STUDENT")){
    STUDENT student("STUDENT");
    STUDENT student2("STUDENT");
    char ch,ch2;
    if(student.search_membersById(MemberId))
    {
                cout<<"\n******Member record is found******";

   ifstream ifs;
   ifs.open("STUDENT.DAT",ios::binary);
   ifs.seekg(0,ios::beg);
while(ifs.read((char*)&student,sizeof(STUDENT)))
 {
     if(!strcmp(MemberId,student.member_id))
     {



        cout<<"\n*******MEMBER  DETAILS******\n"<<endl;

 cout<<"\t\tStudent Member Id: "<<student.member_id<<endl;delay(500);
 cout<<"\t\tMember Category: "<<student.category<<endl;delay(500);
     cout<<"\t\tMember Name: "<<student.member_name<<endl;delay(500);
     cout<<"\t\tRoll Number: "<<student.roll_no<<endl;delay(500);
     cout<<"\t\t Sex: "<<student.sex<<endl;delay(500);
```

```
        cout<<"\t\tSemester: "<<student.semester<<endl;delay(500);
        cout<<"\t\tDepartment: "<<student.department<<endl;delay(500);
        cout<<"\t\tMember Address: "<<student.member_address<<endl;delay(500);
        cout<<"\t\t PinCode Number: "<<student.pincode<<endl;delay(500);
        cout<<"\t\t Contact Number: "<<student.contact_no<<endl;delay(500);
        cout<<"\t\tEmail Id: "<<student.email_id<<endl;delay(500);


            cout<<"\nPress any key to continue*********\n";
            _getch();
            fflush(stdin);
            break;
        }
    }
    ifs.close();
                        cout<<"\nDo you realy want to update this record? (y/n)";
                   ch=_getch();
                   if(ch=='y'||ch=='Y')
                   {
                        cout<<"\nEnter New Address of the Student Member: ";
                        cin.getline(student.member_address,200);
                       cout<<"\nPress any key to continue****";
                        _getch();
    cout<<"\nProcessing ";
            int x=12;
            while(x>0)
            {cout<<". ";
              delay(400);
              x--;
            }
                        fstream fs("STUDENT.DAT",ios::in|ios::out|ios::binary|ios::ate);
                        fs.seekg(0);
                        while(fs.read((char*)&student2,sizeof(STUDENT)))
                        {
                            if(!strcmp(MemberId,student2.member_id))
                            {
                                fs.seekp(fs.tellp()-sizeof(STUDENT));
                                fs.write((char*)&student,sizeof(STUDENT));
                            }
                        }
                        fs.close();
                        cout<<"\n*****Member's name  is successfuly modified*****";
```

```
                }
    }
    else
    {cout<<"\n!!!!!MEMBER record not found!!!!!";}

}



     if(!strcmp(category,"STAFF")){
    STAFF staff("STAFF");
    STAFF staff2;
    char ch,ch2;
    if(staff.search_membersById(MemberId))
    {
     cout<<"\n******Member record is found******";
   ifstream ifs;
   ifs.open("STAFF.DAT",ios::binary);
     ifs.seekg(0,ios::beg);
while(ifs.read((char*)&staff,sizeof(STAFF)))
 {
     if(!strcmp(MemberId,staff.member_id))
     {
cout<<"\t\t*******MEMBER  DETAILS******\n"<<endl;


     cout<<"\t\tStaff Member Id: "<<staff.member_id<<endl;delay(500);
     cout<<"\t\tMember Category: "<<staff.category<<endl;delay(500);
     cout<<"\t\tMember Name: "<<staff.member_name<<endl;delay(500);
     cout<<"\t\tSex: "<<staff.sex<<endl;delay(500);
     cout<<"\t\tDesignation: "<<staff.designation<<endl;delay(500);
     cout<<"\t\tQualification: "<<staff.qualification<<endl;delay(500);
     cout<<"\t\tDepartment: "<<staff.department<<endl;delay(500);
     cout<<"\t\tMember Address: "<<staff.member_address<<endl;delay(500);
     cout<<"\t\tPinCode Number: "<<staff.pincode<<endl;delay(500);
     cout<<"\t\tContact Number: "<<staff.contact_no<<endl;delay(500);
     cout<<"\t\tEmail Id: "<<staff.email_id<<endl;delay(500);
     cout<<"\nPress any key to continue*********\n";
         _getch();
         fflush(stdin);
         break;
     }
 }
```

```
 ifs.close();
                    cout<<"\nDo you realy want to update this record? (y/n)";
                ch=_getch();
                if(ch=='y'||ch=='Y')
                {
                    cout<<"\nEnter New Address of the Staff Member : ";
                    cin.getline(staff.member_address,220);
                   cout<<"\nPress any key to continue****";
                    _getch();
cout<<"\nProcessing ";
        int x=12;
        while(x>0)
        {cout<<". ";
          delay(400);
          x--;
        }
                    fstream fs("STAFF.DAT",ios::in|ios::out|ios::binary|ios::ate);
                    fs.seekg(0);
                    while(fs.read((char*)&staff2,sizeof(STAFF)))
                    {
                        if(!strcmp(MemberId,staff2.member_id))
                        {
                            fs.seekp(fs.tellp()-sizeof(STAFF));
                            fs.write((char*)&staff,sizeof(STAFF));
                        }
                    }
                    fs.close();
                    cout<<"\n*****Member's name  is successfully modified*****";
                }
    }
    else
    {
        cout<<"\n!!!!!MEMBER record not found!!!!!";
     }
}
}
```

```
void MEMBER::modify_pincode(char *MemberId)
{
    clrscr();
     if(!strcmp(category,"STUDENT")){
    STUDENT student("STUDENT");
    STUDENT student2("STUDENT");
    char ch,ch2;
    if(student.search_membersById(MemberId))
    {
                cout<<"\n******Member record is found******";

   ifstream ifs;
   ifs.open("STUDENT.DAT",ios::binary);
   ifs.seekg(0,ios::beg);
while(ifs.read((char*)&student,sizeof(STUDENT)))
 {
     if(!strcmp(MemberId,student.member_id))
     {



        cout<<"\n*******MEMBER  DETAILS******\n"<<endl;

 cout<<"\t\tStudent Member Id: "<<student.member_id<<endl;delay(500);
 cout<<"\t\tMember Category: "<<student.category<<endl;delay(500);
     cout<<"\t\tMember Name: "<<student.member_name<<endl;delay(500);
     cout<<"\t\tRoll Number: "<<student.roll_no<<endl;delay(500);
     cout<<"\t\t Sex: "<<student.sex<<endl;delay(500);
     cout<<"\t\tSemester: "<<student.semester<<endl;delay(500);
     cout<<"\t\tDepartment: "<<student.department<<endl;delay(500);
     cout<<"\t\tMember Address: "<<student.member_address<<endl;delay(500);
     cout<<"\t\t PinCode Number: "<<student.pincode<<endl;delay(500);
     cout<<"\t\t Contact Number: "<<student.contact_no<<endl;delay(500);
     cout<<"\t\tEmail Id: "<<student.email_id<<endl;delay(500);

         cout<<"\nPress any key to continue*********\n";
         _getch();
         fflush(stdin);
         break;
```

```
        }
    }
    ifs.close();
                        cout<<"\nDo you realy want to update this record? (y/n)";
                  ch=_getch();
                  if(ch=='y'||ch=='Y')
                  {
                      cout<<"\nEnter New  Student's pincode: ";
                      cin.getline(student.pincode,10);
                     cout<<"\nPress any key to continue****";
                      _getch();
   cout<<"\nProcessing ";
           int x=12;
           while(x>0)
           {cout<<". ";
             delay(400);
             x--;
           }
                      fstream fs("STUDENT.DAT",ios::in|ios::out|ios::binary|ios::ate);
                      fs.seekg(0);
                      while(fs.read((char*)&student2,sizeof(STUDENT)))
                      {
                          if(!strcmp(MemberId,student2.member_id))
                          {
                              fs.seekp(fs.tellp()-sizeof(STUDENT));
                              fs.write((char*)&student,sizeof(STUDENT));
                          }
                      }
                      fs.close();
                      cout<<"\n*****Member's name  is successfully modified*****";
                  }
       }
       else
       {cout<<"\n!!!!!MEMBER record not found!!!!!";}

   }


        if(!strcmp(category,"STAFF")){
       STAFF staff("STAFF");
       STAFF staff2;
```

```
    char ch,ch2;
    if(staff.search_membersById(MemberId))
    {
     cout<<"\n******Member record is found******";
   ifstream ifs;
   ifs.open("STAFF.DAT",ios::binary);
     ifs.seekg(0,ios::beg);
while(ifs.read((char*)&staff,sizeof(STAFF)))
 {
     if(!strcmp(MemberId,staff.member_id))
     {
cout<<"\t\t*******MEMBER  DETAILS******\n"<<endl;

     cout<<"\t\tStaff Member Id: "<<staff.member_id<<endl;delay(500);
     cout<<"\t\tMember Category: "<<staff.category<<endl;delay(500);
     cout<<"\t\tMember Name: "<<staff.member_name<<endl;delay(500);
     cout<<"\t\tSex: "<<staff.sex<<endl;delay(500);
     cout<<"\t\tDesignation: "<<staff.designation<<endl;delay(500);
     cout<<"\t\tQualification: "<<staff.qualification<<endl;delay(500);
     cout<<"\t\tDepartment: "<<staff.department<<endl;delay(500);
     cout<<"\t\tMember Address: "<<staff.member_address<<endl;delay(500);
     cout<<"\t\tPinCode Number: "<<staff.pincode<<endl;delay(500);
     cout<<"\t\tContact Number: "<<staff.contact_no<<endl;delay(500);
     cout<<"\t\tEmail Id: "<<staff.email_id<<endl;delay(500);
     cout<<"\nPress any key to continue*********\n";
         _getch();
         fflush(stdin);
         break;
     }
 }
 ifs.close();
                    cout<<"\nDo you realy want to update this record? (y/n)";
               ch=_getch();
               if(ch=='y'||ch=='Y')
               {
                   cout<<"\nEnter New  Staff Member's pin code: ";
                   cin.getline(staff.pincode,10);
                  cout<<"\nPress any key to continue****";
                   _getch();
cout<<"\nProcessing ";
        int x=12;
```

```
          while(x>0)
          {cout<<". ";
            delay(400);
            x--;
          }
                      fstream fs("STAFF.DAT",ios::in|ios::out|ios::binary|ios::ate);
                      fs.seekg(0);
                      while(fs.read((char*)&staff2,sizeof(STAFF)))
                      {
                          if(!strcmp(MemberId,staff2.member_id))
                          {
                              fs.seekp(fs.tellp()-sizeof(STAFF));
                              fs.write((char*)&staff,sizeof(STAFF));
                          }
                      }
                      fs.close();
                      cout<<"\n*****Member's name  is successfuly modified*****";
                  }
      }
      else
      {
          cout<<"\n!!!!!MEMBER record not found!!!!!";
       }
  }
  }
```

```
  void BOOK::modify_avail(char *BookId)
  {
      BOOK bk;
                      fstream fs("BOOK.DAT",ios::in|ios::out|ios::binary|ios::ate);
                      fs.seekg(0);
                      while(fs.read((char*)&bk,sizeof(BOOK)))
                      {
                          if(!strcmp(BookId,bk.book_id))
```

```
                            {
                                fs.seekp(fs.tellp()-sizeof(BOOK));
                                if(bk.avail==0){
                                 bk.avail=1;
                                }
                                else{
                                 bk.avail=0;
                                }
                                fs.write((char*)&bk,sizeof(BOOK));
                                break;
                            }
                        }
                        fs.close();

   }
```

```
/*****************************************
          PROPOSAL CLASS DEFINITION
*****************************************/
class PROPOSAL
{
    private:
    char member_id[20];
    char member_name[40];
    char member_email[50];
    char book_name[60];
    char authorName[60];
    char publisher[60];
    time_t date;
  public:
      PROPOSAL()
      {
          time_t date=time(NULL);
      }
      void addBookProposal(void);
      void seeProposedBooks(void);
```

```
      void deleteProposedItems(void);
      void getdata(void)
      {
          cout<<"\n\tProposed By : "<<member_name;
          cout<<"\n\t\tMember ID :   "<<member_id;
          cout<<"\n\t\tEmail Id :    "<<member_email;
          cout<<"\n\n\tProposed Book:"<<book_name;
          cout<<"\n\t\tAuthor Name : "<<authorName;
          cout<<"\n\t\tPublishers  : "<<publisher;
      }
};

  void PROPOSAL::addBookProposal(void)
  {
      char ch;
    time_t t = time(NULL);
    date = t;
      char memID[20],name[40],email[50];
      clrscr();
      gotoxy(15,4);
      cout<<"BOOK PROPOSAL FORM";
      gotoxy(13,5);
      cout<<"~~~~~~~~~~~~~~~~~~~~~";
      gotoxy(7,8);
      cout<<"Enter Your Member ID: ";
      gotoxy(7,9);
      cout<<"Enter Your Name: ";
      gotoxy(7,10);
      cout<<"Enter Your Email ID: ";
      gotoxy(30,8);
      cin.getline(memID,20);
      gotoxy(30,9);
      cin.getline(name,40);
      gotoxy(30,10);
      cin.getline(email,50);
      do{
        strcpy(member_id,memID);
        strcpy(member_name,name);
        strcpy(member_email,email);
      gotoxy(7,12);
      cout<<"Enter Proposed Book's Name: ";
```

```
    gotoxy(7,13);
    cout<<"Enter Author Name: ";
    gotoxy(7,14);
    cout<<"Enter Publishers Name: ";
      gotoxy(36,12);
    cin.getline(book_name,60);
    gotoxy(36,13);
    cin.getline(authorName,60);
    gotoxy(36,14);
    cin.getline(publisher,60);
      cout<<"Processing ";
      int x=12;
      while(x>0)
      {
          cout<<". ";
          delay(400);
          x--;
      }
          fstream fs("BookProposal.DAT",ios::app);
          fs.write((char*)this,sizeof(PROPOSAL));
          fs.close();
          gotoxy(7,18);
          cout<<"Item successfully saved.........";
          gotoxy(7,20);
          cout<<"Do you want to enter more books?(y/n)";
          //char ch;
          ch=_getch();
    }while(ch=='y' || ch=='Y');
}




void PROPOSAL::seeProposedBooks(void)
{
        clrscr();
  PROPOSAL proposal,proposal2,proposal3,proposal4;
  int file_pointer_position;
  char memID[30],memName[30],memEmail[50];
  int r1=8,row;
  gotoxy(26,2);
```

```
    SetColor(11);

    cout<<"LIST  OF  PROPOSED  BOOKS  WITH   CORRESPONDING   MEMBERS";

    cout<<"\n\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~";

    SetColor(13);

    cout<<"\n\n MEMBER ID       MEMBER NAME            BOOK NAME                    PUBLISHERS
PROPOSED ON";

cout<<"\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
";

    ifstream ifs("BookProposal.DAT",ios::binary);

    ifstream ifs4;

    ofstream ofs;

    while(ifs.read((char*)&proposal,sizeof(PROPOSAL)))

    {

    file_pointer_position=ifs.tellg();

    strcpy(memID,proposal.member_id);

    strcpy(memName,proposal.member_name);

    strcpy(memEmail,proposal.member_email);

    ifs4.open("Special.DAT",ios::binary);

   while(ifs4.read((char*)&proposal4,sizeof(PROPOSAL)))

   {

      if(!strcmp(proposal4.member_id,memID))

      {

          ifs4.close();

          ifs4.clear();

          goto level4;

      }

   }

   ifs4.close();

   ifs4.clear();

    ofs.open("Special.DAT",ios::binary|ios::app);

    ofs.write((char*)&proposal,sizeof(PROPOSAL));

    ofs.close();

    ofs.clear();

    ifs.seekg(0,ios::beg);

    row=r1;

    while(ifs.read((char*)&proposal2,sizeof(PROPOSAL)))

        {

            if(!strcmp(proposal2.member_id,memID))

            {

                gotoxy(37,r1);

                cout<<proposal2.book_name;
```

```
                gotoxy(66,r1);
                cout<<proposal2.publisher;
                gotoxy(86,r1);
                cout<<ctime(&proposal2.date);
                r1++;
            }
        }
        gotoxy(1,row+((r1-row)/2));
        cout<<memID;
        gotoxy(13,(row+(r1-row)/2));
        cout<<memName;
        gotoxy(0,r1);
        cout<<"-------------------------------------------------------------------------------------------------
----------";
    r1=r1+2;
    ifs.clear();
    ifs.ignore();
ifs.seekg(file_pointer_position,ios::beg);
goto level5;
  level4: ifs.seekg(file_pointer_position,ios::beg);
   level5:;
    }
    ifs.close();
    _getch();
   remove("Special.DAT");
  }




void PROPOSAL::deleteProposedItems(void)
  {
        clrscr();
       PROPOSAL proposal,proposal2;
       char s[30],ch;
       int flag=0;
        ifstream ifs;
        ofstream ofs;
         if(!ifs)
            cout<<"\n!!!!!!!!!!File can't be found!!!!!!!!!!!!\n";
        else{
                do{
                        clrscr();
```

```
            cout<<"\n\tEnter BOOK NAME: ";
            cin>>s;
ifs.open("BookProposal.DAT",ios::binary|ios::in);

while(ifs.read((char*)&proposal,sizeof(PROPOSAL)))
{
        if(!strcmp(s,proposal.book_name))
        {
            cout<<"\n\n\t*****Item Found*****";
            proposal.getdata();
            flag=1;
            break;
        }
}
ifs.close();
if(flag==1)
    {cout<<"\n\n\tDo you want to DELETE this record from file? (y/n)";
    ch=_getch();
    if(ch=='y' || ch=='Y')
    {
   cout<<"\n\n\tProcessing ";
              int x=12;
        while(x>0)
        {
            cout<<". ";
            delay(400);
            x--;
        }
        cout<<"\n";
     ofs.open("temp.DAT",ios::binary|ios::out);
     ifs.open("BookProposal.DAT",ios::binary|ios::in);
     ifs.seekg(0,ios::beg);
while(ifs.read((char*)&proposal2,sizeof(PROPOSAL)))
{
        if(strcmp(s,proposal2.book_name))
        {
            ofs.write((char*)&proposal2,sizeof(PROPOSAL));
        }
}
ifs.close();
ofs.close();
```

```
            remove("BookProposal.DAT");
            rename("temp.DAT","BookProposal.DAT");
            cout<<"\n\n\tThe Record has successfully deleted from the Database"<<endl;
           }
           else
            break;
            }
            else{
                cout<<"\nBook Record Not Found\n";
            }
            cout<<"\n\n\tPress any key to continue*********\n";
        _getch();
        fflush(stdin);
           cout<<"\n\n\tDo you want to DELETE more records? (y/n)";
        ch=_getch();
        if(ch=='n' || ch=='N')
            break;
        }while(ch=='y'||ch=='Y');
        }
  }



/*************************
SETTINGS  CLASS  DEFINITION
*************************/

class SETTINGS
{
    public:
    int bookFineForStaff;
    int bookFineForStudent;
    int maxBooksForStaff;
    int maxBooksForStudent;
    time_t retBookWithinForStudent;
    time_t retBookWithinForStaff;
    private:
        char adminUserName[30];
    char adminPassword[15];
    public:
        SETTINGS()
```

```
{
    bookFineForStaff=7;
    bookFineForStudent=5;
    maxBooksForStaff=9;
    maxBooksForStudent=6;
    retBookWithinForStudent=1296000;
    retBookWithinForStaff=2419200;
    strcpy(adminUserName,"a");
    strcpy(adminPassword,"1");
}
char* ret_adminUserName()
{
            ifstream ifs("SETTINGS.DAT");
            ifs.read((char*)this,sizeof(SETTINGS));
            ifs.close();
            return adminUserName;
}

        char* ret_adminPassword()
{
            ifstream ifs("SETTINGS.DAT");
            ifs.read((char*)this,sizeof(SETTINGS));
            ifs.close();
            return adminPassword;
}

void modify_bookFineForStaff()
{
    clrscr();
           SETTINGS settings;
               ofstream ofs("SETTINGS.DAT");
               //ofs.seekg(0);
                                    ifstream ifs("SETTINGS.DAT");
            ifs.read((char*)this,sizeof(SETTINGS));
            ifs.close();
               cout<<"\n\t\tEnter fine for Staff members for books:  ";
               cin>>bookFineForStaff;
               ofs.write((char*)this,sizeof(SETTINGS));
               ofs.close();
    cout<<"\n\n\t\tProcessing ";
   int x=12;
```

```
   while(x>0)
   {cout<<". ";
     delay(400);
     x--;
   }
               cout<<"\n\n\t\tFine for Books for staff members successfully updated";
}


        void modify_bookFineForStudent()
{
    clrscr();
           SETTINGS settings;
               ofstream ofs("SETTINGS.DAT");
            ifstream ifs("SETTINGS.DAT");
            ifs.read((char*)this,sizeof(SETTINGS));
            ifs.close();
               cout<<"\n\t\tEnter fine for Student members for books:  ";
               cin>>bookFineForStudent;
               ofs.write((char*)this,sizeof(SETTINGS));
               ofs.close();
                                                    cout<<"\n\n\t\tProcessing ";
   int x=12;
   while(x>0)
   {cout<<". ";
     delay(400);
     x--;
   }
               cout<<"\n\n\t\tFine for Books for student members successfully updated";
}


        void modify_maxBooksForStaff()
{
    clrscr();
           SETTINGS settings;
               ofstream ofs("SETTINGS.DAT");
               //ofs.seekg(0);
                                    ifstream ifs("SETTINGS.DAT");
            ifs.read((char*)this,sizeof(SETTINGS));
            ifs.close();
               cout<<"\n\t\tEnter maximum no Book can be issued for Staff members for books:  ";
               cin>>maxBooksForStaff;
```

```
                ofs.write((char*)this,sizeof(SETTINGS));
                ofs.close();
                                        cout<<"\n\n\t\tProcessing ";
   int x=12;
   while(x>0)
   {cout<<". ";
     delay(400);
     x--;
   }
                cout<<"\n\n\t\tBook limit for student members successfully updated";
}


            void modify_maxBooksForStudent()
{
    clrscr();


            ofstream ofs("SETTINGS.DAT");
            ifstream ifs("SETTINGS.DAT");
            ifs.read((char*)this,sizeof(SETTINGS));
            ifs.close();
                cout<<"\n\t\tEnter maximum no Book can be issued for Student members for books:  ";
                cin>>maxBooksForStudent;
                ofs.write((char*)this,sizeof(SETTINGS));
                ofs.close();
                cout<<"\n\n\t\tProcessing ";
   int x=12;
   while(x>0)
   {cout<<". ";
     delay(400);
     x--;
   }
                cout<<"\n\n\t\tBook limit for staff members successfully updated";
}




void modify_retBookWithinForStudent()
{clrscr();
    time_t mytime;
           SETTINGS settings;
                ofstream ofs("SETTINGS.DAT");
```

```
            ifstream ifs("SETTINGS.DAT");
            ifs.read((char*)this,sizeof(SETTINGS));
            ifs.close();
               cout<<"\n\t\tEnter maximum no Days Book can be kept by a Student member :  ";
               cin>>mytime;
               retBookWithinForStudent=mytime*86400;

               ofs.write((char*)this,sizeof(SETTINGS));
               ofs.close();
                cout<<"\n\n\t\tProcessing ";
   int x=12;
   while(x>0)
   {cout<<". ";
     delay(400);
     x--;
   }
            cout<<"\n\n\t\tNo of Book keeping days for student members successfully updated";
}


void modify_retBookWithinForStaff()
{
    clrscr();
    time_t mytime;
           SETTINGS settings;
               ofstream ofs("SETTINGS.DAT");
               cout<<"\n\t\tEnter maximum no Days Book can be kept by a Staff member :  ";
               cin>>mytime;
               retBookWithinForStaff=mytime*86400;
               ofs.write((char*)this,sizeof(SETTINGS));
               ofs.close();
                cout<<"\n\n\t\tProcessing ";
   int x=12;
   while(x>0)
   {cout<<". ";
     delay(400);
     x--;
   }
            cout<<"\n\n\t\tNo of Book keeping days for staff members successfully updated";
}
```

```
   void modify_adminUserName();
 void modify_adminPassword();
};

       void SETTINGS::modify_adminUserName()
        {
            clrscr();
            int valid=0;
            char uname[30],uname2[30],uname3[30];
            SETTINGS settings,settings2;
            ifstream ifs("SETTINGS.DAT");
            ifs.read((char*)&settings,sizeof(SETTINGS));
            ifs.close();
            while(!valid)
          {
            cout<<"\n\t\tEnter old User Name : ";
            cin>>uname;
            if(!strcmp(uname,settings.adminUserName))
            {
                cout<<"\n\t\tEnter new User Name : ";
                cin>>uname2;
                cout<<"\n\t\tConfirm User Name : ";
                cin>>uname3;
                if(!strcmp(uname2,uname3))
                {
                    ifstream ifs("SETTINGS.DAT");
                    ifs.read((char*)this,sizeof(SETTINGS));
                    ifs.close();
                    strcpy(adminUserName,uname2);
                    ofstream ofs("SETTINGS.DAT");
                    ofs.write((char*)this,sizeof(SETTINGS));
                    ofs.close();
                    cout<<"\n\n\t\tProcessing ";
                    int x=12;
                    while(x>0)
                    {cout<<". ";
                    delay(400);
                    x--;
                    }
                    cout<<"\n\n\t\t*****Use Name successfully updated*****";
```

```
            valid=1;
            _getch();

        }
        else
            {
                cout<<"\n\t\tUser Name Mismatched!!!! Please Enter Correctly";
                valid=0;
            }
    }
    else
        {
            cout<<"\n\t\t!!!!Enter valid User Name!!!!";
            valid=0;
        }
  }
}



void SETTINGS::modify_adminPassword()
{
    clrscr();
    int valid=0;
    char pwd[30],pwd2[30],pwd3[30];
    SETTINGS settings,settings2;
    ifstream ifs("SETTINGS.DAT");
    ifs.read((char*)&settings,sizeof(SETTINGS));
    ifs.close();
    while(!valid)
  {
    cout<<"\n\t\tEnter old Password : ";
    int p = 0;
        do
    {
        pwd[p] = _getch();
        if (pwd[p] != 13)
        cout<<"*";
        p++;
    } while (pwd[p-1] != 13);
```

```
pwd[p-1] = 0;



if(!strcmp(pwd,settings.adminPassword))
{
    cout<<"\n\t\tEnter new Password : ";
    p = 0;
    do
{
    pwd2[p] = _getch();
    if (pwd2[p] != 13)
    cout<<"*";
    p++;
} while (pwd2[p-1] != 13);

pwd2[p-1] = 0;



    cout<<"\n\t\tConfirm Password : ";
p = 0;
    do
{
    pwd3[p] = _getch();
    if (pwd3[p] != 13)
    cout<<"*";
    p++;
} while (pwd3[p-1] != 13);

pwd3[p-1] = 0;
    if(!strcmp(pwd2,pwd3))
    {
        ifstream ifs("SETTINGS.DAT");
        ifs.read((char*)this,sizeof(SETTINGS));
        ifs.close();
        strcpy(adminPassword,pwd2);
        ofstream ofs("SETTINGS.DAT");
        ofs.write((char*)this,sizeof(SETTINGS));
        ofs.close();
        cout<<"\n\n\t\tProcessing ";
        int x=12;
        while(x>0)
```

```
                    {cout<<". ";
                    delay(400);
                    x--;
                    }
                    cout<<"\n\n\t\t*****Password successfully updated*****";
                    valid=1;
                    _getch();
                }
                else
                    {
                        cout<<"\n\t\tPassword Mismatched!!!! Please Enter Correctly";
valid=0;
                    }
            }
            else
                {
                    cout<<"\n\t\t!!!!Enter valid Password!!!!";
                    valid=0;
                }
          }
        }
```

```
/***************************************************************************
This function will generate a list of issued books with specified member_id
***************************************************************************/

void ISSUE::list_of_books_issued_to(char *memID)
{
    SETTINGS settings;
    ifstream ifs;
    ifs.open("SETTINGS.DAT");
    ifs.read((char*)&settings,sizeof(SETTINGS));
    ifs.close();
    STAFF stf("STAFF");
    ISSUE isu;
    int f=0,fine=0,row=10;
    time_t t;
    t=time(NULL);
```

```
        char buffer[30],buffer2[30];
        char name[40];
    struct tm *ptr_time,*ptr_time2;
    ifs.open("ISSUE.DAT",ios::binary);
    SetColor(11);
    cout<<"\n\n\t\t\t\t   BOOKS  ISSUED  TO ";
    cout<<"\n\t\t\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~";
    SetColor(13);
    cout<<"\n\n  BOOK ID             BOOK  NAME                        ISSUE DATE                RETURN DATE
FINE(in Rs.)";
    cout<<"\n-------------------------------------------------------------------------------------------------------
-----------------";
    while(ifs.read((char*)&isu,sizeof(ISSUE)))
        {

        if( !strcmp(isu.member_id,memID))
        {

        strcpy(name,isu.member_name);
        gotoxy(55,3);
        cout<<name;
            if(isu.rdate < t )
            {
                gotoxy(2,row);
                cout<<isu.book_id;
                gotoxy(14,row);
                cout<<isu.book_title;
                gotoxy(51,row);
                ptr_time=localtime(&isu.idate);
                strftime(buffer, 30,"%d %B(%A),%Y",ptr_time);
                cout<<buffer;
                gotoxy(78,row);
                ptr_time=localtime(&isu.rdate);
                strftime(buffer, 30,"%d %B(%A),%Y",ptr_time);
                cout<<buffer;
                gotoxy(105,row);
                if(!strcmp(isu.category,"STUDENT"))
                cout<<((t - isu.rdate)/86400)*settings.bookFineForStudent;
                if(!strcmp(isu.category,"STAFF"))
                cout<<((t - isu.rdate)/86400)*settings.maxBooksForStaff;
                row=row+2;
            }
```

```
            else
            {
                gotoxy(2,row);
                cout<<isu.book_id;
                gotoxy(14,row);
                cout<<isu.book_title;
                gotoxy(51,row);
                ptr_time=localtime(&isu.idate);
                strftime(buffer, 30,"%d %B(%A),%Y",ptr_time);
                cout<<buffer;
                gotoxy(78,row);
                ptr_time=localtime(&isu.rdate);
                strftime(buffer, 30,"%d %B(%A),%Y",ptr_time);
                cout<<buffer;
                gotoxy(105,row);
                cout<<" NIL";
                row=row+2;
            }
        }
        }
        gotoxy(1,row);
cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~";
        gotoxy(0,row+1);
        cout<<" TOTAL  FINE
"<<isu.fine_generation(memID);
        cout<<"\n\n Press any key to continue";
        _getch();
}




/*************************************************************************************
        This function displays the list of issued books with corresponding name
*************************************************************************************/

void ISSUE::isuued_bookList_with_correspoding_members()
{
```

```
    clrscr();

    STUDENT student("STUDENT");

    STAFF staff("STAFF");

    BOOK book;

    ISSUE isu,isu2,isu3,isu4;

    int file_pointer_position;

    char memID[30],memName[30];

    int r1=8,row;

    gotoxy(26,2);

    cout<<"LIST  OF  ISSUED  BOOKS  WITH   CORRESPONDING   MEMBERS";

    cout<<"\n\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~";

    cout<<"\n\n MEMBER ID       MEMBER NAME         BOOK ID              BOOK NAME                   FINE
TOTAL FINE";

cout<<"\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~";

    ifstream ifs("ISSUE.DAT",ios::binary);

    ifstream ifs4;

    ofstream ofs;

    while(ifs.read((char*)&isu,sizeof(ISSUE)))

    {


    file_pointer_position=ifs.tellg();

    strcpy(memID,isu.member_id);

    strcpy(memName,isu.member_name);

    ifs4.open("Special.DAT",ios::binary);

   while(ifs4.read((char*)&isu4,sizeof(ISSUE)))

   {

       if(!strcmp(isu4.member_id,memID))

       {

           ifs4.close();

           ifs4.clear();

           goto level4;

       }

   }

   ifs4.close();

   ifs4.clear();

    ofs.open("Special.DAT",ios::binary|ios::app);

    ofs.write((char*)&isu,sizeof(ISSUE));

    ofs.close();

    ofs.clear();

    ifs.seekg(0,ios::beg);
```

```
    row=r1;
    while(ifs.read((char*)&isu2,sizeof(ISSUE)))
        {
            if(!strcmp(isu2.member_id,memID))
            {
                gotoxy(38,r1);
                cout<<isu2.book_id;
                gotoxy(50,r1);
                cout<<isu2.book_title;
                gotoxy(86,r1);
                cout<<isu3.fine_for(isu2.book_id)<<" /-";
                r1++;
            }
        }
        gotoxy(1,row+((r1-row)/2));
        cout<<memID;
        gotoxy(13,(row+(r1-row)/2));
        cout<<memName;
        gotoxy(97,(row+(r1-row)/2));
        cout<<isu3.fine_generation(memID)<<" /-";
        gotoxy(0,r1);
        cout<<"-------------------------------------------------------------------------------------------------
-----";
    r1=r1+2;
    ifs.clear();
    ifs.ignore();
ifs.seekg(file_pointer_position,ios::beg);
goto level5;
  level4: ifs.seekg(file_pointer_position,ios::beg);
   level5:;
    }
    ifs.close();
    _getch();
   remove("Special.DAT");

}
```

```
/*****************************************************************
        This function will calculate fine for specified member id
```

```
*****************************************************************/


int ISSUE::fine_for(char *bkID)
{
    SETTINGS settings;
    ifstream ifs;
    ifs.open("SETTINGS.DAT");
    ifs.read((char*)&settings,sizeof(SETTINGS));
    ifs.close();
    ISSUE isu;
    int f=0;
    time_t t;
    t=time(NULL);
    ifs.open("ISSUE.DAT",ios::binary);
    while(ifs.read((char*)&isu,sizeof(ISSUE)))
        {
        if( !strcmp(isu.book_id,bkID) && !strcmp(isu.category,"STUDENT"))
        {

            if(isu.rdate < t )
            {
                f=((t - isu.rdate )/86400)*(settings.bookFineForStudent);
                break;
            }
        }

                if( !strcmp(isu.book_id,bkID) && !strcmp(isu.category,"STAFF"))
        {

            if(isu.rdate < t )
            {
                f=((t - isu.rdate )/86400)*(settings.bookFineForStaff);
                break;
            }
        }
    }
    return f;
}
```

```
/*************************************************************************************************
        This function will generate a list of books for a particular member id having fines
*************************************************************************************************/

void ISSUE::books_with_fines(char *memID)
{
    SETTINGS settings;
    ifstream ifs;
    ifs.open("SETTINGS.DAT");
    ifs.read((char*)&settings,sizeof(SETTINGS));
    ifs.close();
    clrscr();
    ISSUE isu;
    int f=0,fine=0,row=6;
    time_t t;
    t=time(NULL);
    ifs.open("ISSUE.DAT",ios::binary);
    cout<<"\n\n\t\t\t\t     BOOKS   WITH   FINES ";
    cout<<"\n\t\t\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~";
    cout<<"\n\tBOOK ID             BOOK  NAME                    FINE(in Rs.)";
    while(ifs.read((char*)&isu,sizeof(ISSUE)))
        {
        if( !strcmp(isu.member_id,memID) && !strcmp(isu.category,"STUDENT"))
        {
            if(isu.rdate < t )
            {
                gotoxy(8,row);
                cout<<isu.book_id;
                gotoxy(16,row);
                cout<<isu.book_title;
                gotoxy(31,row);
                cout<<((t - isu.rdate)/86400)*settings.bookFineForStudent;
                row=row+2;
            }
        }

                if( !strcmp(isu.member_id,memID) && !strcmp(isu.category,"STAFF"))
        {
            if(isu.rdate < t )
            {
```

```
                gotoxy(8,row);
                cout<<isu.book_id;
                gotoxy(16,row);
                cout<<isu.book_title;
                gotoxy(31,row);
                cout<<((t - isu.rdate)/86400)*settings.bookFineForStaff;
                row=row+2;
            }
        }
        }
        gotoxy(6,row);
        cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~";
        gotoxy(5,row+1);
        cout<<"\t Total                                                    "<<isu.fine_generation(memID);
        cout<<"\n Press any key to continue";
        _getch();
}
```

```
/*****************************************************************************
    This function will calculate the total fine for a particular member_id
*****************************************************************************/
int ISSUE::fine_generation(char *memID)
{
    SETTINGS settings;
    ifstream ifs;
    ifs.open("SETTINGS.DAT");
    ifs.read((char*)&settings,sizeof(SETTINGS));
    ifs.close();
    ISSUE isu;
    int f=0,fine=0;
    time_t t;
    t=time(NULL);
    ifs.open("ISSUE.DAT",ios::binary);
    while(ifs.read((char*)&isu,sizeof(ISSUE)))
        {
        if( !strcmp(isu.member_id,memID) && !strcmp(isu.category,"STUDENT"))
        {
```

```
            if(isu.rdate < t )
            {
                f=(t -isu.rdate)/86400;
                fine= fine + (f*settings.bookFineForStudent);
            }
        }

            if( !strcmp(isu.member_id,memID)&& !strcmp(isu.category,"STAFF"))
        {

            if(isu.rdate < t )
            {
                f=(t -isu.rdate)/86400;
                fine= fine + (f*settings.bookFineForStaff);
            }
        }
    }
    return fine;
}




/*************************************************
        This function is used to issue a book
*************************************************/

void ISSUE::issueBook(){
ISSUE isu,isu5;
int f;
SETTINGS settings;
ifstream ifstr("SETTINGS.DAT",ios::binary);
ifstr.read((char*)&settings,sizeof(SETTINGS));
ifstr.close();
    BOOK book;
    MEMBER member;
    STUDENT student("STUDENT");
    STAFF staff("STAFF");
    char buffer[30],buffer2[30],bkID[30],ch,memID[30],ctg[10];
    struct tm *ptr_time;
```

```
        ptr_time=localtime(&isu.idate);
        strftime(buffer, 30,"%d %B(%A),%Y",ptr_time);

        int valid;
        fstream fs;
    do
    {
valid = 1 ;
while (1)
{
clrscr() ;
gotoxy(5,2) ;
cout <<"Date : " <<buffer;
gotoxy(72,1) ;
cout <<"<0>=Exit" ;
gotoxy(5,5) ;
cout <<"Enter Code of the Book to be issued" ;
gotoxy(5,6) ;
cout <<"                  or                        " ;
gotoxy(5,7) ;
cout <<"Press <ENTER> for help  " ;
gets(bkID);
if (bkID[0] == 48)
return;
if (strlen(bkID)<1)
book.lists() ;
else
break ;
}

            if (!searchbooks_by_BookId(bkID))
{
valid = 0 ;
gotoxy(5,10) ;
cout <<"\7Record not found" ;
gotoxy(5,11) ;
cout <<"Press <ESC> to exit or any other key to continue..." ;
ch = _getch() ;
if (ch == 27)
return ;
}
```

```
} while (!valid) ;

if (searchbooks_by_BookId(bkID) && check_avail(bkID)==0)
{
gotoxy(5,10) ;
cout <<"Sorry!!!!!!!!!!  ";
searchbks_by_BookId(bkID);
cout<<"  is not available!!!!!!!!!" ;
gotoxy(5,12) ;
cout <<"Kindly issue any other Book" ;
gotoxy(5,13) ;
cout <<"See List of Books" ;
getch() ;
return ;
}
do
{
valid = 1 ;
while (1)
{
level500:clrscr() ;
gotoxy(5,2) ;
cout <<"Date : " <<buffer;
gotoxy(72,1) ;
cout <<"<0>=Exit" ;
gotoxy(5,5) ;
cout <<"Book Name to be issued: " ;searchbks_by_BookId(bkID) ;
            gotoxy(5,7) ;
cout<<"Enter Category of the member(STUDENT / STAFF): ";
            gets(ctg);
            if(strcmp(ctg,"STUDENT")&& strcmp(ctg,"STAFF"))
            {
                gotoxy(5,9);
                cout<<"Not a valid Category";
                gotoxy(5,11);
                cout<<"Check Category of the Member and try again";
                _getch();
                goto level500;
            }
gotoxy(5,8) ;
cout <<"Enter Member ID of the Member     or  " ;
```

```
gotoxy(5,9);
cout <<"Press <ENTER> for help :   " ;
gets(memID);
            f=isu.fine_generation(memID);
            if(f > 0)
            {
                    gotoxy(5,10);
                    cout<<"Sorry !!!!! ";//<<member.get_memberName(memID)<<" ,You have some Books with Fine";
                    gotoxy(5,12);
                    cout<<"Kindly return those books otherwise you can not issue any further book";
                    gotoxy(5,14);
                    cout<<"Press Enter to see the list of books having fine";
                    gotoxy(15,15);
                    cout<<"or,";
                    gotoxy(5,16);
                    cout<<"Press any key to continue";
                    char ch;
                    ch=_getch();
                    if(ch==13)
                    isu.books_with_fines(memID);
            }
if (memID[0] == 48)
return;
if (strlen(memID)<1)
{
if(!strcmp(ctg,"STUDENT"))
      {
           student.lists();
      }
      if(!strcmp(ctg,"STAFF"))
{
staff.lists() ;
}
}
else
{
if(!memIDFound(memID))
            {
                gotoxy(5,11);
                cout<<"Member Id Not found";
                gotoxy(5,12);
```

```
                cout<<"Check Member ID and try again";
                _getch();
                goto level500;
            }


else
     break;
}
            if (!staff.search_membersById(memID) && valid && !strcmp(ctg,"STAFF"))
{
valid = 0 ;
gotoxy(5,14) ;
cout <<"Staff Member Record not found" ;
gotoxy(5,15) ;
cout <<"Press <ESC> to exit or any other key to continue..." ;
ch = _getch() ;
if (ch == 27)
return ;
}

if (!student.search_membersById(memID) && valid && !strcmp(ctg,"STUDENT"))
{
valid = 0 ;
gotoxy(5,14) ;
cout <<"Student Member Record not found" ;
gotoxy(5,15) ;
cout <<"Press <ESC> to exit or any other key to continue..." ;
ch = _getch() ;
if (ch == 27)
return ;
}
} while (!valid) ;
     char ch2;
        gotoxy(5,16);
        cout<<"Press Enter to issue";
        ch2=getch();
        if(ch2==13){
                gotoxy(5,18);
            cout<<"Processing ";
           int x=12;
           while(x>0)
```

```
            {cout<<". ";
              delay(400);
              x--;
            }
int counts=1;
        ifstream ifs("ISSUE.DAT",ios::binary);
    while(ifs.read((char*)&isu5,sizeof(ISSUE)))
    {
        if(isu5.member_id,memID)
        {
            counts++;
        }
    }
    ifs.close();
            if(counts>=settings.maxBooksForStudent && !strcmp(ctg,"STUDENT"))
            {
                gotoxy(6,21);
                cout<<"You can issue maximum 6 books at a time";
                gotoxy(6,23);
                cout<<"You have already issued 6 books, can not issue more Books";
                gotoxy(6,25);
                cout<<"Kindly return those books for further transactions";
                gotoxy(4,27);
                cout<<"Press any key for continue.............";
                _getch();
                return;
            }

                        if(counts>=settings.maxBooksForStaff && !strcmp(ctg,"STAFF"))
            {
                gotoxy(6,21);
                cout<<"You can issue maximum 9 books at a time";
                gotoxy(6,23);
                cout<<"You have already issued 9 books, can not issue more Books";
                gotoxy(6,25);
                cout<<"Kindly return those books for further transactions";
                gotoxy(4,27);
                cout<<"Press any key for continue.............";
                _getch();
                return;
            }
```

```
            ISSUE isu2;
            isu2.set_data(bkID,memID,ctg,"ISSUED");
            fs.open("ISSUE.DAT",ios::app|ios::binary);
            fs.write((char *)&isu2,sizeof(ISSUE) );
            book.modify_avail(bkID);
            gotoxy(5,21);
            book.searchbks_by_BookId(bkID);
            cout<<"  is Successfully issued to Member id "<<memID;
            struct tm *ptr_time2;
            ptr_time2=localtime(&isu2.rdate);
            strftime(buffer2, 30,"%d %B(%A),%Y",ptr_time2);
            gotoxy(5,23);
            cout<<"You should return the book Within "<<buffer2;
            fs.close();
            fs.open("TRANSACTION.DAT",ios::app|ios::binary);
            fs.write((char*)&isu2,sizeof(ISSUE));
            fs.close();
        }
}
```

```
/*****************************************************************
        This function is used to set data to the ISSUE object
*****************************************************************/

void ISSUE::set_data(char *bkID,char *memID,char *ctg,char *stts)
{
    SETTINGS settings;
ifstream ifstr("SETTINGS.DAT",ios::binary);
ifstr.read((char*)&settings,sizeof(SETTINGS));
ifstr.close();
    ISSUE issue;
    BOOK bk;
    MEMBER mem;
    STAFF staff("STAFF");
    STUDENT student("STUDENT");
    ifstream ifs;
```

```
ifs.open("BOOK.DAT",ios::binary);
while(ifs.read((char*)&bk,sizeof(BOOK)))
{
    if(!strcmp(bk.book_id, bkID))
    {
        strcpy(book_title,bk.book_title);
    break;
    }
}
ifs.close();
if(!strcmp(ctg,"STAFF"))
{

ifs.open("STAFF.DAT",ios::binary);
while(ifs.read((char*)&staff,sizeof(STAFF)))
{
    if(!strcmp(staff.member_id, memID))
    {
        strcpy(member_name,staff.member_name);
        strcpy(category,ctg);
                idate = issue.idate;
rdate = idate + settings.retBookWithinForStaff;
    break;
    }
}
}
if(!strcmp(ctg,"STUDENT"))
{
ifs.open("STUDENT.DAT",ios::binary);
while(ifs.read((char*)&student,sizeof(STUDENT)))
{
    if(!strcmp(student.member_id, memID))
    {
        strcpy(member_name,student.member_name);
        strcpy(category,ctg);
            idate = issue.idate;
            rdate = idate + settings.retBookWithinForStudent;
    break;
    }
}
}
```

```
    ifs.close();
    strcpy(book_id,bkID);
    strcpy(member_id,memID);
    strcpy(status,stts);
}









/****************************************************************
        This function is used to deposit a book to the library
****************************************************************/

void ISSUE::depositBook()
{
  16:
        ISSUE isu;
        ISSUE isu10;
int f;
char ch;
    BOOK book;
    MEMBER member;
    STUDENT student("STUDENT");
    STAFF staff("STAFF");
    time_t t=time(NULL);
    char buffer[30],buffer2[30],bkID[30],memID[30];
struct tm *ptr_time;
ptr_time=localtime(&t);
strftime(buffer, 30,"%d %B(%A),%Y",ptr_time);
    do
{
clrscr() ;
gotoxy(5,2) ;
cout <<"Date : " <<buffer;
gotoxy(72,1) ;
cout <<"Press 0 to go back to the previous menu" ;
gotoxy(5,5) ;
cout <<"Enter Code of the Book to be deposited:   ";
            gets(bkID);
            if(bkID[0]=='0')
```

```
                return;
             if(strlen(bkID)<1)
                        goto l6;
            if(strlen(bkID)>1){
                    if(fine_for(bkID)>0)
                    {
                        gotoxy(5,6);
                        cout<<"You have to pay fine of worth Rs. "<< fine_for(bkID);
                        gotoxy(5,7);
                        cout<<"****Kindly try to return book within the due date for the next time
onwards****";
                        gotoxy(5,8);
                        cout<<"Press any key to continue.................";
                        _getch();
                    }
                    int flag=0;

                    ifstream ifs10("ISSUE.DAT",ios::binary);
                    while(ifs10.read((char*)&isu10,sizeof(ISSUE)))
                    {
                        if(!strcmp(bkID,isu10.book_id))
                        {
                            flag=1;
                            break;
                        }
                    }
                    ifs10.close();
                    if(flag==1)
                        {
                    gotoxy(5,9);
                    cout<<"Processing ";
                    int x=12;
           while(x>0)
           {cout<<". ";
             delay(400);
             x--;
           }
                     ifstream ifs;
                     ifs.open("ISSUE.DAT",ios::binary);
                     ifs.seekg(0,ios::beg);
                            ofstream ofs;
                            ofs.open("temp.DAT",ios::binary);
```

```
                    while(ifs.read((char*)&isu,sizeof(ISSUE)))
                    {
                        if(strcmp(isu.book_id,bkID))
                        {

                            ofs.write((char*)&isu,sizeof(ISSUE));
                        }
                    }
                    ifs.close();
                    ofs.close();
                    remove("ISSUE.DAT");
                    rename("temp.DAT","ISSUE.DAT");
                    strcpy(isu10.status,"RETURNED");
                    isu.modify_avail(bkID);
             gotoxy(5,11);
             book.searchbks_by_BookId(bkID);
             cout<<" is successfully deposited to the Library";
             ofs.open("TRANSACTION.DAT",ios::app|ios::binary);
             ofs.write((char*)&isu10,sizeof(ISSUE));
             ofs.close();
                 }
                 else
                 {gotoxy(5,13);
                     cout<<"!!!!!! This Book was not issued at all !!!!!!";
                 _getch();
                 }
            gotoxy(5,15);
            cout<<"Do you want to deposit more books??(y/n)";
            ch=_getch();
             }

             }while(ch=='y' || ch=='Y');
}









/************************************************
```

```
        This function is used to renew a book
************************************************/



void ISSUE::renewBook()
{
    level200:;
    SETTINGS settings;
ifstream ifstr("SETTINGS.DAT",ios::binary);
ifstr.read((char*)&settings,sizeof(SETTINGS));
ifstr.close();
    ISSUE isu;
    int f;
    char ch;
    BOOK book;
    MEMBER member;
    STUDENT student("STUDENT");
    STAFF staff("STAFF");
    time_t t=time(NULL);
    char buffer[30],buffer2[30],bkID[30],memID[30];
    struct tm *ptr_time;
    ptr_time=localtime(&t);
    strftime(buffer, 30,"%d %B(%A),%Y",ptr_time);
    do
{
clrscr() ;
gotoxy(5,2) ;
cout <<"Date : " <<buffer;
gotoxy(72,1) ;
cout <<"Press 0 to go back to the previous menu" ;
gotoxy(5,5) ;
cout <<"Enter Code of the Book to be renewed:  ";
            gets(bkID);
            if(bkID[0]=='0')
                return;

            if(strlen(bkID)==0)
                goto level200;
            if(strlen(bkID)>1){
                    if(fine_for(bkID)>0)
                    {
```

```
                        gotoxy(5,6);
                        cout<<"You have to pay fine of worth Rs. "<< fine_for(bkID);
                        gotoxy(5,7);
                        cout<<"****Kindly try to return book within the due date for the next time
onwards****";
                        gotoxy(5,8);
                        cout<<"Press any key to continue.................";
                        _getch();
                    }
                    gotoxy(5,9);
                    cout<<"Processing  ";
                    int x=12;
            while(x>0)
            {cout<<". ";
              delay(400);
              x--;
            }
                     fstream fs("ISSUE.DAT",ios::binary|ios::ate);
                     fs.seekg(0,ios::beg);
                    int flag=0;
                    while(fs.read((char*)&isu,sizeof(ISSUE)))
                    {
                        if(!strcmp(isu.book_id,bkID) && !strcmp(isu.category,"STUDENT"))
                        {
                            isu.idate=t;
                            isu.rdate=t + settings.retBookWithinForStudent;
                            fs.seekp(fs.tellp()-sizeof(ISSUE));
                            fs.write((char*)&isu,sizeof(ISSUE));
                            ptr_time=localtime(&isu.rdate);
                            strftime(buffer2, 30,"%d %B(%A),%Y",ptr_time);
                            flag=1;
                            break;
                        }

                       if(!strcmp(isu.book_id,bkID) && !strcmp(isu.category,"STAFF"))
                        {
                            isu.idate=t;
                            isu.rdate=t + settings.retBookWithinForStaff;
                            fs.seekp(fs.tellp()-sizeof(ISSUE));
                            fs.write((char*)&isu,sizeof(ISSUE));
                            ptr_time=localtime(&isu.rdate);
                            strftime(buffer2, 30,"%d %B(%A),%Y",ptr_time);
```

```
                        flag=1;
                        break;
                    }
                }
                fs.close();
                if(flag==0)
                    {
                      gotoxy(5,13);
                    cout<<"The Book with ID "<<bkID<<" was not issued at all !!!!!!!";
                   gotoxy(5,15);
                    cout<<"Please enter correct Book ID";
                    gotoxy(5,17);
                    cout<<"Press Any key to continue.....";
                    _getch();
                    return;
                    }
            }



            gotoxy(5,11);
            book.searchbks_by_BookId(bkID);
            cout<<" is successfully renewed on "<<buffer;
            gotoxy(5,13);
            cout<<"You have to return the Book within "<<buffer2;
            gotoxy(5,15);
            cout<<"Do you want to renew more books??(y/n)";
            ch=_getch();
            }while(ch=='y' || ch=='Y');
}
```

```
void MYMENU::studentDBmaintain(void)
```

```
{
    STUDENT student("STUDENT");
MYMENU menu;
char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(14);
cout <<" STUDENT DATABASE MAINTANACE MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   REGISTER  NEW STUDENT" ;
gotoxy(29,10) ;
cout <<"02 >   DELETE STUDENT RECORD" ;
gotoxy(29,11) ;
cout <<"03 >   MODIFY STUDENT RECORD" ;
gotoxy(29,12);
cout <<"04 >   EXIT " ;
gotoxy(29,15) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
               student.add_members();
           }
else
if (ch == '2')
{
            student.delete_member_records();
}else
if (ch == '3')
{
            MYMENU men;
            men.stdModifyMenu();
}
else
```

```
if (ch == '4')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a\a\a\a\a\a\a";
}
}
```

```
void MYMENU::stdModifyMenu(void){

            STUDENT student("STUDENT");
MYMENU menu;
char memID[20];
        char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(14);
cout <<" STUDENT  RECORD  MODIFY  MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   MODIFY  STUDENT  NAME" ;
gotoxy(29,10) ;
cout <<"02 >   MODIFY  STUDENT  EMAIL  ADDRESS" ;
gotoxy(29,11) ;
```

```
cout <<"03 >   MODIFY  STUDENT  RESIDENTIAL  ADDRESS" ;
gotoxy(29,12);
cout <<"04 >   MODIFY   STUDENT  CONTACT  NUMBER" ;
        gotoxy(29,13);
cout <<"05 >   MODIFY   PIN CODE" ;
gotoxy(29,14);
cout <<"06 >   EXIT " ;
gotoxy(29,17) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
                cout<<"Enter  the member  id to update record: ";
                gets(memID);
                student.modify_name(memID);
           }
else
if (ch == '2')
{
                cout<<"Enter  the member  id to update record: ";
                gets(memID);
                student.modify_email(memID);
}
else
if (ch == '3')
{
                                cout<<"Enter  the member  id to update record: ";
                gets(memID);
                student.modify_address(memID);
}
else
if (ch == '4')
{
                                cout<<"Enter  the member  id to update record: ";
                gets(memID);
                student.modify_cntctNo(memID);
}
                 else
if (ch == '5')
{
                                cout<<"Enter  the member  id to update record: ";
```

```
                gets(memID);
                student.modify_pincode(memID);
}
else
if (ch == '6')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";}
}

void MYMENU::bookDBmaintain(void){

            BOOK book;
MYMENU menu;
        char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(14);
cout <<" BOOK DATABASE MAINTANACE MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   REGISTER  NEW BOOK " ;
gotoxy(29,10) ;
cout <<"02 >   DELETE BOOK RECORD" ;
gotoxy(29,11) ;
cout <<"03 >   MODIFY BOOK RECORD" ;
gotoxy(29,12);
cout <<"04 >   EXIT " ;
gotoxy(29,15) ;
```

```
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
               book.add_new_books();
           }
else
if (ch == '2')
{
            book.delete_BookRecord();
}
else
if (ch == '3')
{
MYMENU men;
            men.bookModifyMenu();
}
else
if (ch == '4')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";}
}


void MYMENU::bookModifyMenu(void){
            char bkID[20];
            BOOK book;
MYMENU menu;
        char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
```

```
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(14);
cout <<" BOOK  MODIFY  MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   MODIFY  WHOLE  RECORD" ;
gotoxy(29,10) ;
cout <<"02 >   MODIFY  AUTHOR'S  NAME" ;
gotoxy(29,11) ;
cout <<"03 >   MODIFY  PUBLISHER'S  NAME" ;
gotoxy(29,12);
cout <<"04 >   MODIFY   BOOK  TITLE" ;
gotoxy(29,13);
cout <<"05 >   MODIFY   PRICE  OF  BOOK" ;
gotoxy(29,14);
cout <<"06 >   EXIT " ;
gotoxy(29,17) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
                           cout<<"Enter  the Book  ID to update record: ";
               gets(bkID);
               book.modify_whole(bkID);
           }
else
if (ch == '2')
{

cout<<"Enter  the Book  ID to update record: ";
               gets(bkID);
            book.modify_author(bkID);
}
else
if (ch == '3')
{
cout<<"Enter  the Book  ID to update record: ";
               gets(bkID);
           book.modify_publishers(bkID);
}
```

```
else
if (ch == '4')
{

cout<<"Enter  the Book  ID to update record: ";
               gets(bkID);
            book.modify_title(bkID);
}
else
if (ch == '5')
{

cout<<"Enter  the Book  ID to update record: ";
               gets(bkID);
            book.modify_price(bkID);
}
else
if (ch == '6')
{
         exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";
}
}


        void MYMENU::staffDBmaintain(void){
        STAFF staff("STAFF");
        MYMENU menu;
        char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(14);
```

```
cout <<" STAFF DATABASE MAINTANACE MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   REGISTER  NEW STAFF" ;
gotoxy(29,10) ;
cout <<"02 >   DELETE STAFF RECORD" ;
gotoxy(29,11) ;
cout <<"03 >   MODIFY STAFF RECORD" ;
gotoxy(29,12);
cout <<"04 >   EXIT " ;
gotoxy(29,15) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
               staff.add_members();
           }
else
if (ch == '2')
{
            staff.delete_member_records();
}
if (ch == '3')
{
MYMENU men;
            men.stfModifyMenu();
}
else
if (ch == '4')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<<"\a\a\a\a\a\a\a\a\a\a\a";}

        }
```

```
        void MYMENU::stfModifyMenu(void){

            STAFF staff("STAFF");
MYMENU menu;
char memID[20];
        char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(14);
cout <<" STAFF RECORD  MODIFY  MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   MODIFY  STAFF  NAME" ;
gotoxy(29,10) ;
cout <<"02 >   MODIFY  STAFF  EMAIL  ADDRESS" ;
gotoxy(29,11) ;
cout <<"03 >   MODIFY  STAFF  RESIDENTIAL  ADDRESS" ;
gotoxy(29,12);
cout <<"04 >   MODIFY   STAFF  CONTACT  NUMBER" ;
gotoxy(29,13);
cout <<"05 >   MODIFY   PIN CODE" ;
gotoxy(29,14);
cout <<"06 >   EXIT " ;
gotoxy(29,17) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
               cout<<"Enter  the member  id to update record: ";
               gets(memID);
```

```
                staff.modify_name(memID);
                _getch();
            }
else
if (ch == '2')
{
                                cout<<"Enter  the member  id to update record: ";
                gets(memID);
                staff.modify_email(memID);
}
else
if (ch == '3')
{
                                cout<<"Enter  the member  id to update record: ";
                gets(memID);
                staff.modify_address(memID);
}
Else if (ch == '4')
{
                                cout<<"Enter  the member  id to update record: ";
                gets(memID);
                staff.modify_cntctNo(memID);
                _getch();
}
else
if (ch == '4')
{
                  cout<<"Enter  the member  id to update record: ";
                gets(memID);
                staff.modify_pincode(memID);
                _getch();
}
else
if (ch == '5')
{
             exit(0);
}
else   if(ch==27)
         {
             break;
         }
```

```
else cout<"\a\a\a\a\a\a\a\a\a";}
}

void MYMENU::introduction(){
    clrscr() ;
gotoxy(31,5) ;
SetColor(14);
cout <<"Welcome to Library Management Project" ;
gotoxy(33,7) ;
    //system("COLOR A1");
gotoxy(25,10) ;
SetColor(10);
cout <<"This  project has facility of maintaining  records" ;
gotoxy(25,11) ;
SetColor(10);
cout <<"of BOOKS and MEMBERS." ;
gotoxy(25,13) ;
//settextstyle();
cout <<"This  project  can  hold more  than  10,000  books" ;
gotoxy(25,14) ;
cout <<"records." ;
gotoxy(25,16) ;
cout <<"Maximum 6 books,2 digital media and one magazine";
gotoxy(25,17) ;
cout<<"can be issued to a member at a time. If he/she" ;
gotoxy(25,18);
cout <<"does  not return book upto 15 days he/she have  to" ;
gotoxy(25,19) ;
cout <<"pay fine of Rs.5/- per day." ;
gotoxy(30,25) ;
//SetColor(12);
    cout<<"Press any key to continue further";
_getch() ;
}

void MYMENU::section_menu(){
char ch ;
while (1)
{
clrscr() ;
SetColor(13);
```

```
gotoxy(29,6) ;
cout <<"SECTION   MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   ADMIN SECTION" ;
gotoxy(29,10) ;
cout <<"02 >   USER SECTiON" ;
gotoxy(29,11) ;
cout <<"03 >   EXIT " ;
gotoxy(29,14) ;
cout <<"Enter your choice:  " ;
ch=getche();
if (ch == '1')
           {
               admin_menu();
           }
else
if (ch == '2')
{
            user_menu();
}
elseif (ch == '3')
{
            exit(0);
}
else cout<"\a\a\a\a\a\a\a\a\a\a\a";
}
}

void MYMENU::databaseMaintanance_menu(void){
MYMENU menu;
char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(12);
```

```
cout <<"  DATABASE MAINTANACE MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   STUDENT DATABASE MAINTANACE" ;
gotoxy(29,10) ;
cout <<"02 >   STAFF DATABASE MAINTANANCE" ;
gotoxy(29,11) ;
cout <<"03 >   BOOK DATABASE MAINTANACE" ;
gotoxy(29,12);
cout <<"07 >   EXIT " ;
gotoxy(29,15) ;
cout <<"Enter your choice:  " ;
ch=getche();
        if (ch == '1')
           {
               menu.studentDBmaintain();
           }
else
if (ch == '2')
{
            menu.staffDBmaintain();
}
        else
if (ch == '3')
{
            menu.bookDBmaintain();
}
else
if (ch == '4')
{
            exit(0);
}
else  if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a\a";
}
}
```

```
void MYMENU::issueMaintanance_menu(void){
ISSUE myissue;
MYMENU menu;
        char ch ;
while (1){
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<" ISUUE AND DEPOSIT MAINTANACE MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >    ISUUE MANAGEMENT SECTION" ;
gotoxy(29,10) ;
cout <<"02 >   DEPOSIT MANAGEMENT SECTION" ;
gotoxy(29,11);
cout <<"02 >   RENEW MANAGEMENT SECTION" ;
gotoxy(29,12);
cout <<"03 >   EXIT " ;
gotoxy(29,15) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
              myissue.issueBook();
              _getch();
           }
else
if (ch == '2')
{
          myissue.depositBook();
}
else
if (ch == '3')
{
          myissue.renewBook();
        }
else
```

```
if (ch == '4')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";}
}

void MYMENU::reportGeneration(void){

        BOOK  book;;
        STUDENT student("STUDENT");
        STAFF staff("STAFF");
   MYMENU menu;
        char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<" VIEW REPORTS" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   COMPLETE  BOOK  LIST" ;
gotoxy(29,10) ;
cout<<"02 >   ISSUED  BOOK  LIST  WITH  CORRESPONDING  MEMBERS";
gotoxy(29,11);
cout <<"03 >   COMPLETE  STUDENT   MEMBER  LIST" ;
gotoxy(29,12) ;
cout <<"04 >   COMPLETE  STAFF   MEMBER  LIST" ;
gotoxy(29,13) ;
cout <<"05 >   LIST  OF  BOOKS  ISSUED  TO  A  SPECIFIED  MEMBER" ;
gotoxy(29,14);
```

```
     cout <<"06 >   LIST  OF  BOOKS  ISSUED  TO  A  SPECIFIED  MEMBER  HAVING  FINE" ;
gotoxy(29,15);
cout<<"07  >   TOTAL  FINE  UP TO  THE  DATE  FOR  A SPECIFIED  MEMBER";
gotoxy(29,16);
cout<<"08  >   FINE  FOR  A  SPECIFIED  BOOK  UP TO  THE  DATE";
gotoxy(29,17);
cout <<"09 >   EXIT " ;
gotoxy(29,20) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1'){
             book.lists();
          }
else
if (ch == '2')
{
ISSUE isu;
   isu.isuued_bookList_with_correspoding_members();
}
else
if (ch == '3')
{
        student.lists();
}
else
if (ch == '4')
{
            staff.lists();
}
else
if (ch == '5')
{
          char memID[30];
            ISSUE isu;
            clrscr();
            cout<<"\tEnter Member Id:   ";
            gets(memID);
            isu.list_of_books_issued_to(memID);
           // staff.lists();
}
        Else if (ch == '6')
```

```
{
            char memID[30];
            ISSUE isu;
            clrscr();
            cout<<"\tEnter Member Id:   ";
            gets(memID);
            isu.books_with_fines(memID);
}
        elseif (ch == '7')
{
            char memID[30];
            ISSUE isu;
            clrscr();
            cout<<"\tEnter Member Id:   ";
            gets(memID);
            SetColor(11);
            MEMBER member;
            if(staff.search_membersById(memID)  ||  student.search_membersById(memID))
            cout<<"\n\tYour Total fine up to the date is  "<<isu.fine_generation(memID);
            else
            cout<<"\n\tMember ID not found ";
            cout<<"\n\tPress any key to continue........ ";
            _getch();
}
        elseif (ch == '8')
{
            char bookID[30];
            ISSUE isu;
            clrscr();
            cout<<"\tEnter Book Id:   ";
            gets(bookID);
            SetColor(11);
            if(book.searchbooks_by_BookId(bookID))
            cout<<"\tThe total fine up to the date for this book is  "<<isu.fine_for(bookID);
            else
            cout<<"\n\tBook ID not found ";
            cout<<"\n\tPress any key to continue........ ";
            _getch();
}
else
if (ch == '9')
```

```
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";}

}
void MYMENU::userReportGeneration(void){

        BOOK  book;;
        STUDENT student("STUDENT");
        STAFF staff("STAFF");
   MYMENU menu;
   MEMBER member;
        char ch ;
while (1){
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<" VIEW REPORTS" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01  >   COMPLETE  BOOK  LIST" ;
gotoxy(29,10) ;
cout<<"02  >   ISSUED  BOOK  LIST  WITH  CORRESPONDING  MEMBERS";
gotoxy(29,11) ;
cout <<"03  >   LIST  OF  BOOKS  ISSUED  TO  A  SPECIFIED  MEMBER" ;
gotoxy(29,12);
 cout <<"04  >   LIST  OF  BOOKS  ISSUED  TO  A  SPECIFIED  MEMBER  HAVING  FINE" ;
gotoxy(29,13);
cout<<"05  >    TOTAL  FINE  UP TO  THE  DATE  FOR  A SPECIFIED  MEMBER";
gotoxy(29,14);
cout<<"06  >    FINE  FOR  A  SPECIFIED  BOOK  UP TO  THE  DATE";
```

```
gotoxy(29,15);
cout <<"07  >   EXIT " ;
gotoxy(29,18) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
               book.lists();
           }
else
if (ch == '2')
{
           ISSUE isu;
           isu.isuued_bookList_with_correspoding_members();
}
else
if (ch == '3')
{
            char memID[30];
            ISSUE isu;
            clrscr();
            cout<<"\tEnter Member Id:   ";
            gets(memID);
            isu.list_of_books_issued_to(memID);
}
else
if (ch == '4')
{            char memID[30];
            ISSUE isu;
            clrscr();
            cout<<"\tEnter Member Id:   ";
            gets(memID);
            isu.books_with_fines(memID);
}
else
if (ch == '5')
{
            char memID[30];
            ISSUE isu;
            clrscr();
            cout<<"\tEnter Member Id:   ";
```

```
            gets(memID);
            SetColor(11);
            if(staff.search_membersById(memID)  ||  student.search_membersById(memID))
            cout<<"\n\tYour Total fine up to the date is  "<<isu.fine_generation(memID);
            else
            cout<<"\n\tMember ID not found ";
            cout<<"\n\tPress any key to continue........ ";
            _getch();
}
        Else if (ch == '6')
{
            char bookID[30];
            ISSUE isu;
            clrscr();
            cout<<"\tEnter Book Id:   ";
            gets(bookID);
            SetColor(11);
            if(book.searchbooks_by_BookId(bookID))
            cout<<"\tThe total fine up to the date for this book is  "<<isu.fine_for(bookID);
            else
            cout<<"\n\tBook ID not found ";
            cout<<"\n\tPress any key to continue........ ";
            _getch();
}
else
if (ch == '7')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";
}
}



void MYMENU::proposedBooksMenu()
{
```

```
    MYMENU menu;
    char ch,ctg[10] ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<"  PROPOSED  BOOKS  OPTIONS" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >    SEE  PROPOSED  BOOK  ITEMS" ;
gotoxy(29,10) ;
cout <<"02 >   DELETE  RECORDS" ;
gotoxy(29,11);
cout <<"03 >   EXIT " gotoxy(29,14) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
            _getch();
           }
else
if (ch == '2')
{
          _getch();
}
else
if (ch == '3')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";
```

```
}
}



void MYMENU::proposedItems(void){
        MYMENU menu;
        PROPOSAL proposal;
        char ch,ctg[10] ;
while (1){
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<" VIEW  PROPOSED  ITEMS" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   VIEW  PROPOSED  BOOKS" ;
gotoxy(29,10) ;
cout <<"02 >   DELETE  PROPOSED  ITEMS" ;
        gotoxy(29,11);
cout <<"03 >   EXIT " ;
gotoxy(29,14) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
                proposal.seeProposedBooks();
           }
else
if (ch == '2')
{
            proposal.deleteProposedItems();
}
else
if (ch == '4')
{
            exit(0);
}
```

```
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";
}
}

void MYMENU::userProposedItems(void){
        MYMENU menu;
        PROPOSAL proposal;
        char ch,ctg[10] ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<" VIEW  PROPOSED  ITEMS" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   LIST OF PROPOSED BOOKS" ;
gotoxy(29,10) ;
cout <<"02 >   ADD PROPOSAL FOR BOOKS" ;
gotoxy(29,11);
cout <<"04 >   EXIT " ;
gotoxy(29,14) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
        {
            proposal.seeProposedBooks();
        }
else
if (ch == '2')
{
            proposal.addBookProposal();
```

```
}
else if (ch == '3')
{      exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";
}

}
```

```
void MYMENU::searchMenu(void)
{
   MYMENU menu;
         char ch,ctg[10] ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<" SEARCH  OPTIONS  MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >    SEARCH  BOOKS  OPTIONS" ;
gotoxy(29,10) ;
cout <<"02 >   SEARCH  MEMBERS OPTIONS" ;
gotoxy(29,11);
cout <<"03 >   EXIT " ;
gotoxy(29,15) ;
cout <<"Enter your choice:  " ;
```

```
ch=_getch();
if (ch == '1')
           {
            menu.searchBooksMenu();
           }
else
if (ch == '2')
{
            menu.searchMembersMenu();
}
else
if (ch == '3')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";}
}

void MYMENU::userSearchMenu(void)
{
    MYMENU menu;
        char ch,ctg[10] ;
while (1){
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<" SEARCH  OPTIONS  MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >    SEARCH  BOOKS  OPTIONS" ;
gotoxy(29,10) ;
cout <<"02 >   SEARCH  MEMBERS OPTIONS" ;
```

```
gotoxy(29,12);
cout <<"03 >   EXIT " ;
gotoxy(29,15) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
            menu.searchBooksMenu();
           }
else
if (ch == '2')
{
            menu.userSearchMembersMenu();
}
else
if (ch == '3')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";}
}

void MYMENU::searchBooksMenu(void){
BOOK book;
    MYMENU menu;
        char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<" SEARCH BOOKS OPTIONS" ;
gotoxy(29,7) ;
```

```
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >    SEARCH  BOOKS  BY ID" ;
gotoxy(29,10) ;
cout <<"02 >   SEARCH  BOOKS  BY  AUTHORS'S  NAME" ;
gotoxy(29,11) ;
cout <<"03 >   SEARCH  BOOKS  BY  PUBLISHER'S  NAME" ;
gotoxy(29,12) ;
cout <<"04 >   SEARCH  BOOKS  BY  BOOK  TITLE" ;
gotoxy(29,13);
cout <<"05 >   EXIT " ;
gotoxy(29,17) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
           {
               book.searchbooks_by_BookId();
           }
else
if (ch == '2')
{
            book.searchbooks_by_AuthorName();
}
if (ch == '3')
{
                    book.searchbooks_by_Publishers();
}
if (ch == '4')
{
            book.searchbooks_by_BookName();
}
else
if (ch == '5')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
```

```
else cout<"\a\a\a\a\a\a\a\a\a";}
}

void MYMENU::searchMembersMenu(void){
STUDENT student("STUDENT");
STAFF staff("STAFF");
   MYMENU menu;
        char ch,ctg[10] ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<" SEARCH  MEMBERS  OPTIONS" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >    SEARCH  MEMBERS  BY  ID" ;
gotoxy(29,10) ;
cout <<"02 >   SEARCH  MEMBERS  BY  EMAIL  ADDRESS" ;
gotoxy(29,11) ;
cout <<"03 >   SEARCH  MEMBERS  BY  DEPARTMENT  NAME" ;
gotoxy(29,12);
cout <<"04 >   EXIT " ;
gotoxy(29,16) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
          {
              clrscr();
         cout<<"\tENTER  MEMBER  CATEGORY :   ";
         gets(ctg);
         if(!strcmp(ctg,"STUDENT"))
            student.search_membersById();
         else if(!strcmp(ctg,"STAFF"))
            staff.search_membersById();
            else
            {
```

```
                cout<<"\n\t!**!**!**!Please enter a valid category name!**!**!**!";
                _getch();
                break;
             }
           }
else
if (ch == '2')
{                    clrscr();
          cout<<"\tENTER  MEMBER  CATEGORY :   ";
          gets(ctg);
          if(!strcmp(ctg,"STUDENT"))
             student.search_membersByEmail();
          else if(!strcmp(ctg,"STAFF"))
             staff.search_membersByEmail();
             else
             {
                cout<<"\n\t!**!**!**!Please enter a valid category name!**!**!**!";
                _getch();
                break;
             }
}
if (ch == '3')
{
                clrscr();
          cout<<"\tENTER  MEMBER  CATEGORY :   ";
          gets(ctg);
          if(!strcmp(ctg,"STUDENT"))
             student.search_membersByDepartment();
          else if(!strcmp(ctg,"STAFF"))
             staff.search_membersByDepartment();
             else
             {
                cout<<"\n\t!**!**!**!Please enter a valid category name!**!**!**!";
                _getch();
                break;
             }
}
else
if (ch == '4')
{
             exit(0);
```

```
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";}
}

void MYMENU::userSearchMembersMenu(void){
STUDENT student("STUDENT");
STAFF staff("STAFF");
MYMENU menu;
        char ch,ctg[10] ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13);
cout <<" SEARCH  MEMBERS  OPTIONS" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   SEARCH  MEMBERS  BY  EMAIL  ADDRESS" ;
gotoxy(29,10) ;
cout <<"02 >   SEARCH  MEMBERS  BY  DEPARTMENT  NAME" ;
gotoxy(29,11);
cout <<"03 >   EXIT " ;
gotoxy(29,16) ;
cout <<"Enter your choice:  " ;
ch=_getch();
if (ch == '1')
{
           clrscr();
         cout<<"\tENTER  MEMBER  CATEGORY :   ";
         gets(ctg);
         if(!strcmp(ctg,"STUDENT"))
            student.search_membersByEmail();
```

```
         else if(!strcmp(ctg,"STAFF"))
            staff.search_membersByEmail();
            else
            {
                cout<<"\n\t!**!**!**!Please enter a valid category name!**!**!**!";
                _getch();
                break;
            }
}
if (ch == '2')
{                clrscr();
         cout<<"\tENTER  MEMBER  CATEGORY :   ";
         gets(ctg);
         if(!strcmp(ctg,"STUDENT"))
            student.search_membersByDepartment();
         else if(!strcmp(ctg,"STAFF"))
            staff.search_membersByDepartment();
            else
            {
                cout<<"\n\t!**!**!**!Please enter a valid category name!**!**!**!";
                _getch();
                break;
            }
}
else
if (ch == '3')
{
            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";
}
}

void MYMENU::systemSettings(void){
SETTINGS settings;
ifstream ifstr("SETTINGS.DAT",ios::binary);
```

```
ifstr.read((char*)&settings,sizeof(SETTINGS));
ifstr.close();
MYMENU menu;
char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(12);
cout <<"  SYSTEM  SETTINGS  OPTIONS" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   CHANGE USER NAME" ;
gotoxy(29,10) ;
cout <<"02 >   CHANGE PASSWORD" ;
gotoxy(29,11) ;
cout <<"03 >   CHANGE BOOKS LIMIT FOR STAFF MEMBER" ;
gotoxy(29,12) ;
cout <<"04 >   CHANGE BOOKS LIMIT FOR STUDENT MEMBER" ;
gotoxy(29,13);
cout <<"05 >   CHANGE NO. OF BOOK KEEPING DAYS FOR STAFF";
gotoxy(29,14);
cout<<"06 >   CHANGE NO. OF BOOK KEEPING DAYS FOR STUDENT";
gotoxy(29,15);
cout<<"07 >   CHANGE FINE VALUE FOR A BOOK FOR STUDENT";
gotoxy(29,16);
cout<<"08 >   CHANGE FINE VALUE FOR A BOOK FOR STAFF";
gotoxy(29,17);
cout <<"09 >   EXIT " ;
gotoxy(29,20) ;
cout <<"Enter your choice:  " ;
ch=getche();
        if (ch == '1')
           {
               settings.modify_adminUserName();
           }
else
```

```
if (ch == '2')
{
            settings.modify_adminPassword();
}
        else
if (ch == '3')
{
            settings.modify_maxBooksForStaff();
}
        else
if (ch == '4')
{
            settings.modify_maxBooksForStudent();
}
        else
if (ch == '5')
{
            settings.modify_retBookWithinForStaff();
}
else
if (ch == '6')
{
            settings.modify_retBookWithinForStudent();
}
else  if (ch == '7')
{
            settings.modify_bookFineForStudent();
}
        else
        if (ch == '8')
{
            settings.modify_bookFineForStaff();
}

else
if (ch == '9')
{
            exit(0);
}
else if(ch==27)
        {
```

```
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a\a";
}
}
```

```
void MYMENU::admin_menu(){
MYMENU menu;
SETTINGS settings;
ifstream ifstr("SETTINGS.DAT",ios::binary);
ifstr.read((char*)&settings,sizeof(SETTINGS));
ifstr.close();
char ch ,chr;
char uname[30],password[30],repassword[30];
while(1){
clrscr();
gotoxy(70,3);
cout<<"Press <0> To go back to the previous menu";
gotoxy(30,10);
cout<<"ENTER USER NAME:   ";
gotoxy(30,13);
cout<<"ENTER PASSWORD:     ";
gotoxy(30,15);
cout<<"RE-ENTER PASSWORD:     ";
gotoxy(49,10);
cin>>uname;
if(uname[0]=='0')
return;
gotoxy(49,13);
int p = 0;
do{
password[p] = _getch();
if (password[p] != 13)
cout<<"*";
p++;
} while (password[p-1] != 13);
password[p-1] = 0;
```

```
gotoxy(49,15);
  p = 0;
do
{
repassword[p] = _getch();
if (repassword[p] != 13)
cout<<"*";
p++;
} while (repassword[p-1] != 13);


repassword[p-1] = 0;


if(!strcmp(uname,settings.ret_adminUserName()) && !strcmp(password,repassword)  &&
!strcmp(password,settings.ret_adminPassword())){



    while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to LOG  OUT";
SetColor(11);
gotoxy(29,6) ;
cout <<"ADMINISTRATION  MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   DATABASE MAINTANACE" ;
gotoxy(29,10) ;
cout <<"02 >   ISUUE or DEPOSIT MAINTANANCE" ;
gotoxy(29,11) ;
cout <<"03 >   REPORT GENERATION" ;
gotoxy(29,12) ;
cout <<"04 >   SEE PROPOSED ITEMS" ;
gotoxy(29,13) ;
cout <<"05 >   SEARCH BOOKS" ;
gotoxy(29,14) ;
cout <<"06 >   SYSTEM SETTINGS" ;
gotoxy(29,15) ;
cout <<"07 >   EXIT " ;
gotoxy(29,18) ;
```

```
cout <<"Enter your choice:  " ;
ch=getche();
if (ch == '1')
            {
                menu.databaseMaintanance_menu();
            }
else
if (ch == '2')
{
             menu.issueMaintanance_menu();
}
         else
if (ch == '3')
{
             menu.reportGeneration();
}
         else if (ch == '4')
{
             menu.proposedItems();
}
else
if (ch == '5')
{
             menu.searchMenu();
}
else if (ch == '6')
{
             menu.systemSettings();
}
else
if (ch == '7')
{
             exit(0);
}
else if(ch==27){
                 clrscr();
                          gotoxy(30,12);
             cout<<"Processing ";
                         int x=10;
            while(x>0)
            {cout<<". ";
```

```
              delay(400);
              x--;
            }
             cout<<"Logging Out";
            x=10;
            while(x>0)
            {cout<<". ";
              delay(400);
              x--;
            }
             gotoxy(30,15);
             cout<<"You have successfully Logged Out";
             gotoxy(30,17);
             cout<<"Press any key for the Administrator Log In Page";
             gotoxy(40,18);
             cout<<"Or ,";
             gotoxy(30,19);
             cout<<"Press ESC for Main Menu";
             char ch;
             ch=_getch();
             if(ch==27)
             return;
             else
             break;
}
else cout<"\a\a\a\a\a\a\a\a\a\a\a\a\a\a\a\a\a\a\a";

}
}
else
{
    clrscr();
 gotoxy(30,10);
cout<<"Check user name or password and Try again . . . . ";
gotoxy(30,12);
cout<<"Press any key to continue......";
_getch();
}
}
}
```

```
void MYMENU::user_menu(){
STAFF staff("STAFF");
STUDENT student("STUDENT");
ISSUE issue;
SETTINGS settings;
ifstream ifstr("SETTINGS.DAT",ios::binary);
ifstr.read((char*)&settings,sizeof(SETTINGS));
ifstr.close();
BOOK book;
    MYMENU menu;
        char ch ;
while (1)
{
clrscr() ;
gotoxy(0,2);
SetColor(15);
cout<<"   Press ESC to go back to the previous menu";
gotoxy(29,6) ;
SetColor(13)
cout <<"      USER  MENU" ;
gotoxy(29,7) ;
cout <<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" ;
gotoxy(29,9) ;
cout <<"01 >   SEARCH  OPTIONS" ;
gotoxy(29,10) ;
cout <<"02 >   REPORT  GENERATION" ;
gotoxy(29,11) ;
cout <<"03 >   ITEMS  PROPOSAL" ;
gotoxy(29,12);
cout <<"05 >   EXIT " ;
gotoxy(29,16) ;
cout <<"Enter your choice:  " ;
ch=_getch();
    if (ch == '1')
      {
          menu.searchMenu();
       }
else if (ch == '2')
{         menu.userReportGeneration();
}
```

```
else if (ch == '3')
{
            menu.userProposedItems();
}
else if (ch == '4')
{            exit(0);
}
else
        if(ch==27)
        {
            break;
        }
else cout<"\a\a\a\a\a\a\a\a\a";
}

}
```

```
int main(){
MYMENU menu;
BOOK book;
MEMBER member;
STUDENT student("STUDENT");
STAFF staff("STAFF");
ISSUE isu;
//staff.search_membersByDepartment();
//staff.modify_cntctNo("8383");
//isu.books_with_fines("9876");
menu.introduction();
menu.section_menu();
//student.lists_byDepartment("CSE");
//book.lists_byAuthorName("Herbert Schildt");
//book.lists_byPublishers("TMH");
//book.lists_byTitle("Complete Refernce Java");
//isu.list_of_books_issued_to();
//isu.isuued_bookList_with_correspoding_members();
//time_t t=time(NULL);
//cout<<ctime(&t);
//staff.getStaffMemberDataByID();
```

```
//staff.search_membersById();
//student.search_membersByEmail();
//isu.issueBook();
//book.lists();
//student.lists();
//cout<<student.get_memberName("1234");
//cout<<book.getBookName("2863");
//isu.setBookName("2863");
//isu.setMemberName("8383");
//isu.Disp();
//cout<<book.getBookName("2863");
//member=staff.get_memberName("2948");
//cout<<member.member_name;
/*book.modify_avail("1234");
book.modify_avail("9988");
book.modify_avail("2795");
book.modify_avail("5678");
book.modify_avail("7762");*/
return 0;
}
```

Chapter 5

TESTING

Topics Covered:

5.1 UNIT TESTING

5.2 INTEGRATION TESTING

5.3 SYSTEM TESTING

5.1 UNIT TESTING

Unit testing is undertaken when a module has been created and successfully Reviewed .In order to test a single module we need to provide a complete environment i.e. besides the module we would require.

- The procedures belonging to other modules that the module under test calls
- Non local data structures that module accesses
- A procedure to call the functions of the module under test with appropriate parameters

Unit testing was done on each and every module that is described under module description of chapter 4.

Example Test cases of issueBook ( ) function:

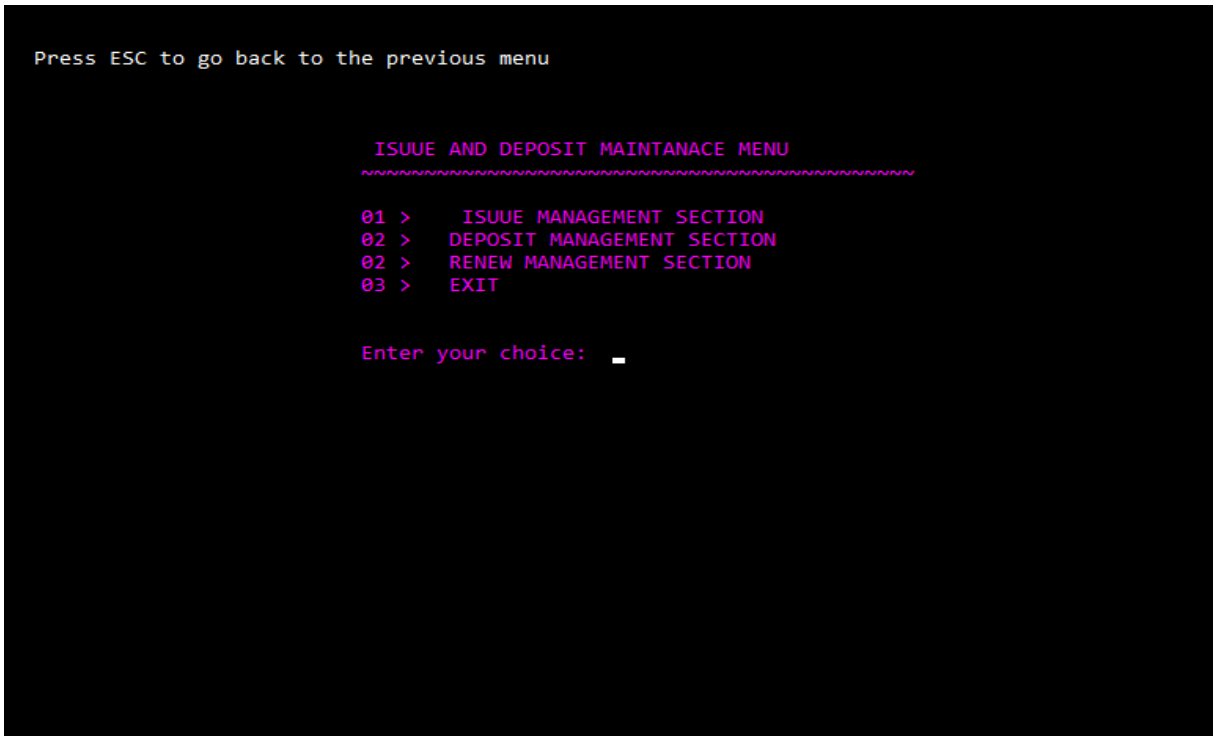
TEST CASE 1 :

System: Promting to accept book ID

Input: bkID=0;

Output: Promt to the Issue and Deposit Maintanance Menu.

Output Screen:



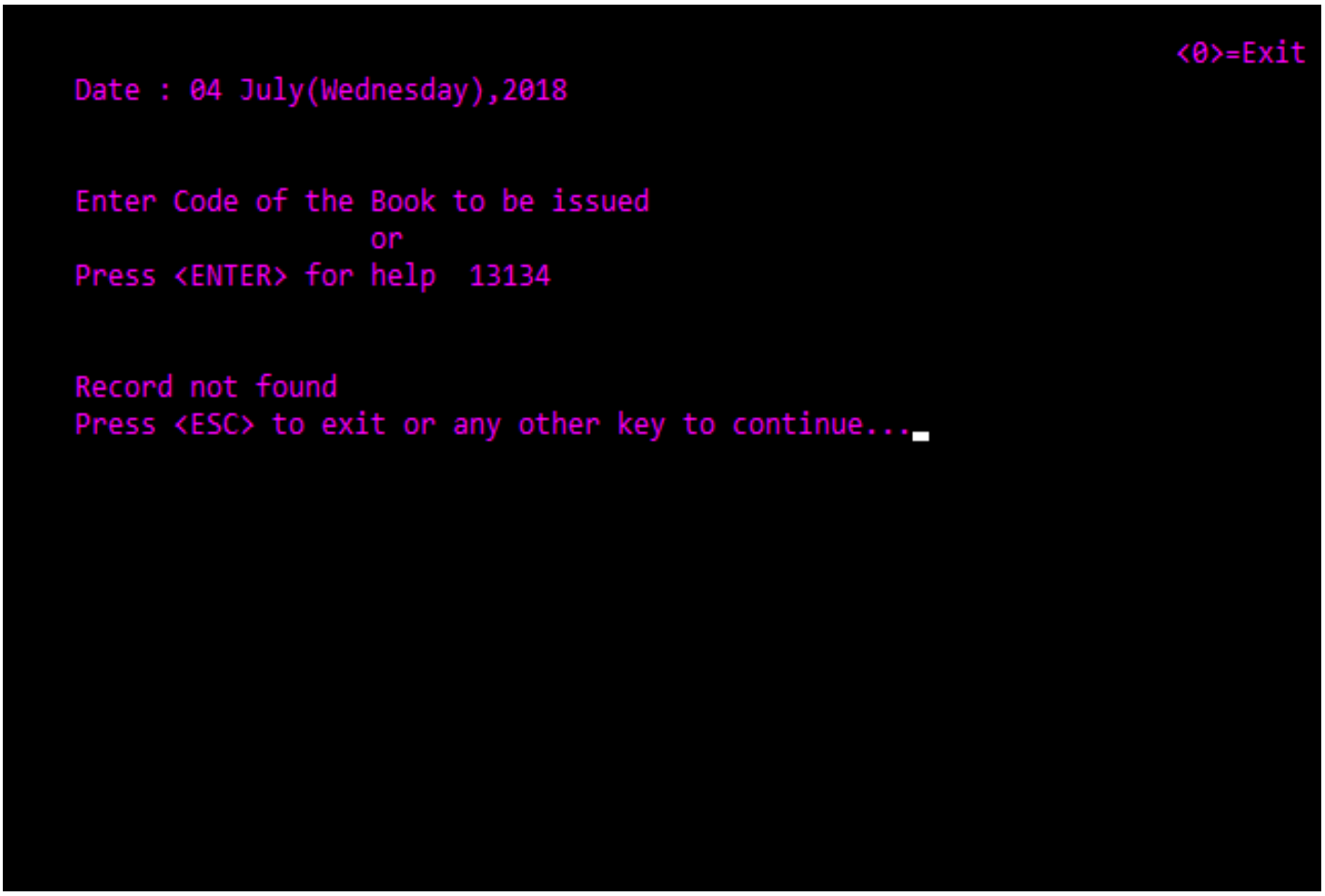
TEST CASE 2 :

System: Promting to accept book ID

Input: "13134" (Invalid Set)

Output: Record not found Message will be displayed.

Output Screen:



TEST CASE 3 :

System: Promting to accept book ID

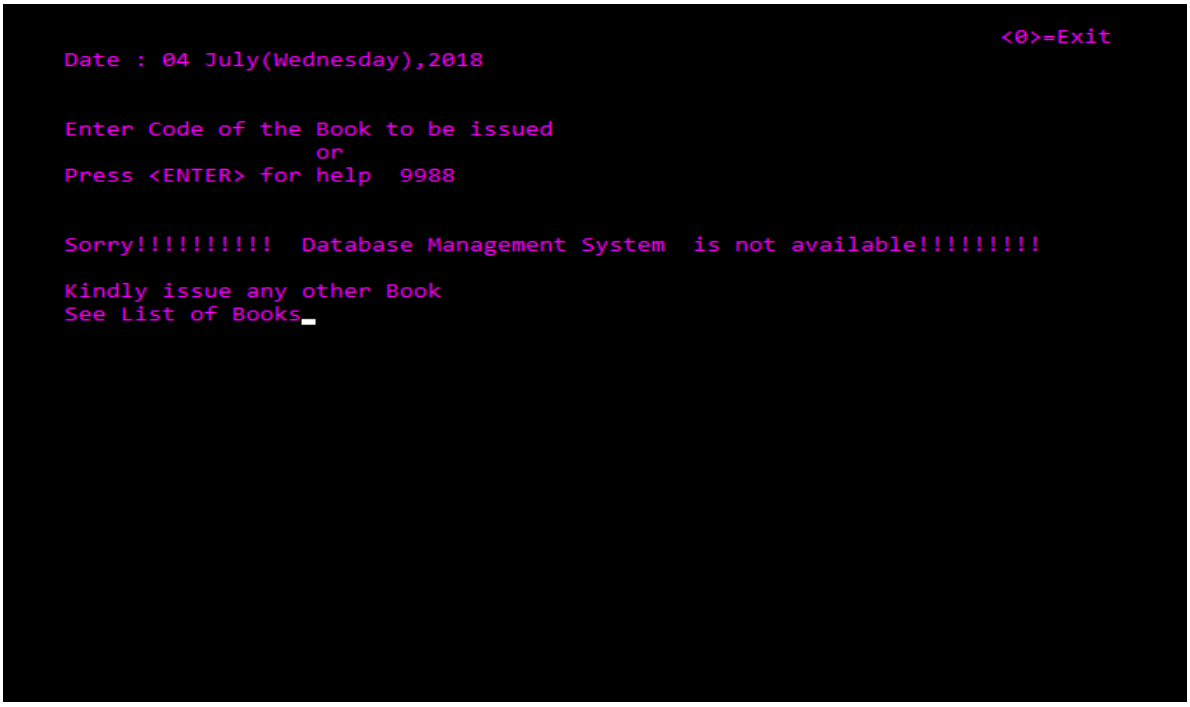
Input: "" (Blank Input)

Output: Complete Book list will be displayed.

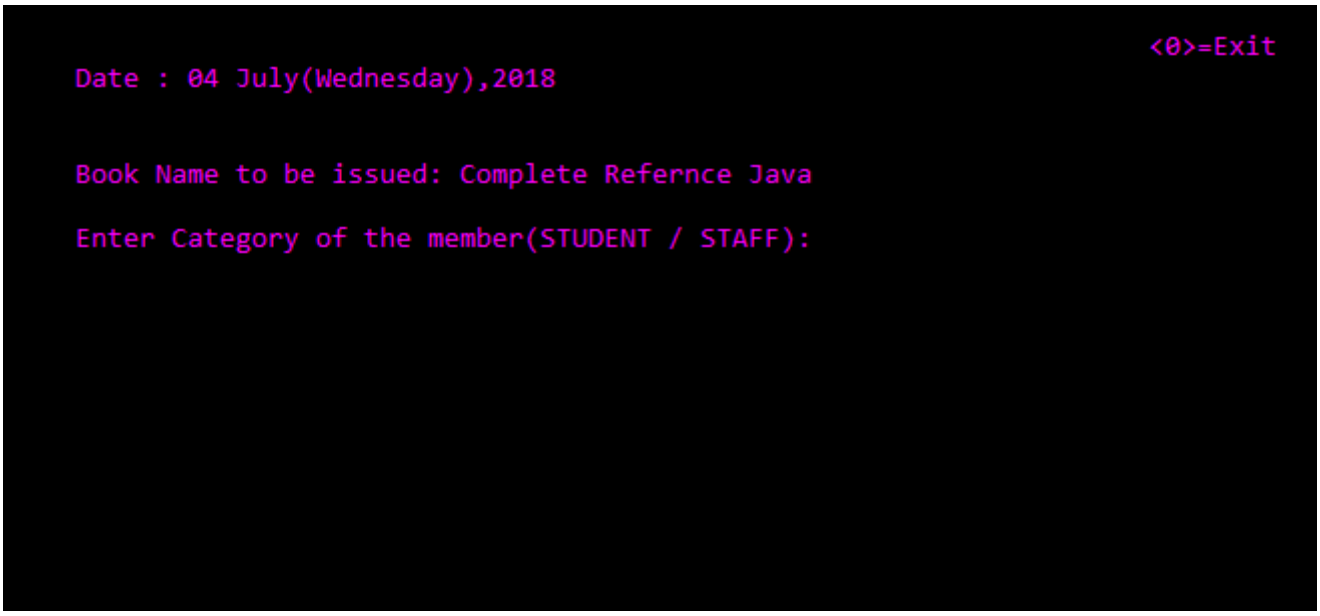
Output Screen:

```
                                        LIST OF BOOKS
                                      ~~~
```

Output Screen1:



Output Screen2:



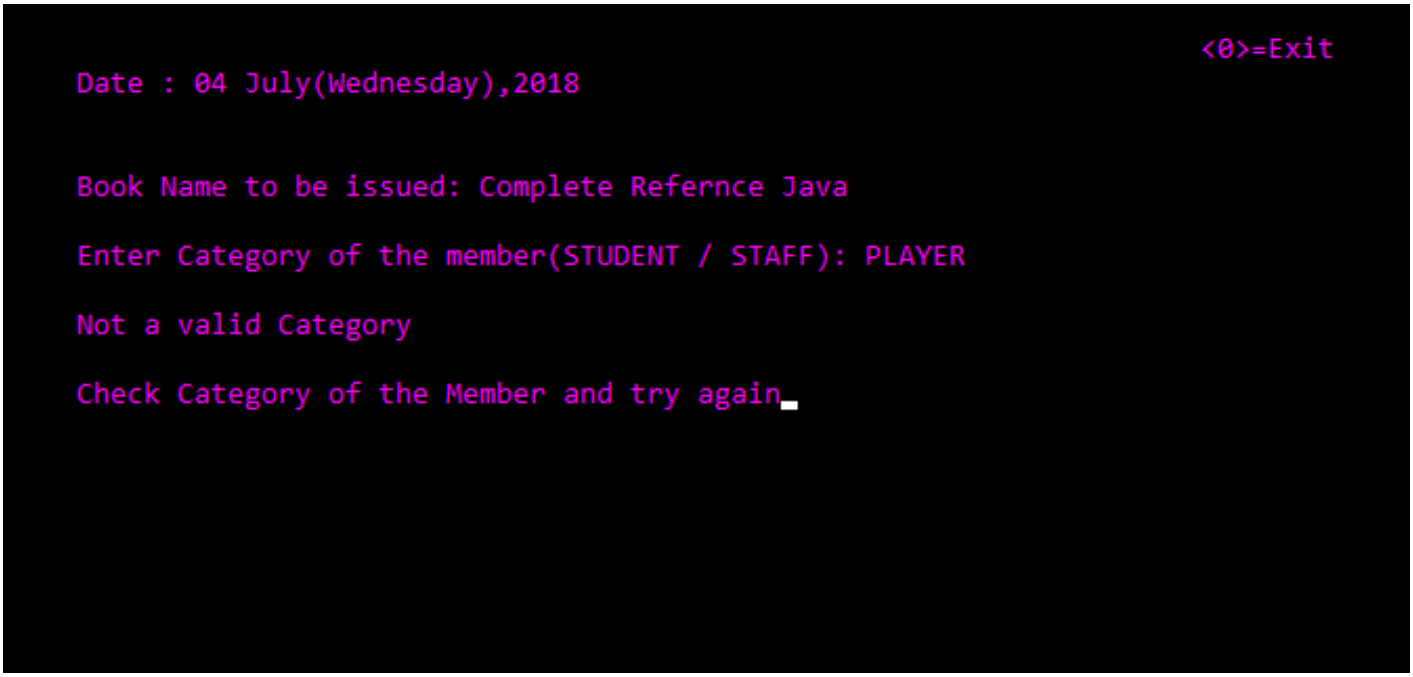
TEST CASE 5 :

System: Book ID has entered correctly (bkID="5678") and promting to accept Member Category.

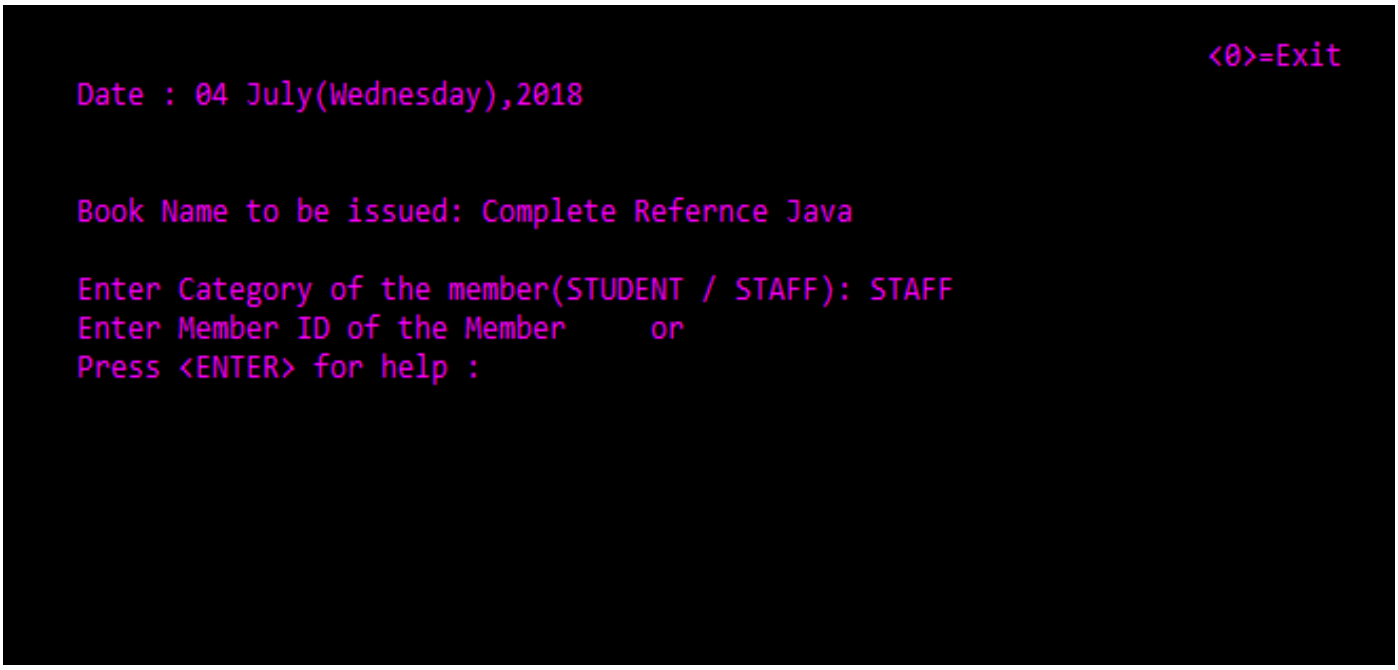
Input: "STUDENT" or "STAFF" (Valid Input)

Output: If the category entered is valid the promting to accept member id otherwise displays Proper messege.

Output Screen1:



Output Screen2:



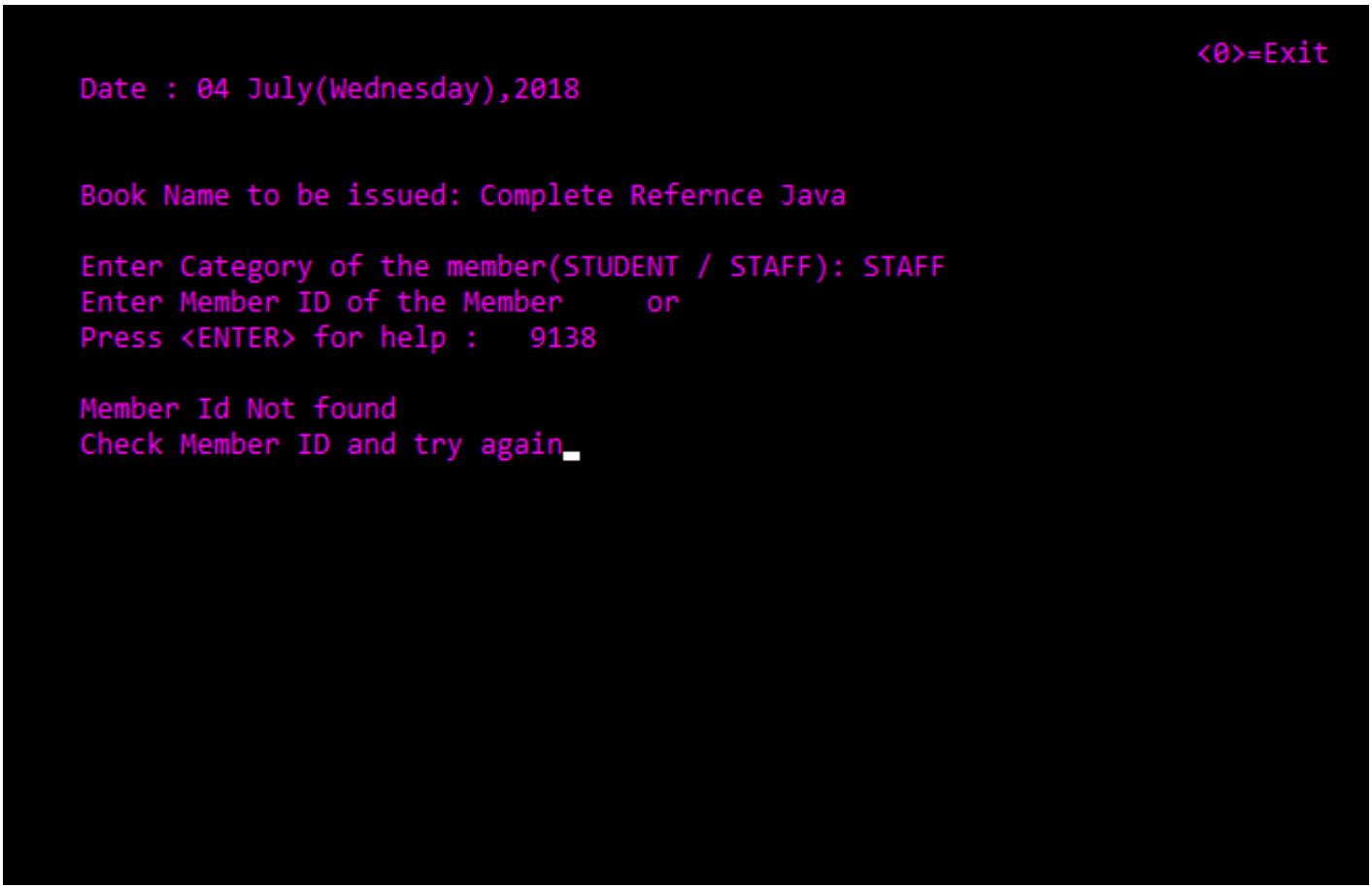
TEST CASE 6 :

System: Book ID has entered correctly (bkID="5678") and Member Category Entered Correctly (STUDENT or STAFF).Now promting to accept Member ID.

Input: 9138 (Invalid Input)

Output: Record not found Message will be displayed.

Output Screen1:



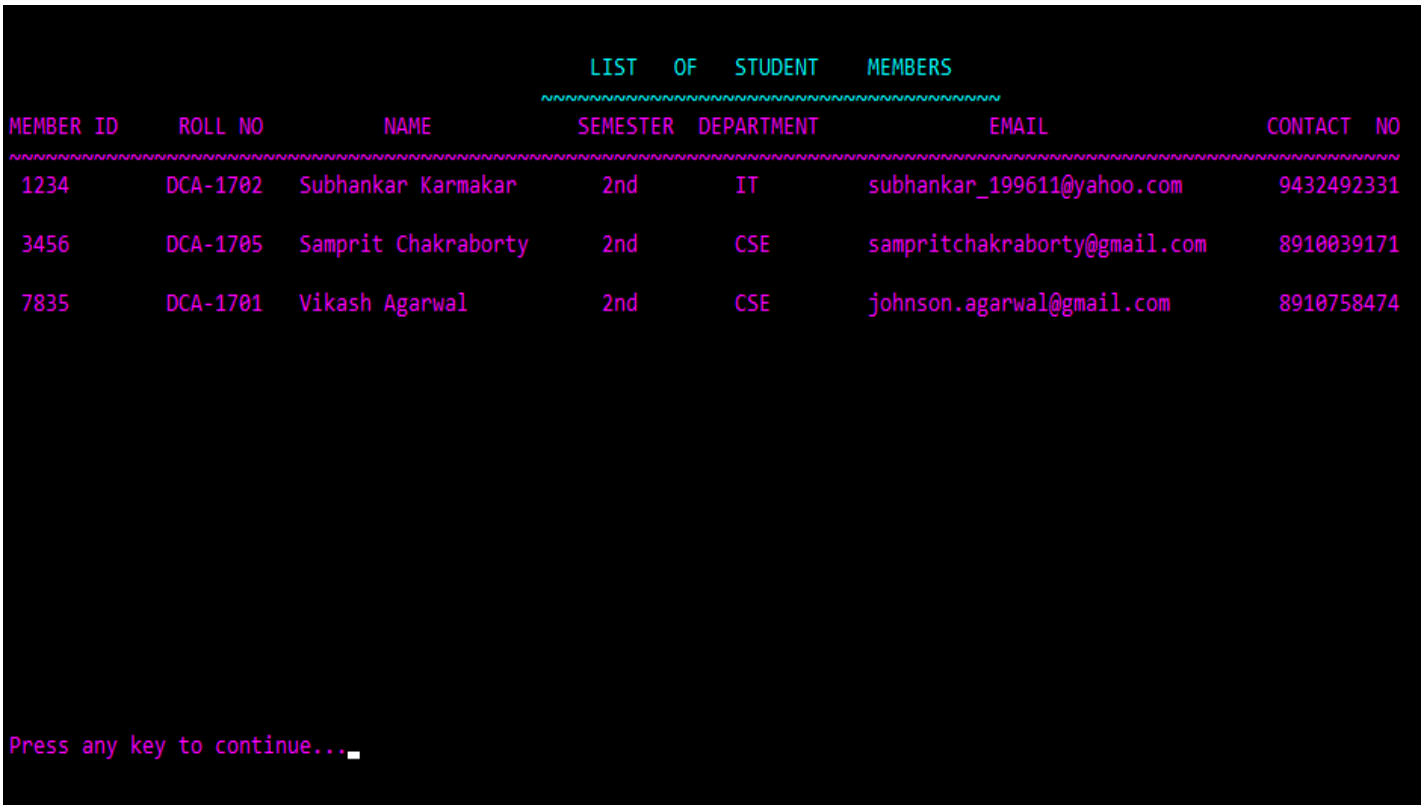
TEST CASE 7 :

System: Book ID has entered correctly (bkID="5678") and Member Category Entered Correctly (STUDENT or STAFF).Now promting to accept Member ID.

Input: "" (Blank Input)

Output: Complete Member List as per entered category will be displayed.

Output Screen:



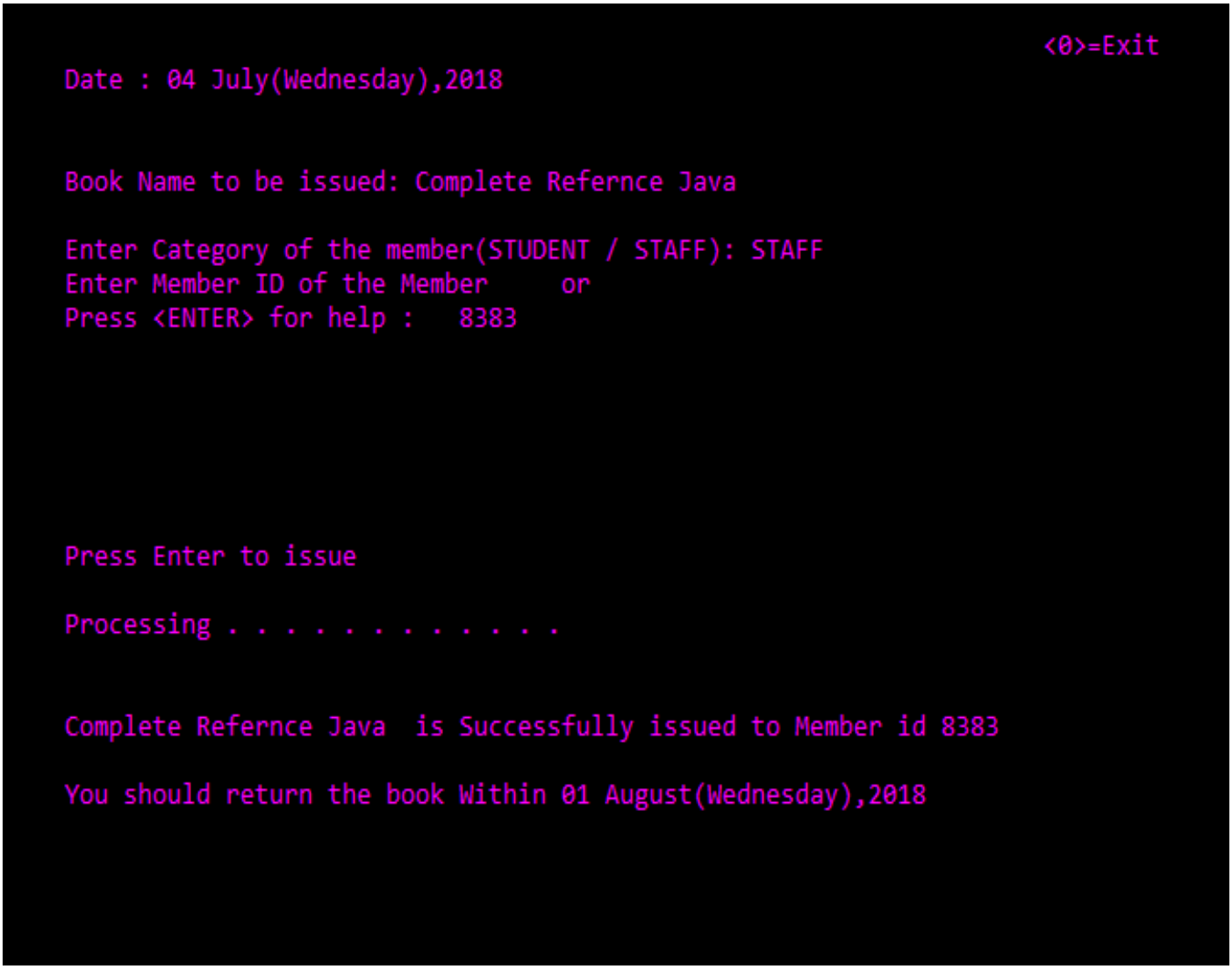
TEST CASE 8 :

System: Book ID has entered correctly (bkID=”5678”) and Member Category Entered Correctly (STUDENT or STAFF).Now promting to accept Member ID.

Input: “8383” (Valid Set)

Output: Successfully issued and corresponding mesege with return date will be displayed.

Output Screen:



5.2 INTEGRATION TESTING

In this type of testing we test various integration of the project module by providing the input .The primary objective is to test the module interfaces in order to ensure that no errors are occurring when one module invokes the other module.

5.3 SYSTEM TESTING TESTING

The aim of the system testing process was to determine all defects in our project .The program was subjected to a set of test inputs and various observations were made and based on these observations it will be decided whether the program behaves as expected or not.Our Project went through two levels of testing

1. Unit testing
2. Integration testing

Chapter 6

CONCLUSION AND FUTURE WORK

This project is fully hard coded by me in C/C++ domain. I have tried my best to make it user friendly as much as possible and implemented it as works like real software. Latter on I will implement mouse programming on this project to make it more user-friendly. Also I wish to convert this library management software in online system using C# and .NET with MVC.

Chapter 7

References

- Fundamentals of software engineering by Rajib Mall, PHI learning
- C++ Complete Reference, Herbert Scheldt
- Programming With C++, John Hubbard
- https://www.geekforgeeks.com
- https://www.tutorialpoint.com