LIBRARY MANAGEMENT

INDEX

1. Certificate
2. Acknowledgement
3. Introduction of project
4. Software Requirement Specification
5. Services provided to the user
6. Architectural Representation
7. Coding
8. Output
9. Future Enhancement
10. Bibliography

CERTIFICATE

This is to certify that the project work **"LIBRARY MANAGEMENT SYSTEM"** is a bonafide record of work done by **Master Shreehari N wadawadagi** under my guidance and supervision of.

Neema Motagi

Basaveshwar International Public School.

ACKNOWLEDGMENT

I am extremely grateful to **Miss Neema Motagi Mca**, Teacher of Department of Computer Science for his able guidance and useful suggestions, which helped me in completing the project work, in time.

I would also like to thank all the teaching and non-teaching staff of Computer Science department who helped me directly or indirectly in the completion of this project.

Finally, yet importantly, I would like to express my heartfelt thanks to my beloved parents for their blessings, my friends/classmates for their help and wishes for the successful completion of this project.

Shreehari N wadawadagi

Introduction

Objective of project: To provide a Library Management System for college library, which would provide all library functions .

Rational: To improve library uses services and reduce paperwork.

Scope of Project:

- To make the existing system more efficient.
- To provide a user friendly environment where user can be serviced better.
- Make functioning of library faster.
- Provide a system where the library staff can catch defaulters and not let them escape.
- To minimize the loss done to books.

Requirements Specification

Functional requirements:

- Accession number, roll number and teacher identification must all be unique as they form the primary keys of the respective tables.
- All new books must be entered in the accession table first, to avoid problems later.
- A book must not be deleted from student profile unless and untill she pays the appropriate fine or the same book.
- While inserting values in the database, only valid values must be entered.

Data requirements:

- The Library Management System shall be required to maintain information about its users and books.
- It shall store databases for students, teachers and books.
- The book database stores information about a book number book title, author, to which class the book belongs to.
- In the same database student or teachers details will also be stored if he/she issues the book.

Software requirements: The system on which the Library Management System is installed must have Microsoft windows in it, and Turbo C software.

Hardware requirements:- Mouse, Keyboard, Monitor

Services provided to the user

The Library Management System automates the basic library functions to aid in the day-to-day operations of a library. It supports functions such as issue, return, the very basic functions of searching for a particular book, etc.
It also maintains data about books about books, teachers, students records that are required during various library operations. The software aims to make the system user friendly and efficient.

The functions that the Library Management System provides are as follows:

1. INSERT: This operation is performed when new data needs to be added to the system, for e.g. when department purchases a new book, the book's entry is inserted in the books database. This option has three choices:
    a) Book: This choice allows entering data about newly purchased books into the books database. The data entered includes book's author, title, publisher, cost and various other fields provided in the form. The data must be accurate and must be entered in the correct format as indicated in the forms.
    b) Teacher: This option is used for entering data for a new teacher in the teacher's database. This option is chosen when a teacher joins college.
    c) Student: This will enter new record for a student in student's database. This option is chosen when a student is enrolled in the college.
2. DELETE: This operation clears the existing records in the various databases. It is used when for e.g. a member leaves college or when book is disposed of from library. But care must be taken while performing this operation and permission taken from the head of library because the system could lose any important data.
    It can be performed on all databases and on three choices are:
    a) Book: This will enter a null value for the book whose accession number is entered in the field provided in the respected form. This operation is done when a book is disposed of the library.
    b) Teacher: This will clear the record for a particular teacher whose identification number is entered. This option is chosen when a teacher leaves college.
    c) Student: This will clear the record for the particular student whose record needs to be deleted by entering her roll number in the required field. This option is chosen when a student leaves college.
3. UPDATE: This function updates data in the various records. This operation is supported by all the three entries:

a) Book: This function generally would not be required for updating a book's status as that data wouldn't change.
b) Teacher: This will update the data of particular teacher, whose data has changed like address, phone number, etc. by entering her identification number.
c) Student: This will update the data of student like address, course, etc. by entering student's roll number.

4. SEARCH: This function is used to search particular data from the database. This function can search for data related to all the three entities:
   a) Book: To search for a particular book, to know whether it is currently available in library or not. This can be done by entering value in any one or more fields in the form to perform the search such as title or author name.
   b) Teacher: This will find out the particular teacher who currently has the book for which search is being carried out.
   c) Student: This will find out the particular student who possesses the particular book.
5. ISSUE: This operation is used for issuing a book to a member of the library. For this operation to be successful the member must meet some criteria like she should not have issued books to her maximum limit previously. All these checks are done by software. If the operation is successful, then the system automatically stores the date of issue and the due date by which the book must be returned.
   a) Student: When a student loans a book, the entry of the book is stored automatically in the student's database with the due date of that book.
   b) Teacher: In case a member of teaching staff loans a book the entry is stored in teacher's database with the due date of the book.
6. RETURN: Using this operation a member returns the items, which she loaned, from the library back to it. If the book, which is loaned is not returned within specified time the member ends up as a defaulter and she is required to pay fine which is calculated automatically by the software.
   a) Teacher: It will delete the corresponding entry made in teacher's database.
   b) Student: It will delete the corresponding entry made in student's database.
7. DISPLAY: This is used to display each and every record, i.e. record of every book, teacher and student in the library.
   a) Book: Record of every book, i.e. it's accession number, author name, publisher name, etc.

b) Teacher: Record of every teacher, i.e. her id, department, no. of books issued, etc., who is member of the college library.
c) Student: Record of every student, i.e. her roll number, course, no of books issued, etc., who is member of the college library.

8. EXIT: This takes user out of the application.

Architectural Representation

- Security options are also imposed by setting the administrator password.
- The real user who know the password can only access all the details
- Following operations can be done in the automation system.
    - Storing the details of the book.
    - Deleting the book.
    - Searching the book by:-
        - Class
        - Book number
        - Availability and non availability
        - Book issue.
        - Book return.
        - Storing the details of the deleted books.
        - Displaying all the books details.

Code

```
#include<iostream.h>
#include<conio.h>
#include<fstream.h>
#include<string.h>
#include<stdlib.h>
#include<iomanip.h>
#include<ctype.h>
#include<stdio.h>
void book_search();
void book_no_search(char bn[]);
void class_search(int cl);
void modifystu(int n);
void book_adoption();
void book_rturn();
void book_delete_records();
void avail_or_non_able(int ch);
fstream ofs;
class entry
{
    public:
    char book_no[7]; int cls,stu_cls,stu_roll;
    char book_name[25],author[15],status[5],stu_name[15];
    public:
    void read();
    void stat();
    void display(int ch);
    char* getbook_no()
    {
            return(book_no);
    }
    char* getstatus()
    {
            return(status);
    }
    int getcls()
    {
            return(cls);
    }
```

```
};

void entry::read()
{
    int count;
    char ans;
    cout<<"are you sure you want enter the details of book?... (y/n)"<<endl;
    cin>>ans;
    if(ans=='y'||ans=='Y')
    {
            cout<<"\n enter the book_no : ";
            cin>>book_no;
            cout<<"\n enter the book_name : ";
            cin>>book_name;
            cout<<"\n enter the author : ";
            cin>>author;
            cout<<"\n enter the class to which the book belongs to : ";
            cin>>cls;
    }
}

void entry :: stat()
{
    strcpy(status,"A");
    strcpy(stu_name,"NAS");
    stu_cls=0;
    stu_roll=0;
}

void entry :: display(int ch)
{
    if(ch==1)
    {

    cout<<setw(5)<<book_no<<setw(10)<<book_name<<setw(10)<<author
<<setw(5)<<cls<<setw(10)<<status<<setw(8)<<stu_name<<setw(8)<<stu_c
ls<<setw(8)<<stu_roll<<endl;
    }
    else
    {
```

```
    cout<<"\n_____
_____\n";
            cout<<"book no  book_name  author  class  status  stu_name
stu_cls  stu_roll\n";

    cout<<"_____
_____\n";
    }
}

entry en;
void book_display()
{
    ofs.open("libraray_management.txt", ios::in|ios::binary);
    ofs.seekg(0,ios::beg);
    en.display(0);
    while(ofs.read((char*)&en,sizeof(entry)))
    {
            en.display(1);
    }
    ofs.close();
getch();
}

void book_delete_records()
{
    ofs.open("delete_records.txt", ios::in|ios::binary);
    ofs.seekg(0,ios::beg);
    en.display(0);
    while(ofs.read((char*)&en,sizeof(entry)))
    {
            en.display(1);
    }
    ofs.close();
getch();
}



void book_deposit()
```

```
{
    char s='y';
    ofs.open("libraray_management.txt", ios::out|ios::binary|ios::app);
    ofs.seekg(0,ios::end);
    while(s=='y'|| s=='Y')
    {
            en.read();
            en.stat();
            cout<<"\n want to enter  more records? (y/n)";
            cin>>s;
            ofs.write((char *)&en, sizeof(entry));

    }
    ofs.close();
getch();
}

void book_adoption()
{
    int tmp=1;
    cout<<"book_adoption\n";
    char f;
    cout<<"are you sure you want 2 take that book....(y/n)\n";
    cin>>f;
    if(f=='y' || f=='Y')
    {
            char bn[7];
            cout<<"enter the book_number\n";
            cin>>bn;
            book_no_search(bn);
            ofs.open("libraray_management.txt", ios::in|ios::binary|ios::out);
            ofs.seekg(0,ios::beg);
            while(ofs.read((char*)&en,sizeof(entry)))
            {

                        if((strcmpi(en.getbook_no(),bn)==0) &&
(strcmpi(en.getstatus(),"A")==0))
                        {

                                modifystu(0);
```

```
                                tmp=0;
                                int pos=-1*sizeof(en);
                                ofs.seekg(pos,ios::cur);
                                ofs.write((char*)&en,sizeof(entry));
                                cout<<"\n\n\t Record Updated";
                                break;
                        }
                }
                if(tmp==1)
                {
                        cout<<"NOT FOUND"<<endl;
                }
        }
        ofs.close();
getch();
}

void modifystu(int n)
{
        if(n==0)
        {
                strcpy(en.status,"NA");
                cout<<"\nenter the student name\n";
                cin>>en.stu_name;
                cout<<"enter the student class\n";
                cin>>en.stu_cls;
                cout<<"enter the roll.no of the student\n";
                cin>>en.stu_roll;
        }
        else
        {
                strcpy(en.status,"A");
                strcpy(en.stu_name,"NAS");
                en.stu_cls=0;
                en.stu_roll=0;
        }
getch();
}
void book_rturn()
{
```

```
    int tmp=1;
    cout<<"book_rturn\n";
    char f;
    cout<<"are you sure you want 2 return that book....(y/n)\n";
    cin>>f;
    if(f=='y' || f=='Y')
    {
            char bn[7];
            cout<<"enter the book_number\n";
            cin>>bn;
            book_no_search(bn);
            ofs.open("libraray_management.txt", ios::in|ios::binary|ios::out);
            ofs.seekg(0,ios::beg);
            while(ofs.read((char*)&en,sizeof(entry)))
            {

                        if((strcmpi(en.getbook_no(),bn)==0) &&
(strcmpi(en.getstatus(),"NA")==0))
                        {

                                modifystu(1);
                                tmp=0;
                                int pos=-1*sizeof(en);
                                ofs.seekg(pos,ios::cur);
                                ofs.write((char*)&en,sizeof(entry));
                                cout<<"\n\n\t Record Updated";
                                break;
                         }
            }
            if(tmp==1)
            {
                    cout<<"NOT FOUND"<<endl;
            }
    }
    ofs.close();
getch();
}
```

```
void book_delete()
{
    char bn[7];int tmp=1;
    clrscr();
    cout<<"\n\n\n\tDELETE BOOK ...";
    cout<<"\nenter the book_no that you want to delete\n";
    cin>>bn;
    ofs.open("libraray_management.txt",ios::in|ios::out|ios::binary);
    fstream fp2,fp1;
    fp2.open("temp.txt",ios::out|ios::binary);
    fp1.open("delete_records.txt",ios::out|ios::binary|ios::app);
    ofs.seekg(0,ios::beg);
    while(ofs.read((char*)&en,sizeof(entry)))
    {
            if((strcmpi(en.getbook_no(),bn)==0) &&
(strcmpi(en.getstatus(),"A")==0))
            {
                    fp1.seekg(0,ios::end);
                    fp1.write((char*)&en,sizeof(entry));
                    tmp=0;
            }
            else
            {
                    fp2.write((char*)&en,sizeof(entry));
            }
    }
    if(tmp==1)
    {
            cout<<"RECORD NOT FOUND\n";
    }
    else
    {
            cout<<"\n\n\tRecord Deleted ..";
    }

fp2.close();
ofs.close();
fp1.close();
remove("libraray_management.txt");
```

```
rename("temp.txt","libraray_management.txt");
getch();
}

void book_search()
{
    cout<<"book_search\n";
    int ch;
    cout<<"enter 1 for book no. search\n";
    cout<<"enter 2 for class search\n";
    cout<<"enter 3 for available books\n";
    cout<<"enter 4 for non_available books\n";
    cin>>ch;
    switch(ch)
    {
            case 1:
            char bn[7];
            cout<<"enter the book_number\n";
            cin>>bn;
            book_no_search(bn);
            break;
            case 2:
            int cl;
            cout<<"enter the class\n";
            cin>>cl;
            class_search(cl);
            case 3:
            avail_or_non_able(1);
            break;
            case 4:
            avail_or_non_able(0);
            break;
            default: cout<<"invalid entry";
    }
}

void book_no_search(char bn[])
{
    int temp=1;
    ofs.open("libraray_management.txt", ios::in|ios::binary);
```

```
    ofs.seekg(0,ios::beg);
    en.display(0);
    while(ofs.read((char *)&en,sizeof(entry)))
    {

            if(strcmpi(en.getbook_no(),bn)==0)
            {
                    en.display(1);
                    temp=0;
            }
    }
    if(temp==1)
    {
            cout<<"NOT FOUND\n";

    }
    ofs.close();
}

void class_search(int cl)
{
    int temp=1;
    en.cls=0;
    ofs.open("libraray_management.txt", ios::in|ios::binary);
    ofs.seekg(0,ios::beg);
    en.display(0);
    while(ofs.read((char*)&en,sizeof(entry)))
    {
            if(en.getcls()==cl)
            {
                    en.display(1);
                    temp=0;
            }
    }
    if(temp==1)
    {
            cout<<"NOT FOUND\n";

    }
    ofs.close();
```

```
}


void main()
{
    clrscr();
    int choice;
    char ans='y',password[5];
    strcpy(password,getpass("\n\tenter the administrator password\n\t"));
    if(strcmp(password,"hari")==0)
    {

        while(ans=='y'||ans=='Y')
        {     clrscr();
              cout<<"1.book deposit\n";
              cout<<"2.book adoption\n";
              cout<<"3.book search\n";
              cout<<"4.book delete\n";
              cout<<"5.book display\n";
              cout<<"6.book return\n";
              cout<<"7.book delete rcords\n";
              cout<<"8.exit\n";
              cin>>choice;
              switch(choice)
              {
                    case 1:clrscr();
                    book_deposit();
                    break;
                    case 2:clrscr();
                    book_adoption();
                    break;
                    case 3:clrscr();
                    book_search();
                    break;
                    case 4:clrscr();
                    book_delete();
                    break;
                    case 5:clrscr();
                    book_display();
```

```
                        break;
                        case 6:clrscr();
                        book_rturn();
                        break;
                        case 7: clrscr();
                        book_delete_records();
                        break;
                        case 8:exit(0);
                        break;
                        default:cout<<"\ninvalid choice\n";
                }
        cout<<"\nwant to continue.....(y/n)\n" ;
        cin>>ans;
        }
    }
    else
    {
    cout<<"login denied";
    }
ofs.close();
getch();
}

void avail_or_non_able(int ch)
{
    int temp=1;
    if(ch==1)
    {
        ofs.open("libraray_management.txt", ios::in|ios::binary);
        ofs.seekg(0,ios::beg);
        en.display(0);
        while(ofs.read((char*)&en,sizeof(entry)))
        {
            if(strcmpi(en.getstatus(),"A")==0)
            {
                en.display(1);
                temp=0;
            }
        }
        if(temp==1)
```

```
            {
            cout<<"NOT FOUND\n";

            }
        }
        else
        {
            ofs.open("libraray_management.txt", ios::in|ios::binary);
            ofs.seekg(0,ios::beg);
            en.display(0);
            while(ofs.read((char*)&en,sizeof(entry)))
            {
                if(strcmpi(en.getstatus(),"NA")==0)
                {
                    en.display(1);
                    temp=0;
                }
            }
            if(temp==1)
            {
                cout<<"NOT FOUND\n";
            }
            ofs.close();
        }
    }
```

Future enhancement

- This project is prepared using C++ language without using any graphical user interaction.
- As more interactive program is more user friendly, by using Graphical User Interface this project can be made more user friendly.
- The use of validation rules improves the efficiency of project.
- Efficiency and performance can be increased by using the few more parameters of the book.
- At last if it is connected to the main database of the school.
- The project can be implemented in schools and colleges.

Bibliography

1. Let Us C by Yashavant Kanetkar
2. Computer Science, C++ by Sumita Arora
3. The Complete Reference, C++ by Herbert Schildt
4. Software Engineering by Roger S. Pressman