1. Project description

Library management is a sub-discipline of institutional management that focuses on specific issues faced by libraries and library management professionals.

The purpose of the application is automation of library ,it provide facilities to student or member to search for the required books and it allows the administrator or librarian to Issue & return books to students and can create & delete membership of students.

The main aim of this project is to implement an application which deals with maintaining Library activities like Book issues, book returns and Administration activities.This system will reduce manual work for maintaining records in files.

Regular transactions which include book issue, returns etc. and exceptional transactions that are related to loss of books, damage of books etc. also will have to be handled by the system.Through our project user can add members, add books, search members, search books,update information, edit information, borrow and return books in quick time. Our proposed system has the following advantages.

User friendly interface

Fast access to database

Less error

More Storage Capacity

Search facility

Look and Feel Environment

Quick transaction

All the manual difficulties in managing the Library have been rectified by implementing computerization.

Our project is only a humble venture to satisfy the needs in a library. Several user friendly coding have also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the organization.

Platform requirements

Hardware/ Software	Hardware / Software element	Specification /version
Hardware	Processor	Intel core to duo
	RAM	1 GB
	Hard Disk	100 GB
Software	OS	Windows XP
	C++ related IDE	Turbo C++

Table 1.1 Platform Requirements

Module Identification

Following are the modules that comprise the system

Serial Number	Name of the Module
1.	Book Issue Management Module
2.	Book Returns Management Module
3.	Administration Management Module
4.	Student Record and Book Records Module

Table 1.2 Module Identification

1.1 Book Issue Management Module:

In this module we use issue operation 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. 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.

1.2 Book Returns Management Module:

In this module we use Deposit Operation 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. It will delete the corresponding entry made in student's database.

1.3 Administration Management Module :

The functions that the Administartion 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) 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 mustbe taken while performing this operation and permission taken from the head of library because the system could loose any important data.

It can be performed on all databases and on two 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)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 two enteries:

a)Book: This function generally would not be required for updating a book's status as that data wouldn't change.

b)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 two 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)Student: This will find out the particular student who possesses the particular book.

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

1.4 Student Record and Book Records Module:

In this Module we use Display Operation 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)Student: Record of every student, i.e. her roll number, course, no of books issued, etc., who is member of the college library.

The purpose of this project is to make transaction of books easy for the students and also for librarian to store the details of the book.

Our project is only a humble venture to satisfy the needs in a library. Several user friendly coding have also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the organization.

Concepts Used for this Project :

Classes,Functions,Files.

Opening a File:

A file must be opened before you can read from it or write to it. Either the ofstream or fstream object may be used to open a file for writing and ifstream object is used to open a file for reading purpose only.
Following is the standard syntax for open() function, which is a member of fstream, ifstream, and ofstream objects.
Syntax : void open(const char *filename, ios::openmode mode);

Closing a File:

When a C++ program terminates it automatically closes flushes all the streams, release all the allocated memory and close all the opened files. But it is always a good practice that a programmer should close all the opened files before program termination.
Following is the standard syntax for close() function, which is a member of fstream, ifstream, and ofstream objects.

void close();

Writing to a File:

While doing C++ programming , you write information to a file from your program using the stream insertion operator (<<) just as you use that operator to output information to the screen. The only difference is that you use an ofstream or fstream object instead of the c out object.

Reading from a File:

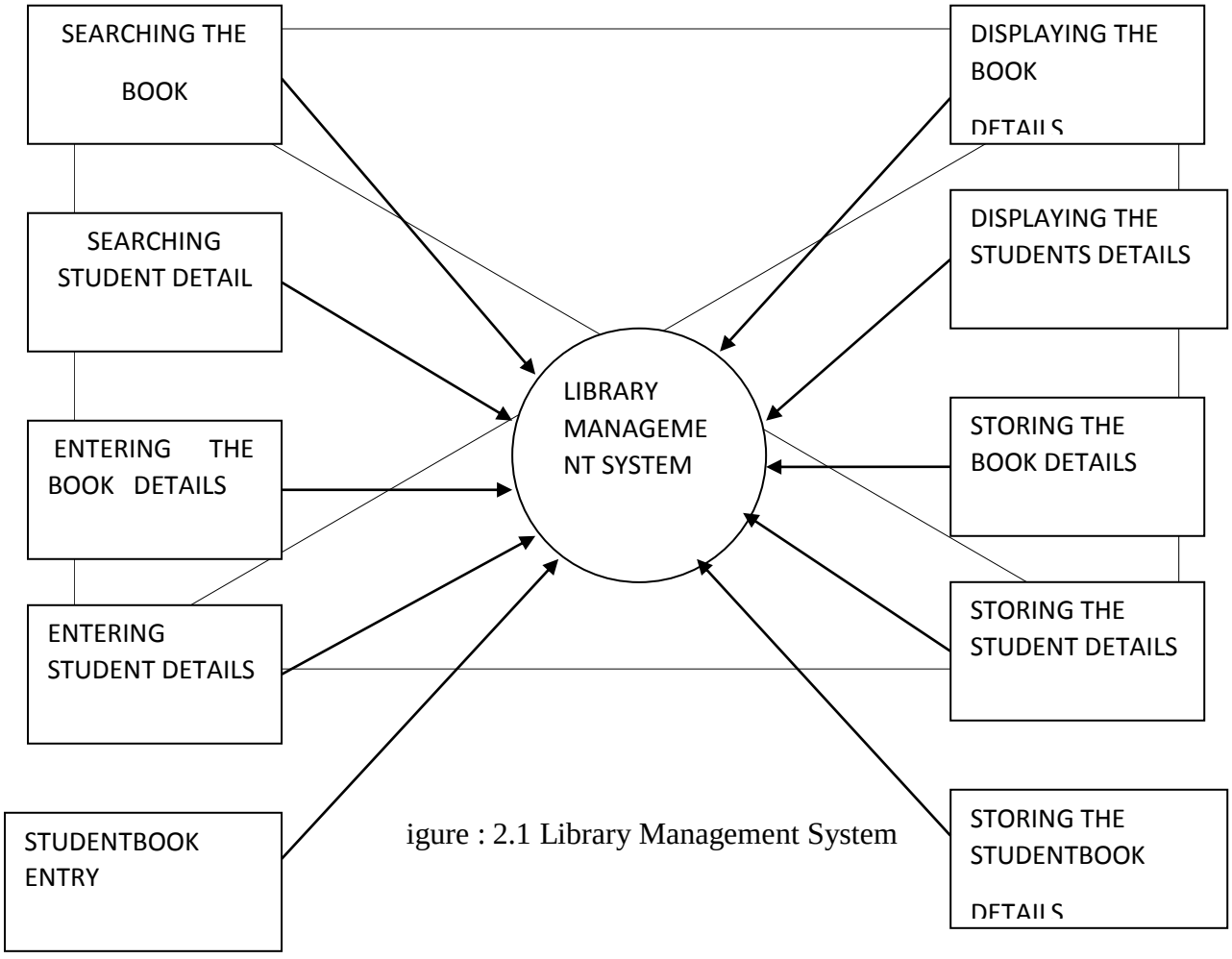
You read information from a file into your program using the stream extraction operator (>>) just as you use that operator to input information from the keyboard. The only difference is that you use an ifstream or fstream object instead of the c in object.

2. DESIGN:

LEVEL 0 DFD DIAGRAM:

In Level 0 DFD diagram, it will display the menus of the project that is what are all we can do in the project. From that we can select the option what we are going to do. Based upon the condition it display the next screen for the selected operation.

LEVEL O DFD DIAGRAM



igure : 2.1 Library Management System

LEVEL 1 DFD DIAGRAM:

In Level 1 DFD diagram it takes the input details and store the details in the database. It takes the details separately for each table and store details separately in each table.

TO STORE THE DETAILS OF MEMBERS:

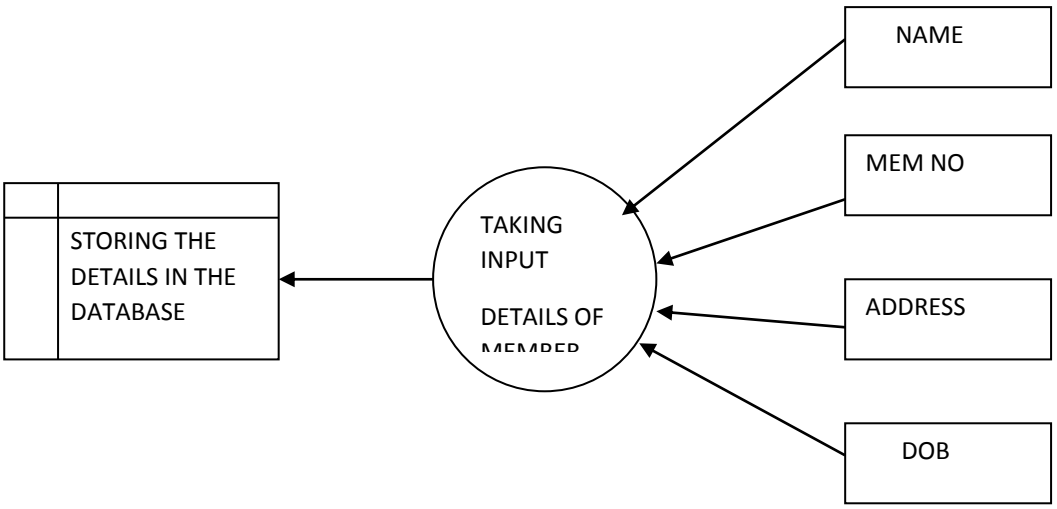


Figure 2.2 Storing Details of Members

TO STORE THE DETAILS OF BOOKS:

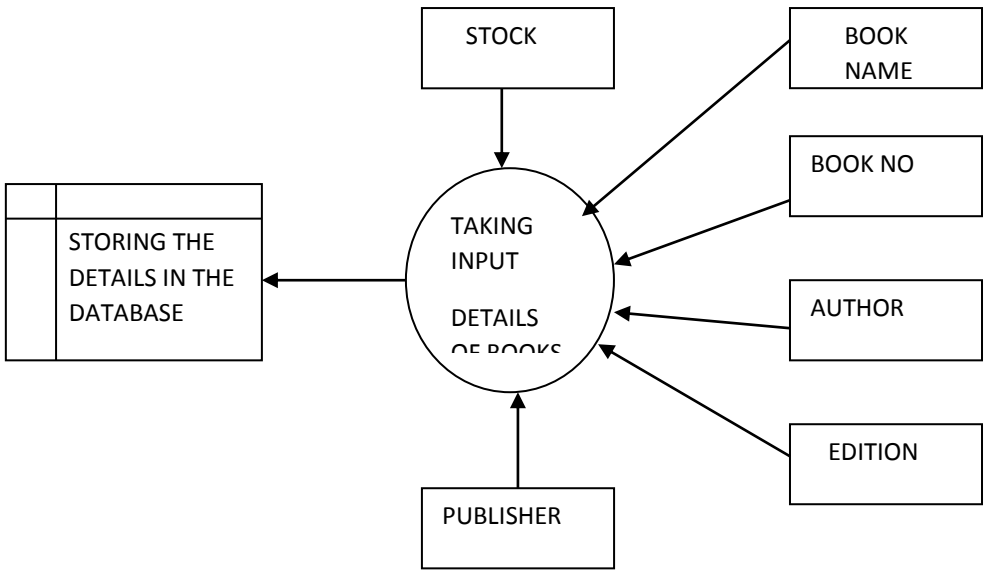


Figure 2.3 Storing Details of Books

TO STORE THE DETAILS OF TRANSACTIONS

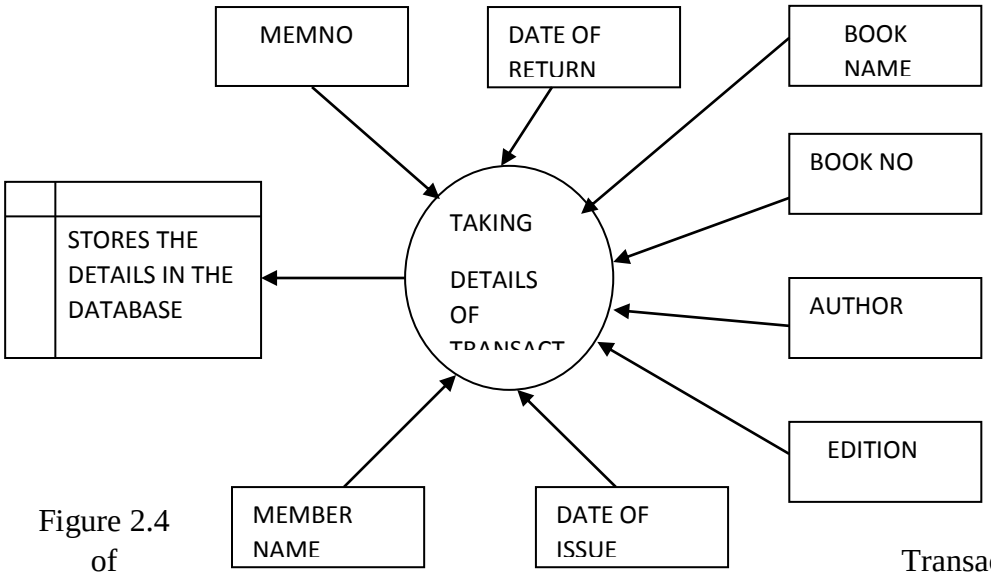


Figure 2.4 Storing Details of Transactions

LEVEL 2 DFD DIAGRAM:

In Level 2 DFD diagram it displays the selected details or the details of all books present in the library and the details of selected members or all the members who are having the membership in the library and the transaction details of members who have taken books.

DISPLAY THE MEMBER DETAILS:

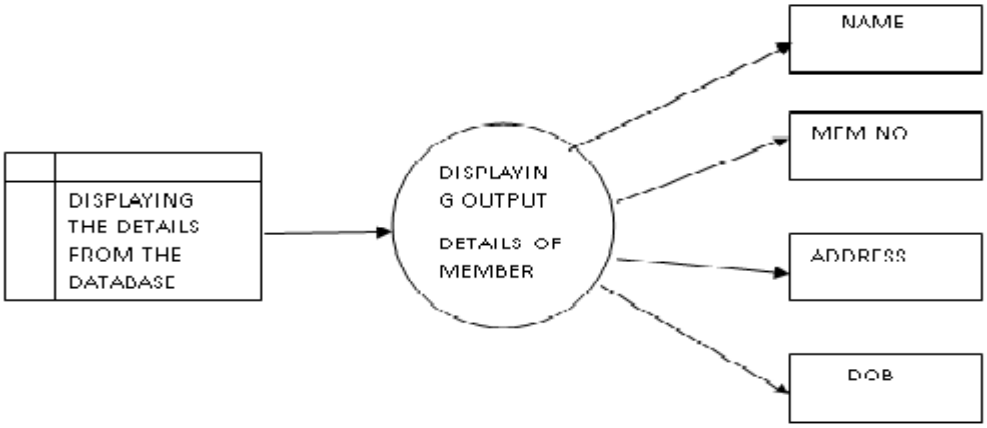


Figure 2.5 Displaying Member Details

TO DISPLAY THE DETAILS OF BOOKS:

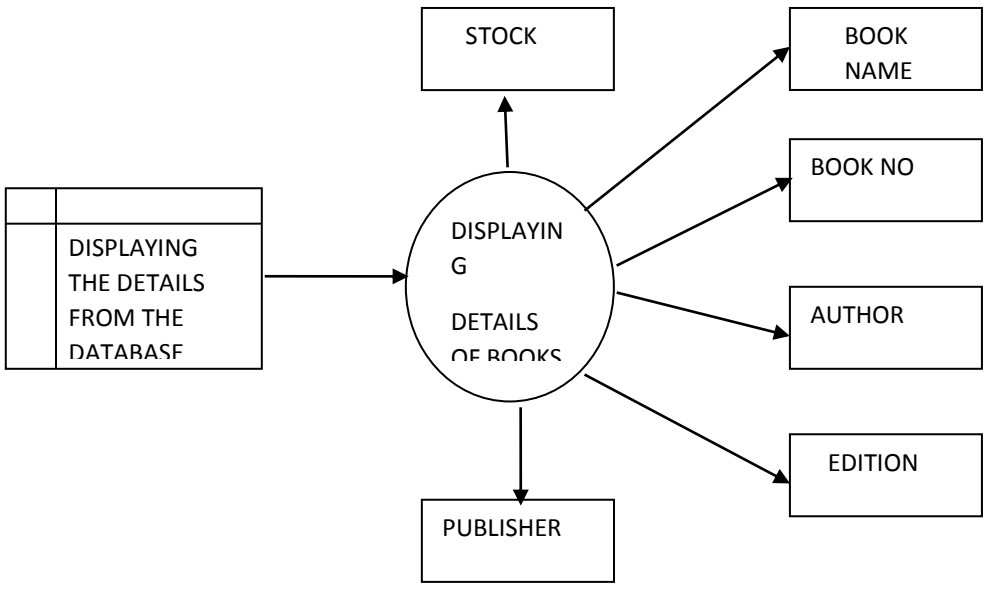


Figure 2.6 Displaying Details of Books

DISPLAY THE DETAILS OF TRANSACTIONS

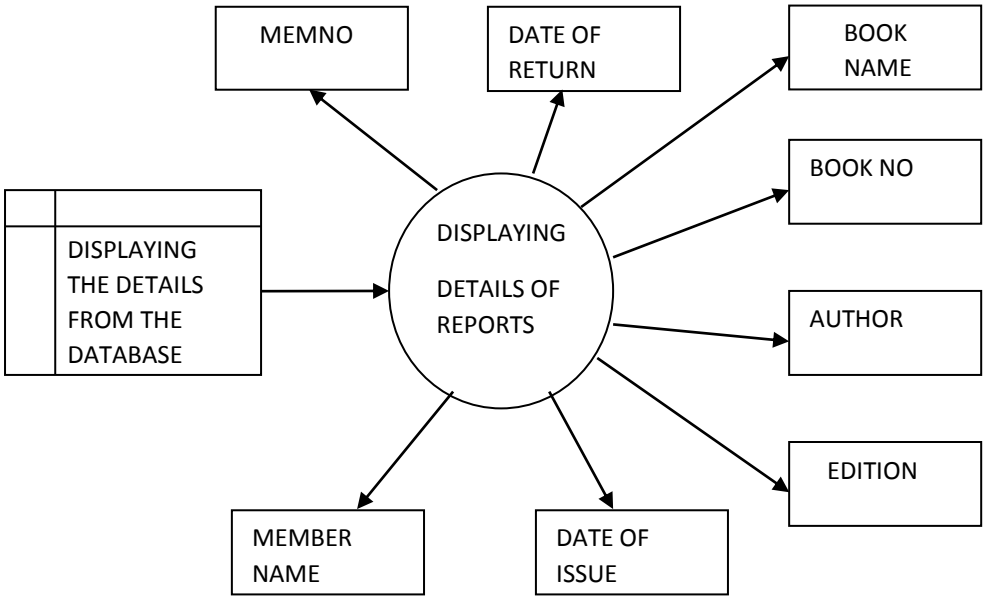


Figure 2.7 Displaying Details of transactions

Overview

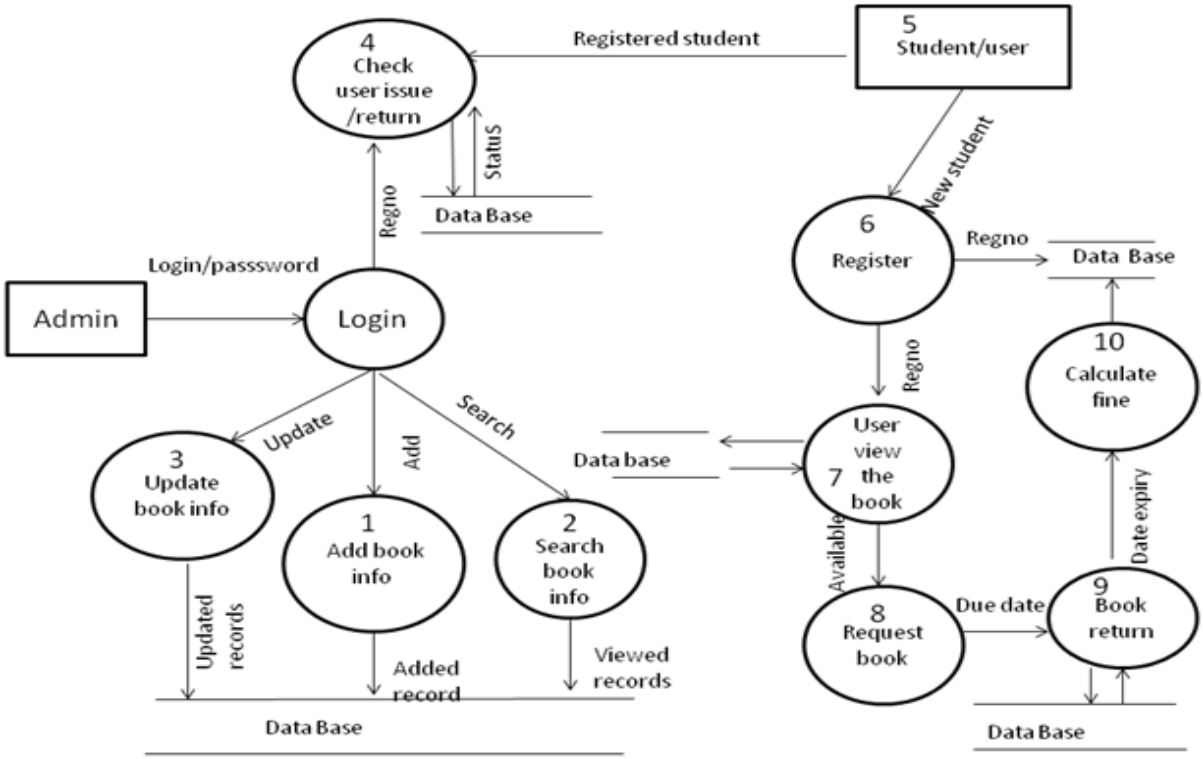


Figure 2.8 Overview of Library Management System

3. Source Code:

```
#include<fstream.h>
#include<conio.h>
#include<stdio.h>
#include<process.h>
#include<string.h>
#include<iomanip.h>
class book
{
        char bno[6];
        char bname[50];
        char aname[20];
  public:
        void create_book()
        {
                cout<<"\nNEW BOOK ENTRY...\n";
                cout<<"\nEnter The book no.";
                cin>>bno;
                cout<<"\n\nEnter The Name of The Book ";
                gets(bname);
                cout<<"\n\nEnter The Author's Name ";
                gets(aname);
                cout<<"\n\n\nBook Created..";
        }

        void show_book()
        {
                cout<<"\nBook no. : "<<bno;
                cout<<"\nBook Name : ";
                puts(bname);
                cout<<"Author Name : ";
```

```
                puts(aname);
        }

        void modify_book()
        {
                cout<<"\nBook no. : "<<bno;
                cout<<"\nModify Book Name : ";
                gets(bname);
                cout<<"\nModify Author's Name of Book : ";
                gets(aname);
        }

        char* retbno()
        {
                return bno;
        }

        void report()
        {cout<<bno<<setw(30)<<bname<<setw(30)<<aname<<endl;}


};      //class ends here
class student
{
        char admno[6];
        char name[20];
        char stbno[6];
        int token;
public:
        void create_student()
        {
                clrscr();
```

```
        cout<<"\nNEW STUDENT ENTRY...\n";
        cout<<"\nEnter The admission no. ";
        cin>>admno;
        cout<<"\n\nEnter The Name of The Student ";
        gets(name);
        token=0;
        stbno[0]='/0';
        cout<<"\n\nStudent Record Created..";
}

void show_student()
{
        cout<<"\nAdmission no. : "<<admno;
        cout<<"\nStudent Name : ";
        puts(name);
        cout<<"\nNo of Book issued : "<<token;
        if(token==1)
                cout<<"\nBook No "<<stbno;
}

void modify_student()
{
        cout<<"\nAdmission no. : "<<admno;
        cout<<"\nModify Student Name : ";
        gets(name);
}

char* retadmno()
{
        return admno;
}
```

```
        char* retstbno()
        {
                return stbno;
        }

        int rettoken()
        {
                return token;
        }

        void addtoken()
        {token=1;}

        void resettoken()
        {token=0;}

        void getstbno(char t[])
        {
                strcpy(stbno,t);
        }

        void report()
        {cout<<"\t"<<admno<<setw(20)<<name<<setw(10)<<token<<endl;}

};      //class ends here
fstream fp,fp1;
book bk;
student st;

void write_book()
{
        char ch;
```

```
	fp.open("book.dat",ios::out|ios::app);
	do
	{
		clrscr();
		bk.create_book();
		fp.write((char*)&bk,sizeof(book));
		cout<<"\n\nDo you want to add more record..(y/n?)";
		cin>>ch;
	}while(ch=='y'||ch=='Y');
	fp.close();
}

void write_student()
{
	char ch;
	fp.open("student.dat",ios::out|ios::app);
	do
	{
		st.create_student();
		fp.write((char*)&st,sizeof(student));
		cout<<"\n\ndo you want to add more record..(y/n?)";
		cin>>ch;
	}while(ch=='y'||ch=='Y');
	fp.close();
}
void display_spb(char n[])
{
	cout<<"\nBOOK DETAILS\n";
	int flag=0;
	fp.open("book.dat",ios::in);
	while(fp.read((char*)&bk,sizeof(book)))
	{
```

```
                if(strcmpi(bk.retbno(),n)==0)
                {
                        bk.show_book();
                        flag=1;
                }
        }

        fp.close();
        if(flag==0)
                cout<<"\n\nBook does not exist";
        getch();
0}

void display_sps(char n[])
{
        cout<<"\nSTUDENT DETAILS\n";
        int flag=0;
        fp.open("student.dat",ios::in);
        while(fp.read((char*)&st,sizeof(student)))
        {
                if((strcmpi(st.retadmno(),n)==0))
                {
                        st.show_student();
                        flag=1;
                }
        }

        fp.close();
        if(flag==0)
                cout<<"\n\nStudent does not exist";
        getch();
}
```

```
void modify_book()
{
        char n[6];
        int found=0;
        clrscr();
        cout<<"\n\n\tMODIFY BOOK REOCORD.... ";
        cout<<"\n\n\tEnter The book no. of The book";
        cin>>n;
        fp.open("book.dat",ios::in|ios::out);
        while(fp.read((char*)&bk,sizeof(book)) && found==0)
        {
                if(strcmpi(bk.retbno(),n)==0)
                {
                        bk.show_book();
                        cout<<"\nEnter The New Details of book"<<endl;
                        bk.modify_book();
                        int pos=-1*sizeof(bk);
                        fp.seekp(pos,ios::cur);
                        fp.write((char*)&bk,sizeof(book));
                        cout<<"\n\n\t Record Updated";
                        found=1;
                }
        }

        fp.close();
        if(found==0)
                cout<<"\n\n Record Not Found ";
        getch();
}
void modify_student()
{
        char n[6];
```

```
        int found=0;
        clrscr();
        cout<<"\n\n\tMODIFY STUDENT RECORD... ";
        cout<<"\n\n\tEnter The admission no. of The student";
        cin>>n;
        fp.open("student.dat",ios::in|ios::out);
        while(fp.read((char*)&st,sizeof(student)) && found==0)
        {
                if(strcmpi(st.retadmno(),n)==0)
                {
                        st.show_student();
                        cout<<"\nEnter The New Details of student"<<endl;
                        st.modify_student();
                        int pos=-1*sizeof(st);
                        fp.seekp(pos,ios::cur);
                        fp.write((char*)&st,sizeof(student));
                        cout<<"\n\n\t Record Updated";
                        found=1;
                }
        }

        fp.close();
        if(found==0)
                cout<<"\n\n Record Not Found ";
        getch();
}

void delete_student()
{
        char n[6];
        int flag=0;
        clrscr();
```

```
        cout<<"\n\n\n\tDELETE STUDENT...";
        cout<<"\n\nEnter The admission no. of the Student You Want To Delete : ";
        cin>>n;
        fp.open("student.dat",ios::in|ios::out);
        fstream fp2;
        fp2.open("Temp.dat",ios::out);
        fp.seekg(0,ios::beg);
        while(fp.read((char*)&st,sizeof(student)))
        {
                if(strcmpi(st.retadmno(),n)!=0)
                        fp2.write((char*)&st,sizeof(student));
                else
                        flag=1;
        }

        fp2.close();
        fp.close();
        remove("student.dat");
        rename("Temp.dat","student.dat");
        if(flag==1)
                cout<<"\n\n\tRecord Deleted ..";
        else
                cout<<"\n\nRecord not found";
        getch();
}

void delete_book()
{
        char n[6];
        clrscr();
        cout<<"\n\n\n\tDELETE BOOK ...";
        cout<<"\n\nEnter The Book no. of the Book You Want To Delete : ";
```

```
        cin>>n;
        fp.open("book.dat",ios::in|ios::out);
        fstream fp2;
        fp2.open("Temp.dat",ios::out);
        fp.seekg(0,ios::beg);
        while(fp.read((char*)&bk,sizeof(book)))
        {
                if(strcmpi(bk.retbno(),n)!=0)
                {
                        fp2.write((char*)&bk,sizeof(book));
                }
        }

        fp2.close();
        fp.close();
        remove("book.dat");
        rename("Temp.dat","book.dat");
        cout<<"\n\n\tRecord Deleted ..";
        getch();
}
void display_alls()
{
        clrscr();
        fp.open("student.dat",ios::in);
        if(!fp)
        {
                cout<<"ERROR!!! FILE COULD NOT BE OPEN ";
                getch();
                return;
        }

        cout<<"\n\n\t\tSTUDENT LIST\n\n";
```

```
        cout<<"=============================================================
===========\n";
        cout<<"\tAdmission No."<<setw(10)<<"Name"<<setw(20)<<"Book Issued\n";
        cout<<"=============================================================
===========\n";

        while(fp.read((char*)&st,sizeof(student)))
        {
                st.report();
        }

        fp.close();
        getch();
}
void display_allb()
{
        clrscr();
        fp.open("book.dat",ios::in);
        if(!fp)
        {
                cout<<"ERROR!!! FILE COULD NOT BE OPEN ";
                getch();
                return;
        }

        cout<<"\n\n\t\tBook LIST\n\n";

        cout<<"Book Number"<<setw(20)<<"Book Name"<<setw(25)<<"Author\n";

       while(fp.read((char*)&bk,sizeof(book)))
        {
                bk.report();
```

```
        }
        fp.close();
        getch();
}
void book_issue()
{
        char sn[6],bn[6];
        int found=0,flag=0;
        clrscr();
        cout<<"\n\nBOOK ISSUE ...";
        cout<<"\n\n\tEnter The student's admission no.";
        cin>>sn;
        fp.open("student.dat",ios::in|ios::out);
        fp1.open("book.dat",ios::in|ios::out);
        while(fp.read((char*)&st,sizeof(student)) && found==0)
        {
                if(strcmpi(st.retadmno(),sn)==0)
                {
                        found=1;
                        if(st.rettoken()==0)
                        {
                                cout<<"\n\n\tEnter the book no. ";
                                cin>>bn;
                                while(fp1.read((char*)&bk,sizeof(book))&& flag==0)
                                {
                                        if(strcmpi(bk.retbno(),bn)==0)
                                        {
                                                bk.show_book();
                                                flag=1;
                                                st.addtoken();
                                                st.getstbno(bk.retbno());
                                                int pos=-1*sizeof(st);
```

```
                                        fp.seekp(pos,ios::cur);
                                        fp.write((char*)&st,sizeof(student));
                                        cout<<"\n\n\tBook issued successfully\n\nPlease
Note: Write the current date in backside of your book and submit within 15 days fine Rs. 1
for each day                                                    after 15 days period";
                                }
                        }
                        if(flag==0)
                                cout<<"Book no does not exist";
                }
                else
                        cout<<"You have not returned the last book ";

        }
    }
    if(found==0)
        cout<<"Student record not exist...";
    getch();
    fp.close();
    fp1.close();
}
void book_deposit()
{
   char sn[6],bn[6];
   int found=0,d,flag=0,day,fine;
   clrscr();
   cout<<"\n\nBOOK DEPOSIT ...";
   cout<<"\n\n\tEnter The student's admission no.";
   cin>>sn;
   fp.open("student.dat",ios::in|ios::out);
   fp1.open("book.dat",ios::in|ios::out);
   while(fp.read((char*)&st,sizeof(student)) && found==0)
```

```
{
        if(strcmpi(st.retadmno(),sn)==0)
        {
                found=1;
                if(st.rettoken()==1)
                {
                      while(fp1.read((char*)&bk,sizeof(book))&& flag==0)
                      {
                        if(strcmpi(bk.retbno(),st.retstbno())==0)
                      {
                              bk.show_book();
                              flag=1;
                              cout<<"\n\nBook deposited in no. of days";
                              cin>>day;cout<<\n\ncheck whether the book is damaged\n\nif
it is damaged click 1 else 0;cin>>d;
                              if(day>15)
                              {
                                fine=(day-15)*1;
                                cout<<"\n\nFine has to deposited Rs. "<<fine;
                              }
             if(d==1){cout<<"\n\npay double the amount of respective book";


                                      st.resettoken();
                                      int pos=-1*sizeof(st);
                                      fp.seekp(pos,ios::cur);
                                      fp.write((char*)&st,sizeof(student));
                                      cout<<"\n\n\t Book deposited successfully";
                      }
                }
               if(flag==0)
                cout<<"Book no does not exist";
```

```
                    }
                else
                        cout<<"No book is issued..please check!!";
                }
            }
    if(found==0)
        cout<<"Student record not exist...";
        getch();
 fp.close();
 fp1.close();
 }
void intro()
{
        clrscr();
        gotoxy(35,5);
        cout<<"LIBRARY";
        gotoxy(35,8);
        cout<<"MANAGEMENT";
        gotoxy(35,11);
        cout<<"SYSTEM";
        cout<<"\n\nMADE BY :1. Av Manikanta      2.N Yuva Srikanth      3.Yasasvini
4.Latha Sai Sri";
        cout<<"\n\nCOLLEGE :KL UNIVERSITY ";
        getch();
}


void admin_menu()
{
        clrscr();
        int ch2;
        cout<<"\n\n\n\tADMINISTRATOR MENU";
```

```
cout<<"\n\n\t1.CREATE STUDENT RECORD";
cout<<"\n\n\t2.DISPLAY ALL STUDENTS RECORD";
cout<<"\n\n\t3.DISPLAY SPECIFIC STUDENT RECORD ";
cout<<"\n\n\t4.MODIFY STUDENT RECORD";
cout<<"\n\n\t5.DELETE STUDENT RECORD";
cout<<"\n\n\t6.CREATE BOOK ";
cout<<"\n\n\t7.DISPLAY ALL BOOKS ";
cout<<"\n\n\t8.DISPLAY SPECIFIC BOOK ";
cout<<"\n\n\t9.MODIFY BOOK ";
cout<<"\n\n\t10.DELETE BOOK ";
cout<<"\n\n\t11.BACK TO MAIN MENU";
cout<<"\n\n\tPlease Enter Your Choice (1-11) ";
cin>>ch2;
switch(ch2)
{
        case 1: clrscr();
                write_student();break;
        case 2: display_alls();break;
        case 3:
                char num[6];
                clrscr();
                cout<<"\n\n\tPlease Enter The Admission No. ";
                cin>>num;
                display_sps(num);
                break;
        case 4: modify_student();break;
        case 5: delete_student();break;
        case 6: clrscr();
                write_book();break;
        case 7: display_allb();break;
        case 8: {
                char num[6];
```

```
                clrscr();
                cout<<"\n\n\tPlease Enter The book No. ";
                cin>>num;
                display_spb(num);
                break;
                }
        case 9: modify_book();break;
        case 10: delete_book();break;
        case 11: return;
        default:cout<<"\a";
    }
    admin_menu();
}
void main()
{
    char ch;
    intro();
    do
    {
        clrscr();
        cout<<"\n\n\n\tMAIN MENU";
        cout<<"\n\n\t01. BOOK ISSUE";
        cout<<"\n\n\t02. BOOK DEPOSIT";
        cout<<"\n\n\t03. ADMINISTRATOR MENU";
        cout<<"\n\n\t04. EXIT";
        cout<<"\n\n\tPlease Select Your Option (1-4) ";
        ch=getche();
        switch(ch)
        {
            case '1':clrscr();
                book_issue();
                break;
```

```
                    case '2':book_deposit();
                            break;
                    case '3':admin_menu();
                            break;
                    case '4':exit(0);
                    default :cout<<"\a";
                }
        }while(ch!='4');
}
```

4.Snapshosts of Output

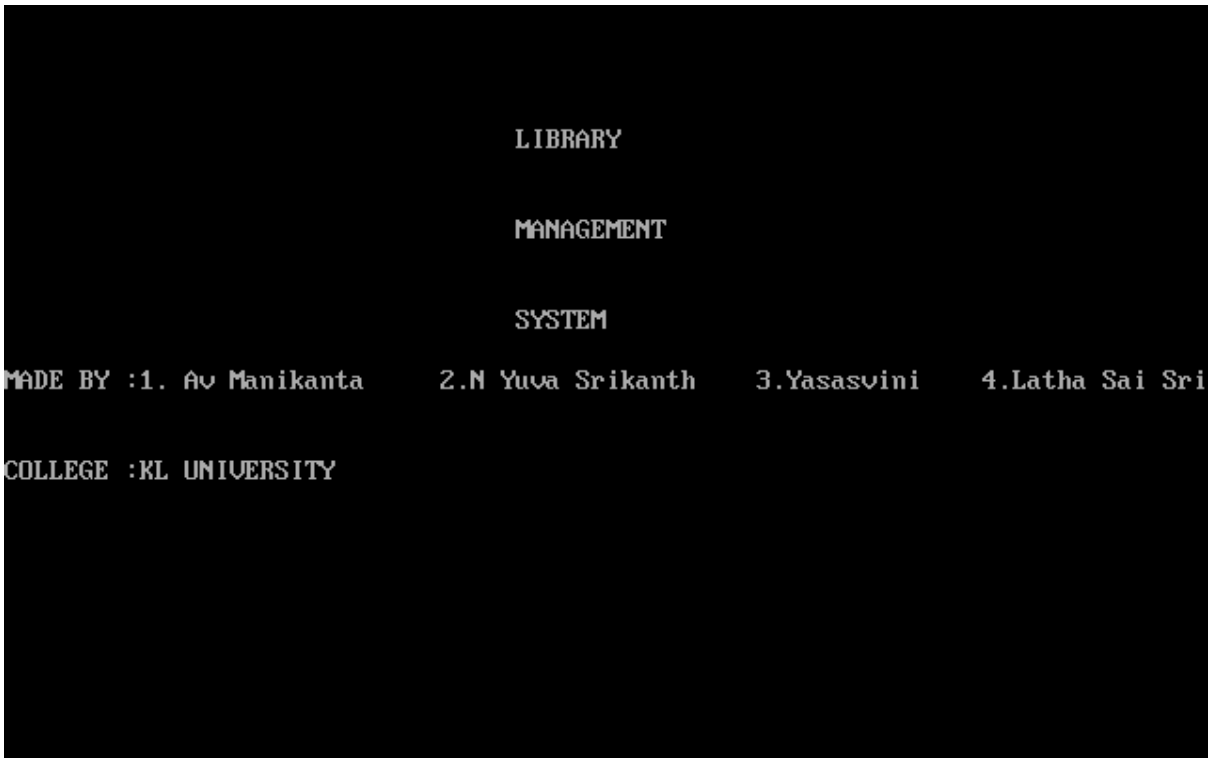


Figure 4.1 Introduction of Library Management System

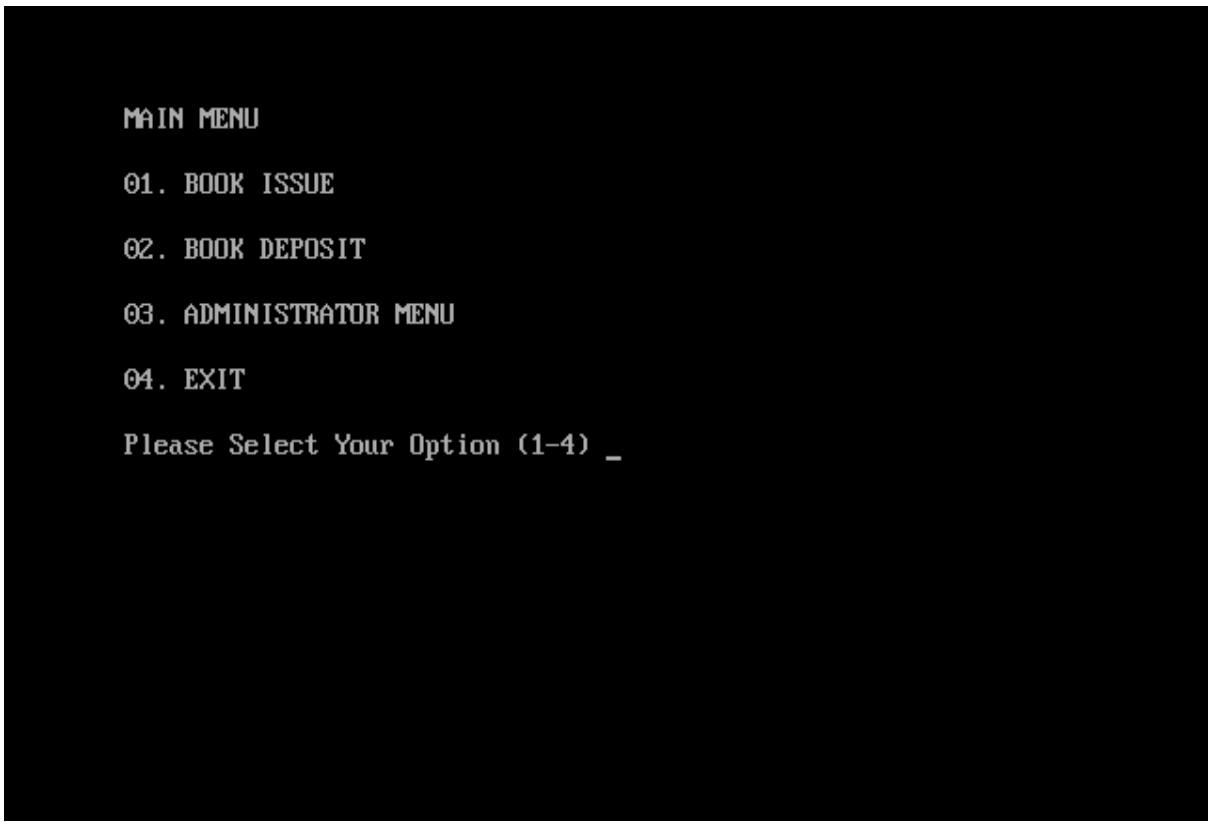


Figure 4.2 Main menu of Library Management System

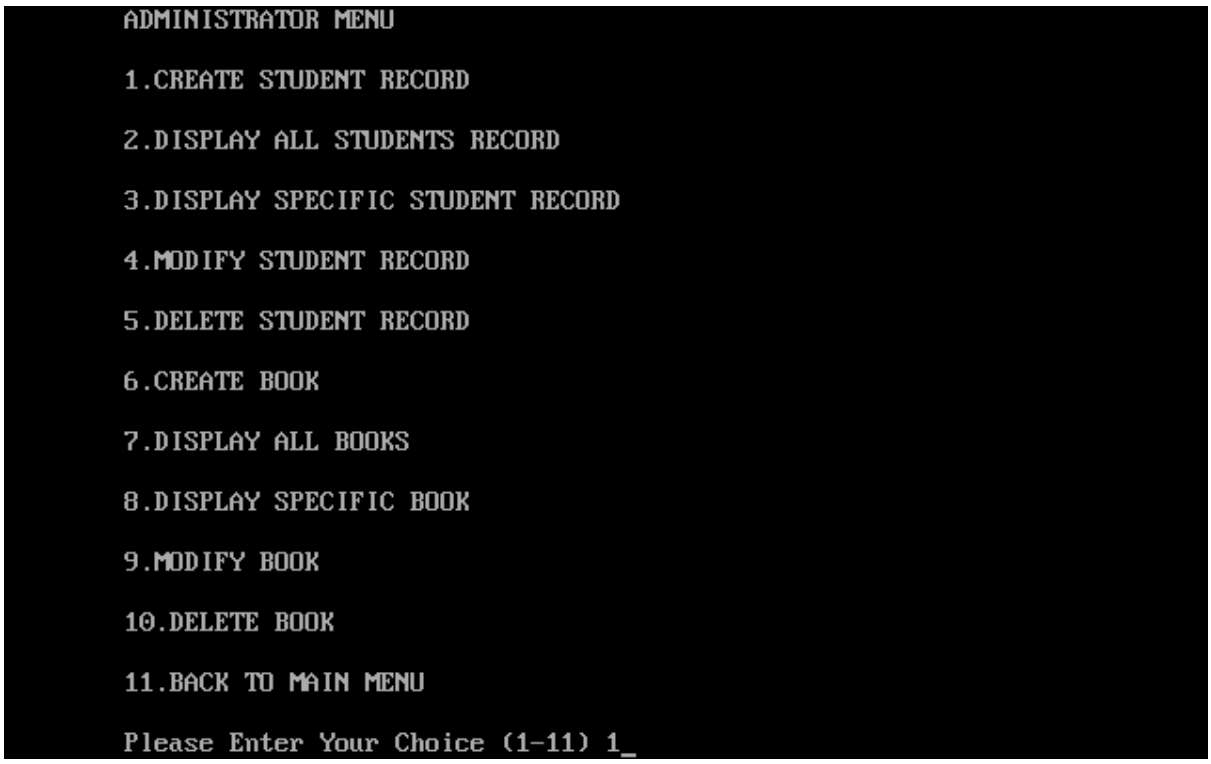


Figure 4.3 Administrator Menu

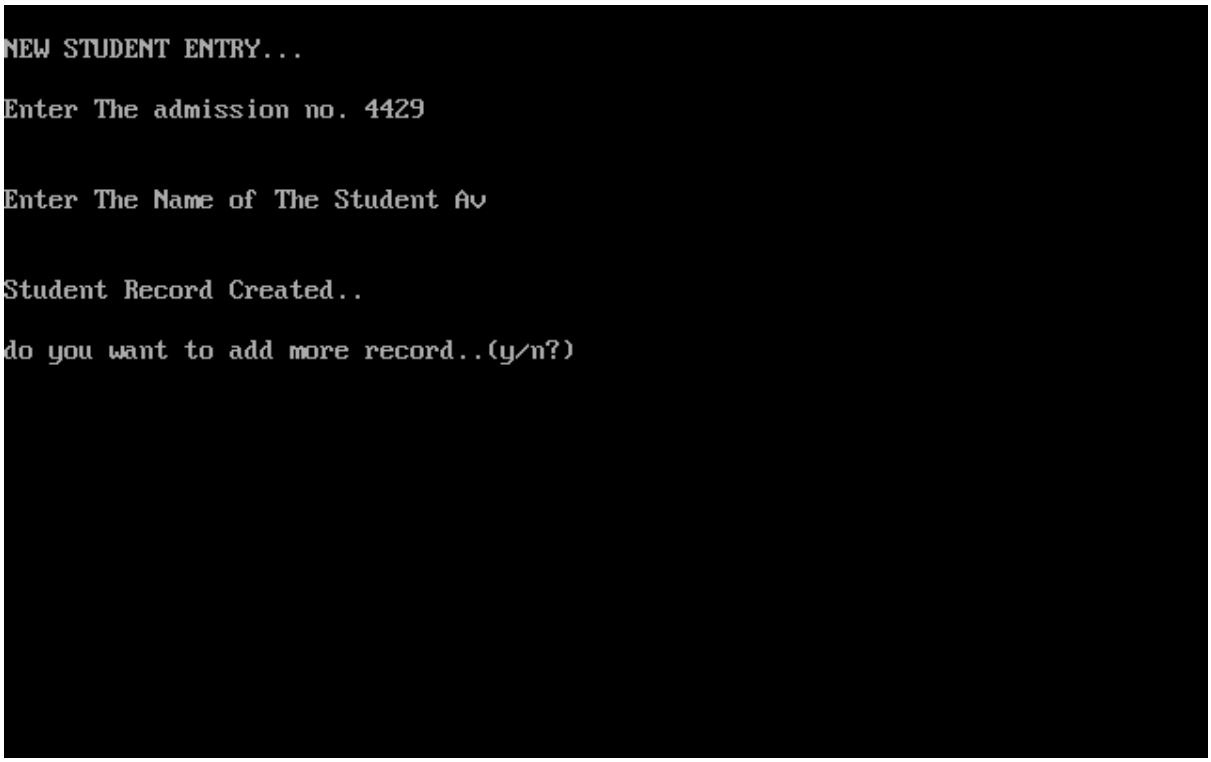


Figure 4.4 New Student Entry

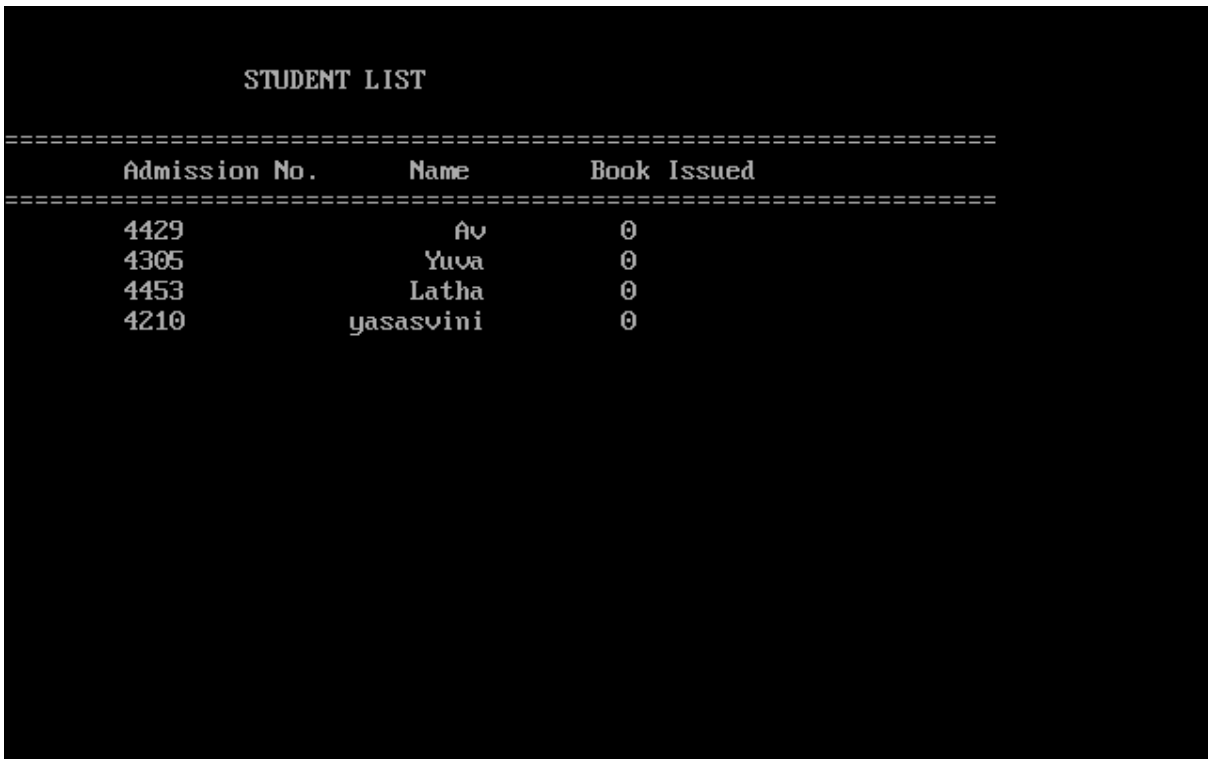


Figure 4.5 List of Students registered

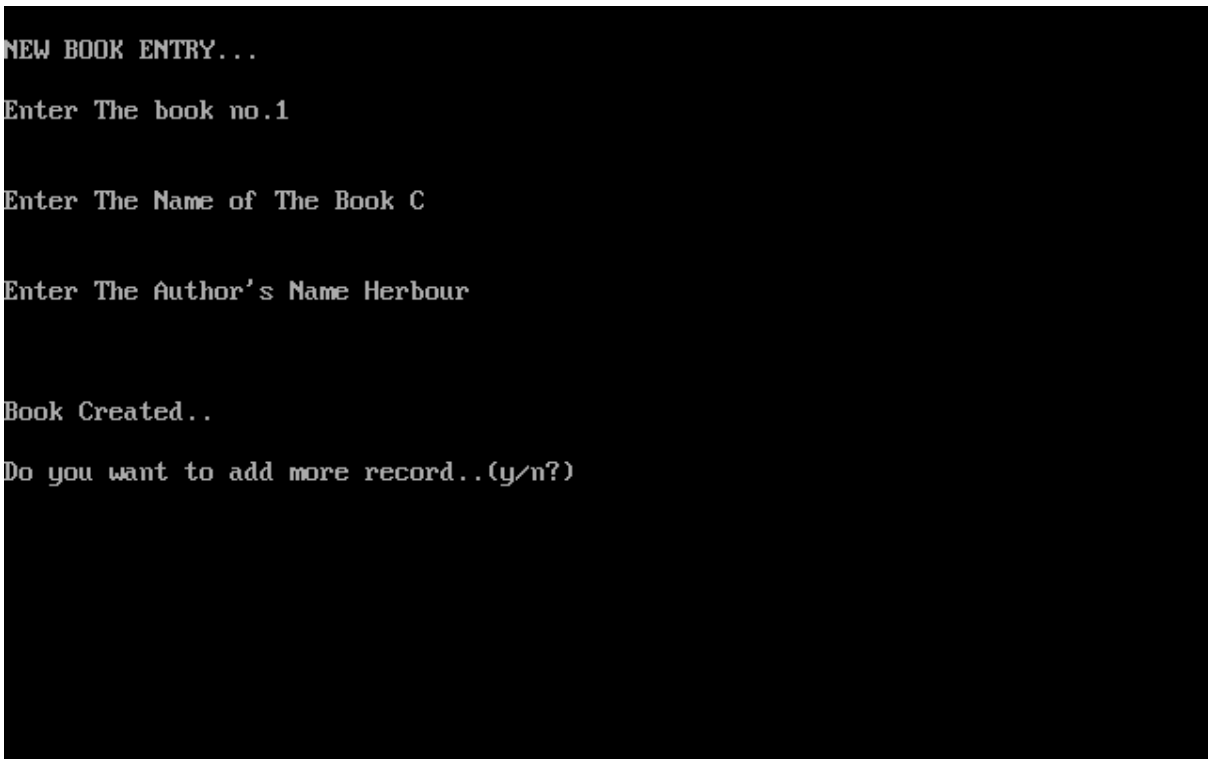


Figure 4.6 Entry for new books

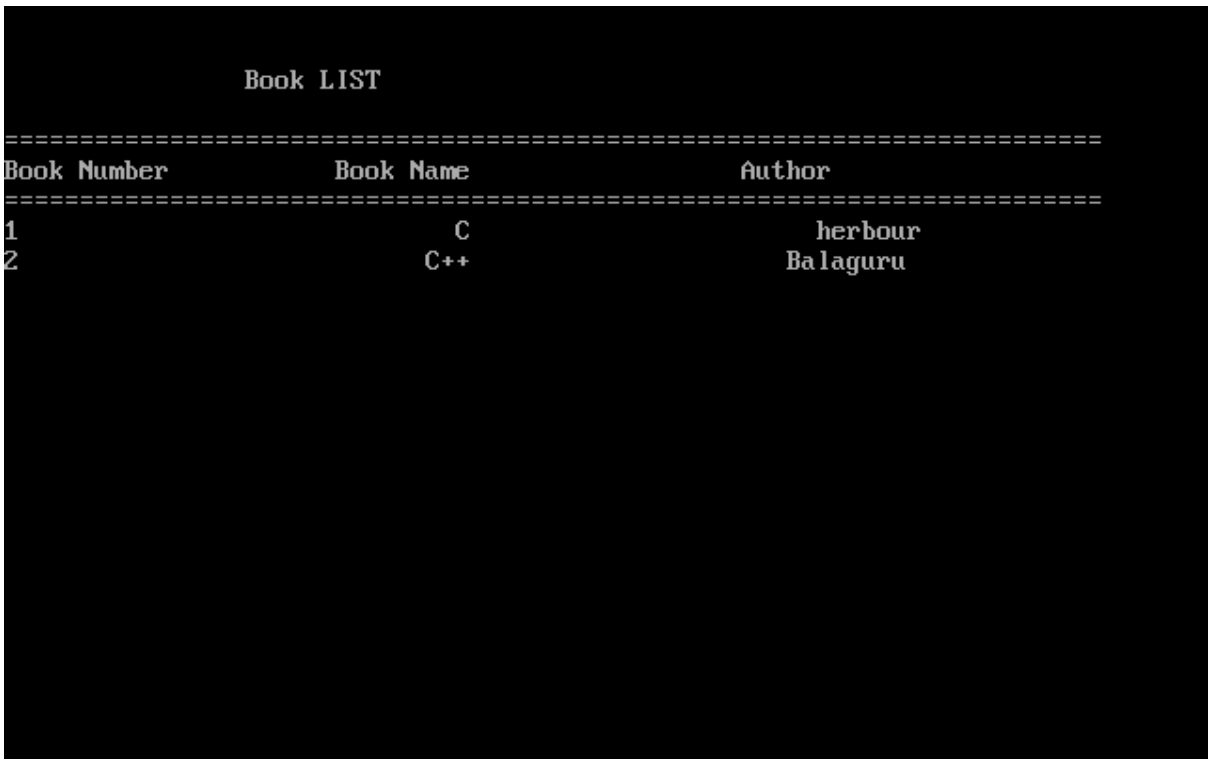


Figure 4.7 List of books present in library

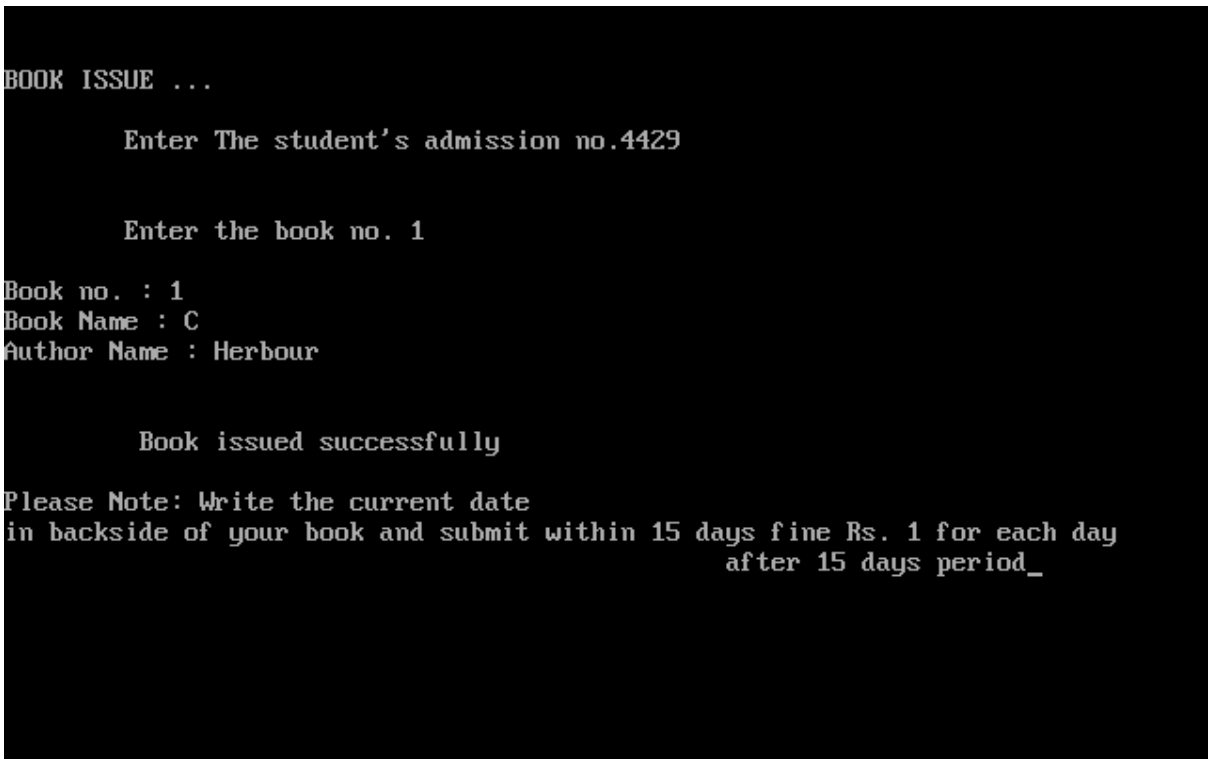


Figure 4.8 Student issuing a book

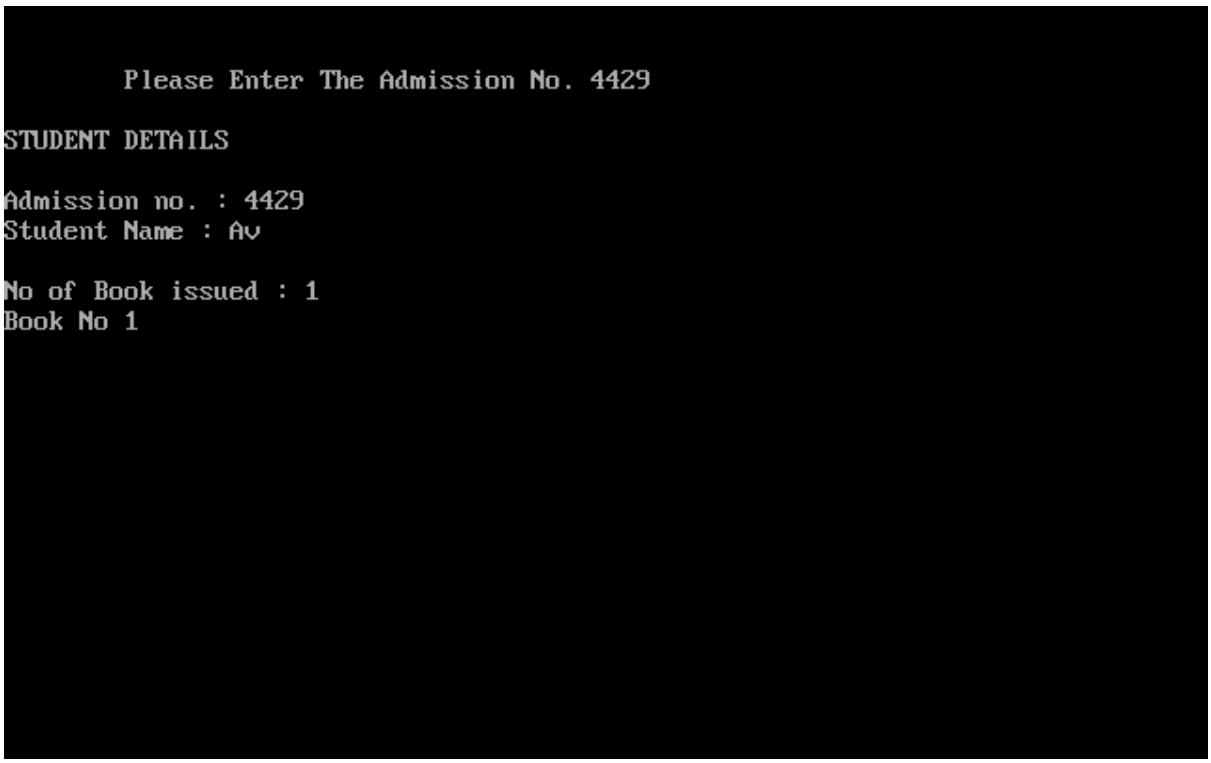


Figure 4.9 Student details

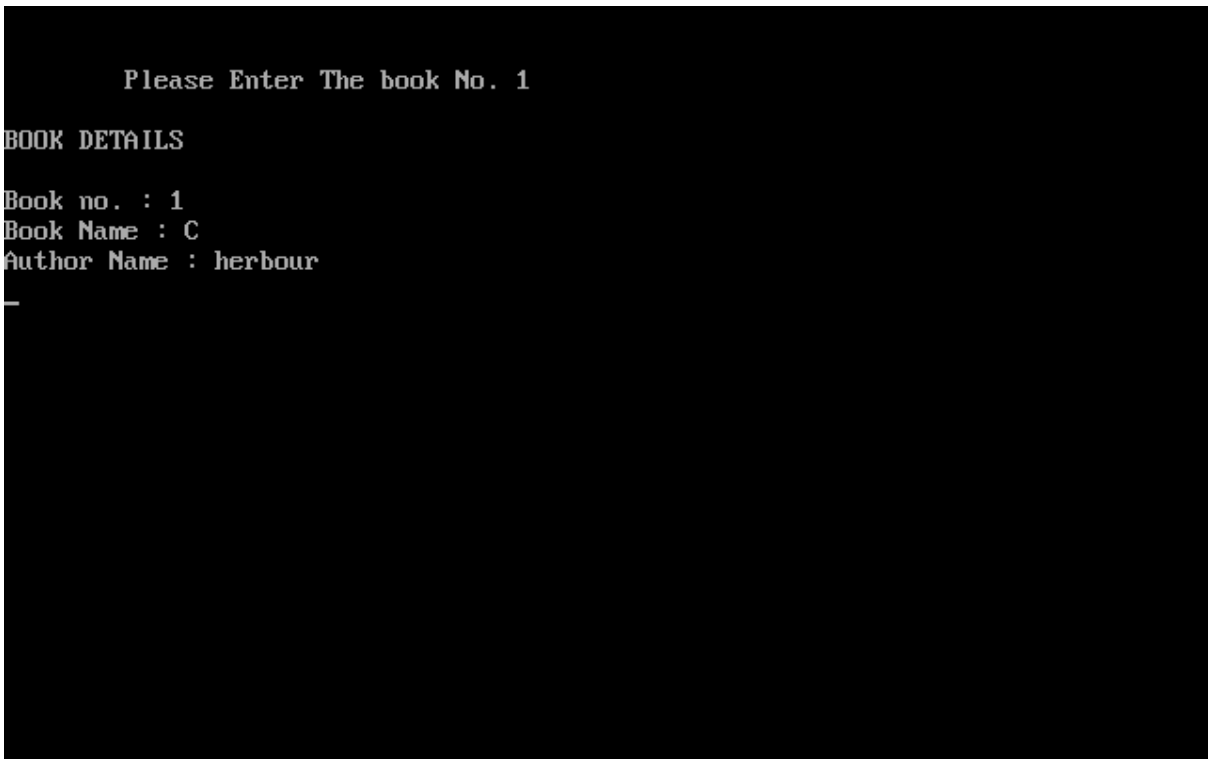


Figure 4.10 Book details

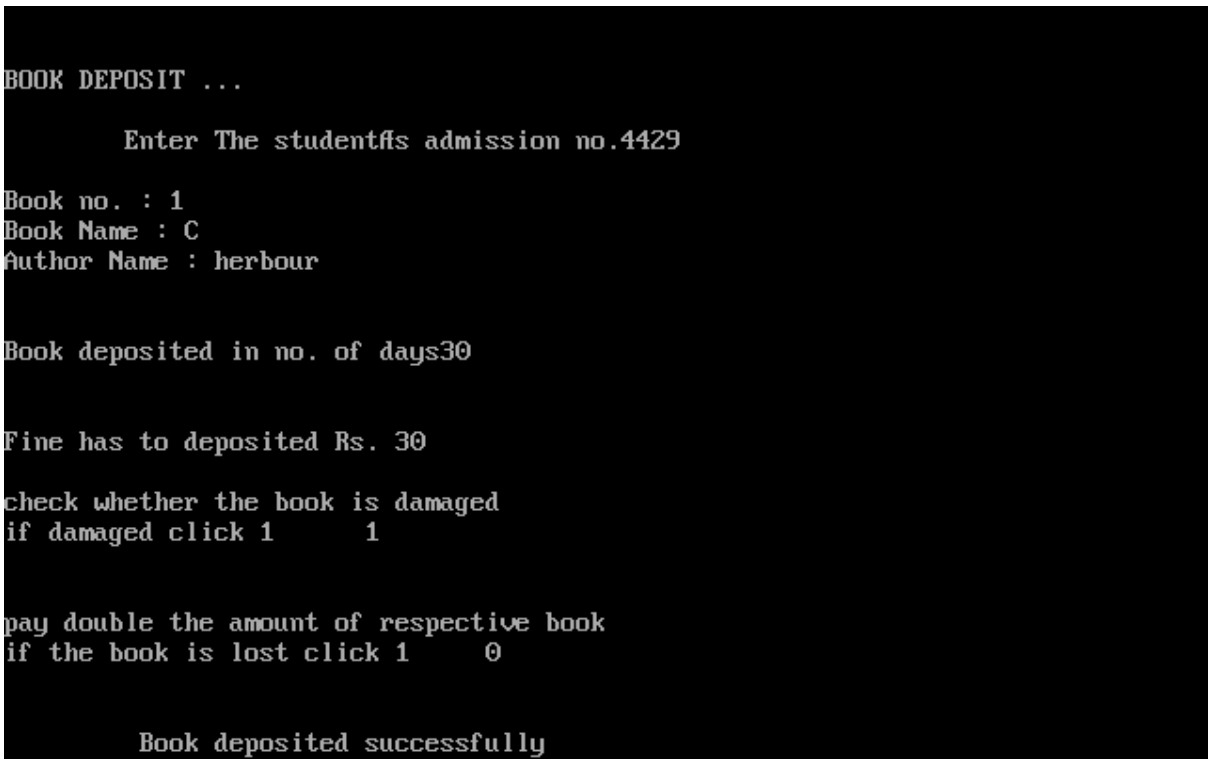


Figure 4.11 Student depositing a book

5. **References**:

a. Budd, T. (1997b), An Introduction to Object-Oriented Programming, 2nd edn, Addison-Wesley.

b. Ghezzi, C., Jayazeri, M. & Mandrioli, D. (1998), Fundamentals of Software Engineering, 2nd edn,Prentice-Hall.

c. K. Appel and W. Haken. (1976), Every Planar Map is 4-colorable, Bull. Amer. Math. Soc.**,** vol. 82, pp. 711-712