Library Database Management System

# Design Document:

[Design Document: 1](#_Toc156680423)

[**Main idea:** 2](#_Toc156680424)

[**Functions Prototypes and the Functionality of the Functions:** 3](#_Toc156680425)

[**Structures:** 6](#_Toc156680426)

[**Constants:** 7](#_Toc156680427)

[**Arrays:** 7](#_Toc156680428)

[**Input Variables:** 8](#_Toc156680429)

[**Output Variables:** 8](#_Toc156680430)

[**Interaction Between Functions:** 9](#_Toc156680431)

### **Main idea:**

1. Display the menu and input the user’s choice(from 1 to 10).
2. Open and Read from “books.dat” for the first 7 options.
   1. If option is 1, first input the new record in declared variables. Then increment the last accession number by 1 and append the “books.dat” .
   2. If option is 2, first input the book name or accession number and then search for the book. Once the record is found, delete the record.
   3. If option is 3, first input the book name. Then check whether the book is issuable or not. If issuable, check if the book has already been issued or not. If not issued, then input the current date and Employee-ID/ Student-ID, change the field issued to “true”, change the date of issuance field to the date entered and the “Issued to” field to the Employee-ID/ Student-ID entered. If it is a student, the return date will be set to next 15 days, otherwise, 4 months.
   4. If option is 4, first input the book name. Input the current date. Then change the field of issued to false. Set the field of Employee-ID/ Student-ID to ‘0’. If the current date is greater than the return date,a fine will be calculated and displayed. Then set the field of return date to “00/00/0000”
   5. If option is 5, input the next option (a to c)
      1. If option is a, input the author’s name. Compare the input with the records of the file. Display the records that match the author’s name.
      2. If option is b, input the co-author’s name. Compare the input with the records of the file. Display the records that match the co-author’s name.
      3. If option is c, input the accession number. Compare the input with the records of the file. Display the record that matches the accession number.
   6. If option is 6, input the next option (a to b) and make a new file.
      1. If option is a, arrange the data based on the author’s name and store it in the new file.
      2. If option is b, arrange the data based on the Book Title and store it in the new file.
   7. If option is 7, input the next option (a to b).
      1. If option is a, input the name of the author. Then display all the books that are written by the same author.
      2. If option is b, input the subject. Then display all the books that are of the same subject.
3. Open and Read from “users.dat” for the options (from 8 to 9).
   1. If option is 8, input the Employee-ID/ Student-ID of the record you want to change. Then display another menu, asking the user what they want to change. Input the new field in a variable. Search through the records and change the specified field in the record.
   2. If option is 9, input the next option (a to c).
      1. If option is a, input the name. Display the record from “users.dat” matching the name as well as the books issued to him and his age.
      2. If option is b, input the ID. Display the record from “users.dat” matching the ID as well as the books issued to him and his age.
      3. If option is c, input the Cell Number. Display the record from “users.dat” matching the Cell Number as well as the books issued to him and his age.
4. If option is 10, exit and end the program.

### **Functions Prototypes and the Functionality of the Functions:**

Justification for passing ofstream and ifstream by reference:

1. The files are accessed multiple times and hence having them open once and sending them by reference is much more efficient.
2. Declaring new variables in functions was discouraged.

Functions:

1. Function to Display the Menu:

*void Menu();*

This function outputs the whole menu.

1. Function to read the file *“books.dat”* into the array:

void ReadFileBook(ifstream& finBook, Book books[]);

This function reads the file *“books.dat”* and stores the values in the structure array *“books”*.

1. Function to read the file *“users.dat”* into the array:

void ReadFileUser(ifstream& finUser, Book users[]);

This function reads the file *“users.dat”* and stores the values in the structure array *“users”*.

1. Function to insert a new book into *“books.dat”*:

void BookInsert(ofstream& foutBook, Book books[]);

This function is used to add a new record of a book input from the user into the file *“books.dat”*.

1. Function to delete a book from *“books.dat”*:

void BookDelete(ofstream& foutBook, Book books[]);

This function is used to delete a particular record specified by the user from the *“books.dat”* file.

1. Function to issue a book:

void BookIssue(ofstream& foutBook, Book books[]);

This function is used to issue a book to an employee or student and also to change certain values in *“books.dat”*.

1. Function to return a book:

void BookReturn(ofstream& foutBook, Book books[]);

This function is used when a student or employee returns a book. It changes certain values in *“books.dat”*.

1. Function to search a book:

void BookSearch(const Book books[]);

This function is used to search a book using the authors name, co-authors name, or the accession number and display its complete record.

1. Function to sort the books in ascending order:

void BookSort(Book books[]);

This function is used to arrange the records in ascending order and store it in a new file either based on the author's name or the book title.

1. Function to display the books specified by the input:

void BookDisplay(Book books[]);

This function displays all the books written by a specific author, or belonging to a particular subject.

1. Function to edit *“users.dat”*:

void UserEdit(ofstream& foutUser, User users[]);

This function modifies the *“users.dat”* file according to the field specified and the input to replace the field with.

1. Function to search and display user information:

void UserSearch(User users[], Book books[]);

This function searches a user based on name, ID, or cell number and displays the complete user record along with the book titles issued to him and their age. This function calls the *“Age()”* function to calculate the user's age and then display it.

1. Function to get the current date:

void GetDate(Date& date);

This function is used to get the input a date from the user.

1. Function to calculate the age of user:

Date Age(Date dob, Date currentDate);

This function is used to calculate the age of the user.

### 

### 

### 

### 

### 

### 

### 

### **Structures:**

*struct Date*

*{*

*int day;*

*int month;*

*int year;*

*};*

This structure will be used to store different Dates in their proper formats.

*struct Book*

*{*

*char author[30];*

*char coauthor[30];*

*char title[45];*

*char publisher[30];*

*char subject[30];*

*char ID[5];*

*int accessionNumber;*

*int yearOfPub;*

*bool issuable;*

*bool issuedOrNot;*

*Date issuanceOfBook;*

*Date returnDate;*

*};*

This structure will hold the information regarding any book and will use the “Date” structure to store the Dates related to the book.

*struct User*

*{*

*char name[30];*

*char fathersName[30];*

*char rank[10];*

*char department[30];*

*char address[70];*

*char ID[5];*

*char cellNumber[15];*

*Date dob;*

*};*

This structure will store the information of all the users using the library facilities, be it faculty, staff, or students.

### **Constants:**

*const int FINE = 50;*

The constant *“FINE”* is used to store the fine per day of the books.

*const int SIZE = 1000;*

The constant *“SIZE”* is used to store the maximum size of the arrays and is used when declaring the array.

### **Arrays:**

*Book books[SIZE];*

The array *“books”* stores all the records from the file *“books.dat”* and stores it in its attributes accordingly(*‘books.author’, ‘books.coauthor’,* etc).

*User users[SIZE];*

The array *“users”* stores all the records from the file *“users.dat”* and stores it in its attributes accordingly(*‘users.name’, ‘users.fathersName’,* etc).

### **Input Variables:**

*ifstream finBook;*

*“finBook”* is used to input from the file *“books.dat”* whenever needed. It will be declared in *“int main”* and will be passed to the functions that need it.

*ifstream finUser;*

*“finUser”* is used to input from the file *“books.dat”* whenever needed. It will also be declared in *“int main”* and will be passed to the functions that need it.

*char bookFileName[10];*

The character array *“bookFileName”* will input the name of the file of books.

*char userFileName[10];*

The character array *“userFileName”* will input the name of the file of users.

### **Output Variables:**

*ofstream foutBook;*

*“foutBook”* is used to output to the file *“books.dat”* whenever needed. It will be declared in *“int main”* and will be passed to the functions that need it.

*ofstream foutUser;*

*“foutUser”* is used to output to the file *“users.dat”* whenever needed. It will also be declared in *“int main”* and will be passed to the functions that need it.

### 

### **Interaction Between Functions:**

Main

Menu

ReadFileBook

ReadFileUser

BookInsert

BookDelete

BookIssue

BookReturn

BookSearch

BookSort

BookDisplay

UserEdit

UserSearch

Age

GetDate