LIBRARY MANAGEMENT SYSTEM

Next time, you can develop a virtual library. What I mean by a virtual library is that, you keep lets say 3 stock of a particular book.

Then you can control which client accessed that book, and if the number of books goes to zero (all books were issued), you can put that particular book out of stock.

Additionally, in the next semester, you can also make available, different research papers.

Session 2022-2026

Submitted By:

Zainab Idrees 2022-CS-199

Supervised By:

Dr. Awais Hassan

Department of Computer Science

University of Engineering and Technology

Lahore Pakistan

Table of Contents

ABSTRACT

Library management system is a software application built to automate and manage the activities of a library. However, several problems can arise. It is crucial to solve these issues to ensure that the system runs effectively. One common issue is inaccurate and outdated information. If the library management system is not updated regularly, it can lead to errors during the reserve and purchase process, resulting in inaccurate information about available books. Therefore, the library management system must be updated regularly, and all changes must be recorded accurately in the system. Another problem is difficulty finding and locating books. Libraries may have a huge collection of books, which makes it challenging to find what they need. To solve this issue, the library management system should have a search feature that enables users to find books using author, title, subject. The system should also have a feature that displays the availability and location of the books. Overdue items and late fees can also be a problem. To solve this issue, the library management system should have a feature that automatically generates notifications and reminders to members about overdue items. It should also have a fine calculation feature that calculates late fees based on the number of days that the item is overdue. To conclude, an effective library management system must be regularly updated, have a user-friendly interface that enables easy search and location of resources, and have features that ensure notifications and accurate calculation.

BRIEF DESCRIPTION OF THE LIBRARY MANAGEMENT SYSTEM

A Login and Sign-Up functionality is provided in this library management system. The system should ask the user to enter their credentials in order to use the features of the system. If the user does not exist, the system should display a proper message and ask the user to sign up instead. The system should only let a registered user access its functionalities. A menu that will provide access to the features of the Library Management System. Note that the system should only terminate when the user instructs to do so. Otherwise, after execution of each function the system should ask the user to return to the menu or execute the functionality again. The storage of Books. A book should have a title. The retrieval of books stored in the database by searching. For this purpose, the user can input the title of the book. Adding new books into the library database and deleting a book from the database. The database will in the form of text files on the local storage. Lending a book to a person. Returning an issued book to the library. In this way, the online library management system helps us to look for thousands of books of different genres in a few minutes without wasting the time and searching one book through all the books, all thanks to computer science.

USERS OF LIBRARY MANAGEMENT SYSTEM

- User (Students of any educational institution)
- Admin (The Librarian)

FUNCTIONAL REQUIREMENTS

ADMIN

As an	*I want to perform*	*so that I can*
Admin	Sign up	Create an account
Admin	Sign In	Manage the library system
Admin	View	View the books present in the library
Admin	Add	Add a new book to the library for the users
Admin	Update	Provide the users the latest version of a book
Admin	Delete	Delete the books that are no longer of use

USER

As an	*I want to perform*	*so that I can*
User	Sign Up	Create an account
User	Sign In	Manage the library system
User	View	View the books present in the library
User	Purchase	Buy books for long term use
User	Reserve	Save books to read them later

VALIDATIONS

- Any particular book should have the information such as book title.
- An admin or a user must sign up before signing in to the account.
- An admin or a user cannot sign up using the same username as the previous admin or user has entered, so the usernames must be unique.
- If the user or admin enters a number other than the choices given in the menu, they will have to enter the choice again.
- When a book is reserved, it is stored in a file in their respective directories.
- The library has the capacity of 10 books, therefore more than 10 books cannot be stored in the library.

SORTING

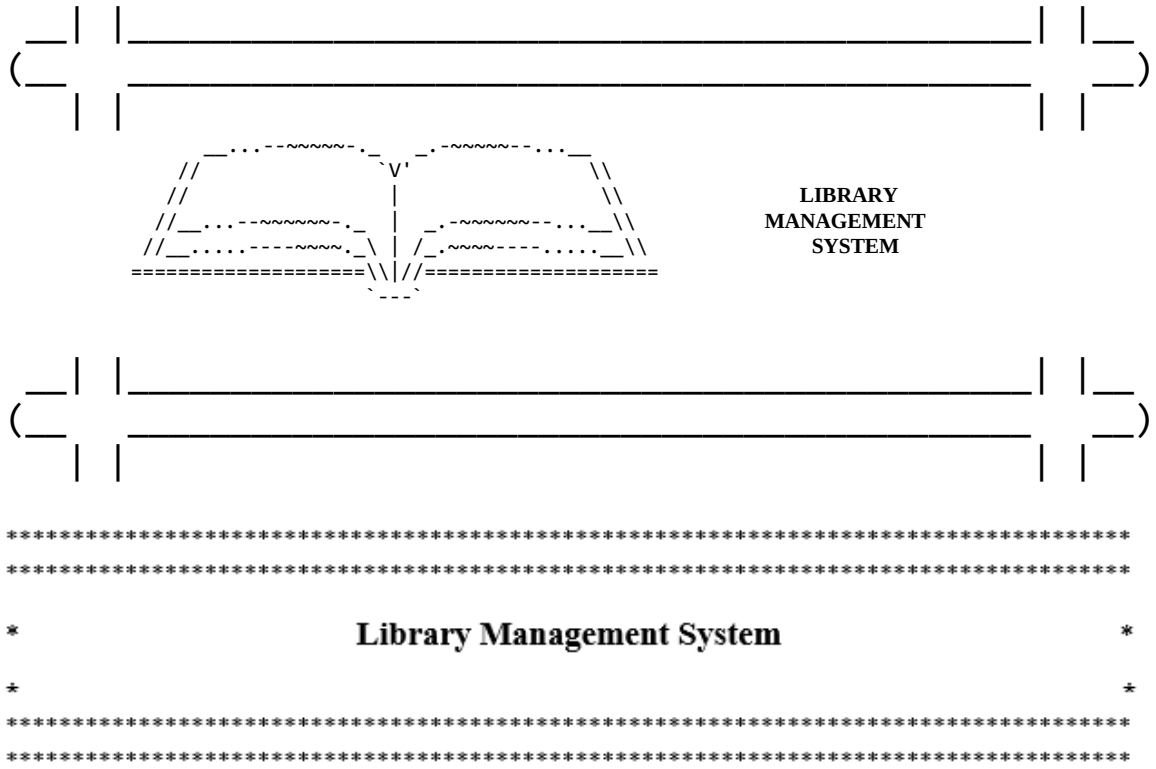
The books must be sorted according to their respective prices in an ascending order, in this way, the users find it easy to find a particular book without looking through all the books in the library.

RECOMMENDATIONS

Harry Potter the Series: A journey of a boy and his friends fighting for their adventure! (234 reservations & 34 purchases!)

Peer-e-Kaamil: The story of a man finding his true self after repenting on the countless mistakes he did throughout his life, but everything changed when he met a girl!
(123 reservations, 20 purchases!)
Reserve yours NOW! Come first get first service!!

WIREFRAMES

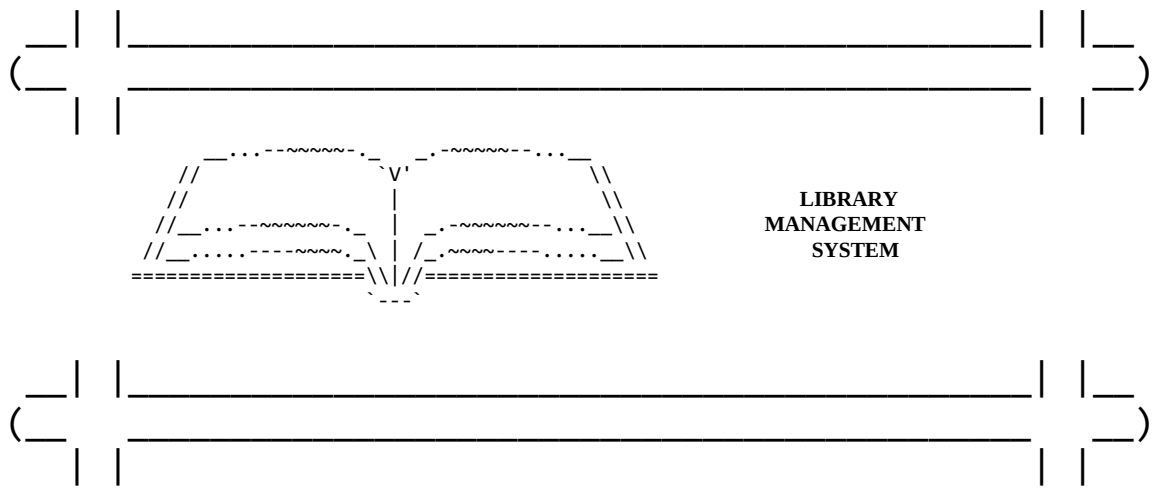


Press (1) for Admin

Press (2) for User

Enter your choice:

(1.1) FIGURE 1: Person Page




```
**********************************************************************************
**********************************************************************************
*                        Library Management System                               *
*                                                                                *
**********************************************************************************
**********************************************************************************
```

Enter your choice: 1

Press (1) to Sign Up

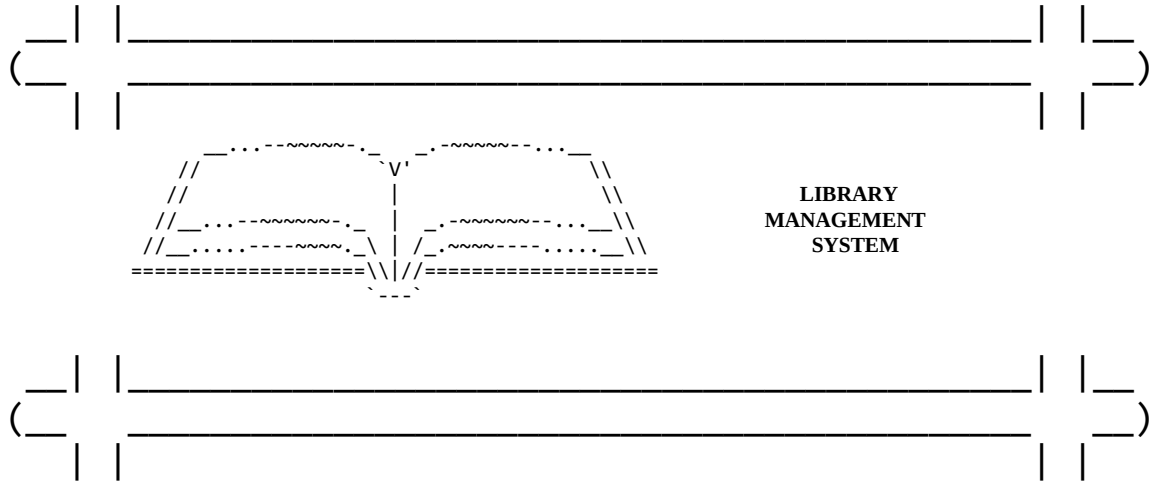
Press (2) to Login In

Press (3) to View Users

Press (4) to go Back

Enter your choice:

(1.1.1) FIGURE 2: Admin StartUp Page



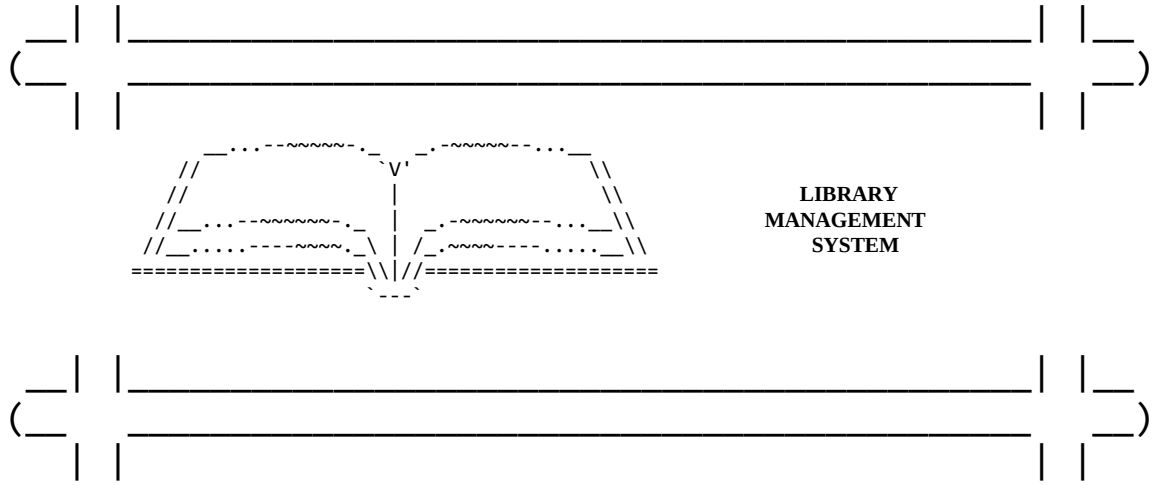

```
**********************************************************************************
**********************************************************************************
*                        Library Management System                               *
*                                                                                *
**********************************************************************************
**********************************************************************************
```

Enter your choice: 1

Enter your username: Zainab

Enter your password: 123

(1.1.2) FIGURE 3: Admin Sign Up Page



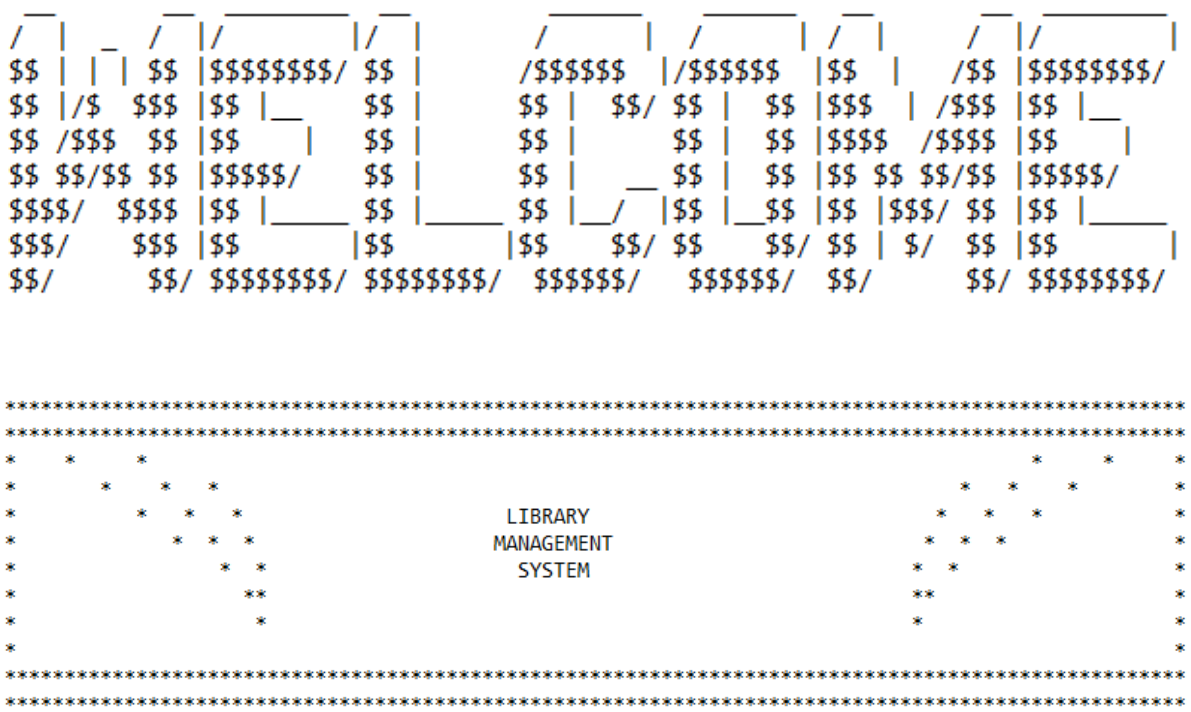

```
**********************************************************************************
**********************************************************************************
*                        Library Management System                               *
*                                                                                *
**********************************************************************************
**********************************************************************************
```

Enter your choice: 2

Enter your name: Zainab

Enter your password: 123

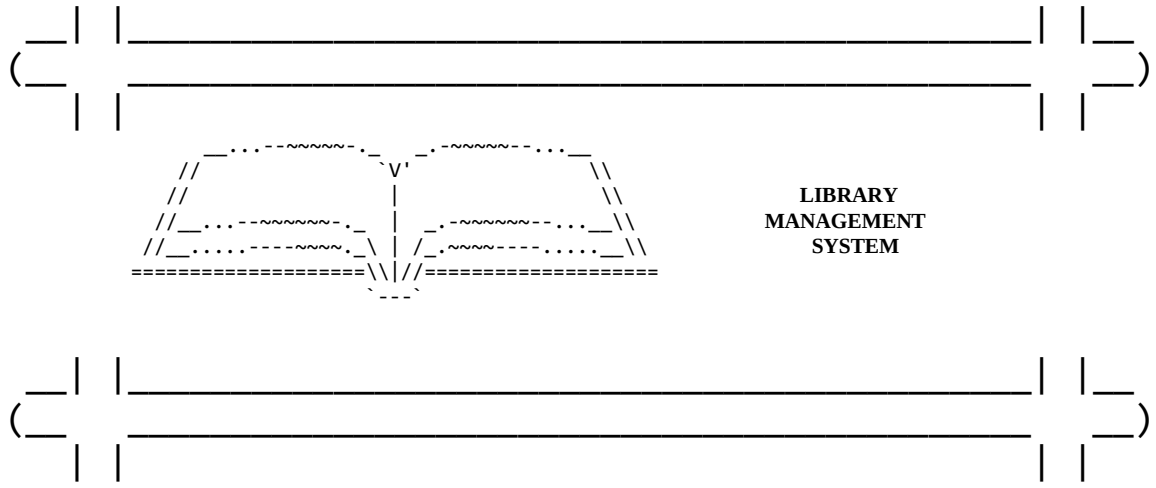
(1.1.3) FIGURE 4: Admin Sign In Page



Press any number to continue….

2

(1.1.4) FIGURE 5: Admin Welcome Page




```
**********************************************************************************
**********************************************************************************
*                        Library Management System                               *
*                                                                                *
**********************************************************************************
**********************************************************************************
```

Press (1) to View the list of books

Press (2) to Add a book

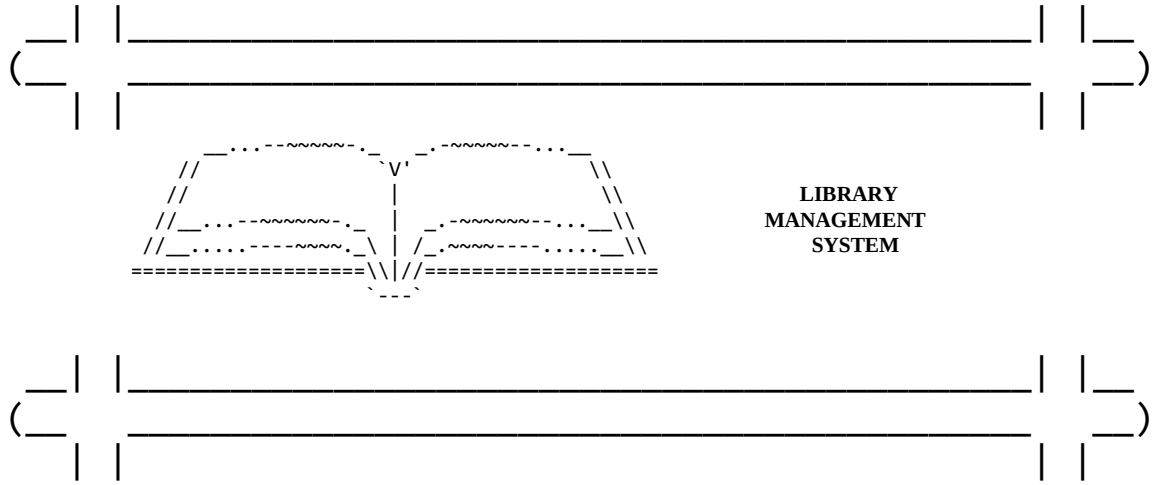
Press (3) to Update a book

Press (4) to Delete a book

Press (5) to go back

Enter your choice:

(1.1.5) FIGURE 6: Admin Menu Page




```
**********************************************************************************
**********************************************************************************
*                        Library Management System                               *
*                                                                                *
**********************************************************************************
**********************************************************************************
```

Enter your choice: 1

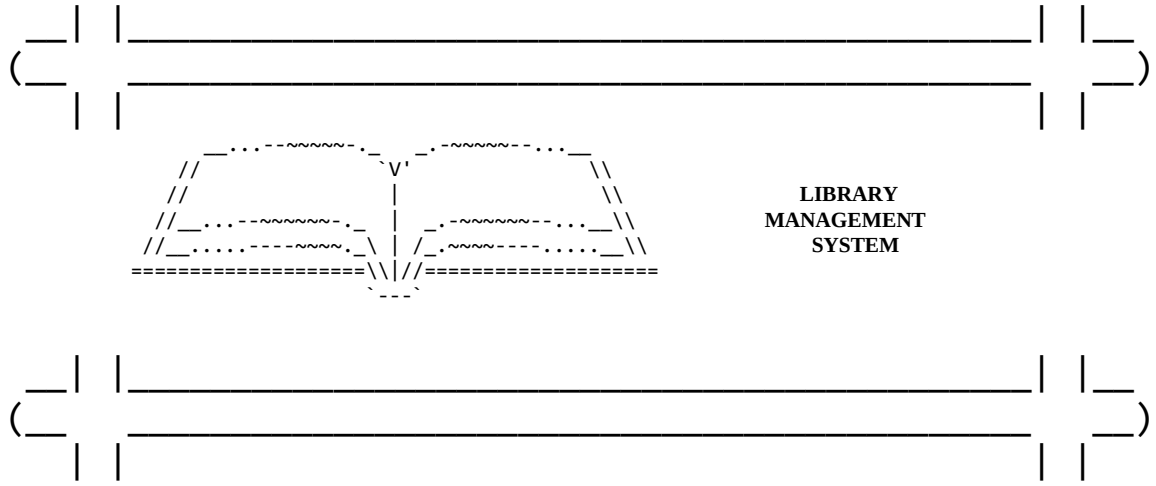
!!!!!!! WELCOME TO THE LIBRARY!!!!!!!!!!

The following books are present in our library:

0. The_Alchemist
1. The_Red_Pot
2. Peer_e_Kaamil
3. Jannat_k_Pattay
4. Harry_Potter_The_Series
5. Forty_Rules_of_Love

Press any key to continue….

(1.1.6) FIGURE 7: Books View Page




```
**********************************************************************************
**********************************************************************************
*                        Library Management System                               *
*                                                                                *
**********************************************************************************
**********************************************************************************
```

Enter your choice: 2

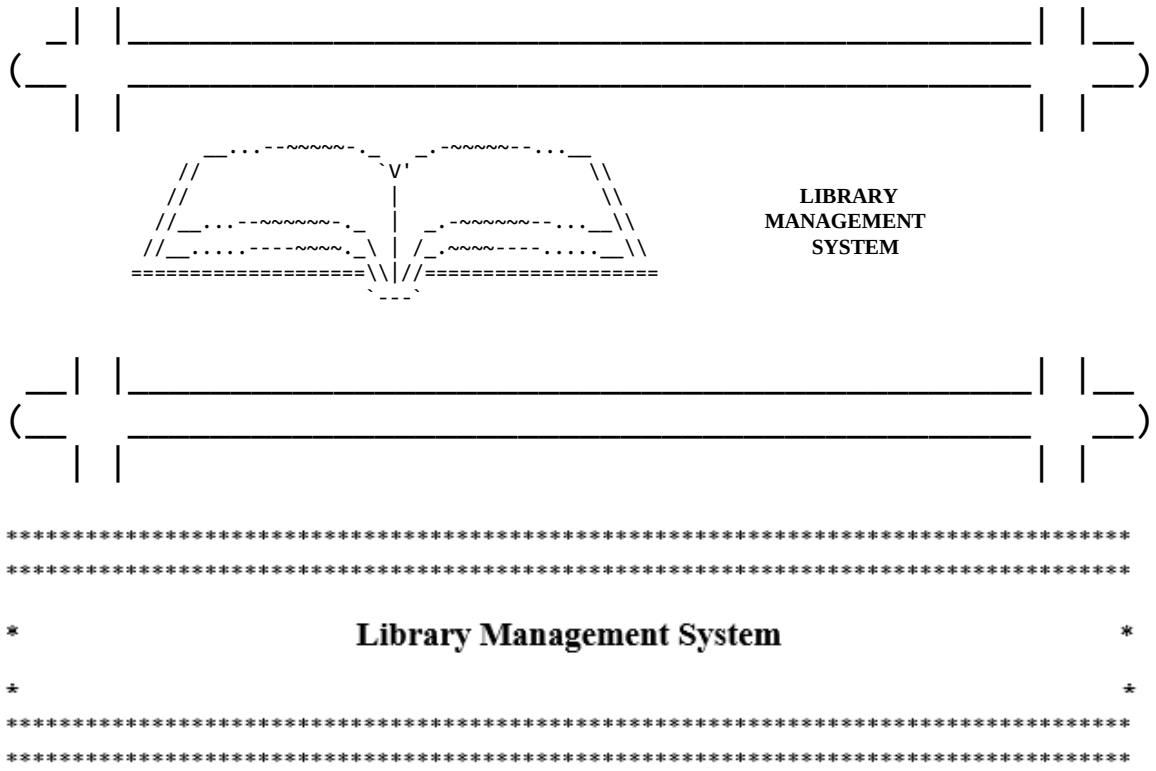
Enter the number of books you want to add: 1

Enter the name of the book you want to add: Mysterious

The book has been added in our library:

0. The_Alchemist
1. The_Red_Pot
2. Peer_e_Kaamil
3. Jannat_k_Pattay
4. Harry_Potter_The_Series
5. Forty_Rules_of_Love
6. Mysterious

(1.1.7) FIGURE 8: Book Add Page



Enter your choice: 3

The following books are present in our library:

0. The_Alchemist
1. The_Red_Pot
2. Peer_e_Kaamil
3. Jannat_k_Pattay
4. Harry_Potter_The_Series
5. Forty_Rules_of_Love
6. Mysterious

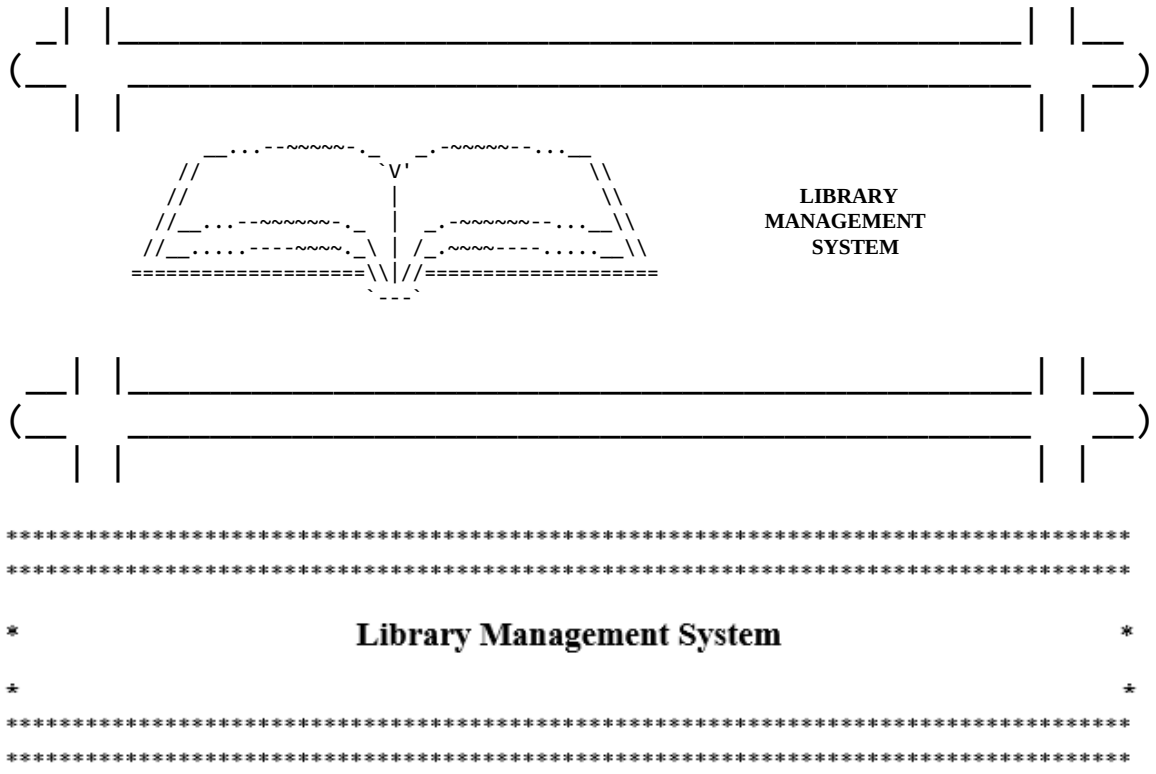
Enter the number of books you want to Update: 1

Enter the name of the book you want to update: Mysterious

Enter the name of the updated book: Mysteriousism

0. The_Alchemist
1. The_Red_Pot
2. Peer_e_Kaamil
3. Jannat_k_Pattay
4. Harry_Potter_The_Series
5. Forty_Rules_of_Love
6. Mysteriousism

(1.1.8) FIGURE 9: Book Update Page



Enter your choice: 4

The following books are present in our library:

0. The_Alchemist
1. The_Red_Pot
2. Peer_e_Kaamil
3. Jannat_k_Pattay
4. Harry_Potter_The_Series
5. Forty_Rules_of_Love
6. Mysteriousism

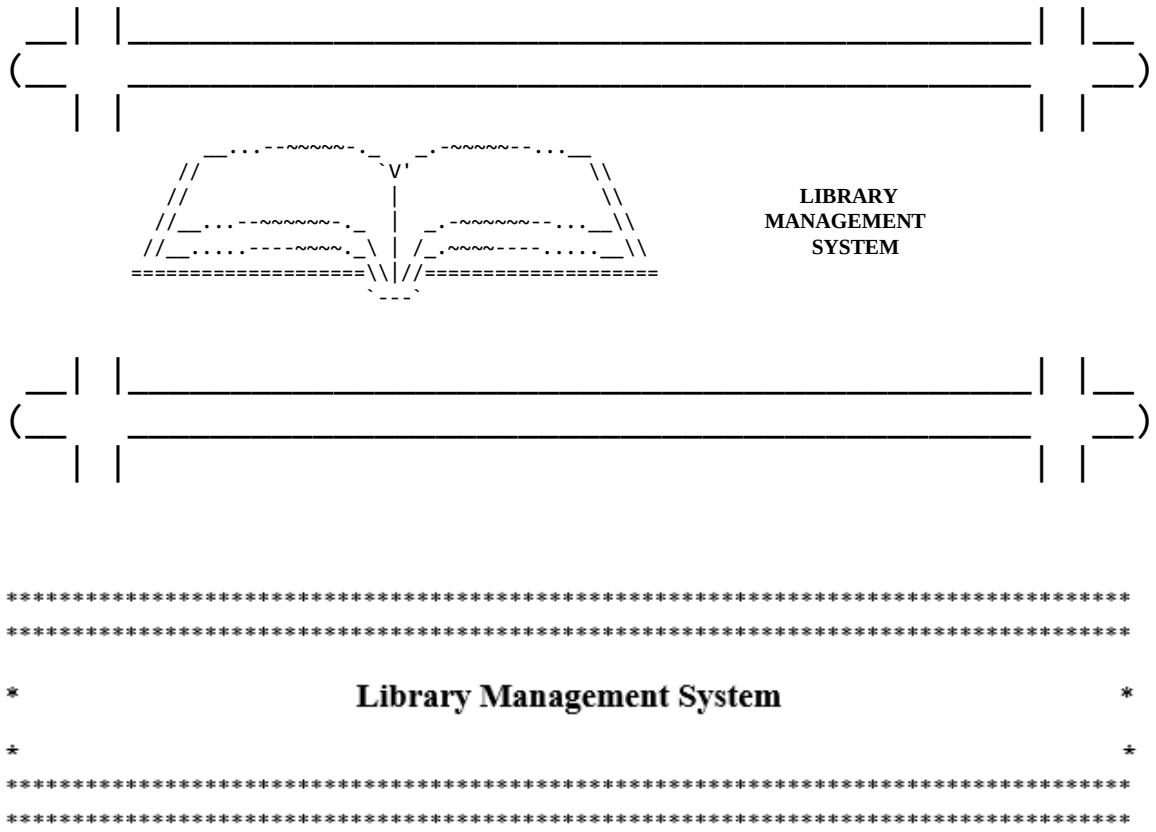
Enter the number of books you want to Delete: 1

Enter the name of the book you want to Delete: Mysteriousism

Your book has been Deleted!

0. The_Alchemist
1. The_Red_Pot
2. Peer_e_Kaamil
3. Jannat_k_Pattay
4. Harry_Potter_The_Series
5. Forty_Rules_of_Love

(1.1.9) FIGURE 10: Book Delete Page



Enter your choice: 2

Press (1) to Sign Up

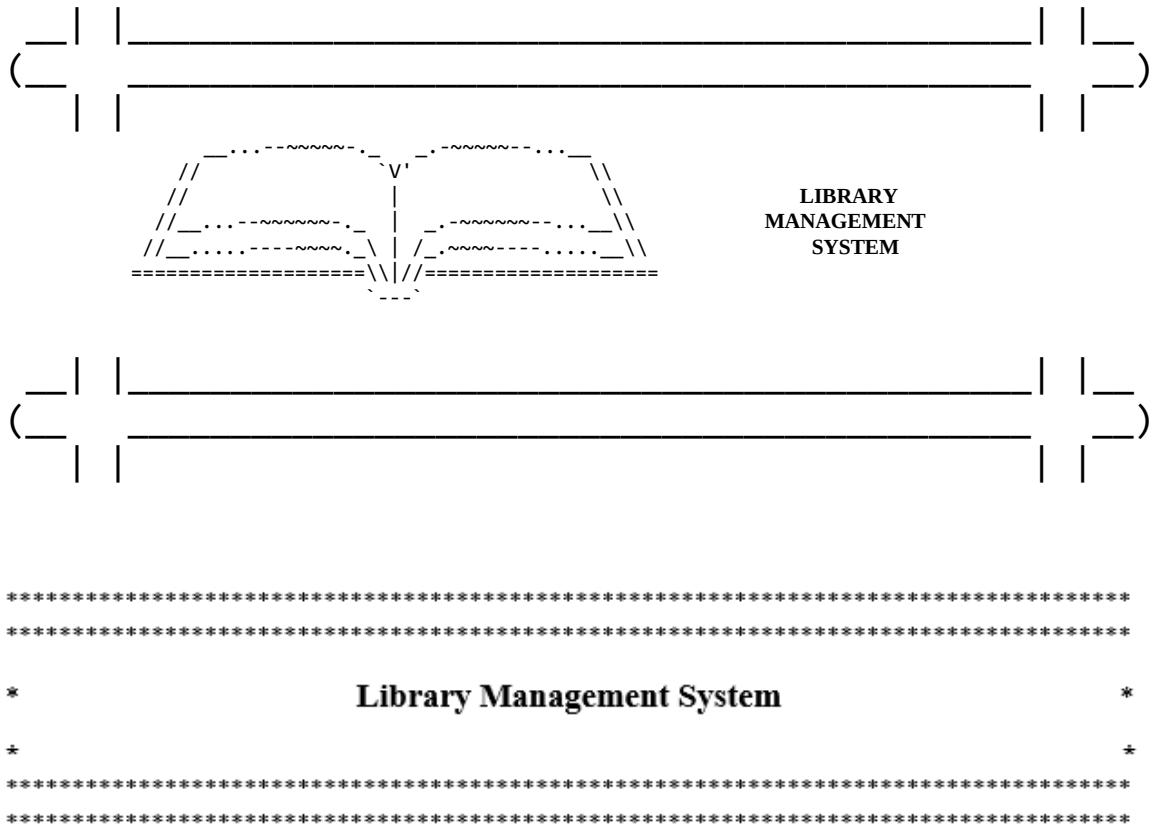
Press (2) to Login In

Press (3) to View Users

Press (4) to go Back

Enter your choice:

(1.2.1) FIGURE 11: User StartUp Page

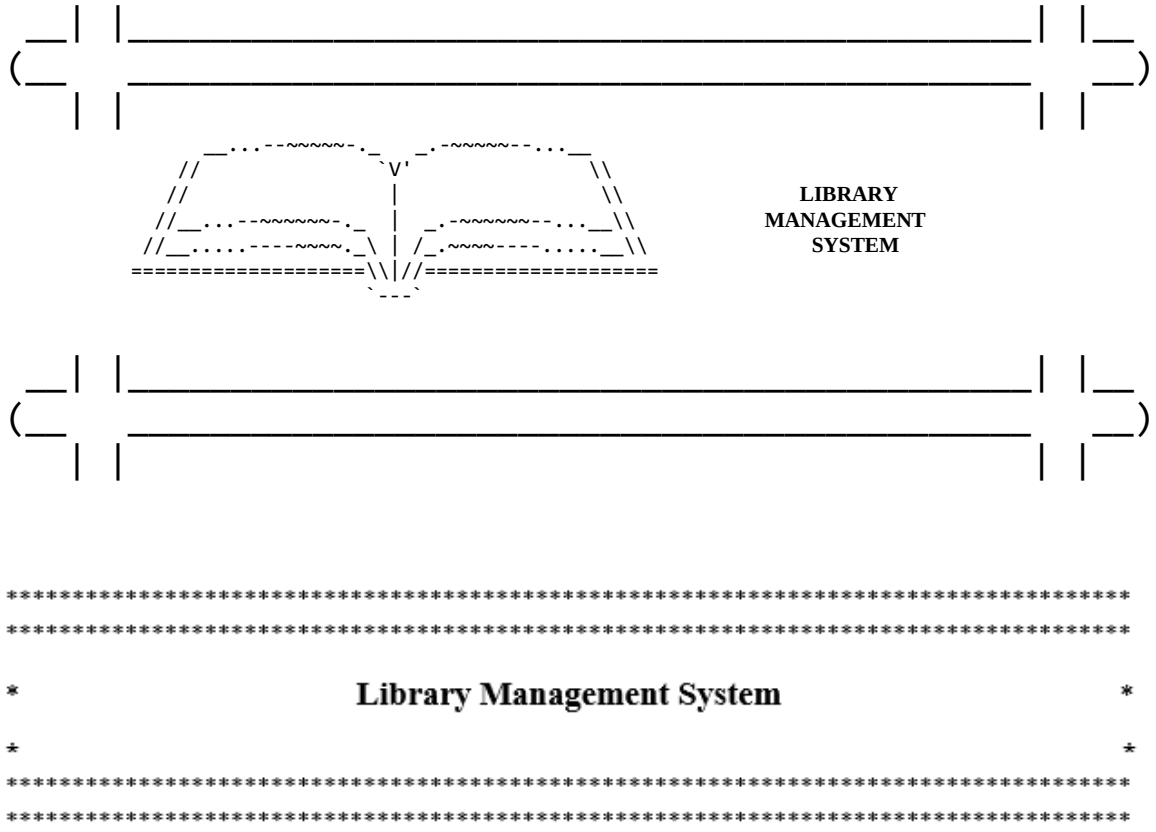


Enter your choice: 1

Enter your username: Zainab

Enter your password: 123

(1.2.2) FIGURE 12: User Sign Up Page

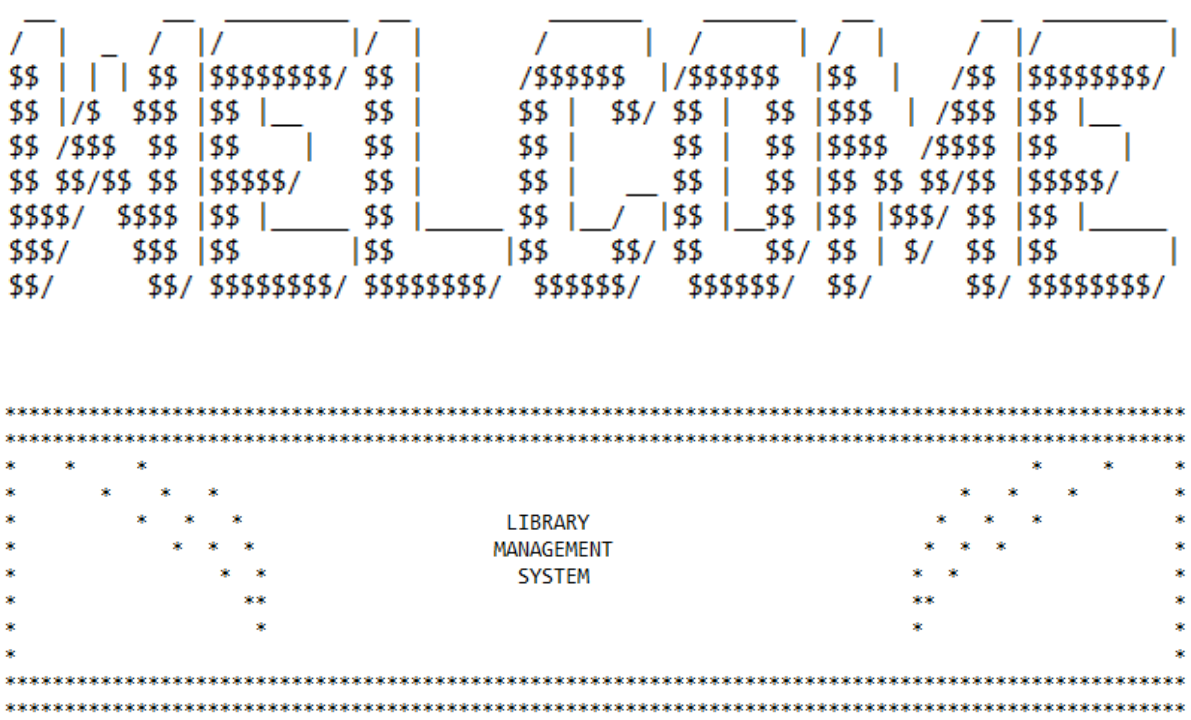


Enter your choice: 2

Enter your username: Zainab

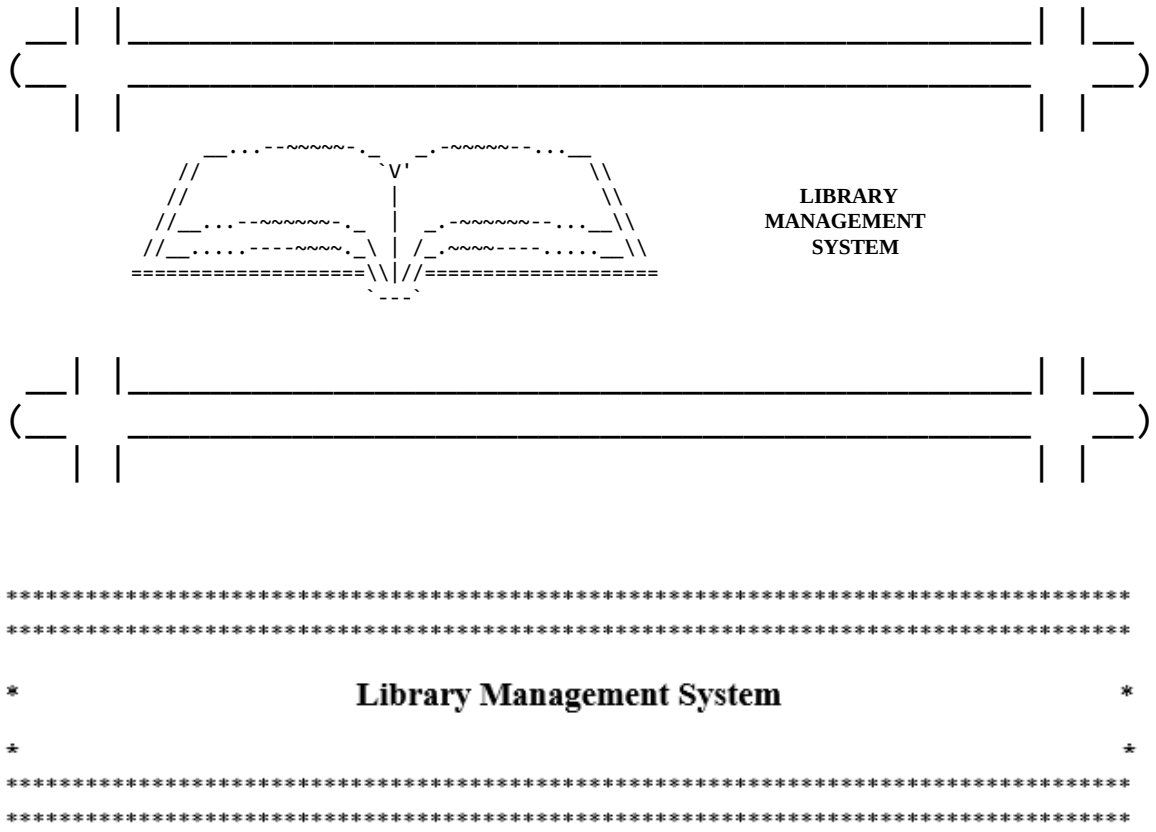
Enter your password: 123

(1.2.3) FIGURE 13: User Sign In Page

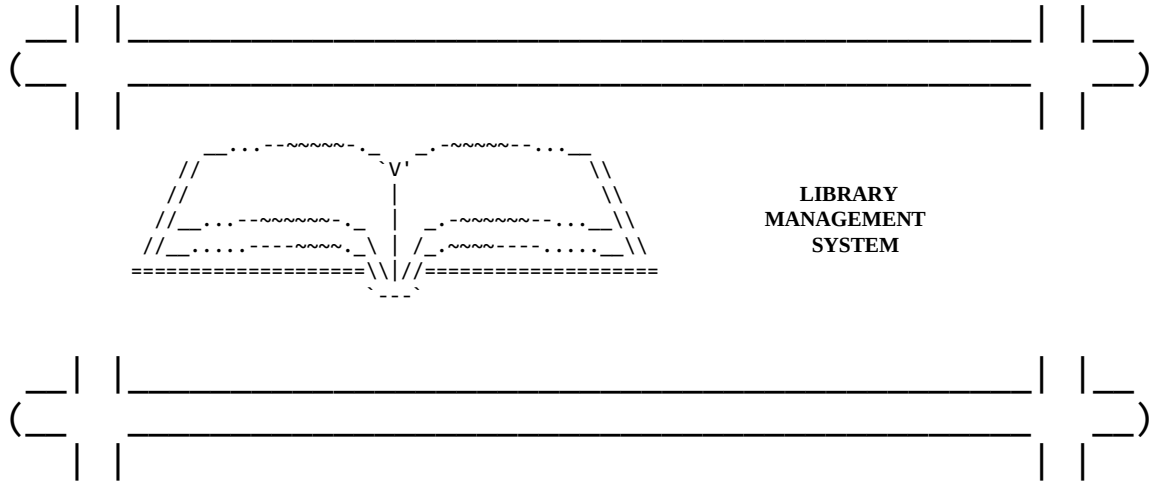


Press any key to continue….

(1.2.4) FIGURE 14: User Welcome Page



(1.2.5) FIGURE 15: User Menu Page




```
**********************************************************************************
**********************************************************************************
*                          Library Management System                             *
*                                                                                *
**********************************************************************************
**********************************************************************************
```

Enter your choice: 1

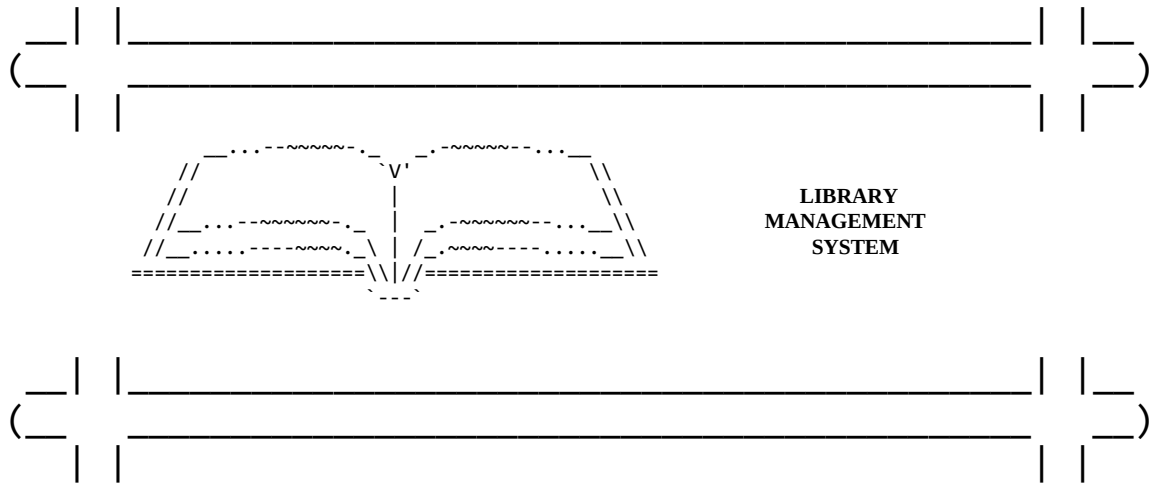
!!!!!!! WELCOME TO THE LIBRARY!!!!!!!!!!

The following books are present in our library:

0. The_Alchemist
1. The_Red_Pot
2. Peer_e_Kaamil
3. Jannat_k_Pattay
4. Harry_Potter_The_Series
5. Forty_Rules_of_Love

Press any key to continue….

(1.2.6) FIGURE 16: Books View Page




```
**********************************************************************************
**********************************************************************************
*                        Library Management System                               *
*                                                                                *
**********************************************************************************
**********************************************************************************
```

Enter your choice: 2

!!!!!!! WELCOME TO THE LIBRARY!!!!!!!!!!

The following books are present in our library:

0. The_Alchemist
1. The_Red_Pot
2. Peer_e_Kaamil
3. Jannat_k_Pattay
4. Harry_Potter_The_Series
5. Forty_Rules_of_Love

The price of each book is 120$.

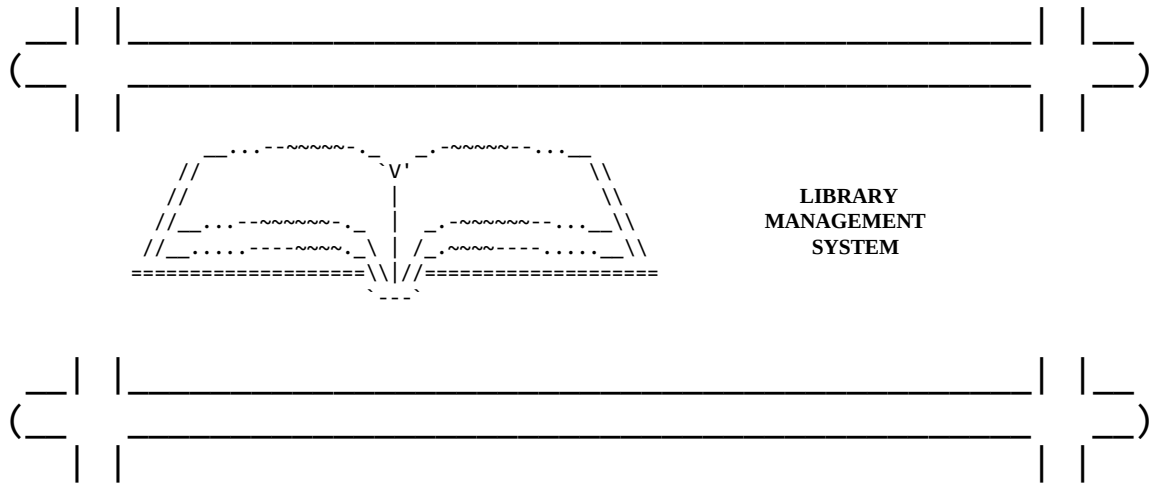
Enter the number of books you want to purchase: 2

Enter the name of the book that you want to purchase: The_Alchemist, The_Red_Pot

Your Book has been purchased for 240$!

Press any key to go continue…...

(1.2.7) FIGURE 17: Books Purchase Page



****
****

* **Library Management System** *

*
****
****

Enter your choice: 3

!!!!!!! WELCOME TO THE LIBRARY!!!!!!!!!!

The following books are present in our library:

0. The_Alchemist
1. The_Red_Pot
2. Peer_e_Kaamil
3. Jannat_k_Pattay
4. Harry_Potter_The_Series
5. Forty_Rules_of_Love

Enter the number of books that you want to Reserve: 1

Enter the name of the book that you want to Reserve: The_Alchemist

Your Book has been Reserved!

Press any key to continue…...

(1.2.8) FIGURE 18: Books Reserve Page

DATA STRUCTURES

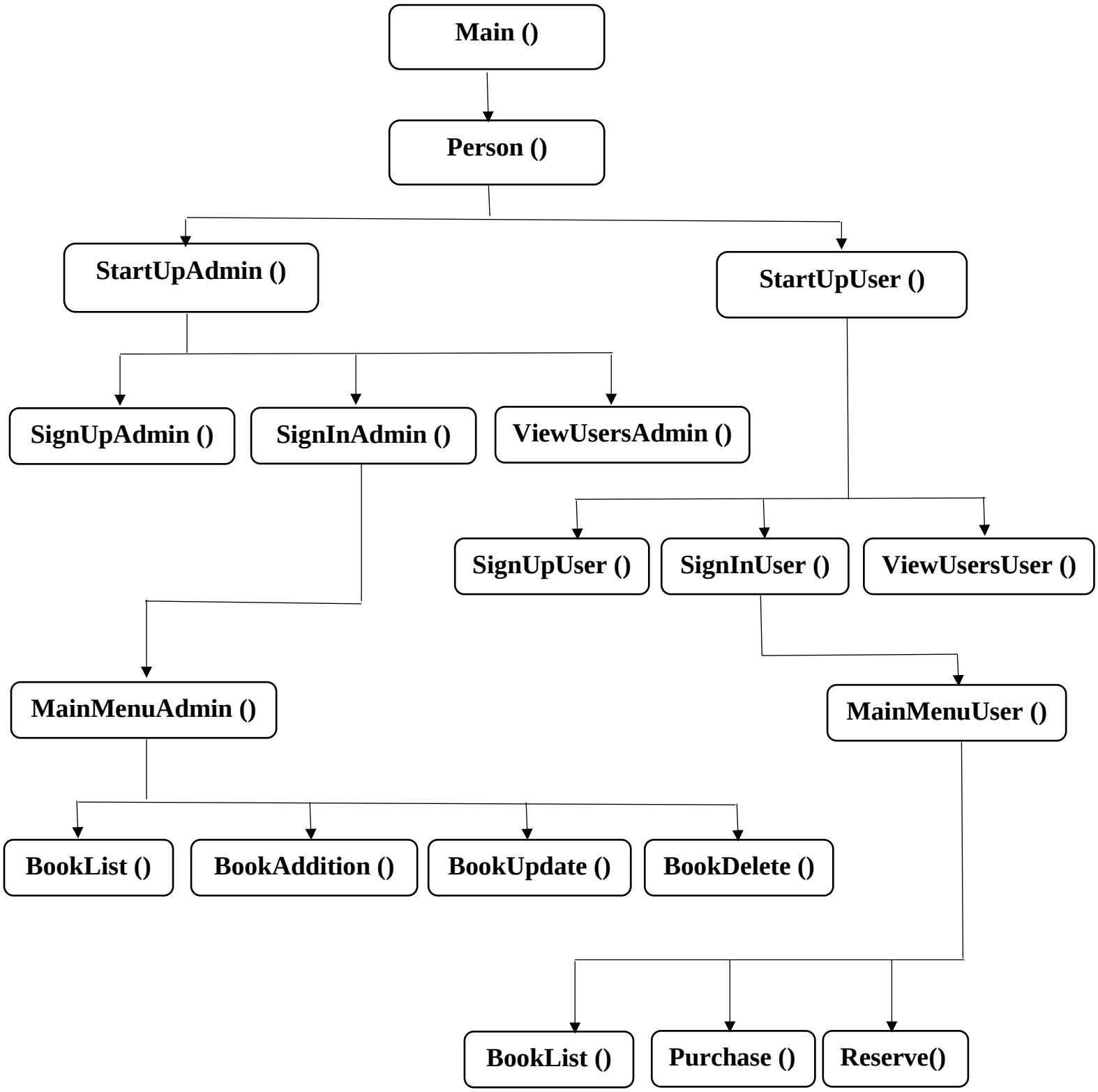
```
59
60  // Declaring the Global Variables
61
62  const int size = 17;
63  string password_arrayA[size];
64  string username_arrayA[size];
65
66  string password_arrayU[size];
67  string username_arrayU[size];
68  int userCount = 0;
69  int adminCount = 0;
70  int reservebookcount = 0;
71  const int SIZE = 10;
72
73  // Declaring the Global arrays
74
75  string books[SIZE] = {"The_Alchemist", "The_Red_Pot", "Peer_e_Kaamil", "Jannat_k_pattay",
76                        "Harry_Potter_The_Series", "Forty_Rules_of_love", "", "", "", ""};
77
78  string reserveBooks[SIZE] = {"", "", "", "", "", "", "", "", "", ""};
79
```

FUNCTION PROTOTYPES

```
 8
 9  // Giving prototypes for application Decoration functions
10  void welcomeScreen();
11  void header2();
12  void homepage();
13
14  // To ensure if the person entering the app is a user or an admin
15  void person();
16
17  // Giving prototypes For the admin
18
19  string startupAdmin();
20  bool signInAdmin(string userName, string password);
21  bool isValidUsernameAdmin(string userName);
22  void signUpArrayAdmin(string userName, string password);
23  void viewUsersAdmin();
24  void mainMenuAdmin();
25  void mainMenuAdminInput();
26  void booklist(int number);
27  void booklistWelcome();
28  void bookaddition(string book);
29  void booksAfterAddition();
30  void bookUpdate();
31  void booksAfterUpdation();
32  void bookDelete();
33  void booksAfterDeletion();
34
```

```
35  // Giving prototypes For the users
36
37  string startupUser();
38  bool signInUser(string userName, string password);
39  bool isValidUsernameUser(string userName);
40  void signUpArrayUser(string userName, string password);
41  void viewUsersUser();
42  void mainMenuUser();
43  void mainMenuUserInput();
44  void purchase();
45  void purchaseCalculate();
46  void reserve();
47
48  // Giving prototypes for file handling for both user and admin
49
50  void userstore(string userName, string password);
51  void adminstore(string userName, string password);
52  void userRead();
53  void adminRead();
54  void bookload();
55  void bookstore();
56  void bookReserveWrite();
57  void bookReserveRead();
58  string parsItems(string itemName, int itemRate);
59
```

FUNCTIONS WORKING FLOW



COMPLETE CODE OF LIBRARY MANAGEMENT SYSTEM

```
#include <iostream>
#include <conio.h>
#include <fstream>

// First we add libararies

using namespace std;

// Giving prototypes for application Decoration functions
void welcomeScreen();
void header2();
void homepage();

// To ensure if the person entering the app is a user or an admin
void person();

// Giving prototypes For the admin

string startupAdmin();
bool signInAdmin(string userName, string password);
bool isValidUsernameAdmin(string userName);
void signUpArrayAdmin(string userName, string password);
void viewUsersAdmin();
void mainMenuAdmin();
void mainMenuAdminInput();
void booklist(int number);
void booklistWelcome();
void bookaddition(string book);
void booksAfterAddition();
void bookUpdate();
void booksAfterUpdation();
void bookDelete();

void booksAfterDeletion();

// Giving prototypes For the users

string startupUser();
bool signInUser(string userName, string password);
bool isValidUsernameUser(string userName);
void signUpArrayUser(string userName, string password);
void viewUsersUser();
void mainMenuUser();
void mainMenuUserInput();
void purchase();
void purchaseCalculate();
void reserve();

// Giving prototypes for file handling for both user and admin
```

```
48  // Giving prototypes for file handling for both user and admin
49
50  void userstore(string userName, string password);
51  void adminstore(string userName, string password);
52  void userRead();
53  void adminRead();
54  void bookload();
55  void bookstore();
56  void bookReserveWrite();
57  void bookReserveRead();
58  string parsItems(string itemName, int itemRate);
59
60  // Declaring the Global Variables
61
62  const int size = 17;
63  string password_arrayA[size];
64  string username_arrayA[size];
65
66  string password_arrayU[size];
67  string username_arrayU[size];
68  int userCount = 0;
69  int adminCount = 0;
70  int reservebookcount = 0;
71  const int SIZE = 10;
72
73  // Declaring the Global arrays
74
75  string books[SIZE] = {"The_Alchemist", "The_Red_Pot", "Peer_e_Kaamil", "Jannat_k_pattay",
76                        "Harry_Potter_The_Series", "Forty_Rules_of_love", "", "", "", ""};
77
78  string reserveBooks[SIZE] = {"", "", "", "", "", "", "", "", "", ""};

79
80  main()
81  {
82      userRead();
83      adminRead();
84      bookReserveRead();
85      bookload();
86      system("cls");
87      welcomeScreen();
88      header2();
89      cout << "Press any number to continue......." << endl;
90      getch();
91      bool decision = 0;
92      string choice1 = "0";
93      string choice2 = "0";
94      string username;
95      string password;
96      string choice = "0";
```

```
97
98      // here 1 is to signUp , 2 is to signIn, 3 is to view , and 4 is to exit
99      while (choice != "3")
100
101     {
102         person();
103         cout << "Enter your choice: ";
104         cin >> choice;
105
106         // this will give an option to choose between using the application as an admin or as a us
107
108         system("cls");
109         welcomeScreen();
110         header2();
111
112         if (choice == "1")
113         {
114             while (choice1 != "4")
115
116             {
117                 choice1 = startupAdmin();
118                 system("cls");
119                 welcomeScreen();
120                 header2();
121                 string userName, password;
122
123                 if (choice1 == "1")
124
125                 // which is for signup
126
127                 {
128                     cout << "Enter your username: ";
```

```
128                   cout << "Enter your username: ";
129                   cin >> userName;
130
131                   cout << "Enter your password: ";
132                   cin >> password;
133
134                   decision = isValidUsernameAdmin(userName);
135
136                   if (decision == true)
137                   {
138                       adminstore(userName, password);
139                       signUpArrayAdmin(userName, password);
140                       cout << "User Created Successfully" << endl;
141                       adminstore(userName, password);
142                   }
143                   else
144                   {
145                       cout << "Username already exists, Try Again!" << endl;
146
147                       // The admin cannot sign up using the name that already exists in the application
148                       cout << "Press any number to continue......." << endl;
149                       getch();
150                   }
151               }
```

```
152
153             else if (choice1 == "2")
154
155             // which is for login
156
157             {
158                 cout << "Enter your username: ";
159                 cin >> userName;
160
161                 cout << "Enter your password: ";
162                 cin >> password;
163
164                 decision = signInAdmin(userName, password);
165
166                 if (decision == true)
167                 {
168                     cout << "Login is Successful" << endl;
169                     system("cls");
170                     homepage();
171                     cout << "Press any number to continue......." << endl;
172                     getch();
173                     mainMenuAdmin();
174                 }
175
176                 else
177
178                 {
179                     cout << "Invalid Credentials, Try again!" << endl;
180
181                     // this appears if the entered credentials are invalid
182                 }
183             }
```

```
184
185             else if (choice1 == "3")
186             {
187                 viewUsersAdmin();
188             }
189
190             else if (choice1 == "4")
191             {
192                 choice1 = "0";
193                 break;
194             }
195         }
196     }
197
198     else if (choice == "2")
199
200     // ************************** which is for the USER **********************************
201     {
202
203         system("cls");
204         welcomeScreen();
205
206         while (choice2 != "4")
207         {
208
```

```
209                 choice2 = startupUser();
210                 system("cls");
211                 welcomeScreen();
212                 header2();
213                 string userName, password;
214                 if (choice2 == "1") // for signup
215                 {
216                     cout << "Enter your username: ";
217                     cin >> userName;
218 
219                     cout << "Enter your password: ";
220                     cin >> password;
221 
222                     decision = isValidUsernameUser(userName);
223 
224                     if (decision == true)
225                     {
226                         signUpArrayUser(userName, password);
227 
228                         cout << "User Created Successfully" << endl;
229                         userstore(userName, password);
230                     }
231                     else
232                     {
233                         cout << "Username already exists, Try Again!" << endl;
234 
235                         // The admin cannot sign up using the name that already exists in the application
236                     }
237                 }
238 
239                 else if (choice2 == "2") // for signin
240                 {
241                     cout << "Enter your username: ";
242                     cin >> userName;
243 
244                     cout << "Enter your password: ";
245                     cin >> password;
246 
247                     decision = signInUser(userName, password);
248 
249                     if (decision == true)
250                     {
251                         cout << "Login is Successful" << endl;
252                         system("cls");
253                         homepage();
254                         cout << "Press any number to continue......." << endl;
255                         getch();
256                         mainMenuUser();
257                     }
258 
```

```
259                     else
260
261                     {
262                         cout << "Invalid Credentials, Try again!" << endl;
263
264                         // this appears if the entered credentials are invalid
265                     }
266                 }
267
268                 else if (choice2 == "3") // to view present users
269                 {
270                     viewUsersUser();
271                 }
272                 else if (choice2 == "4")
273                 {
274                     choice2 = "0";
275                     break;
276                 }
277             }
278         }
279         else if (choice == "3")
280         {
281             break;
282         }
283     }
284 }
285
286 // ****************************** ADMIN FUNCTIONS**************************************
287
288 void person()
289 {
290
291     cout << "Press (1) for admin: " << endl;
292
293     // if 1 is entered as the choice, the admin will use the application.
294
295     cout << "Press (2) for user: " << endl;
296
297     // if 2 is entered as the choice, the user will use the application.
298 }
299 string startupAdmin()
300 {
301
302     string choice1;
303
304     cout << "Press (1) to Sign UP: " << endl;
305
306     // When 1 is entered, it shows the menu for login.
307
308     cout << "Press (2) to Login: " << endl;
309
310     // When 2 is entered, it shows the menu for signup.
311
```

```
312     cout << "Press (3) to View users: " << endl;
313
314     // When 3 is entered, it shows the list of our present admins.
315
316     cout << "Press (4) to go back" << endl;
317
318     // When 0 is entered, it exits the application.
319
320     cout << "Enter your choice: ";
321     cin >> choice1;
322
323     return choice1;
324 }
325
326 bool signInAdmin(string userName, string password) // The function of sign in for the admin
327 {
328
329     bool result = false;
330
331     for (int x = 0; x < userCount; x++)
332     {
333         if (username_arrayA[x] == userName && password_arrayA[x] == password)
334
335         {
336             result = true;
337             break;
338         }
339     }
340     return result;
341 }
342
343 bool isValidUsernameAdmin(string userName) // The function to check if the username is valid, i.e it doesn't already exists
344 {
345     bool result = true;
346
347     for (int x = 0; x < userCount; x++)
348     {
349         if (username_arrayA[x] == userName)
350
351         {
352             result = false;
353             break;
354         }
355     }
356     return result;
357 }
358
359 void signUpArrayAdmin(string userName, string password) // The function for sign up for the admin
360 {
361     username_arrayA[userCount] = userName;
362     password_arrayA[userCount] = password;
363
364     userCount++;
365 }
```

```
366
367 void viewUsersAdmin() // The function to view the admins that are currently signed up in our application
368
369 {
370     cout << "Username"
371          << "\t"
372          << "Password" << endl
373          << endl;
374
375     for (int x = 0; x < size; x++)
376     {
377         cout << username_arrayA[x] << "\t\t" << password_arrayA[x] << endl;
378     }
379 }
380
381 void mainMenuAdmin() // The function to show the main menu of the admin
382 {
383     string choicee = "0"; // for the mainmenu
384     int number;
385     string book;
386     while (choicee != "5")
387     {
388         system("cls");
389         welcomeScreen();
390         header2();
391
392         mainMenuAdminInput();
393
394         cout << "Enter your choice: " << endl;
395         cin >> choicee;
396
397         if (choicee == "1")
398         {
399             system("cls");
400             welcomeScreen();
401             header2();
402             booklist(number);
403             cout << "Press any number to continue......." << endl;
404             getch();
405         }
406         else if (choicee == "2")
407         {
408             system("cls");
409             welcomeScreen();
410             header2();
411             bookaddition(book);
412             bookstore();
413             cout << "Press any number to continue......." << endl;
414             getch();
415         }
416         else if (choicee == "3")
417         {
418             system("cls");
419             welcomeScreen();
420             header2();
421             bookUpdate();
422             bookstore();
423             cout << "Press any number to continue......." << endl;
424             getch();
425         }
```

```
426         else if (choicee == "4")
427         {
428             system("cls");
429             welcomeScreen();
430             header2();
431             bookDelete();
432             bookstore();
433             cout << "Press any number to continue......." << endl;
434             getch();
435         }
436         else if (choicee == "5")
437         {
438             break;
439         }
440     }
441 }
442 void mainMenuAdminInput() // Function that displays the admin's menu on the console
443 {
444     cout << "Press (1) to View list of books" << endl;
445     cout << "Press (2) to Add a book" << endl;
446     cout << "Press (3) to Update a book" << endl;
447     cout << "Press (4) to Delete a book" << endl;
448     cout << "Press (5) to Logout" << endl;
449 }
450
451 void booklist(int number) // Function that displays the list of books already present in the library
452 {
453     int count = 6;
454     // there are already 6 books present in the library
455
456     booklistWelcome();
457
458     cout << "the lists of books is " << endl;
459     for (int x = 0; x < SIZE; x++)
460     {
461         if (books[x] != "")
462         {
463             cout << x + 1 << ": " << books[x] << endl;
464         }
465     }
466 }
467
468 void booklistWelcome()
469 {
470     cout << "                        WELCOME TO THE LIBRARY!                                    " << endl;
471     cout << "The following is the list of the books present in our library: " << endl;
472 }
473
474 void bookaddition(string book) // Function that adds the books in the library
475 {
476     int NumOfBookAdd;
477     int count = 6;
478
479     cout << "Enter the number of books you want to add: ";
480     cin >> NumOfBookAdd;
481
```

```
482     if (count > NumOfBookAdd && (count + NumOfBookAdd) < 10)
483     {
484         for (int x = count; x < (count + NumOfBookAdd); x++)
485         {
486             cout << "ADD the name of the book you want to add" << endl;
487             cin >> book;
488             // book added without spaces
489 
490             books[x] = book;
491             cout << "A new book " << book << "\t"
492                 << "has been add to the library." << endl;
493         }
494     }
495 
496     else if ((count + NumOfBookAdd) > 10)
497 
498     {
499         cout << "  only 10 books can be added to the library" << endl;
500     }
501 
502     booksAfterAddition();
503     getch();
504 }
505 void booksAfterAddition() // Function that displays the list of the books after the new book has been added to the library
506 {
507     for (int x = 0; x < SIZE; x++)
508     {
509         cout << x + 1 << ":" << books[x] << endl;
510     }
511 }
512 
513 void bookUpdate() // Function that updates the books in the library according to the admin
514 {
515     string book, newbook;
516 
517     int count = 6;
518 
519     cout << "the lists of books is " << endl;
520     for (int x = 0; x < SIZE; x++)
521     {
522         if (books[x] != "")
523         {
524             cout << x + 1 << ": " << books[x] << endl;
525         }
526     }
527 
528     cout << "Enter the book you want to update: " << endl;
529     cin >> book;
530 
531     for (int x = 0; x < SIZE; x++)
532     {
533         if (books[x] == book)
534         {
535             cout << "enter the updated name of the book" << endl;
536             cin >> newbook;
537             books[x] = newbook;
538             cout << "the name has been updated" << endl;
539             break;
540         }
```

```
541         else
542         {
543             if (x == SIZE)
544             {
545                 cout << "The book doesn't exist in the library!";
546                 bookUpdate();
547             }
548             else
549             {
550                 continue;
551             }
552         }
553     }
554
555     booksAfterUpdation();
556     getch();
557 }
558
559 void booksAfterUpdation() // Function that displays the books after updation
560 {
561     int count = 6;
562     cout << "the lists of books is " << endl;
563     for (int x = 0; x < SIZE; x++)
564     {
565         if (books[x] != "")
566         {
567             cout << x + 1 << ": " << books[x] << endl;
568         }
569     }
570 }
571
572 void bookDelete() // Function that deletes the books present in the library according to the admin
573 {
574     string book, deletebook;
575
576     int count = 6;
577
578     cout << "the lists of books is " << endl;
579     for (int x = 0; x < count; x++)
580     {
581
582         cout << x << ": " << books[x] << endl;
583     }
584
585     cout << "enter the name of the book you want to delete: " << endl;
586     cin >> deletebook;
587
588     for (int x = 0; x < SIZE; x++)
589     {
590         if (books[x] == deletebook)
591         {
592             books[x] = "";
593             cout << "the name has been deleted" << endl;
594             break;
595         }
596         else
597         {
598             if (x == SIZE)
599             {
600                 cout << "The book doesn't exist in the library!";
601                 bookUpdate();
602             }
```

```
603             else
604             {
605                 continue;
606             }
607         }
608     }
609     booksAfterDeletion();
610     getch();
611 }
612 
613 void booksAfterDeletion() // function that displays the books after deletion
614 {
615     int count = 6;
616 
617     cout << "the list of books after deleting the book is" << endl;
618 
619     for (int x = 0; x < count; x++)
620     {
621         if (books[x] != "")
622         {
623             cout << x << ": " << books[x] << endl;
624         }
625     }
626 }
627 
628 // ********************************* USER FUNCTIONS ***************************************
629 

630 string startupUser()
631 {
632     string choice1;
633 
634     cout << "Press (1) to Sign Up: " << endl;
635 
636     // When 1 is entered, it shows the menu for signup.
637 
638     cout << "Press (2) to Login: " << endl;
639 
640     // When 2 is entered, it shows the menu for login.
641 
642     cout << "Press (3) to View users: " << endl;
643 
644     // When 3 is entered, it shows the list of our present users.
645 
646     cout << "Press (4) to go back" << endl;
647 
648     // When 0 is entered, it exits the application.
649 
650     cout << "Enter your choice: ";
651     cin >> choice1;
652 
653     return choice1;
654 }
655 
656 bool signInUser(string userName, string password) // Function for signing in for the user
657 {
658 
659     bool result = false;
660 
661     for (int x = 0; x < userCount; x++)
```

```
    {
        if (username_arrayU[x] == userName && password_arrayU[x] == password)

        {
            result = true;
            break;
        }
    }
    return result;
}

bool isValidUsernameUser(string userName) // Function to check if the username entered is valid i.e it doesn't already exist
{
    bool result = true;

    for (int x = 0; x < userCount; x++)
    {
        if (username_arrayU[x] == userName)

        {
            result = false;
            break;
        }
    }
    return result;
}

void signUpArrayUser(string userName, string password) // Function to sign up for the user
{
    username_arrayU[userCount] = userName;
    password_arrayU[userCount] = password;


    userCount++;
}

void viewUsersUser() // Function to view users that are currently using the application

{
    cout << "Usernames"
         << "\t"
         << "Passwords" << endl
         << endl;

    for (int x = 0; x < size; x++)
    {
        if (username_arrayU[x] != " " && password_arrayU[x] != " ")
        {
            cout << username_arrayU[x] << "\t\t" << password_arrayU[x] << endl;
        }
    }
}

void mainMenuUser() // Function that displays the main menu for the user
{
    string choice3 = "0"; // used in mainmenu for user
    int number;


    while (choice3 != "4")
    {
        mainMenuUserInput();
        cout << "Enter your choice: " << endl;
        cin >> choice3;

        if (choice3 == "1")
        {
            system("cls");
            welcomeScreen();
            header2();
            booklist(number);
        }
```

```
732         else if (choice3 == "2")
733         {
734             system("cls");
735             header2();
736             purchase();
737         }
738         else if (choice3 == "3")
739         {
740             system("cls");
741             header2();
742             reserve();
743         }
744         else if (choice3 == "4")
745         {
746             break;
747         }
748     }
749 }
750 void mainMenuUserInput() // Function that displays the user's menu on console
751 {
752     cout << "Press (1) if you want to see the books list" << endl;
753     cout << "Press (2) if you want to Purchase a book from the library" << endl;
754     cout << "Press (3) if you want to Reserve a book from the library" << endl;
755     cout << "Press (4) to go back....." << endl;
756 }
757
758 void purchase() // Function for the user to purchase a book
759 {
760     int number;
761     string book;
762     int count = 6;
763
764     booklist(number);
765     purchaseCalculate();
766     cout << "Press any number to continue......." << endl;
767     getch();
768 }
769 void purchaseCalculate() // Function that calculates the price
770 {
771     string purchaseBook;
772     float price, number;
773     cout << "PRICE FOR EACH BOOK IS 120$." << endl;
774
775     cout << "Enter the number of books you want to purchase: " << endl;
776     cin >> number;
777
778     for (int x = 0; x < number; x++)
779     {
780         cout << "enter the name of the book you want to Purchase: " << endl;
781         cin >> purchaseBook;
782     }
783
784     price = number * 120;
785
786     cout << "YOUR BOOK HAS BEEN PURCHASED FOR: " << price << "$" << endl;
787 }
788
```

```
789  void reserve() // Function to reserve the book
790  {
791      string book;
792      int number;
793
794      cout << "Enter the number of books you want to reserve: " << endl;
795      cin >> number;
796
797      booklist(number);
798      for (int x = 0; x < number; x++)
799      {
800          cout << "Enter the name of the book you want to reserve: " << endl;
801          cin >> book;
802          reserveBooks[reservebookcount] = book;
803          reservebookcount++;
804      }
805      bookReserveWrite();
806      cout << "Your book has been reserved!" << endl;
807  }
808
809  void welcomeScreen() // Function that displays the header1 on the console as needed
810  {
811
812      cout << " __| |_____________________________________________________________________________| |__      " << endl;
813      cout << "(__   _____________________________________________________________________________   __)    " << endl;
814      cout << "   | |                                                                             | |      " << endl;
815      cout << "                __...--~~~~~-._   _.-~~~~~--...__                                    " << endl;
816      cout << "             //               `V'               \\           LIBRARY                 " << endl;
817      cout << "            //                 |                 \\         MANAGEMENT               " << endl;
818      cout << "           //__...--~~~~~~-._  |  _.-~~~~~~--...__\\          SYSTEM                  " << endl;
819      cout << "          //__.....----~~~~._  |  _.~~~~----.....__\\                                  " << endl;
820      cout << "         ====================\\\|//====================                                 " << endl;
821      cout << "                            `---`                                                    " << endl;
822      cout << " __| |_____________________________________________________________________________| |__      " << endl;
823      cout << "(__   _____________________________________________________________________________   __)    " << endl;
824      cout << "   | |                                                                             | |      " << endl
825          << endl;
826  }
827
828  void header2() // Function that displays the header2 on the console as needed
829  {
830
831      cout << " ********************************************************************************************************  " << endl;
832      cout << " ********************************************************************************************************  " << endl;
833      cout << " *                                                                                                    *  " << endl;
834      cout << " *                                  Library Management System                                         *  " << endl;
835      cout << " *                                                                                                    *  " << endl;
836      cout << " ********************************************************************************************************  " << endl;
837      cout << " ********************************************************************************************************  " << endl
838          << endl
839          << endl;
840  }
841
842  void homepage() // Function that displays the header3 on the console
843  {
844
845      cout << "  __       __  ________  __        ______    ______   __       __  ________   " << endl;
846      cout << " /  |  _  /  |/        |/  |      /      |  /      | /  |     /  |/        |  " << endl;
847      cout << " $$ | | | $$ |$$$$$$$$/ $$ |      /$$$$$$  |/$$$$$$  |$$ |   /$$ |$$$$$$$$/   " << endl;
848      cout << " $$ |/$  $$$ |$$ |__    $$ |      $$ |  $$/ $$ |  $$ |$$$  | /$$$ |$$ |__      " << endl;
849      cout << " $$ /$$$  $$ |$$    |   $$ |      $$ |      $$ |  $$ |$$$$  /$$$$ |$$    |     " << endl;
850      cout << " $$ $$/$$ $$ |$$$$$/    $$ |      $$ |   __ $$ |  $$ |$$ $$ $$/$$ |$$$$$/      " << endl;
851      cout << " $$$$/  $$$$ |$$ |_____ $$ |_____ $$ |__/  |$$ |__$$ |$$ |$$$/ $$ |$$ |_____   " << endl;
852      cout << " $$$/    $$$ |$$       |$$       |$$    $$/ $$    $$/ $$ | $/  $$ |$$       |  " << endl;
853      cout << " $$/      $$/ $$$$$$$$/ $$$$$$$$/  $$$$$$/   $$$$$$/  $$/      $$/ $$$$$$$$/   " << endl
854          << endl;
855
```

```
856     cout << " ********************************************************************************************  " << endl;
857     cout << " ********************************************************************************************  " << endl;
858     cout << " *    *    *                                                          *    *    *  " << endl;
859     cout << " *      *   *  *                                                    *  *   *      *  " << endl;
860     cout << " *         *  *  *                 LIBRARY                        *  *  *         *  " << endl;
861     cout << " *            * * *                MANAGEMENT                    * * *           *  " << endl;
862     cout << " *              * *                  SYSTEM                    * *             *  " << endl;
863     cout << " *                **                                          **               *  " << endl;
864     cout << " *                 *                                          *                *  " << endl;
865     cout << " *                                                                             *  " << endl;
866     cout << " ********************************************************************************************  " << endl;
867     cout << " ********************************************************************************************  " << endl
868         << endl
869         << endl;
870 }
871
872 // ************************************ File handling functions for both user and admin *********************************
873
```

```
874 void userstore(string userName, string password)
875 {
876     fstream user;
877     user.open("user1.txt", ios::app);
878     user << userName << ',' << password << endl;
879     user.close();
880 }
881
882 void adminstore(string userName, string password)
883 {
884     fstream admin;
885     admin.open("admin.txt", ios::app);
886     admin << userName << ',' << password << endl;
887     admin.close();
888 }
889
890 void userRead()
891 {
892     string word;
893     fstream user;
894     user.open("user1.txt", ios::in);
895     while (!user.eof())
896     {
897         getline(user, word);
898         if (word == "")
899             break;
900         username_arrayU[userCount] = parsItems(word, 1);
901         password_arrayU[userCount] = (parsItems(word, 2));
902         userCount = userCount + 1;
903     }
904     user.close();
```

```
905  }
906
907  void adminRead()
908  {
909      string word;
910      fstream admin;
911      admin.open("admin1.txt", ios::in);
912      while (!admin.eof())
913      {
914          getline(admin, word);
915          if (word == "")
916              break;
917          username_arrayA[adminCount] = parsItems(word, 1);
918          password_arrayA[adminCount] = (parsItems(word, 2));
919          adminCount = adminCount + 1;
920      }
921  }
922
923  string parsItems(string itemName, int itemRate)
924  {
925      int commaCount = 1;
926      string item;
927      for (int x = 0; x < itemName.length(); x++)
928      {
929          if (itemName[x] == ',')
930          {
931              commaCount = commaCount + 1;
932          }
```

```
933             else if (commaCount == itemRate)
934             {
935                 item = item + itemName[x];
936             }
937         }
938         return item;
939     }
940
941     void bookstore()
942     {
943         string word;
944         fstream book;
945         book.open("book.txt", ios::out);
946         for (int x = 0; x < SIZE; x++)
947         {
948             book << books[x] << endl;
949         }
950     }
951
952     void bookload()
953     {
954         int x = 0;
955         string word;
956         fstream book;
957         book.open("book.txt", ios::in);
958         while (!book.eof())
959         {
960             getline(book, word);
961             if (word == "")
962             {
963                 break;
964             }
965             books[x] = word;
966             x++;
967         }
968     }
969
970     void bookReserveWrite()
971     {
972         string word;
973         fstream reserveBook;
974         reserveBook.open("reserveBook.txt", ios::out);
975         for (int x = 0; x < SIZE; x++)
976         {
977             reserveBook << reserveBooks[x] << endl;
978         }
979     }
980
```

```
981  void bookReserveRead()
982  {
983      int x = 0;
984      string word;
985      fstream reserveBook;
986      reserveBook.open("reserveBook.txt", ios::in);
987      while (!reserveBook.eof())
988      {
989          getline(reserveBook, word);
990          if (word == "")
991          {
992              break;
993          }
994          reserveBooks[x] = word;
995          x++;
996      }
997  }
```

WEAKNESSES IN LIBRARY MANAGEMENT SYSTEM

- The name of the books must be entered without spaces which is a big weakness in this project
- The Purchasing function could have been better if it had more options
- The Reserving function is not elaborated enough
- Some of the functions are not obeying the single responsibility principle
- The actual code in the cpp file has the void functions as bool which were added later to the actual code.
- My data is split in different places

FUTURE DIRECTIONS

In the next semester, I want to focus more on making sure all the functions are obeying the single responsibility. I will cover the problems such as making sure my data is not split in different places so the code would not be messy. I will challenge myself by making a more complex business application with more functional requirements that covers all the weaknesses of my project.

RUBRICS

2022-CS-199 Zainab Idrees

	A-Extensive Evidence	**B-Convincing Evidence**	**C-Limited Evidence**	**D-No Evidence**
Documentation Formatting **Grade:**	All the documentation meets all the criteria.	Documentation is well formatted but some of the criteria is not fulfilled.	Documentation is required a lot of improvement.	Documentation is not Available
Documentation Formatting Criteria: In **Binder**, **Title** Page, **Header**-Footers, Font **Style**, Font **Size** all are all consistence and according to given **guidelines**. Project **Poster** is professionally design and well presented				
Documentation Contents **Grade:**	Documentation includes all of the criteria.	Documentation meet more than 80% of the criteria given.	Documentation meet more than 50% of the criteria.	When the documentation meet less than 50% of the criteria.
Documentation Contents Criteria: Title Page - **Table** of Contents - Project **Abstract** - **Functional** Requirements - **Wire** Frames –**Data Flow** Diagram-**Data** Structure (Arrays)-**Function** Headers and Description -Project **Code.** - **Weakness** in the Project and **Future** Directions. - **Conclusion** and What your **Learn** from the Project and Course and What is your **Future** Planning.				
Project Complexity **Grade:**	Project has at least 2 user's types and each user has at least 5 functionalities.	Project complexity meet 80% criteria given in extensive evidence	Project complexity meet 50% criteria given in extensive evidence	Project complexity meet less than 50% criteria given in extensive evidence
Code Style **Grade:**	All Code style criteria is followed	All code style criteria followed but some improvements required	lot of improvements required in coding style.	**Did not follow** code style,
Code Style Criteria: Consistent code style. Code is well indented. Variable and Function names are well defined. White Spaces are well used. Comments are added.				
Code Documentation Mapping **Grade:**	Code and documentation is synchronized.	Code and documentation does not synchronized at **some** places	Code and documentation does not synchronized at **many** places	Code and documentation **does not** synchronized.
Data Structure (Arrays) **Grade:**	Data structure is sufficient for the project requirements	Data Structure is sufficient but require improvement to meet project requirements.	Data structure is not sufficient and need a lot of improvement	Data Structure is not properly identified and declared.
Modularity **Grade:**	Meet all Modularity criteria	Meet all Modularity criteria but at some places it is missing	Do not sufficiently meet the modularity criteria.	No modularity or very minimum modularity.
Modularity criteria: Functions are defined for each major feature. Functions are independent (identify from parameter list and return types).				
Validations **Grade:**	Validations on all number type inputs are applied	Validations are applied but at some places it is missing.	Validations are missing at lot of places	No Validations are used
File Handling **Grade:**	Separate files for separate data. Data in csv format	File handing require some improvements	File handing require a lot of improvements	Not implemented
Aesthetics of the User Interface **Grade:**	UI is presentable. Proper coloring, Headers and clear screen is done	UI require some improvements	UI require a lot of improvements	Not implemented
Presentation and Demo **Grade:**	Presentation and Demo was 100% working	Presentation and Demo require some improvements	Presentation and Demo require a lot of improvements	Presentation was not ok and Demo was not working
Student Understanding with the Code. **Grade:**	Student has complete understanding how the code is working and knows the concept.	Student has good understand but some place he does not know the concepts	Student has a very little understand and lack the major concepts.	Student does not have any level of understanding of the code.

2022-CS-199 Zainab Idrees

	A-Extensive Evidence	**B-Convincing Evidence**	**C-Limited Evidence**	**D-No Evidence**
Documentation Formatting **Grade:**	All the documentation meets all the criteria.	Documentation is well formatted but some of the criteria is not fulfilled.	Documentation is required a lot of improvement.	Documentation is not Available
Documentation Formatting Criteria: In **Binder**, **Title** Page, **Header**-Footers, Font **Style**, Font **Size** all are all consistence and according to given **guidelines**. Project **Poster** is professionally design and well presented				
Documentation Contents **Grade:**	Documentation includes all of the criteria.	Documentation meet more than 80% of the criteria given.	Documentation meet more than 50% of the criteria.	When the documentation meet less than 50% of the criteria.
Documentation Contents Criteria: Title Page - **Table** of Contents - Project **Abstract** - **Functional** Requirements - **Wire** Frames –**Data Flow** Diagram-**Data** Structure (Arrays)-**Function** Headers and Description -Project **Code.** - **Weakness** in the Project and **Future** Directions. - **Conclusion** and What your **Learn** from the Project and Course and What is your **Future** Planning.				
Project Complexity **Grade:**	Project has at least 2 user's types and each user has at least 5 functionalities.	Project complexity meet 80% criteria given in extensive evidence	Project complexity meet 50% criteria given in extensive evidence	Project complexity meet less than 50% criteria given in extensive evidence
Code Style **Grade:**	All Code style criteria is followed	All code style criteria followed but some improvements required	lot of improvements required in coding style.	**Did not follow** code style,
Code Style Criteria: Consistent code style. Code is well indented. Variable and Function names are well defined. White Spaces are well used. Comments are added.				
Code Documentation Mapping **Grade:**	Code and documentation is synchronized.	Code and documentation does not synchronized at **some** places	Code and documentation does not synchronized at **many** places	Code and documentation **does not** synchronized.
Data Structure (Arrays) **Grade:**	Data structure is sufficient for the project requirements	Data Structure is sufficient but require improvement to meet project requirements.	Data structure is not sufficient and need a lot of improvement	Data Structure is not properly identified and declared.
Modularity **Grade:**	Meet all Modularity criteria	Meet all Modularity criteria but at some places it is missing	Do not sufficiently meet the modularity criteria.	No modularity or very minimum modularity.
Modularity criteria: Functions are defined for each major feature. Functions are independent (identify from parameter list and return types).				
Validations **Grade:**	Validations on all number type inputs are applied	Validations are applied but at some places it is missing.	Validations are missing at lot of places	No Validations are used
File Handling **Grade:**	Separate files for separate data. Data in csv format	File handing require some improvements	File handing require a lot of improvements	Not implemented
Aesthetics of the User Interface **Grade:**	UI is presentable. Proper coloring, Headers and clear screen is done	UI require some improvements	UI require a lot of improvements	Not implemented
Presentation and Demo **Grade:**	Presentation and Demo was 100% working	Presentation and Demo require some improvements	Presentation and Demo require a lot of improvements	Presentation was not ok and Demo was not working
Student Understanding with the Code. **Grade:**	Student has complete understanding how the code is working and knows the concept.	Student has good understand but some place he does not know the concepts	Student has a very little understand and lack the major concepts.	Student does not have any level of understanding of the code.