



Data Algorithm
Report
CSE: 2117

Submission: 13 Sept 2020

Library Management System

Submitted by
The Celestial Knights

Submitted to
Md. Saidur Rahman Kohinoor

Lecturer
Department of CSE

Contents

1. Introductions

- 1.1 Purpose
- 1.2 Project Scope
- 1.3 Project Goals

2. Description

2.1 Product Features

- SignUp\LogIn
- Borrow
- Donate
- Game

2.2 Functions\Environments Used

- File
- Map
- String
- Return

2.3 Designing and Implementations

- GotoXY
- Basic Coloring

2.4 Recreational

- A simple game of Cricket

3. Group Members

- Md. Hridoy Chowdhury (ID-1912020075)
- Naimur Rashid Rahib (ID-1912020052)
- Nakib Islam Chowdhury (ID-1912020056)
- Mahbuba Khanom (ID-1912020085)
- Umma Oyra Shimu (ID-1912020089)
- Tanjina Islam (ID-1912020062)

4. User Interface

4.1 Inputs and Outputs

-ScreenShots of actual program

5. **Reference**

6. **Conclusion**

1. Introduction

Library Management System is a software-based program which enables its users to donate and borrow books to and from one another. It is developed in order to create a platform which will help giving and taking of books a lot easier. An accurate implementation of a working library can be conducted using this program.

1.1 Purpose

The Library Management is developed with the intention to help users of this program to keep track of the books and other library related things.

We all know the struggles of a new student when he first admits into an university. Buying course related books every semester is very costly. After four months, we can give away these books to juniors who would need to buy these books. So, we tried to solve this problem by developing a program in which you can donate books and others in need can borrow it.

1.2 Project Scope

The Library Management System is not only going ease the pain of buying new books every semester but also going to help in-need students, to get the books for free which can surely save a lot of money. And this program keeps privacy because you can directly see the available books in the borrowing section with the contact information of the person who is donating.

1.3 Project Goals

1. To ease the pain of borrowing books.
2. To help the in-need students by letting them borrow books for free.
3. To ease the process of donating a book.

2. Product Description

2.1 Product Features

SignUp\LogIn –

The **SignUp** feature will help you to make your own profile with some of your Basic information. And the **LogIn** feature will enable you to log into your profile where if you want you can donate books or check out all the available books for borrow. During signup you will be asked to give your name, email, mobile number. During login the program will ask you about the information you gave earlier if they match, only then you can login to your profile. This will protect your privacy and information.

Donate

The **Donate** feature can be used to donate a book. When an user donates a book then the name of his donated book(s) will be added to the available booklist.

Borrow

The **Borrow** feature contains 2 important parts.

List of all available books

This feature would allow you to see all the available books that you can borrow. Even if you don't know the name of the book, you can just look for it in the list.

Search

This feature just directly enables you to search for the book you are looking. If this book is available for borrowing, you will find the list of people who are donating this book along with their contact number.

2.2 Functions/Environments used

We wrote the whole using the IDE named, **Codeblocks**. Multiple functions is used in this program and particular functions was called to do particular jobs.

File

File is basically used in a program for data collection. It was also used here for the same purpose. At first a file was made which will contain the name of all the books which are available for borrowing. Now every time a book got listed for donation. A file of the same name as the book would be built. Which contains the name and contact number of its donor.

Map

Map is used to keep count of elements in a program. In our project map was used to count the number of book(s) available in the inventory.

When user donates a book(s) the name of that book is added in our previously built file which contains all available books name. When this book name is added in the list we use a (+) sign beside it. Which means a book of this name has been added in the list. Similarly, if a book is borrowed and the donor confirms it, we also put this book(s) name in to the list but this time with (-) sign.

String

String is a function which is used to input information from the user. Here, we used strings to take information from the user like, user's name, E-mail, contact number, passwords and book names as well.

Getline() function was used when given string contains more than one word.

Return

Return functions was used so that user can smoothly go back and forth from his current displayed page.

2.3 Designing and Implementations

GotoXY

The function **gotoXY** is used to declare the coordinates X and Y in the console where positions of all the outputs are determined. We used gotoXY to print the outputs in an particular position in the console, so that it looks neat and gives a structure.

Basic Coloring

We used some colors just to give the program a different and give it a little taste. Basic coloring function in the C-language library was used. The exact color which was used is '5F', which tells the console to print the background 'Purple' and the text in the foreground 'Bright White'.

2.4 Recreational

A simple game of Cricket

Favorite game of our Team Leader Md. Hridoy Chowdhury, Cricket was added just give our own little taste. It is actually a barebone version the game where you can play by giving inputs through your keyboard. The rules of the game is provided inside the program.

3. Members Contribution

Md. Hridoy Chowdhury

Being a team leader, I had to outline the project task first and assign each task to my team members. Collecting and evaluating their task was part of my job. I had to debug some of their code and make it smooth and more optimized for the project. Many of the time I had to give Logical support to my team members.

The later part of the game was created by me. Where I had to make it more dynamic. If user loses toss, then computer automatically decides who would bat/ball first.

Users and computers scores were kept in an array. After end of 2nd innings these scores were compared to declare the Winner!!

Naimur Rashid Rahib

He was Given the designing and Implementation sector. He used gotoXY and basic coloring to do so. He also built the Donation function, where users give the name of the book he wants to donate. The name of this book is taken and are being listed on the Book list.

Nakib Islam Chowdhury

He created the SignUp function. For that user basic information were taken from him and then it was kept on a file which was built with the user name. He was also given the task to use map and then show which books are available in what numbers.

Mahbuba Khanom

She was given the task to create the Log In function. Earlier when User signed up, he gave his information. To log in he was asked to give his name, email and password of the account.

Earlier the file which was created by the user name is opened and his earlier given email and password is now checked with the email and password that he gave now to log in. If Email and password matches user can successfully log in.

Since she already knows how to read a file she was given the task to read and show all the available book list.

Umma Oyra Shimu

She was given the task to build the Edit Function. This also asks the user to give all necessary Information. But instead of creating a new file using the user name, this time the already existing file is appended

Tanjina Islam

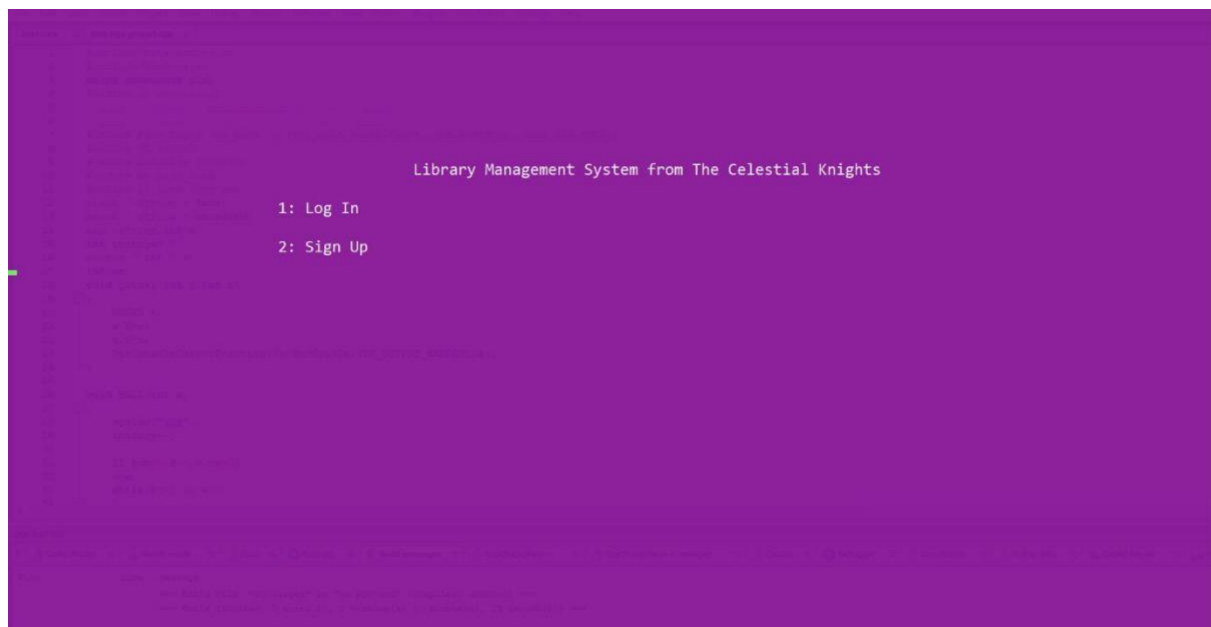
She was given the task to finish the initial part of the game. Which requires toss and Choosing to “BAT or Ball”. For this task she created a new function and used rand() function for generating random numbers.

4. User Interface

4.1 Inputs and outputs

ScreenShots of the actual program

SignUp/LogIn Page:



Sign up:

[illegible]**LogIn:**

```

1 #!/usr/bin/perl
2
3 use strict;
4 use warnings;
5
6 my $username = "root";
7 my $password = "root";
8
9 my $url = "http://10.10.10.10:8080/";
10
11 my $login_url = $url . "login.php";
12
13 my $login_data = {
14     "username" => $username,
15     "password" => $password,
16 };
17
18 my $login_headers = {
19     "Content-Type" => "application/x-www-form-urlencoded",
20 };
21
22 my $login_response = $curl -X POST -d @$login_data -H @$login_headers $login_url;
23
24 my $login_status = $login_response =~ /login\.php/;
25
26 if ($login_status) {
27     print "Login successful!\n";
28 } else {
29     print "Login failed!\n";
30 }
31
32 my $url = "http://10.10.10.10:8080/";
33
34 my $url_data = {
35     "url" => $url,
36 };
37
38 my $url_headers = {
39     "Content-Type" => "application/x-www-form-urlencoded",
40 };
41
42 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
43
44 my $url_status = $url_response =~ /url\.php/;
45
46 if ($url_status) {
47     print "URL request successful!\n";
48 } else {
49     print "URL request failed!\n";
50 }
51
52 my $url_data = {
53     "url" => $url,
54 };
55
56 my $url_headers = {
57     "Content-Type" => "application/x-www-form-urlencoded",
58 };
59
60 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
61
62 my $url_status = $url_response =~ /url\.php/;
63
64 if ($url_status) {
65     print "URL request successful!\n";
66 } else {
67     print "URL request failed!\n";
68 }
69
70 my $url_data = {
71     "url" => $url,
72 };
73
74 my $url_headers = {
75     "Content-Type" => "application/x-www-form-urlencoded",
76 };
77
78 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
79
80 my $url_status = $url_response =~ /url\.php/;
81
82 if ($url_status) {
83     print "URL request successful!\n";
84 } else {
85     print "URL request failed!\n";
86 }
87
88 my $url_data = {
89     "url" => $url,
90 };
91
92 my $url_headers = {
93     "Content-Type" => "application/x-www-form-urlencoded",
94 };
95
96 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
97
98 my $url_status = $url_response =~ /url\.php/;
99
100 if ($url_status) {
101     print "URL request successful!\n";
102 } else {
103     print "URL request failed!\n";
104 }
105
106 my $url_data = {
107     "url" => $url,
108 };
109
110 my $url_headers = {
111     "Content-Type" => "application/x-www-form-urlencoded",
112 };
113
114 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
115
116 my $url_status = $url_response =~ /url\.php/;
117
118 if ($url_status) {
119     print "URL request successful!\n";
120 } else {
121     print "URL request failed!\n";
122 }
123
124 my $url_data = {
125     "url" => $url,
126 };
127
128 my $url_headers = {
129     "Content-Type" => "application/x-www-form-urlencoded",
130 };
131
132 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
133
134 my $url_status = $url_response =~ /url\.php/;
135
136 if ($url_status) {
137     print "URL request successful!\n";
138 } else {
139     print "URL request failed!\n";
140 }
141
142 my $url_data = {
143     "url" => $url,
144 };
145
146 my $url_headers = {
147     "Content-Type" => "application/x-www-form-urlencoded",
148 };
149
150 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
151
152 my $url_status = $url_response =~ /url\.php/;
153
154 if ($url_status) {
155     print "URL request successful!\n";
156 } else {
157     print "URL request failed!\n";
158 }
159
160 my $url_data = {
161     "url" => $url,
162 };
163
164 my $url_headers = {
165     "Content-Type" => "application/x-www-form-urlencoded",
166 };
167
168 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
169
170 my $url_status = $url_response =~ /url\.php/;
171
172 if ($url_status) {
173     print "URL request successful!\n";
174 } else {
175     print "URL request failed!\n";
176 }
177
178 my $url_data = {
179     "url" => $url,
180 };
181
182 my $url_headers = {
183     "Content-Type" => "application/x-www-form-urlencoded",
184 };
185
186 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
187
188 my $url_status = $url_response =~ /url\.php/;
189
190 if ($url_status) {
191     print "URL request successful!\n";
192 } else {
193     print "URL request failed!\n";
194 }
195
196 my $url_data = {
197     "url" => $url,
198 };
199
200 my $url_headers = {
201     "Content-Type" => "application/x-www-form-urlencoded",
202 };
203
204 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
205
206 my $url_status = $url_response =~ /url\.php/;
207
208 if ($url_status) {
209     print "URL request successful!\n";
210 } else {
211     print "URL request failed!\n";
212 }
213
214 my $url_data = {
215     "url" => $url,
216 };
217
218 my $url_headers = {
219     "Content-Type" => "application/x-www-form-urlencoded",
220 };
221
222 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
223
224 my $url_status = $url_response =~ /url\.php/;
225
226 if ($url_status) {
227     print "URL request successful!\n";
228 } else {
229     print "URL request failed!\n";
230 }
231
232 my $url_data = {
233     "url" => $url,
234 };
235
236 my $url_headers = {
237     "Content-Type" => "application/x-www-form-urlencoded",
238 };
239
240 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
241
242 my $url_status = $url_response =~ /url\.php/;
243
244 if ($url_status) {
245     print "URL request successful!\n";
246 } else {
247     print "URL request failed!\n";
248 }
249
250 my $url_data = {
251     "url" => $url,
252 };
253
254 my $url_headers = {
255     "Content-Type" => "application/x-www-form-urlencoded",
256 };
257
258 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
259
260 my $url_status = $url_response =~ /url\.php/;
261
262 if ($url_status) {
263     print "URL request successful!\n";
264 } else {
265     print "URL request failed!\n";
266 }
267
268 my $url_data = {
269     "url" => $url,
270 };
271
272 my $url_headers = {
273     "Content-Type" => "application/x-www-form-urlencoded",
274 };
275
276 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
277
278 my $url_status = $url_response =~ /url\.php/;
279
280 if ($url_status) {
281     print "URL request successful!\n";
282 } else {
283     print "URL request failed!\n";
284 }
285
286 my $url_data = {
287     "url" => $url,
288 };
289
290 my $url_headers = {
291     "Content-Type" => "application/x-www-form-urlencoded",
292 };
293
294 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
295
296 my $url_status = $url_response =~ /url\.php/;
297
298 if ($url_status) {
299     print "URL request successful!\n";
300 } else {
301     print "URL request failed!\n";
302 }
303
304 my $url_data = {
305     "url" => $url,
306 };
307
308 my $url_headers = {
309     "Content-Type" => "application/x-www-form-urlencoded",
310 };
311
312 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
313
314 my $url_status = $url_response =~ /url\.php/;
315
316 if ($url_status) {
317     print "URL request successful!\n";
318 } else {
319     print "URL request failed!\n";
320 }
321
322 my $url_data = {
323     "url" => $url,
324 };
325
326 my $url_headers = {
327     "Content-Type" => "application/x-www-form-urlencoded",
328 };
329
330 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
331
332 my $url_status = $url_response =~ /url\.php/;
333
334 if ($url_status) {
335     print "URL request successful!\n";
336 } else {
337     print "URL request failed!\n";
338 }
339
340 my $url_data = {
341     "url" => $url,
342 };
343
344 my $url_headers = {
345     "Content-Type" => "application/x-www-form-urlencoded",
346 };
347
348 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
349
350 my $url_status = $url_response =~ /url\.php/;
351
352 if ($url_status) {
353     print "URL request successful!\n";
354 } else {
355     print "URL request failed!\n";
356 }
357
358 my $url_data = {
359     "url" => $url,
360 };
361
362 my $url_headers = {
363     "Content-Type" => "application/x-www-form-urlencoded",
364 };
365
366 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
367
368 my $url_status = $url_response =~ /url\.php/;
369
370 if ($url_status) {
371     print "URL request successful!\n";
372 } else {
373     print "URL request failed!\n";
374 }
375
376 my $url_data = {
377     "url" => $url,
378 };
379
380 my $url_headers = {
381     "Content-Type" => "application/x-www-form-urlencoded",
382 };
383
384 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
385
386 my $url_status = $url_response =~ /url\.php/;
387
388 if ($url_status) {
389     print "URL request successful!\n";
390 } else {
391     print "URL request failed!\n";
392 }
393
394 my $url_data = {
395     "url" => $url,
396 };
397
398 my $url_headers = {
399     "Content-Type" => "application/x-www-form-urlencoded",
400 };
401
402 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
403
404 my $url_status = $url_response =~ /url\.php/;
405
406 if ($url_status) {
407     print "URL request successful!\n";
408 } else {
409     print "URL request failed!\n";
410 }
411
412 my $url_data = {
413     "url" => $url,
414 };
415
416 my $url_headers = {
417     "Content-Type" => "application/x-www-form-urlencoded",
418 };
419
420 my $url_response = $curl -X POST -d @$url_data -H @$url_headers $url;
421
422 my $url_status = $url_response =~ /url\.php/;
423
424 if ($url_status) {
425     print "URL request successful!\n";
426
```


Main Menu:

```
1 #!/usr/bin/perl
2
3 use strict;
4 use warnings;
5
6 my $username = "Rahib";
7 my $password = "123456789";
8
9 my $username_in = "Username: ";
10 my $password_in = "Password: ";
11
12 my $username_out = "Welcome $username!";
13
14 my $choice;
15
16 while (1) {
17     print "1: Borrow Book\n";
18     print "2: Donate Book\n";
19     print "3: Confirm Donation(If someone has taken book from you,let us know)\n";
20     print "4: Edit Info\n";
21     print "5: Game\n";
22
23     $choice = <STDIN>;
24
25     if ($choice == 1) {
26         # Borrow Book
27     }
28     elsif ($choice == 2) {
29         # Donate Book
30     }
31     elsif ($choice == 3) {
32         # Confirm Donation
33     }
34     elsif ($choice == 4) {
35         # Edit Info
36     }
37     elsif ($choice == 5) {
38         # Game
39     }
40     else {
41         print "Invalid choice\n";
42     }
43 }
```

Borrow:

```
1 #!/usr/bin/perl
2
3 use strict;
4 use warnings;
5
6 my $username = "Rahib";
7 my $password = "123456789";
8
9 my $username_in = "Username: ";
10 my $password_in = "Password: ";
11
12 my $username_out = "Welcome $username!";
13
14 my $choice;
15
16 while (1) {
17     print "1: Check All Available Books\n";
18     print "2: Borrow a Book\n";
19     print "3: Back\n";
20
21     $choice = <STDIN>;
22
23     if ($choice == 1) {
24         # Check All Available Books
25     }
26     elsif ($choice == 2) {
27         # Borrow a Book
28     }
29     elsif ($choice == 3) {
30         # Back
31     }
32     else {
33         print "Invalid choice\n";
34     }
35 }
```

Donate:

[illegible]

Game:

A screenshot of a web browser displaying a simple web page. The browser's address bar shows a local file path. The page has a blue header with the text "Welcome to our Sports Arena". Below the header is a white main content area with a blue border. Inside this area, there is a list of links: "1: Play", "2: How to Play", and "3: Return Home". Below the links is a paragraph of text: "This is a simple web page. It is designed to be easy to use and navigate. It contains a list of links and a paragraph of text." The browser's status bar at the bottom shows "Done!".

6. Conclusion

This whole project was started with the intention to help and ease our problems of getting new books. We all face the problems of buying books that we need only for a limited amount of time. This project was aimed towards solving those problems and to minimize the efforts of borrowing a book. If we can use this **Library Management System** to its full potential then we should meet the goal that we set in the beginning of this project. Thank You for believing in us. I hope we did well. “Stay home, Stay safe” during this epidemic.

5. Reference

File Tutorial;

https://www.youtube.com/playlist?list=PLm6UpFb35TJ5ntNKXyPWinCx_tqf8V5jGn

C programme to change Output text and background colour

https://youtu.be/q_jRca7woxE

C++ GotoXY function

<https://youtu.be/fv54E9kZUvA>

Random Number Generator

[https://stackoverflow.com/questions/822323/how-to-generate-a-random-int-in-c#:~:text=The%20rand\(\)%20function%20in,seed\)%20to%20set%20a%20seed](https://stackoverflow.com/questions/822323/how-to-generate-a-random-int-in-c#:~:text=The%20rand()%20function%20in,seed)%20to%20set%20a%20seed)