

UNIVERSITY OF GUJRAT



A WORLD CLASS UNIVERSITY

Project: Search Engine

Object Oriented Programming

Submitted to

Mr. Naveed Abbas

Submitted By

Name: M Hanzala Zaheer

Roll no: 22014198-060

Name: Gul e Raana

Roll no: 22014198-049

Name: Iqra Aslam

Roll no: 22014198-044

Class: BSSE-A-22

Course code: SE-103

Submit Date: 1 /08/2023

“SOURCE CODE”

```
1  #include <iostream>
2  #include <istream>
3  #include <fstream>
4  #include <stdlib.h>
5  #include <string.h>
6  #include <conio.h>
7  #include <vector>
8  #include <iomanip>
9  using namespace std;
10 //all function used in program
11 void login();
12 void registr();
13 void forgot();
14 void menu();
15 void search_engine_page();
16 void search();
17 void add_keywords();
18 void check_keywords();
19 void Edit_keyword();
20 void project_page();
21 //global variables
22 char ch;
23 int i , count = 0;
24 //main function
25 int main()
26 {
27     if (count == 1)
28     {
29         search_engine_page();
30     }
31     else{
32         project_page();
33         menu();
34     }
35     return 0;
36 }
```

```

37
38 void project_page()
39 {
40     cout << right << endl
41         << endl;
42     cout << setw(30) << " "
43         << "_____ \n";
44     cout << setw(30) << " "
45         << "/" \n";
46     cout << setw(30) << " "
47         << "/" Project: SEARCH ENGINE \n";
48     cout << setw(30) << " "
49         << "/" \n";
50     cout << setw(30) << " "
51         << "/" Submitted to: Mr. Naveed Abbas \n";
52     cout << setw(30) << " "
53         << "/" \n";
54     cout << setw(30) << " "
55         << "/" submitted by \n";
56     cout << setw(30) << " "
57         << "/" \n";
58     cout << setw(30) << " "
59         << "/" Name: M Hanzala Zaheer \n";
60     cout << setw(30) << " "
61         << "/" Roll no: 22014198-060 \n";
62     cout << setw(30) << " "
63         << "/" \n";
64     cout << setw(30) << " "
65         << "/" Name: Gul e Raana \n";
66     cout << setw(30) << " "
67         << "/" Roll no: 22014198-049 \n";
68     cout << setw(30) << " "
69         << "/" \n";
70     cout << setw(30) << " "
71         << "/" Name: Iqra Aslam \n";
72     cout << setw(30) << " "
73         << "/" Roll no: 22014198-044 \n";
74     cout << setw(30) << " "
75         << "/" \n";

```

```

76     cout << setw(30) << " "
77         << " / \n";
78
79     cout << endl
80         << endl
81         << endl
82         << setw(30) << " ";
83     system("pause");
84 }
85
86 void Edit_keyword()
87 {
88     // Declare variables
89     string keyword;
90     string line;
91     string new_keyword;
92     vector<string> keywords_vec;
93     bool keyword_exists;
94     int index = 0;
95
96     // Open the file in read mode
97     ifstream file("keywords.txt");
98
99     // read each line and store it in a vector
100
101     while (getline(file, line))
102     {
103         keywords_vec.push_back(line);
104     }
105
106     // Close the file
107     file.close();
108
109     // Get the keyword to edit
110     cout << "Enter the keyword you want to edit: ";
111     getline(cin, keyword);
112     ifstream readf("keywords.txt");
113     while (getline(readf, line))
114     {

```

```

115         // Check if the keyword is equal to the line
116         if (keyword == line)
117         {
118             keyword_exists = true;
119             break;
120         }
121     }
122     readf.close();
123     if (keyword_exists == false)
124     {
125         cout << keyword << " keyword does not exist in database\n";
126         system("pause");
127         search_engine_page();
128     }
129
130     // Find the index of the keyword in the vector
131     for (int i = 0; i < keywords_vec.size(); i++)
132     {
133         if (keywords_vec[i] == keyword)
134         {
135             index = i;
136             break;
137         }
138     }
139
140     // If the keyword was found, ask the user for the new keyword
141     {
142         cout << "Enter the new keyword: ";
143         getline(cin, new_keyword);
144
145         // Replace the old keyword with the new keyword
146         keywords_vec[index] = new_keyword;
147
148         // Open the file in write mode
149         ofstream file("keywords.txt");
150
151         // Write the new keywords to the file
152         for (string keyword : keywords_vec)
153         {

```

```

154             file << keyword << endl;
155         }
156
157         // Close the file
158         file.close();
159     }
160     system("pause");
161     search_engine_page();
162 }
163
164 void check_keywords()
165 {
166     system("cls");
167     cout << "List of Keywords\n-----\n";
168     string line;
169     ifstream read("keywords.txt");
170     while (getline(read, line))
171     {
172         cout << line << endl;
173     }
174     cout << "\n\n";
175     system("pause");
176     search_engine_page();
177 }
178 void add_keywords()
179 {
180     // Declare variables
181     string keyword;
182     string line;
183     int num_keywords;
184
185     cout << "How many keywords do you want to add? ";
186     cin >> num_keywords;
187     cin.ignore();
188     // Create a file to store the keywords
189     ofstream file("keywords.txt", ios::app);
190     for (int i = 0; i < num_keywords; i++)
191     {
192         // Get input from the user

```

```

193         cout << "Enter keyword no " << i + 1 << " : ";
194         getline(cin, keyword);
195         // Write the keyword to the file in the next line
196         file << keyword << endl;
197     }
198     // Close the file
199     file.close();
200     system("pause");
201     search_engine_page();
202 }
203
204 void search()
205 {
206     // Declare variables
207     string keyword;
208     string line;
209     fstream file;
210
211     // Prompt the user for a keyword
212     cout << "Enter a keyword: ";
213     cin >> keyword;
214
215     // Open the file in read mode
216     file.open("keywords.txt", ios::in);
217
218     // Search for the keyword in the file
219     while (getline(file, line))
220     {
221         // If the line contains the keyword, print it
222         if (line.find(keyword) != string::npos)
223         {
224             cout << line << endl;
225         }
226     }
227
228     // Close the file
229     file.close();
230     system("pause");
231     search_engine_page();

```



```

232 }
233
234 void search_engine_page()
235 {
236     int op;
237     system("cls");
238     cout << right << endl
239         << endl;
240     cout << setw(30) << " "
241         << "_____ \n";
242     cout << setw(30) << " "
243         << "/" / \n";
244     cout << setw(30) << " "
245         << "/" Search Engine / \n";
246     cout << setw(30) << " "
247         << "/" / \n\n";
248     cout << setw(30) << " "
249         << "_____ \n";
250     cout << setw(30) << " "
251         << "/" / \n";
252     cout << setw(30) << " "
253         << "/" 1. Search / \n";
254     cout << setw(30) << " "
255         << "/" 2. Add keywords or Queries / \n";
256     cout << setw(30) << " "
257         << "/" 3. Keywords List / \n";
258     cout << setw(30) << " "
259         << "/" 4. Edit Keywords / \n";
260     cout << setw(30) << " "
261         << "/" 5. Log Out / \n";
262     cout << setw(30) << " "
263         << "/" / \n\n";
264
265     cout << setw(30) << " "
266         << "----> Enter your choice: ";
267     cin >> op;
268
269     cin.ignore();
270     switch (op)

```

```

271     {
272     case 1:
273         // search
274         system("cls");
275         search();
276
277         break;
278     case 2:
279         // add keywords
280         system("cls");
281         add_keywords();
282         break;
283     case 3:
284         // check keywords
285         system("cls");
286         check_keywords();
287         break;
288     case 4:
289         // edit keywords
290         system("cls");
291         Edit_keyword();
292         break;
293     case 5:
294         // Logout
295         menu();
296         break;
297     default:
298         cout << endl
299             << setw(30) << " "
300             << "Enter a valid option!\n";
301         cout << setw(30) << " ";
302         system("pause");
303         search_engine_page();
304         break;
305     }
306 }
307
308
309

```

```

310 void menu()
311 {
312     system("cls");
313     int choice;
314     cout << endl;
315     << endl;
316     cout << setw(30) << " "
317     << " _____ \n";
318     cout << setw(30) << " "
319     << "/" / \n";
320     cout << setw(30) << " "
321     << "/" Search Engine / \n";
322     cout << setw(30) << " "
323     << "/" / \n\n";
324     cout << setw(30) << " "
325     << " _____ \n";
326     cout << setw(30) << " "
327     << "/" / \n";
328     cout << setw(30) << " "
329     << "/" 1. LOGIN / \n";
330     cout << setw(30) << " "
331     << "/" 2. REGISTER / \n";
332     cout << setw(30) << " "
333     << "/" 3. FORGOT PASSWORD (or) USERNAME / \n";
334     cout << setw(30) << " "
335     << "/" 4. Exit / \n";
336     cout << setw(30) << " "
337     << "/" / \n\n";
338
339     cout << setw(30) << " "
340     << "----> Enter your choice: ";
341     cin >> choice;
342     cout << endl;
343     switch (choice)
344     {
345     case 1:
346         login();
347         break;
348     case 2:

```

```

349         registr();
350         break;
351     case 3:
352         forgot();
353         break;
354     case 4:
355
356         cout << setw(30) << " "
357             << "Thanks for using this program.\n\n\n\n";
358         break;
359     default:
360         cout << setw(30) << " "
361             << "Enter a valid option!\n\n";
362         cout << setw(30) << " ";
363         system("pause");
364         menu();
365     }
366 }
367
368
369
370 void login()
371 {
372     string user, u;
373     char pass[100], p[100];
374     system("cls");
375     cout << endl
376         << endl;
377     cout << setw(30) << " "
378         << "_____ \n";
379     cout << setw(30) << " "
380         << " / \n";
381     cout << setw(30) << " "
382         << " / \n";
383     cout << setw(30) << " "
384         << " / _____ / \n\n\n";
385     cout << right << setw(35) << " "
386         << "Enter email or username: ";
387     cin >> user;

```

```

388     cout << endl
389         << setw(35) << " "
390         << "Enter Password: ";
391     ch = '\0';
392     i = 0;
393     while ((ch = _getch()) != '\r')
394     {
395         if (ch == '\b' && i > 0)
396         {
397             cout << "\b \b";
398             i--;
399         }
400         else
401         {
402             pass[i++] = ch;
403             cout << '*';
404         }
405     }
406     cout << endl;
407     pass[i] = '\0';
408
409
410     ifstream input("database.txt");
411     while (input >> u >> p)
412     {
413         if (u == user && strcmp(p, pass) == 0)
414         {
415             {
416                 count = 1;
417                 system("cls");
418             }
419         }
420     }
421     input.close();
422
423     if (count == 1)
424     {
425         cout << endl
426             << endl;
427         cout << setw(30) << " "

```

```

427         << " _____ \n";
428     cout << setw(30) << " "
429         << "/" / \n";
430     cout << setw(30) << " "
431         << "/" DASHBOARD / \n";
432     cout << setw(30) << " "
433         << "/" / \n\n\n";
434     cout << setw(35) << " "
435         << "Hello! " << user << endl
436         << endl;
437     cout << setw(35) << " "
438         << "[LOGIN SUCCESSFUL]\n";
439     cout << setw(35) << " "
440         << "You are now logged in as " << user << endl
441         << endl
442         << endl;
443     cout << setw(35) << " ";
444     system("pause");
445     main();
446 }
447
448 else
449 {
450     cout << endl
451         << setw(35) << " "
452         << "LOGIN ERROR\n";
453     cout << setw(35) << " "
454         << "Please check your username and password\n\n\n";
455     cout << setw(35) << " ";
456     system("pause");
457     menu();
458 }
459 }
460
461 void registr()
462 {
463
464     string reguser;
465     char regpass[100], cp[100];

```

```

466 up:
467     system("cls");
468     cout << endl
469         << endl;
470     cout << setw(30) << " "
471         << "_____ \n";
472     cout << setw(30) << " "
473         << "/ \n";
474     cout << setw(30) << " "
475         << "/ Sign Up \n";
476     cout << setw(30) << " "
477         << "/_____ / \n\n\n";
478     cout << setw(35) << " "
479         << "Enter email or username: ";
480     cin >> reguser;
481     cout << endl
482         << setw(35) << " "
483         << "Enter Password: ";
484     ch = '\0';
485     i = 0;
486
487
488     while ((ch = _getch()) != '\r')
489     {
490         if (ch == '\b' && i > 0)
491         {
492             cout << "\b \b";
493             i--;
494         }
495         else
496         {
497             regpass[i++] = ch;
498             cout << '*';
499         }
500     }
501     regpass[i] = '\0';
502     cout << endl
503         << setw(35) << " "
504         << "Enter Confirm Password: ";

```

```

505 ch = '\0';
506 i = 0;
507 while ((ch = _getch()) != '\r')
508 {
509     if (ch == '\b' && i > 0)
510     {
511         cout << "\b \b";
512         i--;
513     }
514     else
515     {
516         cp[i++] = ch;
517         cout << '*';
518     }
519 }
520 cp[i] = '\0';
521 cout << endl;
522
523
524
525
526 if (strcmp(regpass, cp) == 0) // 0 mtlb true
527 {
528     ofstream reg("database.txt", ios::app);
529     reg << reguser << ' ' << regpass << endl;
530     system("cls");
531     cout << endl
532          << endl;
533     cout << setw(30) << " "
534          << "_____ \n";
535     cout << setw(30) << " "
536          << " / \n";
537     cout << setw(30) << " "
538          << " / \n";
539     cout << setw(30) << " "
540          << " / _____ \n\n";
541     cout << setw(35) << " "
542          << "Registration Sucessful\n\n\n";
543     cout << setw(35) << " ";

```



```

544         system("pause");
545         menu();
546     }
547     else
548     {
549         cout << endl
550             << setw(35) << " "
551             << "Sign Up failed! Invalid passwords, Try again\n\n\n";
552         cout << setw(35) << " ";
553         system("pause");
554         system("cls");
555         goto up;
556     }
557 }
558 void forgot()
559 {
560     system("cls");
561     cout << endl
562         << endl;
563     cout << setw(30) << " "
564         << "_____ \n";
565     cout << setw(30) << " "
566         << " / \n";
567     cout << setw(30) << " "
568         << " / \n";
569     cout << setw(30) << " "
570         << " /_____ / \n\n\n";
571     cout << setw(35) << " "
572         << "Forgotten ? We're here for help\n";
573     cout << setw(35) << " "
574         << "Mail us: 22014198-060@uog.edu.pk\n";
575     cout << setw(35) << " "
576         << "Mail us: 22014198-049@uog.edu.pk\n";
577     cout << setw(35) << " "
578         << "Mail us: 22014198-044@uog.edu.pk\n";
579     cout << setw(35) << " ";
580     system("pause");
581     menu();
582 }

```

Program Documentation

This document provides a comprehensive overview of the provided C++ source code, which implements a simple search engine application with user authentication and various functionalities. The code is structured into several functions, each serving a specific purpose within the program. Below is a detailed step-by-step explanation of each function and its role within the program.

1. **project_page Function:**

This function displays project information in a formatted manner. It showcases the names, roll numbers, and project details of the group members.

2. **Edit_keyword Function:**

This function allows users to edit keywords stored in the "keywords.txt" file.

It reads the keywords from the file and stores them in a vector.

The user is prompted to enter the keyword they wish to edit.

If the entered keyword exists in the vector, the user is prompted to provide a new keyword.

The old keyword is replaced with the new keyword in the vector.

The vector's contents are then written back to the "keywords.txt" file.

3. check_keywords Function:

This function reads and displays the list of keywords stored in the "keywords.txt" file. Each keyword is printed on a separate line for easy readability.

4. add_keywords Function:

Users can input the number of keywords they want to add.

For each keyword, the user is prompted to input a keyword, and each keyword is written to the "keywords.txt" file on a separate line.

5. search Function:

Users are prompted to enter a keyword they want to search for.

The function opens the "keywords.txt" file in read mode.

It then searches for the entered keyword within the file, line by line.

If a line containing the keyword is found, it is printed on the screen.

6. search_engine_page Function:

This function serves as the main interface for the search engine application.

It presents users with a menu of options, including search, add keywords, view keywords, edit keywords, and log out.

Users input their choice, and the corresponding functionality is executed based on their selection.

The menu is looped until the user chooses to log out.

7. menu Function:

The main menu function displays the initial options for the program.

Users can choose to log in, register, recover a forgotten password, or exit the program.

The selected option directs users to the corresponding function's implementation.

8. login Function:

Users are prompted to input their username and password.

The function reads the username and password combinations from the "database.txt" file.

If a matching username and password are found, the user is logged in and directed to the search_engine_page function.

9. registr Function:

Users can register by providing a username and password.

The function prompts users to enter their desired username and password.

The password is confirmed by asking the user to re-enter it.

If the passwords match, the username and password are written to the "database.txt" file for future authentication.

10. forgot Function:

This function provides contact information for users who need help with forgotten passwords.

Users are directed to contact the specified email addresses for assistance.

11. main Function:

The program's entry point that initializes the application.

It checks the value of the count variable to determine whether a user is logged in or not.

If a user is logged in (count is 1), the search_engine_page function is called to provide access to search functionalities.

If no user is logged in (count is 0), the project_page function is displayed to showcase the project details, and then the main menu is shown using the menu function.

Overall Summary:

The provided C++ program implements a basic search engine application with user authentication, keyword management, and project information display. Users can log in, register, search for keywords, add keywords, view a list of keywords, edit keywords, and log out. The program showcases the use of functions to modularize different functionalities, and each function contributes to creating a user-friendly search engine experience.

Step-by-Step explanation of how the program works in sequence when executed

Main Function (main):

The program starts execution from the main function.

The count variable is checked. If count is 1, it means a user is logged in, so the search_engine_page function is called.

If count is 0, it means no user is logged in, so the project_page function is displayed followed by the menu function.

project_page Function:

This function displays formatted project details and group member information.

It presents a visually appealing representation of the project and contributors' names and roll numbers.

Users can press any key to continue, after which the menu function is called.

menu Function:

The main menu function is displayed, offering options like login, registration, forgotten password recovery, and program exit.

Users input their choice based on the provided options.

Depending on the choice, the relevant function is called:

If "1" is selected, the login function is called.

If "2" is selected, the registr function is called.

If "3" is selected, the forgot function is called.

If "4" is selected, a farewell message is displayed, and the program terminates.

login Function:

Users enter their username and password.

The function reads the "database.txt" file line by line, matching entered username and password combinations.

If a match is found, the count variable is set to 1, and the user is redirected to the search_engine_page function.

If no match is found, an error message is displayed, and users are redirected back to the menu function.

registr Function:

Users enter their desired username and password.

The password is confirmed by entering it again.

If the passwords match, the username and password are written to the "database.txt" file for future authentication.

If the passwords do not match, an error message is displayed, and the user is prompted to retry.

forgot Function:

Users are provided with contact information for assistance with forgotten passwords.
Email addresses are displayed for users to reach out for help.

search_engine_page Function:

Users who are logged in (count is 1) are directed to this function.

The function displays a menu with various options like search, adding keywords, viewing keywords, editing keywords, and logging out.

Users input their choice, and the corresponding functionality is executed.

After each action, users are brought back to the search_engine_page menu.

search Function:

Users input a keyword they want to search for.

The function searches for the keyword within the "keywords.txt" file.

If a match is found, the corresponding line (keyword) is printed on the screen.

add_keywords Function:

Users specify the number of keywords they want to add.

For each keyword, users input the keyword itself.

Keywords are appended to the "keywords.txt" file.

check_keywords Function:

This function reads the "keywords.txt" file and prints out all stored keywords line by line.

Edit_keyword Function:

Users input a keyword they wish to edit.

The function checks if the keyword exists.

If the keyword exists, users input a new keyword, which replaces the old keyword in the "keywords.txt" file.

Logout and Continuation:

Upon logging out (count is set to 0), users are redirected back to the menu function.

Users can continue using the program by logging in again or performing other actions through the main menu.

Overall, the program provides an interactive and organized environment for users to interact with a simulated search engine, manage keywords, and navigate through various functionalities. It achieves this through a series of modularized functions, making the user experience seamless and intuitive.

Different Interfaces of the Program

```
Project:      SEARCH ENGINE
Submitted to: Mr. Naveed Abbas
              submitted by
Name:         M Hanzala Zaheer
Roll no:      22014198-060
Name:         Gul e Raana
Roll no:      22014198-049
Name:         Iqra Aslam
Roll no:      22014198-044
```

Press any key to continue . . . _

Search Engine

1. LOGIN
2. REGISTER
3. FORGOT PASSWORD (or) USERNAME
4. Exit

----> Enter your choice: █

Search Engine

1. Search
2. Add keywords or Queries
3. Keywords List
4. Edit Keywords
5. Log Out

----> Enter your choice: