

Proiect Programare si limbaj de programare 2 C++/C

“Sistem de gestiune a fisierelor”:

Un program care sa creeze,sa citeasca,sa modifice
si sa stearga un fisier

Studenti:

Boza Silviu Gabriel

Profesori coordonatori:

Olteanu Gabriela

Neculoiu Georgian

Cuprins

1. Introducere.....	1
2. Analiza detaliata a codul.....	3
3. Structura principala.....	3
4. Functii implementate: Creerea de fisier.....	4
5. Functii implementate: Citirea fisier.....	5
6. Functii implementate: Modificarea fisierului.....	6
7. Functii implementate: Stergerea Fisierului.....	7
8. Utilizarea codului in alte aplicatii.....	9
9. Analiza rezultatelor.....	10
10. Posibile imbunatatiri.....	10
11. Codul integral.....	11
12. Concluzie.....	13
13. Bibliografie.....	14

I. Introducere

Proiectul „Sistem de gestiune a fișierelor” are ca scop dezvoltarea unei aplicații în limbajul C++ care permite utilizatorului să gestioneze fișiere text în mod interactiv, prin intermediul unei interfețe simple în linia de comandă. Gestiunea fișierelor este o componentă esențială în orice sistem de operare și aplicație. informatică Funcționalitățile principale sunt:

- Proiectul „Sistem de gestiune a fișierelor” are ca scop dezvoltarea unei aplicații în limbajul **Crearea** unui fișier nou
- **Citirea** conținutului unui fișier
- **Modificarea** (suprascierea) fișierului
- **Ștergerea** fișierului

Acest tip de aplicație este util pentru învățarea lucrului cu fluxuri de fișiere (fstream) și interacțiunea de bază cu sistemul de operare.

II. Analiza detaliata a codului

2.1. Structura peincipala”:

```
Int main() {
// Afiseaza meniul si citeste optiunea utilizatorului
```

Se foloseste un meniu interactive pentru a permite utilizatorului sa aleaga actiuni.

Optiunile sunt procesate printr-un switch,in functie de input-ul utilizatorului.

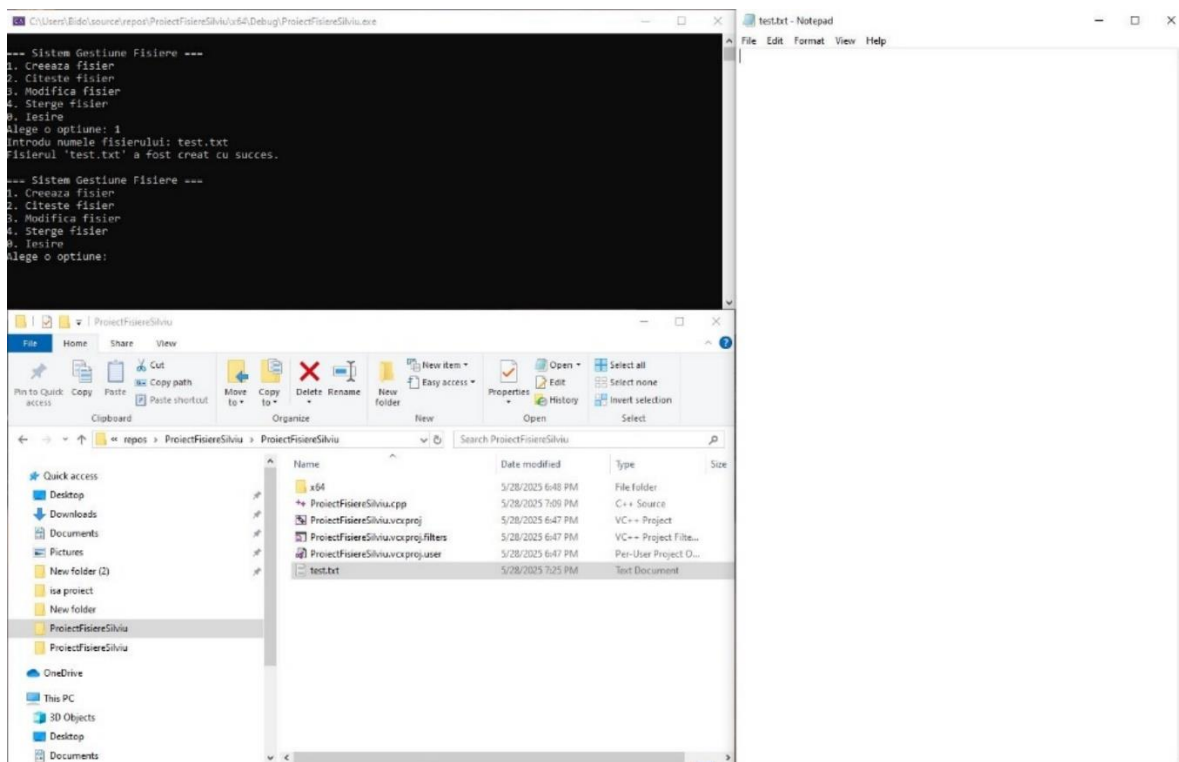
.

2.2. Functii implementate: Creerea de fisier

creeazaFisier()

ofstream fisier(test.txt);

Creeaza un fisier gold aca nu exista deja
Daca exista,il suprascrie (implicit in ofstream)

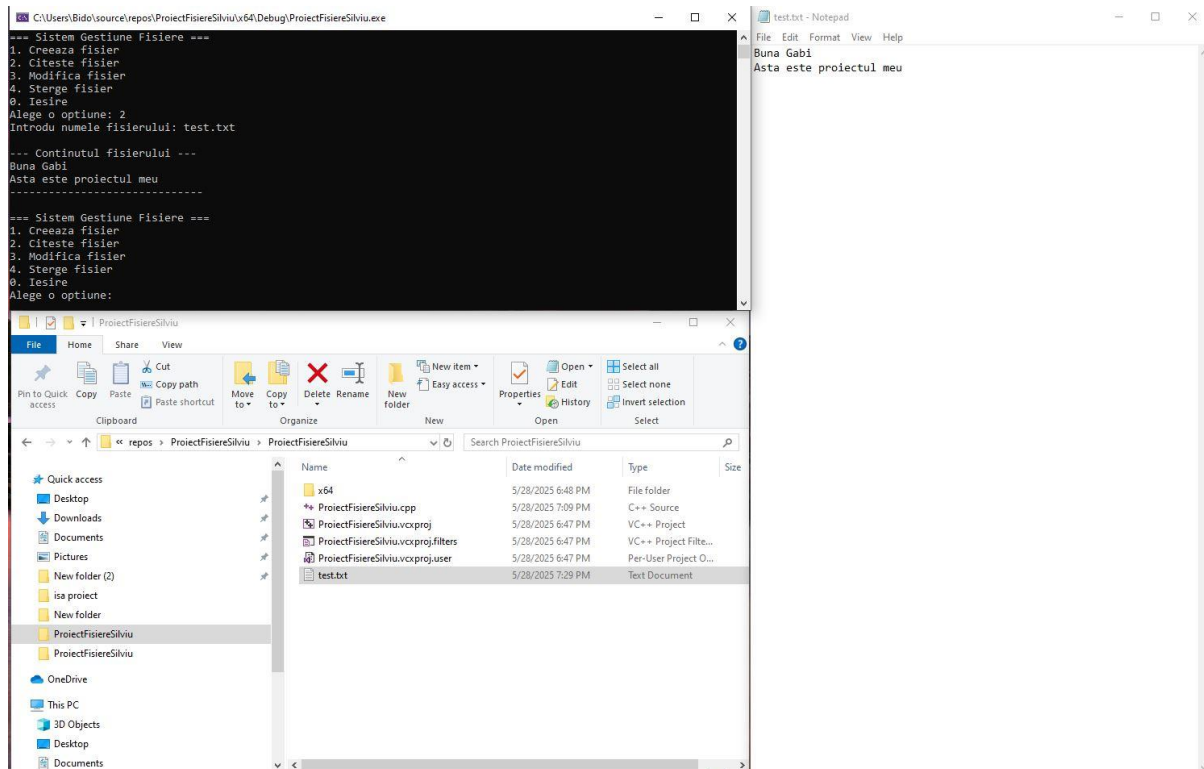


2.3. Functii implementate: Citirea fisierului

`citesteFisier()`

```
ifstream fisier(test.txt);
getline(fisier,linie);
```

Deschide fisierul in mod “read-only”.
Afiseaza continutul linie cu linie.

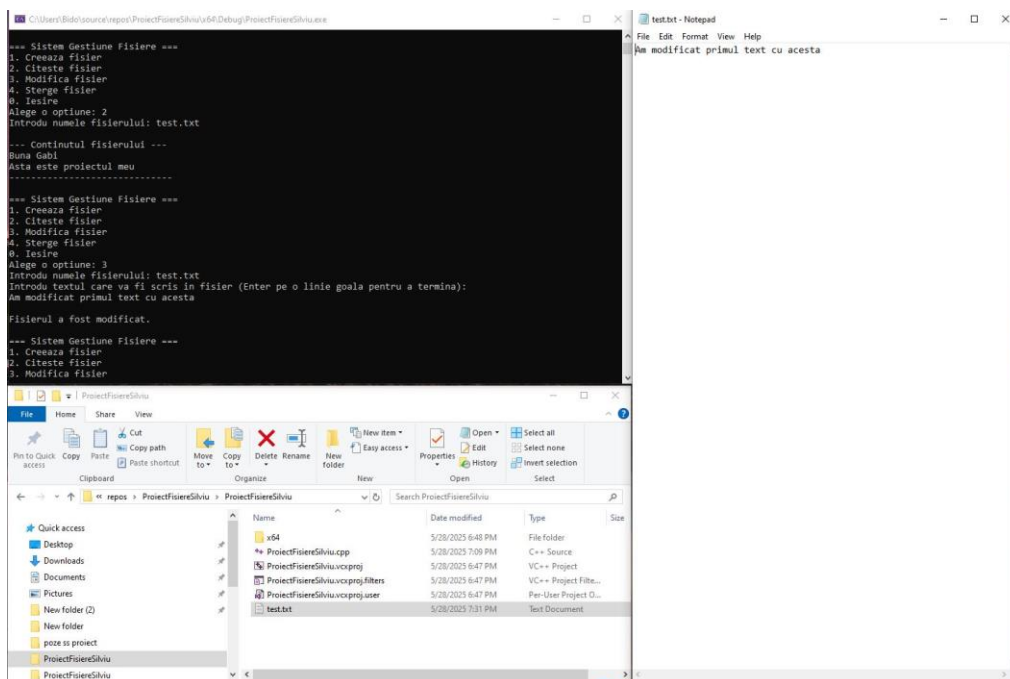


2.4. Functii implementate: Modificarea fisierului

modificaFisier()

```
ofstream fisier(text.txt);
fisier <<linie <<endl;
```

Deschide fisierul in mod “write”,suprascriind continutul
Permite introducerea de text pana la o linie goala

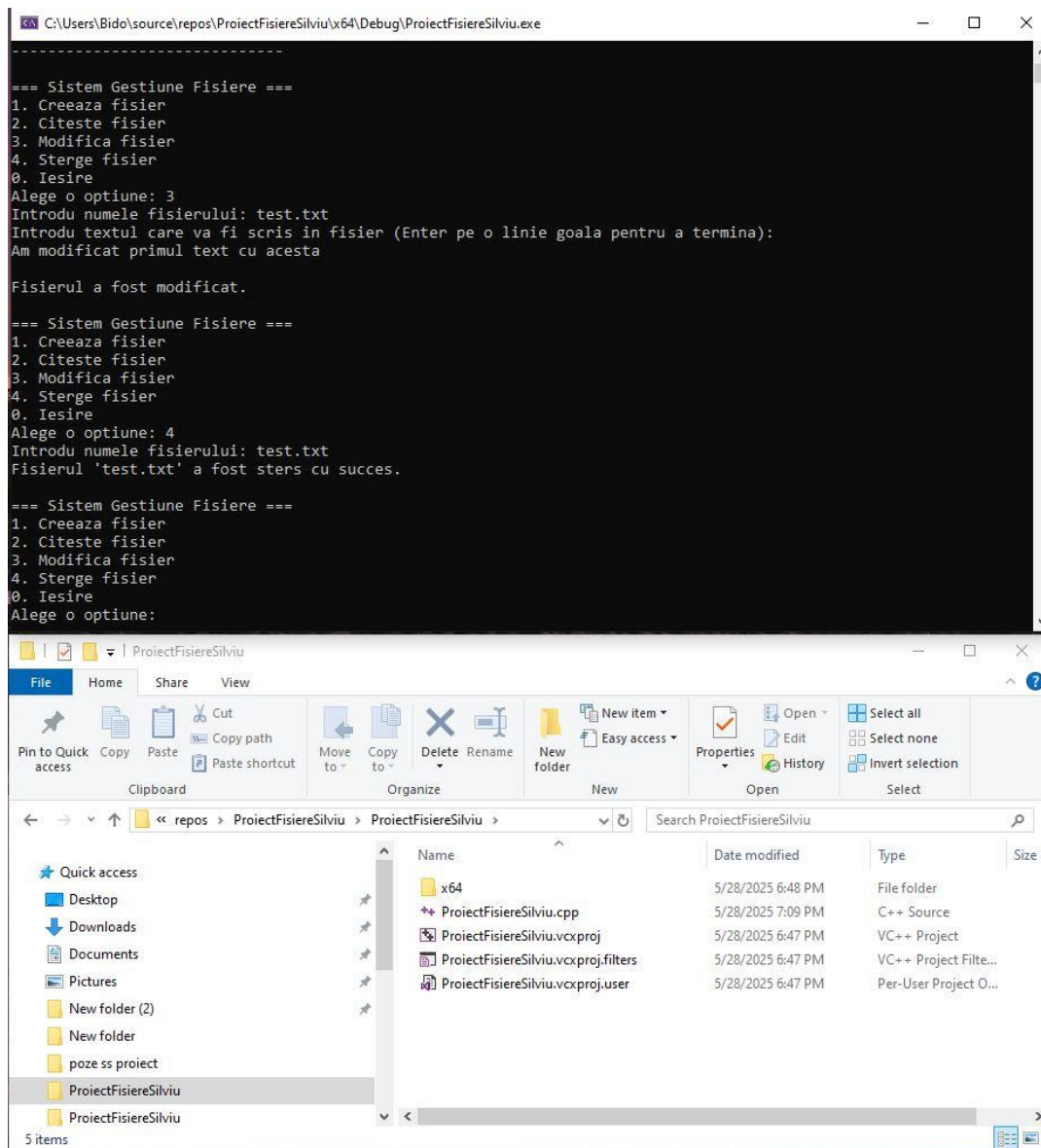


2.5. Functii implementate: Stergerea Fisierului

`stergeFisier()`

`remove(test.txt.c_str());`

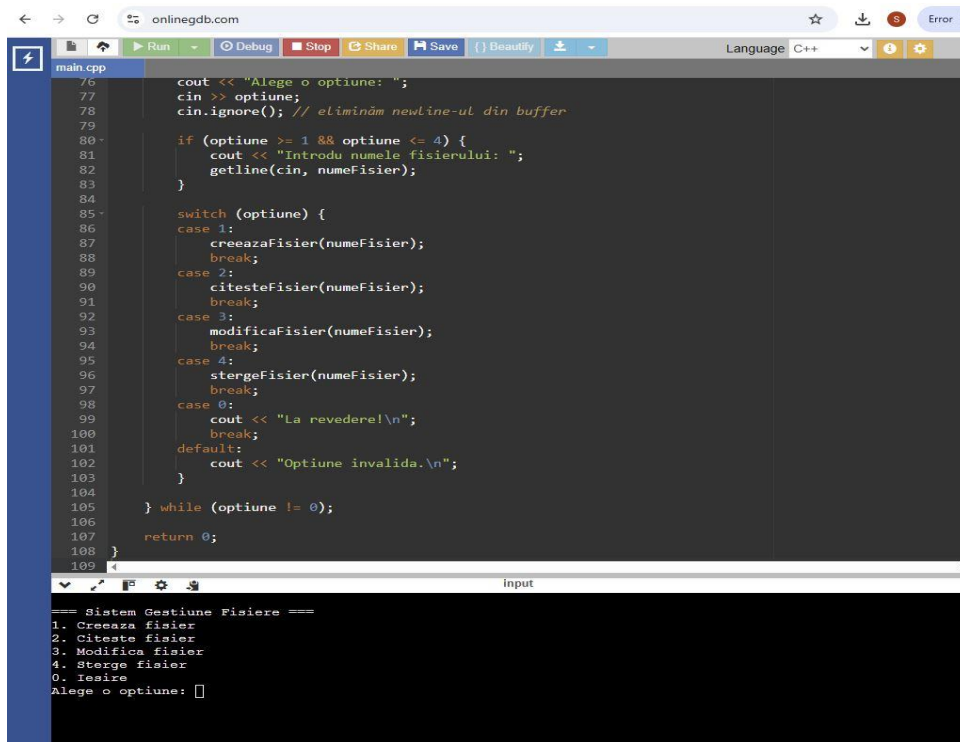
Foloseste functia C `remove()` pentru a sterge un fisier
Returneaza 0 la success, astfel afiseaza eroare



- **Fluxuri de fișiere:** C++ utilizează obiectele `ifstream`, `ofstream` și `fstream` pentru citire, scriere și ambele simultan.
- **Tratamentul erorilor:** Este implementat minim (ex: verificarea dacă fișierul a fost deschis cu succes). Codul poate fi extins cu tratamente specifice de excepții (`try-catch`).
- **Funcționalitate interactivă:** Programul rulează într-o buclă `do-while`, ceea ce permite utilizatorului să efectueze mai multe acțiuni într-o singură sesiune.
- **Compatibilitate multiplatformă:** Codul este portabil, fiind scris folosind doar biblioteci standard.

2.6. Utilizarea codului si in alte aplicatii:

Codul programului poate sa ruleze si in alte aplicatii,nu doar in VS 2022,cum ar fi “GDB Online Debugger”



```
main.cpp
76 cout << "Alege o optiune: ";
77 cin >> optiune;
78 cin.ignore(); // eliminăm newline-ul din buffer
79
80 if (optiune >= 1 && optiune <= 4) {
81     cout << "Introdu numele fisierului: ";
82     getline(cin, numeFisier);
83 }
84
85 switch (optiune) {
86     case 1:
87         creeazaFisier(numeFisier);
88         break;
89     case 2:
90         citesteFisier(numeFisier);
91         break;
92     case 3:
93         modificaFisier(numeFisier);
94         break;
95     case 4:
96         stergeFisier(numeFisier);
97         break;
98     case 0:
99         cout << "La revedere!\n";
100         break;
101     default:
102         cout << "Optiune invalida.\n";
103 }
104
105 } while (optiune != 0);
106
107 return 0;
108 }
109 }
```

```
==== Sistem Gestiune Fisiere ====
1. Creeaza fisier
2. Citeste fisier
3. Modifica fisier
4. Sterge fisier
0. Iesire
Alege o optiune: 
```

3. Analiza rezultatelor

Aplicația a fost testată în Visual Studio 2022 pe Windows 10. Funcțiile de creare, citire, modificare și ștergere au fost validate cu mai multe tipuri de fișiere și conținut.

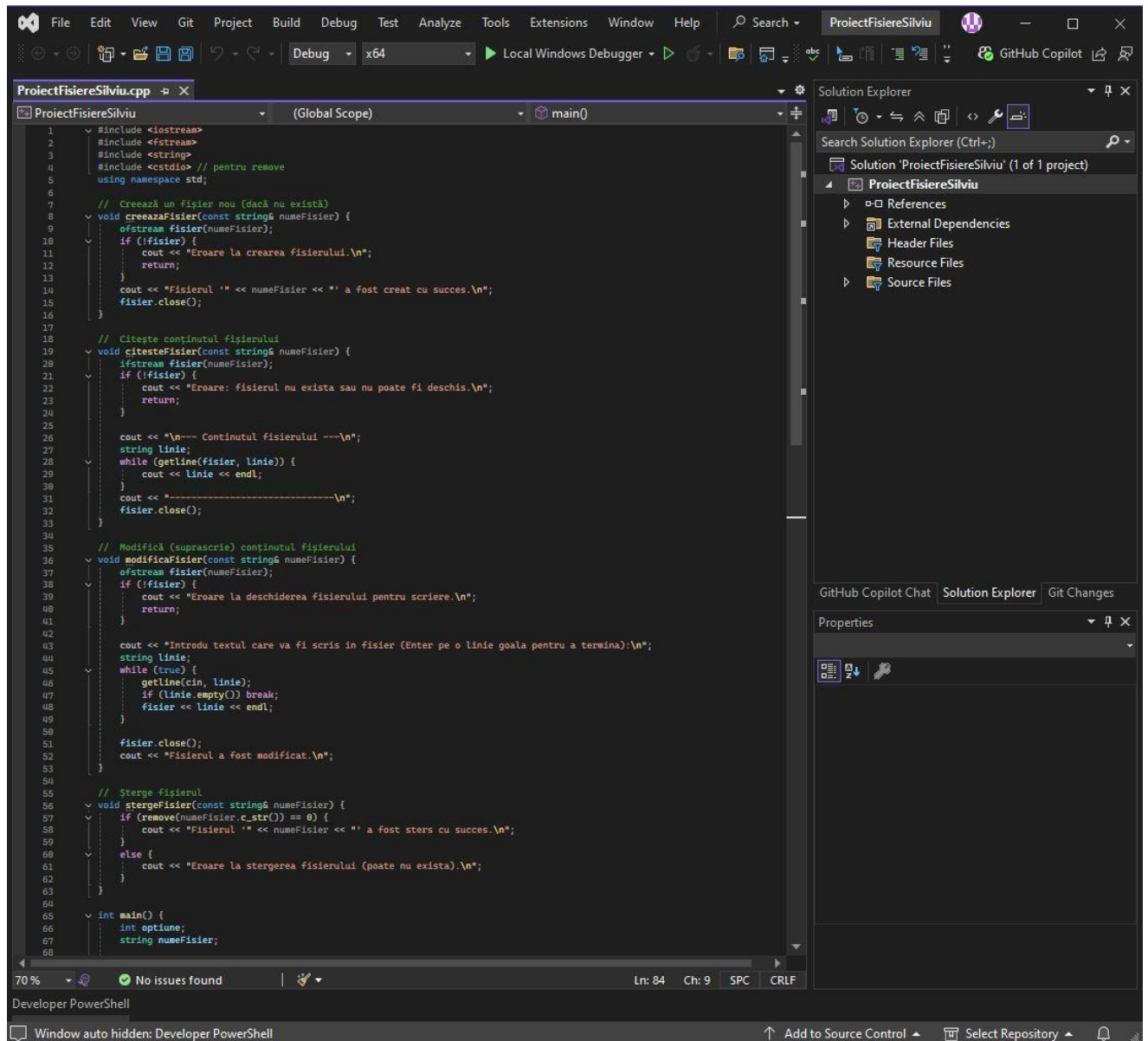
Observații:

- Utilizatorul poate crea și edita oricâte fișiere dorește, atâta timp cât are permisiuni de scriere.
- Nu apar erori critice dacă fișierul nu există, dar afișarea de mesaje informative poate fi îmbunătățită.
- Utilizatorul trebuie să introducă numele fișierului manual — nu există completare automată sau validare avansată.

4. Posibile imbunatatiri:

Adăugare în fișier	Deschiderea fișierului în mod <code>ios : : app</code> pentru a păstra conținutul existent
Listarea fișierelor	Integrarea unei funcții care afișează toate fișierele <code>.txt</code> din director
Jurnalizare/logging	Salvarea fiecărei acțiuni într-un fișier „log.txt”
Validare input	Verificarea ca numele fișierului să nu conțină caractere nepermise (<code>?</code> , <code>*</code> , <code>:</code> etc.)
Interfață grafică	Folosirea bibliotecii Qt pentru a realiza o versiune cu butoane, meniuri și ferestre
Criptare simplă	Aplicarea unui algoritm XOR sau Caesar pentru criptarea textului din fișier

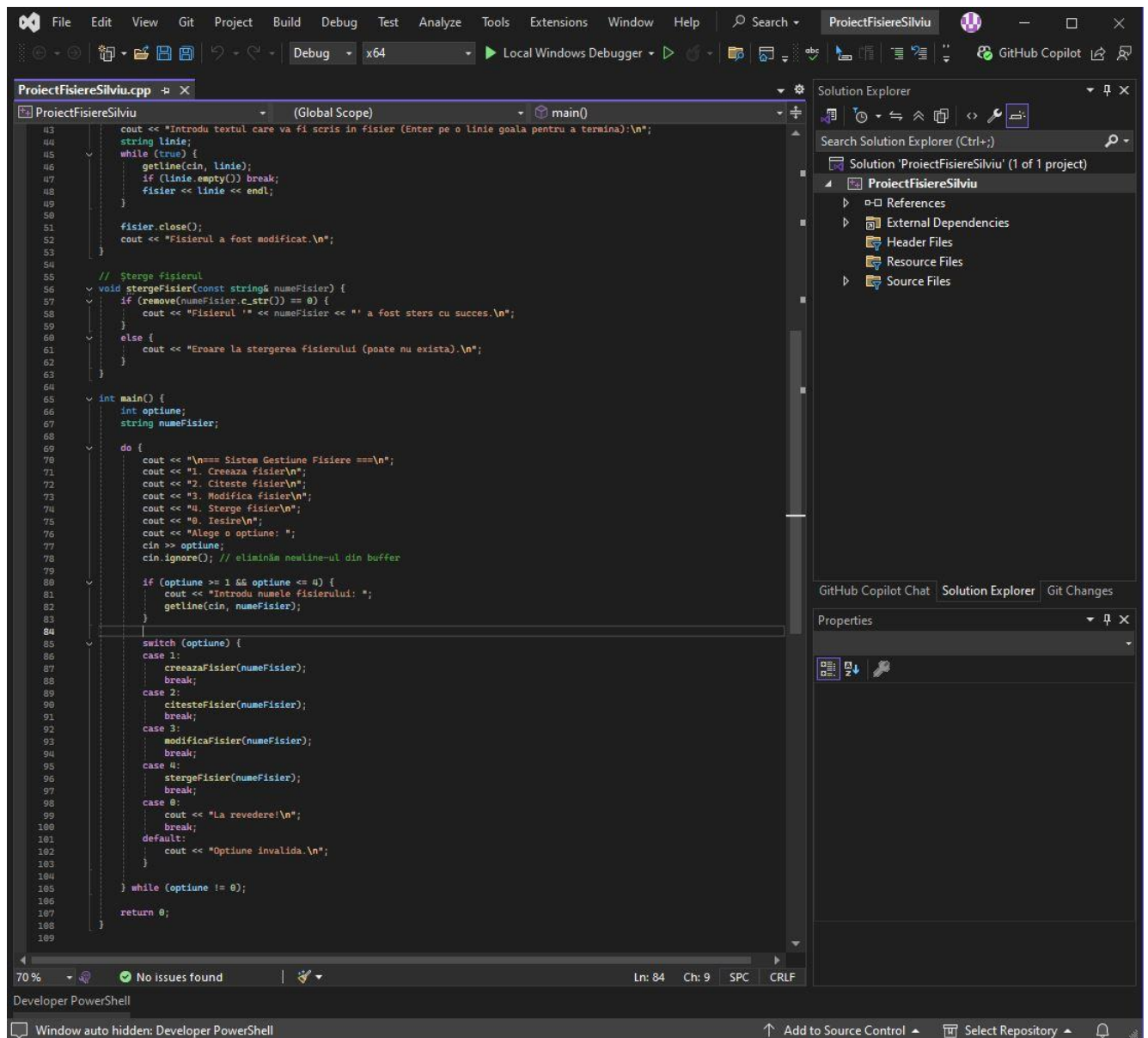
5. Codul integral:



```

1  #include <iostream>
2  #include <fstream>
3  #include <string>
4  #include <cstdio> // pentru remove
5  using namespace std;
6
7  // Creează un fisier nou (dacă nu există)
8  void creeazaFisier(const string& numeFisier) {
9      ofstream fisier(numeFisier);
10     if (!fisier) {
11         cout << "Eroare la crearea fisierului.\n";
12         return;
13     }
14     cout << "Fisierul '" << numeFisier << "' a fost creat cu succes.\n";
15     fisier.close();
16 }
17
18 // Citește conținutul fisierului
19 void citesteFisier(const string& numeFisier) {
20     ifstream fisier(numeFisier);
21     if (!fisier) {
22         cout << "Eroare: fisierul nu exista sau nu poate fi deschis.\n";
23         return;
24     }
25     cout << "\n--- Conținutul fisierului ---\n";
26     string linie;
27     while (getline(fisier, linie)) {
28         cout << linie << endl;
29     }
30     cout << "-----\n";
31     fisier.close();
32 }
33
34 // Modifică (suprascrie) conținutul fisierului
35 void modificaFisier(const string& numeFisier) {
36     ofstream fisier(numeFisier);
37     if (!fisier) {
38         cout << "Eroare la deschiderea fisierului pentru scriere.\n";
39         return;
40     }
41
42     cout << "Introdu textul care va fi scris in fisier (Enter pe o linie goala pentru a termina):\n";
43     string linie;
44     while (true) {
45         getline(cin, linie);
46         if (linie.empty()) break;
47         fisier << linie << endl;
48     }
49
50     fisier.close();
51     cout << "Fisierul a fost modificat.\n";
52 }
53
54 // Șterge fisierul
55 void stergeFisier(const string& numeFisier) {
56     if (remove(numeFisier.c_str()) == 0) {
57         cout << "Fisierul '" << numeFisier << "' a fost sters cu succes.\n";
58     }
59     else {
60         cout << "Eroare la stergerea fisierului (poate nu exista).\n";
61     }
62 }
63
64 int main() {
65     int optiune;
66     string numeFisier;
67
68

```



```

43     cout << "Introdu textul care va fi scris in fisier (Enter pe o linie goala pentru a termina):\n";
44     string linie;
45     while (true) {
46         getline(cin, linie);
47         if (linie.empty()) break;
48         fisier << linie << endl;
49     }
50
51     fisier.close();
52     cout << "Fisierul a fost modificat.\n";
53 }
54
55 // Sterge fisierul
56 void stergeFisier(const string& numeFisier) {
57     if (remove(numeFisier.c_str()) == 0) {
58         cout << "Fisierul '" << numeFisier << "' a fost sters cu succes.\n";
59     }
60     else {
61         cout << "Eroare la stergerea fisierului (poate nu exista).\n";
62     }
63 }
64
65 int main() {
66     int optiune;
67     string numeFisier;
68
69     do {
70         cout << "\n=== Sistem Gestiune Fisiere ===\n";
71         cout << "1. Creeaza fisier\n";
72         cout << "2. Citeste fisier\n";
73         cout << "3. Modifica fisier\n";
74         cout << "4. Sterge fisier\n";
75         cout << "0. Iesire\n";
76         cout << "Alege o optiune: ";
77         cin >> optiune;
78         cin.ignore(); // elimina newline-ul din buffer
79
80         if (optiune >= 1 && optiune <= 4) {
81             cout << "Introdu numele fisierului: ";
82             getline(cin, numeFisier);
83         }
84
85         switch (optiune) {
86             case 1:
87                 creeazaFisier(numeFisier);
88                 break;
89             case 2:
90                 citesteFisier(numeFisier);
91                 break;
92             case 3:
93                 modificaFisier(numeFisier);
94                 break;
95             case 4:
96                 stergeFisier(numeFisier);
97                 break;
98             case 0:
99                 cout << "La revedere!\n";
100                break;
101            default:
102                cout << "Optiune invalida.\n";
103        }
104    } while (optiune != 0);
105
106    return 0;
107 }
108
109

```

Dupa ce codul a fost scris cu succes se foloseste comanda Ctrl+ F5 sau se apasa pe Local Windows Debugger pentru a rula codul si a deschide aplicatia de unde putem crea,modifica,sterge si citii un fisier aratand in felul urmator:

```
C:\Users\Bido\Desktop\ProiectFisiere\App VS2022-Proiect\x64\Debug\ProiectFisiereSilviu.exe

=== Sistem Gestione Fisiere ===
1. Creeaza fisier
2. Citeste fisier
3. Modifica fisier
4. Sterge fisier
0. Iesire
Alege o optiune:
```

6. Concluzie

Proiectul „Sistem de gestiune a fișierelor” oferă o bază solidă pentru înțelegerea lucrului cu fișiere în C++. Codul este clar, modular și poate fi extins cu ușurință. Fiind scris într-un limbaj compilat precum C++, asigură eficiență și control total asupra operațiilor de I/O.

Această aplicație poate servi ca punct de plecare pentru proiecte mai complexe, cum ar fi:

- un editor de fișiere simplificat
- un program de backup automatizat
- un sistem de organizare a fișierelor personale

Prin extinderea funcționalităților, acest proiect poate evolua într-o aplicație reală, cu impact educațional sau chiar profesional.

Bibliografie

cplusplus.com – File streams

Tutorial oficial despre lucrul cu fișiere în C++, explică utilizarea ifstream, ofstream și fstream.

GeeksforGeeks – File Handling in C++

Explicații clare și moderne despre clasele pentru fișiere și modul de citire/scriere în C++.

Microsoft Learn – Working with files in C++

Documentația oficială Microsoft pentru lucrul cu fișiere folosind biblioteca standard C++ în Visual Studio.

Programiz – C++ File Handling

Un ghid didactic cu exemple simple, orientat spre începători.

Stack Overflow – C++ file input/output questions

Forum activ cu întrebări și soluții legate de gestiunea fișierelor în C++. Util pentru rezolvarea erorilor.

TutorialsPoint – C++ Files and Streams

Prezentare structurată a operațiilor cu fișiere în C++.