Console File Explorer Application in C++

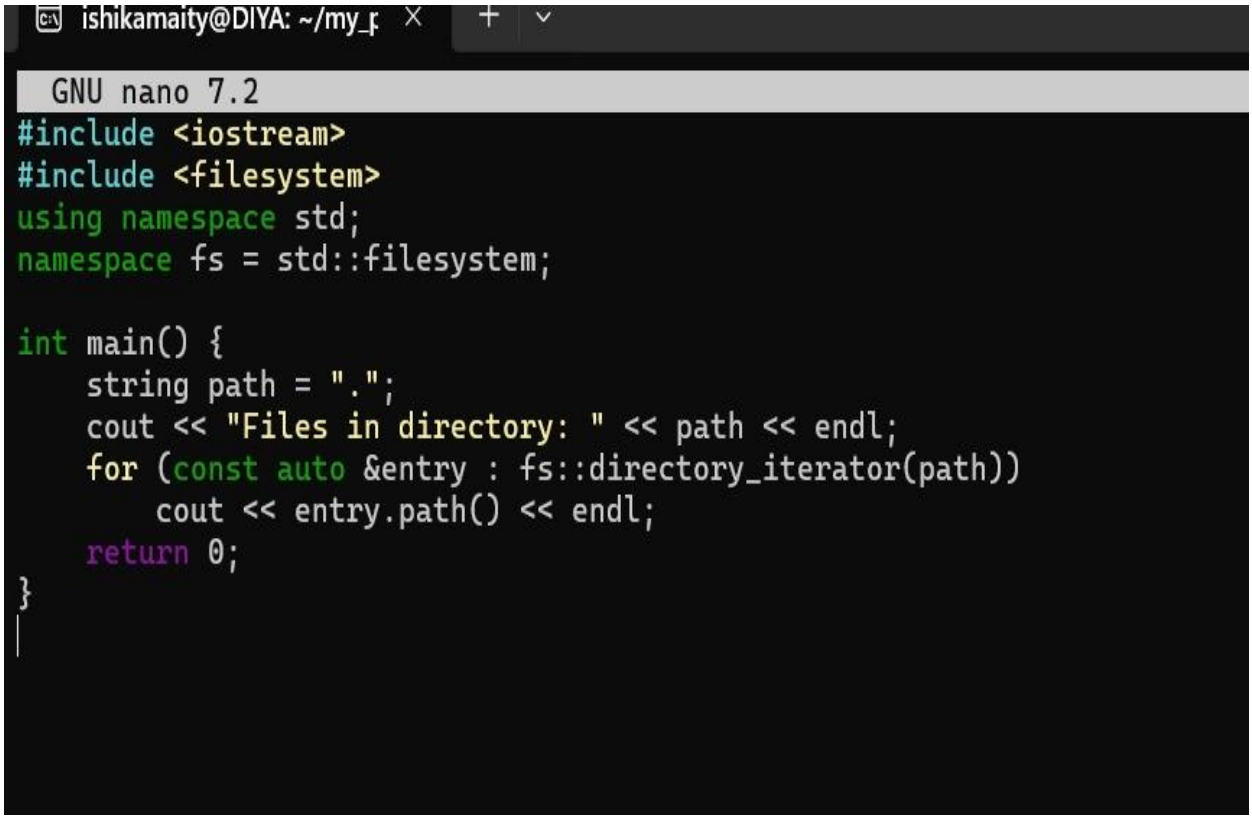
1. Title

Console File Explorer Application in C++

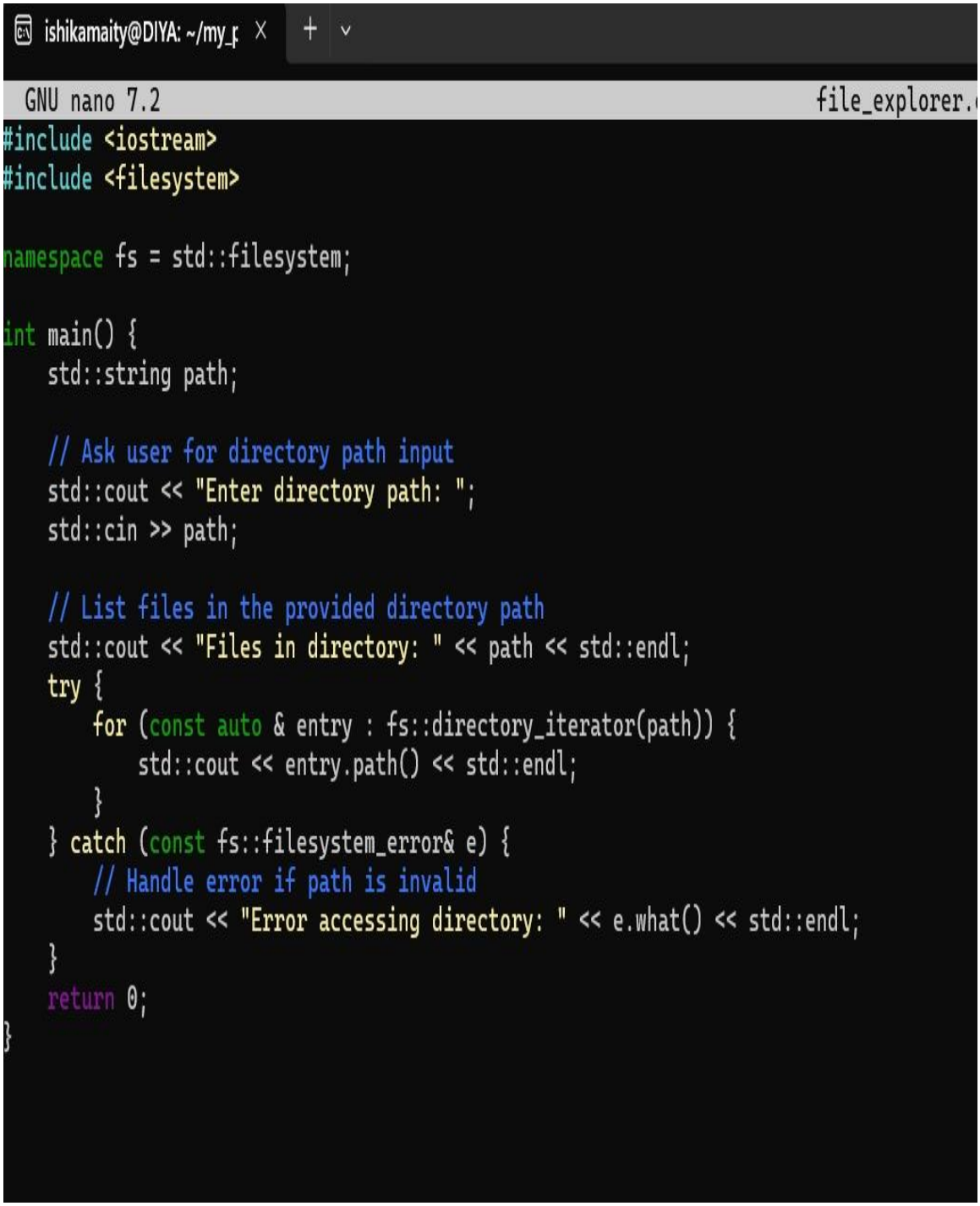
2. Codes

The project is developed stepwise with the following features:

-Directory listing to show the files and folders in a specified path.



-User input for navigation into different directories.



-File operations including create and delete files.

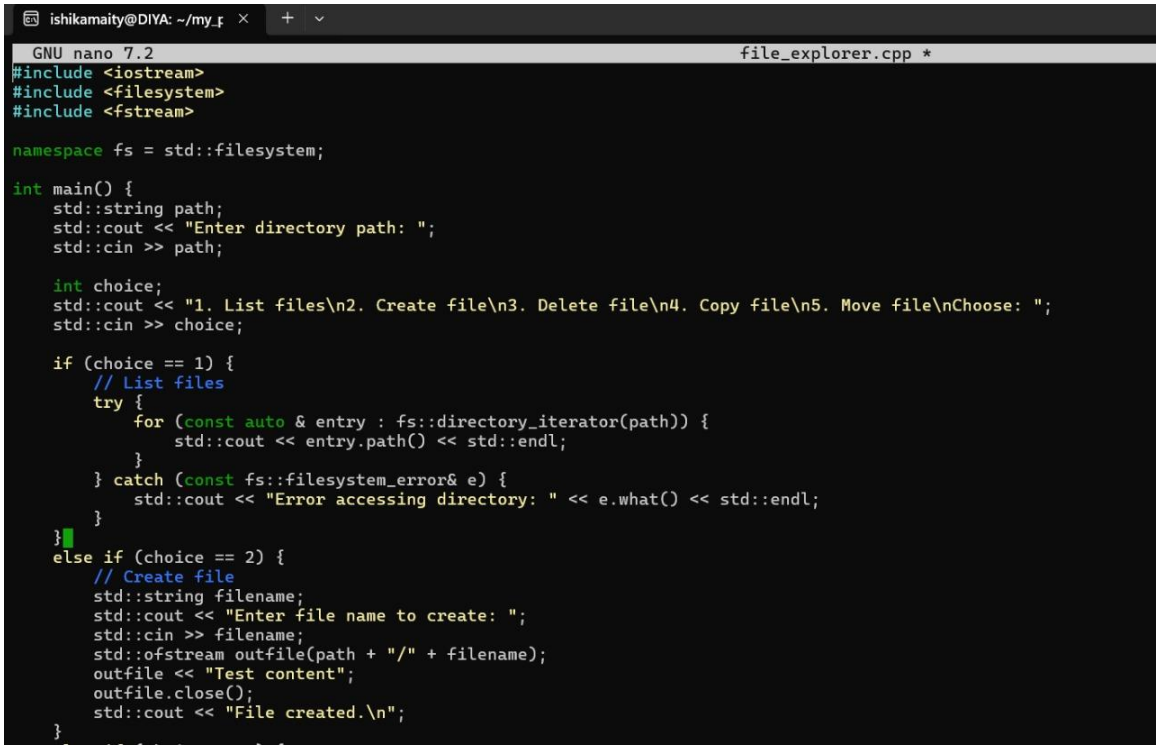
ishikamaity@DIYA: ~/my_p

GNU nano 7.2 file_explorer.cpp

```cpp
#include <fstream>  // Add this at the top of your file (if not already included)
#include <iostream>
#include <filesystem>

namespace fs = std::filesystem;

int main() {
    std::string path;
    std::cout << "Enter directory path: ";
    std::cin >> path;

    int choice;
    std::cout << "1. List files\n2. Create file\n3. Delete file\nChoose: ";
    std::cin >> choice;

    if (choice == 1) {
        // List files
        try {
            for (const auto & entry : fs::directory_iterator(path)) {
                std::cout << entry.path() << std::endl;
            }
        } catch (const fs::filesystem_error& e) {
            std::cout << "Error accessing directory: " << e.what() << std::endl;
        }
    }
    else if (choice == 2) {
        // Create file
        std::string filename;
        std::cout << "Enter file name to create: ";
        std::cin >> filename;
        std::ofstream outfile(path + "/" + filename);
        outfile << "Test content";  // You can change file content here
        outfile.close();
        std::cout << "File created.\n";
    }
```

```cpp
    else if (choice == 3) {
        // Delete file
        std::string filename;
        std::cout << "Enter file name to delete: ";
        std::cin >> filename;
        if (fs::remove(path + "/" + filename)) {
            std::cout << "File deleted.\n";
        } else {
            std::cout << "File not found or could not delete.\n";
        }
    }
    else {
        std::cout << "Invalid choice.\n";
    }
    return 0;
}
```

- File manipulation with copy and move (rename) capabilities.



```
ishikamaity@DIYA: ~/my_f
 GNU nano 7.2                                                file_explorer.cpp *
#include <iostream>
#include <filesystem>
#include <fstream>

namespace fs = std::filesystem;

int main() {
    std::string path;
    std::cout << "Enter directory path: ";
    std::cin >> path;

    int choice;
    std::cout << "1. List files\n2. Create file\n3. Delete file\n4. Copy file\n5. Move file\nChoose: ";
    std::cin >> choice;

    if (choice == 1) {
        // List files
        try {
            for (const auto & entry : fs::directory_iterator(path)) {
                std::cout << entry.path() << std::endl;
            }
        } catch (const fs::filesystem_error& e) {
            std::cout << "Error accessing directory: " << e.what() << std::endl;
        }
    }
    else if (choice == 2) {
        // Create file
        std::string filename;
        std::cout << "Enter file name to create: ";
        std::cin >> filename;
        std::ofstream outfile(path + "/" + filename);
        outfile << "Test content";
        outfile.close();
        std::cout << "File created.\n";
    }
```

```
    else if (choice == 3) {
        // Delete file
        std::string filename;
        std::cout << "Enter file name to delete: ";
        std::cin >> filename;
        if (fs::remove(path + "/" + filename)) {
            std::cout << "File deleted.\n";
        } else {
            std::cout << "File not found or could not delete.\n";
        }
    }
    else if (choice == 4) {
        // Copy file
        std::string src, dest;
        std::cout << "Enter source file name: ";
        std::cin >> src;
        std::cout << "Enter destination file name: ";
        std::cin >> dest;
        try {
            fs::copy(path + "/" + src, path + "/" + dest, fs::copy_options::overwrite_existing);
            std::cout << "File copied.\n";
        } catch (fs::filesystem_error &e) {
             std::cout << "Error copying file: " << e.what() << std::endl;
        }
    }
```

```
    else if (choice == 5) {
        // Move/Rename file
        std::string src, dest;
        std::cout << "Enter source file name: ";
        std::cin >> src;
        std::cout << "Enter new file name or destination: ";
        std::cin >> dest;
        try {
            fs::rename(path + "/" + src, path + "/" + dest);
            std::cout << "File moved/renamed.\n";
        } catch (fs::filesystem_error &e) {
            std::cout << "Error moving file: " << e.what() << std::endl;
        }
    }
    else {
        std::cout << "Invalid choice.\n";
    }
    return 0;
}
```

- File search to find files by name in the directory.

```
 GNU nano 7.2                                  file_explorer.cpp
#include <iostream>
#include <fstream>
#include <filesystem>

namespace fs = std::filesystem;

int main() {
    std::string path;
    std::cout << "Enter directory path: ";
    std::cin >> path;

    while (true) {
        int choice;
        std::cout << "1. List files\n2. Create file\n3. Delete file\n4. Copy file\n5. Move file\n6. Search file\n7. Exit\nChoose: ";
        std::cin >> choice;

        if (choice == 1) {
            // List files
            try {
                for (const auto & entry : fs::directory_iterator(path)) {
                    std::cout << entry.path() << std::endl;
                }
            } catch (const fs::filesystem_error& e) {
                std::cout << "Error accessing directory: " << e.what() << std::endl;
            }
        }
        else if (choice == 2) {
            // Create file code
            std::string filename;
            std::cout << "Enter file name to create: ";
            std::cin >> filename;
            std::ofstream outfile(path + "/" + filename);
            outfile << "Test content";
            outfile.close();
            std::cout << "File created.\n";
        }
```

ishikamaity@DIYA: ~/my_f

GNU nano 7.2 file_explorer.cpp

```
    else if (choice == 3) {
        // Delete file code
        std::string filename;
        std::cout << "Enter file name to delete: ";
        std::cin >> filename;
        if (fs::remove(path + "/" + filename))
            std::cout << "File deleted.\n";
        else
            std::cout << "File not found or could not delete.\n";
    }
    else if (choice == 4) {
        // Copy file code
        std::string src, dest;
        std::cout << "Enter source file name: ";
        std::cin >> src;
        std::cout << "Enter destination file name: ";
        std::cin >> dest;
        try {
            fs::copy(path + "/" + src, path + "/" + dest, fs::copy_options::overwrite_existing);
            std::cout << "File copied.\n";
        } catch (fs::filesystem_error &e) {
            std::cout << "Error copying file: " << e.what() << std::endl;
        }
    }
```

ishikamaity@DIYA: ~/my_f

GNU nano 7.2 file_explorer.cpp

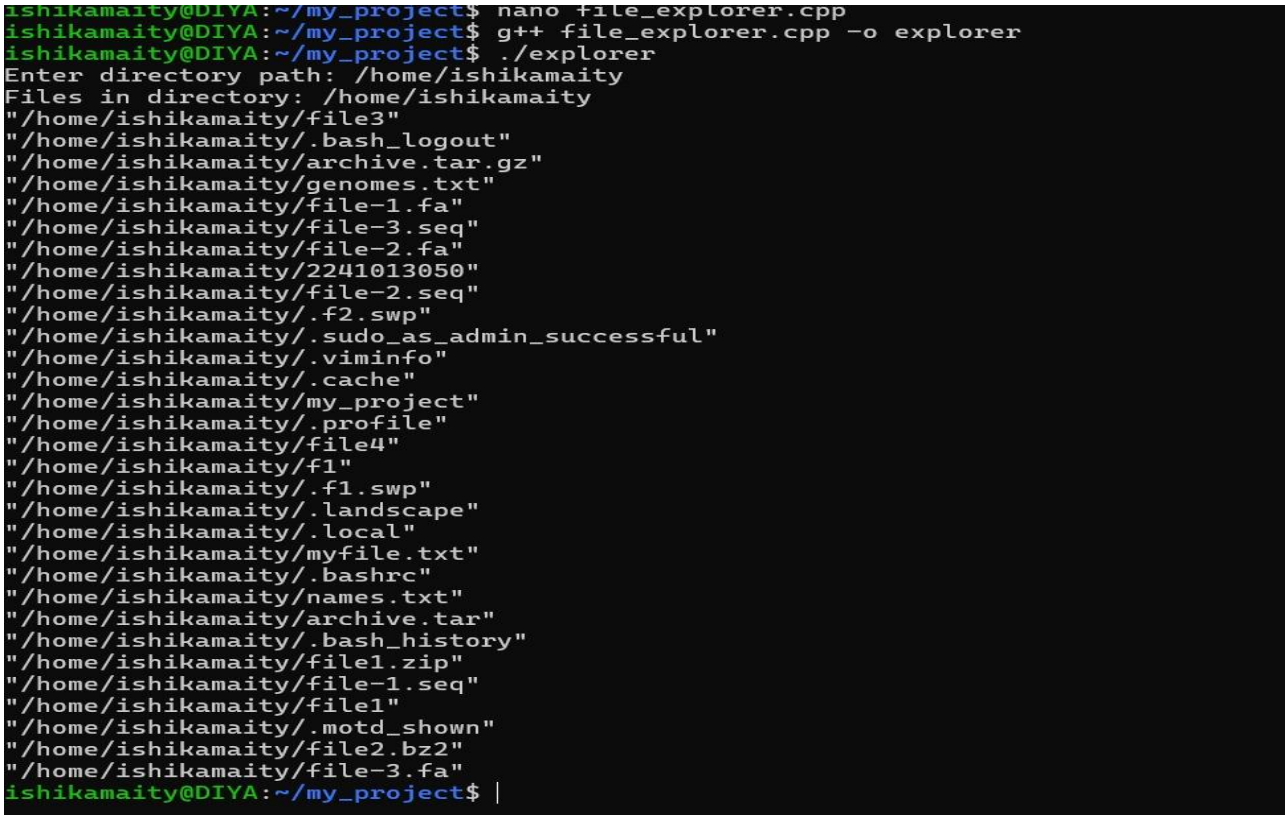
```
    }
    else if (choice == 5) {
        // Move file code
        std::string src, dest;
        std::cout << "Enter source file name: ";
        std::cin >> src;
        std::cout << "Enter new file name or destination: ";
        std::cin >> dest;
        try {
            fs::rename(path + "/" + src, path + "/" + dest);
            std::cout << "File moved/renamed.\n";
        } catch (fs::filesystem_error &e) {
            std::cout << "Error moving file: " << e.what() << std::endl;
        }
    }
    else if (choice == 6) {
        // Search file code
        std::string search;
        std::cout << "Enter file name to search: ";
        std::cin >> search;
        bool found = false;
        try {
            for (const auto &entry : fs::directory_iterator(path)) {
                if (entry.path().filename() == search) {
                    std::cout << "Found: " << entry.path() << std::endl;
                    found = true;
                }
            }
        } catch (const fs::filesystem_error& e) {
            std::cout << "Error accessing directory: " << e.what() << std::endl;
        }
        if (!found)
            std::cout << "File not found.\n";
    }
```

```
            if (!found)
                std::cout << "File not found.\n";
        }
        else if (choice == 7) {
            std::cout << "Exiting.\n";
            break;
        }
        else {
            std::cout << "Invalid choice. Please try again.\n";
        }
    }
    return 0;
}
```

3.Screenshots of output

Screenshots include:

- Listing files in a directory.

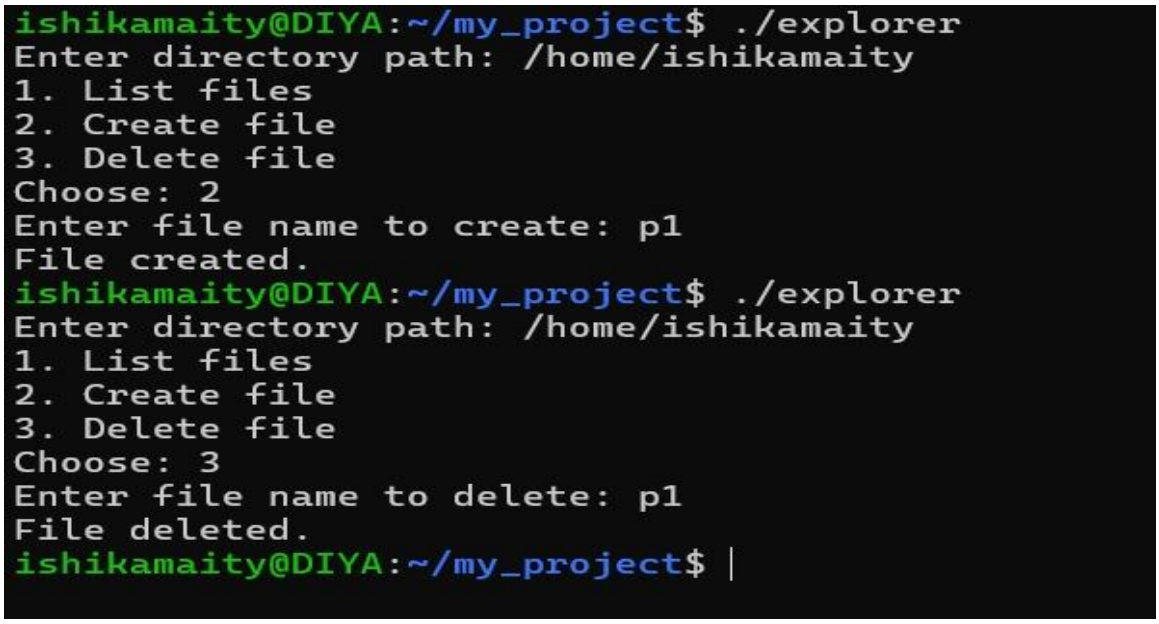


- Creating a new file.

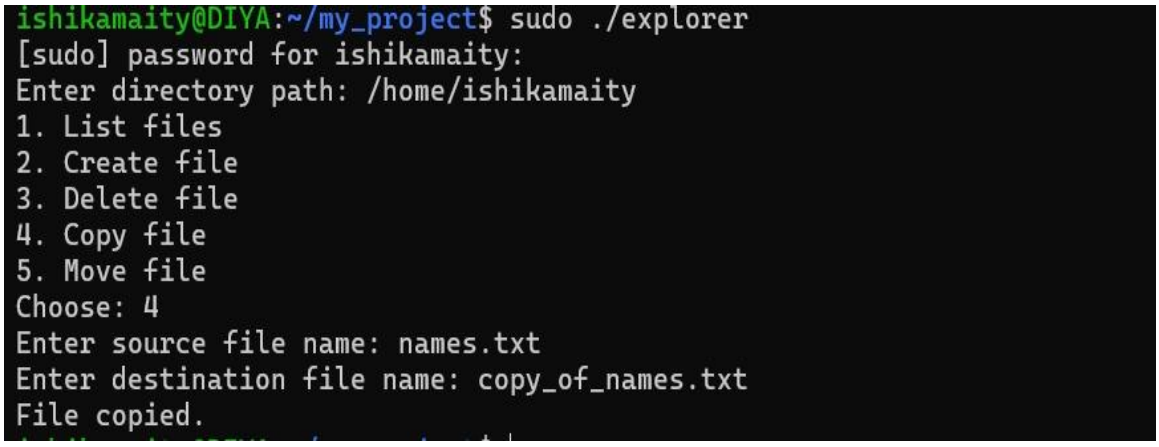
```
ishikamaity@DIYA:~/my_project$ nano file_explorer.cpp
ishikamaity@DIYA:~/my_project$ g++ file_explorer.cpp -o explorer
ishikamaity@DIYA:~/my_project$ ./explorer
Enter directory path: /home/ishikamaity
1. List files
2. Create file
3. Delete file
Choose: 1
"/home/ishikamaity/file3"
"/home/ishikamaity/.bash_logout"
"/home/ishikamaity/archive.tar.gz"
"/home/ishikamaity/genomes.txt"
"/home/ishikamaity/file-1.fa"
"/home/ishikamaity/file-3.seq"
"/home/ishikamaity/file-2.fa"
"/home/ishikamaity/2241013050"
"/home/ishikamaity/file-2.seq"
"/home/ishikamaity/.f2.swp"
"/home/ishikamaity/.sudo_as_admin_successful"
"/home/ishikamaity/.viminfo"
"/home/ishikamaity/.cache"
"/home/ishikamaity/my_project"
"/home/ishikamaity/.profile"
"/home/ishikamaity/file4"
"/home/ishikamaity/f1"
"/home/ishikamaity/.f1.swp"
"/home/ishikamaity/.landscape"
"/home/ishikamaity/.local"
"/home/ishikamaity/myfile.txt"
"/home/ishikamaity/.bashrc"
"/home/ishikamaity/names.txt"
"/home/ishikamaity/archive.tar"
"/home/ishikamaity/.bash_history"
"/home/ishikamaity/file1.zip"
"/home/ishikamaity/file-1.seq"
"/home/ishikamaity/file1"
"/home/ishikamaity/.motd_shown"
"/home/ishikamaity/file2.bz2"
"/home/ishikamaity/file-3.fa"
```

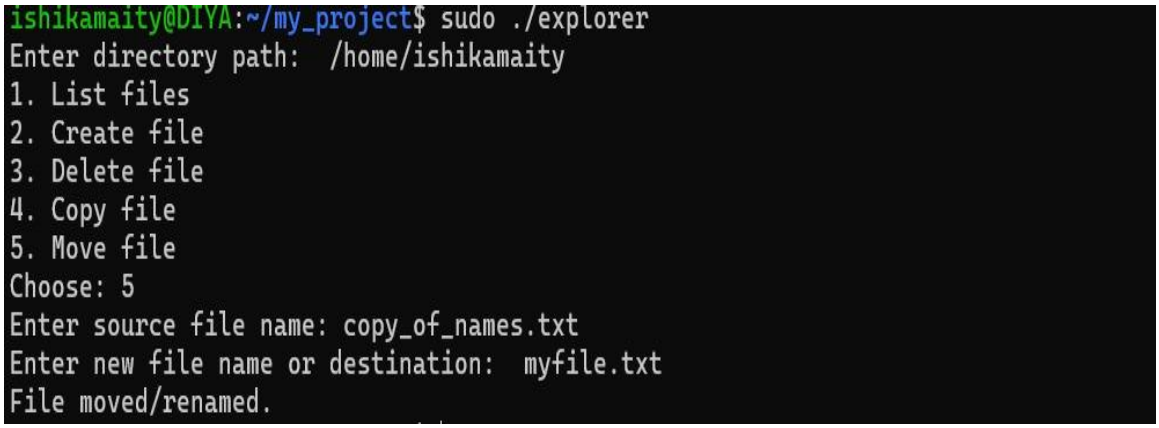
```
ishikamaity@DIYA:~/my_project$ ./explorer
Enter directory path: /home/ishikamaity
1. List files
2. Create file
3. Delete file
Choose: 2
Enter file name to create: p1
File created.
ishikamaity@DIYA:~/my_project$ ./explorer
Enter directory path: /home/ishikamaity
1. List files
2. Create file
3. Delete file
Choose: 3
Enter file name to delete: p1
File deleted.
ishikamaity@DIYA:~/my_project$
```

- Deleting a file.



- Copying and moving files.





- Searching for files.