School of Future Tech

CASE STUDY

Simple Login and Registration System

Group-12

Rima Maji (150096725215)
Yash Tambade (150096725173)
Tanvish Poojari (150096725203)

- **Create a login system with username and password**
- **Store Credentials securely in files and validate user login**

INDEX

1. Introduction to the Case Study

User authentication is an essential part of modern software systems. Almost every application, including websites, mobile apps, and desktop software, requires a login and registration system to manage users securely. The login system ensures that only authorized users can access the system, while the registration system allows new users to create accounts.

This case study focuses on the design and implementation of a Simple Login and Registration System using C++ programming language. The system allows users to register by creating a username and password and later log in using those credentials. The user information is stored in a text file, and password security is ensured using hashing techniques.

This case study demonstrates the practical use of file handling, password hashing, and object-oriented programming concepts in C++. It also highlights how secure authentication systems can be developed using basic programming tools.

---

2. Problem Statement / Case Background (Abstract)

Background

In many software applications, user authentication is required to protect sensitive data and restrict unauthorized access. Storing passwords in plain text is insecure because attackers can easily access them. Therefore,

password hashing is used to convert passwords into secure formats before storing them.

File handling provides a simple way to store and retrieve user data without using complex databases. This makes it suitable for small-scale applications and beginner-level projects.

Abstract

This case study presents the design and implementation of a Login and Registration System using C++. The system allows users to register with a username and password, which is securely stored using password hashing. The hashed password and username are saved in a text file using file handling techniques.

During login, the entered password is hashed and compared with the stored hashed password. If the credentials match, access is granted; otherwise, access is denied. The system demonstrates secure password handling, file operations, and user authentication mechanisms. This project provides a basic foundation for developing secure authentication systems in real-world applications.

---

3. Case Study Design

The Login and Registration System is designed with the following components:

User Registration Module

- User enters username and password
- Password is converted into hashed format
- Username and hashed password are stored in a file named users.txt

User Login Module

- User enters username and password
- Password is hashed again
- System compares entered credentials with stored credentials
- If matched, login is successful
- If not matched, login fails

File Storage System

- User data is stored using file handling
- Data is saved permanently for future login attempts

Menu System

The system provides options:

1. Register new user
2. Login existing user
3. Exit program

---

4. Methods & Algorithms Technology Applied in the Problem Statement / Case Study

Methods Used:

❖ File Handling

File handling is used to store and retrieve user data.

❖ Functions used:

- ofstream → to store user data
- ifstream → to read user data

Example: ofstream file("users.txt", ios::app);

Password Hashing

Password hashing is used for security. Instead of storing plain passwords, hashed passwords are stored.

Example: hash<string> hasher;

return to_string(hasher(password));

This improves security.

User Authentication Algorithm

- ❖ Steps:

Registration:

1. Take username input
2. Take password input
3. Hash password
4. Store username and hashed password in file

Login:

1. Take username input
2. Take password input
3. Hash password
4. Compare with stored data
5. Grant or deny access

Technology Stack Used

- ❖ Programming Language: C++

- ❖ Concepts Used:
- File Handling
- Password Hashing
- Functions
- Conditional Statements

- Menu-driven Programming

❖ Development Environment
- Visual Studio Code / CodeBlocks / Turbo C++

❖ Storage Method:
- Text file (users.txt)

---

5. Problem Statement / Case Study Implementation Details and Screenshots

❖ Implementation Details

The system consists of the following functions :

- HashPassword()

  Converts password into hashed format.

- RegisterUser()

  Registers a new user and stores data in file.

- LoginUser()

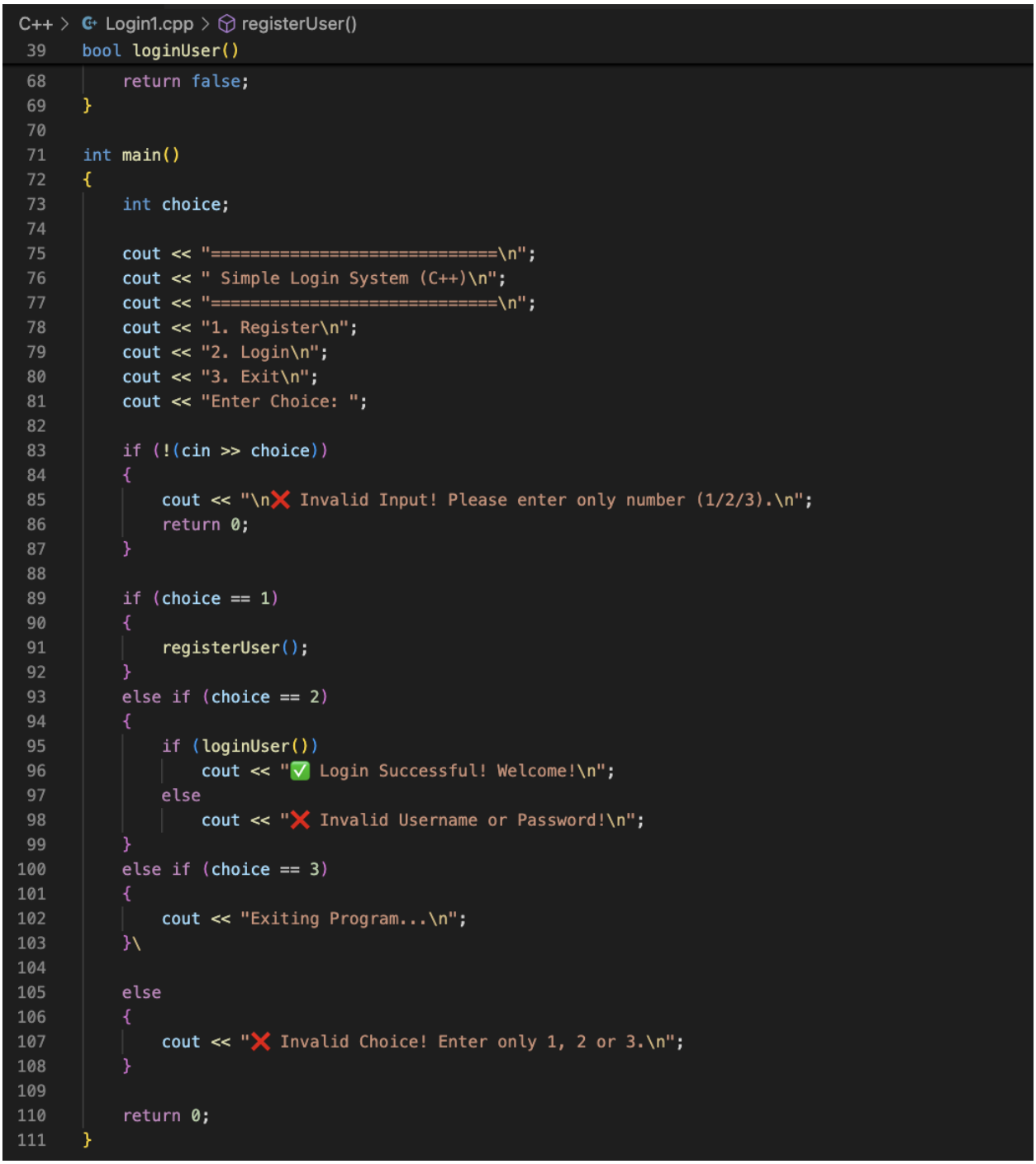
  Verifies username and password.

- Main()

Provides menu options and controls program flow.

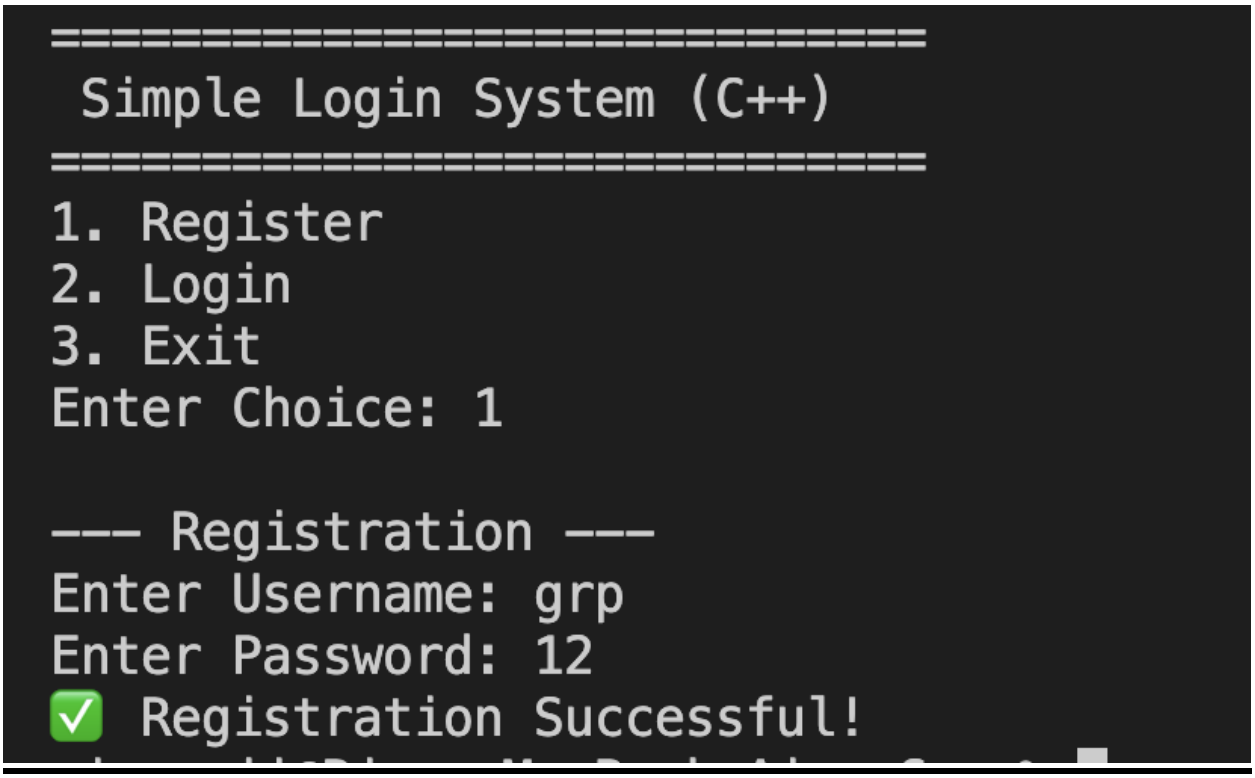
C++ > Login1.cpp > registerUser()

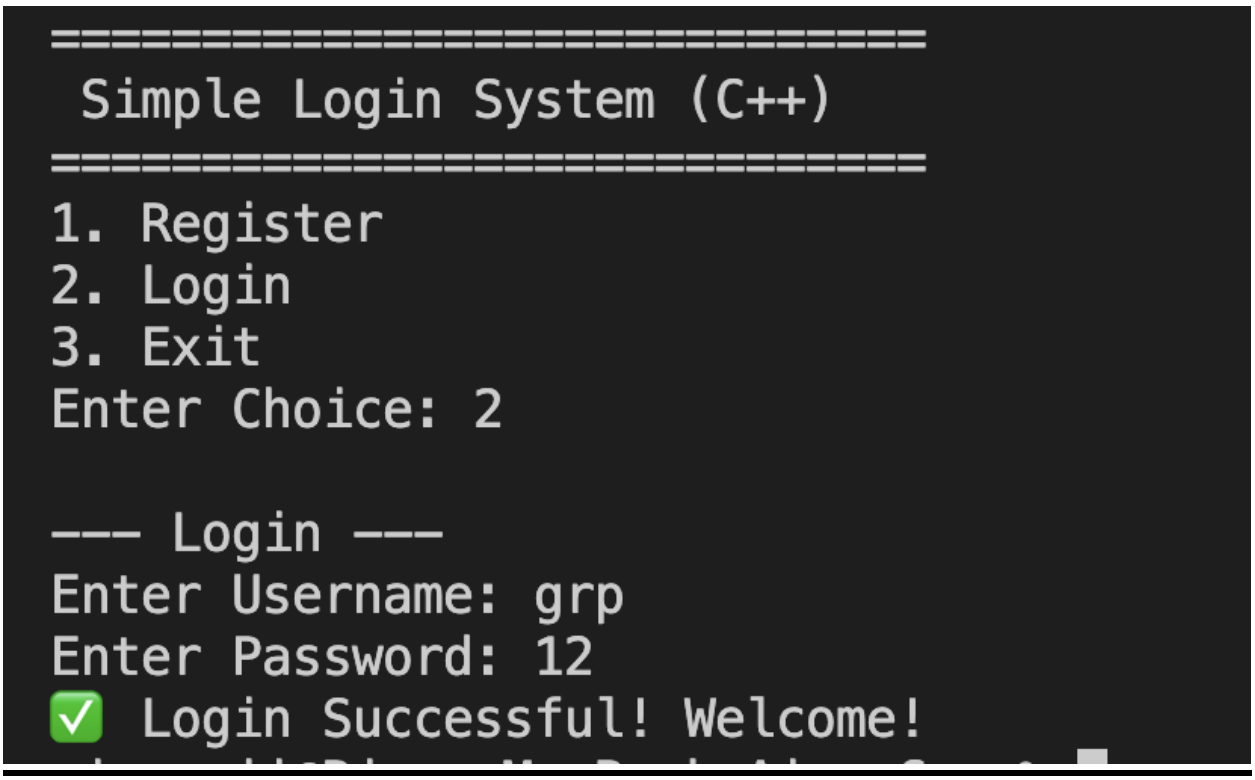
```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <functional>

using namespace std;

string hashPassword(string password)
{
    hash<string> hasher;
    return to_string(hasher(password));
}

void registerUser()
{
    string username, password;

    cout << "\n--- Registration ---\n";
    cout << "Enter Username: ";
    cin >> username;

    cout << "Enter Password: ";
    cin >> password;

    ofstream file("users.txt", ios::app);

    if (file.is_open())
    {
        file << username << " " << hashPassword(password) << endl;
        file.close();
        cout << "✅ Registration Successful!\n";
    }
    else
    {
        cout << "❌ File Error!\n";
    }
}

bool loginUser()
{
    string username, password;
    string storedUser, storedPass;

    cout << "\n--- Login ---\n";
    cout << "Enter Username: ";
    cin >> username;

    cout << "Enter Password: ";
    cin >> password;

    string hashedPass = hashPassword(password);

    ifstream file("users.txt");

    if (file.is_open())
    {
        while (file >> storedUser >> storedPass)
        {
            if (storedUser == username && storedPass == hashedPass)
            {
                file.close();
                return true;
            }
```

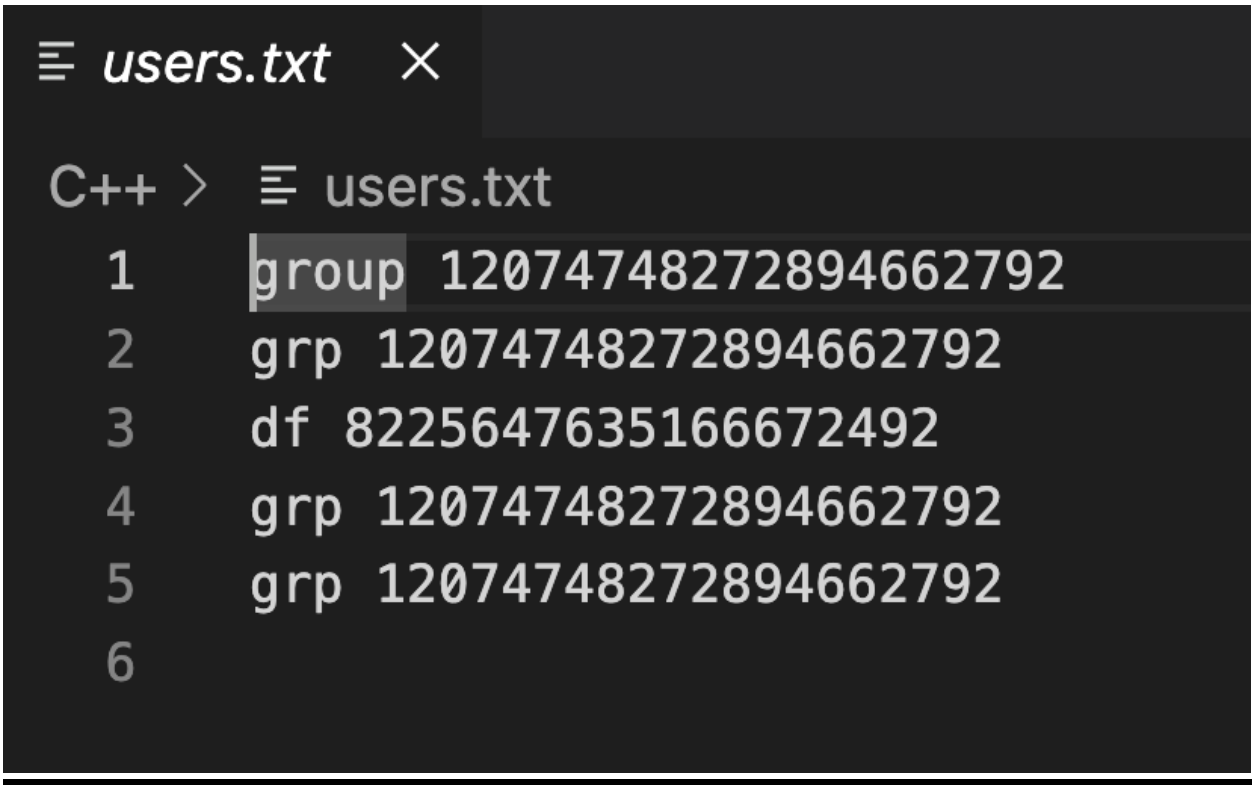


This was Code Implementation

Screenshots:







---

6. Problem Statement / Case Study Results and Conclusion

Results

The system successfully performs:

- User registration
- Secure password storage
- User login verification
- File-based data storage

Example Output:

- Registration successful
- Login successful

Conclusion

This case study successfully demonstrates the implementation of a Login and Registration System using C++. The system uses file handling to store user data and password hashing to improve security.

The project helps in understanding important programming concepts such as file handling, password security, and user authentication. This system can serve as a foundation for more advanced authentication systems used in real-world applications.

---

7. References

- C++ Programming Documentation
  https://cplusplus.com
- File Handling in C++
  https://www.geeksforgeeks.org
- Password Hashing Concepts
  https://www.geeksforgeeks.org/hashing-in-cpp
- University Lecture Notes