Project Documentation

Title page:

Course name: Basics of Programming 2

Course ID: BMEVIIIAA03
Faculty Code: 5NAA8
Full Name: Muhammad Hameez Khan
Neptune code: TFBB32

Lab Group ID: CS16D
Lab Instructor: Vaitkus Márton
Project title: Facebook like database program.
Type of the Project: Final Project.
Submission date: 19/05/2024.

Introduction

This document is a comprehensive guide to the "Facebook Simulator" program, designed to manage a simple social network system where we store the data of various users which can also join and leave several groups. The program offers functionalities such as adding new members, searching for members, deleting members, and managing groups like adding or removing a member from a group through a console-based menu system. The old user's data is loaded into the program and it is saved including with the new users data. The documentation will walk through the user interface, program execution, input/output specifications, internal program structure, and provide guidance on testing and possible future enhancements.

Program Interface

To run the program:

Environment Setup:
--Ensure that the C++ compiler is installed (e.g., GCC, Clang, MSVC). This program is developed to be run on systems that support standard C++ libraries and console operations.

--make sure that the file “facebook.txt” is in the same folder as the project.

Execution Command:
--Compile the code using a C++ compiler with a command such as:
--g++ -o facebook main.cpp
--execute it by opening the facebook.cpp on Visual Studio and pressing Ctrl+Alt+N
--by running facebook.exe application in the folder on Windows.

Termination:
--The program can be terminated by selecting the exit option which is “8” from the main menu.

Program Execution

Upon launching the program, users are greeted by a main menu with options to manage members and groups:

Main Menu Options:

1.Enter a new member in the data.
2.Search a member in the data.
3.Delete a member from the data.
4.Add a member to a group.
5.Remove a member from a group.
6.List all group members.
7.List all members in the data.
8.Exit the program.

Each option is interacted with via numerical input corresponding to the menu item. After selecting an option, users follow instructions to input additional information or make further selections.

1.Enter a new member in the data.:
User can enter a new member in the database.

2.Search a member in the data.
User can search for one or more members with their nickname.

3.Delete a member from the data.
User can delete a member from the data.

4.Add a member to a group.
User can be added to a group.

5.Remove a member from a group.
User can be removed from a group. If member is not in the group a messaged is displayed
that member is not in the group.

6.List all group members.
listing all the members in the data that have a particular group in common.

7.List all members in the data.
lists all the members in the data.

8.Exit the program.
ends the program and saves the users data in a file.

Input and Output

Input:

--User inputs are gathered using standard input (keyboard) through terminal in the console. -------Inputs include member details (name, family name, given name, place, groups) and menu selections.

Output:

Outputs are displayed on the console, providing feedback on the operations performed such as:
--Confirmation messages if a tasks is executed without any errors.
--exception message : if a tasks is not executed as expected or receives an unexpected input.
--confirming the addition or deletion of members.
--displaying search results.
--listing group members.

Example:

```
Enter nick name:Muhammad
Enter family name:khan
Enter given name:hameez
Enter place name:hyderabad
Enter date of birth(yy/mm/dd): 2 2 2
Do you want to join groups?(y/n)n
```

Program Structure

Files:

The program has a total of 4 files

1. *facebook.cpp:*
   Contains the main function of the program.
2. *Container.cpp:*
   Contains the container class
3. *Member.cpp*:
   Contains the member class

4 *.date.cpp*
   Contains the date class

Main Program:

the most important function of the program from where all the functions are called from, initializes main components and objects and displays main menu.

Classes:

The program is structured into several classes:

Date Class: stores the date of each member and manages date-related functionalities.

Member Class: Stores the personal data of each member and has date class included in it to store the date in an efficient way along with attributes like name, groups, and date of birth.

Container Class: stores the members in a dynamic array and remembers the number of members.

File Management :

The program loads and stores the data of the members from a file called “facebook.txt” using classes such as string stream, ostream and istream data of each member is stored in separate line and each value of a member is separated by commas

Users Interaction :
The user can navigate through the program using the CLI or terminal in Visual studio Code.

Testing and Verification

Method for testing:

The program was tested by manually inputting data and observing the outputs in order to ensure that is no unexpected behaviour and determining and fixing it (if any).
Each function was tested in isolation and then as part of the system to ensure integration.
numerous different types of input were given to the program to detect any output that might crash or throw runtime error in the program.

Verification:

Functional testing was conducted to verify that each feature meets the requirements specified in the user stories.

Integration Testing

Verifies that all the components of the program are working together. This testing involves the interactions between the container class and the member objects and how the data flows within the system during various operations.

Boundary testing:

Testing the program with extreme values of input such as dates or an different value of menu.
recording the behaviour of the program and fixing it incase of malfunction.

User Interface Testing :

Ensuring that the user has no difficulty in seeing the menu or other outputs of the program.

Environment Setup:

Checking the IDE of the program and running it on a standard C++ compiler .
Using methods that does not have undefined behaviour .
ensuring that the program can run in every other popularly used IDE.

Test Results :

The testing regimen uncovered numerous challenges pertaining to input validation, file management, and data manipulation. Incorrect inputs, such as improperly formatted dates, were effectively managed, ensuring the program remained stable and did not succumb to crashes.

Boundary testing exposed several uncalculated behaviors when subjected to extreme conditions, necessitating enhancements in input validation and error management strategies. Integration tests illustrated seamless interactions among various system components, conforming that data was accurately exchanged and processed across different modules.

Furthermore, user interface testing corroborated the effectiveness and clarity of the console prompts, thereby affirming that the system offers an agreeable user experience. These tests collectively highlighted the robustness of the system architecture and its suitability for user interaction, ensuring operational reliability under diverse conditions.

Improvements and Extensions:

The Facebook application, as it currently stands, offers basic functionalities for a social network system using a console-based interface. While the program fulfills its primary objectives, there are several areas where improvements can be made and extensions can be considered which are listed below:

User Interface:

Current State: The interface is entirely console-based, which might not be the most user-friendly approach, especially for non-technical users.

Improvement: Enhance the user interface by integrating a simple graphical user interface (GUI) using libraries like Qt or GTK. This would make the application more accessible and easier to navigate.

Performance Optimization:

Current State: The application handles basic operations well, but performance may degrade with large data sets due to its current data handling strategies.

Improvement: Optimize data structures and algorithms, possibly integrating more efficient data management techniques like indexing or hash tables for faster search and retrieval operations.

User Authentication:

Extension: Add user authentication features to manage access to the system, with login and password protection.

Benefit: Enhances security and personalization, allowing the system to support multiple users with private data.

Difficulties encountered:

Implementing File Handling

as the number of groups each user joins is different. It was somewhat a challenging task to figure out an algorithm to efficiently store and read the data to and from the file.

Implementing OOP

It was bit hectic to figure out how to effectively use the OOP feature of the C++ programming language.

Conclusion:

The "Facebook Simulator" project offers a foundational system for managing a simple social network through a console-based interface. It includes functionalities for member management and group activities, demonstrated through detailed documentation on system operation, structure, and user interaction.

Designed for simplicity and expandability, the system handles fundamental operations such as adding, deleting, and searching members, managing groups, and data persistence. Despite its capabilities, areas for improvement have been identified, including enhancing the user interface, implementing robust error handling.

Future enhancements could transform the simulator from a console-based application to a more robust platform, potentially serving educational purposes or as a prototype for larger projects. This evolution would enhance user engagement and enable exploration of more complex social networking dynamics.

Testing and verification have confirmed the system's functionality under various conditions, emphasizing the need for ongoing refinement as new features are implemented. This project demonstrates the practical application of software development principles in C++, offering a solid foundation for educational exploration and future development in software design and application problem-solving.

References

1. Reading and Writing to Files:
https://www.youtube.com/watch?v=Cz4fI-TUjVk&t=91s
2. Clearing further basics:
https://www.geeksforgeeks.org/c-plus-plus/