Technological Institute of the Philippines -Quezon City
Department of Computer Engineering

Lost and Found Management System: C++ Implementation Using Linked Lists Based

Prepared By: Blackbox:

Mamaril, Justin Kenneth

Reyes, Alexzander

San Jose, Alexander

San Juan, Edson Ray

Titong, Lee Ivan

A Final Project Submitted to:

Ma’am Maria Rizette Sayo

In Partial Fulfillment of the Requirements for

Data Structures and Algorithm (CPE 010)

December 2024

Lost and Found Management System: C++ Implementation Using Linked Lists Based	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed:** November 8, 2024
Section: CPE21S4	**Date Submitted:** December 12, 2024
Name: Mamaril, Justin Kenneth Reyes, Alexzander San Jose, Alexander San Juan, Edson Ray Titong, Lee Ivan	**Instructor:** Ma'am Maria Rizette Sayo

1. Objective(s)

General Objectives:

- follow a specific data structure to be implemented in our program:
  - Linked List
- The created output can be related to Computer Engineering

Specific Objectives:

- The program will be done using C++ programming language
- The program will utilize linked lists as a data structure for processing the lost and found registration
- The terminal of the program and its output will be in a Terminal Window

2. Intended Learning Outcomes (ILOs)

- Develop a C++ code that utilizes linked list for managing lost and found items
- To learn file handling for data storage: Students will develop skills in saving and loading data from text files to maintain records of users email, student number and password, lost items, and claimed items.
- Students will learn to design and implement a console-based application with an interactive user interface.

3. Discussion

1. Introduction

1.1. Background of the Study

In a world characterized by rapid technological advancements, the need for efficient and smooth systems to manage everyday challenges becomes increasingly evident. One common issue in schools, public spaces, and workplaces is the loss of items. This is a common phenomenon that can lead to inconvenience and frustration for both individuals and administrators. To address this issue, the researchers propose creating a "Lost and Found Management System"-based Linked List C++ program. The program aims to transform the way lost items are reported, tracked, and returned, ultimately enhancing the overall experience for users and administrators alike (Ariyarathna, U. 2023).

The Lost and Found Management System seeks to harness the power of modern technology, to create a comprehensive solution that addresses the intricacies of item management and retrieval. By creating C++-based code, user-friendly interfaces, and efficient backend processes, the system offers a dynamic and innovative approach to lost item management (Ariyarathna, U. 2023). This project proposal outlines the key features such as Submit a Lost Item, Search for Items, View Claimed Items and Claim an Item, its functionalities, and benefits of the proposed program, delves into the problem statement it seeks to address, and provides an overview of the objectives, and significance of the study.

1.2. Development of the Program

The main data structure of the Lost and Found Management System is linked lists, which are implemented in C++ to meet the demand for effective tracking, reporting, and claiming of lost things. Because linked lists can be added, deleted, and traversed efficiently to handle item records, they are especially well-suited for this project. A number of fundamental modules will make up this software, each designed to handle particular functions including item submission, searching, showing claimed items, and processing claims.

1.3. Significance of the Study

This research is made with the aim to provide information and knowledge to know the effectiveness of the lost and found management system of the Technological Institute of the Philippines.

Students. The study's findings may assist students reduce the hassle of reporting lost and discovered things on campus. Students frequently misplace their belongings on campus, so a system like this would be handy.

Instructors. The study can be beneficial to the instructors because this can help their students determine the scores and grades for major exams with the desired grade and class standing. It can also be a tool to calculate the overall grades of the students.

Future Researchers. This study will serve as a guide, reference, or blueprint for future researchers in making their projects about calculators or studies about the academic performance of the students. They have the potential to expand and improve the project.

1.4.Scope and Delimitation

The scope of this research is mainly focused on 4 choices; Submit a Lost Item, Search for Items, View Claimed Items and Claim an Item. These processes will be achieved by providing the needed information, which varies for each process.

For the delimitation on the other hand, the study is specifically focused on the usage for the college students at the Technological Institute of the Philippines, Quezon City Campus, the system is calibrated to work within the parameters set by TIP and may not be compatible with other institutions or campuses that have different procedures or requirements. Modifications to the system's functionalities or workflows are not supported, as it follows a particular design and structure intended exclusively for TIP's environment.

2. Methodology

2.1. Standard Library Headers

<iostream> - provides objects which can read user input and output data to the console or to a file. The code uses cout and cin to display text and gather inputs from the user.

<fstream> - provides classes for reading and writing into files or data streams, the code use the header to store the data in text file (ifstream file("accounts.txt");). This also allows the program to save and retrieve data, making it persistent even after the program ends.

<sstream> - provides the manipulation of strings recognizing it as streams, making it a useful header for parsing and formatting string data. This is particularly useful for breaking down complex strings (e.g., lost_items.txt file) into smaller, manageable parts.

<iomanip> - provides advanced utilities for formatting input and output, allowing you to control how data is presented on the console. In the researchers code it utilize <iomanip> to create neatly formatted tables when listing lost items or search results, ensuring that data fields like item ID, description, and location are aligned and easy to read.

2.2. Data Structure

Linked list - is a dynamic data structure that stores data in nodes, where each node contains the data and a pointer to the next node, forming a chain-like sequence. In the researchers program, the linked list efficiently manages lost items, allowing for easy addition, traversal, and removal of items as they are reported or claimed.

2.3 User Interface Design

User Interface Design (UI Design) is the process of creating the visual and interactive elements of a software application or device to ensure it is user-friendly and efficient. It involves designing a console based program that interacts between the user and the system. The interface will feature a console-based, menu-driven structure with the following options:

Login/Register

- When the user selects this option, the system will first check if the user is already registered.
- If the user is registered, they will be prompted to log in by entering their username and password.
- If the user is not registered, they will be asked to create an account by providing the necessary details (e.g., username, password, email, contact info) and will be automatically logged in after successful registration.

Submit Lost Item
Once logged in, users can report a lost item by providing details such as item description, location, and time lost.

Claim an Item
Logged-in users can search through the list of lost items to find an item they wish to claim. They will be able to view details and claim the item if it matches their lost property.

Display Queue
Displays a list of all the lost and found items in the system, showing items submitted by users.

Exit

Closes the application.

4. Materials and Equipment

- A computer or laptop with an operating system that supports C++ development (Windows, macOS, or Linux
- A C++compiler with version (C++11 or latest)
- Dev C++ Embarcadero
- Github

5. Procedure

```
C/C++
Function: Login

While loggedIn is false:
    Prompt user for email and password.
    If user inputs "N", return to login menu.
    Search accounts.txt for matching email and password.
    If match found:
        Set loggedIn = true.
        Display success message.
    Else:
        Display error message.
```

The Login function asks the user for their email and password and checks if they match any record in accounts.txt. If a match is found, the user is logged in. If the user chooses to cancel by typing "N", they are returned to the login menu.

```
C/C++
Function: SignUp

Prompt user for email:
    If email exists in accounts.txt, display error and retry.
    Else:
        Prompt for student number and password.
        Save to accounts.txt.
        Display success message.
```

The SignUp function asks the user for an email, checks if it already exists in the accounts.txt file, and if it doesn't, the user is prompted to provide additional information (student number and password). The new account is then saved to accounts.txt.

```
C/C++
Function: SubmitLostItem

Create new LostItem with details:
    Prompt for itemName, locationLost, dateLost, contactInfo.
    If user inputs "N", return to main menu.
    Increment itemCounter and assign itemID.
    Save LostItem to linked list (lostItemsHead) and lost_items.txt.
    Display confirmation.
```

The SubmitLostItem function collects details from the user about a lost item (name, location, date, and contact information). If the user cancels the process by typing "N", it returns to the main menu. The item is then saved to both a linked list and the lost_items.txt file.

```
C/C++
Function: SearchForItems

Display all items with status Lost from lost_items.txt.
Prompt user for itemID to claim:
    If "N", return to main menu.
    Else:
        Mark item as Claimed in lost_items.txt.
        Display confirmation.
```

The SearchForItems function displays a list of items marked as "Lost" from the lost_items.txt file. The user can then select an item to claim by entering its item ID. If they input "N", the function returns to the main menu. The status of the item is updated to "Claimed".

```
C/C++
Function: ListOfItems
```

```
Read all items from lost_items.txt.
Display all items in a formatted table.
Free memory used by the linked list.
```

The ListOfItems function reads all items from the lost_items.txt file and displays them in a formatted table. After displaying, it frees any memory used by the linked list.

```
C/C++
Function: ViewClaimedItems

Read all items from lost_items.txt with status Claimed.
Display claimed items in a formatted table.
Free memory used by the linked list.
```

The ViewClaimedItems function reads and displays all items with a "Claimed" status from lost_items.txt, presenting them in a formatted table. It also frees the memory used by the linked list.

```
C/C++
Function: Main

Call Intro().
Loop until user logs in or exits:
    Display ShowLoginMenu().
    Based on choice:
        Call Login().
        Call SignUp().
    Exit program if chosen.

If loggedIn is true:
    Loop until user exits main menu:
        Display ShowMainMenu().
        Based on choice:
            Call appropriate function (SubmitLostItem, SearchForItems, ListOfItems,
ViewClaimedItems).
            Exit program if chosen.
```

The main function first shows the intro screen. Then it loops until the user logs in or chooses to exit. After successful login, the user is presented with the main menu where they can choose to submit lost items, search for items, view a list of items, or view claimed items. The program exits if the user chooses to do so.

Flowchart:

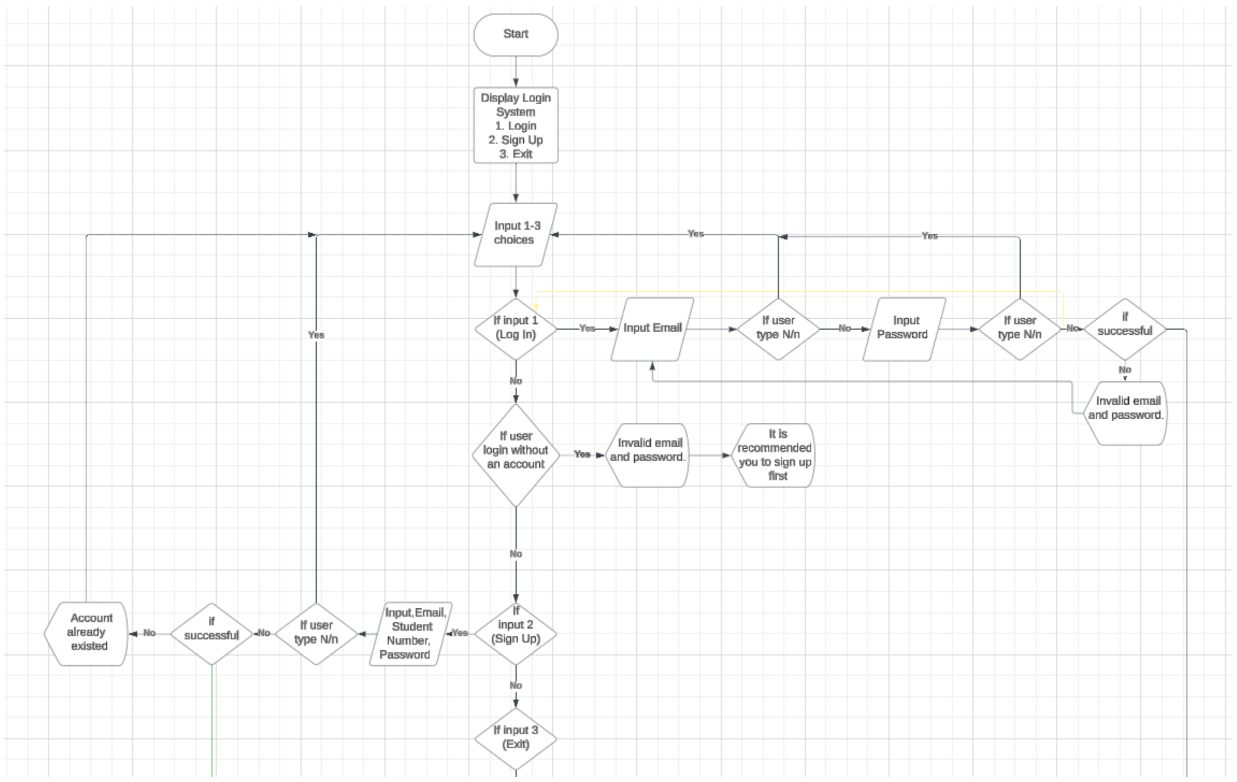


Figure 5.1: *Flowchart of the Program: Signin and Login Option*

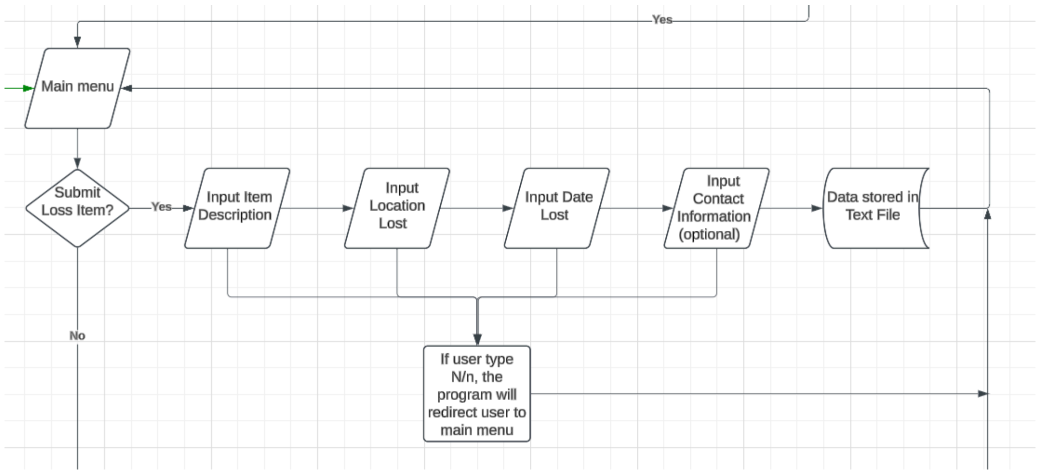


Figure 5.2: *Flowchart of the Program: Main Menu and Option 1(Submit Lost Item)*

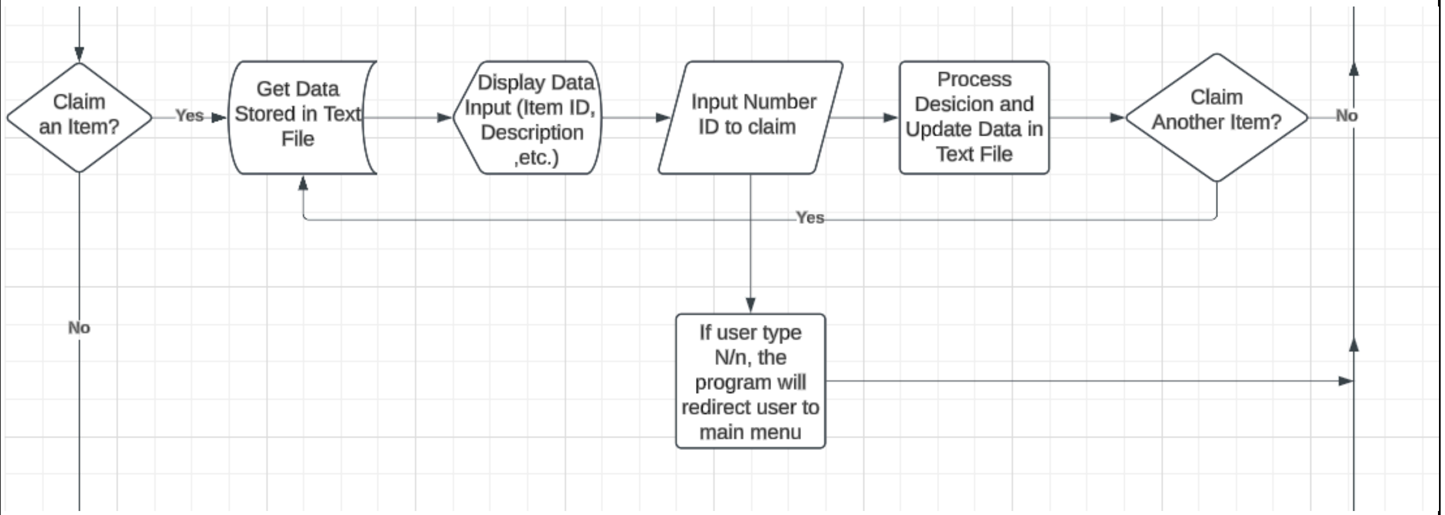

Claim an Item?
Yes
Get Data Stored in Text File
Display Data Input (Item ID, Description ,etc.)
Input Number ID to claim
Process Desicion and Update Data in Text File
Claim Another Item?
No
Yes
No
If user type N/n, the program will redirect user to main menu

Figure 5.3: *Flowchart of the Program: Option 2(Claim an Item)*

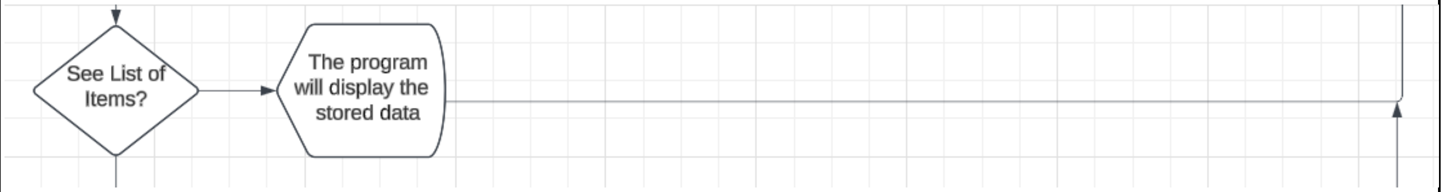


Figure 5.4: *Flowchart of the Program: Option 3(List of Items)*

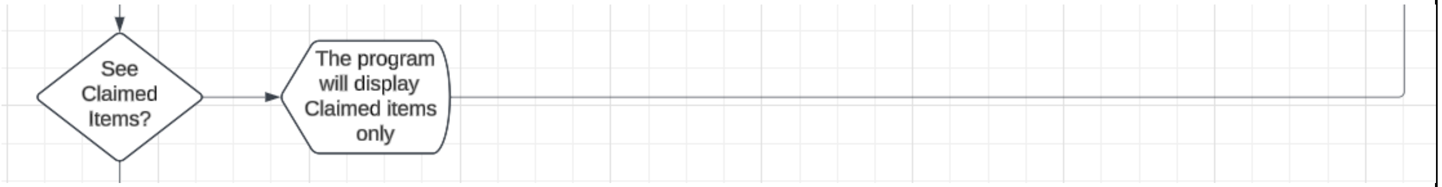


Figure 5.5: *Flowchart of the Program: Option 4(View Claimed Items)*

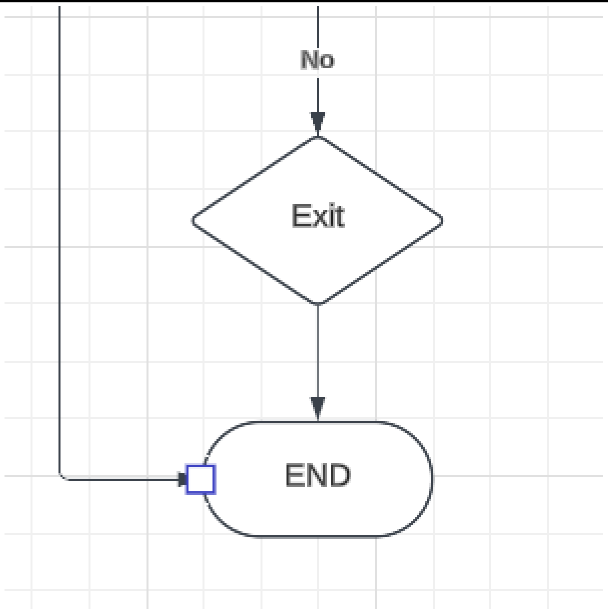


Figure 5.6: *Flowchart of the Program: Option 5 (Exit) and end of program*

Pseudocodes:

```
C/C++
Function: Login

While loggedIn is false:
      Prompt user for email and password.
      If user inputs "N", return to login menu.
      Search accounts.txt for matching email and password.
      If match found:
            Set loggedIn = true.
            Display success message.
      Else:
            Display error message.
-----------------------------------------------------------------
Function: SignUp

Prompt user for email:
      If email exists in accounts.txt, display error and retry.
      Else:
            Prompt for student number and password.
            Save to accounts.txt.
            Display success message.
-----------------------------------------------------------------
Function: SubmitLostItem
```

```
Create new LostItem with details:
       Prompt for itemName, locationLost, dateLost, contactInfo.
       If user inputs "N", return to main menu.
Increment itemCounter and assign itemID.
Save LostItem to linked list (lostItemsHead) and lost_items.txt.
Display confirmation.
-----------------------------------------------------------------
Function: SearchForItems

Display all items with status Lost from lost_items.txt.
Prompt user for itemID to claim:
       If "N", return to main menu.
       Else:
              Mark item as Claimed in lost_items.txt.
              Display confirmation.
-----------------------------------------------------------------
Function: ListOfItems

Read all items from lost_items.txt.
Display all items in a formatted table.
Free memory used by the linked list.
-----------------------------------------------------------------
Function: ViewClaimedItems

Read all items from lost_items.txt with status Claimed.
Display claimed items in a formatted table.
Free memory used by the linked list.
-----------------------------------------------------------------
Main Function:

Call Intro().
Loop until user logs in or exits:
       Display ShowLoginMenu().
       Based on choice:
              Call Login().
              Call SignUp().
       Exit program if chosen.
If loggedIn is true:
       Loop until user exits main menu:
              Display ShowMainMenu().
              Based on choice:
                     Call appropriate function (SubmitLostItem,
SearchForItems, ListOfItems, ViewClaimedItems).
                     Exit program if chosen.
```

6. Output

Source Code:

C/C++

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <iomanip>

using namespace std;

string email, password, storedEmail, storedPassword, studentNumber, storedStudentNumber;
bool loggedIn = false;

void Intro() // I N T R O
{
    cout << "\n\n\n\t\t  MANAGEMENT SYSTEM          ";
    cout << "\n\n\t\t    LOST AND FOUND             ";
    cout << "\n\n\t\t     DSA PROJECT               ";
    cout << "\n\n\n\tMADE BY : GROUP 6 BLACKBOX     ";
    cout << "\n\tSCHOOL : TECHNOLOGICAL INSTITUTE   " << endl;
    cout << "                       OF THE          " << endl;
    cout << "                     PHILIPPINES       " << endl;
    cout << "\n\n\n\n\t Press Enter to continue...  ";
    cin.get();
}

void ShowLoginMenu() // L O G  I N
{
    system("cls");
    cout << "\n\n\t\tLOGIN SYSTEM                 " << endl;
    cout << "\n\n\t 1. Login                      " << endl;
    cout << "\n\n\t 2. Sign Up                    " << endl;
    cout << "\n\n\t 3. Exit                       " << endl;
    cout << "\n\n\t Select an option (1-4):       ";
}

void ShowMainMenu() // M A I N   M E N U
{
    system("cls");
    cout << "\n\n\n\t MAIN MENU                       " << endl;
    cout << "\n\n\t 1. Submit a Lost Item            " << endl;
    cout << "\n\n\t 2. Claim an Item                 " << endl;
    cout << "\n\n\t 3. List of Items                 " << endl;
    cout << "\n\n\t 4. View Claimed Items            " << endl;
    cout << "\n\n\t 5. Exit                          " << endl;
    cout << "\n\n\t Please Select Your Option (1-5): ";
}

void Login()  // L O G  I N
{
    cout << "\n\n\t ============================ Log In ============================ " <<
endl;

    while (!loggedIn)
    {
        cout << "\n\n\t Do you want to log in? (Enter your School Email)/(N): ";
```

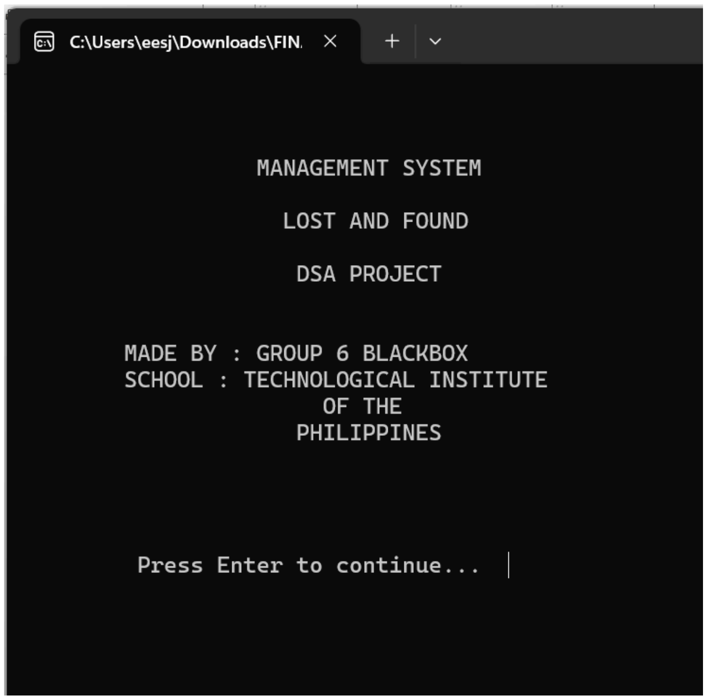
```cpp
        cin >> email;

        if (email == "N" || email == "n")
        {
            cout << "\n\n\t Returning to the login menu...\n";
            return;
        }

        cout << "\n\n\t Enter Password/(N): ";
        cin >> password;

             if (password == "N" || password == "n")
        {
            cout << "\n\n\t Returning to the login menu...\n";
            return;
        }

        ifstream file("accounts.txt");
        bool found = false;

        while (file >> storedEmail >> storedStudentNumber >> storedPassword)
        {
            if (email == storedEmail && password == storedPassword)
            {
                loggedIn = true;
                cout << "\n\n\t Login Successful! Press Enter to continue...\n";
                return;
            }
        }

        if (!found)
        {
            cout << "\n\n\t Invalid! Make sure you have entered the correct email and
password.\n";
        }
    }
}

void SignUp() // S I G N  U P
{
    cout << "\n\n\t ============================ Sign Up ============================ \n";

    while (true)
    {
        cout << "\n\n\t Do you want to sign up? (Enter your School Email)/(N): ";
        cin >> email;

        if (email == "N" || email == "n")
        {
            cout << "\n\n\t Returning to the login menu...\n";
            return;
        }
        ifstream file("accounts.txt");
        bool emailExists = false;

        while (file >> storedEmail >> storedStudentNumber >> storedPassword)
        {
            if (email == storedEmail)
            {
                emailExists = true;
                break;
            }
        }

        if (emailExists)
        {
            cout << "\n\n\t This email is already registered!\n";
            continue;
        }

        cout << "\n\n\t Enter your Student Number/(N): ";
        cin >> studentNumber;
             if (studentNumber == "N" || studentNumber == "n")
        {
            cout << "\n\n\t Returning to the login menu...\n";
            return;
        }
        cout << "\n\n\t Enter Password/(N): ";
        cin >> password;
             if (password == "N" || password == "n")
        {
            cout << "\n\n\t Returning to the login menu...\n";
            return;
        }

        ofstream outFile("accounts.txt", ios::app);
        outFile << email << " " << studentNumber << " " << password << endl;

        cout << "\n\n\t Sign Up Successful! You can now log in.\n";
        return;
    }
}

struct LostItem // V A R I A B L E S
{
    int itemID;
    string itemName;
    string locationLost;
    string dateLost;
    string contactInfo;
    string email;
    string studentNumber;
    string status;
    LostItem* next;
};
```

```
LostItem* lostItemsHead = nullptr;
int itemCounter = 0;

void SubmitLostItem(LostItem*& head, const string& email, const string& studentNumber) // S U
B M I T   L O S T   I T E M S
{
      system("cls");
    LostItem* newItem = new LostItem();

    itemCounter++;
    newItem->itemID = itemCounter;

    cout << "\n\n\t ============== SUBMIT LOST/FOUND ITEM ============== \n";
    cout << "\n\t Enter Item Description (e.g., 'Black Wallet')/(N): ";
    cin.ignore();
    getline(cin, newItem->itemName);

    if (newItem->itemName == "N" || newItem->itemName == "n")
       {
        cout << "\n\n\t Returning to the Main Menu...\n";
        delete newItem;
        return;
    }

    cout << "\n\t Enter Location Lost (e.g., 'Library, 2nd floor')/(N): ";
    getline(cin, newItem->locationLost);

    if (newItem->locationLost == "N" || newItem->locationLost == "n")
       {
        cout << "\n\n\t Returning to the Main Menu...\n";
        delete newItem;
        return;
    }

    cout << "\n\t Enter Date Lost (e.g., '2024-11-11')/(N): ";
    getline(cin, newItem->dateLost);

    if (newItem->dateLost == "N" || newItem->dateLost == "n")
       {
        cout << "\n\n\t Returning to the Main Menu...\n";
        delete newItem;
        return;
    }

    cout << "\n\t Enter Contact Information (optional, press Enter to skip)/(N): ";
    getline(cin, newItem->contactInfo);

    if (newItem->contactInfo == "N" || newItem->contactInfo == "n")
       {
        cout << "\n\n\t Returning to the Main Menu...\n";
        delete newItem;
        return;
    }

    newItem->email = email;
    newItem->studentNumber = studentNumber;
    newItem->status = "Lost";
    newItem->next = head;
    head = newItem;

    ofstream outFile("lost_items.txt", ios::app);
    if (outFile.is_open())
       {
        outFile << newItem->itemID << "|"
                << newItem->itemName << "|"
                << newItem->locationLost << "|"
                << newItem->dateLost << "|"
                << newItem->contactInfo << "|"
                << newItem->email << "|"
                << newItem->studentNumber << "|"
                << newItem->status << endl;
        outFile.close();
        cout << "\n\n\t Item successfully!";
    } else {
        cout << "\n\n\t Error: Unable to save item to the txt files.";
    }

    cout << "\n\n\t Item successfully submitted! Here are the details:\n";
    cout << "\t Item ID: " << newItem->itemID << endl;
    cout << "\t Email: " << newItem->email << endl;
    cout << "\t Student Number: " << newItem->studentNumber << endl;
    cout << "\t Status: " << newItem->status << endl;

    cout << "\n\n\t Press Enter to return to the Main Menu...";
    cin.get();
}

void SearchForItems() // C L A I M  AN  I T E M S
{
       system("cls");
    ifstream inFile("lost_items.txt");
    ofstream tempFile("temp_lost_items.txt");

    if (!inFile.is_open() || !tempFile.is_open()) {
        cout << "\n\n\t Error: Unable to process the file.\n";
        return;
    }

    string line;
    bool hasLostItems = false;

    cout <<
"\n=============================================================================================
==========================================\n";
    cout << "   Item ID    Description      Location Lost      Date Lost      Contact Info
```

```
EmailUsed        StudentNumber         Status\n";
    cout <<
"============================================================================================
=========================================\n";

    while (getline(inFile, line)) {
        stringstream ss(line);
        string itemID, description, location, date, contactInfo, email, studentNumber,
status;

        getline(ss, itemID, '|');
        getline(ss, description, '|');
        getline(ss, location, '|');
        getline(ss, date, '|');
        getline(ss, contactInfo, '|');
        getline(ss, email, '|');
        getline(ss, studentNumber, '|');
        getline(ss, status, '|');

        if (status == "Lost") {
            hasLostItems = true;

            cout << setw(6) << itemID
                 << setw(14) << description
                 << setw(20) << location
                 << setw(20) << date
                 << setw(14) << contactInfo
                 << setw(13) << email
                 << setw(19) << studentNumber
                 << setw(19) << status << endl;
        }
    }

    if (!hasLostItems)
       {
        cout << "\n\n\t No lost items found.\n";
        inFile.close();
        tempFile.close();
        remove("temp_lost_items.txt");
        cout << "\n\t Press Enter to return to the main menu...";
        cin.ignore();
        cin.get();
        return;
    }

    inFile.clear();
    inFile.seekg(0, ios::beg);

    string itemIDToClaim;

    cout << "\n\n\t Enter the Number of Item ID you would like to Claim/(N): ";
    cin >> itemIDToClaim;
    if (itemIDToClaim == "N" || itemIDToClaim == "n") {
        cout << "\n\n\t Returning to the main menu...\n";
        inFile.close();
        tempFile.close();
        remove("temp_lost_items.txt");
        return;
    }

    bool itemFound = false;

    while (getline(inFile, line)) {
        stringstream ss(line);
        string itemID, description, location, date, contactInfo, email, studentNumber,
status;

        getline(ss, itemID, '|');
        getline(ss, description, '|');
        getline(ss, location, '|');
        getline(ss, date, '|');
        getline(ss, contactInfo, '|');
        getline(ss, email, '|');
        getline(ss, studentNumber, '|');
        getline(ss, status, '|');

        if (itemID == itemIDToClaim && status == "Lost") {
            itemFound = true;

            cout << "\n\t Item ID: " << itemID
                 << "\n\t Description: " << description
                 << "\n\t Location Lost: " << location
                 << "\n\t Date Lost: " << date
                 << "\n\t Contact Info: " << contactInfo
                 << "\n\t Email Used: " << email
                 << "\n\t Student Number: " << studentNumber
                 << "\n\t Status: " << status << endl;

            cout << "\n\t --------------------------------------------------\n";
            cout << "\t Successfully Claimed by Email Used: " << email
                 << ", Student Number: " << studentNumber << endl;

            status = "Claimed";
        }

        tempFile << itemID << "|" << description << "|" << location << "|" << date << "|" <<
contactInfo << "|" << email << "|" << studentNumber << "|" << status << endl;
    }

    inFile.close();
    tempFile.close();

    if (itemFound) {
        remove("lost_items.txt");
        rename("temp_lost_items.txt", "lost_items.txt");
        cout << "\n\n\t Item successfully claimed! Would you like to claim another? (Y/N): ";
        char choice;
        cin >> choice;
        if (choice == 'Y' || choice == 'y') {
            SearchForItems();
        }
    } else {
        cout << "\n\n\t Item ID not found or already claimed.\n";
        remove("temp_lost_items.txt");
    }

    cout << "\n\t Press Enter to return to the main menu...";
    cin.ignore();
    cin.get();
}

void ListOfItems() // L I S T  O F  I T E M S
{
    system("cls");
    ifstream inFile("lost_items.txt");

    if (!inFile.is_open()) {
        cout << "\n\n\t Error: Unable to open the file.\n";
        return;
    }

    struct LostItem {
        string itemID;
        string description;
        string location;
        string date;
        string contactInfo;
        string email;
        string studentNumber;
        string status;
        LostItem* next;
    };

    LostItem* head = nullptr;
    LostItem* tail = nullptr;
    string line;

    while (getline(inFile, line)) {
        stringstream ss(line);
        LostItem* newItem = new LostItem();

        getline(ss, newItem->itemID, '|');
        getline(ss, newItem->description, '|');
        getline(ss, newItem->location, '|');
        getline(ss, newItem->date, '|');
        getline(ss, newItem->contactInfo, '|');
        getline(ss, newItem->email, '|');
        getline(ss, newItem->studentNumber, '|');
        getline(ss, newItem->status, '|');
        newItem->next = nullptr;

        if (head == nullptr) {
            head = newItem;
            tail = newItem;
        } else {
            tail->next = newItem;
            tail = newItem;
        }
    }

    inFile.close();

    if (head == nullptr) {
        cout << "\n\n\t No items found.\n";
    } else {
        cout <<
"\n==========================================================================================
==========================================\n";
    cout << "   Item ID    Description      Location Lost      Date Lost      Contact Info
EmailUsed        StudentNumber         Status\n";
    cout <<
"============================================================================================
=========================================\n";

        LostItem* current = head;
        while (current != nullptr) {
            cout << setw(6) << current->itemID
                 << setw(14) << current->description
                 << setw(20) << current->location
                 << setw(20) << current->date
                 << setw(14) << current->contactInfo
                 << setw(13) << current->email
                 << setw(19) << current->studentNumber
                 << setw(19) << current->status << endl;
            current = current->next;
        }
    }

    while (head != nullptr) {
        LostItem* temp = head;
        head = head->next;
        delete temp;
    }

    cout << "\n\n\t Press Enter to go back to the main menu...";
    cin.ignore();
    cin.get();
}

void ViewClaimedItems() // V I E W   C L A I M E D  I T E M S
```

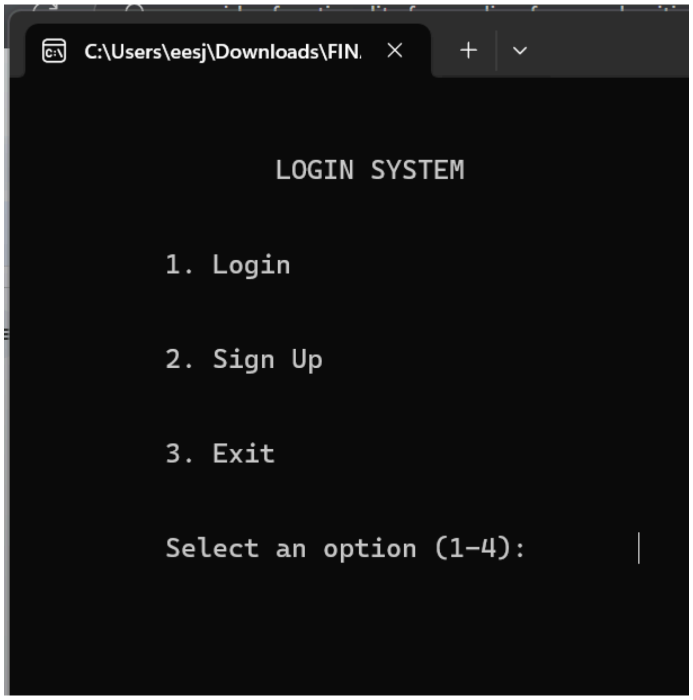
```
{
    system("cls");
    ifstream inFile("lost_items.txt");

    if (!inFile.is_open()) {
        cout << "\n\n\t Error: Unable to open the file.\n";
        return;
    }

    struct LostItem {
        string itemID;
        string description;
        string location;
        string date;
        string contactInfo;
        string email;
        string studentNumber;
        string status;
        LostItem* next;
    };

    LostItem* head = nullptr;
    LostItem* tail = nullptr;
    string line;

    while (getline(inFile, line)) {
        stringstream ss(line);
        LostItem* newItem = new LostItem();

        getline(ss, newItem->itemID, '|');
        getline(ss, newItem->description, '|');
        getline(ss, newItem->location, '|');
        getline(ss, newItem->date, '|');
        getline(ss, newItem->contactInfo, '|');
        getline(ss, newItem->email, '|');
        getline(ss, newItem->studentNumber, '|');
        getline(ss, newItem->status, '|');
        newItem->next = nullptr;

        if (newItem->status == "Claimed") {
            if (head == nullptr) {
                head = newItem;
                tail = newItem;
            } else {
                tail->next = newItem;
                tail = newItem;
            }
        } else {
            delete newItem;
        }
    }

    inFile.close();
    if (head == nullptr) {
        cout << "\n\n\t No claimed items founds.\n";
    } else {
        cout <<
"\n=========================================================================================
==========================================\n";
    cout << "   Item ID    Description      Location Lost      Date Lost      Contact Info
EmailUsed        StudentNumber         Status\n";
    cout <<
"===========================================================================================
=========================================\n";

        LostItem* current = head;
        while (current != nullptr) {
            cout << setw(6) << current->itemID
                 << setw(14) << current->description
                 << setw(20) << current->location
                 << setw(20) << current->date
                 << setw(14) << current->contactInfo
                 << setw(13) << current->email
                 << setw(19) << current->studentNumber
                 << setw(19) << current->status << endl;
            current = current->next;
        }
    }

    while (head != nullptr) {
        LostItem* temp = head;
        head = head->next;
        delete temp;
    }

    cout << "\n\n\t Press Enter to go back to the main menu...";
    cin.ignore();
    cin.get();
}

int main() // I N T  M A I N
{
    int MainChoice, loginChoice;

    Intro();

    do
    {
        ShowLoginMenu();
        cin >> loginChoice;

        switch (loginChoice)
        {
        case 1:
            Login();
            break;
        case 2:
            SignUp();
            continue;
        case 3:
            cout << "\n\n\n\t Exiting the program. Thank you!\n";
            return 0;
        default:
            cout << "\n\n\n\t Invalid choice. Please select between 1-4.\n";
            cin.get();
        }
    } while (loginChoice != 4 && !loggedIn);

    if (loggedIn)
    {
        do
        {
            ShowMainMenu();
            cin >> MainChoice;

            switch (MainChoice)
            {
            case 1:
                SubmitLostItem(lostItemsHead, email, studentNumber);
                break;
            case 2:
                SearchForItems();
                break;
            case 3:
             ListOfItems();
             break;
            case 4:
                ViewClaimedItems();
                break;
            case 5:
                cout << "\n\n\n\t Exiting the program. GG!" << endl;
                break;
            default:
                cout << "\n\n\n\t Invalid choice. Please enter a number between 1 and 5." <<
endl;
            }
        } while (MainChoice != 5);
    }

    return 0;
}
```

Console Output:



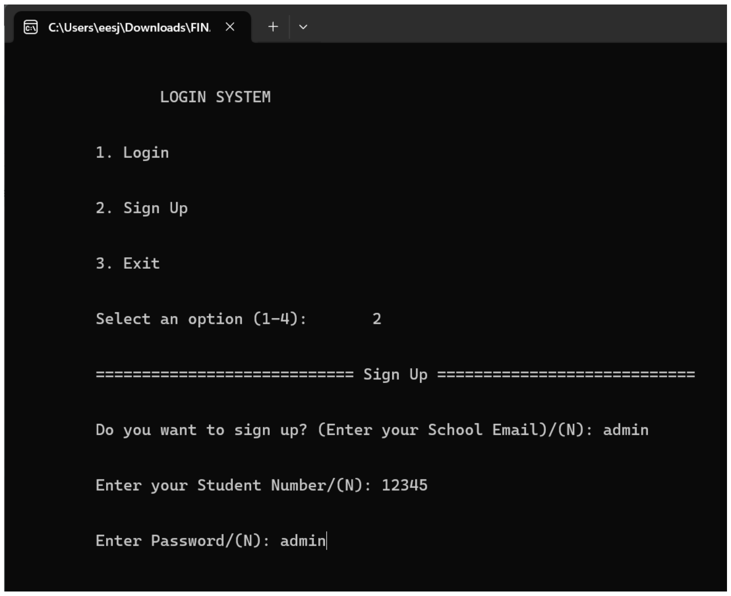
```
C:\Users\eesj\Downloads\FIN.

                MANAGEMENT SYSTEM

                  LOST AND FOUND

                   DSA PROJECT


        MADE BY : GROUP 6 BLACKBOX
        SCHOOL : TECHNOLOGICAL INSTITUTE
                     OF THE
                   PHILIPPINES



         Press Enter to continue...
```

Figure 6.1: *User's Window*



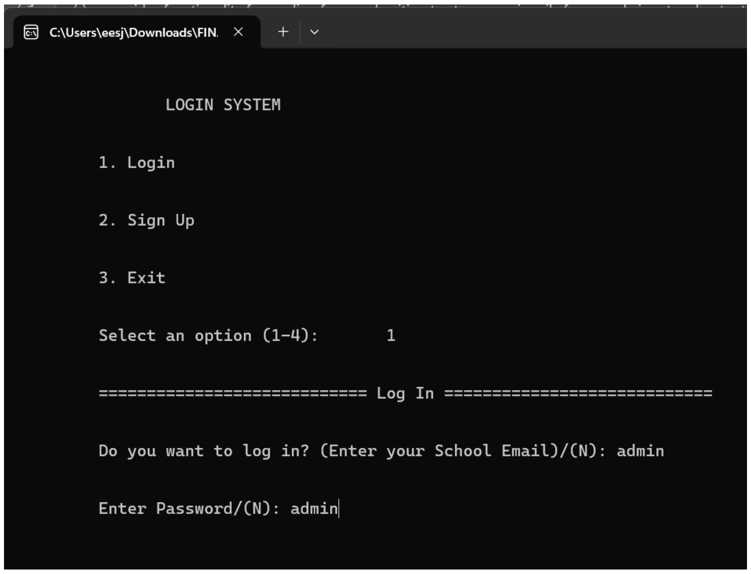
```
C:\Users\eesj\Downloads\FIN.

                LOGIN SYSTEM

          1. Login

          2. Sign Up

          3. Exit

          Select an option (1-4):
```

Figure 6.2: *Login System*



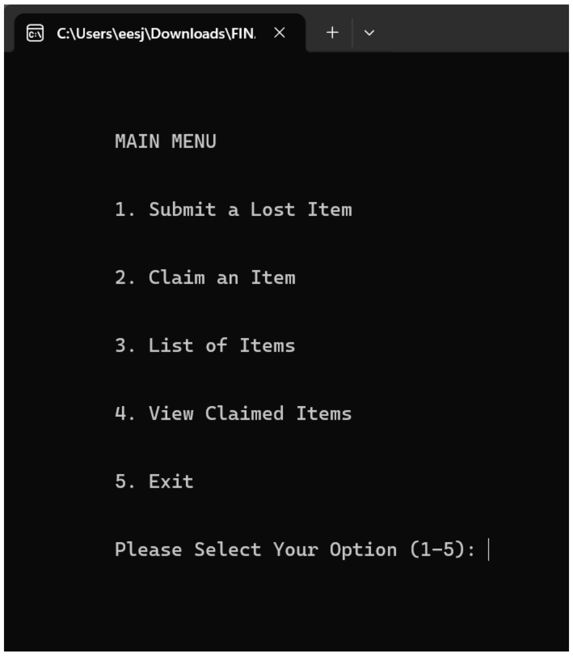
```
C:\Users\eesj\Downloads\FIN.

        LOGIN SYSTEM

    1. Login

    2. Sign Up

    3. Exit

    Select an option (1-4):       2

    =========================== Sign Up ===========================

    Do you want to sign up? (Enter your School Email)/(N): admin

    Enter your Student Number/(N): 12345

    Enter Password/(N): admin
```

Figure 6.3: *User Signup*



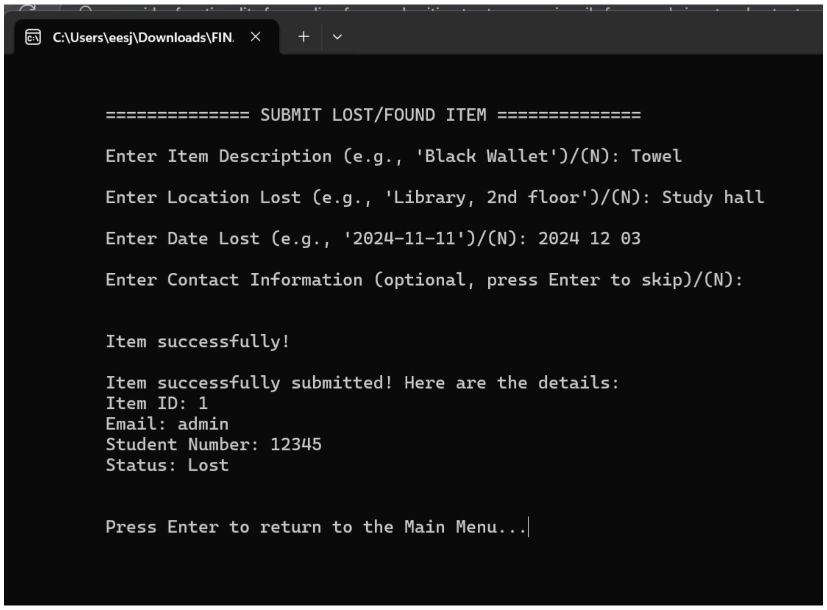
```
C:\Users\eesj\Downloads\FIN.

        LOGIN SYSTEM

    1. Login

    2. Sign Up

    3. Exit

    Select an option (1-4):       1

    =========================== Log In ============================

    Do you want to log in? (Enter your School Email)/(N): admin

    Enter Password/(N): admin
```

Figure 6.4: *User Login*



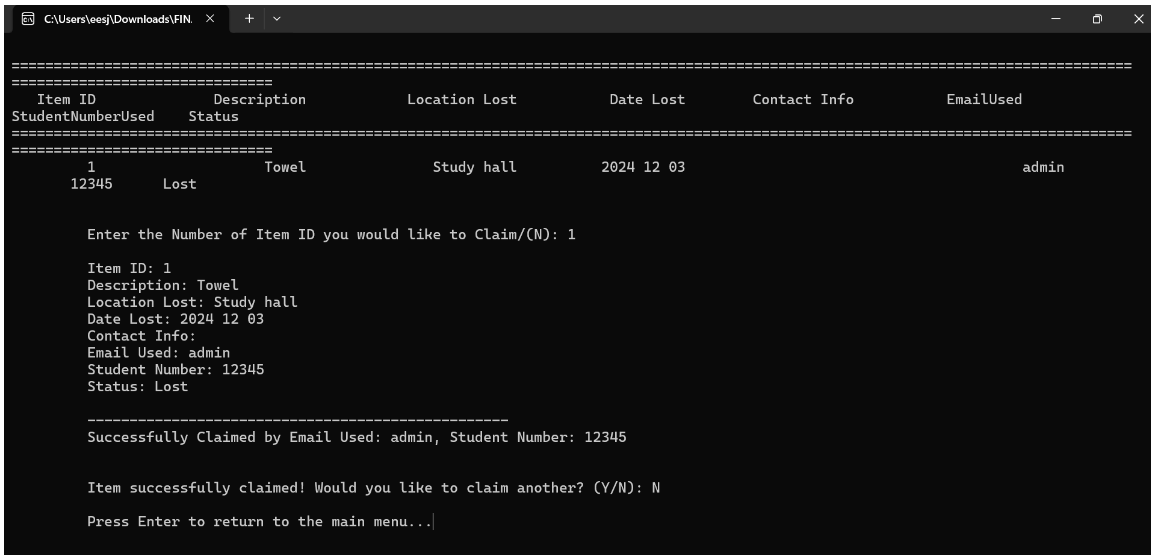
```
MAIN MENU

1. Submit a Lost Item

2. Claim an Item

3. List of Items

4. View Claimed Items

5. Exit

Please Select Your Option (1-5):
```

Figure 6.5*: The program will redirect the User to the Dashboard Main Menu*



```
============== SUBMIT LOST/FOUND ITEM ==============

Enter Item Description (e.g., 'Black Wallet')/(N): Towel

Enter Location Lost (e.g., 'Library, 2nd floor')/(N): Study hall

Enter Date Lost (e.g., '2024-11-11')/(N): 2024 12 03

Enter Contact Information (optional, press Enter to skip)/(N):


Item successfully!

Item successfully submitted! Here are the details:
Item ID: 1
Email: admin
Student Number: 12345
Status: Lost


Press Enter to return to the Main Menu...
```

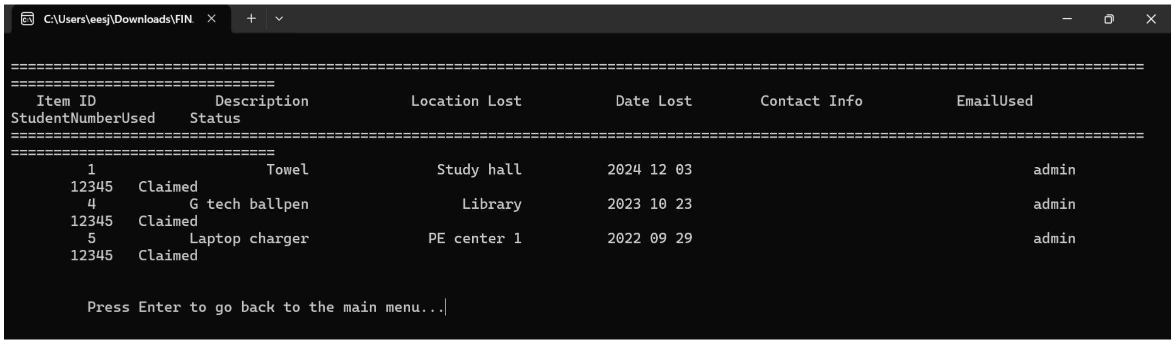
Figure 6.6: *User pressed Option 1(Submit a Lost Item), and filled up the fields*



```
==================================================================================================================================
==============================
   Item ID             Description            Location Lost           Date Lost        Contact Info           EmailUsed
StudentNumberUsed    Status
==================================================================================================================================
==============================
         1                    Towel               Study hall          2024 12 03                                       admin
       12345      Lost


         Enter the Number of Item ID you would like to Claim/(N): 1

         Item ID: 1
         Description: Towel
         Location Lost: Study hall
         Date Lost: 2024 12 03
         Contact Info:
         Email Used: admin
         Student Number: 12345
         Status: Lost

         --------------------------------------------------
         Successfully Claimed by Email Used: admin, Student Number: 12345


         Item successfully claimed! Would you like to claim another? (Y/N): N

         Press Enter to return to the main menu...
```

Figure 6.7: *User pressed Option 2(Claim an Item),*

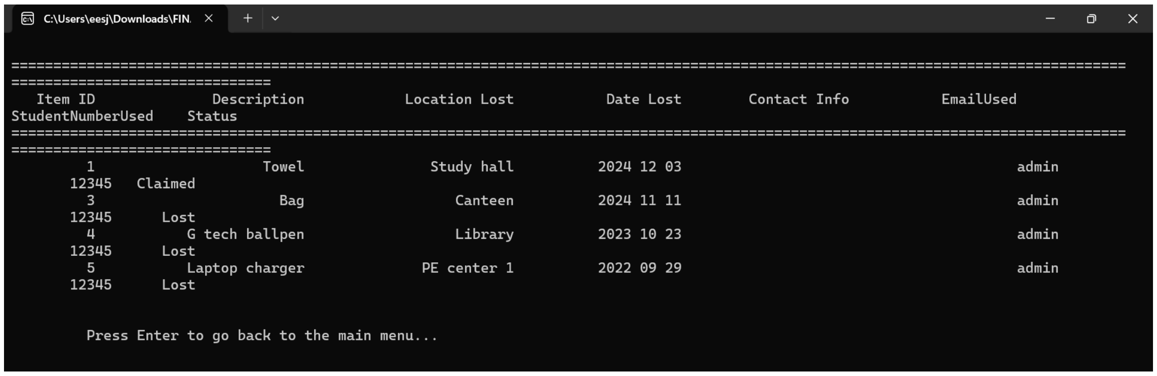
the program will ask the user to enter the Item ID number



```
==================================================================================================================================
==============================
   Item ID             Description            Location Lost           Date Lost        Contact Info           EmailUsed
StudentNumberUsed    Status
==================================================================================================================================
==============================
         1                    Towel               Study hall          2024 12 03                                       admin
       12345   Claimed
         4           G tech ballpen                  Library          2023 10 23                                       admin
       12345   Claimed
         5           Laptop charger             PE center 1          2022 09 29                                       admin
       12345   Claimed


         Press Enter to go back to the main menu...
```

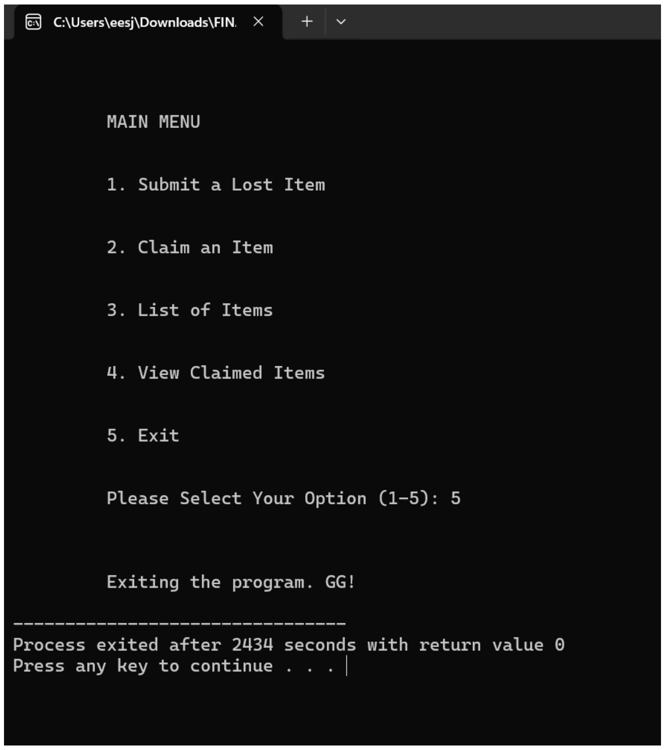
Figure 6.8: *User pressed Option 3(List of Item),*

then the user will be able to see the list of item either its lost or claimed



```
=====================================================================================================================================
==============================
   Item ID             Description            Location Lost           Date Lost        Contact Info           EmailUsed
StudentNumberUsed    Status
=====================================================================================================================================
==============================
         1                   Towel               Study hall          2024 12 03                                        admin
       12345   Claimed
         3                     Bag                  Canteen          2024 11 11                                        admin
       12345      Lost
         4          G tech ballpen                  Library          2023 10 23                                        admin
       12345      Lost
         5          Laptop charger              PE center 1          2022 09 29                                        admin
       12345      Lost


         Press Enter to go back to the main menu...
```

Figure 6.9: *User pressed Option 4(View Claimed items), then the user will be able to see claimed items*



```


          MAIN MENU


          1. Submit a Lost Item


          2. Claim an Item


          3. List of Items


          4. View Claimed Items


          5. Exit


          Please Select Your Option (1-5): 5



          Exiting the program. GG!

--------------------------------
Process exited after 2434 seconds with return value 0
Press any key to continue . . .
```

Figure 7: *User pressed Option 5 (Exit), The program will terminate*

7. Conclusion and Recommendations

I. Conclusion

The Lost and Found Management System implemented using linked lists in C++ provides an efficient and user-friendly approach to managing lost and found items, specifically tailored to the needs of the Technological Institute of the Philippines (TIP). By leveraging linked lists for data management, the system ensures seamless operations such as submitting lost items, searching for found items, and claiming belongings. This initiative successfully addresses the challenges of manual item tracking, offering convenience for students, instructors, and administrators.

The system's design emphasizes simplicity and effectiveness, with clear functionalities accessible through a terminal interface. It fosters better organization within the campus, reducing the hassle associated with lost items while maintaining records securely through file handling techniques. Furthermore, the study underscores the importance of integrating data structures like linked lists into practical applications, enhancing the understanding and application of core programming concepts.

II. Recommendations

- **Automated Notifications:**
  - Introduce an automated notification system to alert users about updates regarding their lost or found items.
- **Search Optimization:**
  - Incorporate filters like date lost, location, and item category to narrow down search results.
  - Use algorithms that account for typos or partial matches when searching for items.
- **Feedback Mechanism**
  - Integrate a feature for users to provide feedback on the system, suggesting improvements and reporting issues.
- **Automated Affidavit Submission**: Integrate a feature where users can fill out and submit an Affidavit of Loss through the system to expedite the claiming process. This would require users to submit an online form that automatically generates a standard Affidavit of Loss with fields for item details and personal information.
- **Administrator System:**
  - Admins can review submitted lost items and approve or reject them based on criteria

such as item description or proof of ownership.

- Admins can verify claimed items against user-submitted information (e.g., the Affidavit of Loss) to ensure the rightful claimant is receiving their item.

8. References

- Lost and Found Management System Project Proposal | PDF. (n.d.). Scribd. https://www.scribd.com/document/665938482/lost-and-found-management-system-project-proposal
- The C++ Standard Template Library (STL). (2015, December 7). GeeksforGeeks. https://www.geeksforgeeks.org/the-c-standard-template-library-stl/
- Linked List in C++. (2024, June 10). GeeksforGeeks. https://www.geeksforgeeks.org/cpp-linked-list/
- Linked List Data Structure. (2024, May 22). GeeksforGeeks. https://www.geeksforgeeks.org/linked-list-data-structure/
- Linked List Data Structure in C++ With Illustration - javatpoint. (n.d.). Www.javatpoint.com. https://www.javatpoint.com/linked-list-data-structure-in-cpp-with-illustration