Icon

Description automatically generated

**(SCHOOL OF COMPUTING AND INFORMATICS)**

**(BACHELOR OF COMPUTER SCIENCES)**

**SEMESTER 1 SESSION 2021/2022**

**CCC 1123 PROBLEM SOLVING AND PROGRAMMING FUNDAMENTAL :**

**Group Name:**

“Programmers Legacy”

**Name of Group Members:**

1. **SaifAlmajd M. H. Almassri** (AIU21102071)

2. **Adib Rahman Rahmatyar** (AIU21203035)

3. **Lassana S Donzo** (AIU21102026)

4. **Mudassiru Nasiru** (-)

5. **Abdul-Wahid Haji Kali** (-)

6. **Aminu Abubakar** (-)

**Lecturer Name:**

Ros Syammimi binti Hamid

**Date of submission:** 18 January 2022

**1.0 Introduction:**

Every institution creates a system that keeps a record of the students' information in and off-campus for a smooth interaction between the students and the record regarding their studies. For this reason, we have decided to create a system that is going to do the same job as stated above. Our project is a system that stores, writes, tracks, updates, and edits data for AIU students that is more efficient for them to interact with the administration and their data systems, and how it works is explained in detail in the manual:

**1.0.1 AIU student information System**, including course and ID, is stored. Any Admission staff can use AIU SIS (AIU Student Information System) to access and modify or to view information about the student.

**1.0.2 AIU-SIS's Most Valuable Advantages**

**Increases Productivity** – AIU-SIS can help AIU Admission become more productive. This increase is due to a reduction in the amount of time spent maintaining records as well as an increase in data accuracy and organization. With less time on their hands, administration and staff can focus on university productivity instead.

**Track**- Keeping track of all of a student's data for educators so that they can the students and their grades and the CGPA of each one of them using the student management software.

**Safe-** Making a locally stored program made it safer for data not to be leaked or hacked, moreover we also did a password of (2071) -that can be changed also- and it used to keep the program safer

**All Staff Parties Involved Have Access to It Online –** As long as you are a staff and you can access the data, you can get the password and run it on any online or offline compiler and it works

**Transparency Increases –** Transparency between Admission, staff, instructors, and pupils is enabled by the program, there will be no one missing on the system from the students and their grades will be shown and displayed in an organized table to keep track by periodically checking.

**2.0 Objective:**

**Our main objective is to have a locally stored data program by implementing the shortest and most effective algorithm.**

Luckily, **we were able to archive this goal almost 200 lines of coding** (Comments Excluded) we made the program that matches our need for AIU System

There is always more objectives, that we also aimed for and made such as:

* **We must have:**

0 Errors ✅

The least amount of lines to write✅

A function for each operation.✅

Clear names for variables and functions.✅

Accurate declarations.✅

Comments ✅

Error Handling message✅

Successfully message confirmations✅

Efficient use of words and understandable to the reader.✅

* **We can’t have:**

Spelling mistakes.✅

bad program functionality.✅

Getting unclear input questions.✅

Unfinished code writings.✅

* **Advanced Future Goals:**

Improved Management of Student Enrollment.✅

Centralized Billing and Accounting Management.

Easier Integration of Administrators, Teachers, and Parents.

Easy Exam Scheduling.

Efficient Management of Student-Related Activities.

* Future AIU SIS Objective

Dashboards that can be used right away

Data gathering and administration may be made easier.

Real-time reporting is simple.

Customized student portfolios with one-stop access to student information, performance, and health records.

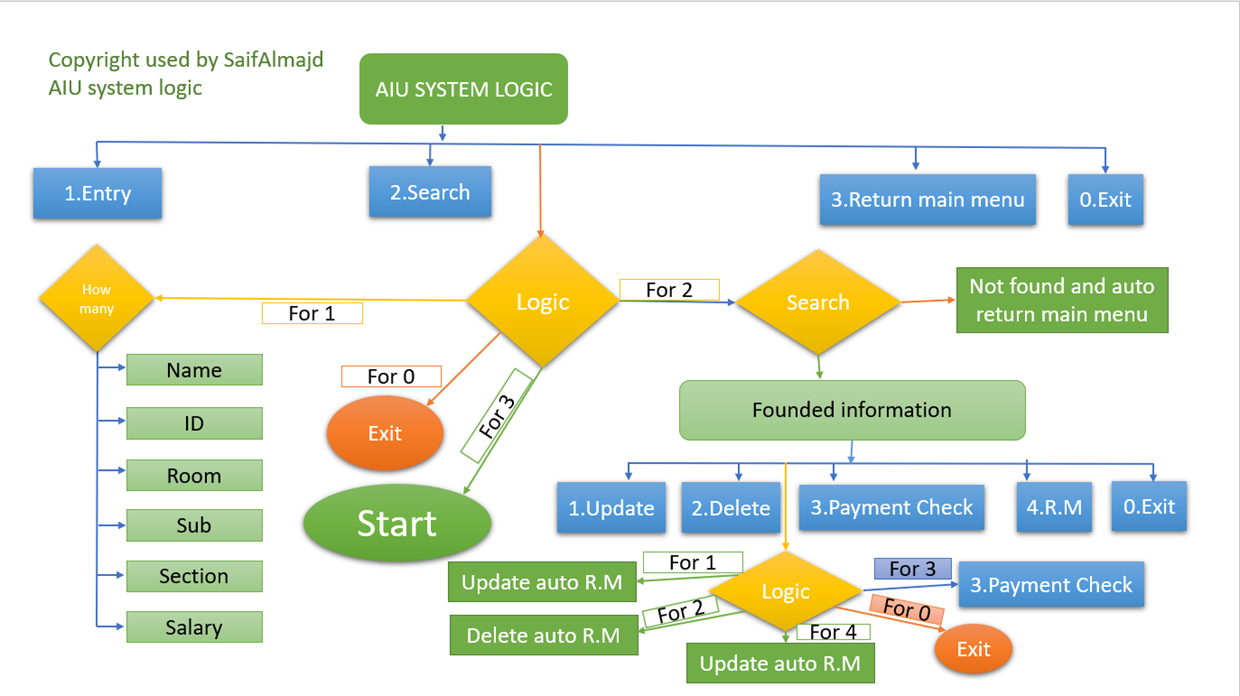
Fee payments are built-in using a platform that links students and parents directly and allows for online payments.

Allowing AIU to keep track of all of your students and to extend beyond the four walls of the classroom. Other activities must be accounted for, maintained, and structured, such as sports, clubs, and other extracurricular activities. The AIU- SIS will hopefully feature that and ensure this—as well as the integrity of each student's record.

Generally:  
This will help in reducing the boring tasks on the human Labor and workload, the cost of employing staff to manage students and school activities is an extra cost. Staff workload is reduced by using this system, allowing teachers to devote more time to the AIU-SIS. As a result, they can reduce teacher workload and save time that could be spent in the classroom to use these human minds and time in something more beneficial.

This will also save natural resources and AIU university will no longer have to rely on paper files and records after implementing a cloud-based solution. Routine administrative tasks can be completed and recorded without the use of paper with a modern AIU-SIS system. This results in the conservation of natural resources and the preservation of a digital audit trail of the data. Supporting materials may also be digitized and uploaded to the student's record as a digital document.   
Lastly, it will be accessible from any location at any time. Because of its accessibility, it is possible to keep a record of everything. It also makes it easier to provide real-time information to all parties. All they need is a smartphone, tablet, or desktop computer to get started.

**AIU System Implemtiations and Plan shortly:**

****

More is in the Project PowerPoint file (Please View).

**3.0: Code: (Can be copied and pasted)**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

AIU Students Admission System

Staff Only are allowed to use this program for registration purposes

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// We First Include All the packages that we need to use differnet data types, variables, constants, subprograms, cursors, and exceptions etc..

#include <stdlib.h>

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <string.h>

// Structure user-built to store student name, ID, CGPA, and course

struct Student

{

char student\_id[100];

char firstname[100];

char lastname[100];

char course[100];

char CGPA[100];

struct Student \*next;

} \* head;

void insert(char \*firstname, char \*lastname, char \*student\_id, char \*course, char \*CGPA) // The function that saves students data to the structure

{

struct Student \*student = (struct Student \*)malloc(sizeof(struct Student)); // making a call to the structure

strcpy(student->student\_id, student\_id); // stores data

strcpy(student->firstname, firstname); // stores data

strcpy(student->lastname, lastname); // stores data

strcpy(student->course, course); // stores data

strcpy(student->CGPA, CGPA); // stores data

student->next = NULL; // stores data

if (head == NULL)

{

// if head is NULL

// set student as the new head

head = student;

}

else

{

// if list is not empty

// insert student in beginning of head

student->next = head;

head = student;

}

printf("Student was Successfully Added\n\n"); // Confirmation

}

void search(char \*student\_id)

{

struct Student \*temp = head; // calls student data

while (temp != NULL)

{

if ((strcmp(temp->student\_id, student\_id) == 0 ))

{

printf("First Name of Student: %s\n", temp->firstname); // printing out data

printf("Last Name of Student: %s\n", temp->lastname); // printing out data

printf("Student ID of Student: %s\n", temp->student\_id); // printing out data

printf("course of Student: %s\n", temp->course); // printing out data

printf("CGPA of Student: %s\n", temp->CGPA); // printing out data

return;

}

temp = temp->next;

}

printf("Student with Student ID %s is not found !!!\n", student\_id); // Error Handling

}

void check\_id(char \*student\_id)

{

struct Student \*temp = head; // calls student data

while (temp != NULL)

{

if ((strcmp(temp->student\_id, student\_id) == 0 ))

{

printf("Record with Student ID %s Already exist !!!\n", student\_id); // Checks if student ID already exist

printf("\n\nThank You for using AIU Student System\n");

exit(0); // Exits program

return;

}

temp = temp->next;

}

printf("Student Got a Unique Student ID (%s) \n\n", student\_id); // Error Handling

}

void update(char \*student\_id)

{

struct Student \*temp = head; // calls student data

while (temp != NULL)

{

if ((strcmp(temp->student\_id, student\_id) == 0 ))

{

printf("Record with Student ID %s Found !!!\n", student\_id);

printf("Enter new First name: ");

scanf("%s", temp->firstname); // Getting Student New Data

printf("Enter new Last name: ");

scanf("%s", temp->lastname); // Getting Student New Data

printf("Enter new Course Name: ");

scanf("%s", temp->course); // Getting Student New Data

printf("Enter new CGPA: ");

scanf("%s", temp->CGPA); // Getting Student New Data

printf("Student Data was Updated Successfully!\n"); // Confirmation

return;

}

temp = temp->next;

}

printf("Student with Student ID %s is not found !!!\n", student\_id); // Error Handling

}

void Delete(char \*student\_id)

{

struct Student \*temp1 = head;

struct Student \*temp2 = head;

while (temp1 != NULL)

{

if ((strcmp(temp1->student\_id, student\_id) == 0 ))

{

printf("Record with Student ID %s Found !!!\n", student\_id);

if (temp1 == temp2)

{ // This condition will run if the record that we need to delete is the first node of the linked list

head = head->next;

free(temp1);

}

else

{

// temp1 is the node we need to delete

// While temp2 is the node previous to temp1

temp2->next = temp1->next;

free(temp1);

}

printf("Record Successfully Deleted !!!\n"); // Confirmation

return;

}

temp2 = temp1;

temp1 = temp1->next;

}

printf("Student with Student ID %s is not found !!!\n", student\_id); // Error Handling

}

void display()

{

struct Student \*temp = head; // calls student data from structure

printf("\n\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 AIU Students List \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n\n"); // Starting Table

printf("========================================================================================================================\n");

printf("First Name \t\t Last Name\t\t Student ID \t\t Course \t\t CGPA \n"); // Table Titles

printf("========================================================================================================================\n\n");

while (temp != NULL)

{ // to print the next student if data exists

printf("%s \t\t\t %s \t\t\t %s \t\t\t %s \t\t %s \n\n", temp->firstname, temp->lastname, temp->student\_id, temp->course, temp->CGPA); // Print from structure to compiler

temp = temp->next;

}

printf("\t\t\t\t\t \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n\n"); // Ending Table

}

int start\_system(){

head = NULL; // Declaration

int choice; // Declaration

char firstname[100]; // Declaration

char lastname[100]; // Declaration

char course[100]; // Declaration

char student\_id[100]; // Declaration

char cgpa[100]; // Declaration

struct Student \*temp = head; // calls student data

printf("\n\n\t\t\t \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*");

printf("\n\n\t\t\t \* \*");

printf("\n\n\t\t\t \* Welcome AIU Staff \*");

printf("\n\n\t\t\t \* \*");

printf("\n\n\t\t\t \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*\n\n");

do

{

printf("Choose one of the following Options\n");

printf("\t\t1 - Add a New Student Details\n");

printf("\t\t2 - Find a Student Details by Student ID \n");

printf("\t\t3 - Delete a Student by Student ID \n");

printf("\t\t4 - Update a Student by Student ID \n");

printf("\t\t5 - Display and List AIU Stored Students \n");

printf("\t\t6 - To Exit\n");

// The above are all options

printf("\tEnter the operation of your choice:\n");

scanf("%d", &choice);

switch (choice)

{

case 1:

printf("\tEnter the Student's First Name: \n");

scanf("%s", firstname);

printf("\tEnter the Student's Last Name: \n");

scanf("%s", lastname);

printf("\tEnter the Student's ID : \n");

scanf("%s", student\_id);

check\_id(student\_id);

printf("\tEnter the Student's Enrolled Course Name: \n");

scanf("%s", course);

printf("\tEnter the Student's CGPA: \n");

scanf("%s", cgpa);

insert(firstname, lastname, student\_id, course, cgpa);

break;

case 2:

printf("Enter Student ID to search: ");

scanf("%s", student\_id);

search(student\_id);

break;

case 3:

printf("Enter Student ID to delete: ");

scanf("%s", student\_id);

Delete(student\_id);

break;

case 4:

printf("Enter Student ID to update: ");

scanf("%s", student\_id);

update(student\_id);

break;

case 5:

display();

break;

case 6:

printf("\nThank You for using AIU Student System\n");

exit(0); // Exits program

break;

default:

printf("\nYou Have Entered a Wrong Operation \nTry Again or Type 8 to Exit\n\n\n");

exit(0);

break;

}

} while (choice != 0);

return (0);

}

int main()

{

char username[100]; // Declaration

char password[100]; // Declaration

printf("\n\n\n\n\n\n\n\n\t\t\t\t\t|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\n\t\t\t\t\t| |\n\t\t\t\t\t| AIU Student System |\n\t\t\t\t\t|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\n\t\t\t\t\t|\t\t |\n\n\n");

printf("\n\n\t\t\t\t\t This is Confidential Data \n\t\t\t You are only allowed to enter Confidential's three times \n\n");

for ( int attempts\_times = 1; attempts\_times <= 3; attempts\_times++) { //Attempt is 1 be default as its the first time

printf("What is The Addamission Username: \n\n");

scanf("%s", username); // Getting Username

printf("What is The Addamission Staff Password \n\n");

scanf("%s", password); // Getting Password

if (((strcmp(username, "M.Ros") == 0) || (strcmp(username, "Admin") == 0 ) ) && (strcmp(password, "ADMIN2071") == 0 )) {

start\_system();

} else

{

printf("You have entered the wrong Confidential's for the %d attempt" , attempts\_times);

if (attempts\_times <= 3){

printf("\n\n\nTry again \n\n\n");

}

else{

printf("\n\n\nSorry, Access Denied!!");

exit(0);

}

}}

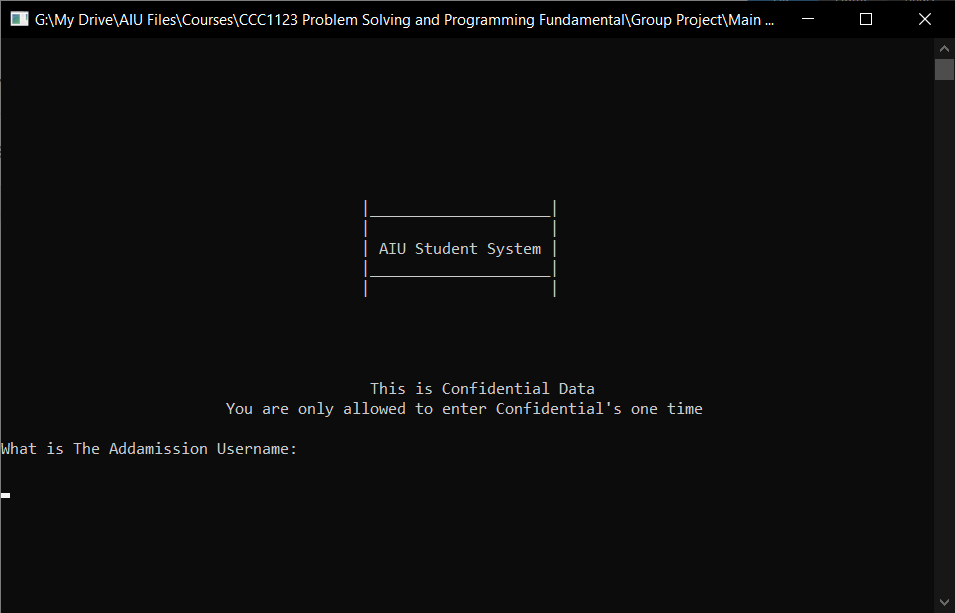
}

**4.0: Manual:**

In this section, we are going to talk about the run process of the code, what is the outcome of the code if it’s handled perfectly and what is the outcome of the code if it’s handled badly. For a code to be a successful code, it needs to be free from all errors and the objective of the code should be met. This code is going to tell us it results if it’s run perfectly as well as it results if there is an error handling.

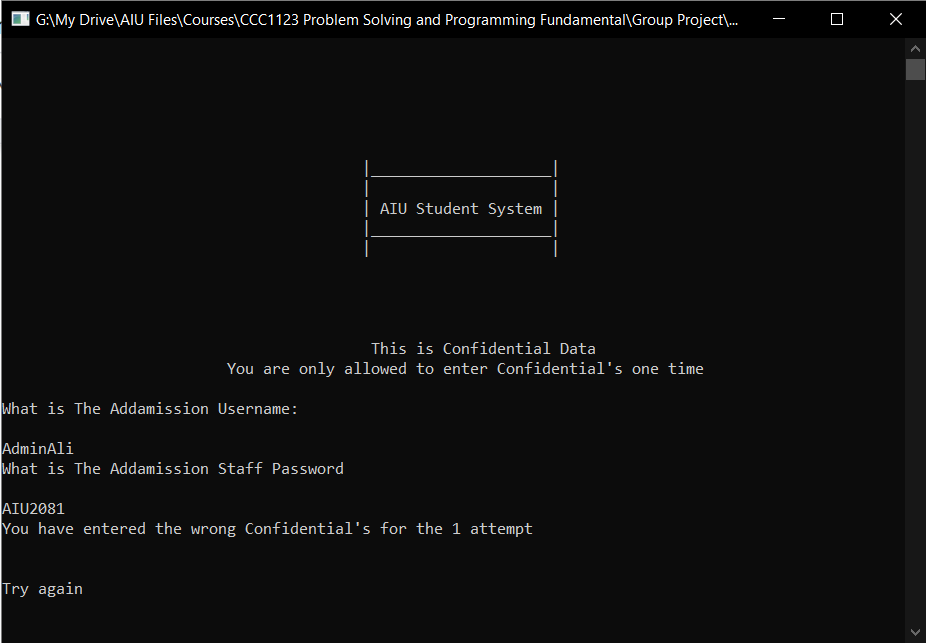
**How to use:**

When the code is running, it’s firstly going to ask for the username and the password. If these things are entered correctly, the code is going to run successfully and display the main menu (Username is either M.Ros or Admin)

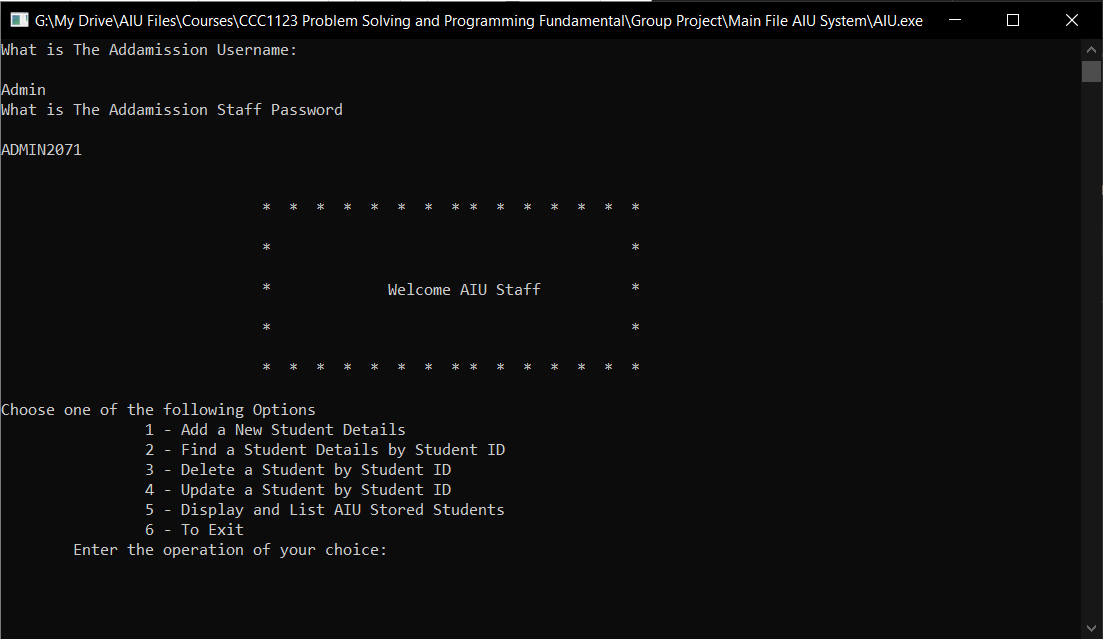
****

and then it will ask you for the password (the Password is ADMIN2071)

If the confidential are wrong it is going to display try again which tells the user that the code that was entered wasn't correct, and it will give the user a total of 3 times to attempt, and then it will kick them out



Otherwise, if entered username and password are correct, then it will display the main menu



**From the main menu, Staff will be given 6 options:-**

**The first option**, we ask the user (AIU register staff in this case) to choose an operation to start from the few options presented, after choosing an option (let’s say they chose to add a new student), for this we have a few steps to follow, we start inputting the student’s details like the following:

**· First Name**

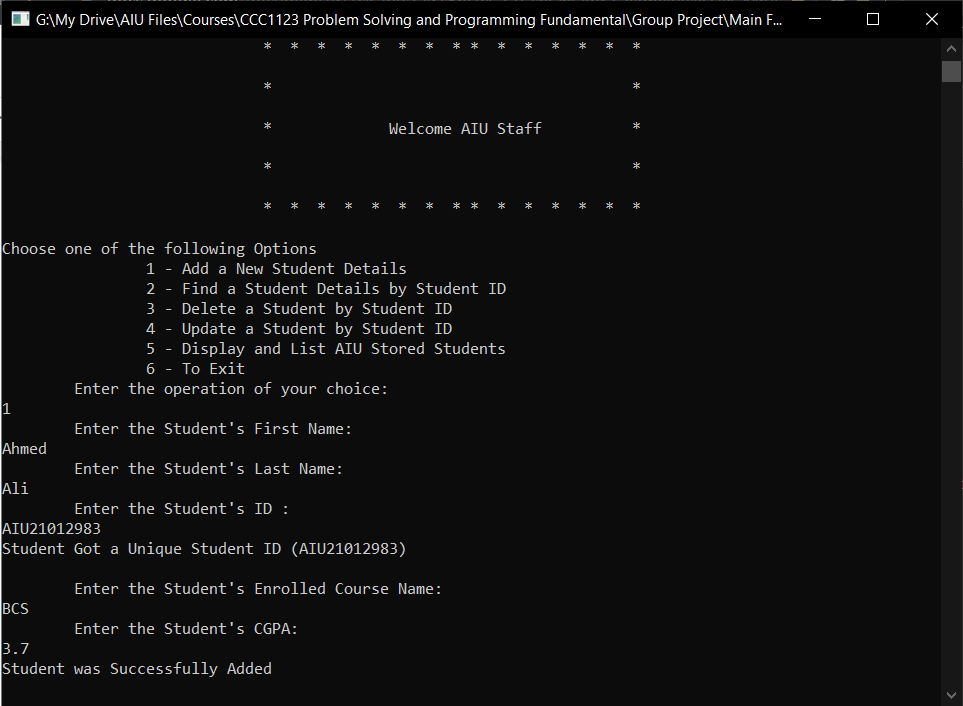
**· Last Name**

**· ID number**

**· Course**

**· CGPA**

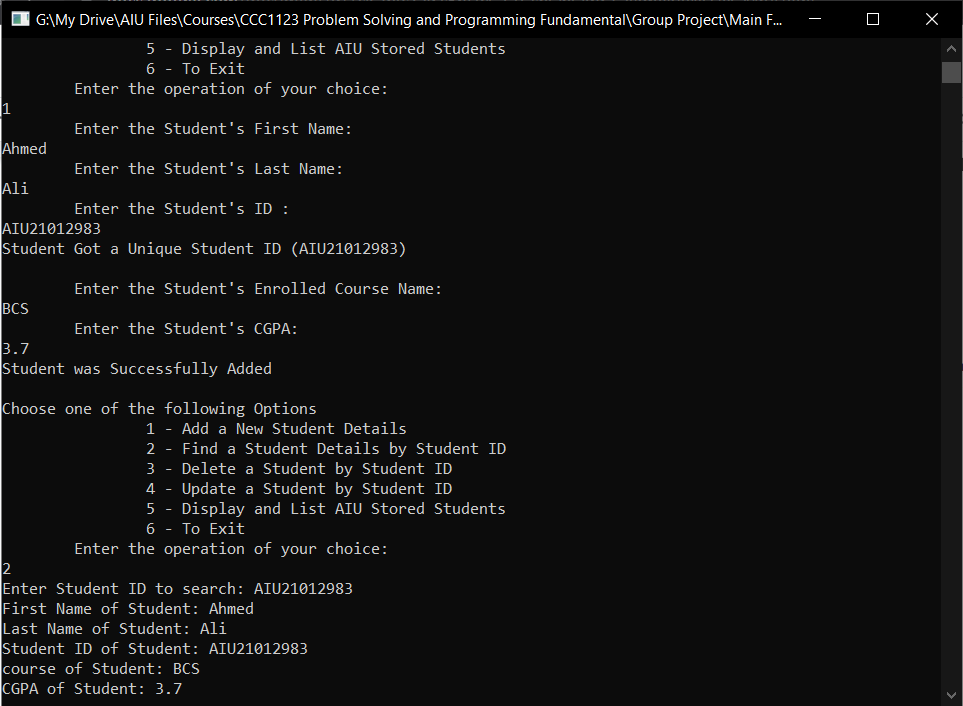
It is used to add a student and it will ask the user for the student information when requested as follows



After confirmation, we print out a message to confirm that the student was registered. And the operation goes almost the same for other options.

**Second option:**

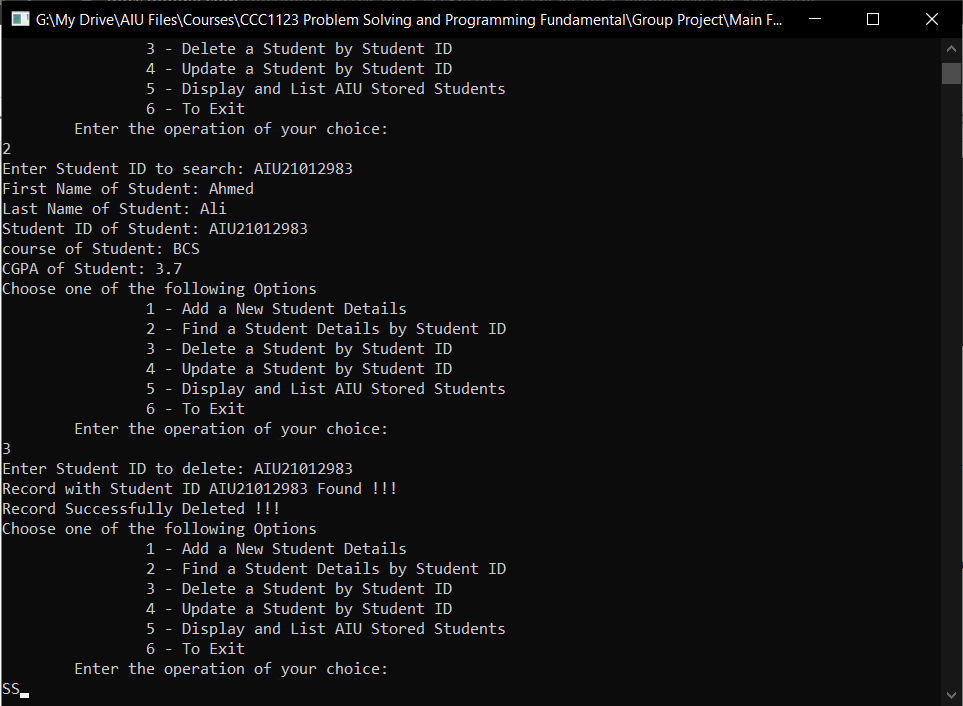
It is used to look for a student using their student ID and it will search and display the student’s information as the following

****

After confirmation, we print out a message to confirm that the student was found. And the operation goes almost the same for other options.

**Third option:**

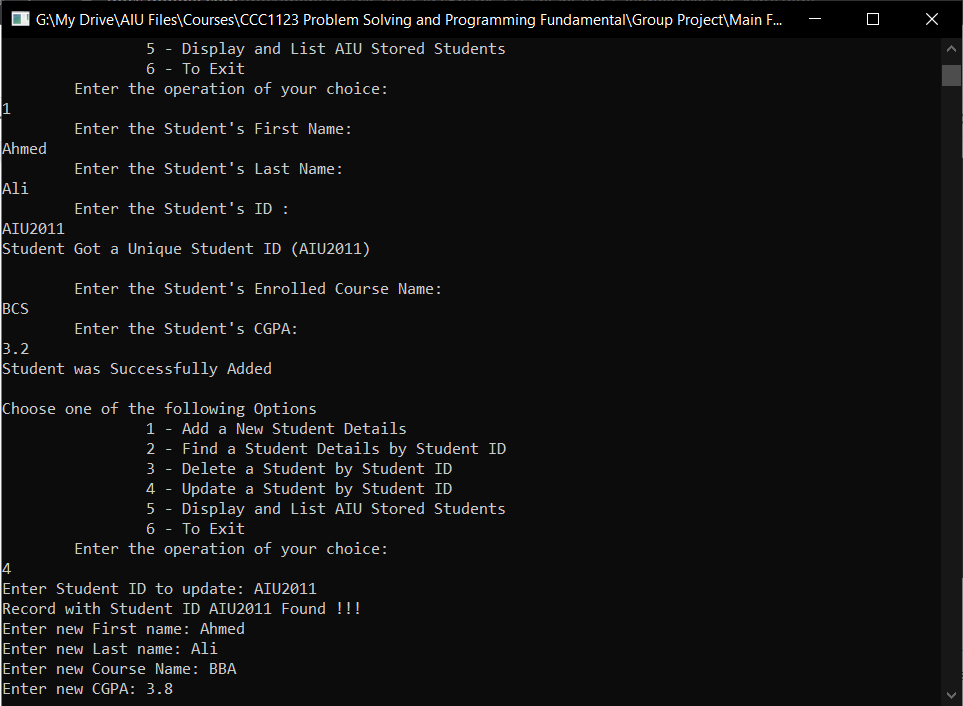
It is used to delete a student from the system and it will give a confirmation message for both finding the student and deleting him as well



After confirmation, we print out a message to confirm that the student was deleted. And the operation goes almost the same for other options.

**Fourth option:**

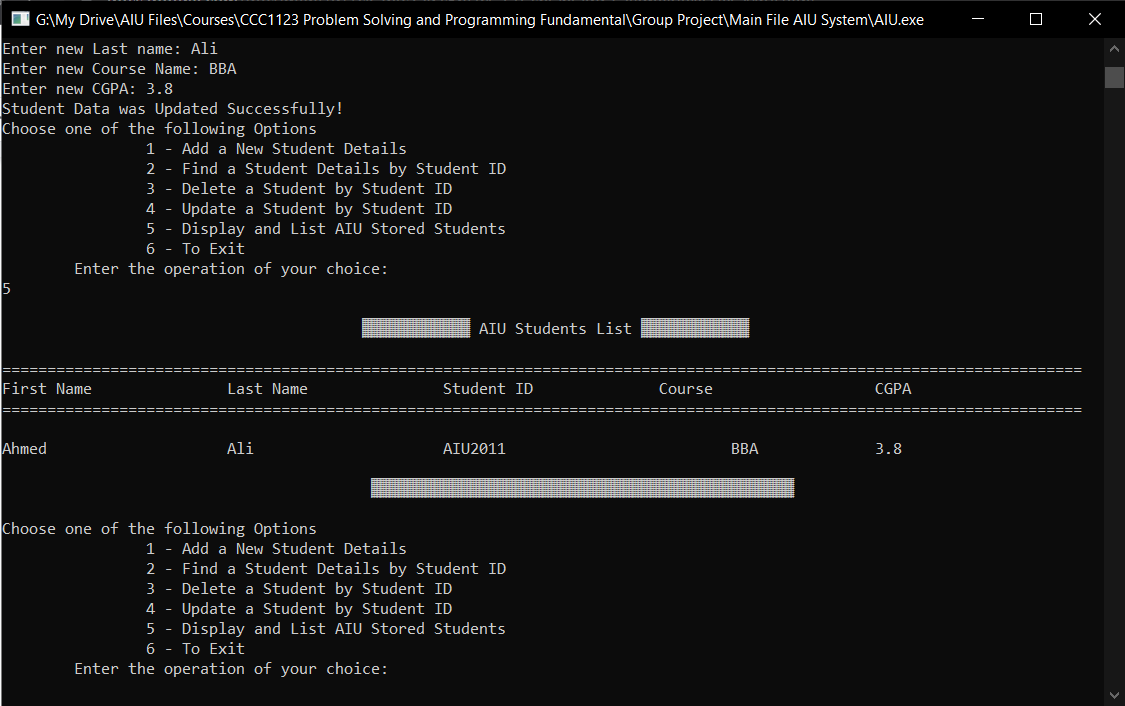
It is used to update the information of an existing student in the system and to do it calls the student saved data and rewrites the updated information as follows



After confirmation, we print out a message to confirm that the student was updated. And the operation goes almost the same for other options.

**Fifth option:**

Is to display and show all the students data that has been stored in the system in a very organized table;



**The sixth and the last option:**

Is used to exit the system after using it and it does nothing but exit the app.  
Note: Use only when you are done using the system as it will not save anything.