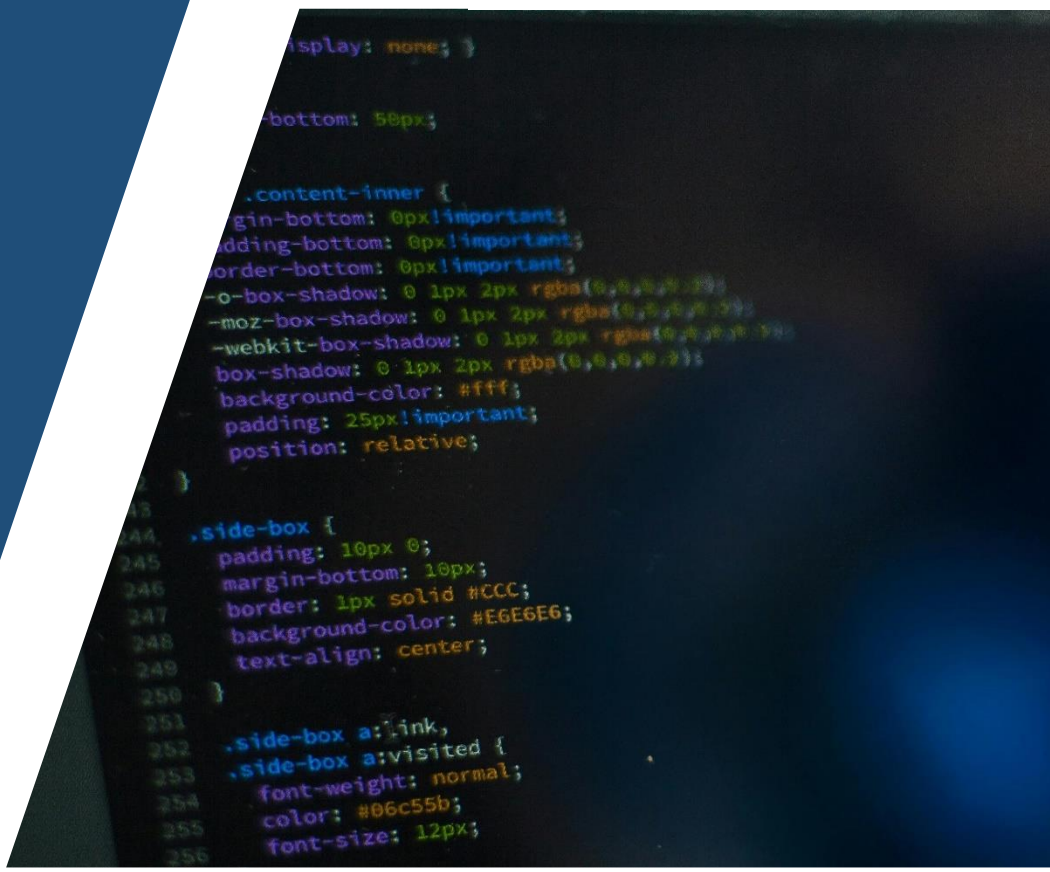
OOP PROJECT



FINAL PROJECT

AMNA TAHIR BUTT (23009105033)

ABEEHA ZIA (23009105007)

BS-IT BATCH 09 SECTION A

“UNIVERSITY COURSE MANAGEMENT SYSTEM”

THEORY PROF: ***PROF. DR. MUMTAZ ALI SHAH***

LAB INSTRUCTOR: ***MISS JAVERIA JALIL***

PROJECT

"UNIVERSITY COURSE MANAGEMENT SYSTEM"

OVERVIEW:

This document describes a university course management system developed in C++. The system is designed to manage courses, students, and instructors, facilitating operations like adding, searching, deleting records, and assigning courses. The system ensures data persistence using file INPUT/OUTPUT operations and includes a simple login mechanism for access control. This university course management system is a practical application of object-oriented programming in C++. It showcases how to manage complex data using classes and file I/O operations, ensuring data persistence and providing a user-friendly interface for managing university-related data.

KEY FEATURES:

Class Descriptions

- **University Class:**

Abstract Base Class: This class defines a standard interface for saving and loading data. It contains two pure virtual functions:

```
virtual void saveToFile(ofstream&) = 0;
virtual void loadFromFile(ifstream&) = 0;
```

Derived Classes: `Course`, `Instructor`, and `Student` inherit from `University` and implement these virtual functions to handle specific data operations.

- **Course Class**

Attributes:

```
char courseName[100];
char courseCode[10];
int credits;
char instructor[100];
```

Methods:

`void saveToFile(ofstream& out) override`: Saves course details to a file.
`void loadFromFile(ifstream& in) override`: Loads course details from a file

- **Instructor Class**

Attributes:

```
char name[100];
char id[10];
char assignedCourses[max][10];
int numCourses;
```

Methods:

`void addCourse(const char* courseCode):` Adds a course to the instructor's list.
`void displayInstructor():` Displays instructor details.
`void saveToFile(ofstream& out) override:` Saves instructor details to a file.
`void loadFromFile(ifstream& in) override:` Loads instructor details from a file.

- **Student Class**

Attributes:

```
int studentID;
char name[100];
char enrolledCourses[max][10];
int numCourses;
```

Methods:

`void addCourse(const char* courseCode):` Adds a course to the student's list.
`void removeCourse(const char* courseCode):` Removes a course from the student's list.
`void displayStudent():` Displays student details.
`void searchCourse(const char* courseCode):` Searches for a course in the student'list.
`void saveToFile(ofstream& out) override:` Saves student details to a file.
`void loadFromFile(ifstream& in) override:` Loads student details from a file.

- **UniversityMgmt Class:**

The `UniversityMgmt` class manages the operations of the university course management system.

Attributes: It includes arrays of `Course`, `Student`, and `Instructor` objects, and counters to keep track of the number of each entity.

```
Student students[max];
Course courses[max];
Instructor instructors[max];
int studentCount = 0;
int courseCount = 0;
int instructorCount = 0;
bool isLoggedIn = false;
```

Methods:

`void addCourse()`: Adds a new course to the system.
`void searchCourse()`: Searches for a course by its code.
`void deleteCourse()`: Deletes a course by its code.
`void displayCourse(Course course)`: Displays course details.
`void addStudent()`: Adds a new student to the system.
`void searchStudent()`: Searches for a student by their ID.
`void deleteStudent()`: Deletes a student by their ID.
`void studentSummaryReport()`: Generates a summary report for all students.
`void addInstructor()`: Adds a new instructor to the system.
`void assignCourseToInstructor()`: Assigns a course to an instructor.
`void assignCourseToStudent()`: Assigns a course to a student.
`void instructorSummaryReport()`: Generates a summary report for all instructors.
`void saveData()`: Saves all data to files.
`void loadData()`: Loads all data from files.

User Interface

The system uses a command-line interface (CLI) to interact with users. The main menu offers various options for managing courses, students, and instructors. Here's an outline of the menu:

Courses:
Add Course
Search Course
Delete Course

Students:
Add Student
Search Student
Delete Student
Student Summary Report

Instructors:
Add Instructor
Assign Course to Instructor
Assign Course to Student
Instructor Summary Report

1. **Data Management:**

Save Data

2. **Logout:**

Logout from the system

The system ensures that only authenticated users can access the main functionalities, using a simple login mechanism with hardcoded credentials:

```
bool login() {

    char username[50], password[50];
    cout << "\n Enter Username: ";
    cin >> username;
    cout << "\n Enter Password: ";
    cin >> password;
    if (strcmp(username, "PROJECT") == 0 && strcmp(password, "OOP") == 0) {
        isLoggedIn = true;
        return true;
    } else {
        cout << "\n INVALID CREDENTIALS!!!!!!!!";
        return false;
    }
}
```

Data Persistence

The system uses files to save and load data, ensuring persistence across sessions. Here's how the system handles data persistence:

- **Saving Data:**

```
void UniversityMgmt::saveData() {
    ofstream courseFile("courses.dat"), studentFile("students.dat"),
instructorFile("instructors.dat");
    for (int i = 0; i < courseCount; i++) {
        courses[i].saveToFile(courseFile);
    }
    for (int i = 0; i < studentCount; i++) {
        students[i].saveToFile(studentFile);
    }
    for (int i = 0; i < instructorCount; i++) {
        instructors[i].saveToFile(instructorFile);
    }
    courseFile.close();
    studentFile.close();
    instructorFile.close();
    cout << "\n Data SAVED Successfully!!!!!!";
}
```

- **Loading Data:**

```
void UniversityMgmt::loadData() {
    ifstream courseFile("courses.dat"), studentFile("students.dat"),
instructorFile("instructors.dat");
    courseCount = 0;
    studentCount = 0;
    instructorCount = 0;
    while (courseFile.peek() != EOF) {
        courses[courseCount].loadFromFile(courseFile);
        courseCount++;
    }
    while (studentFile.peek() != EOF) {
        students[studentCount].loadFromFile(studentFile);
         studentCount++;
    }
    while (instructorFile.peek() != EOF) {
        instructors[instructorCount].loadFromFile(instructorFile);
        instructorCount++;
    }
    courseFile.close();
    studentFile.close();
    instructorFile.close();
    cout << "\nData LOADED Successfully!!!!!!!" << endl;
}
```

CODE

```
#include<iostream>
#include<fstream>
#include<string.h>
#define max 100
using namespace std;
class University{
    public:
     virtual void saveToFile(ofstream&)=0;
     virtual void loadFromFile(ifstream&)=0;
};
class Course:public University{
    public:
     char courseName[100];
     char courseCode[10];
     int credits;
     char instructor[100];
     void saveToFile(ofstream& out)override
     {
       out<<courseName<<"  "<<courseCode<<"  "<<credits<<"
"<<instructor<<endl;
     }
       void loadFromFile(ifstream& in)override
     {
       in>>courseName>>courseCode>>credits>>instructor;
     }
};
class Instructor:public University{
    public:
     char name[100];
     char id[10];
     char assignedCourses[max][10];
     int numCourses;
     Instructor():numCourses(0){}
     void addCourse(const char* courseCode)
     {
       strcpy(assignedCourses[numCourses],courseCode);
       numCourses++;
     }
     void displayInstructor()
     {
       cout<<"\n Instructor ID: "<<id;
       cout<<"\n Name: "<<name;
       cout<<"\n Assigned Courses: ";
       for(int i=0;i<numCourses;i++)
       {

    cout<<assignedCourses[i]<<"  ";
       }
```

```
      cout<<endl;
     }
      void saveToFile(ofstream& out)override
     {
      out<<name<<"  "<<id<<"  "<<numCourses<<"  "<<endl;
      for(int i=0; i<numCourses; i++)
      {

    out<<assignedCourses[i]<<"  ";
      }
      out<<endl;
     }
      void loadFromFile(ifstream& in)override
     {
      in>>name>>id>>numCourses;
      for(int i=0; i<numCourses;i++){

    in>>assignedCourses[i];
      }
     }
};
class Student:public University{
    public:
     int studentID;
     char name[100];
     char enrolledCourses[max][10];
     int numCourses;
     Student():numCourses(0){}
     void addCourse(const char* courseCode )
     {
      strcpy(enrolledCourses[numCourses],courseCode);
      numCourses++;
     }
     void removeCourse(const char* courseCode)
     {
      int i, found=0;
      for(i=0;i<numCourses;i++){
                                                  if
(strcmp(enrolledCourses[i],courseCode)==0){
                                                            found=1;
                                                            break;
                                                  }
      }
      if(found){
                                                  for(int
j=i;j<numCourses-1;j++){

    strcpy(enrolledCourses[j],enrolledCourses[j+1]);
                                                  }
                                                  numCourses--;
      }
     }
```

```
     void displayStudent(){
       cout<<"\n Student ID:  "<<studentID;
       cout<<"\n Name: "<<name;
       cout<<"\n Enrolled Courses: ";
       for(int i=0;i<numCourses;i++)
       {

    cout<<enrolledCourses[i]<<"  ";
       }
       cout<<endl;
     }
     void searchCourse(const char* courseCode){
       for(int i=0;i<numCourses;i++)
       {

    if(strcmp(enrolledCourses[i],courseCode)==0){

    cout<<"\n Course  "<<courseCode<<"FOUND for student
"<<name<<endl;
                                                          return;
                                                 }
       }
       cout<<"\n Course  "<<courseCode<<"NOT FOUND for student
"<<name<<endl;
     }
     void saveToFile(ofstream& out)override
     {
       out<<studentID<<"  "<<name<<"  "<<numCourses<<"  "<<endl;
       for(int i=0; i<numCourses; i++)
       {

    out<<enrolledCourses[i]<<"  ";
       }
       out<<endl;
     }
       void loadFromFile(ifstream& in)override
     {
       in>>studentID>>name>>numCourses;
       for(int i=0; i<numCourses;i++){

    in>>enrolledCourses[i];
       }
     }
};
Student students[max];
Course courses[max];
Instructor instructors[max];
int studentCount=0;
int courseCount=0;
int instructorCount=0;
bool isLoggedIn=false;
```

```
class UniversityMgmt{
     public:
      void addCourse();
      void searchCourse();
      void deleteCourse();
      void displayCourse(Course);
      void addStudent();
      void searchStudent();
      void deleteStudent();
      void studentSummaryReport();
      void addInstructor();
      void assignCourseToInstructor();
      void assignCourseToStudent();
      void instructorSummaryReport();
      void saveData();
      void loadData();
};
void UniversityMgmt::addCourse()
{
    Course course;
    cout<<"\n Enter Course Code: ";
    cin>>course.courseCode;
    cout<<"\n Enter Course Name: ";
    cin>>course.courseName;
    cout<<"\n Enter Credits: ";
    cin>>course.credits;
    strcpy(course.instructor, "");
    courses[courseCount]=course;
    courseCount++;
    cout<<"\n Course ADDED successfully!!!!!";
}
void UniversityMgmt::searchCourse()
{
    char courseCode[10];
    int i, found =0;
    cout<<"\n Enter Course Code: ";
    cin>>courseCode;
     for (i = 0; i < courseCount; i++)
     {
        if (strcmp(courses[i].courseCode,courseCode) == 0)
      {
            found = 1;
            break;
        }
    }
    if (found)
     {
        displayCourse(courses[i]);
    } else
     {
        cout << "\nCourse not found";
    }
```

```
}
void UniversityMgmt::deleteCourse()
{
     char courseCode[10];
    int i, found =0;
    cout<<"\n Enter Course Code to DELETE: ";
    cin>>courseCode;
     for (i = 0; i < courseCount; i++)
     {
        if (strcmp(courses[i].courseCode, courseCode) == 0)
      {
            found = 1;
            break;
        }
    }
    if (found)
    {
     for(int j=i;  j<courseCount- 1;j++)
      {
            courses[j] = courses[j + 1];
        }
        courseCount--;
        cout<<"\n Course DELETED Successfully!!!!!!";
    } else
     {
        cout<<"\n Course NOT FOUND";
    }
}
void UniversityMgmt::displayCourse(Course course)
{
    cout<<"\n Course Code: " <<course.courseCode;
    cout<<"\n Course Name: " <<course.courseName;
    cout<<"\n Credits: " <<course.credits;
    if (strlen(course.instructor) > 0)
     {
        cout<<"\nInstructor: " << course.instructor;
    } else
     {
        cout<<"\nInstructor: Not Assigned";
    }
    cout<<endl;
}
void UniversityMgmt::addStudent(){
    Student student;
    cout<<"\n Enter Student ID: ";
    cin>>student.studentID;
    cout<<"\n Enter Student Name: ";
    cin>>student.name;
    student.numCourses = 0;
    students[studentCount] = student;
    studentCount++;
    cout<<"\n Student ADDED Successfully!!!!!!";
```

```
}
void UniversityMgmt::searchStudent()
 {
    int studentID,i,found = 0;
    cout<<"\n Enter Student ID: ";
    cin>>studentID;
    for (i = 0; i < studentCount; i++)
    {
        if (students[i].studentID == studentID)
      {
            found = 1;
            break;
        }
    }
    if (found)
    {
        students[i].displayStudent();
    } else
     {
        cout <<"\n Student NOT FOUND";
    }
}
void UniversityMgmt::deleteStudent()
{
    int studentID,i,found = 0;
    cout<<"\n Enter Student ID to Delete: ";
    cin>>studentID;
    for(i = 0; i < studentCount; i++){
        if (students[i].studentID == studentID)
      {
            found = 1;
            break;
        }
    }
    if (found)
    {
        for (int j = i; j < studentCount - 1; j++)
      {
            students[j] = students[j + 1];
        }
        studentCount--;
        cout<<"\n Student DELETED Successfully!!!!!!";
    } else
     {
        cout<<"\nStudent NOT FOUND";
    }
}
void UniversityMgmt::studentSummaryReport()
{
    if(studentCount == 0)
     {
        cout<<"\n NO Students in University!!!!!!";
```

```
    } else
     {
        for (int i = 0; i<studentCount; i++)
      {
            students[i].displayStudent();
            cout<<"\n----------------------------------------
------------------------";
        }
    }
}
void UniversityMgmt::addInstructor()
 {
    Instructor instructor;
    cout<<"\n Enter Instructor ID: ";
    cin>>instructor.id;
    cout<<"\n Enter Instructor Name: ";
    cin>>instructor.name;
    instructor.numCourses = 0;
    instructors[instructorCount] = instructor;
    instructorCount++;
    cout<<"\n Instructor ADDED Successfully!!!!!!";
}
void UniversityMgmt::assignCourseToInstructor()
 {
    char instructorID[10],courseCode[10];
    int i,j,foundInstructor = 0, foundCourse = 0;
    cout<<"\n Enter Instructor ID: ";
    cin>>instructorID;
    cout<<"\n Enter Course Code: ";
    cin>>courseCode;
    for(i=0; i<instructorCount; i++)
    {
        if(strcmp(instructors[i].id,instructorID)== 0)
      {
            foundInstructor=1;
            break;
        }
    }
    for (j=0;j<courseCount; j++)
    {
        if (strcmp(courses[j].courseCode,courseCode)== 0)
      {
            foundCourse= 1;
            break;
        }
    }
    if (foundInstructor && foundCourse)
     {
        instructors[i].addCourse(courseCode);
        strcpy(courses[j].instructor, instructors[i].name);
        cout<<"\n Course ASSIGNED TO INSTRUCTOR
Successfully!!!!!!";
```

```
    } else
     {
        if (!foundInstructor)
     cout<<"\n Instructor NOT FOUND";
        if (!foundCourse)
      cout<<"\n Course NOT FOUND";
    }
}
void UniversityMgmt::assignCourseToStudent()
 {
    int studentID, i, j, foundStudent= 0, foundCourse= 0;
    char courseCode[10];
    cout<<"\n Enter Student ID: ";
    cin>>studentID;
    cout<<"\n Enter Course Code: ";
    cin>>courseCode;
    for(i = 0; i < studentCount; i++)
    {
        if (students[i].studentID == studentID)
     {
            foundStudent= 1;
            break;
        }
    }
    for(j = 0; j<courseCount; j++)
    {
        if(strcmp(courses[j].courseCode,courseCode)==0)
      {
            foundCourse = 1;
            break;
        }
    }
    if(foundStudent && foundCourse)
    {
        students[i].addCourse(courseCode);
        cout<<"\n Course ASSIGNED TO STUDENT
Successfully!!!!!!!";
    } else
     {
        if (!foundStudent)
      cout<<"\n Student NOT FOUND";
        if (!foundCourse)
      cout<<"\n Course NOT FOUND";
    }
}
void UniversityMgmt::instructorSummaryReport()
 {
    if (instructorCount == 0)
     {
        cout<<"\n No Instructors in University!";
    } else
    {
```

```
        for (int i = 0; i<instructorCount; i++)
    {
            instructors[i].displayInstructor();
            cout<<"\n----------------------------------------
-------------";
        }
    }
}
void UniversityMgmt::saveData()
{
    ofstream
courseFile("courses.dat"),studentFile("students.dat"),
     instructorFile("instructors.dat");
    for(int i=0;i<courseCount; i++)
     {
        courses[i].saveToFile(courseFile);
    }
    for(int i=0;i<studentCount; i++)
    {
        students[i].saveToFile(studentFile);
    }
    for(int i=0;i<instructorCount; i++)
     {
        instructors[i].saveToFile(instructorFile);
    }
    courseFile.close();
    studentFile.close();
    instructorFile.close();
    cout<<"\n Data SAVED Successfully!!!!!!";
}
void UniversityMgmt::loadData(){
    ifstream
courseFile("courses.dat"),studentFile("students.dat"),
     instructorFile("instructors.dat");
    courseCount= 0;
    studentCount= 0;
    instructorCount= 0;
    while(courseFile.peek()!= EOF)
    {
        courses[courseCount].loadFromFile(courseFile);
        courseCount++;
    }
    while(studentFile.peek()!= EOF)
     {
        students[studentCount].loadFromFile(studentFile);
        studentCount++;
    }
    while(instructorFile.peek() != EOF)
    {

instructors[instructorCount].loadFromFile(instructorFile);
        instructorCount++;
```

```
    }
    courseFile.close();
    studentFile.close();
    instructorFile.close();
    cout<<"\nData LOADED Successfully!!!!!!!"<<endl;
}
bool login()
 {
    char username[50], password[50];
    cout<<"\n Enter Username: ";
    cin>>username;
    cout<<"\n Enter Password: ";
    cin>>password;
    if (strcmp(username,"PROJECT") == 0 &&
strcmp(password,"OOP") == 0)
     {
        isLoggedIn=true;
        return true;
    } else
     {
        cout<<"\n INVALID CREDENTIALS!!!!!!!!";
        return false;
    }
}
void logout()
{
    isLoggedIn=false;
    cout<<"LOGGED OUT Successfully!!!!!!\n";
}
int main()
 {
    UniversityMgmt um;
    int choice;
    if (!login())
    {
        return 0;
    }
    um.loadData();
    do {
     cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
WELCOMEEEE~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"<<endl;
     cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ TO
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"<<endl;
        cout<<"\n\n ==========~~~~~~~~~~~A&A UNIVERSITY
COURSE MANAGEMENT SYSTEM~~~~~~~~~~~==========";
        cout<<"\n 1. Add Course";
        cout<<"\n 2. Search Course";
        cout<<"\n 3. Delete Course";
        cout<<"\n
============================================================
==";
        cout<<"\n 4. Add Student";
```

```
        cout<<"\n 5. Search Student";
        cout<<"\n 6. Delete Student";
        cout<<"\n 7. Student Summary Report";
        cout<<"\n
================================================================
==";
        cout<<"\n 8. Add Instructor";
        cout<<"\n 9. Assign Course to Instructor";
        cout<<"\n 10. Assign Course to Student";
        cout<<"\n 11. Instructor Summary Report";
        cout<<"\n
================================================================
==";
        cout<<"\n 12. Save Data";
        cout<<"\n
================================================================
==";
        cout<<"\n 13. Logout";
        cout<<"\n
================================================================
==";
        cout<<"\n Enter Your Choice (1-13): ";
        cin>>choice;

        switch (choice)
     {
        case 1:
            um.addCourse();
            break;
        case 2:
            um.searchCourse();
            break;
        case 3:
            um.deleteCourse();
            break;
        case 4:
            um.addStudent();
            break;
        case 5:
            um.searchStudent();
            break;
        case 6:
            um.deleteStudent();
            break;
        case 7:
            um.studentSummaryReport();
            break;
        case 8:
            um.addInstructor();
            break;
        case 9:
            um.assignCourseToInstructor();
```

```
            break;
        case 10:
            um.assignCourseToStudent();
            break;
        case 11:
            um.instructorSummaryReport();
            break;
        case 12:
            um.saveData();
            break;
        case 13:
            logout();
            break;
        default:
            cout<<"\n INVALID CHOICE!!!!!!";
            cout<<"\n TRY AGAIN :) \n";
        }
    }
    while (isLoggedIn);

    return 0;
}
```

OUTPUT

LOGIN: When we enter wrong username or password it shows

```
 Enter Username: PROJECT

 Enter Password: oom

 INVALID CREDENTIALS!!!!!!!!
--------------------------------
Process exited after 18.95 seconds with return value 0
Press any key to continue . . .
```

MENU is as following

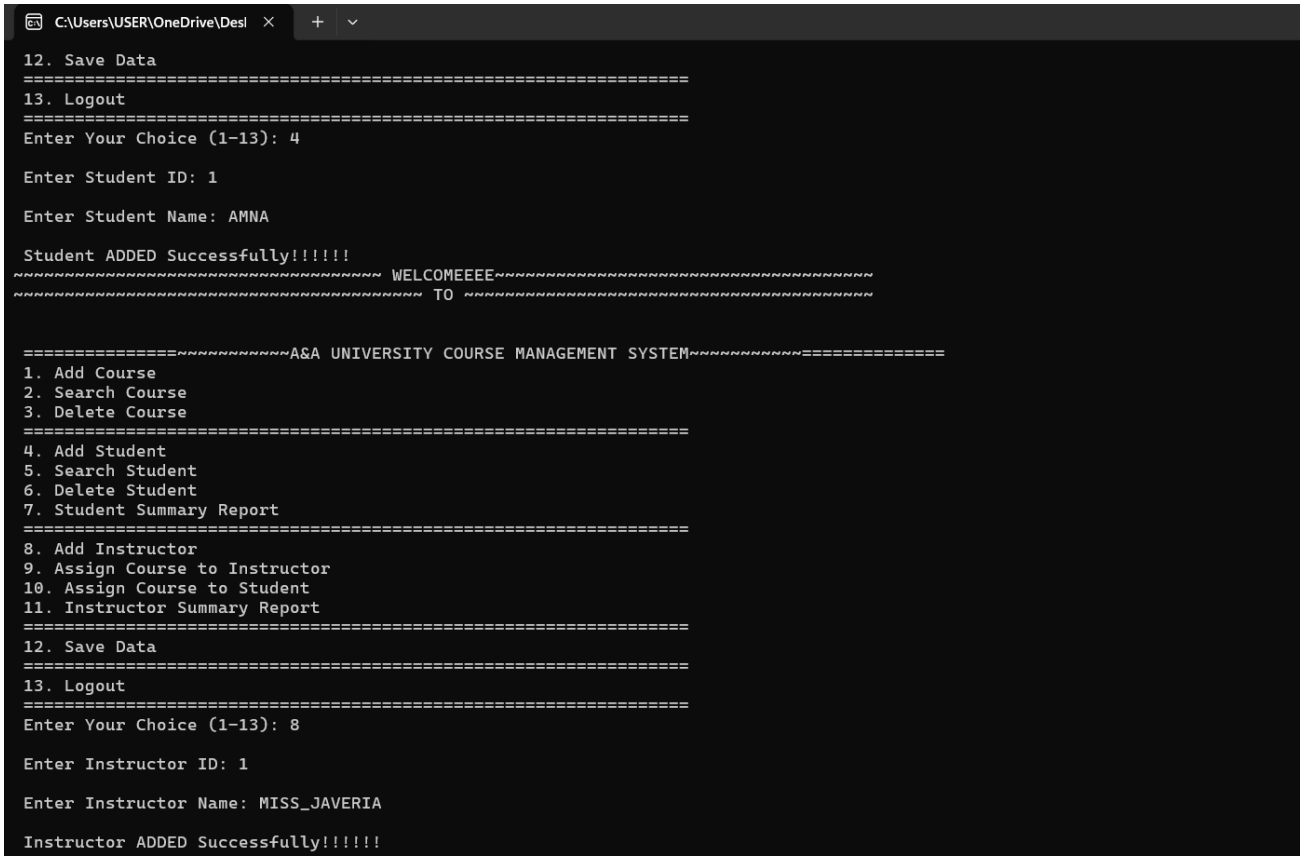
```
C:\Users\USER\OneDrive\Desl

 Enter Username: PROJECT

 Enter Password: OOP

Data LOADED Successfully!!!!!!!
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ WELCOMEEEE~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ TO ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


 ===============~~~~~~~~~~~A&A UNIVERSITY COURSE MANAGEMENT SYSTEM~~~~~~~~~~~==============
 1. Add Course
 2. Search Course
 3. Delete Course
 =================================================================
 4. Add Student
 5. Search Student
 6. Delete Student
 7. Student Summary Report
 =================================================================
 8. Add Instructor
 9. Assign Course to Instructor
 10. Assign Course to Student
 11. Instructor Summary Report
 =================================================================
 12. Save Data
 =================================================================
 13. Logout
 =================================================================
 Enter Your Choice (1-13):
```

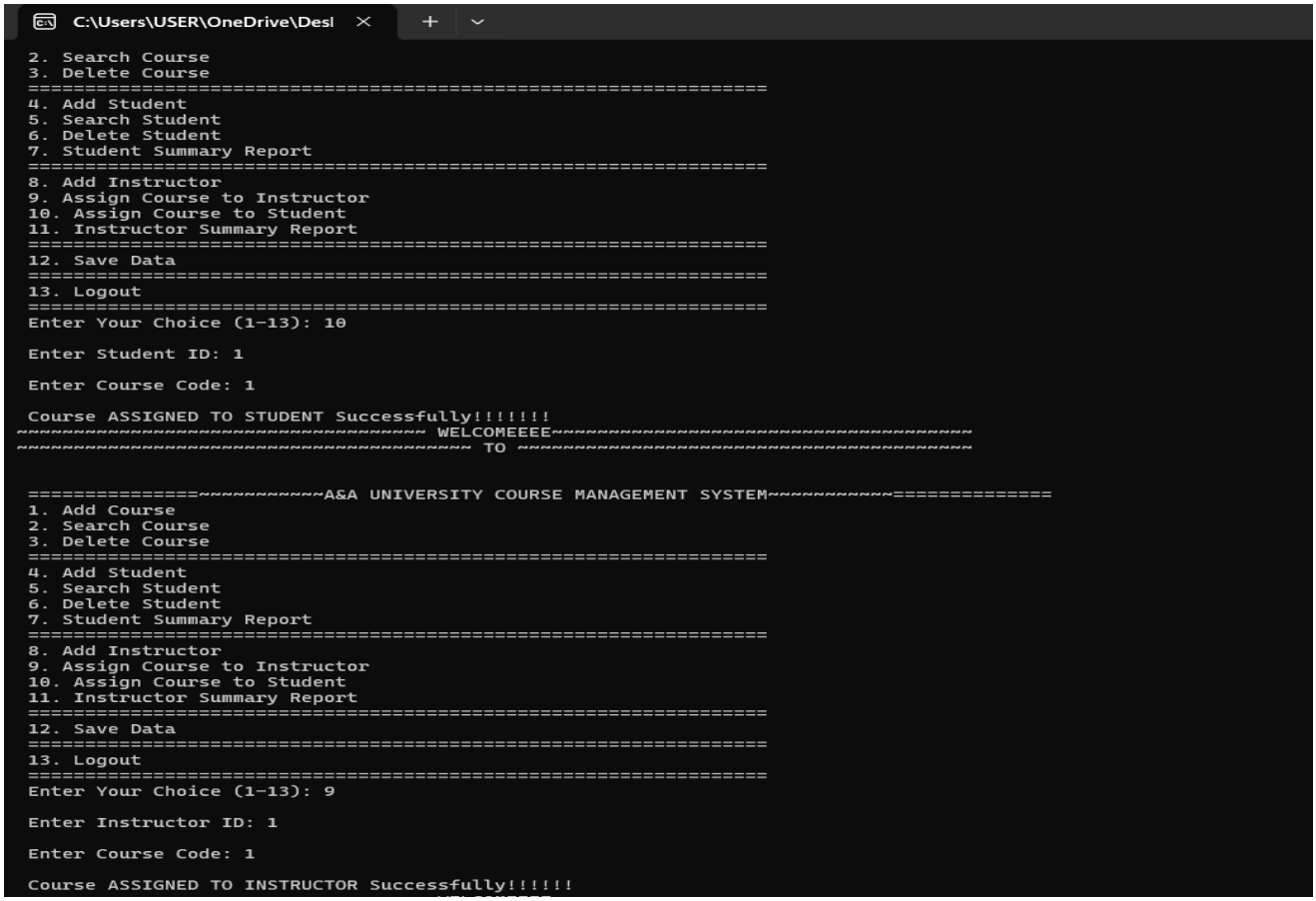
First enter the choice, when we add course in it only then we can search, delete, and assign it to the student or instructor

```
-------------------------------------
Enter Your Choice (1-13): 1

Enter Course Code: 1

Enter Course Name: oop

Enter Credits: 2

Course ADDED successfully!!!!!
```

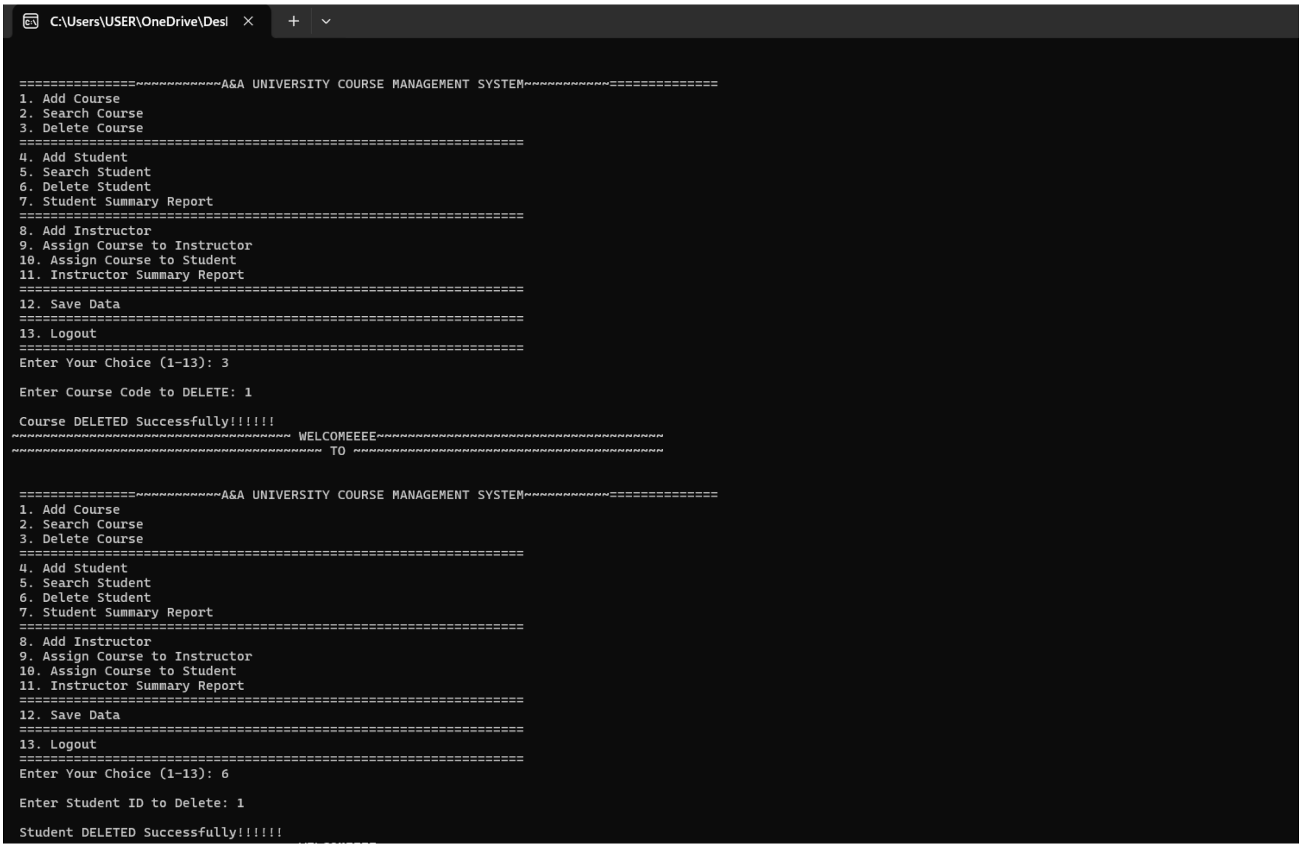
Now, we can add more courses or add students and then instructors and then assign them to students and instructors

```
C:\Users\USER\OneDrive\Desl
=================================================================
12. Save Data
=================================================================
13. Logout
=================================================================
Enter Your Choice (1-13): 1

Enter Course Code: 1

Enter Course Name: OOP

Enter Credits: 2

Course ADDED successfully!!!!!
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ WELCOMEEEE~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ TO ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


===============~~~~~~~~~~~A&A UNIVERSITY COURSE MANAGEMENT SYSTEM~~~~~~~~~~~==============
1. Add Course
2. Search Course
3. Delete Course
=================================================================
4. Add Student
5. Search Student
6. Delete Student
7. Student Summary Report
=================================================================
8. Add Instructor
9. Assign Course to Instructor
10. Assign Course to Student
11. Instructor Summary Report
=================================================================
12. Save Data
=================================================================
13. Logout
=================================================================
Enter Your Choice (1-13): 4

Enter Student ID: 1

Enter Student Name: AMNA

Student ADDED Successfully!!!!!!
```

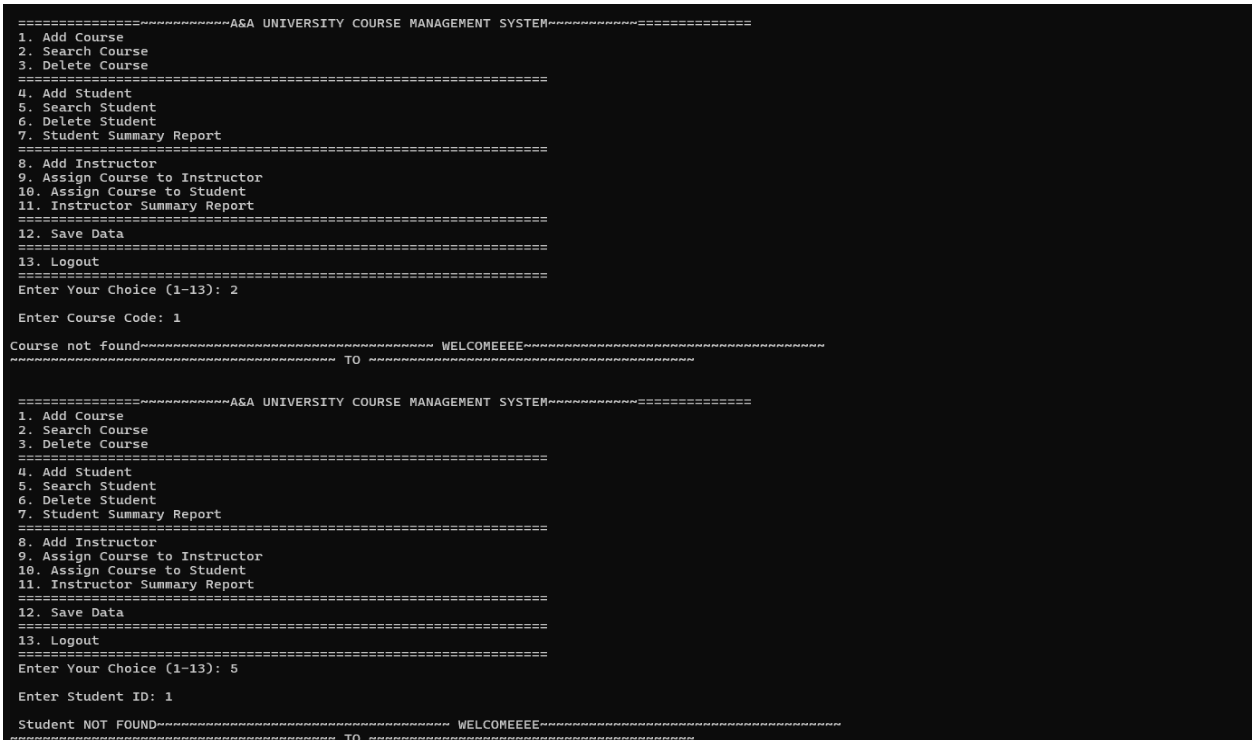

C:\Users\USER\OneDrive\Desl
12. Save Data
13. Logout
Enter Your Choice (1-13): 4
Enter Student ID: 1
Enter Student Name: AMNA
Student ADDED Successfully!!!!!!
WELCOMEEEE
TO
A&A UNIVERSITY COURSE MANAGEMENT SYSTEM
1. Add Course
2. Search Course
3. Delete Course
4. Add Student
5. Search Student
6. Delete Student
7. Student Summary Report
8. Add Instructor
9. Assign Course to Instructor
10. Assign Course to Student
11. Instructor Summary Report
12. Save Data
13. Logout
Enter Your Choice (1-13): 8
Enter Instructor ID: 1
Enter Instructor Name: MISS_JAVERIA
Instructor ADDED Successfully!!!!!!


C:\Users\USER\OneDrive\Desl
2. Search Course
3. Delete Course
4. Add Student
5. Search Student
6. Delete Student
7. Student Summary Report
8. Add Instructor
9. Assign Course to Instructor
10. Assign Course to Student
11. Instructor Summary Report
12. Save Data
13. Logout
Enter Your Choice (1-13): 10
Enter Student ID: 1
Enter Course Code: 1
Course ASSIGNED TO STUDENT Successfully!!!!!!!
WELCOMEEEE
TO
A&A UNIVERSITY COURSE MANAGEMENT SYSTEM
1. Add Course
2. Search Course
3. Delete Course
4. Add Student
5. Search Student
6. Delete Student
7. Student Summary Report
8. Add Instructor
9. Assign Course to Instructor
10. Assign Course to Student
11. Instructor Summary Report
12. Save Data
13. Logout
Enter Your Choice (1-13): 9
Enter Instructor ID: 1
Enter Course Code: 1
Course ASSIGNED TO INSTRUCTOR Successfully!!!!!!

Now, we can search or delete student or course e.tc by entering course code



As we deleted the data, so when we search them there will be no result



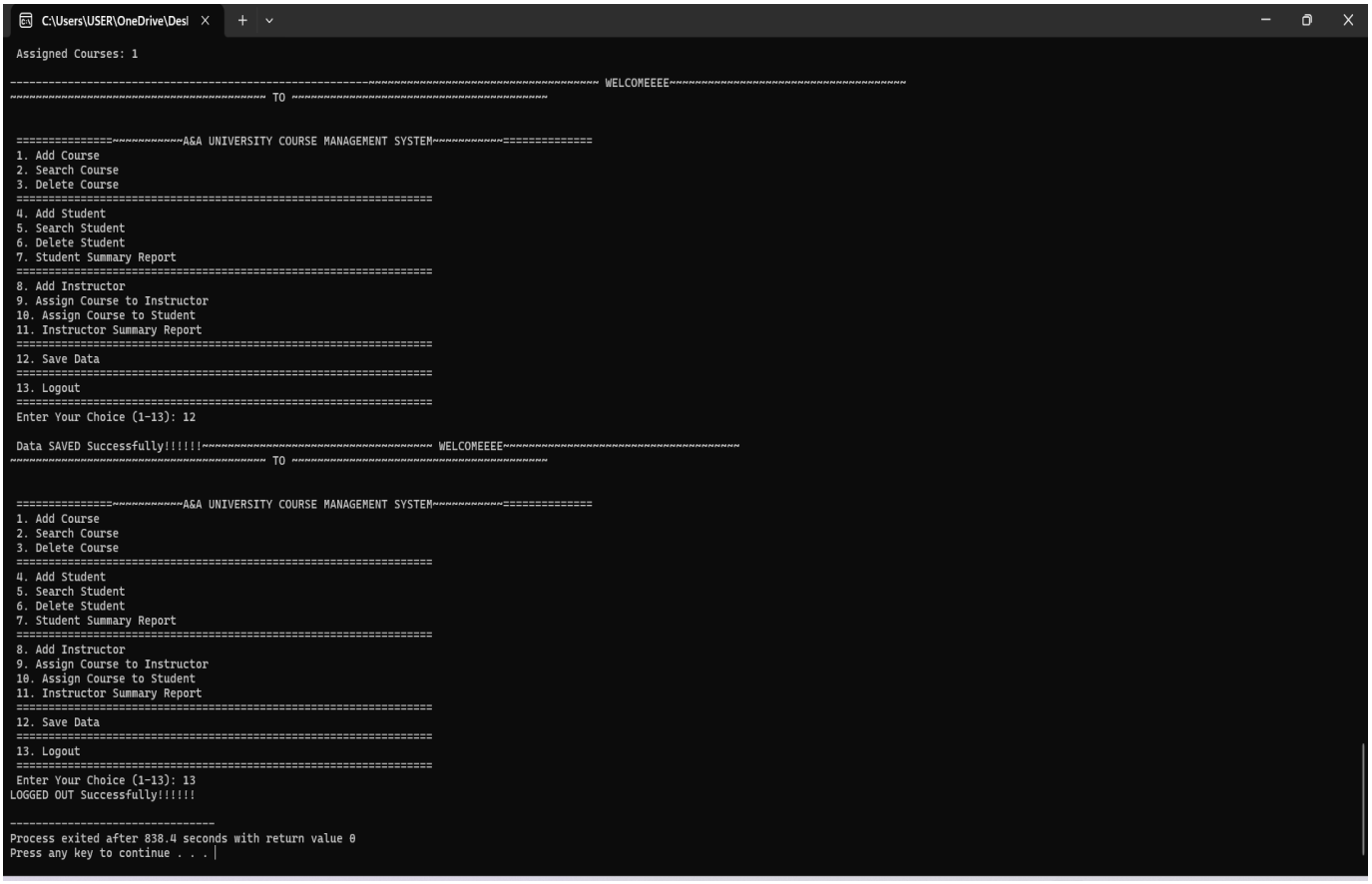
In student summary report, there will be no result as we deleted the data while it shows the instructor summary report when we enter choice 11.

```
 Student NOT FOUND~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ WELCOMEEEE~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ TO ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


 =============~~~~~~~~~~~A&A UNIVERSITY COURSE MANAGEMENT SYSTEM~~~~~~~~~~~=============
 1. Add Course
 2. Search Course
 3. Delete Course
 =================================================================
 4. Add Student
 5. Search Student
 6. Delete Student
 7. Student Summary Report
 =================================================================
 8. Add Instructor
 9. Assign Course to Instructor
 10. Assign Course to Student
 11. Instructor Summary Report
 =================================================================
 12. Save Data
 =================================================================
 13. Logout
 =================================================================
 Enter Your Choice (1-13): 7

 NO Students in University!!!!!!
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ WELCOMEEEE~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ TO ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


 =============~~~~~~~~~~~A&A UNIVERSITY COURSE MANAGEMENT SYSTEM~~~~~~~~~~~=============
 1. Add Course
 2. Search Course
 3. Delete Course
 =================================================================
 4. Add Student
 5. Search Student
 6. Delete Student
 7. Student Summary Report
 =================================================================
 8. Add Instructor
 9. Assign Course to Instructor
 10. Assign Course to Student
 11. Instructor Summary Report
 =================================================================
 12. Save Data
 =================================================================
 13. Logout
 =================================================================
 Enter Your Choice (1-13): 11

 Instructor ID: 1
 Name: MISS_JAVERIA
 Assigned Courses: 1
```

We can also save the data we enter in it

```
C:\Users\USER\OneDrive\Desl

 6. Delete Student
 7. Student Summary Report
 =================================================================
 8. Add Instructor
 9. Assign Course to Instructor
 10. Assign Course to Student
 11. Instructor Summary Report
 =================================================================
 12. Save Data
 =================================================================
 13. Logout
 =================================================================
 Enter Your Choice (1-13): 11

 Instructor ID: 1
 Name: MISS_JAVERIA
 Assigned Courses: 1
------------------------------------------------------------~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ WELCOMEEEE~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ TO ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


 =============~~~~~~~~~~~A&A UNIVERSITY COURSE MANAGEMENT SYSTEM~~~~~~~~~~~=============
 1. Add Course
 2. Search Course
 3. Delete Course
 =================================================================
 4. Add Student
 5. Search Student
 6. Delete Student
 7. Student Summary Report
 =================================================================
 8. Add Instructor
 9. Assign Course to Instructor
 10. Assign Course to Student
 11. Instructor Summary Report
 =================================================================
 12. Save Data
 =================================================================
 13. Logout
 =================================================================
 Enter Your Choice (1-13): 12

 Data SAVED Successfully!!!!!!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ WELCOMEEEE~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ TO ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Now, at last we can logout from our course management system



~ ~ THE END ~ ~