UNIVERSITY DATABASE SYSTEM

MTE innovative project for

OBJECT ORIENTED PROGRAMMING

(SE 203)

Submitted By:

Parth Dipendra (2K19/EP/067) &

Saksham Mishra (2K19/EP/083) &

Raj Chaudhary (2K19/ME/184)

Submitted To:

Ms. Anjali Bansal

Department of Software Engineering

DELHI TECHNOLOGICAL UNIVERSITY

ABSTRACT

This report provides the designing and implementation of a robust university management system. This management system provides a solution to the problems faced by the universities and allows for a more efficient system. This uses the concepts of object oriented programming and database management in order to implement this system.

This project is written in C++ programming language and provides for file handling as well as to store all the data locally on the systems being used. This program will allow for both teacher and student logins and store all of their credentials. In addition, teachers may use this system to assign not just class but also individual assignments directly to the students intended.

Notifications, tests and exams can also be declared using this system. Hence, the program allows for multiple features for both the faculty and the students in a neat manner and makes things more convenient by simply building on the principles of object oriented programming.

INTRODUCTION

We make use of the following concepts of object oriented programming in order to build our program:

➢ Classes & Objects

Classes and Objects are basic concepts of Object Oriented Programming which revolve around real life entities. A class is a user defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects of one type.

Objects are a basic unit of Object-Oriented Programming and represent real life entities. Objects are an instance of classes and share the attributes and the behavior of the class.

➢ Inheritance

The capability of a class to derive properties and characteristics from another class is called Inheritance. Inheritance is one of the most important feature of Object Oriented Programming.

- **Sub Class** - The class that inherits properties from another class is called Subclass or Derived Class.

- **Super Class** - The class whose properties are inherited by subclass is called Base Class or Super class.

➢ Polymorphism

One may define polymorphism as the ability of a message to be displayed in more than one form. Polymorphism may be in the form of operator or function overloading.

➢ Abstraction

Data abstraction is one of the most essential and important feature of object oriented programming. Abstraction means displaying only essential information and hiding the details. Data abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation.

➢ Encapsulation

In Object Oriented Programming, Encapsulation is defined as binding together the data and the functions that manipulate them. Encapsulation also lead to data abstraction or hiding.

➢ File handling

In C++, files are mainly dealt with by using three classes fstream, ifstream, ofstream available in fstream header file. File handling is used to manage the files and the data contained in them. The content in files can be updated, added or deleted using the concept of file handling.

Program

Header files utilized in the code are:

1. #include<iostream>
2. #include<fstream>
3. #include<string.h>

- **#include<iostream>**

iostream is the header file which contains all the functions of program like cout, cin etc. and #include tells the preprocessor to include these header file in the program.

- **#include<fstream>**

Header providing file stream classes:

1. basic_ifstream : Input file stream
2. basic_ofstream : Output file stream
3. basic_fstream : File stream
4. basic_filebuf : File stream buffer

- **#include<string.h>**

The string.h header defines one variable type, one macro, and various functions for manipulating arrays of characters.

Classes Utilized

- Student
- Lecturer
- Assignment for all
- Assignment for one
- Remove teacher
- Remove student (Inherits Remove teacher class)
- Print data (Polymorphism)

Structure for our program

1. Faculty Registration and Login
2. Student Registration and Login
3. Accessing the records
4. Assignment of tasks to students by teachers.
5. Deletion of both Student and Teacher records

The code for our program is as follows:

```
#include <iostream>
#include <fstream>
#include <string.h>
using namespace std;
void now(); // class
class Student
{
   char suserName[20];
   char spassword[20];
   char section[10];
   int id, n;

 public:
   Student() {}
   static int Studentcount; //static integer is used to count
   void FromFile();
   void getFile();
   void updateData(long int idd);
   void deleteSectionassignment(char *k);
   void StudentData();
   char *getName();
   char *getSPassword();
   char *sec();
   int getIdOfStudent();
   void display();
   void storeDataInFile();
};
void Student::storeDataInFile()
{
   Student a;
   a.StudentData();
   ofstream fout;
```

```
    fout.open("studentData.bin", ios::app | ios::binary);
    fout.write((char *)&a, sizeof(class Student));
    fout.close();
}
void Student::getFile()
{
    Student a;
    ifstream fin;
    fin.open("studentData.bin", ios::in | ios::binary);
    fin.read((char *)&a, sizeof(class Student));
    while (!fin.eof())
    {
        a.display();
        fin.read((char *)&a, sizeof(class Student));
    }
    fin.close();
}
inline void Student::display() //inline keyword is used
{
    cout << suserName << " " << section << " " << id << endl;
}

//getting user from the admin
void Student::StudentData()
{
    Student s3;
    cout << "enter the username of the student" << endl;
    cin.getline(suserName, 19);
    cin.ignore();
    cout << "enter the pass word for the student" << endl;
    cin.getline(spassword, 19);
    cin.ignore();
    cout << "enter student section" << endl;
    cin.getline(section, 10);
    cin.ignore();
```

```
   cout << "enter id of the student" << endl;
   cin >> id;
   cin.ignore(1);
   s3.Studentcount = s3.Studentcount + 1;
}

//returnns the password of the student
inline char *Student::getSPassword() //inline keyword is used
{
   return spassword;
}
//returnds the username of the student
inline char *Student::getName() //inline keyword is used
{
   return suserName;
}
//returns the id of the partiuclar student
inline int Student::getIdOfStudent() //inline keyword is used
{
   return id;
}

class Lecturer
{
   char password[20];
   char userName[20];
   int lecturerId;

 public:
   //default constructor
   static int LecturerCount; //static interger used to count
   Lecturer() {}
   //member function to get data from the user
   void getData();
   //getters
```

```
   char *getPassword();
   char *getUserName();
   void display();
   void storeDataInFile();
   void getDataFromFile();
   void FromFile();
};
class AssignmentForAll
{
   char section[10];
   char assignmentforall[200];

 public:
   AssignmentForAll() //default constructor
   {
   }
   void getDataForAssignForAll();
   char *getSec();
   void display();
   void storeDataInFile();
   void searchDataInFile(char *s);
    //this function is just to check while creating the projetc this
can be deleted by the client
   int getDataFromFile();
};
void AssignmentForAll::searchDataInFile(char *s)
{
   AssignmentForAll a;
   ifstream fin;
   fin.open("AssignmentForAll.bin", ios::in | ios::binary);
   fin.read((char *)&a, sizeof(class AssignmentForAll));

   while (!fin.eof())
   {
       if (!strcasecmp(a.section, s))
```

```
        {
            a.display();
        }
        fin.read((char *)&a, sizeof(class AssignmentForAll));
    }
    fin.close();
}
inline void AssignmentForAll::display() //inline keyword is used
{
    cout << section << endl;
    cout << assignmentforall << endl;
};
//returns section
char *AssignmentForAll::getSec() { return section; }
//for gettinf data from the user
void AssignmentForAll::getDataForAssignForAll()
{
    cout << "enter the name of the section" << endl;
    //do not enter morw tham 9 charecters
    cin.getline(section, 9);
    cin.ignore();
    cout << "enter assignment in 200 charecters" << endl;
    cin.getline(assignmentforall, 199);
    cin.ignore();
}
void AssignmentForAll::storeDataInFile()
{
    AssignmentForAll obj;
    obj.getDataForAssignForAll();
    ofstream fout;
    fout.open("AssignmentForAll.bin", ios::app | ios::binary);
    fout.write((char *)&obj, sizeof(class AssignmentForAll));
    fout.close();
}
int AssignmentForAll::getDataFromFile()
```

```
{
   AssignmentForAll a;
   ifstream fin;
   fin.open("AssingnmentForAll.bin", ios::in | ios::binary);
   if (!fin)
   {
       cerr << "that file is not created" << endl;
       return -1;
   }
   fin.read((char *)&a, sizeof(class AssignmentForAll));
   while (!fin.eof())
   {
       a.display();
       fin.read((char *)&a, sizeof(class AssignmentForAll));
   }
   fin.close();
   return 1;
}

class AssignmentForOne
{
   int id;
   char assignment[200];

 public:
   AssignmentForOne()
   {
   }
   int getId();
   void getData();
   void display();
   void storeDataInFile();
   void searchFromFile(int idd);
    //this function is just to check while creating the projetc this
can be deleted by the client
```

```
   int getDataFromFile();
   int iddd();
};
void AssignmentForOne::searchFromFile(int idd)
{
   AssignmentForOne a;
   ifstream fin;
   fin.open("assignmentforone.bin", ios::in | ios::binary);
   fin.read((char *)&a, sizeof(class AssignmentForOne));
   while (!fin.eof())
   {
       if (a.id == idd)
       {
           a.display();
       }
       fin.read((char *)&a, sizeof(class AssignmentForOne));
   }
}
//getter for id

inline int AssignmentForOne::getId() { return id; } //inline keyword
is used

void AssignmentForOne::getData()
{
   cout << "enter the assignment that u want to assign" << endl;
   cin.getline(assignment, 199);
   cin.ignore();
     cout << "enter the id of the student u want to assign the
assignment" << endl;
   cin >> id;
   cin.ignore();
}
inline void AssignmentForOne::display() //inline keywird is used
{
```

```
    cout << assignment << id << endl;
}

void AssignmentForOne::storeDataInFile()
{
    ofstream fout;
    AssignmentForOne a;
    a.getData();
    fout.open("assignmentforone.bin", ios::app | ios::binary);
    fout.write((char *)&a, sizeof(class AssignmentForOne));
    fout.close();
}
int AssignmentForOne::getDataFromFile()
{
    AssignmentForOne a;
    ifstream fin;
    fin.open("assignmentforone.bin", ios::in | ios::binary);
    if (!fin)
    {
        cerr << "that file is not created" << endl;
        return -1;
    }
    fin.read((char *)&a, sizeof(class AssignmentForOne));
    while (!fin.eof())
    {
        a.display();
        fin.read((char *)&a, sizeof(class AssignmentForOne));
    }
    fin.close();
    return 1;
}

int Student::Studentcount = 0;

int AssignmentForOne::iddd()
```

```
{
   return id;
}
void DeleteDataForOne()
{
   ofstream file;
       file.open("assignmentforone.bin",  ios::out  |  ios::binary  |
ios::trunc);
   file.close();
   cout << " All the file content  erase succesfully";
   exit(-1);
}
void DeleteDataForAll()
{
   ofstream file1;
      file1.open("AssignmentForAll.bin",  ios::out  |  ios::binary  |
ios::trunc);
   file1.close();
   cout << " All the file content  erase succesfully";
   exit(-1);
}
char *Student::sec() { return section; }
void Student::FromFile()
{
   Student a;
   AssignmentForAll b;
   AssignmentForOne c;
   char ch;
   int i;
   char nameOfStudent[20];
   char passwordOfStudent[20];
   cout << "enter your username" << endl;
   cin.getline(nameOfStudent, 19);
   cin.ignore();
   ifstream fin;
```

```
   fin.open("studentData.bin", ios::in | ios::binary);
   fin.read((char *)&a, sizeof(class Student));
   while (!fin.eof())
   {
       if (!strcasecmp(a.suserName, nameOfStudent))
       {
           cout << "enter your password" << endl;
           cin.getline(passwordOfStudent, 19);
           cin.ignore();
           if (!strcmp(a.spassword, passwordOfStudent))
           {
               cout << "Welcome Student " << endl;

               do
               {
                   cout << "enter  1 for assignment for you" << endl;
                   cout << "enter  2 for assignment of all" << endl;
                   cin >> i;
                   cin.ignore();

                   switch (i)
                   {
                   case 1:
                       b.searchDataInFile(a.sec());
                       break;
                   case 2:
                       c.searchFromFile(a.id);
                       break;
                   default:
                       break;
                   }
                   cout << "do u want to continue" << endl
                            << "Enter Y||y for continue \n N||n for
return to main menu" << endl;
                   cin >> ch;
```

```
                    cin.ignore(2);
                } while (ch == 'y' || ch == 'Y');
            }
        }
        fin.read((char *)&a, sizeof(class Student));
    }
    if (ch == 'n' || ch == 'N')
    {
        cout << "logged out" << endl;
        now();
    }
    fin.close();
}
// void AssignmentForOne::se

class RemoveTeacher
{
 public:
    void RemoveTeacher_()
    {
        ofstream file;
             file.open("LecturerData.bin", ios::out | ios::binary |
ios::trunc);
        file.close();
        cout << "Teachers Remove succesfully";
        exit(-1);
    }
};
class Removestudent : public RemoveTeacher //inheritence is used
which inherit teacher remove function
{
 public:
    void RemoveStudent()
    {
        ofstream file;
```

```
            file.open("studentData.bin",  ios::out  |  ios::binary  |
ios::trunc);
       file.close();
       cout << endl
            << "Students Remove succesfully";
       exit(-1);
   }
};
class printdata //used for operator overloading
{
   Student s;

 public:
   void print(int i)
   {
       cout << "total no.  is-";
       exit(-1);
   }
};

int Lecturer::LecturerCount = 0;
void Student::deleteSectionassignment(char *k)
{
   AssignmentForAll a;
   fstream file;
    file.open("AssignmentForAll.bin", ios::in | ios::out | ios::ate |
ios::binary);
   file.seekg(0);
   file.read((char *)&a, sizeof(class AssignmentForAll));
   while (!file.eof())
   {
       if (k == a.getSec())
       {
           file.seekp(file.tellp() - sizeof(class AssignmentForAll));
           cin.ignore();
```

```
            a.getDataForAssignForAll();
            file.write((char *)&a, sizeof(class AssignmentForAll));
            file.close();
            cout << "assignment updated succesfully" << endl;
        }
        file.read((char *)&a, sizeof(class AssignmentForAll));
    }
}
void Student::updateData(long int idd)
{
    AssignmentForOne a;
    fstream file;
     file.open("assignmentforone.bin", ios::in | ios::out | ios::ate |
ios::binary);
    file.seekg(0);
    file.read((char *)&a, sizeof(class AssignmentForOne));
    while (!file.eof())
    {
        if (idd == a.iddd())
        {
            file.seekp(file.tellp() - sizeof(class AssignmentForOne));
            cin.ignore();
            a.getData();
            file.write((char *)&a, sizeof(class AssignmentForOne));
            file.close();
            cout << "assignment updated succesfully" << endl;
        }
        file.read((char *)&a, sizeof(class AssignmentForOne));
    }
}
void Decide(int i)
{
    printdata p1; //operator overloading is used for print the data
    AssignmentForAll a1;
    AssignmentForOne a2;
```

```
    Student s1;
    Lecturer L1;
    if (i == 13)
    {
        p1.print(s1.Studentcount);
    }
    else if (i == 14)
    {
        p1.print(L1.LecturerCount);
    }
}
int GetIdStudent()
{
    int m;
    cout << "Enter the id of student u want to update" << endl;
    cin >> m;
    return m;
}
void Lecturer::FromFile()
{
    Lecturer c;
    Student s;
    AssignmentForAll a;
    AssignmentForOne b;

    /********************/
    char currPass[20];
    char currUsername[20];
    char ch;
    cout << "please enter your username" << endl;
    cin.getline(currUsername, 20);
    cin.ignore();
    int swi;
    /********************/
    ifstream fin;
```

```
   fin.open("LecturerData.bin", ios::in | ios::binary);
   fin.read((char *)&c, sizeof(class Lecturer));

   while (!fin.eof())
   {

       if (!strcasecmp(c.userName, currUsername))
       {
           cout << "enter password" << endl;
           cin.getline(currPass, 20);
           cin.ignore();

           if (!strcmp(c.password, currPass))
           {
               cout << "succesful login" << endl;
               cout << endl;

               do
               {
                     cout << "enter 1 assigning assignment to all" <<
endl;
                   cout << "enter 2 for assigning for one" << endl;
                    cout << "enter 3 for deleteing all the individual
assignment u assigned" << endl;
                  cout << "enter 4 for deleteing All the  assignment
u assigned For All" << endl;
                    cout << "enter 5 for removing the saved student
from the system" << endl;
                    cout << "enter 6 for removing the saved teacher
from the system" << endl;

                        cout << "enter 7 for update the individual
assinment" << endl;
                       cout << "enter 8 for update the sectionwise
assignment" << endl;
```

```
                    cout << "enter 9 total no of student or teachers"
<< endl;
                        cout << "For counting the no. of student and
teachers registers in our system" << endl;
                    cout << "enter 10 for exit the program" << endl;
                    cout << "enter your choice" << endl;
                    cin >> swi;
                    cin.ignore();
                    switch (swi)
                    {
                    case 1:

                        a.storeDataInFile();
                        break;
                    case 2:
                        b.storeDataInFile();
                        break;
                    case 3:
                        DeleteDataForOne();
                        break;
                    case 4:
                        DeleteDataForAll();
                        break;
                    case 5:
                        Removestudent r;
                        r.RemoveStudent();
                        break;
                    case 6:
                        r.RemoveTeacher_();
                    case 8:
                        s.deleteSectionassignment(a.getSec());
                        break;
                    case 10:
                        exit(-1);
                    case 7:
```

```
                        s.updateData(GetIdStudent());
                        break;
                    case 11:
                         cout << "For student enter 11 and For teacher
enter 12";
                        int z;
                        cin >> z;
                        Decide(z);
                        break;
                    default:
                        break;
                    }
                      cout << "enter if u want to continue press y||Y
to end enter n||n" << endl;
                    cin >> ch;
                    cin.ignore(2);
                } while (ch == 'Y' || ch == 'y');
            }
        }
        fin.read((char *)&c, sizeof(class Lecturer));
    }

    if (ch == 'n' || ch == 'N')
    {
        cout << "logged out" << endl;
        now();
    }
    fin.close();
}
inline void Lecturer::display() //inline keyword is used
{
    cout << userName << " " << lecturerId << password << endl;
}
void Lecturer::getDataFromFile()
{
```

```
   Lecturer a;
   ifstream fin;
   fin.open("LecturerData.bin", ios::in | ios::binary);
   fin.read((char *)&a, sizeof(class Lecturer));
   while (!fin.eof())
   {
       a.display();
       fin.read((char *)&a, sizeof(class Lecturer));
   }
   fin.close();
}
void Lecturer::storeDataInFile()
{
   LecturerCount++;
   Lecturer a;
   a.getData();
   ofstream fout;
   fout.open("LecturerData.bin", ios::app | ios::binary);
   fout.write((char *)&a, sizeof(class Lecturer));
   fout.close();
   return;
}
//defination for password getter
char *Lecturer::getPassword() { return password; }
//defination for userName getter
char *Lecturer::getUserName() { return userName; }
//defination of member function to get details from the user
void Lecturer::getData()
{
   LecturerCount++;
   cout << "enter UserName" << endl;
   cin.getline(userName, 19);
   cout << "enter password" << endl;
   cin.getline(password, 19);
   cout << "enter lecturer id" << endl;
```

```
    // cin.getline(lecturerId,10);
    cin >> lecturerId;
    cin.ignore();
    return;
}

void now()
{
    Student s;
    Lecturer l;
              cout     <<     "\n*************--UNIVERSITY     MANAGEMENT
SYSTEM--****************\n*************--Made    BY    PARTH,    SAKSHAM,
RAJ--*****************\n\n";
    cout << "Register Student in our system enter 1" << endl;
    cout << "Register Teacher in our system enter 2" << endl;
    cout << "Student Sign in enter 3"
         << endl;
    cout << "Teacher Sign in enter 4"
         << endl;
    cout << "enter 5 for exit"
         << endl;
    int i;
    cin >> i;
    cin.ignore();
    switch (i)
    {
    case 1:
        s.storeDataInFile();
        break;
    case 2:
        l.storeDataInFile();
        break;
    case 3:
        s.FromFile();
        break;
```

```
    case 4:
        l.FromFile();
        break;
    case 5:
        exit(-1);
    default:
        break;
    }
}
int main()
{
    now();
    return 0;
}
```

Results

We have the following outputs from running our program for the university management system:

```
                        UNIVERSITY MANAGEMENT SYSTEM

                        made by parth, raj & saksham

1)Register Student in our system enter 1
2)Register Teacher in our system enter 2
3)Student Sign in enter 3
4)Teacher Sign in enter 4
enter 5 to exit
1
enter the username of the student
Raj

enter the pass word for the student
Chaudhary

enter student section
M3

enter id of the student
2K19/ME/184


...Program finished with exit code 0
Press ENTER to exit console.
```

```
                    UNIVERSITY MANAGEMENT SYSTEM

                    made by parth, raj & saksham

1)Register Student in our system enter 1
2)Register Teacher in our system enter 2
3)Student Sign in enter 3
4)Teacher Sign in enter 4
enter 5 to exit
3
enter your username
Raj

enter your password
Chaudhary

Welcome Student
enter  1 for assignment for all
enter  2 for assignment of you
```

```
*************--UNIVERSITY MANAGEMENT SYSTEM--****************
*************--Made BY PARTH, SAKSHAM, RAJ--******************

Register Student in our system enter 1
Register Teacher in our system enter 2
Student Sign in enter 3
Teacher Sign in enter 4
enter 5 for exit
3
enter your username
Parth

enter your password
Dipendra

Welcome Student
enter  1 for assignment for you
enter  2 for assignment of all
1
do u want to continue
Enter Y||y for continue
 N||n for return to main menu
y


enter  1 for assignment for you
enter  2 for assignment of all
2
Give OOPs paper2
do u want to continue
Enter Y||y for continue
 N||n for return to main menu
```

```
                University Data Management
Register Student in our system enter 1
Register Teacher in our system enter 2
Student Sign in enter 3
Teacher Sign in enter 4
enter 5 for exit
2
enter UserName
Faculty_AP
enter password
engg_phy
enter lecturer id
EP_Fac_01
```

Conclusion

From the following results, we may conclude that we have successfully made a university management system using C++ and implemented it successfully with it’s use being efficient and practical.

Acknowledgements

We would like to express our gratitude to our teacher, Ms. Anjali Bansal, for guiding us through this project and giving us insight on how to make such a university management system

References

1. *www.geeksforgeeks.com*
2. *https://www.javatpoint.com/java-oops-concepts*
3. *Object Oriented Programming With C++ by Balagurusamy, 2008*