|  |
| --- |
| OOP LAB FINAL PROJECT |
| FLEX MANAGEMENT SYSTEM |
| NAME: **MUZAMMIL BIN SOHAIL**  ROLL NO. : **22F-3110**  SECTION: **2A1**  DEGREE: **BS-AI** |

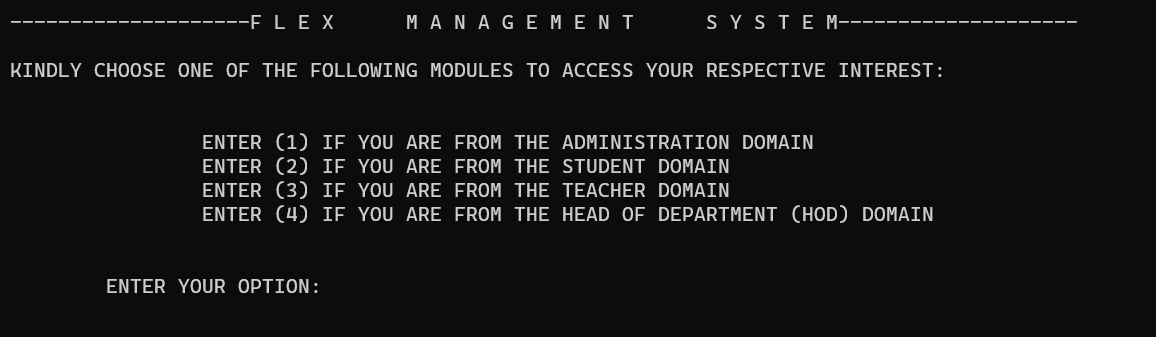
**Contents**

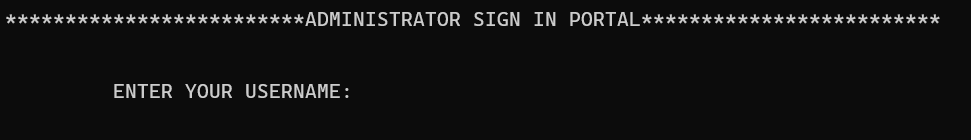
* Some screenshots of the project
* Rough sketch of UML Diagram
* Main Program/Driver Code/Client Code
* Implementation File

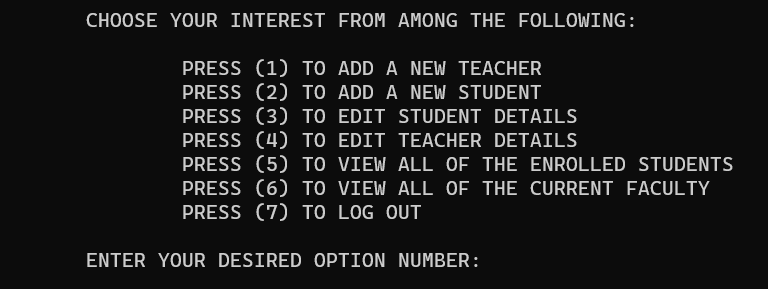
**Key Points:**

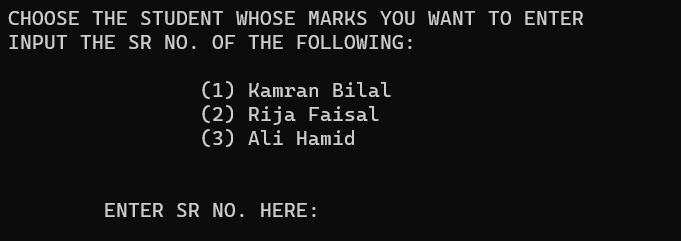
* UML Diagram is yet to be finalised, it would be done shortly before Viva so it is requested to grant some time for that
* Exceptional handling, better user interface and some other features also needs to be improved and everything would be done timely before viva
* The program shows the flow of the program and encompasses all the features including inheritance, workable file handling
* Composition and Aggregation of objects is used
* Overriding has also been used
* Kindly consider the workings of this program and finalise the evaluation at the time of the viva as we would be able to enhance our project. Thank you sir ☺

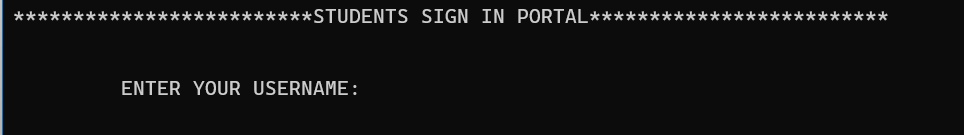
**Some screenshots:**

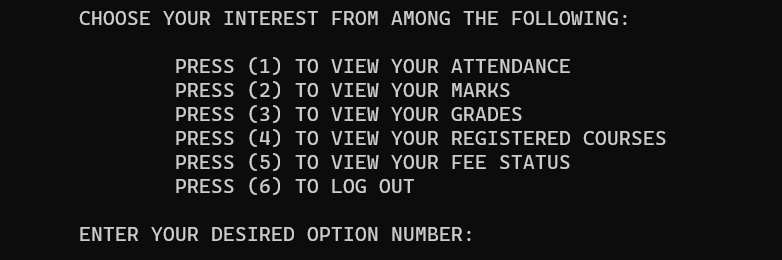
****

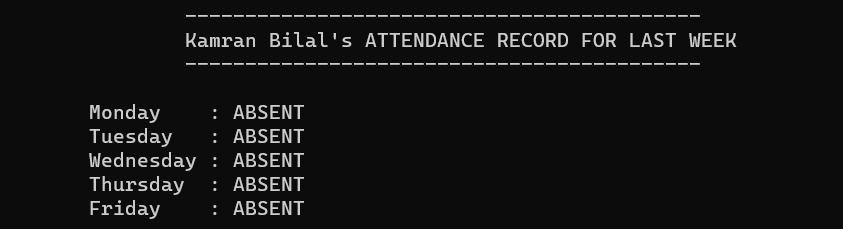
****

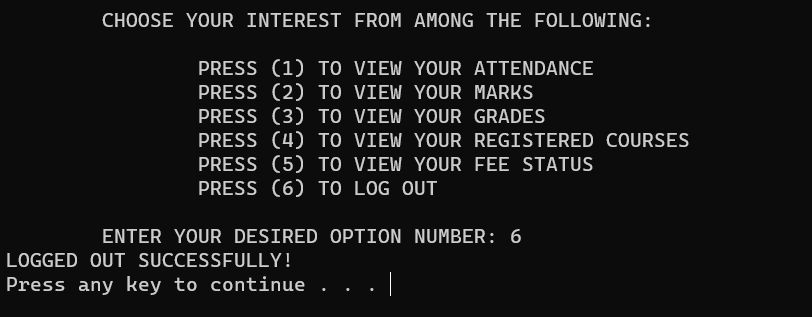
****

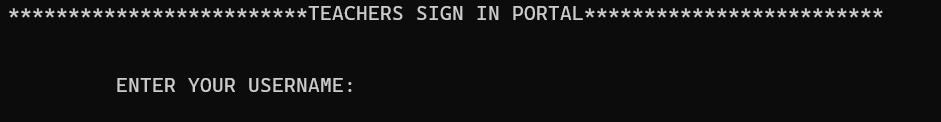
****

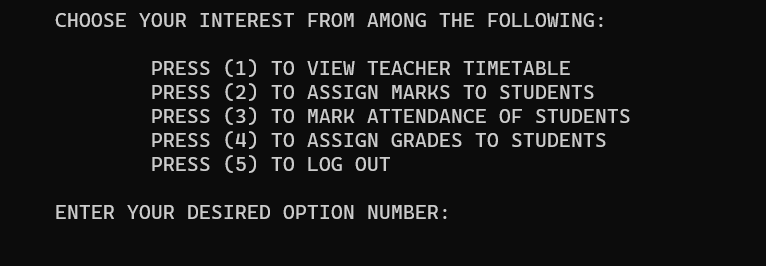
****

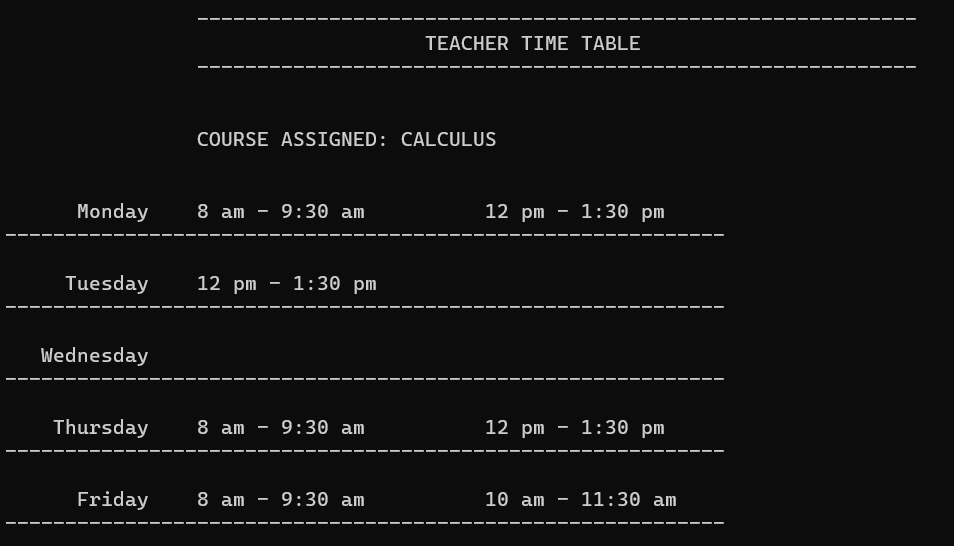
****

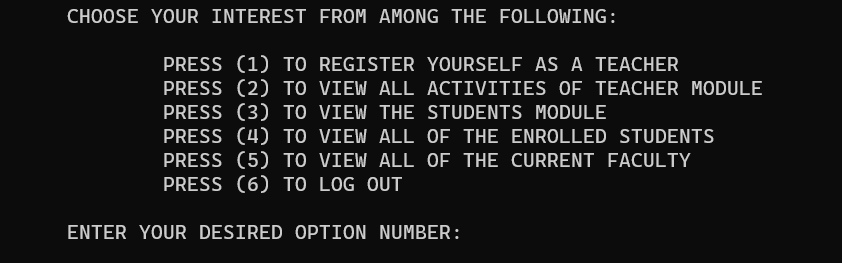
****

****

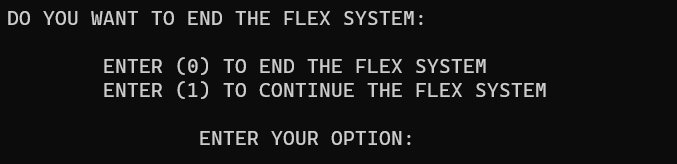
****

****

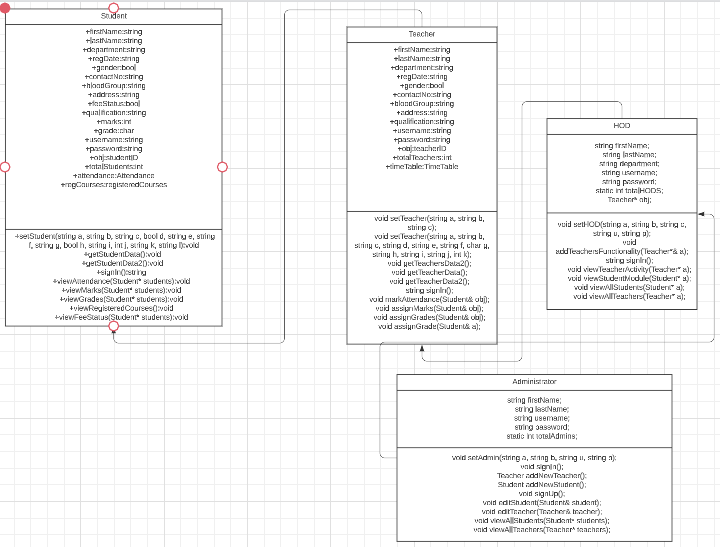
****

****

****

****

**2. Rough Sketch of UML Diagram:**

****

**3. Class File/Header File:**

#include<string>

#include <iostream>

#include<fstream>

#include<iomanip>

#define ADMINISTRATION\_H

using namespace std;

class Student

{

//CLASS ATTENDANCE PART OF THE STUDENT CLASS

class Attendance

{

public:

//FOR 5 WORKING DAYS ATTENDANCE OF THE STUDENT

bool day[5];

//CONSTRUCTOR TO INITIALISE

Attendance()

{

for (int i = 0; i < 5; i++)

day[i] = 0;

}

void setAttendance(bool a, bool b, bool c, bool d, bool e);

void setAttendance(Student& a);

void displayAttendance(Student& a);

};

//CLASS OF STUDENT ID

class StudentID

{

public:

//22

int year;

//F

char city;

//3110

int rollNo;

//CONSTURCTOR

StudentID()

{

year = 00;

city = '\0';

rollNo = 0000;

}

void setID(int a, char b, int c);

void displayStudentID();

};

//CLASS OF REGISTERED COURSES

class registeredCourses

{

public:

int noOfcourses = 3;

string\* courseArr = new string[noOfcourses];

registeredCourses()

{

for (int i = 0; i < noOfcourses; i++)

{

courseArr[i] = '\0';

}

}

void editCourses();

void viewCourse();

void assignGrades(char temp);

void showGrades();

};

public:

//STUDENT CLASS MEMBERS

string firstName;

string lastName;

string department;

string regDate;

bool gender;

string contactNo;

string bloodGroup;

string address;

bool feeStatus;

string qualification;

int marks;

char grade;

string username;

string password;

StudentID obj;

static int totalStudents;

Attendance attendance;

registeredCourses regCourses;

//STUDENT CONSTRUCTOR TO INITIALISE

Student()

{

firstName = "";

lastName = "";

department = "";

gender = 0;

contactNo = "";

bloodGroup = "";

address = "";

feeStatus = 0;

qualification = "";

marks = 0;

username = "";

regDate = "";

grade = '\0';

password = "";

}

void setStudent(string a, string b, string c, bool d, string e, string f, string g, bool h, string i, int j, string k, string l);

void getStudentData();

void getStudentData2();

string signIn();

void viewAttendance(Student\* students);

void viewMarks(Student\* students);

void viewGrades(Student\* students);

void viewRegisteredCourses();

void viewFeeStatus(Student\* students);

};

//CLASS OF TEACHER INHERITTING STUDENT CLASS

class Teacher : public Student

{

//CLASS OF TEACHERS TIMETABLE

class TimeTable

{

public:

string courseTitle;

//5 DAYS IN A WEEK AND 3 SLOTS TO CHOOSE FROM

bool timeTable[5][3];

//CONSTRUCTOR TO INITIALISE

TimeTable()

{

courseTitle = "";

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 3; j++)

{

timeTable[i][j] = 0;

}

}

}

void setTimeTable(string a, bool each[][3]);

void setTimeTable(Teacher& newTeacher);

void displayTimeTable();

};

//CLASS OF TEACHER ID

class teacherID

{

public:

//CS

char department;

//001

int teacherNo;

//CONSTRUCTOR TO INITIALISE

teacherID()

{

department = '\0';

teacherNo = 0000;

}

void setteacherID(string a, int& b);

void getTeacherID();

};

public:

//MEMBERS OF TEACHER

string firstName;

string lastName;

string department;

string username;

string password;

string regDate;

bool gender;

string contactNo;

string qualification;

string address;

int salary;

teacherID obj;

static int totalTeachers;

TimeTable timeTable;

//CONSTRUCTOR OF TEACHER TO INITIALISE

Teacher()

{

firstName = "";

lastName = "";

department = "";

username = "";

password = ""; //pw check?

regDate = ""; //date datatype? i think no need

gender = 0;

contactNo = "";

qualification = "";

address = "";

salary = 0;

}

void setTeacher(string a, string b, string c);

void setTeacher(string a, string b, string c, string d, string e, string f, char g, string h, string i, string j, int k);

void getTeachersData2();

void getTeacherData();

void getTeacherData2();

string signIn();

void markAttendance(Student& obj);

void assignMarks(Student& obj);

void assignGrades(Student& obj);

void assignGrade(Student& a);

};

class HOD : public Teacher, public Student

{

public:

string firstName;

string lastName;

string department;

string username;

string password;

static int totalHODS;

Teacher\* obj;

HOD()

{

firstName = "";

lastName = "";

department = "";

username = "";

password = "";

obj = NULL;

}

void setHOD(string a, string b, string c, string u, string p);

void addTeachersFunctionality(Teacher\*& a);

string signIn();

void viewTeacherActivity(Teacher\* a);

void viewStudentModule(Student\* a);

void viewAllStudents(Student\* a);

void viewAllTeachers(Teacher\* a);

};

class Administrator : public HOD, public Student, public Teacher

{

public:

string firstName;

string lastName;

string username;

string password;

static int totalAdmins;

Administrator()

{

username = "";

password = "";

totalAdmins++;

}

void setAdmin(string a, string b, string u, string p);

void signIn();

Teacher addNewTeacher();

Student addNewStudent();

void signUp();

void editStudent(Student& student);

void editTeacher(Teacher& teacher);

void viewAllStudents(Student\* students);

void viewAllTeachers(Teacher\* teachers);

};

**4. Main Program/Source Code/Driver Code:**

#include <iostream>

#include<string>

#include<cstdlib>

#include<iomanip>

#include<fstream>

#include<ctime>

#include"classes.h"

//RELEVANT HEADER FILES AND LIBRARIES ATTACHED

using namespace std;

//MAX CAPACITY OF EACH TYPE OF MODULE

const int maxStudents = 500;

const int maxTeachers = 500;

const int maxHODS = 500;

const int maxAdmins = 500;

//STATIC DECLARATION OF CURRENT COUNT OF EACH MODULE

int Student::totalStudents = 0;

int Teacher::totalTeachers = 0;

int HOD::totalHODS = 5;

int Administrator::totalAdmins = 3;

int main()

{

//FOR GENERATING NEW RANDOM NUMBER EACH TIME

srand(time(0));

//DYNAMIC ALLOCATION OF EACH TYPE

Teacher\* teachers = new Teacher[maxTeachers];

Student\* students = new Student[maxStudents];

Administrator\* admins = new Administrator[maxAdmins];

HOD\* hods = new HOD[maxHODS];

//TEACHERS INITIALISATION THROUGH PARAMETRISED CONSTUCTOR

//teachers[0].setTeacher("Mustafa", "Kamal", "Computer Science", "mustafakamal18", "abcdef", "17.02.2018", 1, "0300-4428828", "MS - Computer Science", "114 Faisal Block, Iqbal Town, Lahore", 120000);

teachers[0].setTeacher("Mustafa", "Kamal", "Computer Science", "123", "123", "17.02.2018", 1, "0300-4428828", "MS - Computer Science", "114 Faisal Block, Iqbal Town, Lahore", 120000);

teachers[1].setTeacher("Ikram", "Jafar", "Business", "ikramjafar24", "ihjffld", "04.03.2020", 1, "0321-5421138", "M - Business Administration", "3 Mirza Block, Johar Town, Faisalabad", 140000);

teachers[2].setTeacher("Fatima", "Tameer", "Social Sciences", "fatimatameer22", "dasdssa", "01.11.2017", 0, "0343-3432418", "MS - Physics", "43 Chenab Block, Ilyas Town, Karachi", 95000);

bool a[5][3] = { {1,0,1}, {0,0,1}, {0,0,0}, {1,0,1}, {1,1,0} };

teachers[0].timeTable.setTimeTable("CALCULUS", a);

bool b[5][3] = { {0,0,1}, {1,0,1}, {0,1,0}, {1,0,0}, {0,1,0} };

teachers[1].timeTable.setTimeTable("ICT", b);

bool c[5][3] = { {1,0,0}, {0,0,0}, {1,1,0}, {1,0,0}, {1,1,1} };

teachers[2].timeTable.setTimeTable("PF", c);

fstream myOutput;

myOutput.open("teachers.txt");

if (!myOutput.is\_open())

cout << "File not found";

for (int i = 0; i < 3; i++)

{

myOutput << teachers[i].firstName<<endl;

myOutput << teachers[i].lastName << endl;

myOutput << teachers[i].department << endl;

myOutput << teachers[i].username << endl;

myOutput << teachers[i].password << endl;

myOutput << teachers[i].regDate << endl;

myOutput << teachers[i].gender << endl;

myOutput << teachers[i].contactNo << endl;

myOutput << teachers[i].qualification << endl;

myOutput << teachers[i].address << endl;

myOutput << teachers[i].salary << endl;

}

myOutput.close();

//STUDENTS INITIALISATION THROUGH PARAMETRISED CONSTUCTOR

//students[0].setStudent("Kamran", "Bilal", "Computer Science", 1, "0300-1643244", "B - Positive", "143 Ravi Block, Rizauddin Town, Lahore", 1, "A - Levels", 976, "kamranbilal12", "kvjmsd");

students[0].setStudent("Kamran", "Bilal", "Computer Science", 1, "0300-1643244", "B - Positive", "143 Ravi Block, Rizauddin Town, Lahore", 1, "A - Levels", 976, "123", "123");

students[1].setStudent("Rija", "Faisal", "Business", 1, "0315-5293281", "AB - Positive", "122 Bajnur Block, Liaqat Town, Alipur", 1, "A - Levels", 675, "rijafaisal15", "0dsdaaa");

students[2].setStudent("Ali", "Hamid", "Software Engineering", 1, "0344-5453121", "O - Negative", "98 Akbar Block, Shifa Town, Jehlum", 0, "FSC", 1011, "alihamid44", "aasfffs");

myOutput.open("students.txt");

if (!myOutput.is\_open())

cout << "File not found";

for (int i = 0; i < 3; i++)

{

myOutput << students[i].firstName << endl;

myOutput << students[i].lastName << endl;

myOutput << students[i].department << endl;

myOutput << students[i].gender << endl;

myOutput << students[i].contactNo << endl;

myOutput << students[i].bloodGroup << endl;

myOutput << students[i].address << endl;

myOutput << students[i].feeStatus << endl;

myOutput << students[i].qualification << endl;

myOutput << students[i].marks << endl;

myOutput << students[i].username << endl;

myOutput << students[i].password << endl;

}

myOutput.close();

//hods[0].setHOD("Hassan", "Ali", "Software Engineering", "hassanali232", "kgffjk");

hods[0].setHOD("Hassan", "Ali", "Software Engineering", "123", "123");

hods[1].setHOD("Ali", "Ahmed", "Business", "aliahmed44", "asdfjkl");

hods[2].setHOD("Javaid", "Rasheed", "Social Sciences", "javaidrasheed12", "asdfjkl");

hods[3].setHOD("Aleena", "Bilal", "Electrical Engineering", "aleenabilal89", "asdfjkl");

hods[4].setHOD("Iman", "Latif", "Computer Science", "imanlatif99", "asdfjkl");

myOutput.open("hods.txt");

if (!myOutput.is\_open())

cout << "File not found";

for (int i = 0; i < 3; i++)

{

myOutput << hods[i].firstName << endl;

myOutput << hods[i].lastName << endl;;

myOutput << hods[i].department << endl;;

myOutput << hods[i].username << endl;;

myOutput << hods[i].password << endl;;

}

myOutput.close();

//admins[0].setAdmin("Nabeela", "Ashraf", "nashraf", "nabeela36");

admins[0].setAdmin("Nabeela", "Ashraf", "123", "123");

admins[1].setAdmin("Nabeel", "Hamid", "hamdidnabeel", "nabee369");

myOutput.open("admins.txt");

if (!myOutput.is\_open())

cout << "File not found";

for (int i = 0; i < 3; i++)

{

myOutput << admins[i].firstName << endl;

myOutput << admins[i].lastName << endl;;

myOutput << admins[i].username << endl;;

myOutput << admins[i].password << endl;;

}

myOutput.close();

bool end = true;

//FLOW OF PROGRAM NOT ENDED UNTIL CHOSEN

while (end != false)

{

cout << "--------------------F L E X M A N A G E M E N T S Y S T E M--------------------" << endl << endl;

int option;

cout << "KINDLY CHOOSE ONE OF THE FOLLOWING MODULES TO ACCESS YOUR RESPECTIVE INTEREST: ";

cout << endl << endl << endl;

cout << "\t\tENTER (1) IF YOU ARE FROM THE ADMINISTRATION DOMAIN" << endl;

cout << "\t\tENTER (2) IF YOU ARE FROM THE STUDENT DOMAIN" << endl;

cout << "\t\tENTER (3) IF YOU ARE FROM THE TEACHER DOMAIN" << endl;

cout << "\t\tENTER (4) IF YOU ARE FROM THE HEAD OF DEPARTMENT (HOD) DOMAIN" << endl;

cout << endl << endl;

cout << "\tENTER YOUR OPTION: ";

cin >> option;

cout << endl;

//ADMINISTRATION MODULE

if (option == 1)

{

//DISPLAYS THE ADMINISTRATION SIGN IN MENU

admins->signIn();

system("cls");

int subOption = 0;

bool logOut = false;

while (logOut != true)

{

cout << "\tCHOOSE YOUR INTEREST FROM AMONG THE FOLLOWING: \n\n\t\tPRESS (1) TO ADD A NEW TEACHER\n\t\tPRESS (2) TO ADD A NEW STUDENT\n\t\tPRESS (3) TO EDIT STUDENT DETAILS\n\t\tPRESS (4) TO EDIT TEACHER DETAILS\n\t\tPRESS (5) TO VIEW ALL OF THE ENROLLED STUDENTS\n\t\tPRESS (6) TO VIEW ALL OF THE CURRENT FACULTY\n\t\tPRESS (7) TO LOG OUT\n\n\tENTER YOUR DESIRED OPTION NUMBER: ";

cin >> subOption;

//ADD NEW TEACHER

if (subOption == 1)

{

system("cls");

teachers[Teacher::totalTeachers-1] = admins->addNewTeacher();

system("pause");

cout << "TEACHER ADDED SUCCESSFULLY!" << endl;

system("cls");

}

//ADD NEW STUDENT

else if (subOption == 2)

{

system("cls");

students[Student::totalStudents-1]=admins->addNewStudent();

system("pause");

cout << "STUDENT ADDED SUCCESSFULLY!" << endl;

system("cls");

}

//EDIT STUDENT DETAILS

else if (subOption == 3)

{

int thisOption;

system("cls");

cout << "CHOOSE THE STUDENT WHOSE MARKS YOU WANT TO ENTER \n";

cout << "INPUT THE SR NO. OF THE FOLLOWING: \n\n";

for (int i = 0; i < Student::totalStudents; i++)

{

cout << "\t\t(" << i + 1 << ") " << students[i].firstName << " " << students[i].lastName << "\n";

}

cout << "\n\n\tENTER SR NO. HERE: ";

cin >> thisOption;

system("cls");

admins->editStudent(students[thisOption-1]);

}

//EDIT TEACHER DETAILS

else if (subOption == 4)

{

system("cls");

int thisOption;

cout << "CHOOSE THE STUDENT WHOSE MARKS YOU WANT TO ENTER \n";

cout << "INPUT THE SR NO. OF THE FOLLOWING: \n\n";

for (int i = 0; i < Student::totalStudents; i++)

{

cout << "\t\t(" << i + 1 << ") " << students[i].firstName << " " << students[i].lastName << "\n";

}

cout << "\n\n\tENTER SR NO. HERE: ";

cin >> thisOption;

system("cls");

admins->editTeacher(teachers[thisOption-1]);

}

//VIEW ALL STUDENTS

else if (subOption == 5)

{

system("cls");

admins->viewAllStudents(students);

system("pause");

system("cls");

}

//VIEW ALL TEACHERS

else if (subOption == 6)

{

system("cls");

admins->viewAllTeachers(teachers);

system("pause");

system("cls");

}

else if (subOption==7)

{

logOut = true;

cout << "LOGGED OUT SUCCESSFULLY!"<<endl;

system("pause");

}

}

}

//STUDENT MODULE

else if (option == 2)

{

string tempUser;

int thisStuInd;

//DISPLAYS THE STUDENT SIGN IN MENU

tempUser=students->signIn();

for (int i = 0;i < Student::totalStudents;i++)

{

if (students[i].username == tempUser)

{

thisStuInd = i;

}

}

system("cls");

int subOption2 = 0;

bool logOut = false;

while (logOut != true)

{

cout << "\tCHOOSE YOUR INTEREST FROM AMONG THE FOLLOWING: \n\n\t\tPRESS (1) TO VIEW YOUR ATTENDANCE\n\t\tPRESS (2) TO VIEW YOUR MARKS\n\t\tPRESS (3) TO VIEW YOUR GRADES\n\t\tPRESS (4) TO VIEW YOUR REGISTERED COURSES\n\t\tPRESS (5) TO VIEW YOUR FEE STATUS\n\t\tPRESS (6) TO LOG OUT\n\n\tENTER YOUR DESIRED OPTION NUMBER: ";

cin >> subOption2;

//VIEW ATTENDANCE

if (subOption2 == 1)

{

system("cls");

students->viewAttendance(&students[thisStuInd]);

system("pause");

system("cls");

}

//VIEW MARKS

else if (subOption2 == 2)

{

system("cls");

students->viewMarks(&students[thisStuInd]);

system("pause");

system("cls");

}

//VIEW GRADES

else if (subOption2 == 3)

{

system("cls");

students->viewGrades(&students[thisStuInd]);

system("pause");

system("cls");

}

//VIEW REGISTERED COURSES

else if (subOption2 == 4)

{

system("cls");

students[thisStuInd].viewRegisteredCourses();

system("pause");

system("cls");

}

//VIEW FEE STATUS

else if (subOption2 == 5)

{

system("cls");

students[thisStuInd].viewFeeStatus(&students[thisStuInd]);

system("pause");

system("cls");

}

else if (subOption2 == 6)

{

logOut = true;

cout << "LOGGED OUT SUCCESSFULLY!" << endl;

system("pause");

}

}

}

//TEACHER MODULE

else if (option == 3)

{

//DISPLAYS THE TEACHER SIGN IN MENU

string tempUser2;

int thisTeaInd;

tempUser2 = teachers->signIn();

for (int i = 0;i < Teacher::totalTeachers;i++)

{

if (teachers[i].username == tempUser2)

{

thisTeaInd = i;

}

}

system("cls");

int subOption3= 0;

bool logOut = false;

while (logOut != true)

{

cout << "\tCHOOSE YOUR INTEREST FROM AMONG THE FOLLOWING: \n\n\t\tPRESS (1) TO VIEW TEACHER TIMETABLE\n\t\tPRESS (2) TO ASSIGN MARKS TO STUDENTS\n\t\tPRESS (3) TO MARK ATTENDANCE OF STUDENTS\n\t\tPRESS (4) TO ASSIGN GRADES TO STUDENTS\n\t\tPRESS (5) TO LOG OUT\n\n\tENTER YOUR DESIRED OPTION NUMBER: ";

cin >> subOption3;

//DISPLAYS TEACHER TIMETABLE

if (subOption3 == 1)

{

system("cls");

//teachers[0].timeTable.setTimeTable(teachers[0]);

teachers[thisTeaInd].timeTable.displayTimeTable();

system("pause");

system("cls");

}

//ASSIGNS MARKS

else if (subOption3 == 2)

{

int thisOption;

system("cls");

cout << "CHOOSE THE STUDENT WHOSE MARKS YOU WANT TO ENTER \n";

cout << "INPUT THE SR NO. OF THE FOLLOWING: \n\n";

for (int i = 0; i < Student::totalStudents; i++)

{

cout << "\t\t(" << i + 1 << ") " << students[i].firstName << " " << students[i].lastName << "\n";

}

cout << "\n\n\tENTER SR NO. HERE: ";

cin >> thisOption;

system("cls");

teachers->assignMarks(students[thisOption - 1]);

//cout << students[thisOption - 1].marks;

system("pause");

system("cls");

}

//MARKING ATTENDANCE

else if (subOption3 == 3)

{

int thisOption;

system("cls");

cout << "CHOOSE THE STUDENT WHOSE ATTENDANCE YOU WANT TO MARK \n";

cout << "INPUT THE SR NO. OF THE FOLLOWING: \n\n";

for (int i = 0; i < Student::totalStudents; i++)

{

cout << "\t\t(" << i + 1 << ") " << students[i].firstName << " " << students[i].lastName << "\n";

}

cout << "\n\n\tENTER SR NO. HERE: ";

cin >> thisOption;

system("cls");

teachers->markAttendance(students[thisOption - 1]);

system("cls");

students[thisOption - 1].attendance.displayAttendance(students[thisOption - 1]);

system("pause");

system("cls");

}

//ASSIGN GRADES

else if (subOption3 == 4)

{

int thisOption;

system("cls");

cout << "CHOOSE THE STUDENT WHOM YOU WANT TO ASSSIGN THE GRADE \n";

cout << "INPUT THE SR NO. OF THE FOLLOWING: \n\n";

for (int i = 0; i < Student::totalStudents; i++)

{

cout << "\t\t(" << i + 1 << ") " << students[i].firstName << " " << students[i].lastName << "\n";

}

cout << "\n\n\tENTER SR NO. HERE: ";

cin >> thisOption;

system("cls");

teachers->assignGrade(students[thisOption - 1]);

system("pause");

system("cls");

//cout<<"GRADE: "<<students[thisOption - 1].grade;

}

else if (subOption3 == 5)

{

logOut = true;

cout << "LOGGED OUT SUCCESSFULLY!" << endl;

system("pause");

}

}

}

//HOD MODULE

else if (option == 4)

{

//DISPLAYS THE HOD SIGN IN MENU

string tempUser3;

int thishodInd = 0;

tempUser3 = hods->signIn();

for (int i = 0;i < HOD::totalHODS;i++)

{

if (hods[i].username == tempUser3)

{

thishodInd = i;

}

}

system("cls");

int subOption4 = 0;

bool logOut = false;

while (logOut != true)

{

cout << "\tCHOOSE YOUR INTEREST FROM AMONG THE FOLLOWING: \n\n\t\tPRESS (1) TO REGISTER YOURSELF AS A TEACHER\n\t\tPRESS (2) TO VIEW ALL ACTIVITIES OF TEACHER MODULE\n\t\tPRESS (3) TO VIEW THE STUDENTS MODULE\n\t\tPRESS (4) TO VIEW ALL OF THE ENROLLED STUDENTS\n\t\tPRESS (5) TO VIEW ALL OF THE CURRENT FACULTY\n\t\tPRESS (6) TO LOG OUT\n\n\tENTER YOUR DESIRED OPTION NUMBER: ";

cin >> subOption4;

//HOD AS TEACHER

if (subOption4 == 1)

{

hods[thishodInd].addTeachersFunctionality(teachers);

teachers[Teacher::totalTeachers].setTeacher(hods->firstName, hods->lastName, hods->department, hods->username, hods->password, "", '/0', "", "", "", 0);

system("cls");

cout << "\n\tYOUR TEACHER PROFILE HAS BEEN SET UP \n\n\t";

teachers[Teacher::totalTeachers - 1].obj.getTeacherID();

system("pause");

system("cls");

//cout << Teacher::totalTeachers;

}

//VIEW ALL ACTIVITIES OF TEACHER MODULE

else if (subOption4 == 2)

{

system("cls");

hods->viewTeacherActivity(teachers);

system("pause");

system("cls");

}

//ACCESS TO STUDENT MODULE

else if (subOption4 == 3)

{

system("cls");

hods->viewStudentModule(students);

system("pause");

system("cls");

}

//VIEW ALL STUDENTS

else if (subOption4 == 4)

{

system("cls");

admins->viewAllStudents(students);

system("pause");

system("cls");

}

//VIEW ALL TEACHERS

else if (subOption4 == 5)

{

system("cls");

admins->viewAllTeachers(teachers);

system("pause");

system("cls");

}

else if (subOption4 == 6)

{

logOut = true;

cout << "LOGGED OUT SUCCESSFULLY!" << endl;

system("pause");

}

}

}

else

{

system("cls");

}

system("cls");

myOutput.open("teachers.txt");

if (!myOutput.is\_open())

cout << "File not found";

for (int i = 0; i < 3; i++)

{

myOutput << teachers[i].firstName << endl;

myOutput << teachers[i].lastName << endl;

myOutput << teachers[i].department << endl;

myOutput << teachers[i].username << endl;

myOutput << teachers[i].password << endl;

myOutput << teachers[i].regDate << endl;

myOutput << teachers[i].gender << endl;

myOutput << teachers[i].contactNo << endl;

myOutput << teachers[i].qualification << endl;

myOutput << teachers[i].address << endl;

myOutput << teachers[i].salary << endl;

}

myOutput.close();

myOutput.open("students.txt");

if (!myOutput.is\_open())

cout << "File not found";

for (int i = 0; i < 3; i++)

{

myOutput << students[i].firstName << endl;

myOutput << students[i].lastName << endl;

myOutput << students[i].department << endl;

myOutput << students[i].gender << endl;

myOutput << students[i].contactNo << endl;

myOutput << students[i].bloodGroup << endl;

myOutput << students[i].address << endl;

myOutput << students[i].feeStatus << endl;

myOutput << students[i].qualification << endl;

myOutput << students[i].marks << endl;

myOutput << students[i].username << endl;

myOutput << students[i].password << endl;

}

myOutput.close();

myOutput.open("hods.txt");

if (!myOutput.is\_open())

cout << "File not found";

for (int i = 0; i < 3; i++)

{

myOutput << hods[i].firstName << endl;

myOutput << hods[i].lastName << endl;;

myOutput << hods[i].department << endl;;

myOutput << hods[i].username << endl;;

myOutput << hods[i].password << endl;;

}

myOutput.close();

myOutput.open("admins.txt");

if (!myOutput.is\_open())

cout << "File not found";

for (int i = 0; i < 3; i++)

{

myOutput << admins[i].firstName << endl;

myOutput << admins[i].lastName << endl;;

myOutput << admins[i].username << endl;;

myOutput << admins[i].password << endl;;

}

myOutput.close();

cout << "DO YOU WANT TO END THE FLEX SYSTEM: " << endl << endl;

cout << "\tENTER (0) TO END THE FLEX SYSTEM" << endl;

cout << "\tENTER (1) TO CONTINUE THE FLEX SYSTEM";

cout << endl << endl;

cout << "\t\tENTER YOUR OPTION: ";

cin >> end;

system("cls");

}

return 0;

}

**5. Implementation File:**

//RELEVANT HEADER FILES AND LIBRARIES ATTACHED

#include<iostream>

#include"classes.h"

#include<string>

#include<fstream>

#include<iomanip>

#include<string>

using namespace std;

//SETTER OF STUDENT

void Student::setStudent(string a, string b, string c, bool d, string e, string f, string g, bool h, string i, int j, string k, string l)

{

firstName = a;

lastName = b;

department = c;

gender = d;

contactNo = e;

bloodGroup = f;

address = g;

feeStatus = h;

qualification = i;

marks = j;

username = k;

password = l;

obj.setID(23, 'F', totalStudents + 3000);

}

//DISPLAYS STUDENT

void Student::getStudentData()

{

cout << "\n\t\t\t STUDENT USERNAME: " << username << "\n\t\t\t STUDENT PASSWORD: " << password;

cout << "\n\n\tNAME: " << firstName << " " << lastName << "\n\tDEPARTMENT: " << department << "\n\tGENDER: ";

if (gender == 0)

cout << "Female";

else

cout << "Male";

cout << "\n\tCONTACT NO: " << contactNo << "\n\tBLOOD GROUP: " << bloodGroup << "\n\tADDRESS: " << address << "\n\tFEE STATUS: ";

if (feeStatus == 0)

cout << "NOT PAID YET";

else

cout << "PAID";

cout << "\n\tQUALIFICATION: " << qualification << "\n\tMARKS: " << marks << "\n";

}

//DISPLAYS ALL STUDENTS WITH BASIC INFO

void Student::getStudentData2()

{

cout << setw(13) << obj.year << obj.city << "-" << obj.rollNo;

cout << setw(20) << firstName;

cout << setw(20) << lastName;

cout << setw(20) << username;

cout << setw(20) << password;

cout << setw(25) << department;

cout << setw(20) << contactNo;

if (feeStatus == 0)

cout << setw(19) << "NOT PAID YET";

else

cout << setw(19) << "PAID";

cout << endl;

}

//DISPLAYS SIGN IN MENU

string Student::signIn()

{

system("cls");

fstream myInput;

string uCheck, pCheck;

bool uFlag = 0, pFlag = 0;

bool repeat = true;

while (repeat != false)

{

system("cls");

myInput.open("students.txt");

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*STUDENTS SIGN IN PORTAL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout << endl << endl << endl;

cout << "\t ENTER YOUR USERNAME: ";

cin >> username;

cout << "\t ENTER YOUR PASSWORD: ";

cin >> password;

if (!(myInput.is\_open()))

cout << "FILE CANNOT BE OPENED! ";

else

{

while (!myInput.eof())

{

getline(myInput, uCheck);

if (username == uCheck)

{

uFlag = true;

break;

}

else

uFlag = 0;

}

while (!myInput.eof())

{

getline(myInput, pCheck);

if (password == pCheck)

{

pFlag = true;

break;

}

else

pFlag = 0;

}

if (pFlag == 1 && uFlag == 1)

{

repeat = false;

cout << "\n\tLOGGED IN SUCCESSFULLY! :)" << endl;

system("pause");

return username;

}

else

{

cout << endl;

cout << "\n\tINVALID CREDENTIALS, ENTER AGAIN! " << endl;

system("pause");

myInput.close();

repeat = true;

}

}

}

system("pause");

}

//VIEWS ATTENDANCE OF STUDENT

void Student::viewAttendance(Student\* students)

{

students->attendance.displayAttendance(students[0]);

cout << endl;

}

//VIEWS MARKS OF STUDENT

void Student::viewMarks(Student\* students)

{

cout << "MARKS: ";

cout << students[0].marks << endl;

}

//VIEWS GRADES OF STUDENT

void Student::viewGrades(Student\* students)

{

cout << "GRADE: ";

cout << students[0].grade << endl;

}

//VIEWS REGISTERED COURSES OF STUDENT

void Student::viewRegisteredCourses()

{

system("cls");

regCourses.viewCourse();

}

//VIEWS FEE STATUS OF STUDENT

void Student::viewFeeStatus(Student\* students)

{

cout << "FEE STATUS: ";

if (students[0].feeStatus == 1)

cout << "PAID";

else

cout << "NOT PAID";

cout << endl;

}

void Student::registeredCourses::editCourses()

{

for (int i = 0; i < noOfcourses; i++)

{

cout << "ENTER THE NAME OF THE COURSE " << i + 1 << ": ";

cin >> courseArr[i];

}

}

void Student::registeredCourses::viewCourse()

{

for (int i = 0; i < noOfcourses; i++)

{

cout << "NAME OF THE COURSE " << i + 1 << ": " << courseArr[i] << endl;

}

}

void Student::registeredCourses::assignGrades(char temp)

{

for (int i = 0; i < noOfcourses; i++)

{

cout << "ENTER THE GRADE OF THE " << i + 1 << " COURSE: "; courseArr[i] = temp;

}

}

void Student::registeredCourses::showGrades()

{

for (int i = 0; i < noOfcourses; i++)

{

cout << "GRADE OF THE " << i + 1 << " COURSE IS: " << courseArr[i] << endl;

}

}

//SETTER OF STUDENT ID

void Student::StudentID::setID(int a, char b, int c)

{

year = a;

city = b;

rollNo = c;

Student::totalStudents++;

}

//DISPLAY STUDENT ID

void Student::StudentID::displayStudentID()

{

cout << "YOUR STUDENT ID IS: ";

cout << year << city << "-" << rollNo << "\n";

}

//ATTENDANCE SETTER

void Student::Attendance::setAttendance(bool a, bool b, bool c, bool d, bool e)

{

day[0] = a;

day[1] = b;

day[2] = c;

day[3] = d;

day[4] = e;

}

//SET ATTENDANCE BY INPUT

void Student::Attendance::setAttendance(Student& a)

{

string days[5] = { "Monday", "Tuesday", "Wednesday", "Thursday", "Friday" };

for (int i = 0; i < 5; i++)

{

cout << "\tEnter (1) if the " << a.firstName << " " << a.lastName << " was present on " << days[i] << ", otherwise enter (0): ";

cin >> a.attendance.day[i];

}

cout << "\n" << endl;

}

//DISPLAY ATTENDANCE

void Student::Attendance::displayAttendance(Student& a)

{

string week[5] = { "Monday", "Tuesday", "Wednesday", "Thursday", "Friday" };

cout << "\n\t\t-------------------------------------------";

cout << "\n\t\t" << a.firstName << " " << a.lastName << "'s ATTENDANCE RECORD FOR LAST WEEK";

cout << "\n\t\t-------------------------------------------\n\n";

for (int i = 0; i < 5; i++)

{

cout << "\t" << setw(10) << left << week[i] << ": ";

if (day[i] == 0)

cout << setw(10) << "ABSENT";

else

cout << setw(10) << "PRESENT";

cout << endl;

}

}

//SETTER OF TIMETABLE

void Teacher::TimeTable::setTimeTable(string a, bool each[][3])

{

courseTitle = a;

for (int i = 0; i < 5; i++)

for (int j = 0; j < 3; j++)

timeTable[i][j] = each[i][j];

}

//SETTER OF TIMETABLE BY INPUTTING

void Teacher::TimeTable::setTimeTable(Teacher& newTeacher)

{

cout << "\n\tENTER THE COURSE TITLE OFFERED TO THE TEACHER: ";

cin >> newTeacher.timeTable.courseTitle;

cout << "\n\t\tSELECT THE WORKING DAYS FOR THE TEACHER: " << endl;

for (int i = 0; i < 5; i++)

{

system("cls");

int thisOption = 0;

string week[5] = { "Monday", "Tuesday", "Wednesday", "Thursday", "Friday" };

cout << "\n\tENTER (1) IF YOU WANT TO APPOINT LECTURE FOR THE TEACHER ON " << week[i] << ", OTHERWISE ENTER (0) : ";

cin >> thisOption;

if (thisOption == 1)

{

for (int j = 0; j < 3; j++)

{

system("cls");

cout << "\n\t\tSELECT THE TIME SLOTS FOR THE TEACHER ON : " << week[i] << endl;

string times[3] = { "8 am - 9:30 am", "10 am - 11:30 am", "12 pm - 1:30 pm" };

cout << "\n\tENTER (1) IF YOU WANT TO APPOINT TIME SLOT TO THE TEACHER ON " << times[j] << ", OTHERWISE ENTER (0) : ";

cin >> newTeacher.timeTable.timeTable[i][j];

}

}

}

}

//DISPLAYS TIMETABLE

void Teacher::TimeTable::displayTimeTable()

{

system("cls");

cout << "\n\t\t------------------------------------------------------------" << endl;

cout << "\t\t TEACHER TIME TABLE " << endl;

cout << "\t\t------------------------------------------------------------" << endl << endl;

cout << "\n\t\tCOURSE ASSIGNED: " << courseTitle << "\n\n\n";

string week[5] = { "Monday", "Tuesday", "Wednesday", "Thursday", "Friday" };

string time[3] = { "8 am - 9:30 am", "10 am - 11:30 am", "12 pm - 1:30 pm" };

for (int i = 0; i < 5; i++)

{

cout << setw(12) << week[i] << "\t";

for (int j = 0; j < 3; j++)

{

if (timeTable[i][j] == 1)

cout << setw(7) << time[j] << " \t";

}

cout << endl << "------------------------------------------------------------";

cout << "\n\n";

}

}

//SETTER OF TEACHER ID

void Teacher::teacherID::setteacherID(string a, int& b)

{

department = toupper(a[0]);

teacherNo = b + 1;

b++;

}

//DISPLAY TEACHER ID

void Teacher::teacherID::getTeacherID()

{

cout << "YOUR TEACHER ID IS: ";

cout << department << "-" << teacherNo << "\n";

}

//SETTER OF TEACHER FOR HOD

void Teacher::setTeacher(string a, string b, string c)

{

firstName = a;

lastName = b;

department = c;

obj.setteacherID(c, totalTeachers);

}

//SETTER OF TEACHER

void Teacher::setTeacher(string a, string b, string c, string d, string e, string f, char g, string h, string i, string j, int k)

{

firstName = a;

lastName = b;

department = c;

username = d;

password = e;

regDate = f;

gender = g;

contactNo = h;

qualification = i;

address = j;

salary = k;

obj.setteacherID(c, totalTeachers);

}

void Teacher::getTeachersData2()

{

cout << obj.department << "-" << obj.teacherNo;

cout << setw(20) << firstName;

cout << setw(20) << lastName;

cout << setw(20) << username;

cout << setw(20) << password;

cout << setw(25) << department;

cout << setw(20) << contactNo;

cout << setw(20) << salary;

cout << endl;

}

//DISPLAYS TEACHER

void Teacher::getTeacherData() //passwords visible?

{

cout << "\n\t\t\t TEACHER USERNAME: " << username << "\n\t\t\t TEACHER PASSWORD: " << password;

cout << "\n\n\tNAME: " << firstName << " " << lastName << "\n\tDEPARTMENT: " << department << "\n\tREGISTRATION DATE: " << regDate << "\n\tGENDER: ";

if (gender == 0)

cout << "Female";

else

cout << "Male";

cout << "\n\tCONTACT NO: " << contactNo << "\n\tQUALIFICATION: " << qualification;

cout << "\n\tADDRESS: " << address << "\n\tSALARY: " << salary << "\n";

}

//DISPLAYS TEACHER WITH THE BASIC INFO

void Teacher::getTeacherData2() //passwords visible?

{

cout << setw(13) << obj.department << "-" << obj.teacherNo;

cout << setw(16) << firstName;

cout << setw(16) << lastName;

cout << setw(16) << username;

cout << setw(16) << password;

cout << setw(16) << department;

cout << setw(16) << contactNo;

cout << setw(16) << salary;

cout << endl;

}

//DISPLAYS SIGN IN MENU FOR TEACHER

string Teacher::signIn()

{

system("cls");

fstream myInput;

string uCheck, pCheck;

bool uFlag = 0, pFlag = 0;

bool repeat = true;

while (repeat != false)

{

system("cls");

myInput.open("teachers.txt");

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TEACHERS SIGN IN PORTAL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout << endl << endl << endl;

cout << "\t ENTER YOUR USERNAME: ";

cin >> username;

cout << "\t ENTER YOUR PASSWORD: ";

cin >> password;

if (!(myInput.is\_open()))

cout << "FILE CANNOT BE OPENED! ";

else

{

while (!myInput.eof())

{

getline(myInput, uCheck);

if (username == uCheck)

{

uFlag = true;

break;

}

else

uFlag = 0;

}

while (!myInput.eof())

{

getline(myInput, pCheck);

if (password == pCheck)

{

pFlag = true;

break;

}

else

pFlag = 0;

}

if (pFlag == 1 && uFlag == 1)

{

repeat = false;

cout << "\n\tLOGGED IN SUCCESSFULLY! :)" << endl;

system("pause");

return username;

}

else

{

cout << endl;

cout << "\n\tINVALID CREDENTIALS, ENTER AGAIN! " << endl;

system("pause");

myInput.close();

repeat = true;

}

}

}

system("pause");

}

//MARKS ATTENDANCE

void Teacher::markAttendance(Student& obj)

{

system("cls");

obj.attendance.setAttendance(obj);

}

//ASSGNS MARKS

void Teacher::assignMarks(Student& obj)

{

cout << "\n\tENTER THE MARKS (OUT OF 50) OF " << obj.firstName << " " << obj.lastName << ": ";

cin >> obj.marks;

}

//ASSIGNS GRADES

void Teacher::assignGrades(Student& obj)

{

int perc = (obj.marks \* 100) / 100;

if (perc < 50)

obj.grade = 'F';

else if (perc >= 50 && perc <= 60)

obj.grade = 'D';

else if (perc >= 60 && perc <= 70)

obj.grade = 'C';

else if (perc >= 70 && perc <= 80)

obj.grade = 'B';

else if (perc >= 80 && perc <= 100)

obj.grade = 'A';

}

void Teacher::assignGrade(Student& a)

{

cout << "ENTER THE GRADE YOU WANT TO ASSIGN TO " << a.firstName << " " << a.lastName << ": ";

cin >> a.grade;

}

void Administrator::setAdmin(string a, string b, string u, string p)

{

firstName = a;

lastName = b;

username = u;

password = p;

}

void Administrator::signIn()

{

system("cls");

fstream myInput;

string uCheck, pCheck;

bool uFlag = 0, pFlag = 0;

bool repeat = true;

while (repeat != false)

{

system("cls");

myInput.open("admins.txt");

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ADMINISTRATOR SIGN IN PORTAL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout << endl << endl << endl;

cout << "\t ENTER YOUR USERNAME: ";

cin >> username;

cout << "\t ENTER YOUR PASSWORD: ";

cin >> password;

if (!(myInput.is\_open()))

cout << "FILE CANNOT BE OPENED! ";

else

{

while (!myInput.eof())

{

getline(myInput, uCheck);

if (username == uCheck)

{

uFlag = true;

break;

}

else

uFlag = 0;

}

while (!myInput.eof())

{

getline(myInput, pCheck);

if (password == pCheck)

{

pFlag = true;

break;

}

else

pFlag = 0;

}

if (pFlag == 1 && uFlag == 1)

{

repeat = false;

cout << "\n\tLOGGED IN SUCCESSFULLY! :)" << endl;

}

else

{

cout << endl;

cout << "\n\tINVALID CREDENTIALS, ENTER AGAIN! " << endl;

system("pause");

myInput.close();

repeat = true;

}

}

}

system("pause");

}

Teacher Administrator::addNewTeacher()

{

Teacher newTeacher;

cout << "\n\tENTER FIRST NAME: ";

cin.ignore();

getline(cin, newTeacher.firstName);

cout << "\n\tENTER LAST NAME: ";

cin.ignore();

getline(cin, newTeacher.lastName);

cout << "\n\tENTER DEPARTMENT (e.g. SE): "; cin >> newTeacher.department;

cout << "\n\tENTER REGISTRATION DATE (e.g. 19-02-2018): "; cin >> newTeacher.regDate;

cout << "\n\tENTER QUAILIFICATION: "; cin.ignore(); getline(cin, newTeacher.qualification);

cout << "\n\tENTER CONTACT NUMBER: "; cin >> newTeacher.contactNo;

cout << "\n\tENTER TEACHER USERNAME: "; cin >> newTeacher.username;

cout << "\n\tENTER TEACHER PASSWORD: "; cin >> newTeacher.password;

cout << "\n\tENTER ADDRESS: "; cin.ignore(); getline(cin, newTeacher.address);

newTeacher.obj.setteacherID(newTeacher.department, totalTeachers);

system("cls");

newTeacher.timeTable.setTimeTable(newTeacher);

return newTeacher;

}

//void Administrator::signUp()

//{

// cout << "ENTER THE TEACHER ID OF THE NEW TEACHER: ";

// cin >> username;

// cout << "ENTER THE PASSWORD FOR THE NEW TEACHER";

// cin >> password;

//}

Student Administrator::addNewStudent()

{

Student newStudent;

cout << "ENTER FIRST NAME: "; cin >> newStudent.firstName;

cout << "ENTER LAST NAME: "; cin >> newStudent.lastName;

cout << "ENTER FEE STATUS (1 FOR PAID/0 FOR NOT PAID): "; cin >> newStudent.feeStatus;

cout << "ENTER ADDRESS: "; cin >> newStudent.address;

cout << "ENTER CONTACT NUMBER: "; cin >> newStudent.contactNo;

cout << "ENTER QUAILIFICATION: "; cin >> newStudent.qualification;

cout << "ENTER BLOOD GROUP: "; cin >> newStudent.bloodGroup;

cout << "ENTER GENDER (1 FOR MALE/0 FOR FEMALE): "; cin >> newStudent.gender;

cout << "ENTER DEPARTMENT (e.g. AI): "; cin >> newStudent.department;

cout << "ENTER REGISTRATION DATE (e.g. 19-02-2018): "; cin >> newStudent.regDate;

cout << "ENTER STUDENT USERNAME: "; cin >> newStudent.username;

cout << "ENTER STUDENT PASSWORD: "; cin >> newStudent.password;

newStudent.obj.setID((17 + rand() % 5), 'F', (1000 + rand() % 4000 + 1));

return newStudent;

}

void Administrator::editStudent(Student& student)

{

int choice;

cout << "\tPRESS 1 TO EDIT FIRST NAME" << "\n\tPRESS 2 TO EDIT LAST NAME";

cout << "\n\tPRESS 3 TO EDIT FEE STATUS" << "\n\tPRESS 4 TO EDIT ADDRESS";

cout << "\n\tPRESS 5 TO EDIT CONTACT NUMBER" << "\n\n\tENTER YOUR CHOICE: "; cin >> choice;

if (choice == 1)

{

system("cls");

cout << "CURRENT FIRST NAME IS: " << student.firstName << endl;

cout << "ENTER NEW FIRST NAME: "; cin >> student.firstName;

cout << endl;

cout << "UPDATED! :)";

cout << endl;

system("pause");

}

else if (choice == 2)

{

system("cls");

cout << "CURRENT LAST NAME IS: " << student.lastName << endl;

cout << "ENTER NEW LAST NAME: "; cin >> student.lastName;

cout << endl;

cout << "UPDATED! :)";

cout << endl;

system("pause");

}

else if (choice == 3)

{

system("cls");

cout << "CURRENT FEE STATUS IS: " << ((student.feeStatus) ? "PAID" : "NOT PAID");

cout << endl;

cout << "ENTER NEW FEE STATUS: "; cin >> student.feeStatus;

cout << endl;

cout << "UPDATED! :)";

cout << endl;

system("pause");

}

else if (choice == 4)

{

system("cls");

cout << "CURRENT ADDRESS IS: " << student.address << endl;

cout << "ENTER NEW ADDRESS: "; cin >> student.address;

cout << endl;

cout << "UPDATED! :)";

cout << endl;

system("pause");

}

else if (choice == 5)

{

system("cls");

cout << "CURRENT CONTACT NUMBER IS: " << student.contactNo << endl;

cout << "ENTER NEW CONTACT NUMBER: "; cin >> student.contactNo;

cout << endl;

cout << "UPDATED! :)";

cout << endl;

system("pause");

}

system("cls");

}

void Administrator::editTeacher(Teacher& teacher)

{

int choice;

cout << "\tPRESS 1 TO EDIT QUALIFICATION" << "\n\tPRESS 2 TO EDIT SALARY";

cout << "\n\tPRESS 3 TO EDIT ADDRESS" << "\n\tPRESS 4 TO EDIT CONTACT NUMBER" << "\n\n\tENTER YOUR CHOICE: "; cin >> choice;

if (choice == 1)

{

system("cls");

cout << "CURRENT QUALIFICATION IS: " << teacher.qualification << endl;

cout << "ENTER NEW QUALIFICATION: "; cin >> teacher.qualification;

cout << endl;

cout << "UPDATED! :)";

cout << endl;

system("pause");

}

else if (choice == 2)

{

system("cls");

cout << "CURRENT SALARY IS: " << teacher.salary << endl;

cout << "ENTER NEW SALARY: "; cin >> teacher.salary;

cout << endl;

cout << "UPDATED! :)";

cout << endl;

system("pause");

}

else if (choice == 3)

{

system("cls");

cout << "CURRENT ADDRESS IS: " << teacher.address << endl;

cout << "ENTER NEW ADDRESS: "; cin >> teacher.address;

cout << endl;

cout << "UPDATED! :)";

cout << endl;

system("pause");

}

else if (choice == 4)

{

system("cls");

cout << "CURRENT CONTACT NUMBER IS: " << teacher.contactNo << endl;

cout << "ENTER NEW CONTACT NUMBER: "; cin >> teacher.contactNo;

cout << endl;

cout << "UPDATED! :)";

cout << endl;

system("pause");

}

system("cls");

}

void Administrator::viewAllStudents(Student\* students)

{

HOD::viewAllStudents(students);

}

void Administrator::viewAllTeachers(Teacher\* teachers)

{

HOD::viewAllTeachers(teachers);

}

void HOD::setHOD(string a, string b, string c, string u, string p)

{

firstName = a;

lastName = b;

department = c;

username = u;

password = p;

}

void HOD::addTeachersFunctionality(Teacher\*& a)

{

obj = &a[totalTeachers];

}

string HOD::signIn()

{

system("cls");

fstream myInput;

string uCheck, pCheck;

bool uFlag = 0, pFlag = 0;

bool repeat = true;

while (repeat != false)

{

system("cls");

myInput.open("hods.txt");

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*HEAD OF DEPARTMENT SIGN IN PORTAL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout << endl << endl << endl;

cout << "\t ENTER YOUR USERNAME: ";

cin >> username;

cout << "\t ENTER YOUR PASSWORD: ";

cin >> password;

if (!(myInput.is\_open()))

cout << "FILE CANNOT BE OPENED! ";

else

{

while (!myInput.eof())

{

getline(myInput, uCheck);

if (username == uCheck)

{

uFlag = true;

break;

}

else

uFlag = 0;

}

while (!myInput.eof())

{

getline(myInput, pCheck);

if (password == pCheck)

{

pFlag = true;

break;

}

else

pFlag = 0;

}

if (pFlag == 1 && uFlag == 1)

{

repeat = false;

cout << "\n\tLOGGED IN SUCCESSFULLY! :)" << endl;

system("pause");

return username;

}

else

{

cout << endl;

cout << "\n\tINVALID CREDENTIALS, ENTER AGAIN! " << endl;

system("pause");

myInput.close();

repeat = true;

}

}

}

system("pause");

}

void HOD::viewTeacherActivity(Teacher\* a)

{

//views that teacher details

cout << "\n\n\t\t\t TOTAL NUMBER OF CURRENT FACULTY: " << totalTeachers << endl << endl;

for (int i = 0; i < totalTeachers; i++)

{

cout << "------------------------------------------------------------------------------------------------------\n";

a[i].getTeacherData();

cout << "------------------------------------------------------------------------------------------------------\n";

}

}

void HOD::viewStudentModule(Student\* a)

{

//views and manipulates the contents of the student obj passed

cout << "\n\n\t\t\t TOTAL NUMBER OF PRESENT ENROLLED STUDENTS: " << totalStudents << endl << endl;

for (int i = 0; i < totalStudents; i++)

{

cout << "------------------------------------------------------------------------------------------------------\n";

a[i].getStudentData();

cout << "------------------------------------------------------------------------------------------------------\n";

}

}

void HOD::viewAllStudents(Student\* a)

{

cout << "\n\n\t\t\t TOTAL NUMBER OF PRESENT ENROLLED STUDENTS: " << totalStudents << endl << endl;

cout << "------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------\n";

cout << setw(18) << "ROLL NO ";

cout << setw(18) << "FIRST NAME ";

cout << setw(18) << "LAST NAME ";

cout << setw(18) << "USERNAME ";

cout << setw(18) << "PASSWORD ";

cout << setw(18) << "DEPARTMENT ";

cout << setw(18) << "CONTACT NO ";

cout << setw(18) << "FEE STATUS \n";

cout << "------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------\n";

for (int i = 0; i < totalStudents; i++)

{

a[i].getStudentData2();

}

}

void HOD::viewAllTeachers(Teacher\* a)

{

cout << "\n\n\t\t\t TOTAL NUMBER OF CURRENT FACULTY: " << totalTeachers << endl << endl;

cout << "--------------------------------------------------------------------------------------------------------------------------------------------------------\n";

cout << setw(18) << "TEACHER ID ";

cout << setw(18) << "FIRST NAME ";

cout << setw(18) << "LAST NAME ";

cout << setw(18) << "USERNAME ";

cout << setw(18) << "PASSWORD ";

cout << setw(18) << "DEPARTMENT ";

cout << setw(18) << "CONTACT NO ";

cout << setw(18) << "SALARY ";

cout << endl;

cout << "--------------------------------------------------------------------------------------------------------------------------------------------------------\n";

for (int i = 0; i < totalTeachers; i++)

{

a[i].getTeacherData2();

}

}