**Abstract**

The project named ―Placement Management System, is a staff/company information system. The project is developed on the basis of storing and retrieving the information of students who details registered during run time. All the student’s details are written into a comma separated file and the csv file maintains a large database of students wherein all the information of student including the academic performance in terms of the GPA is stored and company information including name of company, eligibility criteria. The software retrieves this data and displays as per the user requirement The System provides the facility of viewing the academic information of the student and company can also search for eligible students and company and also insertion of records by the administrator.

**1. PROBLEM STATEMENT:**

Implementation of PLACEMENT SYSTEM using C++.

**2. INTRODUCTION:**

In this competitive world that we live, nothing is so easy to obtain as we grow day-by-day.

As we run behind something to achieve, time plays a major role. In order to reduce the time that we require for a particular job, we search for a less time-consuming solution. Here is such a solution for placement cell management. The PLACEMENT SYSTEM project is an attempt to take the record of the students and help company to select the students based on their average GPA. Eligible student’s data will be displayed when the requirement of the company is satisfied.

**2.1 OBJECTIVE:**

* To enable admin person to enter marks
* To ensure that companies get eligible candidates

**3 Concepts:**

**3.1 Vectors:**

Vectors are same as dynamic arrays with the ability to resize itself

automatically when an element is inserted or deleted, with their storage being

handled automatically by the container. Vector elements are placed in contiguous

storage so that they can be accessed and traversed using iterators. In vectors,

data is inserted at the end. Inserting at the end takes differential time, as

sometimes there may be a need of extending the array. Removing the last

element takes only constant time because no resizing happens. Inserting and

erasing at the beginning or in the middle is linear in time.

**3.2 Constructors:**

A constructor is a member function of a class which initializes

objects of a class. In C++, Constructor is automatically called when object (instance of class) create. It is special member function of the class. Constructor

has same name as the class itself. Constructors don’t have return type. A

constructor is automatically called when an object is created. If we do not specify a constructor, C++ compiler generates a default constructor for us (expects no

parameters and has an empty body).

**3.3 Friend class:**

A friend class can access private and protected members of other class in which it is declared as friend. It is sometimes useful to allow a particular class to access private members of other class. Friends should be used only for limited purpose. too many functions or external classes are declared as friends of a class with protected or private data, it lessens the value of encapsulation of separate classes in object-oriented programming. Friendship is not inherited. The concept of friends is not there in Java.

**3.4 Files:**

In C++, files are mainly dealt by using three classes fstream, ifstream, ofstream available in fstream headerfile.

**ofstream:** Stream class to write on files

**ifstream:** Stream class to read from files

**fstream:** Stream class to both read and write from/to files.

**3.4.1 CSV File:**

**CSV** is a simple file format used to store tabular data such as a spreadsheet or a database. CSV stands for **Comma Separated Values**. The data fields in a CSV file are separated/delimited by a comma **(‘,‘)** and the individual rows are separated by a newline **(‘\n’)**. CSV File management in C++ is similar to text-type file management, except for a few modifications.

**3.5 Exception Handling:**

One of the advantages of C++ over C is Exception Handling. Exceptions are run-time anomalies or abnormal conditions that a program encounters during its execution. There are two types of exceptions: a) Synchronous, b) Asynchronous (which are beyond the program’s control, Disc failure etc.). C++ provides following specialized keywords for this purpose.

try: represents a block of code that can throw an exception.

catch: represents a block of code that is executed when a particular exception is

thrown.

throw: Used to throw an exception. Also used to list the exceptions that a function

throws, but doesn’t handle itself.

**4.Implementation**

**4.1 Module:**

**Student:**

Student class has the details like student name and semester GPA for 6 semesters. All the details is written into comma separated file. All the functions like getting details from company, students’ details is written here.

**HR:**

HR class is the friend class of student. It has variable like average GPA marks and arrear numbers.

**4.2 Class Design:**

|  |
| --- |
| Student |
| float m1  float m2  float m3  float m4  float m5  float m6  Student();  void get\_Candidates(string,float);  void admin\_createMarkList(string);  void admin\_updateList(string,string,int,float);  void create\_file(string);  friend class HR; |

|  |
| --- |
| HR |
| float avg;  int backlog; |

**4.2.1 Class Description:**

**class Student**

* m1,m2,m3,m4,m5,m6 are the variable representing semester GPA of students.
* Student() is default constructor.
* get\_Candidates(string,float) function gets the candidates details.
* admin\_createMarkList(string) function creates the mark list of student.
* create\_file(string) function creates csv file and writes the student data into it.
* friend class HR makes the class HR friend to class student

**class Hr**

* avg refers to average GPA of students.
* backlog refers to number of arrears held by students.

**5. CODE:**

**5.1 Header code**

#include<iostream>

#include<string>

#include<fstream>

#include<exception>

#include<iomanip>

#include<vector>

using namespace std;

class Student

{

private:

float m1,m2,m3,m4,m5,m6;

public:

Student();

void get\_Candidates(string,float);

void admin\_createMarkList(string);

void admin\_updateList(string,string,int,float);

void create\_file(string);

friend class HR;

};

class HR

{

private:

float avg;

int backlog;

};

**5.2 Application code**

#include"header.h"

using namespace std;

//Driver Program

main()

{

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout<<" MEPCO PLACEMENT SYSTEM FOR HR \n";

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout<<"Enter User Type "<<endl;

cout<<"Option 1. ADMIN"<<endl;

cout<<"Option 2. HR" <<endl;

cout<<"Option : ";

int in\_type;

string batch\_year,company\_name;

float average,new\_mark;

Student S;

cin>>in\_type;

switch(in\_type)

{

case 1:

cout<<"You have Choosen ADMIN" <<endl ;

cout<<"Enter File Name as batch\_name - e.g.2019 : ";

cin>>batch\_year;

S.admin\_createMarkList(batch\_year);

break;

case 2:

cout<<"Welcome" << endl;

cout<<"Please Enter Your Company Name :" << endl;

cin>>company\_name;

cout<<"Please Enter Recruitment Batch Number : " << endl;

cin>>batch\_year;

cout<<"Enter Average marks For the Student : ";

cin>>average;

S.get\_Candidates(batch\_year,average);

break;

default :

cout<<"Undefined Option" << endl ;

exit (1);

break;

}

cout<<"\nThank you for Using Mepco Placement System" << endl;

}

**5.3 Implementation code**

#include"header.h"

using namespace std;

Student::Student()

{

float m1,m2,m3,m4,m5,m6=0.0;

}

void Student::get\_Candidates(string batch,float input\_avg)

{

fstream fin;

float avg;

string file=batch;

fin.open(file,ios::in);

if(!fin)

{

cout<<"File Not Present,Contact Admin" <<endl ;

exit(1);

}

else

{

cout<<"Checking .... ";

}

int count=0;

vector<string> row;

vector<string> name;

string line, word, temp;

while (!fin.eof())

{

row.clear();

getline(fin, line);

stringstream s(line);

while (getline(s, word, ','))

{

row.push\_back(word);

}

float

avg=(stof(row[1])+stof(row[2])+stof(row[3])+stof(row[4])+stof(row[5])+stof(row

[6]))/6;

if(avg>input\_avg)

{

count=count+1;

name.push\_back(row[0]);

}

}

if(count>0)

{

cout<<"Eligible Candidates " << endl ;

for(int i=0;i<count-1;i++)

{

cout<<name[i] << "\n" ;

}

}

else

{

cout<<"No elligible candidate\n";

}

};

bool fexists(const std::string& filename) {

std::ifstream ifile(filename.c\_str());

return (bool)ifile;

}

void Student::create\_file(string filename)

{

fstream newfile;

newfile.open(filename, ios::out | ios::app);

cout<<"Enter the Number Of Students :";

int number\_of\_students;

float sem1,sem2,sem3,sem4,sem5,sem6;

int backlog;

string student\_name;

cin>>number\_of\_students;

for (int loops=1;loops<=number\_of\_students;loops++)

{

cout<<"Enter Student Name" << "[" << loops << "] : " ;

cin>>student\_name;

cout<<"Enter Semester Mark 1 "<< "[" << loops << "] : " ;

cin>>sem1;

cout<<"Enter Semester Mark 2 "<< "[" << loops << "] : " ;

cin>>sem2;

cout<<"Enter Semester Mark 3 "<< "[" << loops << "] : " ;

cin>>sem3;

cout<<"Enter Semester Mark 4 "<< "[" << loops << "] : " ;

cin>>sem4;

cout<<"Enter Semester Mark 5 "<< "[" << loops << "] : " ;

cin>>sem5;

cout<<"Enter Semester Mark 6 "<< "[" << loops << "] : " ;

cin>>sem6;

cout<<"Enter Holding Arreas "<< "[" << loops << "] : " ;

cin>>backlog;

newfile << student\_name << ","

<< sem1 << ","

<< sem2 << ","

<< sem3 << ","

<< sem4 << ","

<< sem5 << ","

<< sem6 << ","

<< backlog<< "\n" ;

}

cout<<"New File Created - ," << filename ;

}

void Student::admin\_createMarkList(string batch)

{

bool file\_check=fexists(batch);

if(file\_check)

{

cout<<"File Already Exist" <<endl ;

exit(1);

}

else

create\_file(batch);

}

**5.4 MakeFile**

PlacementMarkSystem : application.o implementation.o

c++ application.o implementation.o -o PlacementMarkSystem

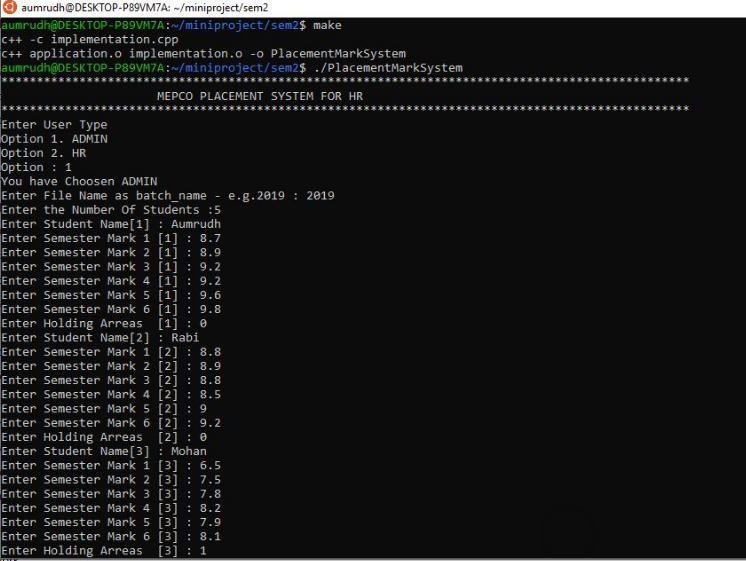
application.o : application.cpp

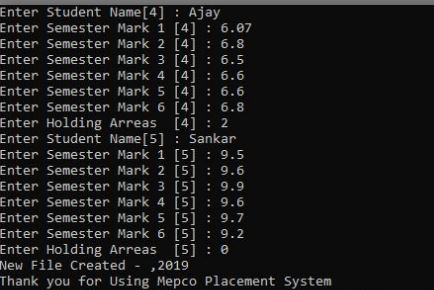
c++ -c application.cpp

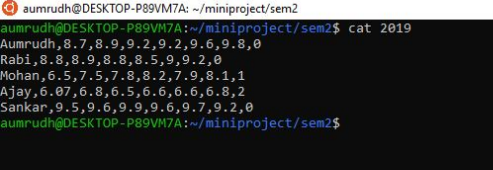
implementation.o:implementation.cpp

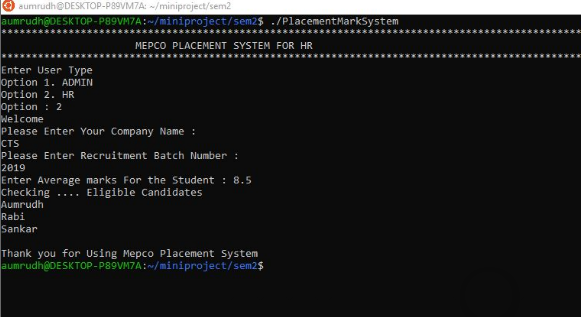
c++ -c implementation.cpp

**6.Output:**









**7. CONCLUSION:**

The Placement system created using c++ looks so simple for the naive users to work with it. Also viewing of all records is also simple. It provides efficiency and comfort for the company and admins by allowing the former to get eligible candidates and the later to enter record. By using this application, we can store records in csv file efficiently in organized manner. This quick response seems to be the main advantage of this application.

**8. REFERENCES:**

1. [www.geeksforgeeks.org](http://www.geeksforgeeks.org)