#include <iostream>

#include <fstream>

#include <string>

#include <sstream>

#include <conio.h>

#include <string.h>

#include <iomanip>

using namespace std;

//this is a base name class where all other classes are derived from

class Name

{

string name;

public:

void setName(string name){this->name=name;}

string getName(){return name;}

};

//this is a company object used to store the companies obtained from the database

class cmp:public Name

{

string location,package;

int id;

int avail\_pos;

int max\_pos;

int number\_of\_placed;

float eligibility;

string placed\_students[30];

public:

//counter used to keep track of the number of companies in the database and assign the corresponding IDs

static int counter;

void setAvailPos()

{//sets the number of available positions in the company

avail\_pos=max\_pos-number\_of\_placed;

}

//setter to set the placed student in the database

void setPlacedStudents(string placed\_students)

{

int pos=-1,curr=-1,i=0;

//if there are no placed\_students or the length of placed students is 0

if(placed\_students=="null")

{

this->number\_of\_placed=0;

}

//if there is only one student

else if(placed\_students.find(",")==-1)

{

this->placed\_students[0]=placed\_students;

this->number\_of\_placed=1;

}

//if there are more than one

else

{

while(placed\_students.find(",",pos+1)!=-1)

{

curr=pos+1;

pos=placed\_students.find(",",curr);

this->placed\_students[i++]=placed\_students.substr(curr,pos-curr);

}

this->placed\_students[i++]=placed\_students.substr(pos+1,placed\_students.length()-pos+1);

this->number\_of\_placed=i;

}

}

void setCmp(string name,string location,string package,float eligibility,int max\_pos,string placed\_students)

{//sets the company details

setName(name);

this->location=location;

this->package=package;

this->eligibility=eligibility;

this->max\_pos=max\_pos;

setPlacedStudents(placed\_students);

setAvailPos();

this->id=++counter;

}

bool addStudent(string name)

{

//to add a new student in the

int i=0;

if(number\_of\_placed==max\_pos)

{

cout<<"Already full"<<endl;

return false;

}

else{

try

{

for(i=0;i<number\_of\_placed;i++)

{

if(placed\_students[i]==name)

throw name;

}

placed\_students[number\_of\_placed]=name;

number\_of\_placed++;

setAvailPos();

return true;

}

catch(string name)

{

cout<<"Student already placed in this company"<<endl;

return false;

}

}

}

/\*

void rmStudent(string name)

{

//to remove the student from the placed student

int i=0;

if(number\_of\_placed==1&&name==placed\_students[0])

{

placed\_students[0]="null";

number\_of\_placed--;

}

else if(name==placed\_students[number\_of\_placed-1])

{

number\_of\_placed--;

}

try

{

for(i=0;i<number\_of\_placed;i++)

{

if(placed\_students[i]==name)

{

for(int j=i;j<number\_of\_placed-1;j++)

{

placed\_students[j]=placed\_students[j+1];

}

number\_of\_placed--;

setAvailPos();

return;

}

}

throw name;

}

catch(string name)

{

cout<<"Not in the company"<<endl;

}

}\*/

void displayCmp()

{

cout<<"\n\nName : "<<getName()<<endl;

cout<<"Location: "<<this->location<<endl;

cout<<"Eligibility :"<<eligibility<<"CGPA or greater"<<endl;

cout<<"Package :"<<package<<endl;

cout<<"Number of students placed :"<<number\_of\_placed<<endl;

cout<<"Number of available positions :"<<avail\_pos<<endl;

cout<<"ID:"<<id<<endl;

}

//getters

int getAvailPos(){return avail\_pos;}

string getLocation(){return location;}

string getPackage(){return package;}

int getMax\_pos(){return max\_pos;}

float getEligibility(){return eligibility;}

string\* getPlaced\_students(){return placed\_students;}

int getNumberOfPlaced(){return number\_of\_placed;}

string getPlaced\_students\_string()

{

if(number\_of\_placed==0)

{

return "null";

}

else{

string students;

int i=0;

for(i=0;i<number\_of\_placed-1;i++)

{

students=students+placed\_students[i]+",";

}

students=students+placed\_students[i];

return students;

}

}

~cmp()

{

counter--;

}

};

//this is a new student class for creating a Newstud object

class NewStud:public Name

{

string email,cmp\_placed,password;

float cgpa;

public:

void setNewStud(string name,string email,string password,string cmp\_placed,float cgpa)

{

setName(name);

this->email=email;

this->password=password;

this->cmp\_placed=cmp\_placed;

this->cgpa=cgpa;

}

string getEmail(){return email;}

string getPassword(){return password;}

string getCmp\_Placed(){return cmp\_placed;}

float getCgpa(){return cgpa;}

};

//student class

class student:public Name

{

string email,password,cmp\_placed;

float cgpa;

bool placed;

public:

void setStudent(string name,string email,string password,string cmp\_placed,float cgpa)

{

setName(name);

this->email=email;

this->password=password;

this->cmp\_placed=cmp\_placed;

this->cgpa=cgpa;

setPlaced();

}

void setPlaced()

{ //sets the placed boolean to true or false

if(cmp\_placed=="null")

{

placed=false;

}

else

placed=true;

}

void operator=(NewStud newStud)

{

setName(newStud.getName());

this->email=newStud.getEmail();

this->password=newStud.getPassword();

this->cmp\_placed=newStud.getCmp\_Placed();

this->cgpa=newStud.getCgpa();

setPlaced();

}

void operator=(student stud)

{

setName(stud.getName());

this->email=stud.getEmail();

this->password=stud.getPassword();

this->cmp\_placed=stud.getCmp\_Placed();

this->cgpa=stud.getCgpa();

setPlaced();

}

void displayStudent()

{

cout<<"Name :"<<getName()<<endl;

cout<<"Email :"<<email<<endl;

cout<<"Password :"<<password<<endl;

cout<<"Company placed in:"<<cmp\_placed<<endl;

cout<<"Cgpa :"<<cgpa<<endl;

}

void setCmpPlaced(string cmp\_placed)

{

this->cmp\_placed=cmp\_placed;

setPlaced();

}

string getEmail(){return email;}

string getPassword(){return password;}

string getCmp\_Placed(){return cmp\_placed;}

float getCgpa(){return cgpa;}

bool getPlaced(){return placed;}

};

//Global variables

cmp company[10];

bool signedIn=false;

student students[30];

student currStudent;

int number\_of\_companies=0;

int number\_of\_students=0;

//static variables

int cmp::counter=0;

//functions

void getCmpDatabase()

{

ifstream cmp\_getdatabase;

string name,location,package,placed\_students;

float eligibility;

int max\_pos;

cmp\_getdatabase.open("./database/company\_database.txt");

int i=0;

while(cmp\_getdatabase.eof()!=1)

{

int flag=0;

if(cmp\_getdatabase.eof())

{

flag=1;

}

cmp\_getdatabase>>name;

cmp\_getdatabase>>location;

cmp\_getdatabase>>package;

cmp\_getdatabase>>eligibility;

cmp\_getdatabase>>max\_pos;

cmp\_getdatabase>>placed\_students;

if(flag==0&&!cmp\_getdatabase.eof())

{

company[i].setCmp(name,location,package,eligibility,max\_pos,placed\_students);

number\_of\_companies++;

i++;

}

}

cmp\_getdatabase.close();

}

void setCmpDatabase()

{

ofstream cmp\_database;

string name,location,package,placed\_students;

float eligibility;

int max\_pos;

cmp\_database.open("./database/company\_database.txt");

for(int i=0;i<number\_of\_companies;i++)

{

cmp\_database<<setw(15)<<setiosflags(ios::left)<<company[i].getName();

cmp\_database<<setw(15)<<setiosflags(ios::left)<<company[i].getLocation();

cmp\_database<<setw(15)<<setiosflags(ios::left)<<company[i].getPackage();

cmp\_database<<setw(15)<<setiosflags(ios::left)<<company[i].getEligibility();

cmp\_database<<setw(15)<<setiosflags(ios::left)<<company[i].getMax\_pos();

cmp\_database<<setiosflags(ios::left)<<company[i].getPlaced\_students\_string();

cmp\_database<<"\n";

}

cmp\_database.close();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Student database\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void getStudentDatabase()

{

string name,email,password,cmp\_placed;

float cgpa;

ifstream student\_getdatabase;

student\_getdatabase.open("./database/student\_database.txt");

int i=0;

while(student\_getdatabase.eof()!=1)

{

int flag=0;

if(student\_getdatabase.eof())

{

flag=1;

}

student\_getdatabase>>name;

student\_getdatabase>>email;

student\_getdatabase>>password;

student\_getdatabase>>cmp\_placed;

student\_getdatabase>>cgpa;

if(!student\_getdatabase.eof()&&flag==0)

{

students[i].setStudent(name,email,password,cmp\_placed,cgpa);

number\_of\_students++;

i++;

}

}

student\_getdatabase.close();

}

void setStudentDatabase()

{

ofstream student\_database;

string name,email,password,cmp\_placed;

float cgpa;

student\_database.open("./database/student\_database.txt");

for(int i=0;i<number\_of\_students;i++)

{

student\_database<<setw(15)<<setiosflags(ios::left)<<students[i].getName();

student\_database<<setw(30)<<setiosflags(ios::left)<<students[i].getEmail();

student\_database<<setw(15)<<setiosflags(ios::left)<<students[i].getPassword();

student\_database<<setw(15)<<setiosflags(ios::left)<<students[i].getCmp\_Placed();

student\_database<<setw(15)<<setiosflags(ios::left)<<students[i].getCgpa();

student\_database<<"\n";

}

student\_database.close();

}

void setCurrStudent(string email,string password)

{

//sets the current user based on email and password

string name,cmp\_placed;

float cgpa;

for(int i=0;i<number\_of\_students;i++)

{

if(students[i].getEmail()==email)

{

name=students[i].getName();

cmp\_placed=students[i].getCmp\_Placed();

cgpa=students[i].getCgpa();

currStudent.setStudent(name,email,password,cmp\_placed,cgpa);

signedIn=true;

return;

}

}

}

void addStudentInDatabase(string name,string email,string password,string cmp\_placed,float cgpa)

{

//creates a new student object and adds it to the students obj array

NewStud student;

student.setNewStud(name,email,password,cmp\_placed,cgpa);

students[number\_of\_students++]=student;

}

//checks if student present in database

bool checkStudentInDatabase(string email,string password)

{

for(int i=0;i<number\_of\_students;i++)

{

if(email==students[i].getEmail())

{

return true;

}

}

return false;

}

void createNewStudent()

{

string name,email,password,cmp\_placed;

float cgpa;

int choice;

cout<<"Please enter the following details in the form :"<<endl;

cout<<"Enter Name: ";

cin.ignore();

cin>>name;

cout<<"Enter Email: ";

cin.ignore();

cin>>email;

cout<<"Enter Password: ";

cin.ignore();

cin>>password;

system("cls");

cout<<"Enter your cgpa: ";

cin.ignore();

cin>>cgpa;

if(checkStudentInDatabase(email,password))

{

setCurrStudent(email,password);

cout<<"Signed In!"<<endl;

return;

}

system("cls");

cmp\_placed="null";

cout<<"Adding you into the database please wait......"<<endl;

addStudentInDatabase(name,email,password,cmp\_placed,cgpa);

cout<<"Student signed-up !"<<endl;

setCurrStudent(email,password);

}

bool checkEmailAndPassword(string email,string password)

{ //checks the password

for(int i=0;i<number\_of\_students;i++)

{

if(email==students[i].getEmail())

{

if(password==students[i].getPassword())

{

return true;

}

return false;

}

}

}

void SignInStudent()

{

while(true){

string email,password;

cout<<"Sign-In Student :"<<endl;

cin.ignore();

cout<<"\nEnter the email : ";

cin>>email;

cout<<"Enter the password : ";

cin>>password;

if(!checkStudentInDatabase(email,password))//checks if the student is in the database returns a boolean

{

cout<<"Student does not exist in the database please sign-up"<<endl;

break;

}

else

{

if(!checkEmailAndPassword(email,password))//checks if the password is correct

{

cout<<"Password incorrect enter again"<<endl;

}

else

{

setCurrStudent(email,password);//sets the current user based on the email

cout<<"Signed in !"<<endl;

break;

}

}

}

}

string joinCmp(int id)

{

system("cls");

int choice;

if(!currStudent.getPlaced())

{

company[id-1].displayCmp();

cout<<"Do you wish to join this company ?"<<endl<<endl;

cout<<"1.Yes 2.No"<<endl;

cin.ignore();

cin>>choice;

if(choice==1)

{

if(currStudent.getCgpa()<company[id-1].getEligibility())

{

return "You are not eligible to join this company based on the company's hiring criteria";

}

cout<<"adding you to the company ... please wait"<<endl;

if(company[id-1].addStudent(currStudent.getName()))

{

for(int i=0;i<number\_of\_students;i++)

{

if(currStudent.getEmail()==students[i].getEmail())

{

currStudent.setCmpPlaced(company[id-1].getName());

students[i]=currStudent;

return "Done! you've joined the company";

}

}

}

else

{

return "Please try again";

}

}

else if(choice==2)

{

return "";

}

}

else

return "Already placed !";

}

void app()

{

int op;

string message="";

while(signedIn)

{

system("cls");

cout<<message<<"\n\n"<<endl;

cout<<"->"<<"WELCOME TO THE PLACEMENT MANAGEMENT SYSTEM!<-"<<endl;

cout<<"Student :"<<currStudent.getName()<<setw(60-currStudent.getName().length())<<setiosflags(ios::right)<<"email:"<<currStudent.getEmail()<<endl;

cout<<"Status :";

if(currStudent.getPlaced())

cout<<"Placed at "<<currStudent.getCmp\_Placed()<<endl;

else

cout<<"Not placed"<<endl;

cout<<"Available companies for placement :\n";

cout<<"Total: "<<number\_of\_companies<<endl;

cout<<"0. Sign-Out"<<endl;

if(number\_of\_companies==0)

{

cout<<"No companies available for placement\n";

}

else

{

for(int i=0;i<number\_of\_companies;i++)

{

if(company[i].getAvailPos()!=0)

company[i].displayCmp();

cout<<"\n\n";

}

cout<<"Enter the ID of the company you wish to apply for or 0 for exit:\n";

cin.ignore();

cin>>op;

if(op==0)

signedIn=false;

else

message=joinCmp(op);

}

}

}

int main()

{

getStudentDatabase();

getCmpDatabase();

int op=0;

while(op!=3)

{

if(!signedIn) //sign-in/sign-up page

{

system("cls");

cout<<"->"<<"WELCOME TO THE PLACEMENT MANAGEMENT SYSTEM!<-\n\n"<<endl;

cout<<"->"<<"1)Sign-Up 2)Sign-In 3)Exit<-\n";

cin>>op;

switch(op)

{

case 1:

{

createNewStudent();

}break;

case 2:

{

SignInStudent();

}break;

case 3:

{

cout<<"Thank you for using our service"<<endl;

}break;

}

}

else//the app page

{

cout<<"Please wait....\n";

app();

}

}

setCmpDatabase();

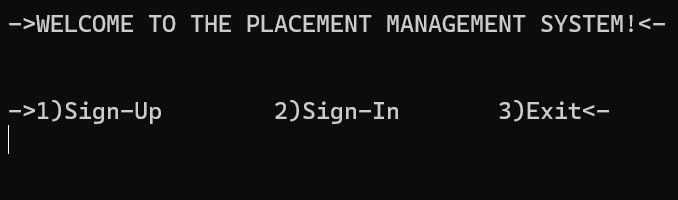
setStudentDatabase();

return 0;

}

OUTPUT:

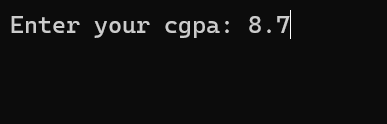
**Case: New User Signs In and wants to get placed in a company->**



**Sign-Up->**



**Enter CGPA**



**Main app page:**

->WELCOME TO THE PLACEMENT MANAGEMENT SYSTEM!<-

Student :Bhavesh email:bh@gmail.com

Status :Not placed

Available companies for placement :

Total: 4

0. Sign-Out

Name : google

Location: bangalore

Eligibility :8.5CGPA or greater

Package :6lpa

Number of students placed :0

Number of available positions :5

ID:1

Name : amazon

Location: noida

Eligibility :8.6CGPA or greater

Package :7lpa

Number of students placed :0

Number of available positions :10

ID:2

Name : microsoft

Location: mumbai

Eligibility :8.1CGPA or greater

Package :9lpa

Number of students placed :0

Number of available positions :8

ID:3

Name : creativeJ

Location: goa

Eligibility :8.4CGPA or greater

Package :7lpa

Number of students placed :1

Number of available positions :9

ID:4

Enter the ID of the company you wish to apply for or 0 for exit:

4

**User Enters 4:**

Name : creativeJ

Location: goa

Eligibility :8.4CGPA or greater

Package :7lpa

Number of students placed :1

Number of available positions :9

ID:4

Do you wish to join this company ?

1.Yes 2.No

1

**Enters 1(Yes)**

**Returns to home page with a message saying youve joined the company**

Done! you've joined the company

->WELCOME TO THE PLACEMENT MANAGEMENT SYSTEM!<-

Student :Bhavesh email:bh@gmail.com

Status :Placed at creativeJ

Available companies for placement :

Total: 4

0. Sign-Out

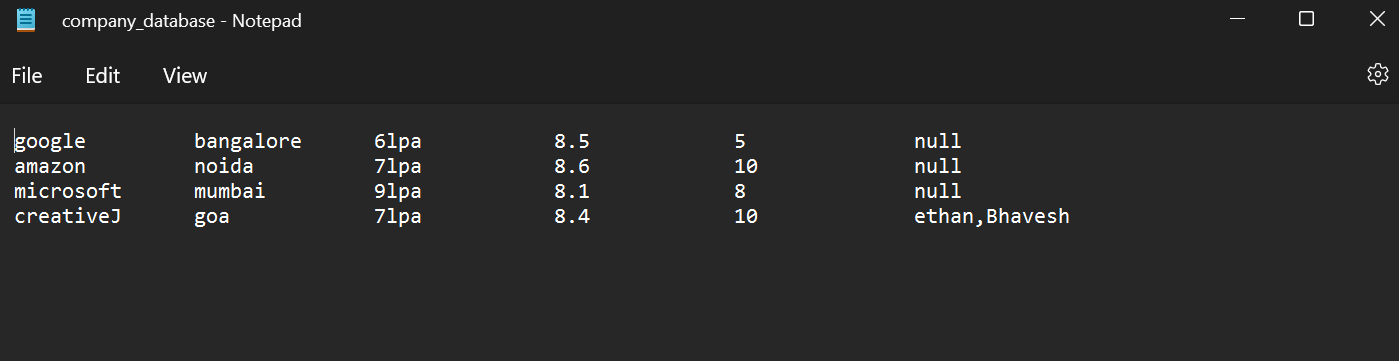
->WELCOME TO THE PLACEMENT MANAGEMENT SYSTEM!<-

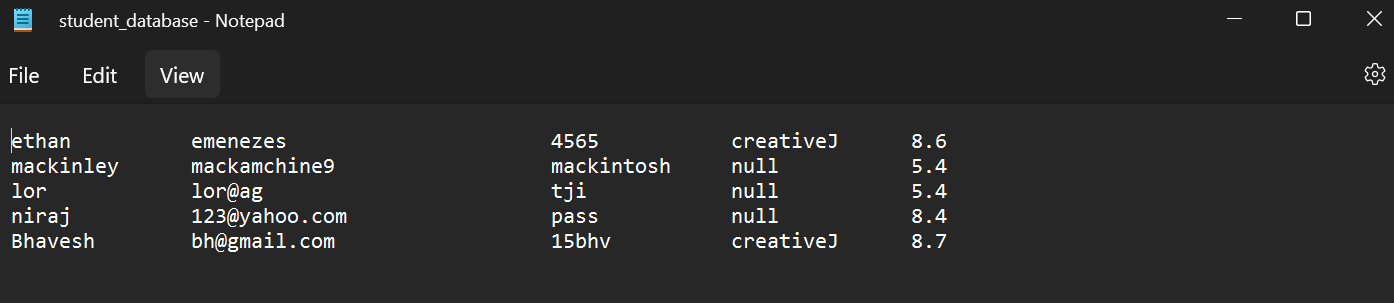
->1)Sign-Up 2)Sign-In 3)Exit<-

3

Thank you for using our service

**Database after the new sign-in**





**A new entry is added in the database (Last entry)**