**IIIT BHUBANESWAR**

**C++ PROJECT ON**

**EMPLOYEE MANAGEMENT SYSTEM**



GUIDED BY:

MR. MUKTIKANTA SAHU SIR

SUBMITTED BY:

* PRANGYA PANDA(B318032)
* SONALI SHAW(B318047)
* SUKANYA SAHOO(B318050)
* SWOSTI PRADHAN(B318051)

ACKNOWLEDGEMENT

We express our Gratitude to Mr. Muktikanta Sahu for his help, guidance throughout the work for this project without which I would not have been able to complete this project to such a success.

Thanks to him for the constant support and guidance to accomplish the task successfully.

PRANGYA PANDA

SONALI SHAW

SUKANYA SAHOO

SWOSTI PRADHAN

CONTENT

1.INTRODUCTION

2.OBJECTIVE

3.SYSTEM SPECIFICATION

a. SOFTWARE REQUIREMENTS

4. LANGUAGE USED

5.CODING

6.IMPLEMENTATION

7.CONCLUSION

8.BIBLIOGRAPHY

INTRODUCTION:

An Employeemanagementsystem is designed to simplify the process of record maintenance of employees in an organization. It helps in managing the information of employees for proper functioning of the organisation. In general, employeemanagementsystem is a part of a comprehensive Human Resource ManagementSystem.

An employeemanagementsystem consists of crucial work-related and important personal information about an employee. In a nutshell, it is an inventory of all employees of an organization.

EmployeeManagementsystem is an application that enables users to create and store Employee Records. This project simplifies the task of maintain records because of its user friendly nature.

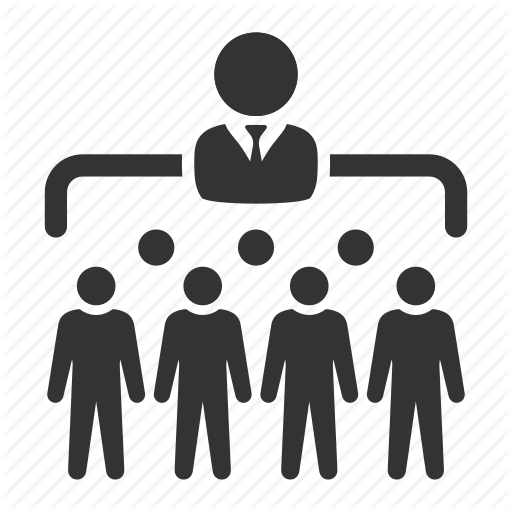
Better performance as a team and an easier way to manage admin tasks are just a couple of positive outcomes from using employee management systems.

The key is to find a system that not only works well for your team, but one that also saves your time, headaches, and money. Any employee management system should cover essential HR features.

It is basically a employee management system helps you be more efficient with your time, and reduces the time your team spends on repetitive, administrative tasks.

This centralized insight is what every good employee management system should offer. Plus, you should also look for a system that provides:

* Optimized workload and streamlined admin tasks
* Data security
* Increased employee engagement



EMPLOYEE MANAGEMENT ICON

OBJECTIVE

In this world of growing technologies everything has been computerized. With large number of work opportunities, the Human workforce has increased. Thus there is a need of a system which can handle the data of such a large number of Employees in an organization. This project simplifies the task of maintain records because of its user friendly nature.

SYSTEM SPECIFICATIONS

SOFTWARE REQUIREMENTS

* Developing Language - C++ (gnu-gcc compiler)

LANGUAGE USED

C++ is a cross-platformed language that can be used to create sophisticated high-performance applications. C++ was developed by Bjarne Stroustrup at Bell Labs in 1979, as an extension to the C language. C++ gives programmers high level of control over system resources and memory. The language was updated 3 major times in 2011,2014, and 2017 to C++ 11, C++14 and C++17.

METHODS INCLUDED:

* LOGIN AS ADMIN
* LOGIN AS EMPLOYEE
* EXIT

ADMINISTRATOR LOGIN:

* USERNAME
* PASSWORD

ADMINISTRATOR MENU:

* ADD NEW EMPLOYEE
* SEARCH EMPLOYEE PROFILE
* VIEW ALL EMPLOYEE PROFILE
* UPDATE EMPLOYEE PROFILE
* DELETE EMPLOYEE PROFILE
* LOGOUT

ADD EMPLOYEE PROFILE:

* EMPLOYEE ID
* EMPLOYEE NAME
* EMPLOYEE PASSWORD
* EMPLOYEE GENDER
* EMPLOYEE QUALIFICATION
* EMPLOYEE DESIGNATION
* EMPLOYEE DEPARTMENT
* EMPLOYEE DATE OF JOINING
* EMPLOYEE DATE OF BIRTH
* EMPLOYEE ADDRESS
* EMPLOYEE CONTACT NUMBER

SEARCH EMPLOYEE PROFILE

EMPLOYEE LOGIN:

* USERNAME
* PASSORD

EMPLOYEE MENU:

* VIEW PROFILE
* LOGOUT

**main.cpp**

#include <iostream>

#include "employee.h"

using namespace std;

int main()

{

int mainchoice, subchoice1, subchoice2;

string Empid, Name, Password, Quf, Dsgn, Dept, Gender, dojoin, dobrth, Address, Contactno;

int pass;

string emp\_id;

try

{

mainMenu:

cout << "\n\n\t\t\t"<< "WELCOME TO J.K.PVL EMPLOYEE DATABASE"<<"\n\n";

cout << "###########################################################################################\n"

<< "\n\n\t\t\t\t[1].LOGIN AS ADMIN\n\n"

<< "\t\t\t\t[2].LOGIN AS EMPLOYEE\n\n"

<< "\t\t\t\t[3].EXIT\n\n"

<< "Enter your choice :";

try

{

cin >> mainchoice;

if ((mainchoice != 1) && (mainchoice != 2) && (mainchoice != 3))

{

throw 10;

}

}

catch (int x)

{

cout << "\n\n Wrong choice!Please enter a valid Number.\n\n";

return main();

}

switch (mainchoice)

{

case 1:

{

start:

cout << "#######################################################################"

<< "\n\t\t\t\t"<<"ADMINISTRATION MENU"<<"\n"

<< "#######################################################################\n";

cout<<"Enter admin password"<<"\n";

cin>>pass;

if(pass==1234)

{

subMenu:

cout << "#################################################################################";

cout << "\n\t\t\t\t"<<"ADMINISTRATION MENU"<<"\n";

cout << "#################################################################################\n"

<< "\n\n\t\t\t\t[1] ADD NEW EMPLOYEE\n\n"

<< "\t\t\t\t[2] VEIW ALL EMPLOYEE PROFILE\n\n"

<< "\t\t\t\t[3] SEARCH EMPLOYEE PROFILE\n\n"

<< "\t\t\t\t[4] UPDATE EMPLOYEE PROFILE\n\n"

<< "\t\t\t\t[5] DELETE EMPLOYEE PROFILE\n\n"

<< "\t\t\t\t[6] LOGOUT\n\n"

<< "Enter your choice :";

try

{

cin >> subchoice1;

if ((subchoice1 != 1) && (subchoice1 != 2) && (subchoice1 != 3) && (subchoice1 != 4) && (subchoice1 != 5) && (subchoice1 != 6))

{

throw 11;

}

}

catch (int y)

{

cout << "\n\nWrong choice!Please enter a valid Number.\n\n";

goto subMenu;

}

switch (subchoice1)

{

case 1:

{

cout<<"Enter Employee UserId:\n";

cin >> Empid;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter employee Name:\n";

cin >> Name;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter employee Password:\n";

cin >> Password;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter employee Qualification:\n";

cin >> Quf;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter employee Designation:\n";

cin >>Dsgn;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter employee Department:\n\n ";

cin >> Dept;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter employee Gender:\n\n ";

cin >> Gender;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter employee date of join:\n\n ";

cin >> dojoin;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter employee date of birth:\n\n ";

cin >> dobrth;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter employee Address:\n\n "; //address change to city

cin >> Address;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

contno:

cout << "\n\nEnter employee Contact No\n\n ";

getline(cin,Contactno);

if (Contactno.size() > 10 || Contactno.size() < 10)

{

cout << "\n\t\t!!!!Please Enter Valid Contact Number:!!!!" << endl;

goto contno;

}

admin ad;

ad.add\_emp(Empid, Name, Password, Quf, Dsgn, Dept, Gender, dojoin, dobrth, Address, Contactno);

goto subMenu;

}; // Sub choice first case

break;

case 2:

{

admin ad1;

ad1.View\_all\_records();

goto subMenu;

};

break;

case 3:

{

cout<<"Enter employeeId of the employee whose record has to be searched\n";

cin>>emp\_id;

admin ad2;

ad2.Search\_emp\_record(emp\_id);

goto subMenu;

};

break;

case 4:

{

cout<<"Enter employeeId of the employee whose record has to be updated\n";

cin>>emp\_id;

admin ad3;

ad3.Update\_emp\_record(emp\_id);

goto subMenu;

};

break;

case 5:

{

cout<<"Enter employeeId of the employee whose record has to be deleted\n";

cin>>emp\_id;

admin ad4;

ad4.Delete\_emp\_record(emp\_id);

goto subMenu;

};

break;

case 6:

goto mainMenu;

break;

} //sub switch one

} //if ends here

else

{

cout<<"Oops!!Wrong Password";

goto start;

}

break;

}; //main choice first case

case 2:

{

subMenu2:

cout << "\n\n\t\t\t"<<"EMPLOYEE MENU"<<"\n\n";

cout << "#################################################################################\n"

<< "\n\n\t\t\t\t[1] VEIW PROFILE\n\n"

<< "\t\t\t\t[2] LOGOUT\n\n";

try

{

cout<<"enter your choice: ";

cin >> subchoice2;

if ((subchoice2 != 1) && (subchoice2 != 2))

{

throw 12;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

}

}

catch (int y)

{

cout << "\n\nWrong choice!Please enter a valid Number.\n\n";

goto subMenu2;

}

switch (subchoice2)

{

case 1:

{

employee e;

e.Login();

goto subMenu2;

};

break;

case 2:

goto mainMenu;

break;

} // second sub switch end

};

break;

case 3:

{

cout << "\n\nYou have logged out successfully!!\n\n";

exit(0);

};

break;

} // main switch end

}

catch (...)//Generic catch block(can catch anything)

{

cout << "\n\nUnexpected Error occoured, Program is terminating!!\n\n";

exit(0);

}

return 0;

}

**employee.h**

#include<iostream>

#include<fstream>

#include<string.h>

#include "admin\_opt.h"

using namespace std;

class employee : public admin

{

public:

int Login();

void View\_emp\_profile(string log\_customerid)

{

string Empid, Name, Password, Quf, Dsgn, Dept, Gender, dojoin, dobrth, Address, Contactno;

ifstream fin;

fin.open("employeedata.txt", ios::in);

while (fin >> Empid >> Name >> Password >> Quf >> Dsgn >> Dept >> Gender >> dojoin >> dobrth >>

Address >> Contactno)

{

if(Empid == log\_customerid)

{

cout << "\n\nEmployee ID: "<<Empid<<"\n";

cout << "\n\nEmployee Name: "<<Name<<"\n";

cout << "\n\nEmployee Password: "<<Password<<"\n";

cout << "\n\nEmployee Qualification: "<<Quf<<"\n";

cout << "\n\nEmployee Designation: "<<Dsgn<<"\n";

cout << "\n\nEmployee Department: "<<Dept<<"\n";

cout << "\n\nGender: "<<Gender<<"\n";

cout << "\n\nEmployee date of join: "<<dojoin<<"\n";

cout << "\n\nEmployee date of birth: "<<dobrth<<"\n";

cout << "\n\nEmployee Address: "<<Address<<"\n";

cout << "\n\nEmployee Contact Number: "<<Contactno<<"\n";

}

} // end of while

fin.close();

}

};

int employee ::Login()

{

int x=0; //to detect correct username and password for login

string userid, pswd;

cout << "===================================================================================="

<< "\n\t\t\t\t"<<" EMPLOYEE LOGIN "<<"\n"

<< "====================================================================================\n"

<< "Enter your UserId:\n\n";

cin >> userid;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter your Password:\n\n";

cin >> pswd;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

ifstream fread;

fread.open("employeedata.txt", ios::in);

//int s1, retype;

string Empid, Name, Password, Quf, Dsgn, Dept, Gender, dojoin, dobrth, Address, Contactno, retype;

while (fread >> Empid >> Name >> Password >> Quf >> Dsgn >> Dept >> Gender >> dojoin >> dobrth >>

Address >> Contactno)

{

if (userid==Empid && pswd==Password)

{

x++;

retype = Empid;

}

}

fread.close(); //end of while

if(x>0)

{

cout <<"Wrong UserID or Password.Try Again!!";

View\_emp\_profile(retype);

}

else

{

cout<< "\n\nLog in Failed!\n\n";

return 1;

}

return 0;

}

**admin\_opt.h**

#include <bits/stdc++.h>

using namespace std;

class admin

{

protected:

string eEmpid;

public:

string eName, eDsgn, eDept, eQuf, edobrth, edojoin, eGender, ePassword, eContactno, eAddress;

admin() {}

void add\_emp(string eEmpid, string eName, string ePassword, string eQuf, string eDsgn, string eDept,

string eGender, string edojoin, string edobrth, string eAddress, string eContactno);

void View\_all\_records();

void Search\_emp\_record(string);

void Update\_emp\_record(string);

void Delete\_emp\_record(string);

~admin()

{

}

};

void admin ::add\_emp(string Empid, string Name, string Password, string Quf, string Dsgn, string Dept,

string Gender, string dojoin, string dobrth, string Address, string Contactno)

{

eEmpid = Empid;

eName = Name;

ePassword = Password;

eQuf = Quf;

eDsgn = Dsgn;

eDept = Dept;

eGender = Gender;

edojoin = dojoin;

edobrth = dobrth;

eAddress = Address;

eContactno = Contactno;

fstream fout;

fout.open("employeedata.txt", ios::app | ios::out | ios::ate);

fout << eEmpid << "\t" << eName << "\t" << ePassword << "\t" << eQuf << "\t" << eDsgn << "\t" << eDept

<< "\t" << eGender << "\t" << edojoin << "\t" << edobrth << "\t" << eAddress << "\t" << eContactno << "\n";

fout.close();

}

void admin ::View\_all\_records()

{

string Empid, Name, Password, Quf, Dsgn, Dept, Gender, dojoin, dobrth, Address, Contactno;

ifstream fin;

fin.open("employeedata.txt", ios::in);

cout << "=====================================================================================================================================\n";

cout<<"\t\t\t\t\t"<<"VIEW ALL EMPLOYEE PROFILE"<<"\t\t\t\t\t\n";

cout << "======================================================================================================================================\n";

cout<<"EID NAME\tPSWD\t QLF.\t DESIGN.\t DPMT\t SEX\t DOJ\t DOB\t ADDRESS\t PH\_NO\n";

cout << "=======================================================================================================================================\n";

while (fin >> Empid >> Name >> Password >> Quf >> Dsgn >> Dept >> Gender >> dojoin >> dobrth >>

Address >> Contactno)

cout<<Empid<<" "<<Name<<"\t"<<Password<<" "<<Quf<<" "<<Dsgn<<" "<<Dept<<" "<<Gender<<" "<<dojoin

<<"\t"<<dobrth<<" "<<Address<<" "<<Contactno<<"\n\n";

fin.close();

}

void admin ::Search\_emp\_record(string log\_customerid)

{

string Empid, Name, Password, Quf, Dsgn, Dept, Gender, dojoin, dobrth, Address, Contactno;

ifstream empfin;

empfin.open("employeedata.txt", ios::in);

while (empfin >> Empid >> Name >> Password >> Quf >> Dsgn >> Dept >> Gender >> dojoin >> dobrth >>

Address >> Contactno)

{

if (Empid == log\_customerid)

{

cout << "\n\nRecord found!\n\n";

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

cout << "\n\nEmployee ID: "<<Empid<<"\n";

cout << "\n\nEmployee Name: "<<Name<<"\n";

cout << "\n\nEmployee Password: "<<Password<<"\n";

cout << "\n\nEmployee Qualification: "<<Quf<<"\n";

cout << "\n\nEmployee Designation: "<<Dsgn<<"\n";

cout << "\n\nEmployee Department: "<<Dept<<"\n";

cout << "\n\nGender: "<<Gender<<"\n";

cout << "\n\nEmployee date of join: "<<dojoin<<"\n";

cout << "\n\nEmployee date of birth: "<<dobrth<<"\n";

cout << "\n\nEmployee Address: "<<Address<<"\n";

cout << "\n\nEmployee Contact Number: "<<Contactno<<"\n";

}

} // end of while

empfin.close();

}

// Delete record function

void admin::Delete\_emp\_record(string log\_customerid)

{

int counter=0;

string Empid, Name, Password, Quf, Dsgn, Dept, Gender, dojoin, dobrth, Address, Contactno;

ifstream fin;

ofstream fout;

fin.open("employeedata.txt", ios::in);

while (fin >> Empid >> Name >> Password >> Quf >> Dsgn >> Dept >> Gender >> dojoin >> dobrth >> Address >> Contactno)

{

fout.open("temp.txt", ios ::out | ios :: app);

if(Empid != log\_customerid)

{

fout << Empid << "\t" << Name << "\t" << Password << "\t" << Quf << "\t" << Dsgn << "\t" << Dept

<< "\t" << Gender << "\t" << dojoin << "\t" << dobrth << "\t" << Address << "\t" << Contactno << "\n";

}

else

{

counter++;

}

fout.close();

} // end of while

fin.close();

if (counter == 0)

{

remove("temp.txt");

cout << "\n\nRecord not found\n\n";

}

if (counter > 0)

{

remove("employeedata.txt");

rename("temp.txt", "employeedata.txt");

cout<<"1 record is Deleted successfully!!\n";

}

} //end of function Delete employee Record

// Search file and update

void admin ::Update\_emp\_record(string log\_customerid)

{

int counter1 = 0;

string Empid, Name, Password, Quf, Dsgn, Dept, Gender, dojoin, dobrth, Address, Contactno;

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

ifstream PassengerIn;

ofstream writeFile;

PassengerIn.open("employeedata.txt", ios::in);

while (PassengerIn >> Empid >> Name >> Password >> Quf >> Dsgn >> Dept >> Gender >> dojoin >> dobrth >>

Address >> Contactno)

{

writeFile.open("temp1.txt", ios ::app);

if (Empid == log\_customerid)

{

cout << "\n\nRecord found!\n\n";

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

cout << "\n\nEnter changed employee Name:\n\n ";

cin >> eName;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter changed employee Password:\n\n "; //address change to city

cin >> ePassword;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter changed employee Qualification:\n\n "; //address change to city

cin >> eQuf;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter changed employee Designation:\n\n "; //address change to city

cin >> eDsgn;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter changed employee Department:\n\n "; //address change to city

cin >> eDept;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter changed employee Gender:\n\n "; //address change to city

cin >> eGender;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter changed employee date of join:\n\n "; //address change to city

cin >> edojoin;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter changed employee date of birth:\n\n "; //address change to city

cin >> edobrth;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "\n\nEnter changed employee Address:\n\n "; //address change to city

cin >> edobrth;

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

contact:

cout << "\n\nEnter changed employee Contact No\n\n ";

getline(cin,eContactno);

if (eContactno.size() > 10 || eContactno.size() < 10)

{

cout << "\n\t\t!!!!Please Enter Valid Contact Number:!!!!" << endl;

goto contact;

}

writeFile << Empid << "\t" << eName << "\t" << ePassword << "\t" << eQuf << "\t" << eDsgn

<< "\t" << eDept << "\t" << eAddress << "\t" << eGender << "\t"

<< edojoin << "\t" << edobrth << "\t" << eAddress << "\t" << eContactno << "\n";

counter1++;

}

else

{

writeFile << Empid << "\t" << Name << "\t" << Password << "\t" << Quf << "\t" << Dsgn << "\t" << Dept

<< "\t" << Gender << "\t" << dojoin << "\t" << dobrth << "\t" << Address << "\t" << Contactno << "\n";

}

writeFile.close();

} // end of while

PassengerIn.close();

if (counter1 == 0)

{

cout << "\n\nRecord could not be found!\n\n";

remove("temp1.txt");

}

if (counter1 > 0)

{

remove("Passenger.txt");

rename("temp1.txt", "Passenger.txt");

cout<<"UPDATED SUCCESSFULLY!!\n";

}

}

Admin

Datamembers:

String eEmpid, eName, eDsgn, eDept, eQuf, edobrth, edojoin, eGender, ePassword, eContactno, eAddress

Member functions:

1. void add\_emp(string eEmpid, string eName, string ePassword, string eQuf, string eDsgn, string eDept,string eGender, string edojoin, string edobrth, string eAddress, string eContactno)
2. void View\_all\_records()
3. void Search\_emp\_record(string)
4. void Update\_emp\_record(string)
5. void Delete\_emp\_record(string)

Employee

Member functions:

1.string Login()

2. void View\_emp\_profile(string log\_customerid)

Admin-

1. void add\_emp(string eEmpid, string eName, string ePassword, string eQuf, string eDsgn, string eDept,string eGender, string edojoin, string edobrth, string eAddress, string eContactno) -Helps entry of employee details
2. void View\_all\_records()-Helps to display details of all employees
3. void Search\_emp\_record(string)-Helps in searching employee details
4. void Update\_emp\_record(string)-Helps to change employee details
5. void Delete\_emp\_record(string)-Helps to delete the employee details of one employee

Employee-

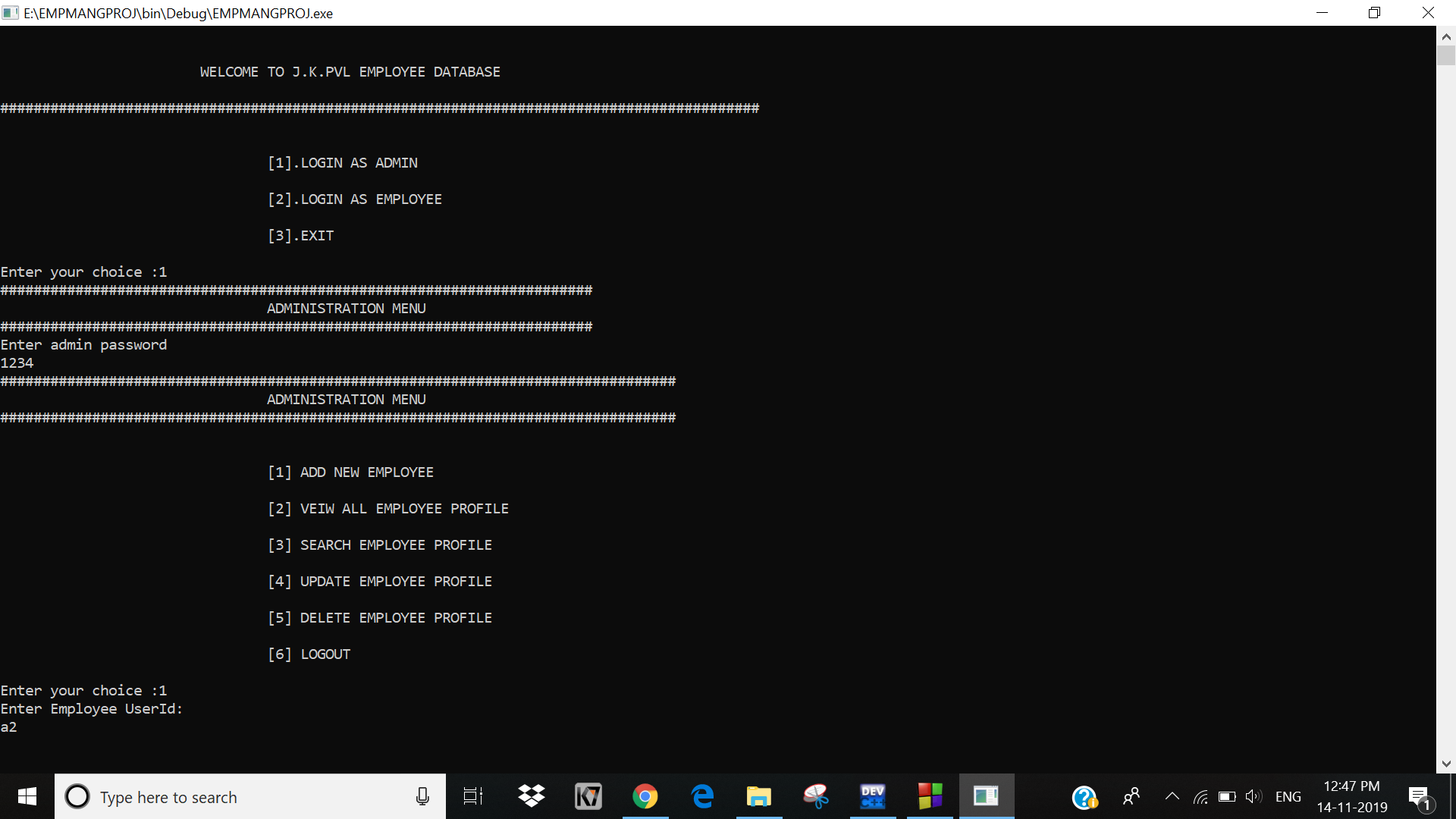
1.string Login()-Helps employee to login into his/her account.

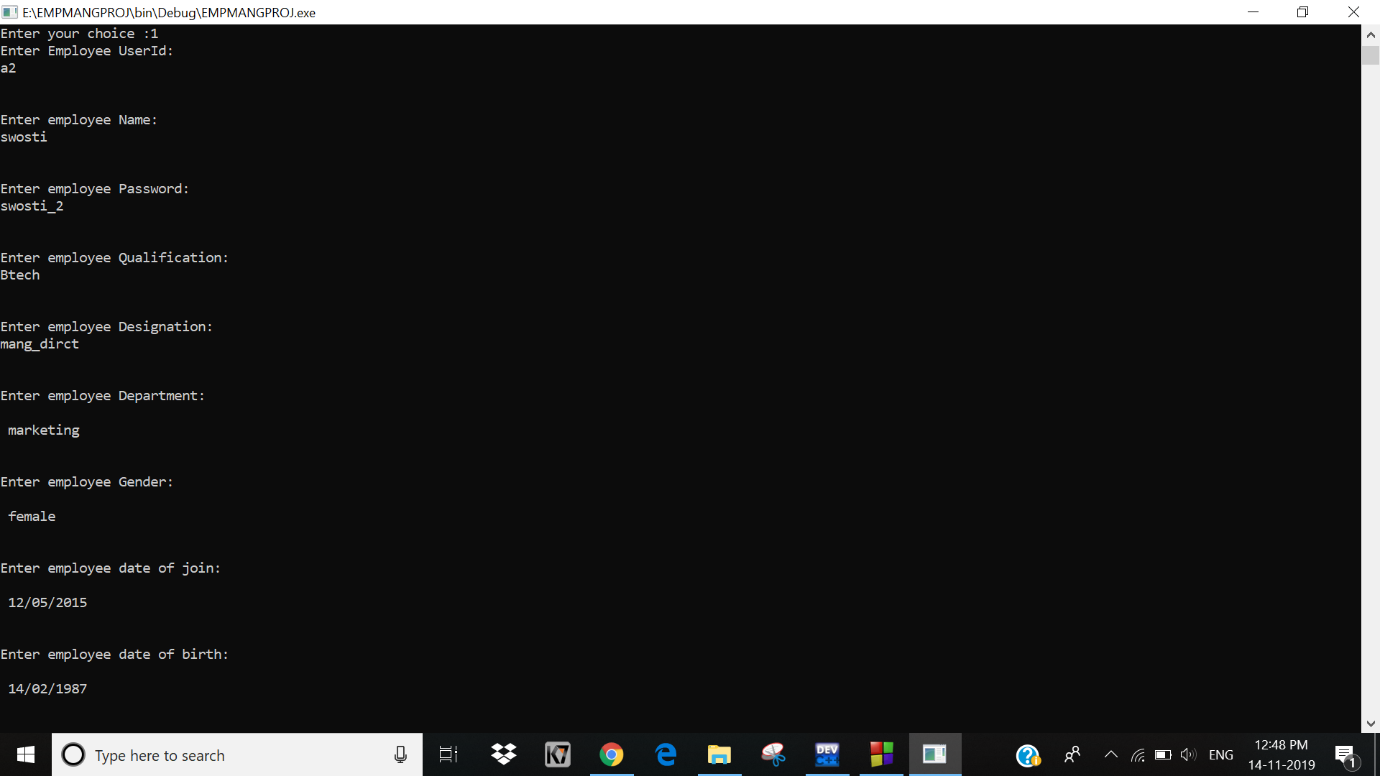
2. void View\_emp\_profile(string log\_customerid)-Helps the employee to view own profile

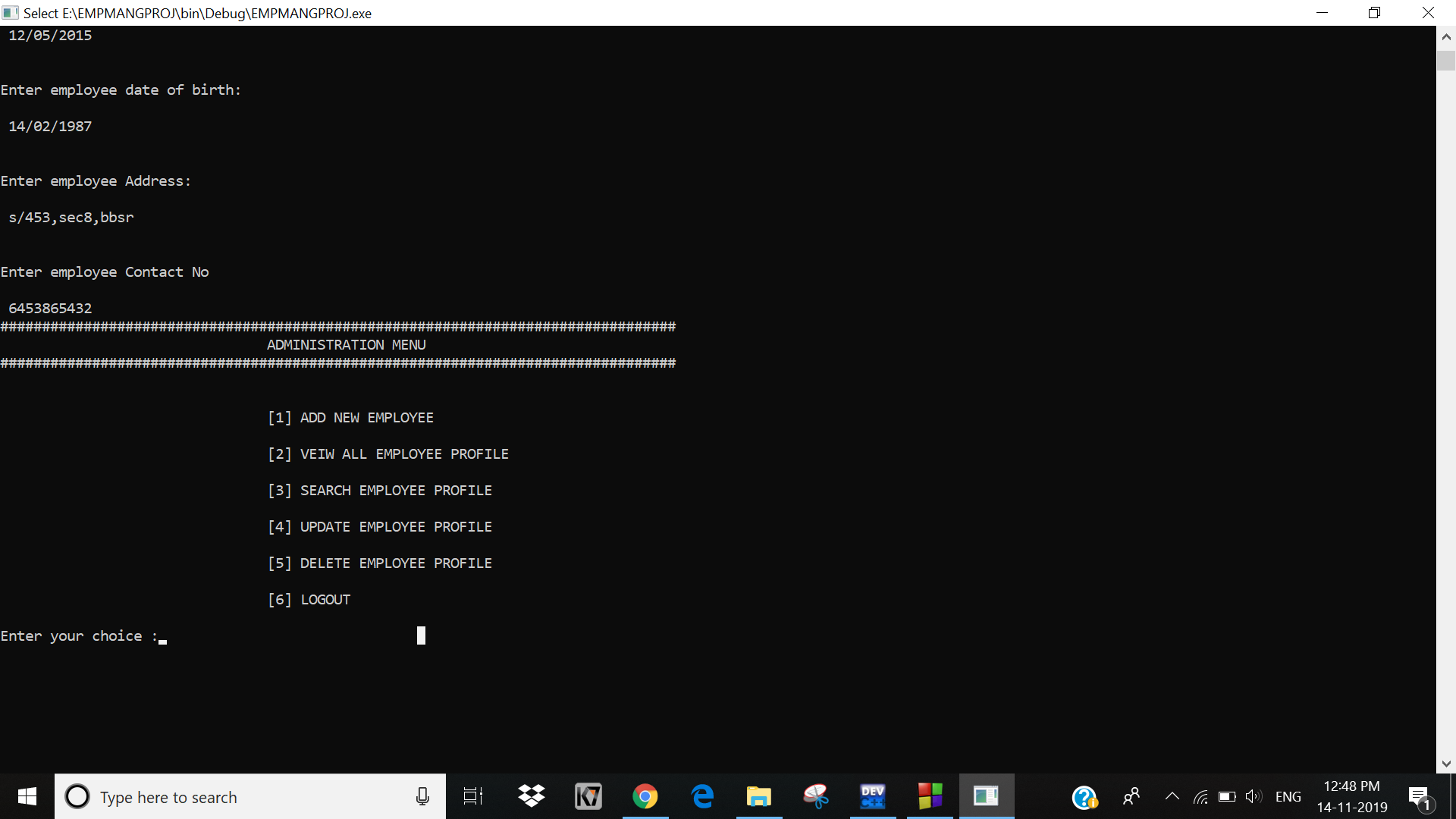
Header files used in the project:

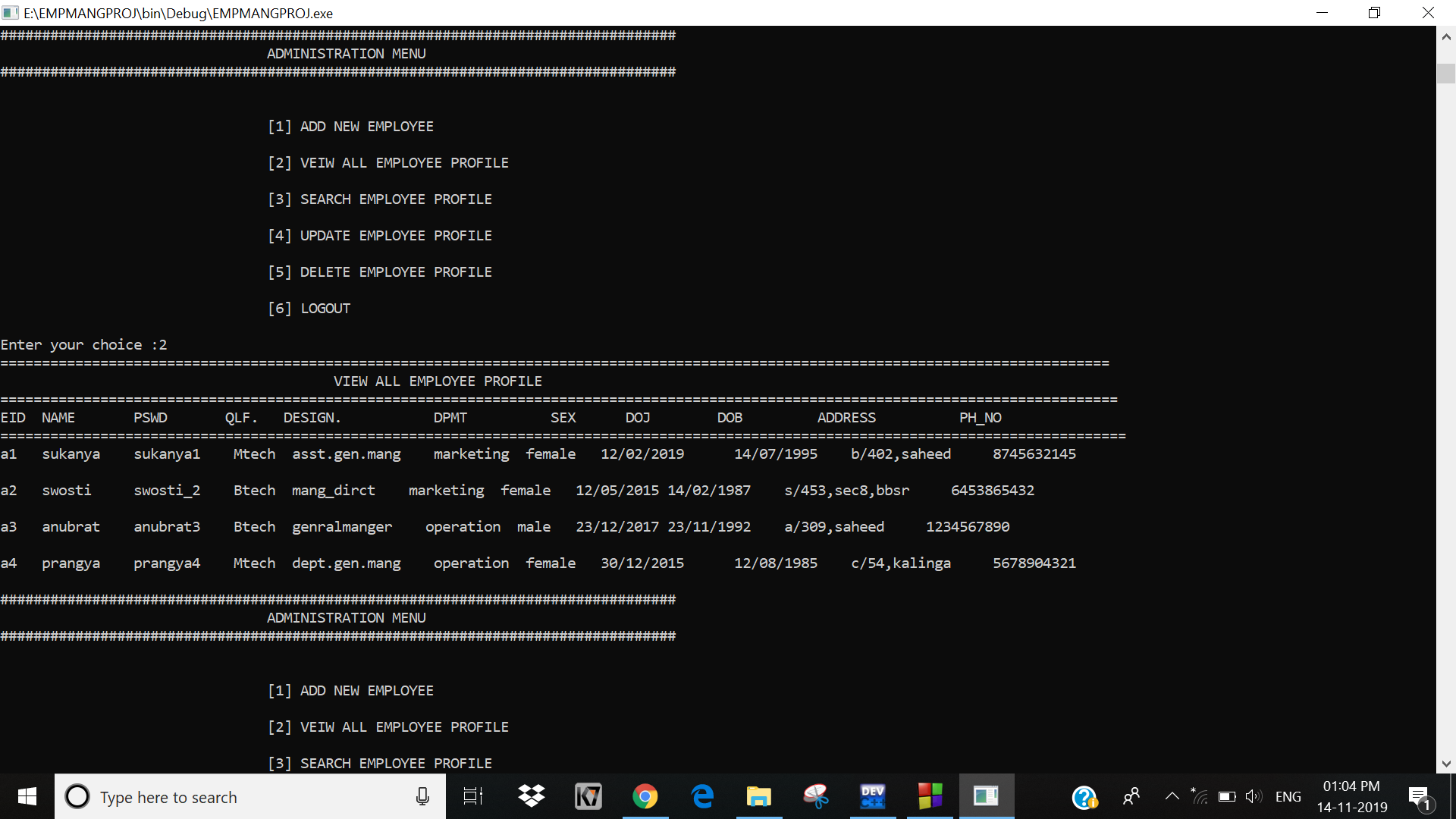
* admin\_opt.h
* employee.h
* iostream
* string.h
* fstream
* bits/stdc++.h

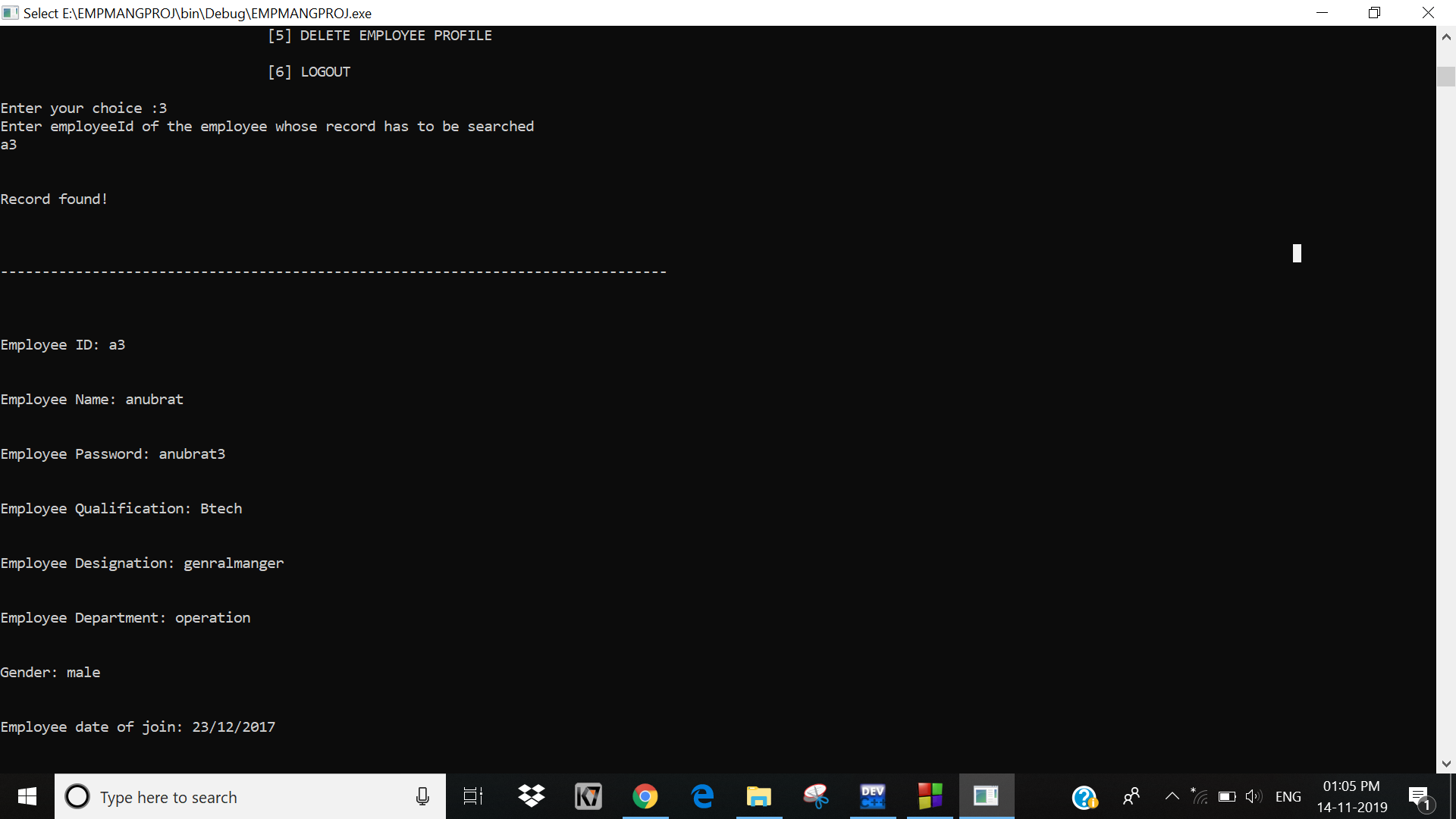
**OUTPUT SCREENSHOTS**

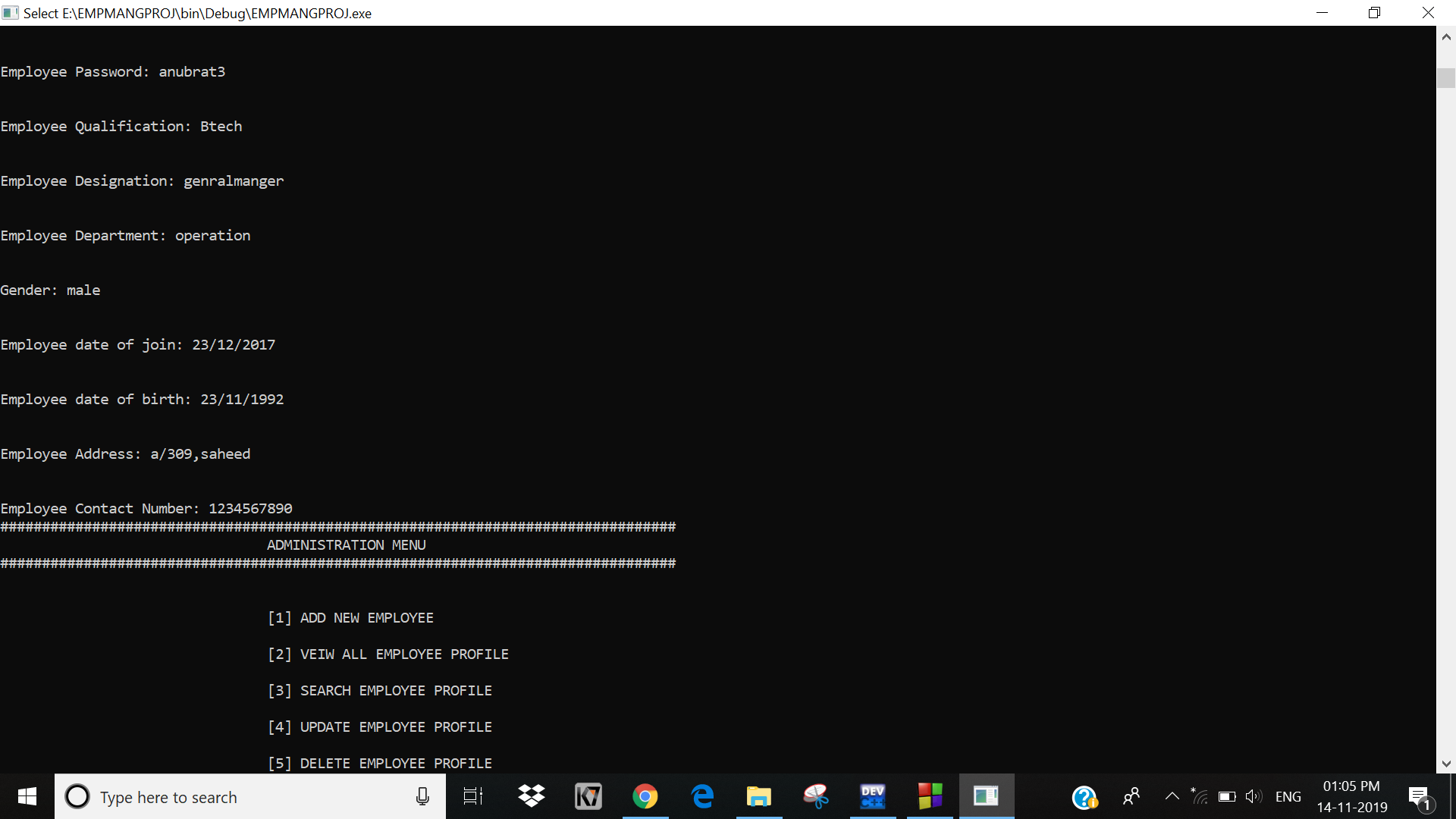
****

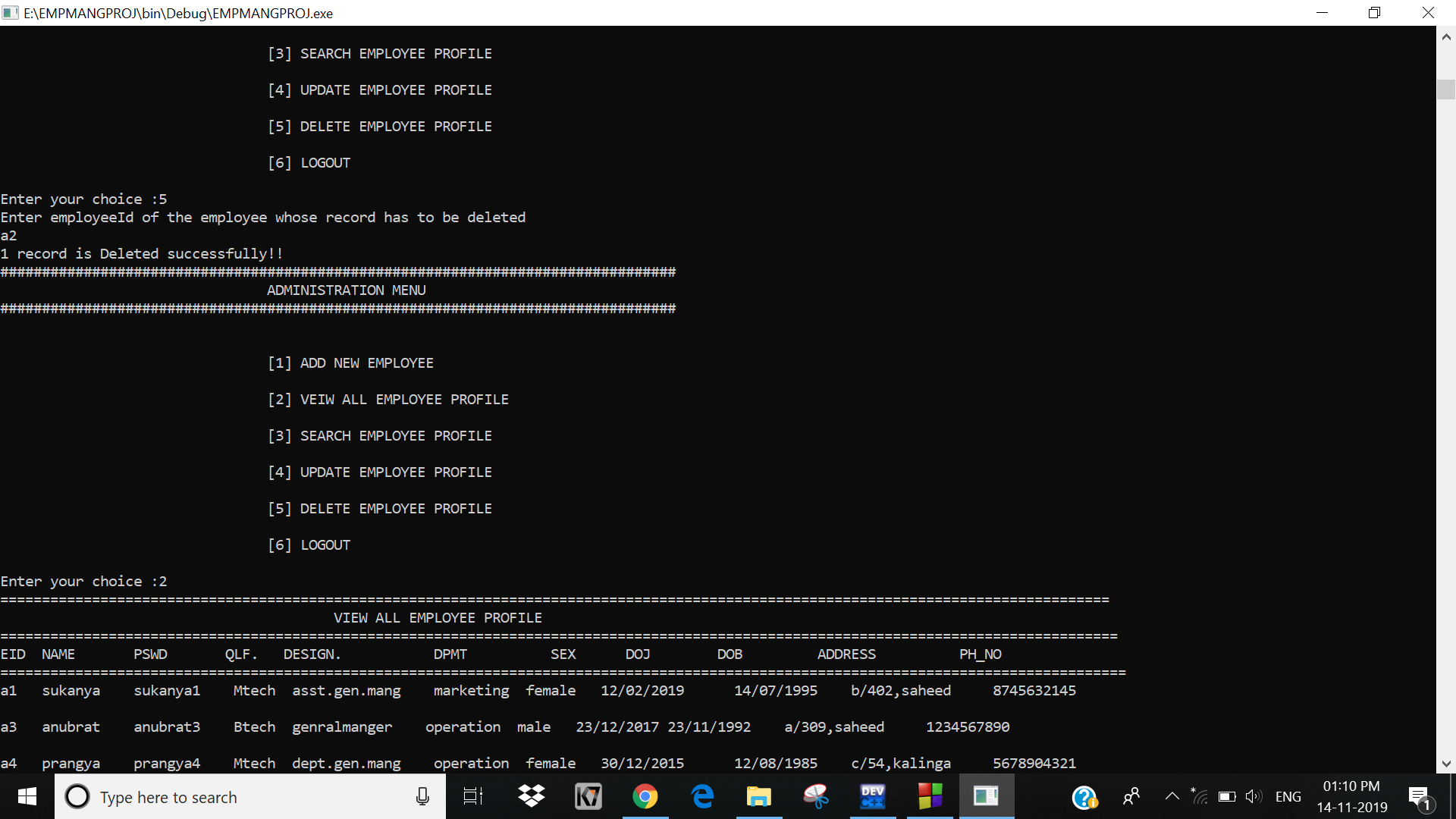
****

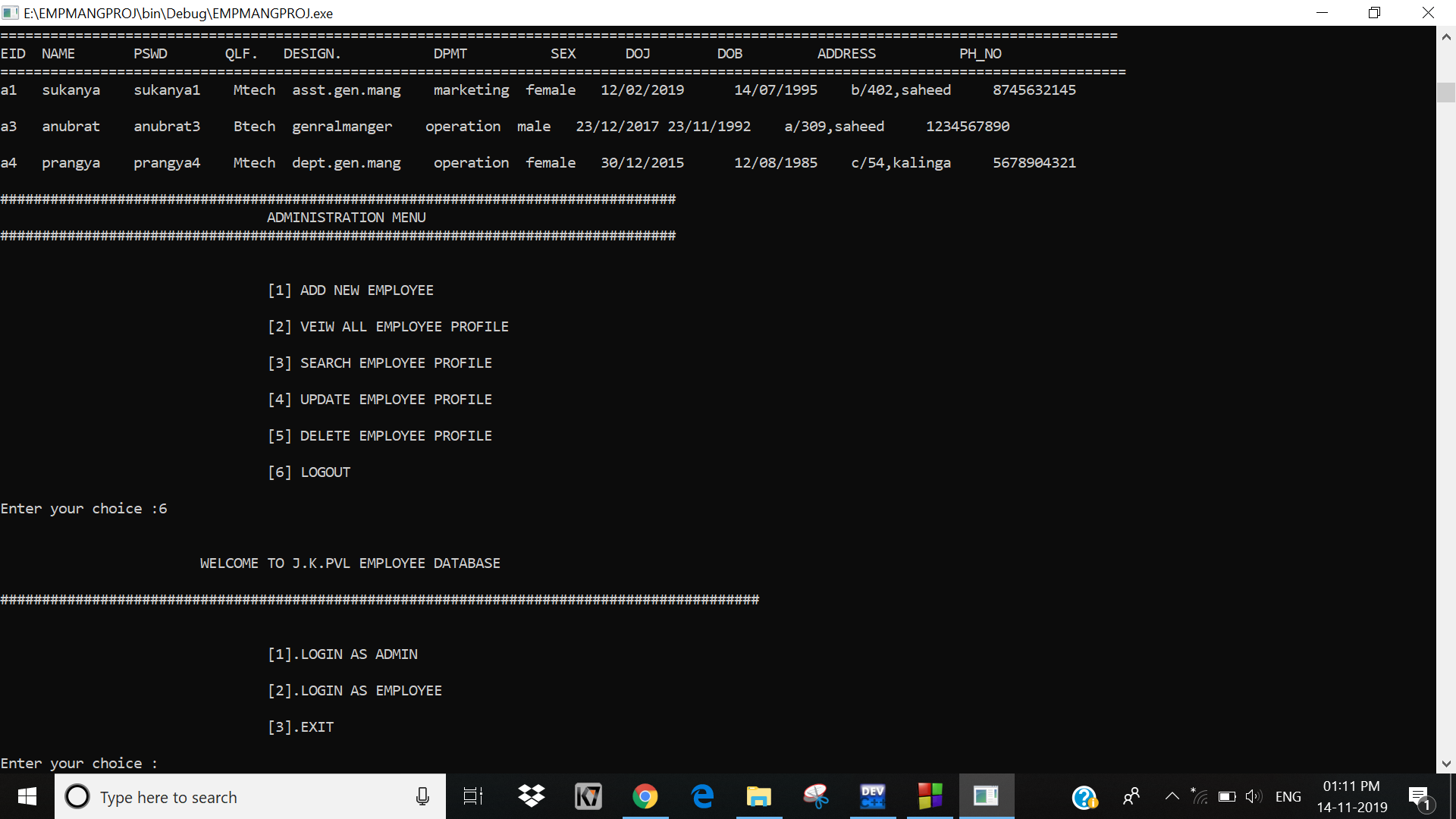
****

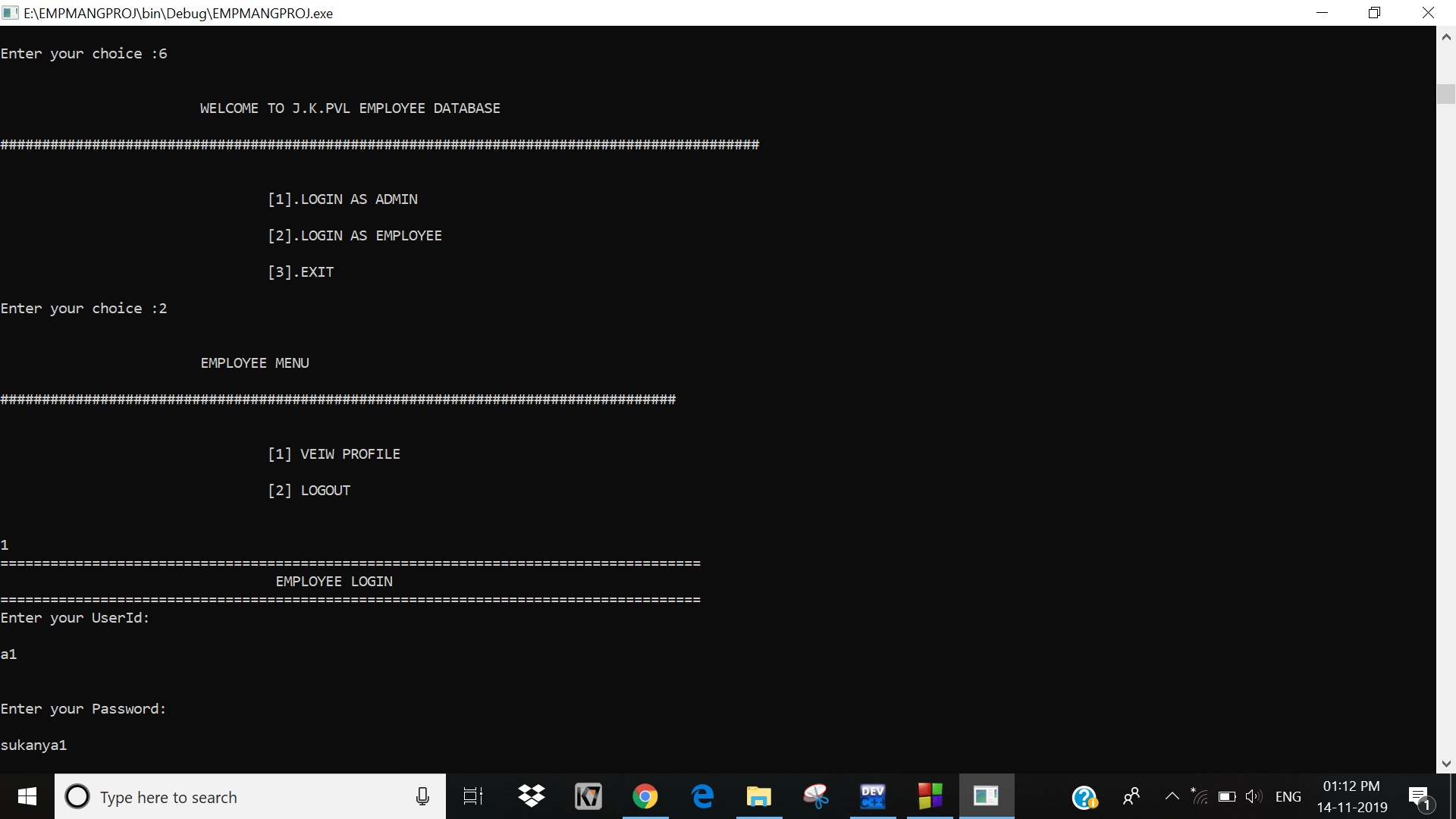
****

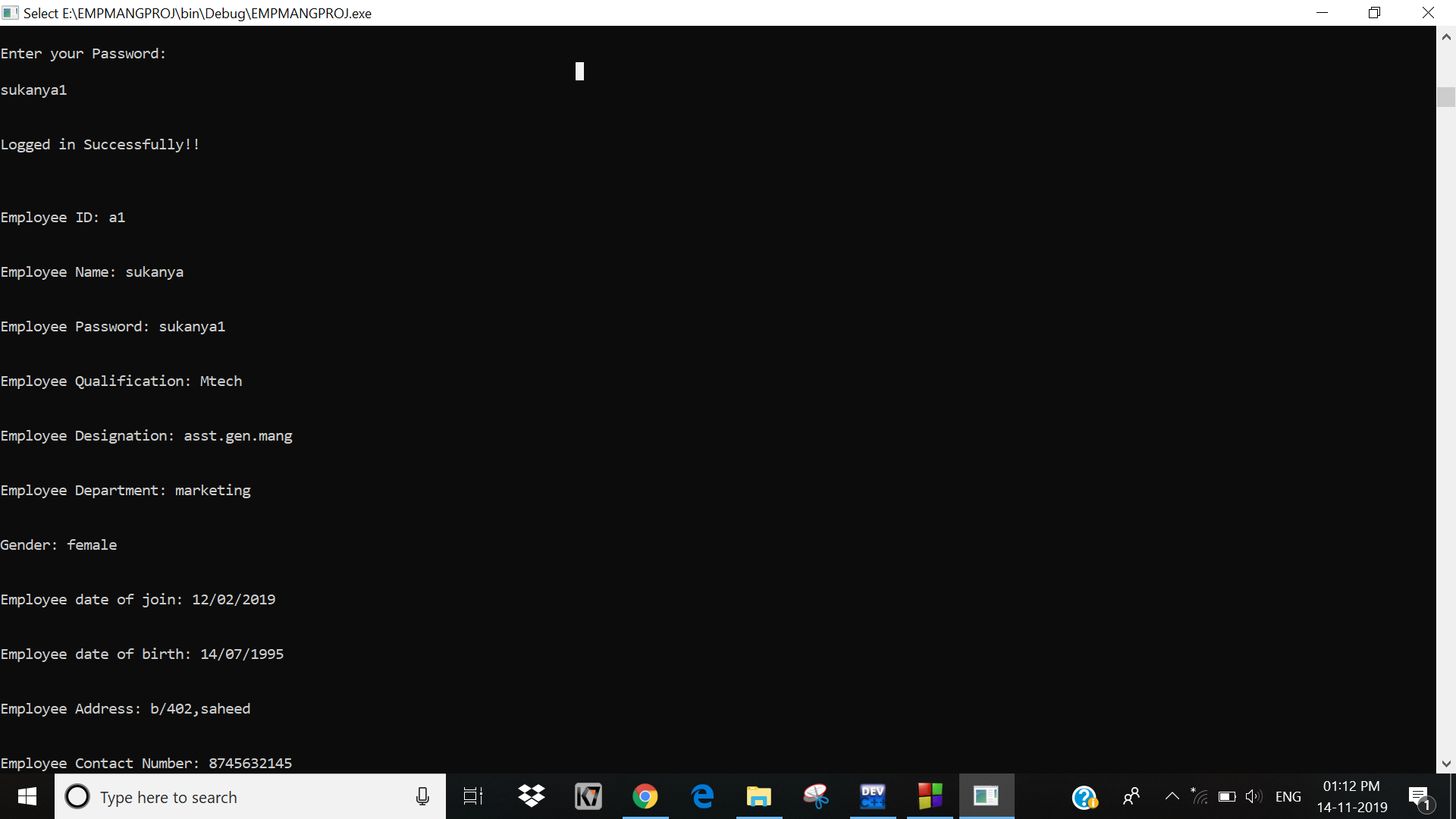
****

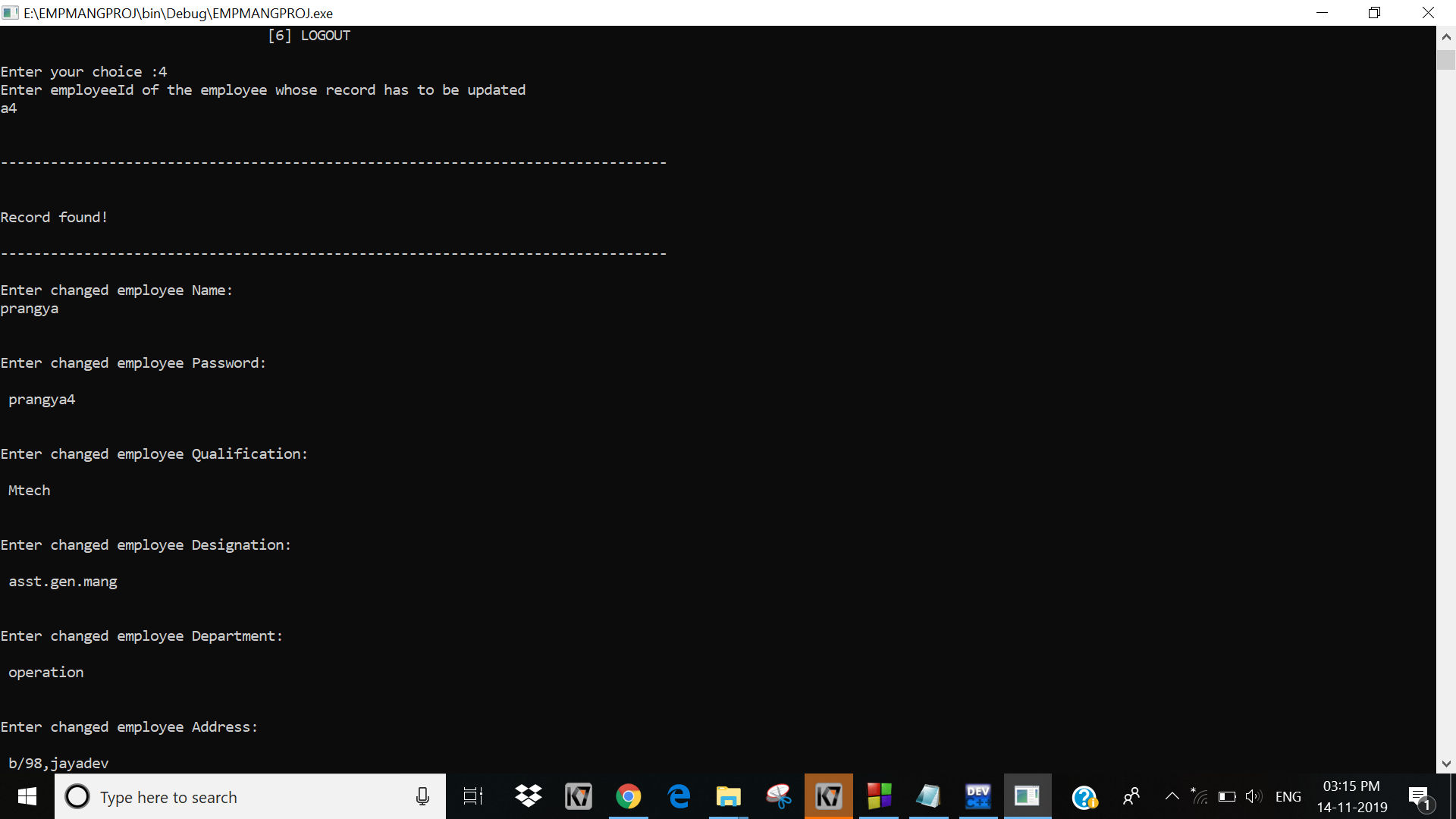
****

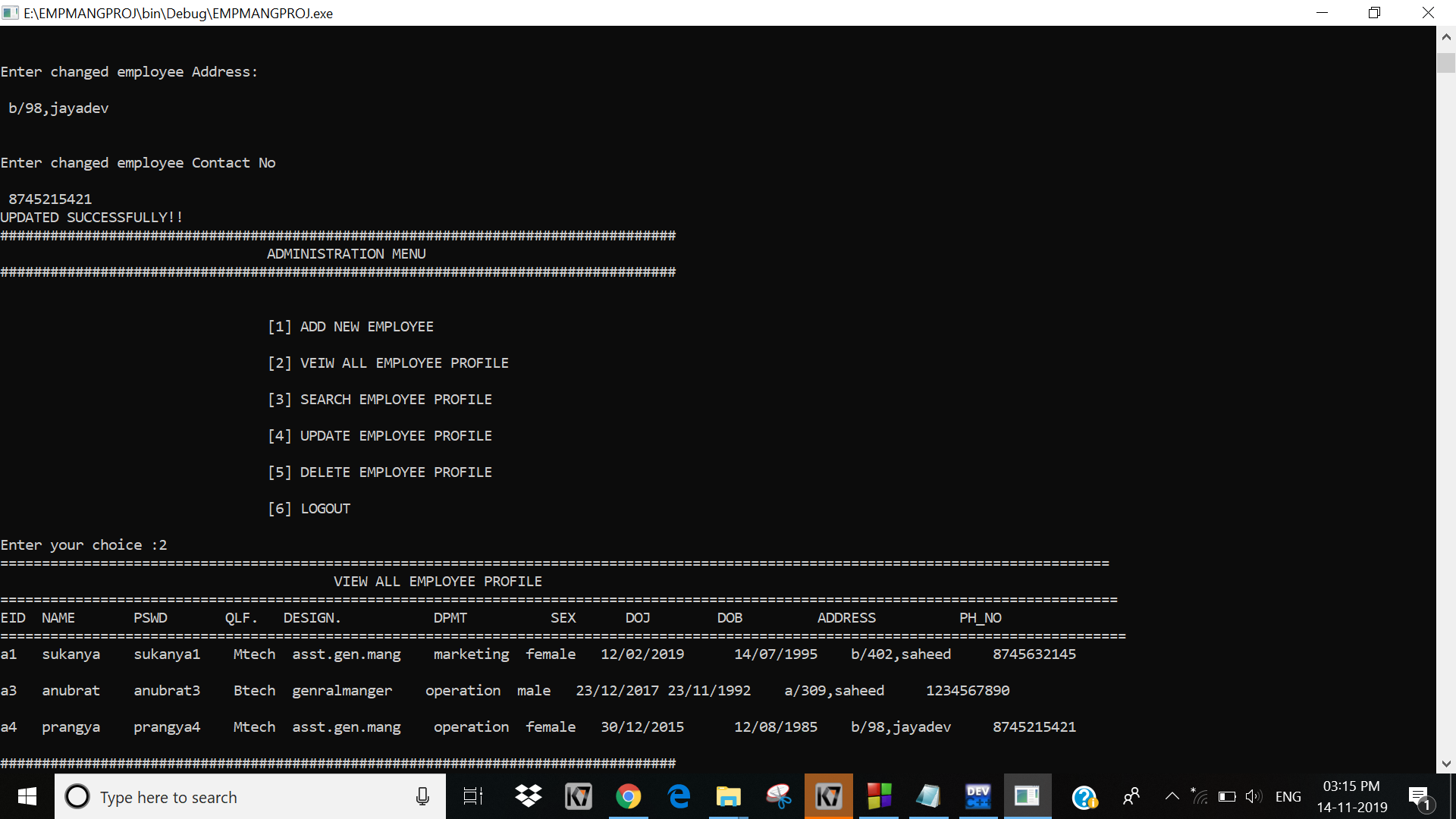
****

****

****

**-------------------**

****

****

CONCLUSION

From this project we conclude that the project is helpful for management of personal details of employee as well as proper functioning of the organisation. The main advantage of this project is time management. Hence, it is the most needed and successful system in present days.