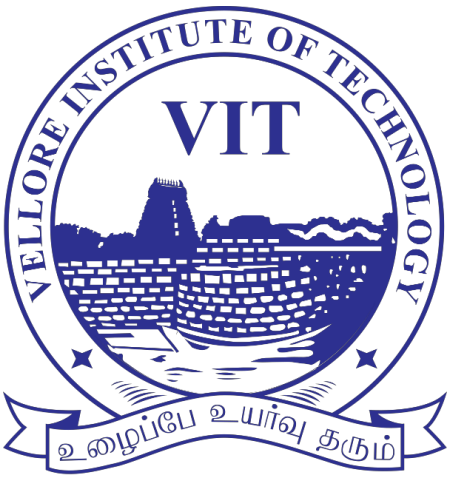


Object Oriented Analysis and Design

WINTER SEMESTER

2019-20 - PROJECT

REPORT

(VIT – EBMS)
VIT – ELECTRONIC BOARD MANAGEMENT SYSTEM

NAME: VARUN SHARMA (17BIT0151)

Eleth Venkat Sai

(18BIT0395)

FACULTY: PROF.BHAVANI S.

COURSE CODE: ITE1007

SLOT: G1

AIM

To develop a project electronic board management system using rational rose software to draw diagrams and implement the command line software in C++.

PROBLEM STATEMENT

Paying bills, reporting a power outage, reporting for a need of a service traditionally was a very time and effort consuming task, now bill payment is made easier by the means of many online payment sites. But still, information about the power outage, requesting for a service or maintenance regarding electricity is still not very easy.

PROBLEM ANALYSIS AND PROJECT PLANNING

There arises a requirement for a system where everything will be stored in one place, whether it is a bill payment, maintenance report, service request, power outage details, etc. The **VIT-EBMS** provides an easy mode of electricity maintenance to general public by means of complete online electricity board maintenance system. It includes online bill payment, electricity status check, call for the maintenance or service, complaints, etc. It will also record number of power outages in a certain period and hence helping in finding the cause for the outage and reducing the outage time as more as possible.

Users will be able to manage everything regarding the electricity maintenance sitting at one place only. Employees will be assigned work based on the nearest possible area.

SOFTWARE REQUIREMENT SPECIFICATION

1. INTRODUCTION

The **VIT-EBMS** provides an easy mode of electricity maintenance to general public by means of complete online electricity board maintenance system. It includes online bill payment, electricity status check, call for the maintenance or service, complaints, etc.

2. OBJECTIVE

The main objective of VIT-EBMS is to help customers to freely interact with our system by registering themselves and apply for the complaints if any. Users will be able to manage everything regarding the electricity maintenance sitting at one place only

3. OVERVIEW

The overview of the project is to design an electronic board management system for Users who will be able to manage everything regarding the electricity maintenance sitting at one place only. Employees will be assigned work based on the nearest possible area and customers will also give ratings to each employee based on which employees will be work.

4. REQUIREMENTS –

FUNCTIONAL REQUIREMENTS –

1. USER REGISTRATION & LOGIN
   - Allows the user to register with the system.
   - Enables the user/admin to customize his/her account settings and preferences
2. VIEW BILLS AND PAYMENT STATUS
   - Enables user to view his/her electricity bill
   - Allows user to check out payment status
3. ADD/REMOVE USERS
   - Allows admin to add/remove users

4. GENERATE BILL

   • Allows the admin to generate bill for particular user

5. STORING METER READING

   • Enables the Electricity Board Employee to provide meter readings of users

6. E-MAIL/SMS NOTIFICATIONS

   • Extends the standard notifications service built into Electricity Billing system

   • Automatically delivers notifications via e-mail and/or text message

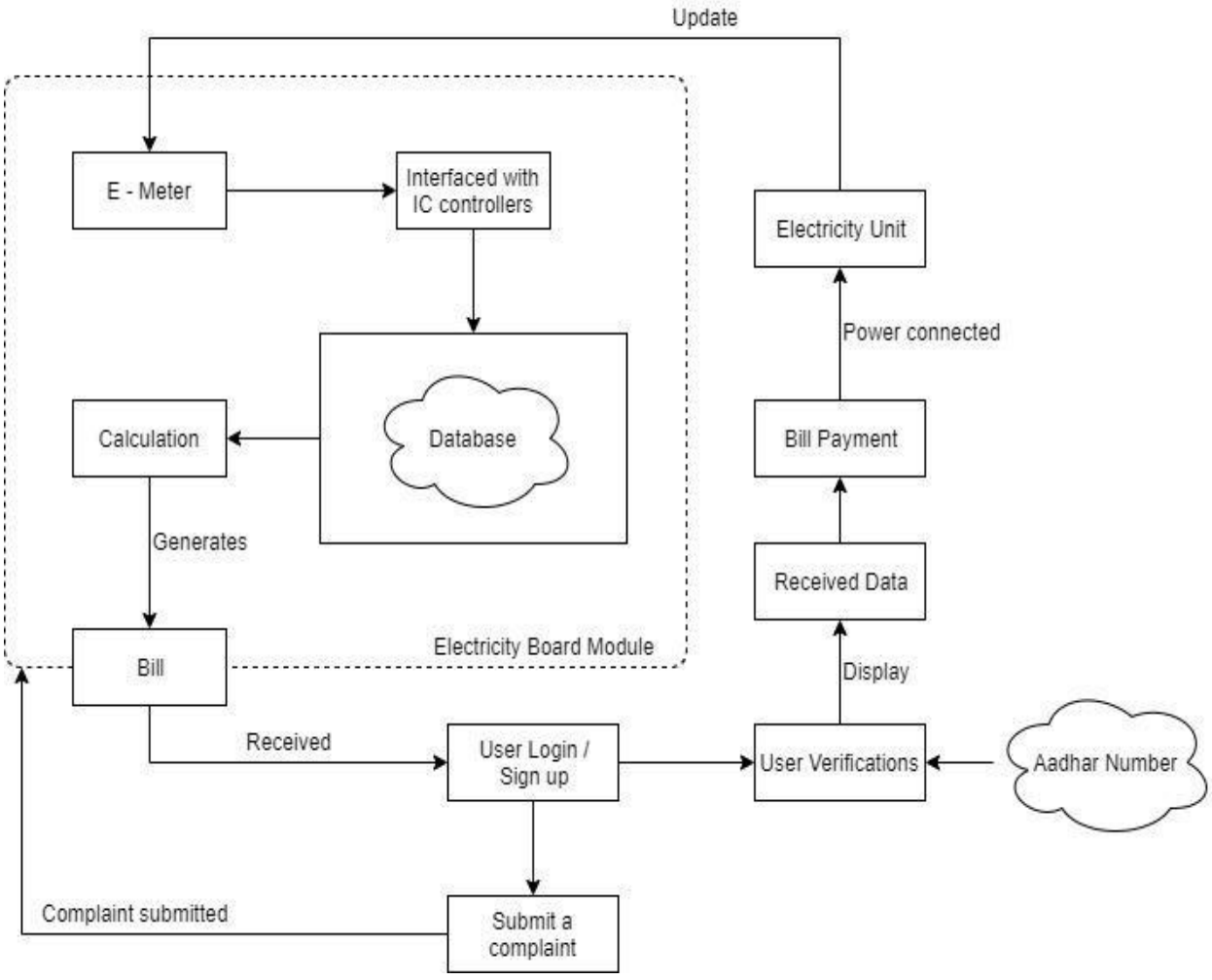
NON – FUNCTIONAL REQUIREMENTS -

1. SECURITY – Our System provides otp and password facilities. Thus, it is highly secured.
2. MAINTAINABILITY – Employees regularly maintains the connections thus making the system more maintainable.
3. FLEXIBILTY – VIT-EBMS adjusts itself based on the requirements of customers.
4. ADAPTABILTY- VIT-EBMS adapts to present scenario through analysis of complaints.
5. USABILITY – It becomes easy for customers to use this kind of user friendly system.

Proposed System Architecture: (Flow Diagram)

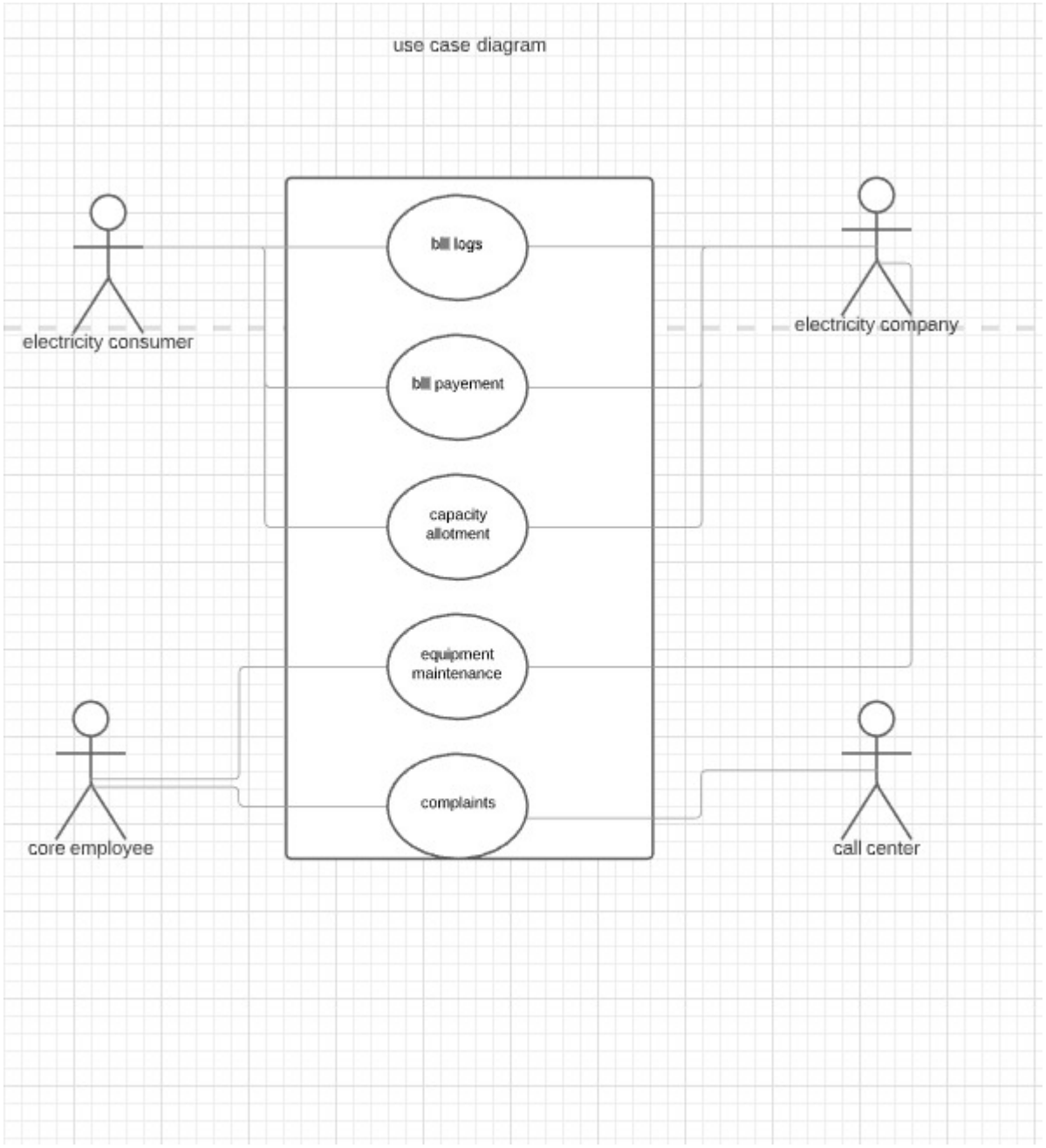
In our proposed system, users and employees registers in system monitored by admin.

Users can view their bill logs and pay their pending bills. Users are given priviledge to complain regarding their bad experiences with our system so that nearest employee could reach out in that particular area and resolve that issue. Payment has many options and is secured through otp generation.



UML DIAGRAMS –

1. Use Case Diagram:



Actors –

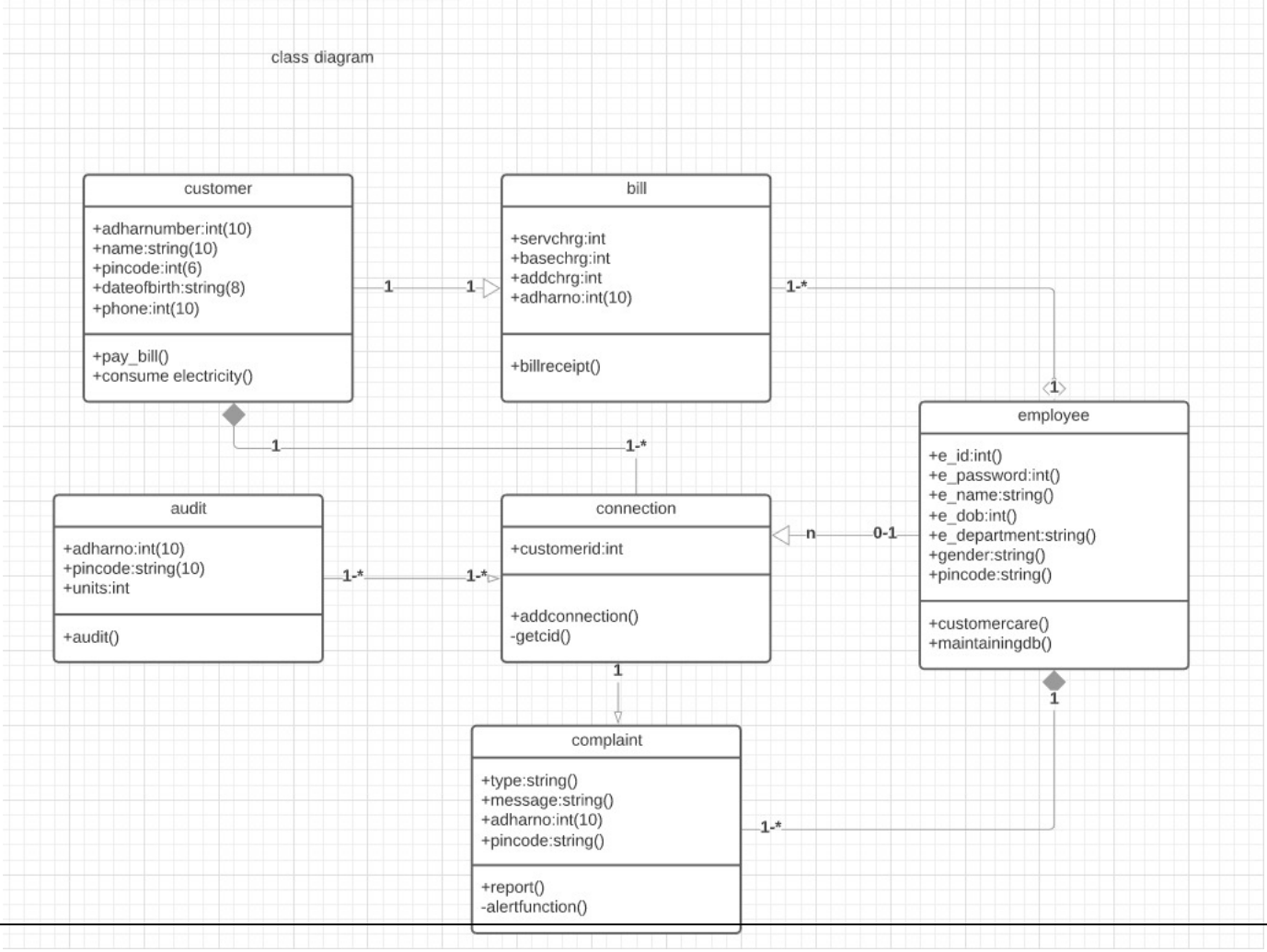
- electricity consumer (user)
- core employee
- call center
- electrical company (also admin)

Activities of Actors –

- ✓ Users pays their bills.
- ✓ Employees works when complaints come.
- ✓ Call Center employees communicates complaints to working employees.
- ✓ Electrical Company(admin) resolves the issue for bill logs, bill payments , etc.

2. Class Diagram:

A class diagram in the unified modeling language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes. It is represented using a rectangle



with three compartments. Top compartment have the class name, middle compartment the attributes and the bottom compartment with operations.

Customer -

aadhar number – cutomer's aadhar card number
name – name of customer
pincode – customer address pincode
date of birth – customer's date of birth
phone – phone number of customer

pay_bill()- customer pays the bill.
consumes_electricity() - customer uses electricity for their purposes.

Bill –

srvchrg – service charge included in bill
basechrg - base charge included in bill
addchrg - additional charge included in bill
adharno – adhar number of customer linked with bill

billreceipt() – bill log having billing details.

Audit – customers examines their billing procedure

adharno - adhar number of customer linked with audit
pincode – pincode of area
units – measure of consumption

audit() – customer examining their bill

Connection – establishing connection of new customers

customerid – unique id of customer

addconnection()- adding connection
getcid() – get customer id of connection

Employee – working employees of VIT-EBMS

e_id – employee id
e_password – employee password
e_name – employee name

e_dob – employee date of birth
e_department – employee department
gender – male/female
working area - pincode

customercare() – call center employee work assigned
maintainingdb() – software employee work assigned

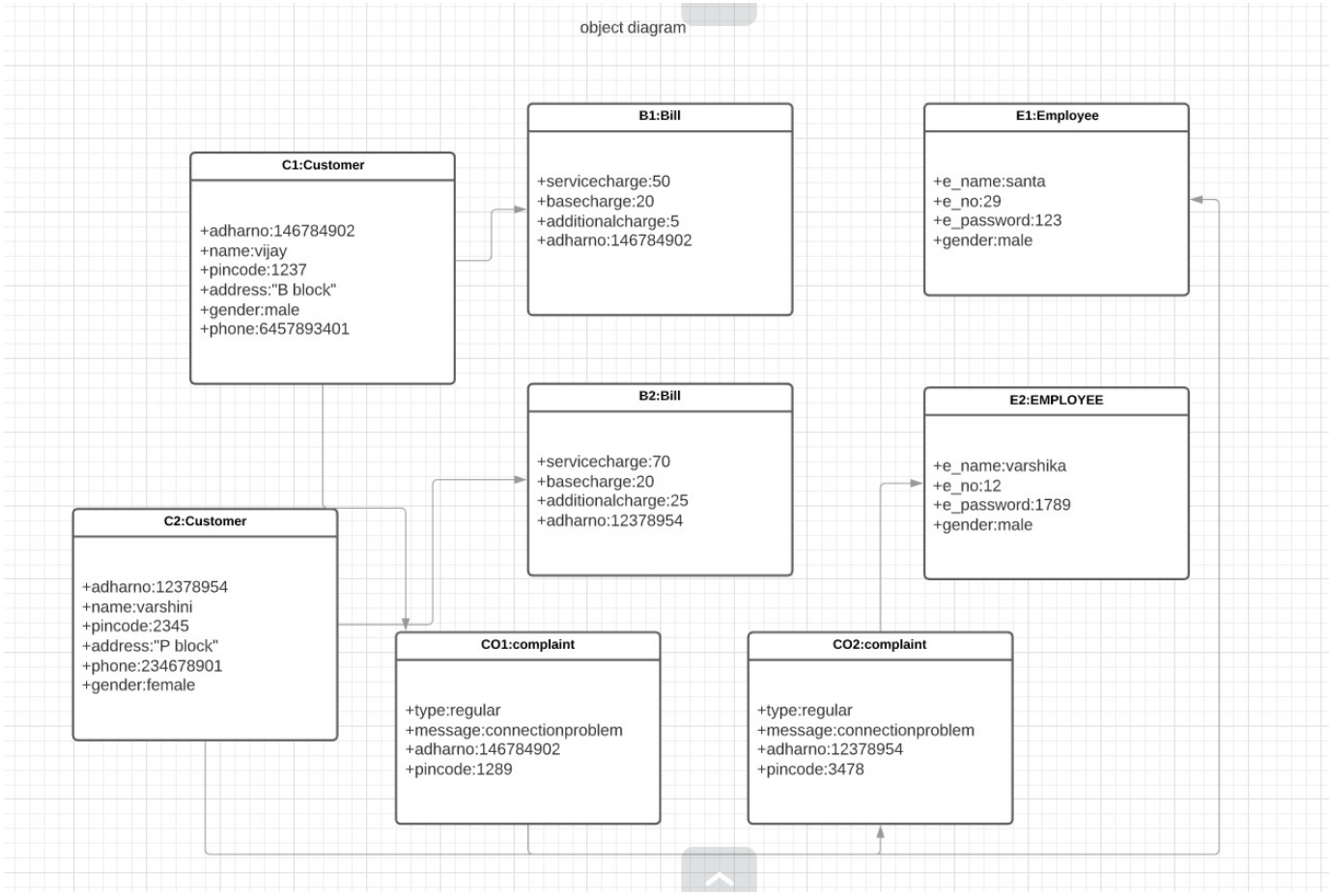
Complaint – customers having bad experiences registers here

type – type of complaint (failure, cut. etc.)
message – customer's message
adharno - customer's adhar number who had complained
pincode – locality of customer where customer has complained

report() – generates report of complaint for better user experience.
alertfunction() – generates alert message to employees working in that area

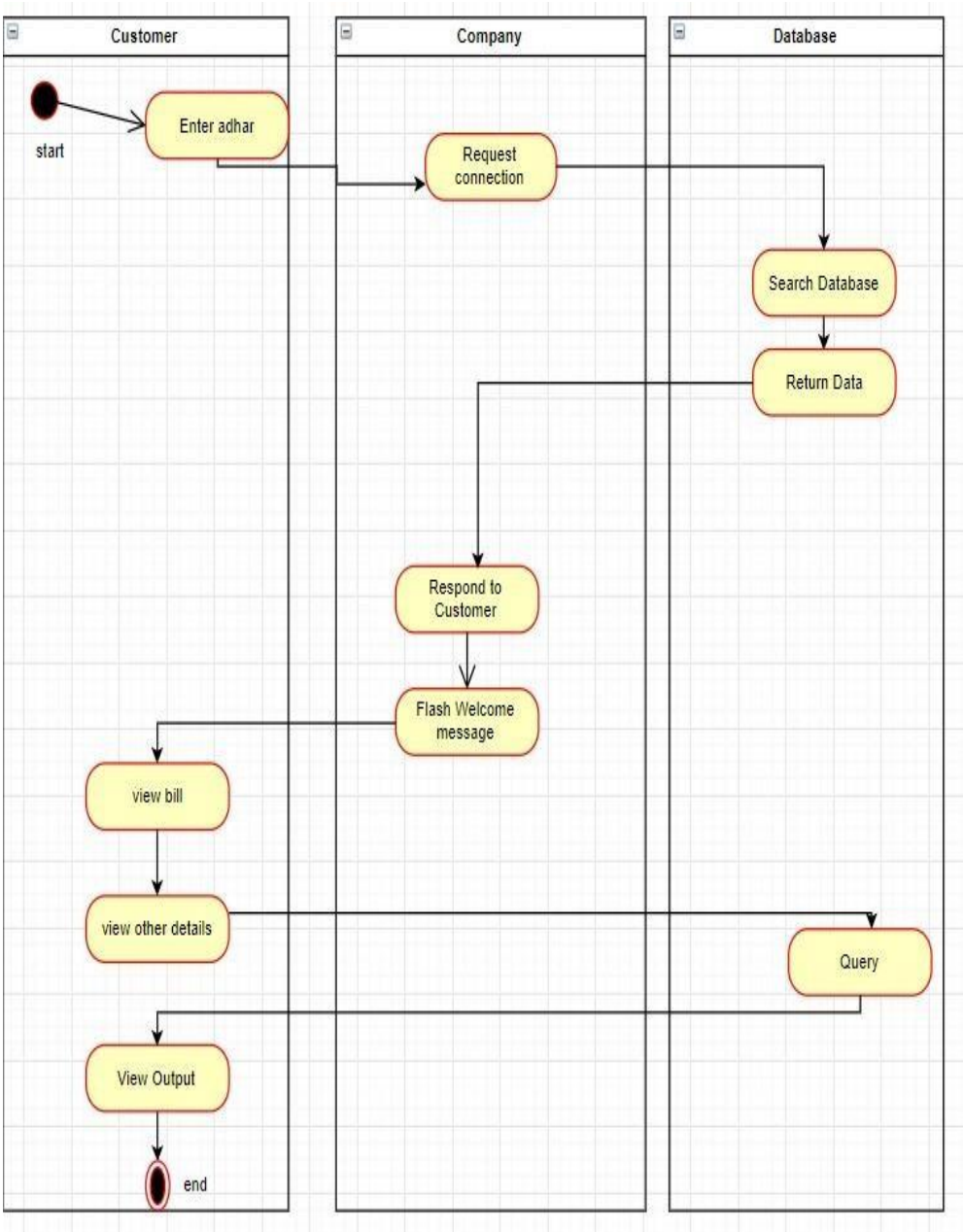
3. Object Diagram:

An object diagram is a UML structural diagram that shows the instances of the classifiers in models. Object diagrams use notation that is similar to that used in class diagrams.



4. Swimlanes Activity Diagram:

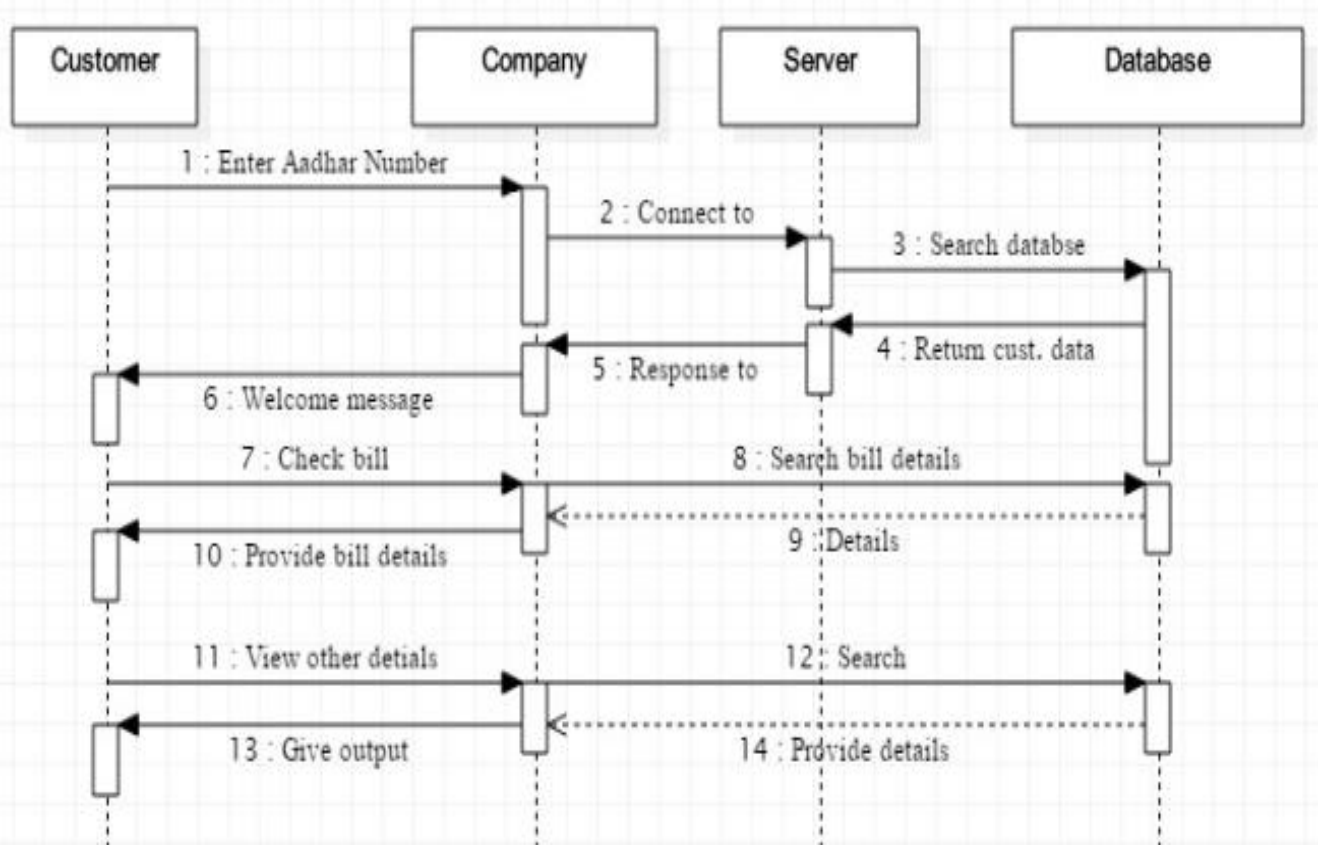
Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control. An activity is shown as an rounded box containing the name of the operation.



5. Sequence Diagram:

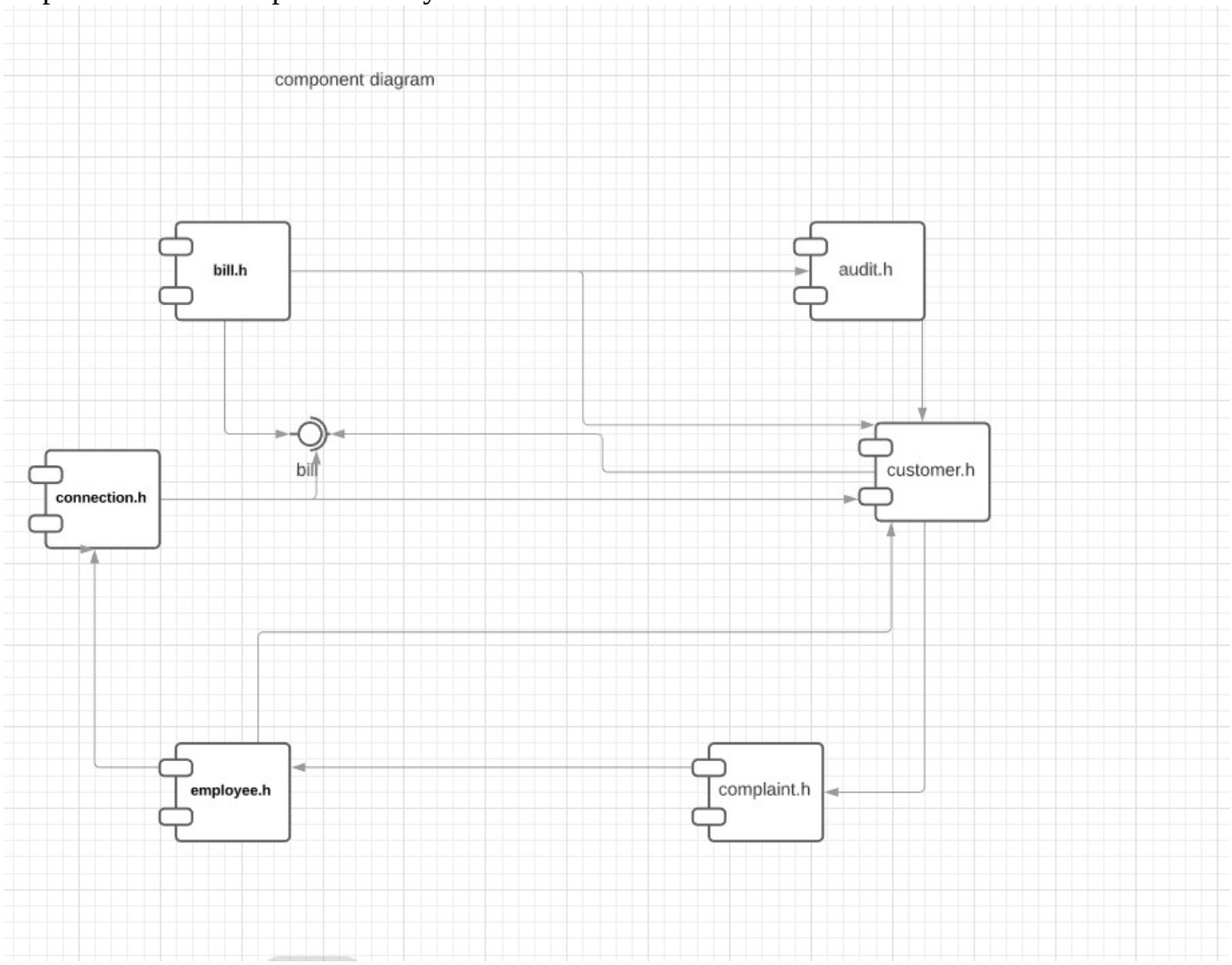
A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. There are two dimensions.

1.Vertical dimension – represent time.
2.Horizontal dimension-represent different objects.



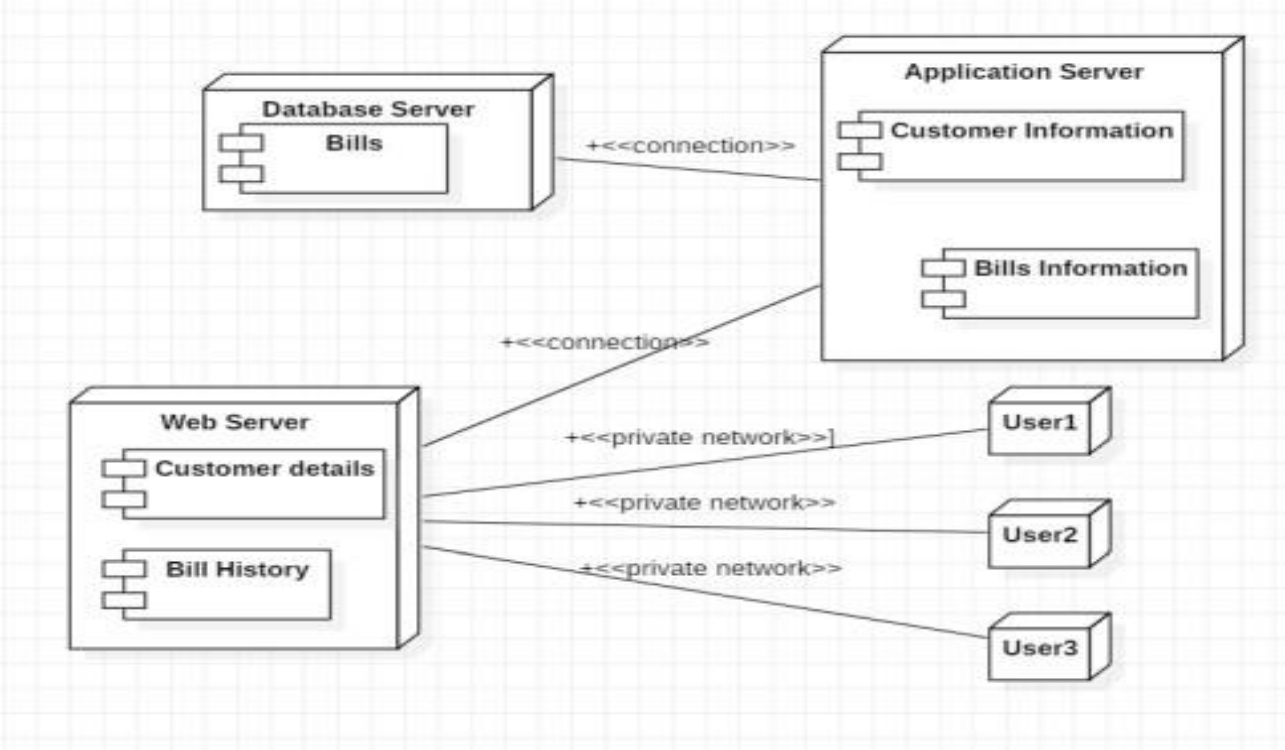
6. Component Diagram:

The components diagram main purpose is to show the structural relationships between the components of a system. It is represented by boxed figure. Dependencies are represented by communication association.



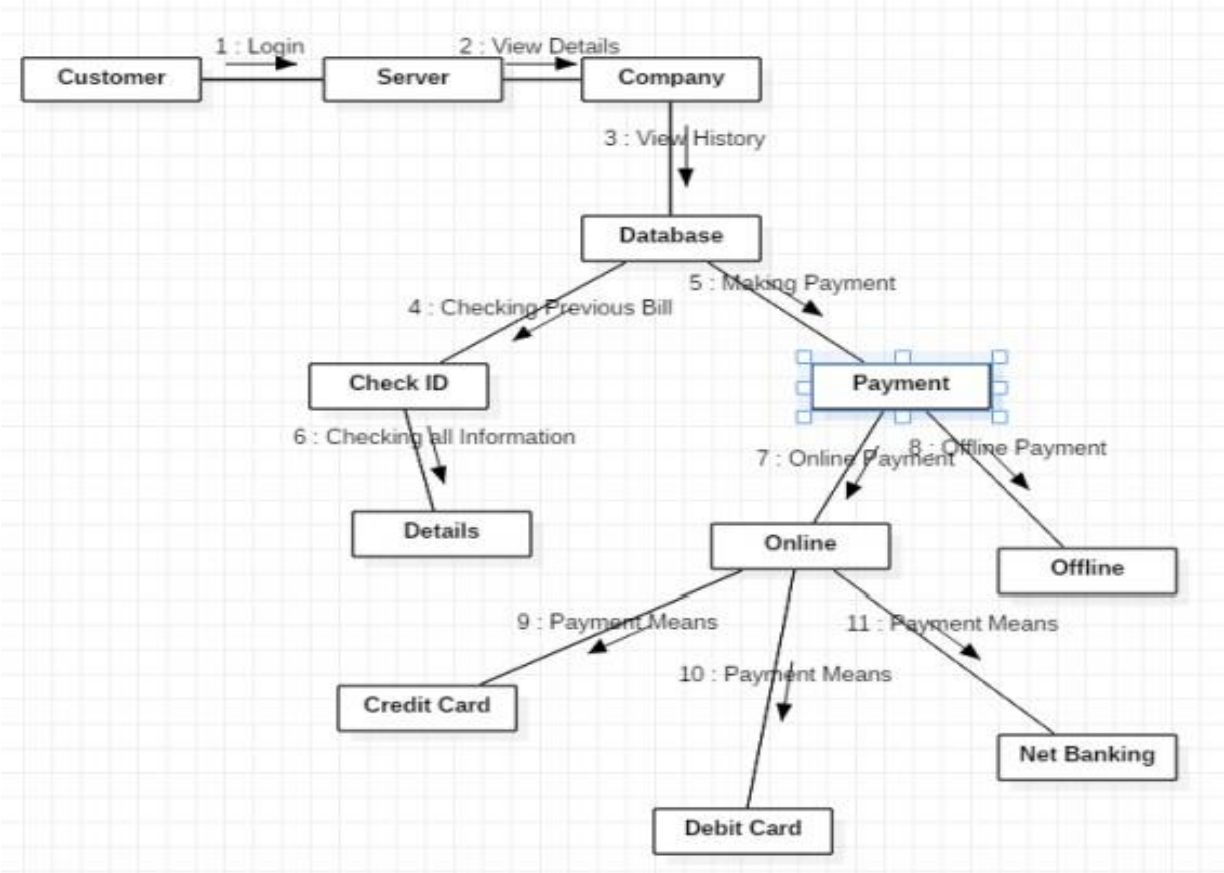
7. Deployment Diagram:

A deployment diagram in the unified modeling language serves to model the physical deployment of artifacts on deployment targets. Deployment diagrams show "the allocation of artifacts to nodes according to the Deployments defined between them. It is represented by 3-dimentional box. Dependencies are represented by communication association.



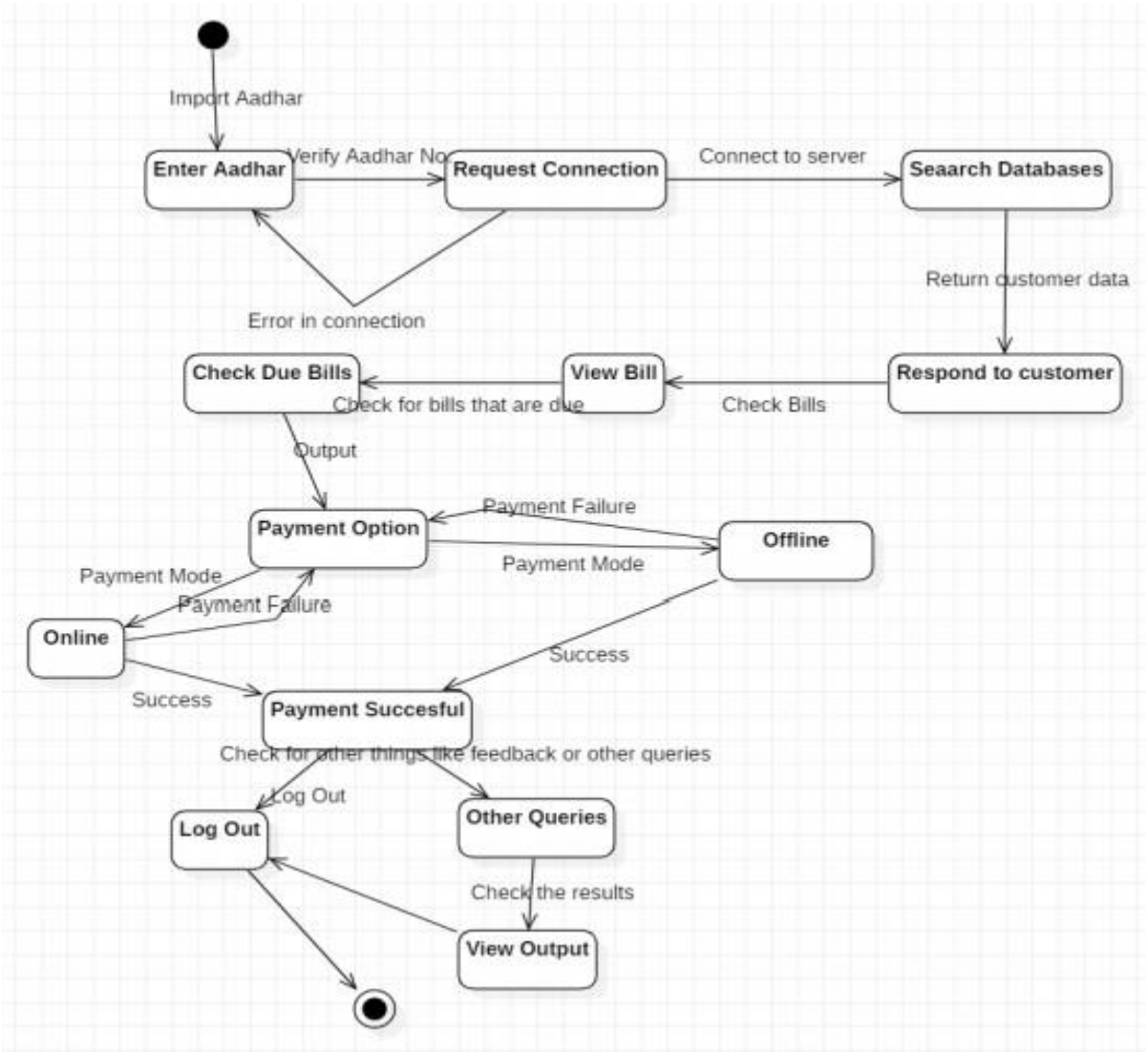
8. Collaboration Diagram:

A collaboration diagram, also called a communication diagram or interaction diagram,. A sophisticated modeling tool can easily convert a collaboration diagram into a sequence diagram and the vice versa. A collaboration diagram resembles a flowchart that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time.



9. State Machine Diagram:

The purpose of state machine diagram is to understand the algorithm involved in performing a method. It is also called as state diagram. A state is represented as a round box, which may contain one or more compartments. An initial state is represented as small dot. A final state is represented as circle surrounding a small dot.



CODE-

1. VIT_EBMS.cpp

// Run This File for output

```
#include<iostream>
#include<cstdio>
#include<cstdlib>
#include<cstring>
#include<ctype.h>
#include"connection.h"
#include"audit.h"
#include"bill.h"
#include"complaint.h"
#include"employee.h"
using namespace std;

vector<employee> employees;


int login(int emp_id,string emp_password) {

        for(int i=0;i<employees.size();i++)
   {
      if((employees[i].e_id==emp_id) &&
        compare(employees[i].e_password,emp_password))
      {
         system("cls");
                        printf("LOGIN SUCCESSFUL\n");
                        cout<<"WELCOME BACK "<<employees[i].e_name<<", \n";
                        return 1;
      }
   }

        return 0;

}

void initialise_employees()
{

employees.emplace_back(employee(1142,"varun123","varun","software","08/10/1998","9376373373","m
ale","135001"));
employees.emplace_back(employee(1112,"@sanskar1","sanskar","operations","28/02/1999","383823373"
,"male","156336"));
```

```
employees.emplace_back(employee(3931,"anish_","anish","hardware","01/11/2000","9919339391","male","334224"));


employees.emplace_back(employee(2210,"priya2210","priya","software","08/11/1998","9376993773","female","311345"));

employees.emplace_back(employee(5382,"rishabh4","rishabh","hardware","14/07/1999","7077553345","male","123456"));

employees.emplace_back(employee(6638,"sai382","sai","head","23/03/2001","9376373333","male","919332"));

employees.emplace_back(employee(1142,"kohli123","virat","software","08/10/1999","7379373373","male","156336"));

}

void viewEmployee(){
	int access_key;
	printf("ENTER THE ADMIN ACCESS KEY: ");
	cin>>access_key;

	if(access_key!=ADMIN_ACCESS_KEY){
	   printf("WRONG ACCESS KEY, ACCESS DENIED!!!\n");
	   system("pause");
	}
	else
	{
	   cout<<"\t ID	PSWD	NAME	DEPART	DOB	PHONE	GENDER	Area Pincode\n";

	   for(int i=0;i<employees.size();i++)
	   {
	      cout<<"\t"<<employees[i].e_id<<"  "<<employees[i].e_password<<"  "<<
	      employees[i].e_name<<"  "<<employees[i].e_department<<"  "<<
	      employees[i].e_dob<<"  "<<employees[i].e_phone<<"  "<<employees[i].gender<<"  "<<
	      employees[i].working_area<<" \n";
	   }

	   system("pause");
	}
}

void displayMenu(string str){
   system("cls");
   int choice;
   while(true) {
```

```
        system("cls");
        system("title VIT ELECTRICITY BOARD MANAGEMENT SYSTEM");
        upperPrint();

        cout<<"\t\tWelcome back, "<<str<<"\n\n";

        printf("1.Connection Maintenance\n2.Energy Audit\n3.Billing\n4.Logout\n");
        printf("YOUR CHOICE: ");
        cin>>choice;

        switch(choice){

            case 1:
                upperPrint();
                connectionMenu();
                break;

            case 2:
                upperPrint();
                auditMenu();
                break;

            case 3:
                upperPrint();
                billMenu();
                break;

            case 4:
                return;

            default:
                printf("INVALID INPUT\n");
        }

    }
}


int main(){
        int choice;
        int emp_id;
        string emp_password,emp_name,emp_dob,emp_phone,emp_dept,emp_gender;
        char comp_type[100000],comp_msg[100000];
        initialise_employees();


        while(true) {
        system("cls");
        system("title VIT ELECTRICITY BOARD MANAGEMENT SYSTEM");
        upperPrint();
        printf("1.Employee Login\n2.Add Employee\n3.View Employee\
```

```
            \n4.Complaint\n5.Pay Bills\n6.Exit\n\nCHOICE: ");

    cin>>choice;
            switch(choice) {
                    case 1:
          {
                            system("cls");
                            system("title EMPLOYEE LOGIN");
                            upperPrint();
                            printf("EMPLOYEE ID: ");
                            cin>>emp_id;
                            printf("EMPLOYEE PASSWORD: ");
                            cin>>emp_password;

                            if(login(emp_id,emp_password)){
                   string name;
                   for(auto it:employees)
                   {
                      if(it.e_id==emp_id && it.e_password==emp_password)
                      {
                         name=it.e_name;
                         break;
                      }
                   }
                                    displayMenu(name);
                            }
                            else{
                printf("LOGIN FAILED TRY AGAIN\n");
                system("pause");
                            }
                            break;
          }
                    case 2:
                       {
                            system("title EMPLOYEE REGISTRATION");

                            cout<<"EMPLOYEE ID: ";
                            cin>>emp_id;

                            printf("\nEMPLOYEE NAME: ");
                            cin>>emp_name;

                while(1){
                   printf("\nEMPLOYEE PASSWORD: ");
                   cin>>emp_password;
                   if(((emp_password.size()))>6)
                      break;
                   else
                      printf("enter more than 6 characters\n");
                }
                printf("\nEMPLOYEE DEPARTMENT: ");
```

```
    cin>>emp_dept;

   printf("\nEMPLOYEE DATE OF BIRTH: ");
    cin>>emp_dob;
    printf("\nEMPLOYEE PHONE NUMBER: ");
    cin>>emp_phone;
    cout<<"\nEMPLOYEE GENDER [ male / female ]: ";
    cin>>emp_gender;

    employee e;

    e.e_id=emp_id;
    e.e_password=emp_password;
    e.e_name=emp_name;
    e.e_department=emp_dept;
    e.e_dob=emp_dob;
    e.e_phone=emp_phone;
    e.gender=emp_gender;
    employee *obj;

    if(obj->newEmployee(e))
    {
       employees.emplace_back(e);
       cout<<"\nEmployee was added successfully !!";

    }

    else
    {
       cout<<"PLEASE TRY AGAIN,INVALID DETAILS\n";
    }
    system("pause");

    break;
            }
case 3:{
    system("title EMPLOYEE DETAILS");
    viewEmployee();
    break;
}
case 4:{
    system("cls");
    system("title COMPLAINTS");
    upperPrint();
    printf("\t\t\tCOMPLAINTS SECTION\n");

    printf("COMPLAINT TYPE (HOME OR REGION) :");
    cin>>comp_type;
    printf("COMPLAINT : ");
    cin>>comp_msg;
    complaints(comp_type,comp_msg);
```

```
            break;
        }
        case 5:{
            system("cls");
            system("title BILLS PAYMENT");
            upperPrint();

            printf("\t\t\tBILLS PAYMENT SECTION\n");
            billMenu();
            break;
        }
        case 6:
            exit(0);
        default:
            printf("Invalid input\n");
            system("pause");

    }
        }
        return 0;
}
```

2. audit.h

```
#include<iostream>
#include<vector>
#include<conio.h>
#include<stdlib.h>
#include<ctype.h>
#include<string.h>
using namespace std;
int ADMIN_ACCESS_KEY = 9498;

class audit
{
public:

    int aadhar_number;
    string pincode;
    int units;

    audit(int an, string pc, int u)
    {
        aadhar_number=an;
        pincode=pc;
        units=u;
    }

};
vector<audit> audits;
```

```
void initialise_audit()
{
   audits.emplace_back(audit(637828832,"126893",384));
   audits.emplace_back(audit(883363662,"678382",102));
   audits.emplace_back(audit(463727283,"478283",338));
   audits.emplace_back(audit(366637272,"123456",4882));
   audits.emplace_back(audit(155362737,"949472",17));
   audits.emplace_back(audit(466272992,"226889",399));
   audits.emplace_back(audit(900007362,"484939",200));
   audits.emplace_back(audit(227663737,"390224",204));
   audits.emplace_back(audit(837553637,"120345",502));

}

void netEnergyAudit(){
   system("title NET ENERGY COST");
   initialise_audit();
   int aadhar_number;
   char pincode[6];
   int units,sum=0;

   for(int i=0;i<audits.size();i++)
   {
      sum=sum+audits[i].units;
   }

   cout<<"Net Energy Consumption is "<<sum<<" Units\n";
}

bool compare(string a, string b)
{
   int n1=a.size();
   int n2=b.size();
   for(int i=0;i<min(n1,n2);i++)
   {
      if(a[i]!=b[i])
         return false;
   }
   return true;
}

void locationCost(char* pincode){
         system("title LOCATION BASED TRACKING");
   initialise_audit();
   int sum=0;

   for(int i=0;i<audits.size();i++)
   {
      if(compare(pincode,audits[i].pincode))
```

```
            sum+=audits[i].units;
        }
        printf("The Units in %s is %d\n",pincode,sum);

    }


    void auditMenu(){
        system("cls");
        char pincode[6];
        int access_key,choice,aadhar_no;
        while(1)
        {
            system("cls");
            system("title ENERGY AUDIT");
            upperPrint();
            printf("\t\t\tENERGY AUDIT\n");
            printf("1.Net Energy Audit\n2.Location based Energy Audit\n3.Back\n");
            printf("CHOICE: ");
            scanf("%d",&choice);

            char name[20],address[20],pincode[6],dob[10],phone[10];
            switch(choice){
                case 1:
                    printf("Enter the access key\n");
                    scanf("%d",&access_key);
                    if(access_key!=ADMIN_ACCESS_KEY){
                        printf("ACCESS DENIED!!\n");
                        break;
                    }
                    netEnergyAudit();
                    break;
                case 2:
                    printf("Enter the PIN code\n");
                    scanf("%s",pincode);
                    locationCost(pincode);
                    break;
                case 3:
                    return;
                default:
                    printf("INVALID INPUT");
            }
            system("pause");
        }
    }
```

3. bill.h

```
#include<iostream>
```

```
using namespace std;
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<ctype.h>
#include<string.h>
#include<time.h>

class bill
{
public:

   int serviceCharges;
   int baseCharges;
   int additionalCharges;
   int aadharNumber;

   bill(int sc, int bc, int ac, int an)
   {
      serviceCharges=sc;
      baseCharges=bc;
      additionalCharges=ac;
      aadharNumber=an;
   }

};

void otp(){
          srand(time(0));
          int otp_no = rand() * 100 + 1 , n;
          otp_no = otp_no % 999998 + 1;
          printf("OTP is sent to the mobile\n");
          FILE *ptr=fopen("otp.txt","w");
   if(ptr==NULL){
      return ;
   }
   else{
      fprintf(ptr,"%d",otp_no);
   }

   fclose(ptr);
   printf("ENTER THE OTP OR -1 to cancel payment: ");
   scanf("%d",&n);
   while(n!=otp_no){
          printf("INVALID OTP\n");
          printf("ENTER THE CORRECT OTP: ");
          scanf("%d",&n);

          if(n==-1)
          return;
          }
```

```
        printf("Your Payment was Successful... \n");
    remove("otp.txt");

}

void paymentGateway(){
        system("title PAYMENT GATEWAY");
        upperPrint();
        int type;
        char payment,debit[16],credit[16],upi[16],cvv[3];

        while(1){
      system("cls");
      upperPrint();
      printf("Choose payment method:\n1.Debit Card\n2.Credit Card\
          \n3.NETBANKING \n4.Offline\n5.Cancel Payment\nCHOICE: ");
      scanf("%d",&type);
      switch(type){
        case 1:
          printf("DEBIT CARD NUMBER (16 DIGITS) :");
          scanf("%s",debit);
          printf("CVV :");
          scanf("%s",cvv);
          if(strlen(cvv)!=3 || strlen(debit)!=16){
            printf("Invalid details\n");
            system("pause");
            break;
          }
          otp();
          return;
        case 2:
          printf("CREDIT CARD NUMBER (16 DIGITS) :");
          scanf("%s",credit);
          printf("CVV :");
          scanf("%s",cvv);
          if(strlen(cvv)!=3 || strlen(credit)!=16){
            printf("Invalid details\n");
            system("pause");
            break;
          }
          otp();
          return;
        case 3:
          char upii[10],cvvv[3];
          printf("NETBANKING REFERNCE NUMBER (10 DIGITS) :");
          scanf("%s",upii);
          printf("PIN (3 DIGITS):");

          scanf("%s",cvvv);
          if(strlen(cvvv)!=3 || strlen(upii)!=10){
            printf("Invalid details\n");
```

```
                system("pause");
                break;
            }
            otp();
            return;
        case 4:
            cout<<"You will have to reach out office for offline payment\n";
            break;
        case 5:
            return;
        default:
            printf("Invalid Input.");
            system("pause");
    }

        }
}

int metering(int aadharNumber){
        system("title METERING");
        int f_units;
        char f_aadhar_number[10],f_pin[6];


        for(int i=0;i<audits.size();i++)
   {
      //cout<<audits[i].aadhar_number<<"--";
      if(audits[i].aadhar_number==aadharNumber)
      {
         return audits[i].units;
                        system("pause");
      }

   }

   return 0;
}

void billing(int aadharNumber){
        system("title BILLING PROCESS");
        int unit=0;
        float cost=0;
        int f_units;
        string f_pin;

        for(int i=0;i<audits.size();i++)
   {

      if(audits[i].aadhar_number==aadharNumber)
      {
         unit=audits[i].units;
```

```
                        break;

        }

    }

    if(unit==0){
        printf("INVALID AADHAR CARD NUMBER!!\n");
        system("pause");
        return;
    }

        if (unit < 150)
            cost = (float)unit * 2.55;

        else if (unit < 250 )
            cost = (float)unit * 4.88;

        else
            cost = (float)unit * 5;

        printf("The cost is %.2f\n",cost);
        system("pause");
        printf("\nRedirecting to payment gateway...\n");
        system("pause");
        paymentGateway();
        system("pause");

}

void billMenu()
{
        int choice,units;
        int aadhar_number;
        initialise_audit();
        while(1)
        {
            system("cls");
            system("title BILLING");
            upperPrint();

            printf("\t\tBILL MAINTENANCE\n");
            printf("1.Metering\n2.Billing\n3.Back\n");
            printf("Enter Choice : ");
            scanf("%d",&choice);
            switch(choice)
            {

                case 1:
                    printf("Enter your Aadhar Card Number : ");
                    cin>>aadhar_number;
```

```
                                units=metering(aadhar_number);
                                if(units==0)
                                {
                                        printf("INVALID AADHAR CARD NUMBER\n");
                                        system("pause");
                                }
                                else
                                {
                                        printf("The Aadhar Card Number %d used %d
units\n",aadhar_number,units);
                                        system("pause");
                                }
                                break;
                        case 2:
                                printf("Enter Aadhar Card Number : ");
                                cin>>aadhar_number;
                                billing(aadhar_number);
                                return;
                        case 3:
                                return;
                        default:
                                printf("INVALID INPUT\n");
                                system("pause");
                }

        }
}
```

4. connection.h

```
#include<iostream>
#include<stdio.h>
#include<stdlib.h>
#include<vector>
#include<string.h>
#include<conio.h>
#include<ctype.h>
#include"customer.h"
using namespace std;

char stars()
{
   for(int i=0;i<80;i++)
      cout<<'*';

}

void box(int x, int y)
{
        for(int i=0;i<y;i++)
```

```
        cout<<(char)x;
}

void initialise_customer(vector<customer> &customers)
{

customers.emplace_back(customer(1373838337,"Varun","haryana",135001,"08/10/1998","9376373373"))
;

customers.emplace_back(customer(6373236737,"jayesh","jharkhand",535041,"28/01/1999","9884736663
"));

}

void upperPrint()
{
   cout<<endl<<endl<<endl;
    box(178,20);
    cout<<"  W E L C O M E   T O   V I T - E B M S ";
    box(178,20);
    cout<<endl<<endl;
    cout<<stars()<<endl<<endl;

}


int getcid(){
        int c_id;
        FILE *temp1 = fopen("cus.txt","r");
        fscanf(temp1,"%d",&c_id);
        fclose(temp1);
        printf("CUSTOMER ID IS %d\n",c_id);
        return c_id;
}

void inccid(int value){
        FILE *temp2 = fopen("cus.txt","w");
        fprintf(temp2,"%d",value);
        fclose(temp2);
}



int addConnection(char *name,char *address,char *pincode,char *dob,char *phone){
        int aadhar_number = getcid();
   FILE *ptr=fopen("connection.txt","a");
   if(ptr==NULL)
   {
      fclose(ptr);
      return 0;
   }
```

```
    else
    {
        aadhar_number++;
        inccid(aadhar_number);
      fprintf(ptr,"%d %s %s %s %s %s\n",aadhar_number,name,dob,phone,address,pincode);
    }
    fclose(ptr);
    return 1;

}

int disconnection(int number,char *reason){
    int aadhar_num;
        char name[20],address[30],pincode[20],dob[10],phone[10];

    FILE *ptr=fopen("connection.txt","r");
        FILE *temp=fopen("temp.txt","w");
    if(ptr==NULL){
      return 0;
    }
    else
    {

      printf("The connection of %d will be disconnected shortly\nReason of disconnection :
%s\n",aadhar_num,reason);

                fclose(ptr);
      fclose(temp);
      if(remove("connection.txt"))
        printf("deleted\n");
      rename("temp.txt","connection.txt");
                system("pause");
                return 1;
    }

    return 0;
}

int displayCustomer(int number, vector<customer> &customers){
        int aadhar_num,flag=-1;
        char name[20],address[30],pincode[20],dob[10],phone[10];
    FILE *ptr=fopen("connection.txt","r");

    if(ptr==NULL){
      return 0;
    }
    else{

        for(int i=0;i<customers.size();i++)
        {
          if(customers[i].adharNumber==number)
```

```
            {
               flag=1;
               cout<<customers[i].adharNumber<<"  "<<
                     customers[i].name<<"  "<<
                     customers[i].address<<"  "<<
                     customers[i].pincode<<"  "<<
                     customers[i].dateofbirth<<"  "<<
                     customers[i].phone<<endl;
                     break;
               }
            }

      }
      fclose(ptr);
      if(flag==-1)
         printf("Invalid Aadhar number try again\n");
      system("pause");
      return 1;
}


void connectionMenu(){
      system("cls");
      upperPrint();
      vector<customer> customers;
      initialise_customer(customers);
      int choice,aadhar_no;
      char reason[100];
      while(1){
         system("cls");
         upperPrint();
         system("title VIT CONNECTION MAINTENANCE");

         printf("1.New Connection\n2.Disconnection\n3.Customer Details\n4.Go Back\n");
         printf("CHOICE: ");
         scanf("%d",&choice);
         char name[20],address[20],pincode[6],dob[10],phone[10];
         switch(choice){
            case 1:
               system("title CONNECTION REGISTRATION");
               printf("CUSTOMER NAME: ");
               scanf("%s",name);
               printf("CUSTOMER ADDRESS: ");
               scanf("%s",address);
               printf("CUSTOMER PINCODE: ");
               scanf("%s",pincode);
               printf("CUSTOMER DATE OF BIRTH: ");

              scanf("%s",dob);
               printf("CUSTOMER PHONE NUMBER: ");
               scanf("%s",phone);
```

```
            if(addConnection(name,address,pincode,dob,phone)){
               printf("CONNECTION REGISTERED\n");
               system("pause");
            }
            else{
               printf("PLEASE TRY AGAIN\n");
            }

            break;
         case 2:
            system("title DISCONNECTION");
            printf("AADHAR NUMBER: ");
            scanf("%d",&aadhar_no);
            printf("REASON: ");
            scanf(" %[^\n]",reason);
            if(!disconnection(aadhar_no,reason)){
               printf("FILE NOT FOUND\n");
            }
            break;
         case 3:
            system("title CONNECTION QUERY");
            printf("CUSTOMER AADHAR NUMBER: ");
            scanf("%d",&aadhar_no);
            if(!displayCustomer(aadhar_no,customers)){
               printf("INVALID CUSTOMER NUMBER\n");
            }
            break;
         case 4:
            return;
         default:
            printf("INVALID INPUT\n");
      }
   }
   system("pause");
}
```

5. employee.h

```
#include<iostream>
#include<vector>
#include<cstdio>
#include<cstdlib>
#include<cstring>
#include<ctype.h>



class employee
{
public:
```

```
int e_id;
string e_password;
string e_name;
string e_department;
string e_dob;
string e_phone;
string gender;
string working_area;

employee()
{
}

employee(int eid, string epass, string en, string ed, string edob, string eph,string g,string wa)
{
   e_id=eid;
   e_password=epass;
   e_name=en;
   e_department=ed;
   e_dob=edob;
   e_phone=eph;
   gender=g;
   working_area=wa;
}

bool newEmployee(employee e)
{
   int id=e.e_id;
   string password=e.e_password;
   string name=e.e_name;
   string department=e.e_department;
   string dob=e.e_dob;
   string phone=e.e_phone;
   string gen=e.gender;
   string wA=e.working_area;

   if(dob.size()!=10 || dob[2]!='/' || dob[5]!='/'){
      printf("DD/MM/YYYY(format is invalid)\n");
      system("pause");
      return false;
   }

   if(phone.size()!=10){
         printf("phone number should be 10 digits\n");

               system("pause");
         return false;
   }

   int i;
```

```
        i=0;
        while(i<phone.size()){
           if(phone[i]<'0' || phone[i]>'9'){
              printf("phone number should contain only numbers\n");
              system("pause");
              return false;
           }
           i++;
        }

        i=0;
        while(i<name.size()){
           if(!isalpha(name[i])){
              printf("name should contain only alphabets\n");
              system("pause");
              return false;
           }
           i++;
        }


        i=0;
        while(i<department.size()){
           if(!isalpha(department[i])){
              printf("department should contain only alphabets\n");
              system("pause");
              return 0;
           }
           i++;
        }

        i=0;
        while(i<wA.size()){
           if(!isdigit(wA[i])){
              printf("Pincode should contain only digits\n");
              system("pause");
              return 0;
           }
           i++;
        }

        return true;

    }

};
```

6. complaint.h

```
#include<iostream>
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<ctype.h>
#include<string.h>

using namespace std;

class complaint
{
   string type;
   string message;
   int aadhar_number;
   string pincode;

   complaint(string ty, string msg, int an,string pn)
   {
      type=ty;
      message=msg;
      aadhar_number=an;
      pincode=pn;
   }


};

void report(char* comp_type,char* comp_msg,int locality){
         printf("The employee has been informed about the issue.\n\
      Problem will be rectified shortly.\n");
         system("pause");
}

void alertfunction(char* comp_type,char* comp_msg,char* locality){
         int aadhar_num,sno=1;
         char name[20],address[30],pincode[20],dob[10],phone[10];

         FILE *customer =fopen("connection.txt","r");


         printf("MESSAGE: %s\n",comp_msg);
         printf("Sending messages to the required members via SMS: \n");

      while(fscanf(customer,"%d %s %s %s %s
%s",&aadhar_num,name,dob,phone,address,pincode)!=EOF)
      {
```

```
if(strcmp(pincode,locality)){

                printf("%d.%s - %s\n",sno,name,phone);
                sno++;
                   }
                 }

                  printf("MESSAGE SEND SUCCESSFULLY, our employees will reach out to you
shortly\n");

      system("pause");
      fclose(customer);

}

void complaints(char* comp_type,char* comp_msg)
{

         if(comp_type=="HOME" || comp_type=="home")
   {
                int aadhar_number;
                printf("Enter Aadhar Card Number\n");
                cin>>aadhar_number;

                FILE *complaint=fopen("complaint.txt","a+");
                int aadhar_num,flag=-1;

                char name[20],address[30],pincode[20],dob[10],phone[10];
            FILE *ptr=fopen("connection.txt","r");
            if(ptr==NULL){
               return ;
            }
            else{
               while(fscanf(ptr,"%d %s %s %s %s
%s",&aadhar_num,name,dob,phone,address,pincode)!=EOF){
            if(aadhar_number==aadhar_num){
               flag=1;
            }
               }
            }
            fclose(ptr);

         if(flag==-1){
         printf("Invalid aadhar number try again\n");
         system("pause");
         return;
                  }
                fprintf(complaint,"\n%d %s %s",aadhar_number,comp_type,comp_msg);
                report(comp_type,comp_msg,aadhar_number);

                fclose(complaint);
```

```
        }

        else if(comp_type,"REGION" || comp_type=="region"){
                char pincode[6];
                printf("Enter PINCODE\n");
                scanf("%s",pincode);

                FILE *alert=fopen("alert.txt","a");
                fprintf(alert,"\n%s %s %s",pincode,comp_type,comp_msg);
                alertfunction(comp_type,comp_msg,pincode);

                fclose(alert);
        }
        else{
                printf("Invalid Input\n");
        }
}
```

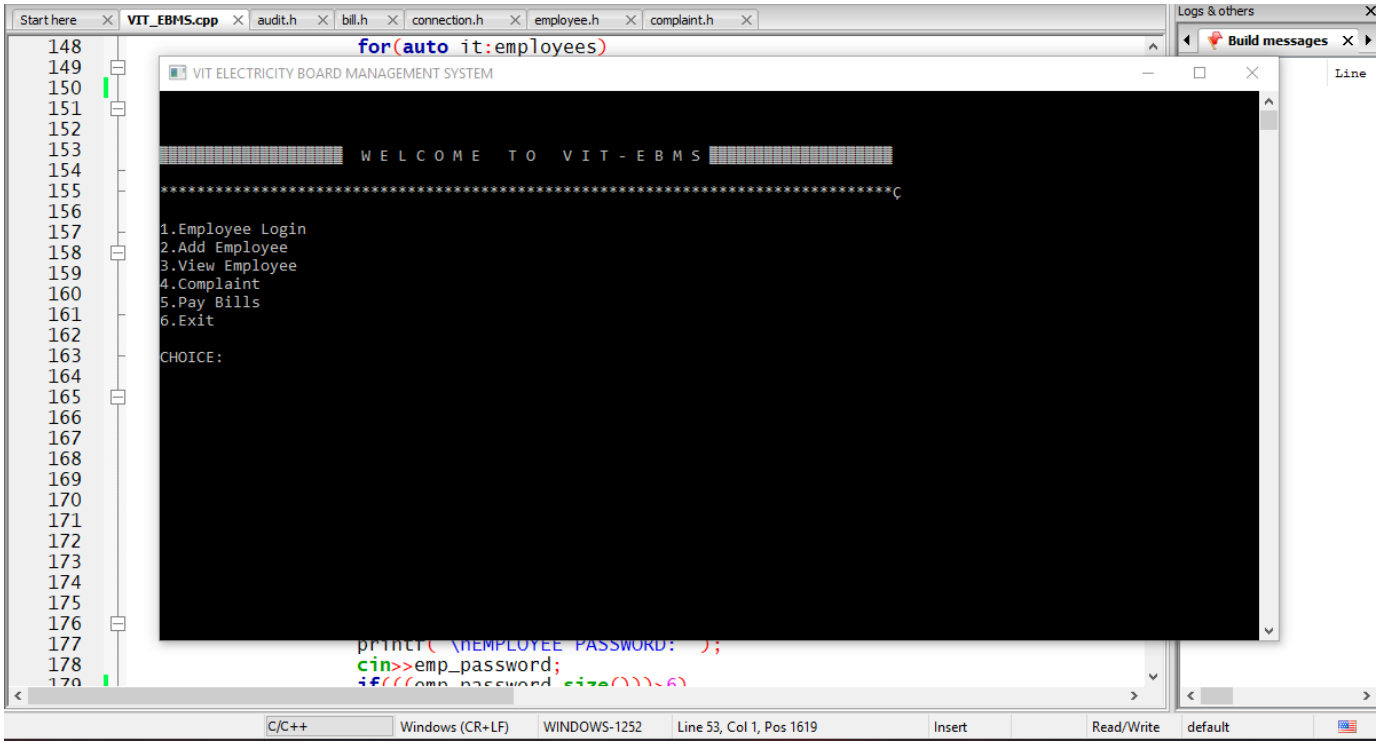
7. customer.h

```
#include<iostream>
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<ctype.h>
#include<string.h>
#include<iostream>
using namespace std;

class customer
{
public:

   int adharNumber;
   string name;
   string address;
   int pincode;
   string dateofbirth;
   string phone;

   customer(int an, string n, string add, int pin,string dob, string ph)
   {
      adharNumber=an;
      name=n;
      address=add;
      pincode=pin;
      dateofbirth=dob;

phone=ph;
   }
```

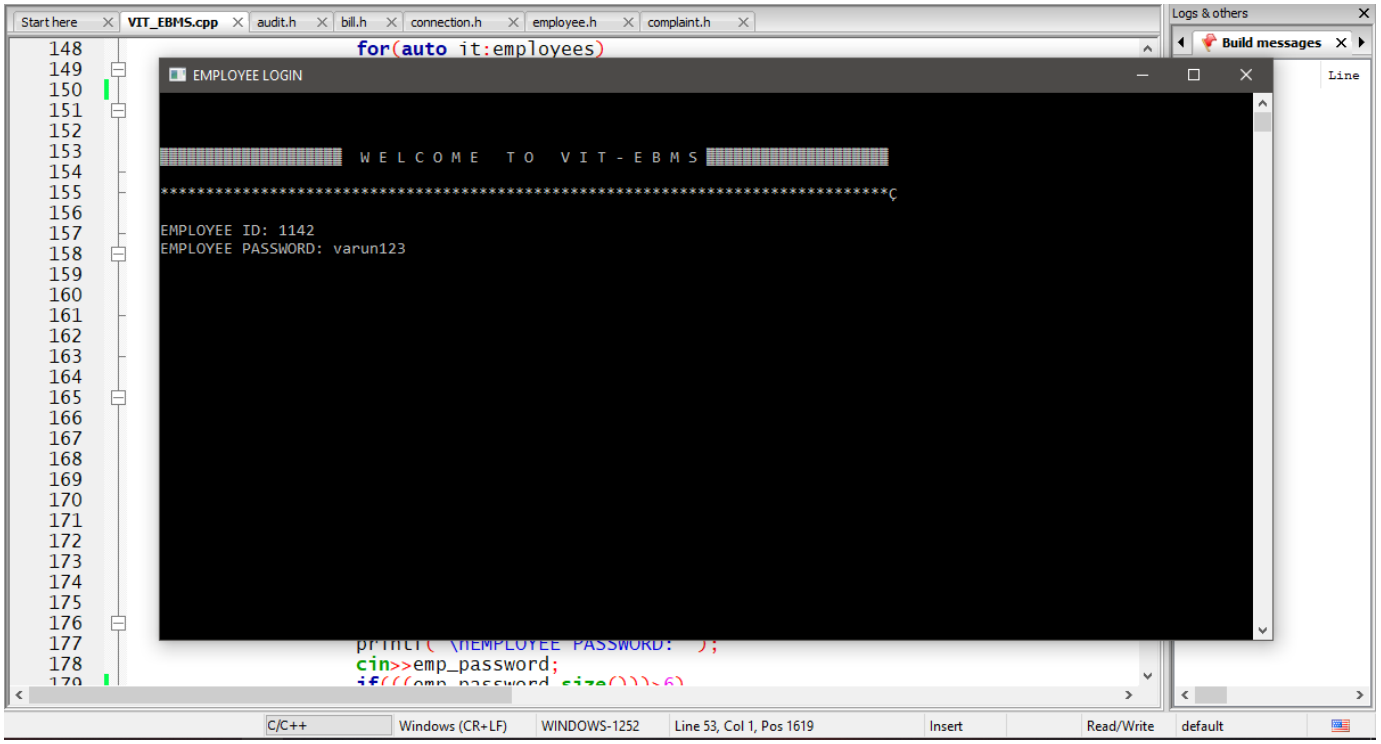
```
bool newCustomer(customer c)
   {
      int an=c.adharNumber;
      string name=c.name;
      string address=c.address;
      string dob=c.dateofbirth;
      int pin=c.pincode;
      string phone=c.phone;

      if(dob.size()!=10 || dob[2]!='/' || dob[5]!='/'){
         printf("DD/MM/YYYY(format is invalid)\n");
         system("pause");
         return false;
      }

      if(phone.size()!=10){
            printf("phone number should be 10 digits\n");
            system("pause");
            return false;
      }

      int i;
      i=0;
      while(i<phone.size()){
         if(phone[i]<'0' || phone[i]>'9'){
            printf("phone number should contain only numbers\n");
            system("pause");
            return false;
         }
         i++;
      }

      i=0;
      while(i<name.size()){
         if(!isalpha(name[i])){
            printf("name should contain only alphabets\n");
            system("pause");
            return false;
         }
         i++;
      }

      return true;

   }

};
```

RESULTS –

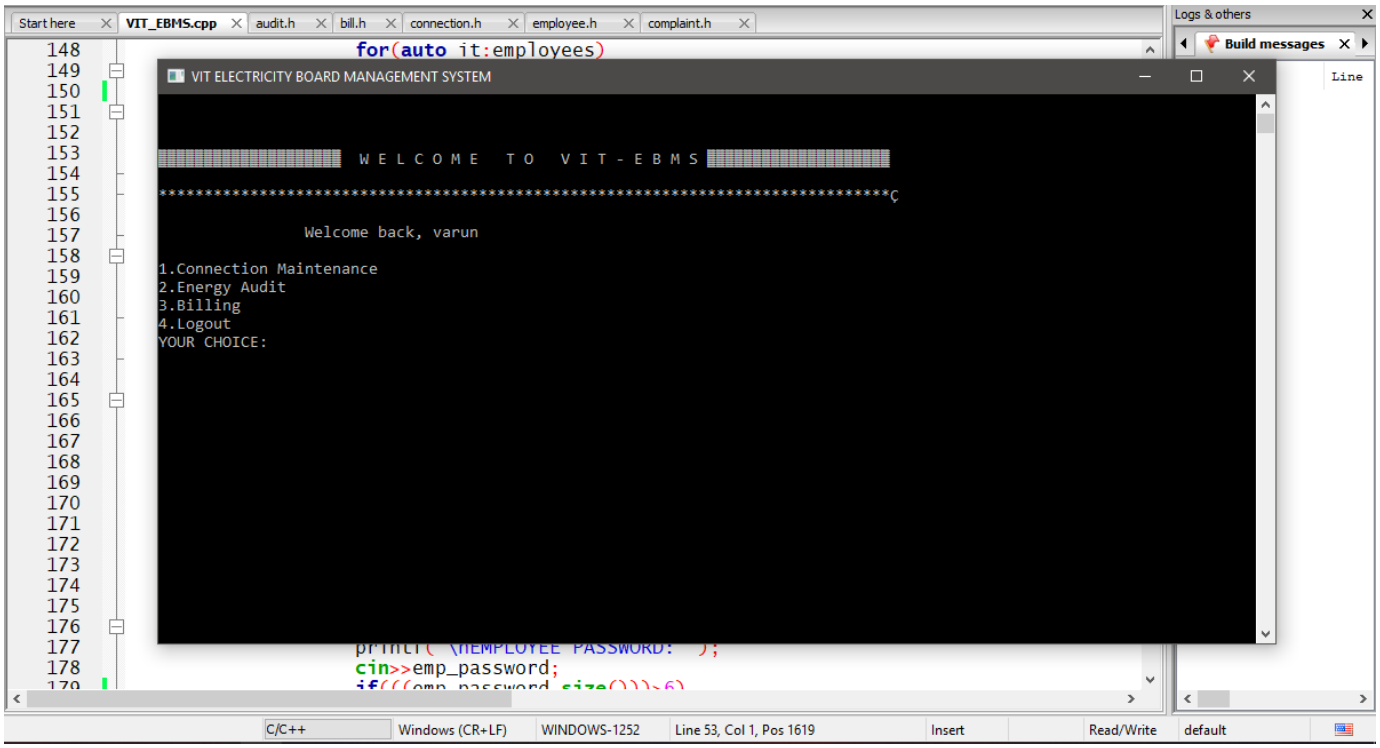
Main Menu



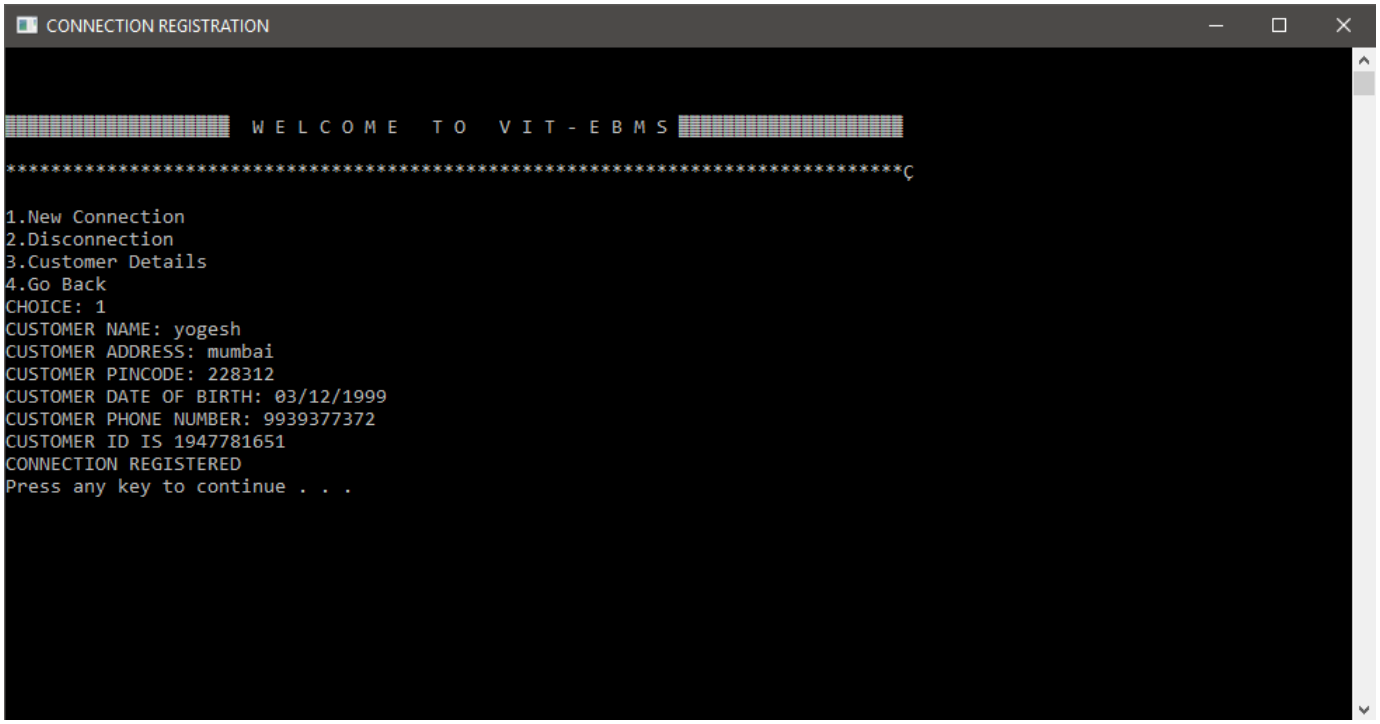
Employee Login



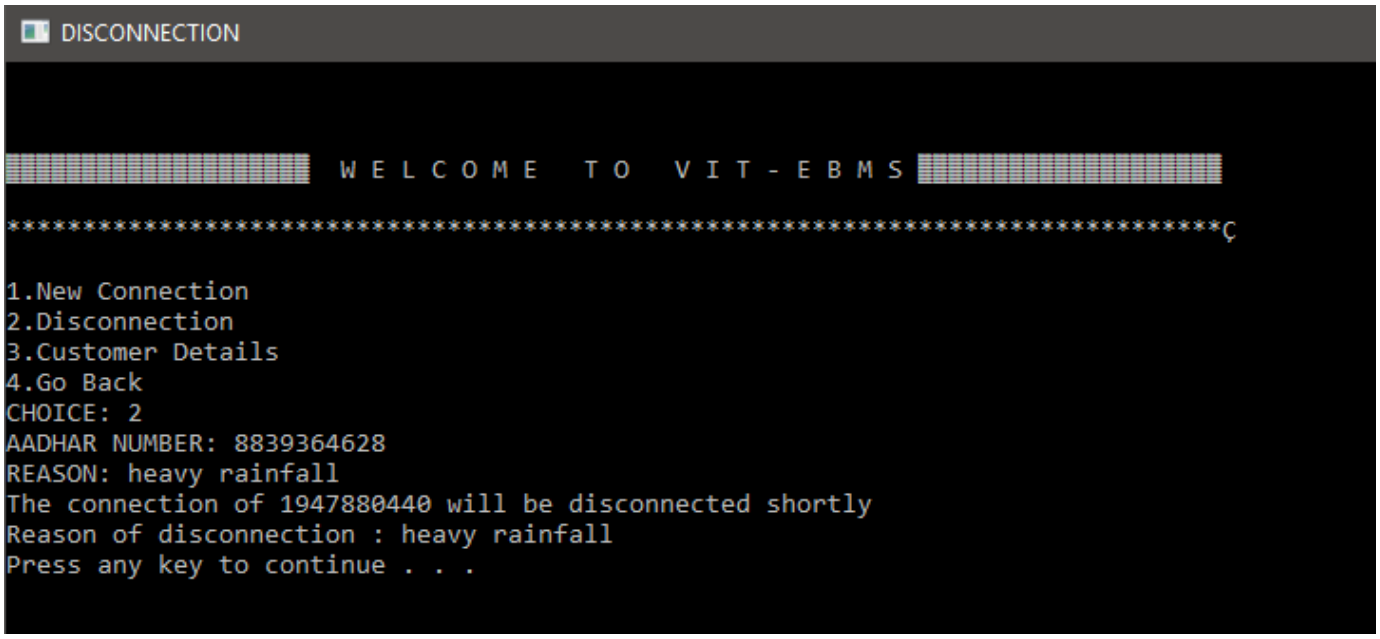
Employee Dashboard -

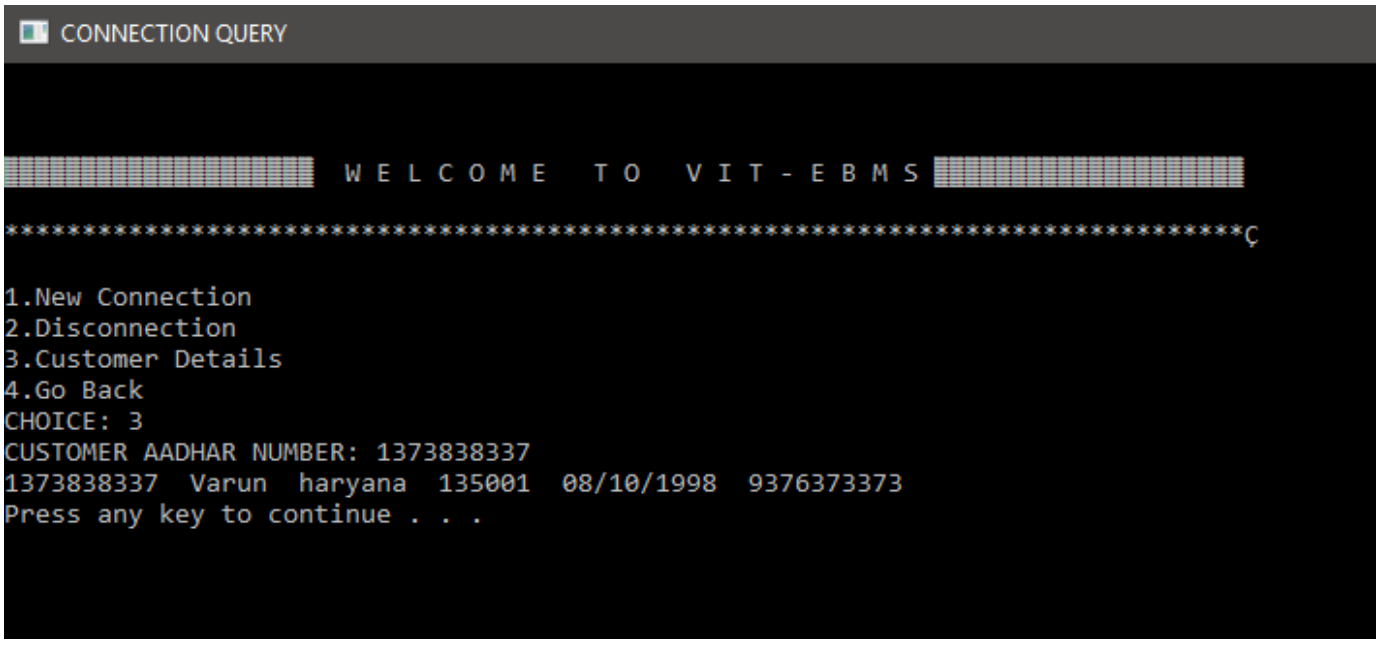


Employee Connection Maintenance –

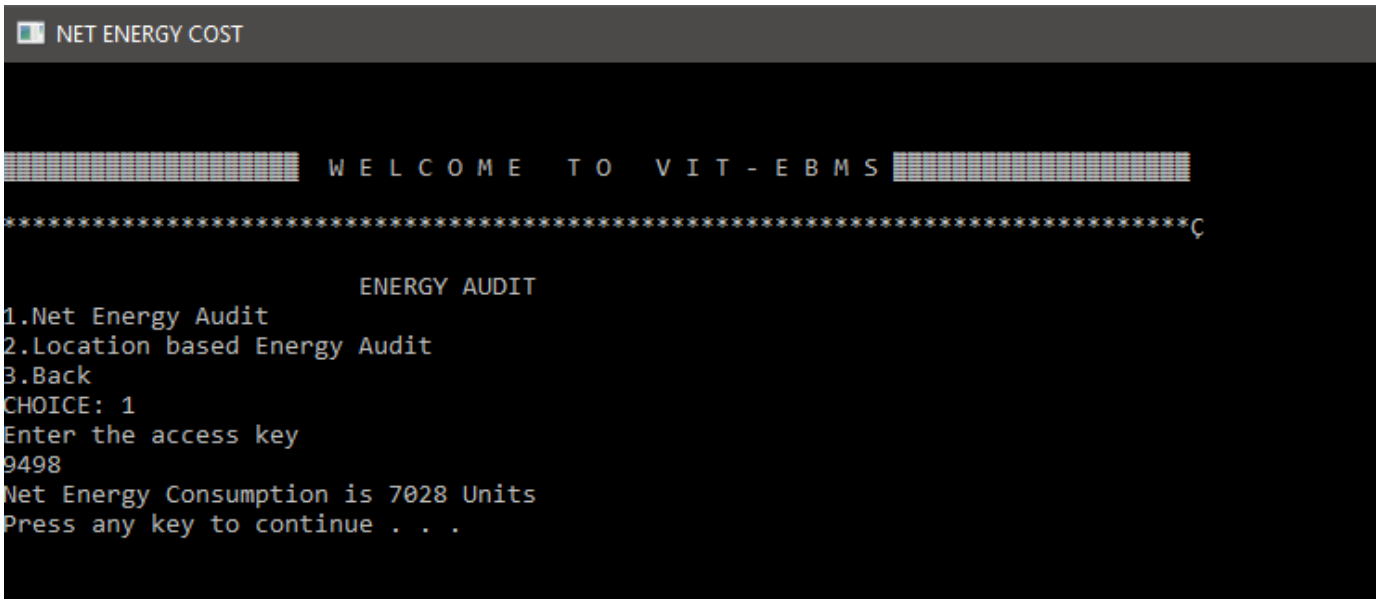


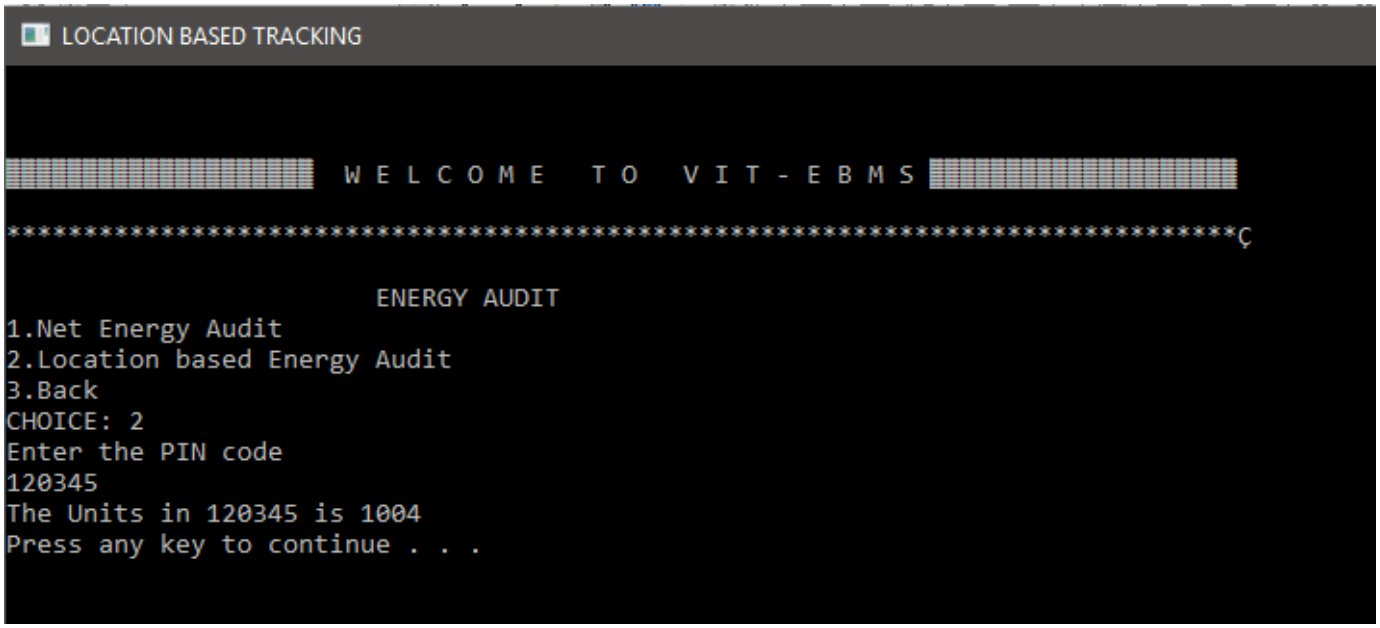
Employee Query Handling -



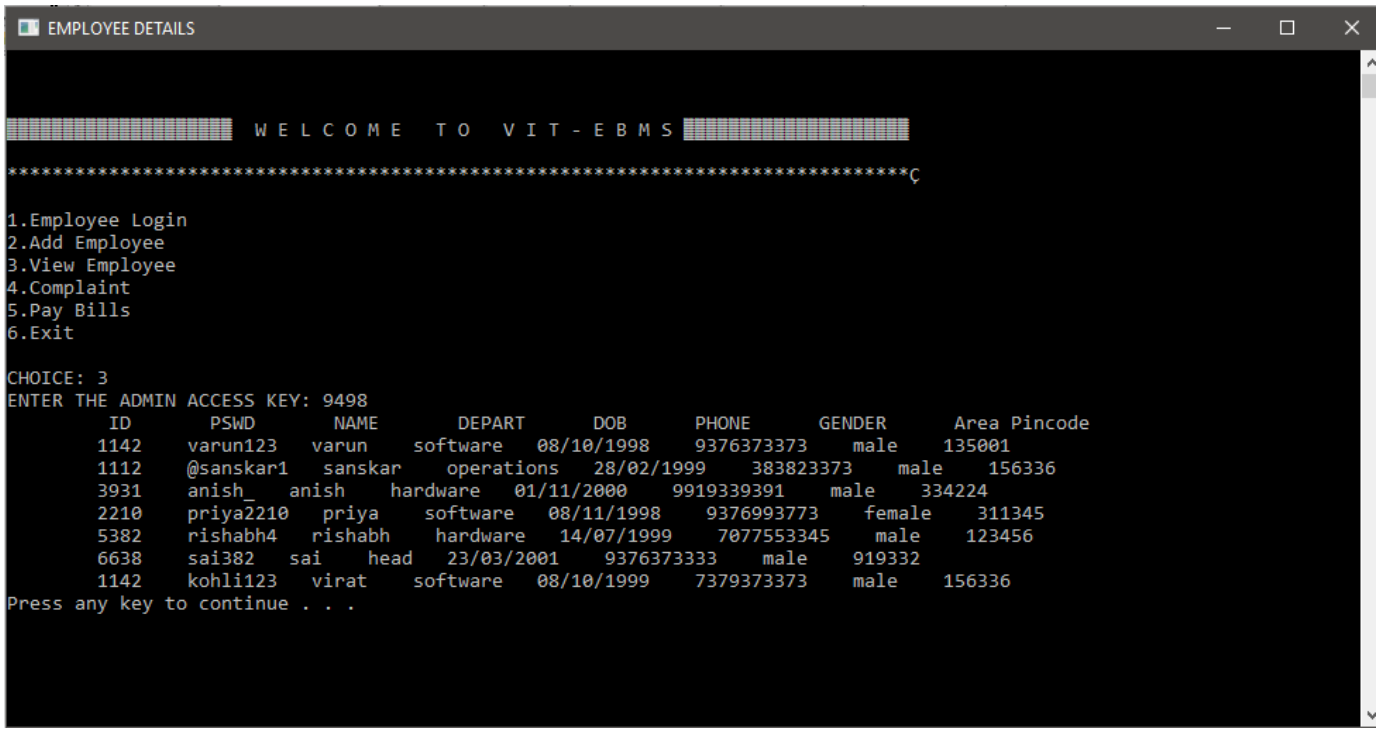


Energy Audit – viewing energy consumptions (total and for one specific user).

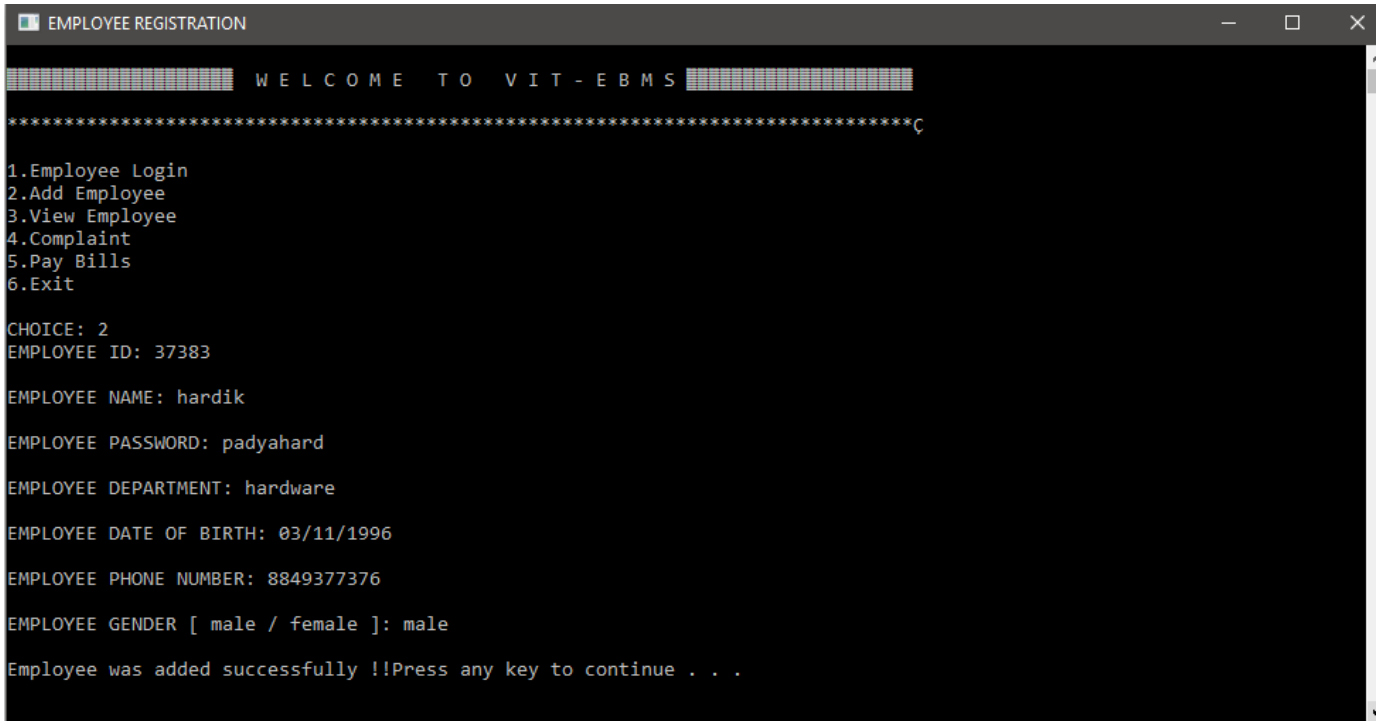




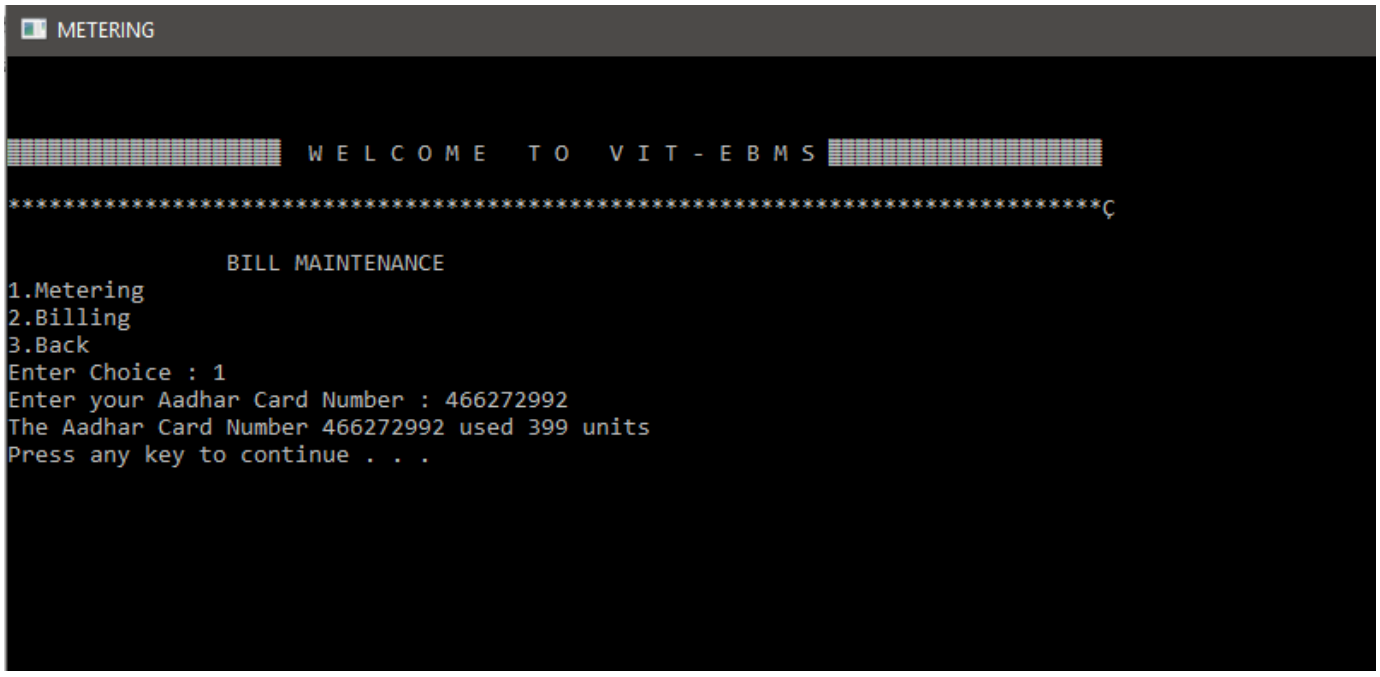
Viewing all employees -



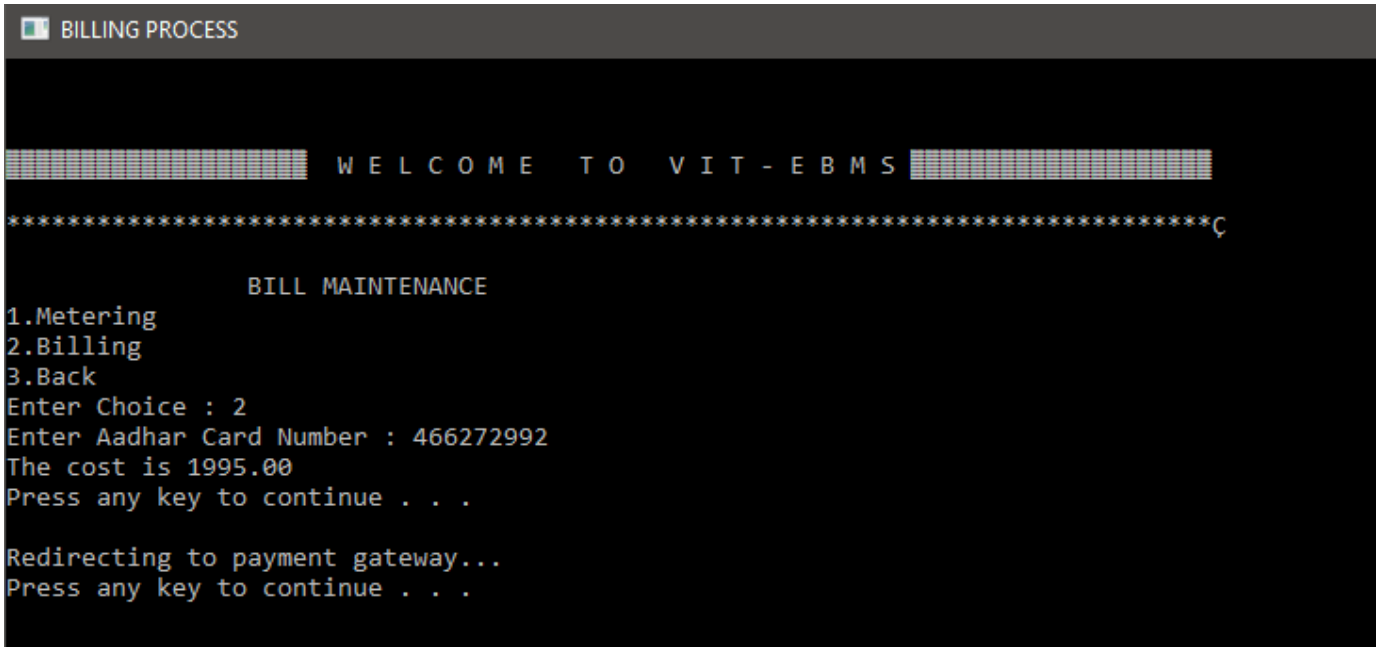
Adding New Employee -



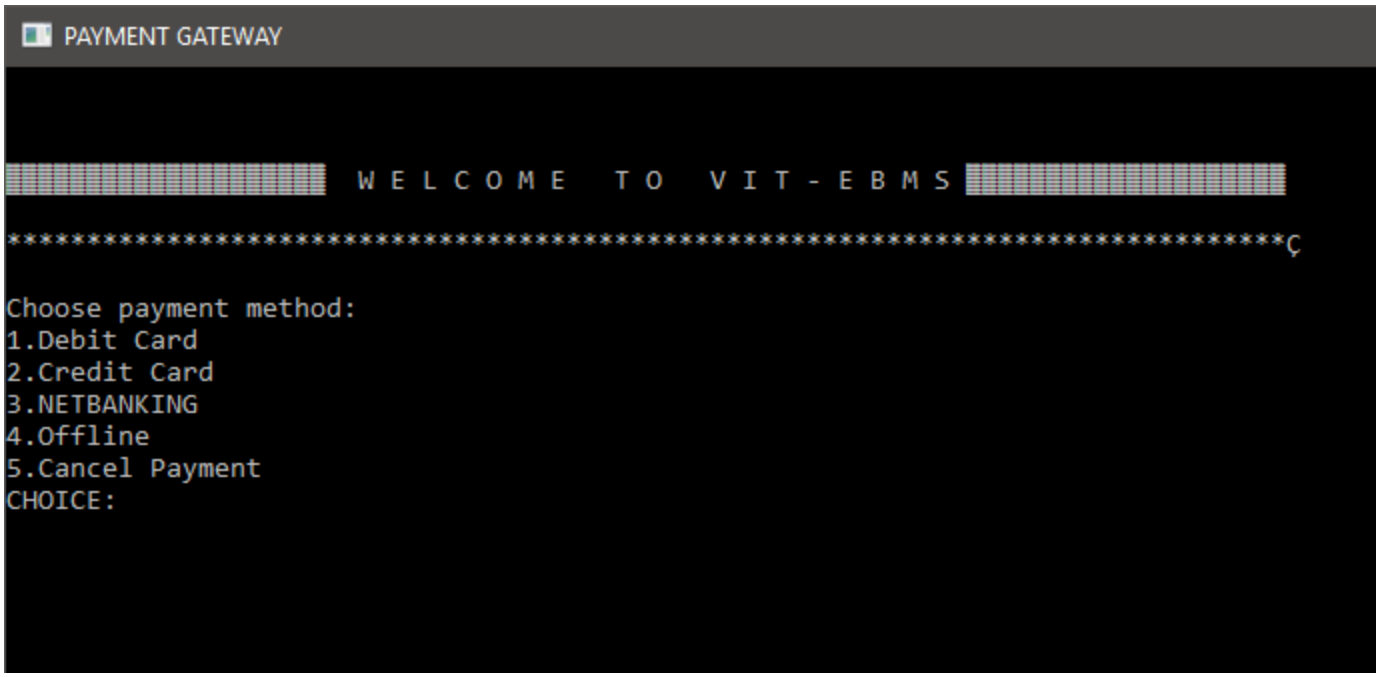
Viewing pending bills –

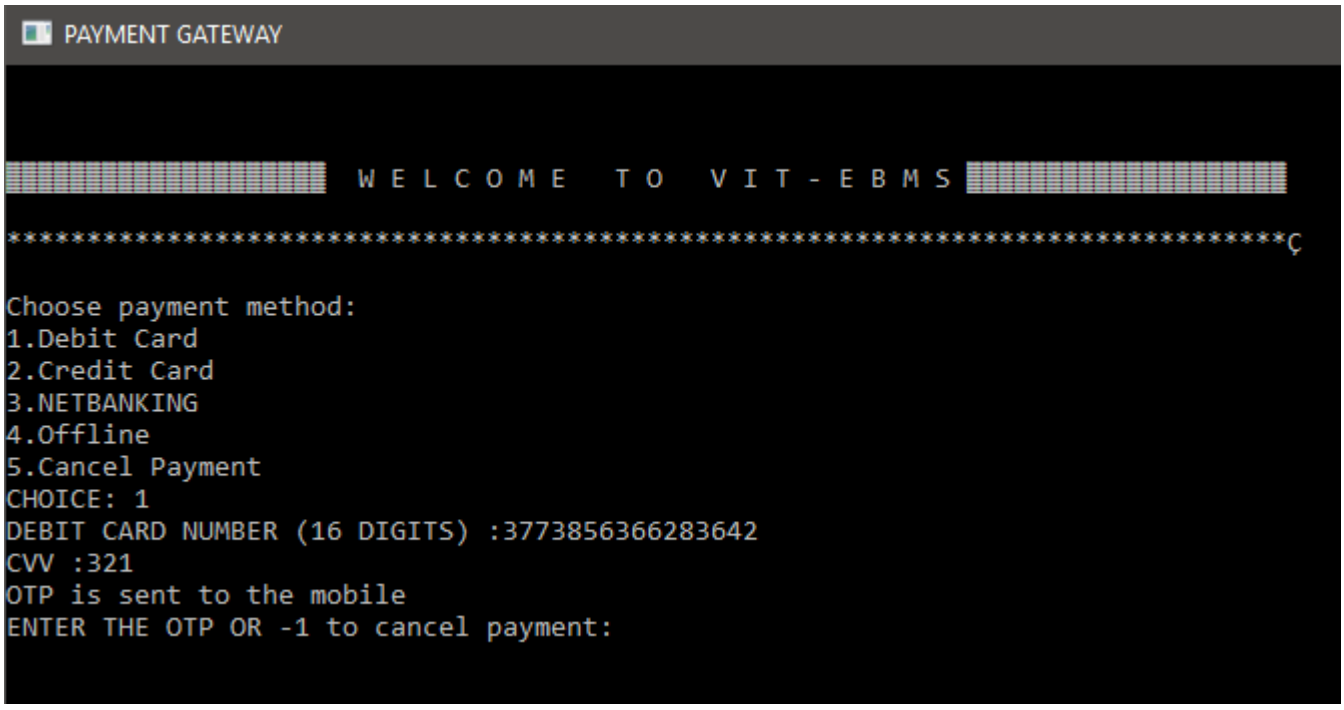


Billing –

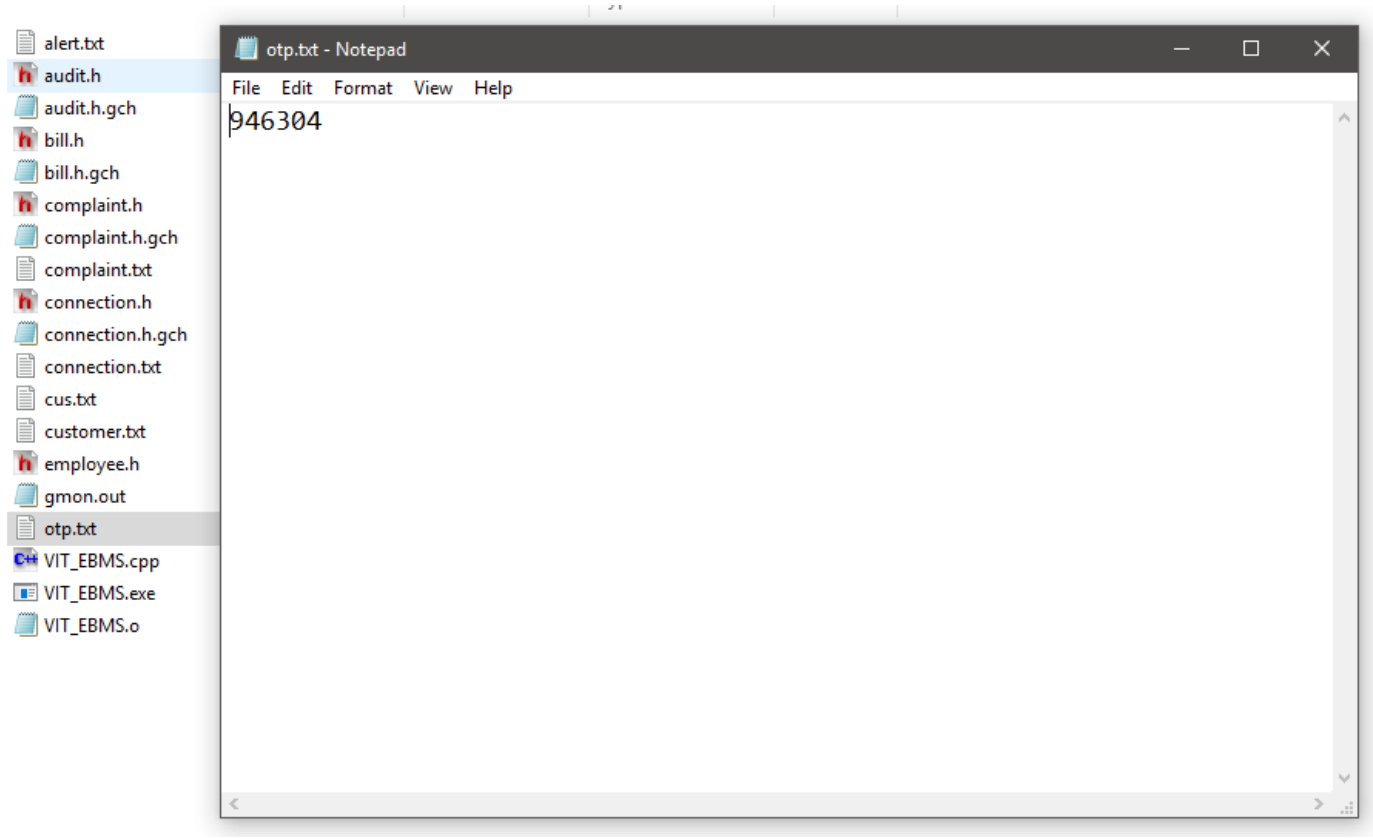


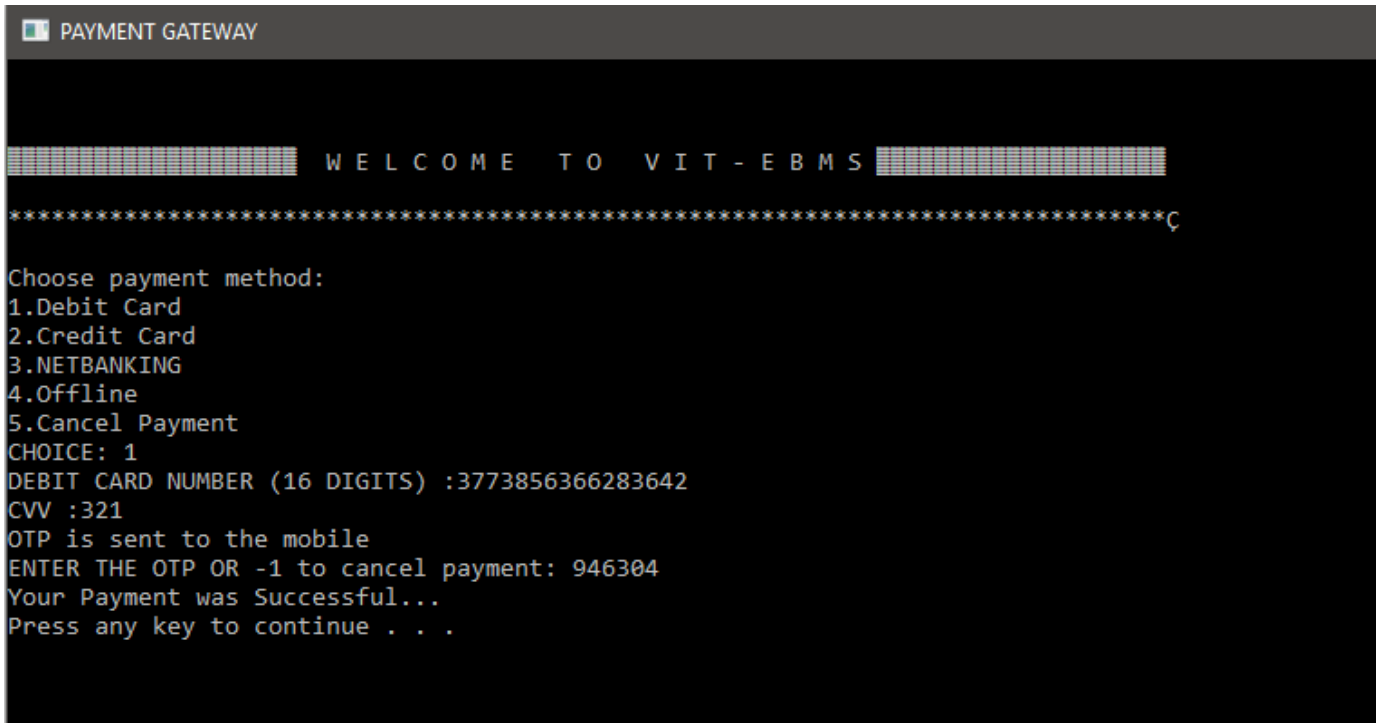
Bill Payment -





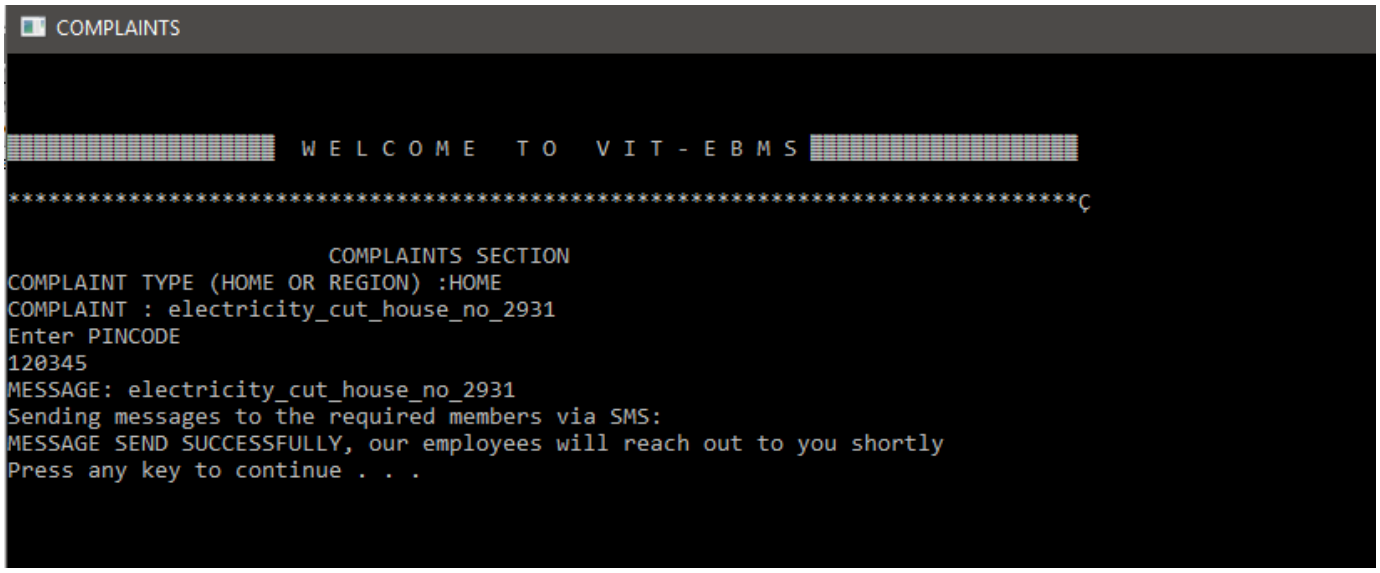
OTP generation-



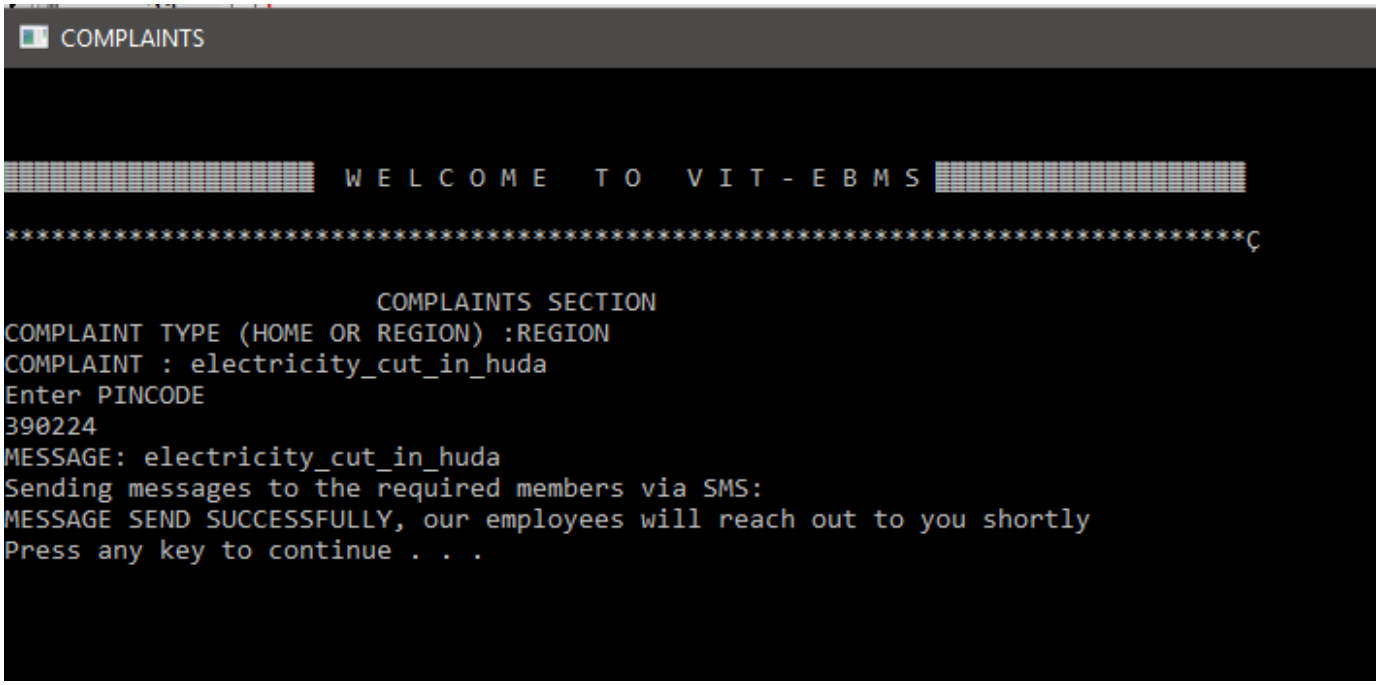


Complaints –

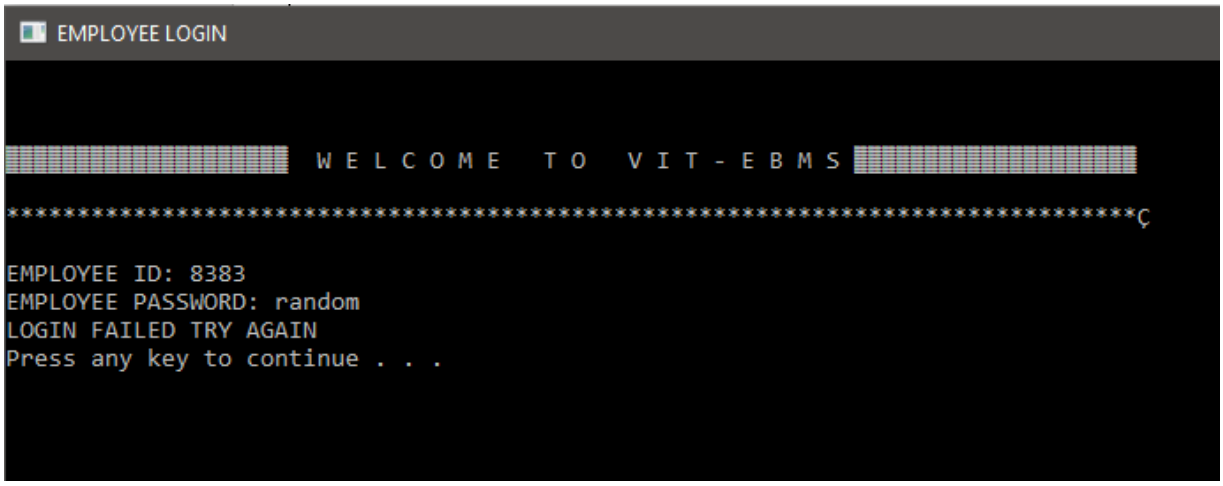
Home Complaints -

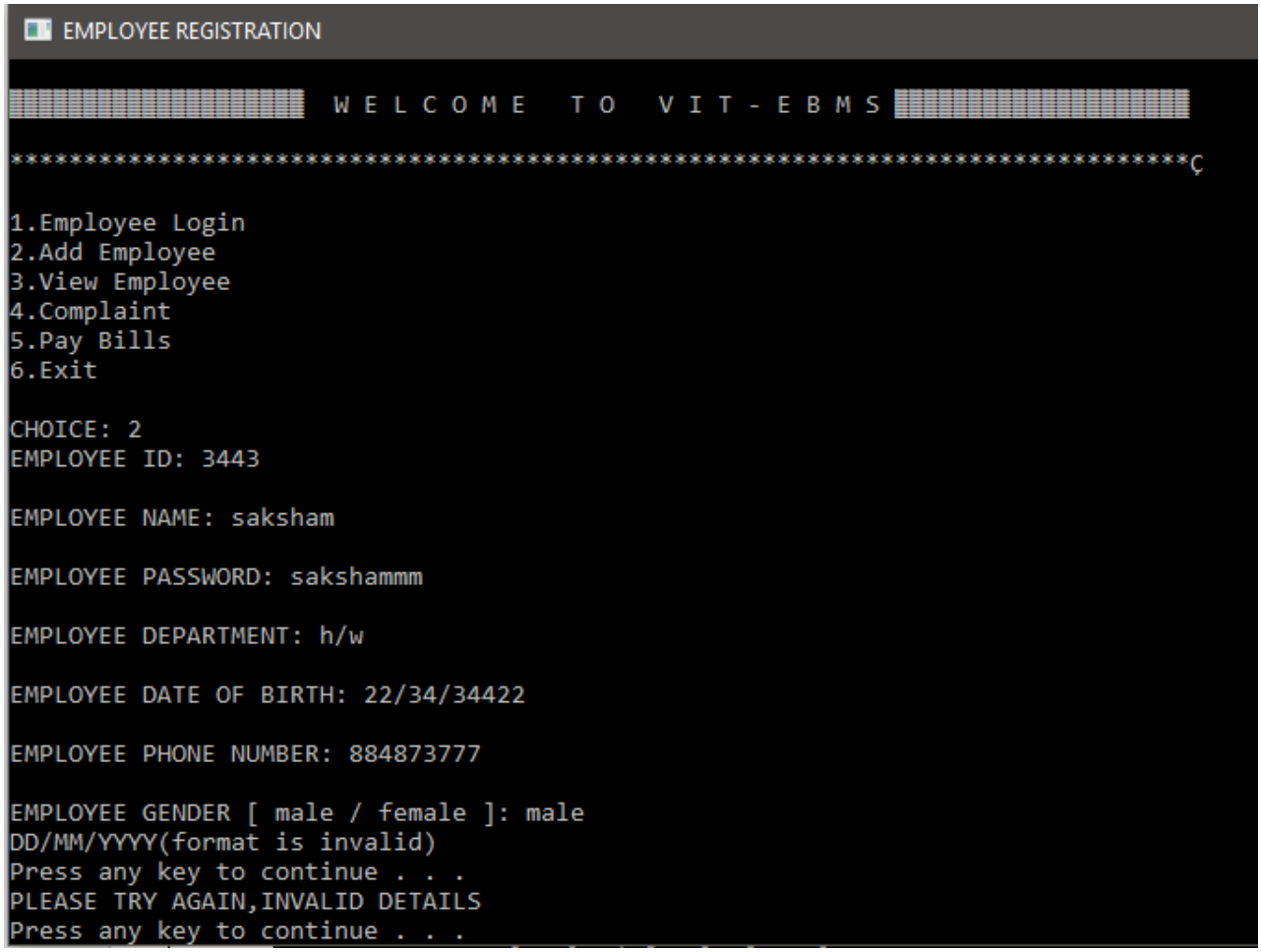


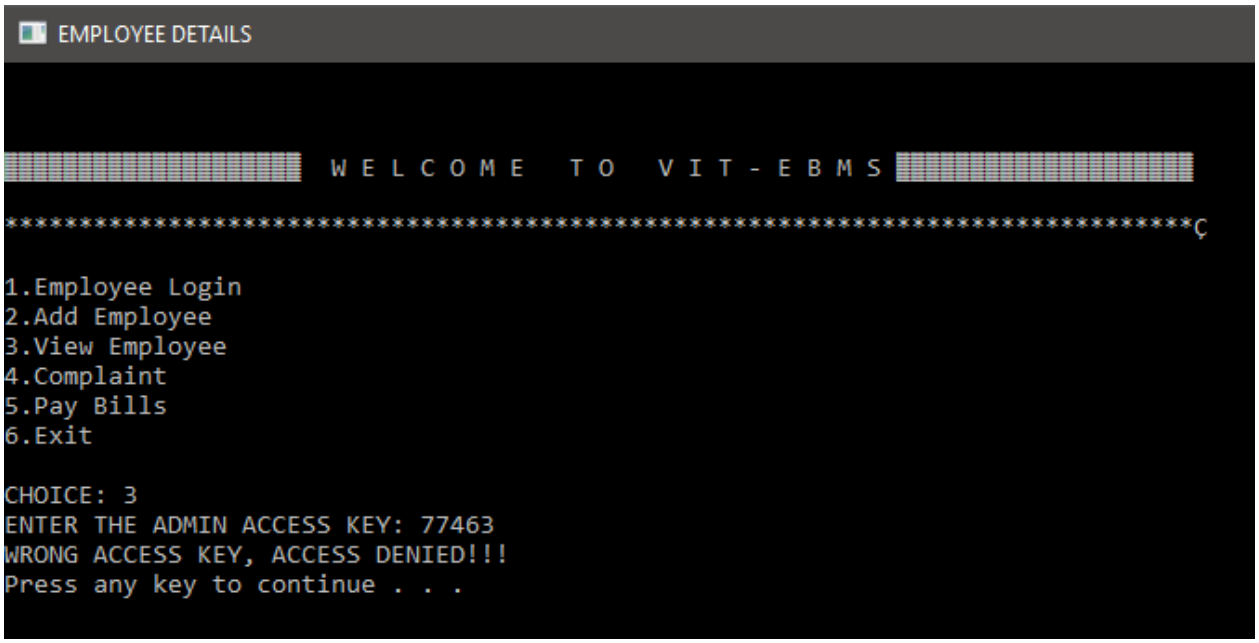
Regional Complaint -

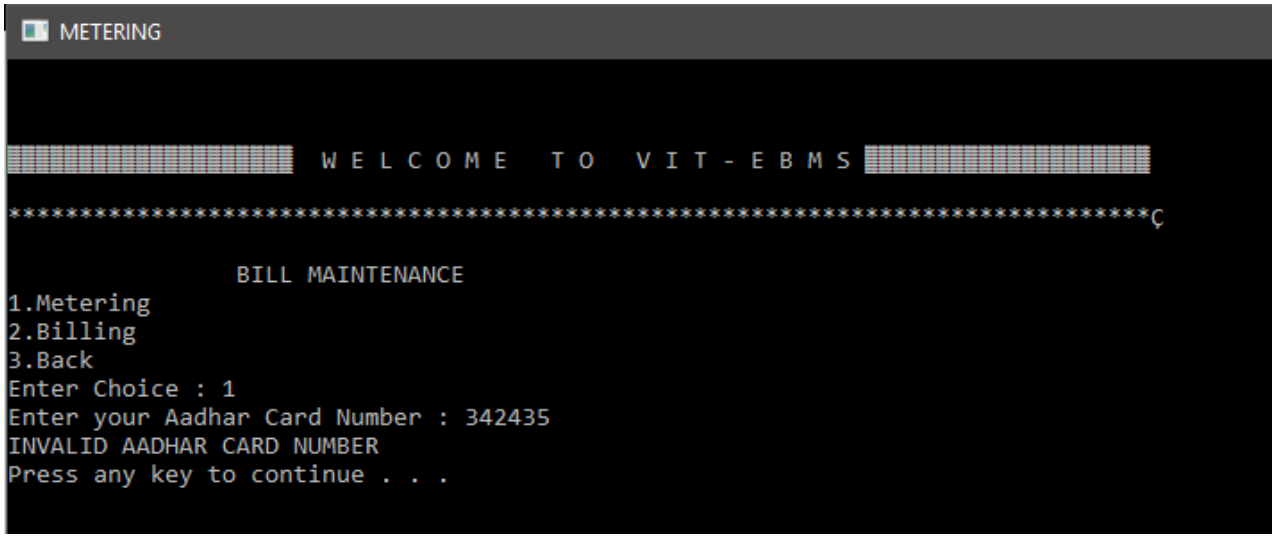


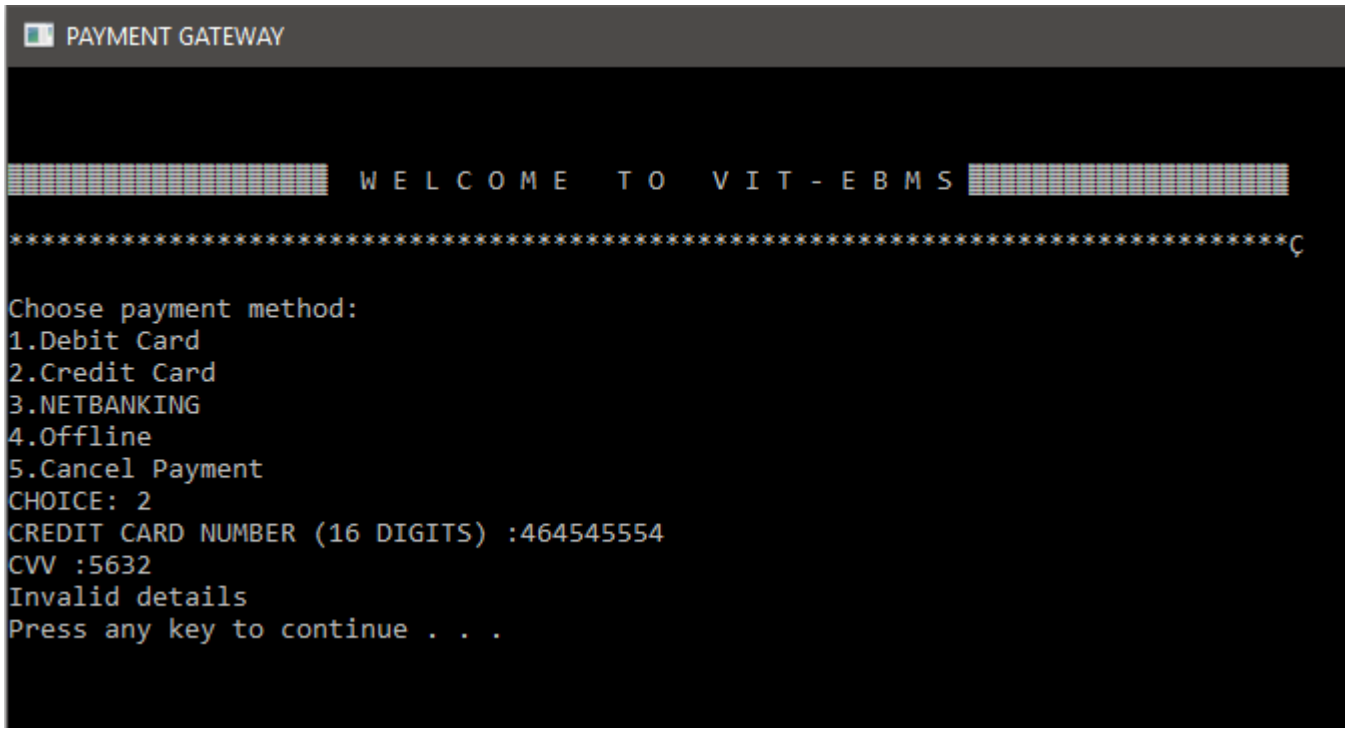
System handling all kinds of errors –






EMPLOYEE DETAILS
W E L C O M E T O V I T - E B M S
1.Employee Login
2.Add Employee
3.View Employee
4.Complaint
5.Pay Bills
6.Exit
CHOICE: 3
ENTER THE ADMIN ACCESS KEY: 77463
WRONG ACCESS KEY, ACCESS DENIED!!!
Press any key to continue . . .


METERING
W E L C O M E T O V I T - E B M S
BILL MAINTENANCE
1.Metering
2.Billing
3.Back
Enter Choice : 1
Enter your Aadhar Card Number : 342435
INVALID AADHAR CARD NUMBER
Press any key to continue . . .



CONCLUSION -

Thus we have successfully designed a system which helps the user to work with the billing cycles, paying bills, managing different departments under which employees are working, managing various complaints and working on them etc. It also reduces the amount of manual data entry and gives greater efficiency. The User Interface of it is very friendly and can be easily used by anyone. It also decreases the amount of time taken to write details and other modules.