Department of Computer Science and Engineering

CSE 207-Project Report

Course Name: Data Structure

Course Code: CSE 207

Section No: 09

Project Name: Call Distributer In Call Centre

Date of submission: 5/30/2024

Submitted To:

Tanni Mittra

Senior Lecturer
Department of Computer Science & Engineering

Submitted By:

Group Member:

Student's Name: Saiful Islam

Student's ID: 2022-3-60-045

Student's Name: Redwan Ahmed

Student's ID: 2022-3-60-155

Student's Name: Umme Mukaddisa

Student's ID: 2022-3-60-317

Project Description

Introduction: Call Distributor in Call Center is an object-oriented based application where we have used the core concept of Data Structure. This application has most three important features as a call distributor which are entering Customer or entering as an agent of the call center and lastly entering as a server. Those main features also carry different type of function to store and renew Data . Therefore, we can say this amazing application can be useful to any real life call center.

Project Explanation:

As Main features and the first look of our project we have used the below features for our project –

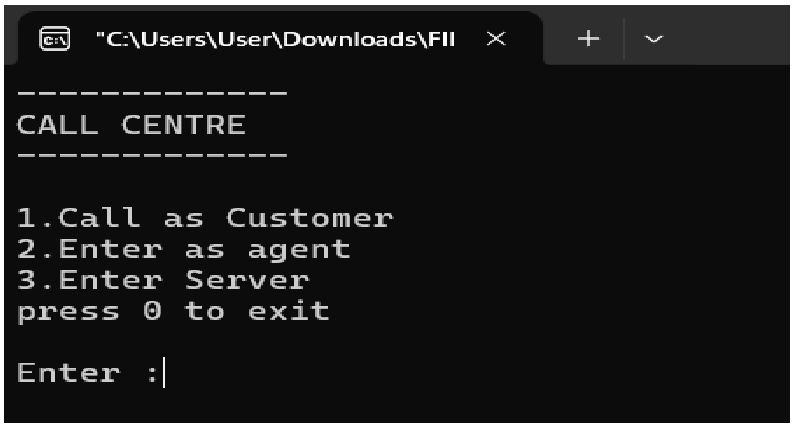
1- Call as Customer

2- Entering as Agent

3- Enter Server

4- Exit , if we intend to take the exit

After running the code first hand we will be asked to choose our likable option.

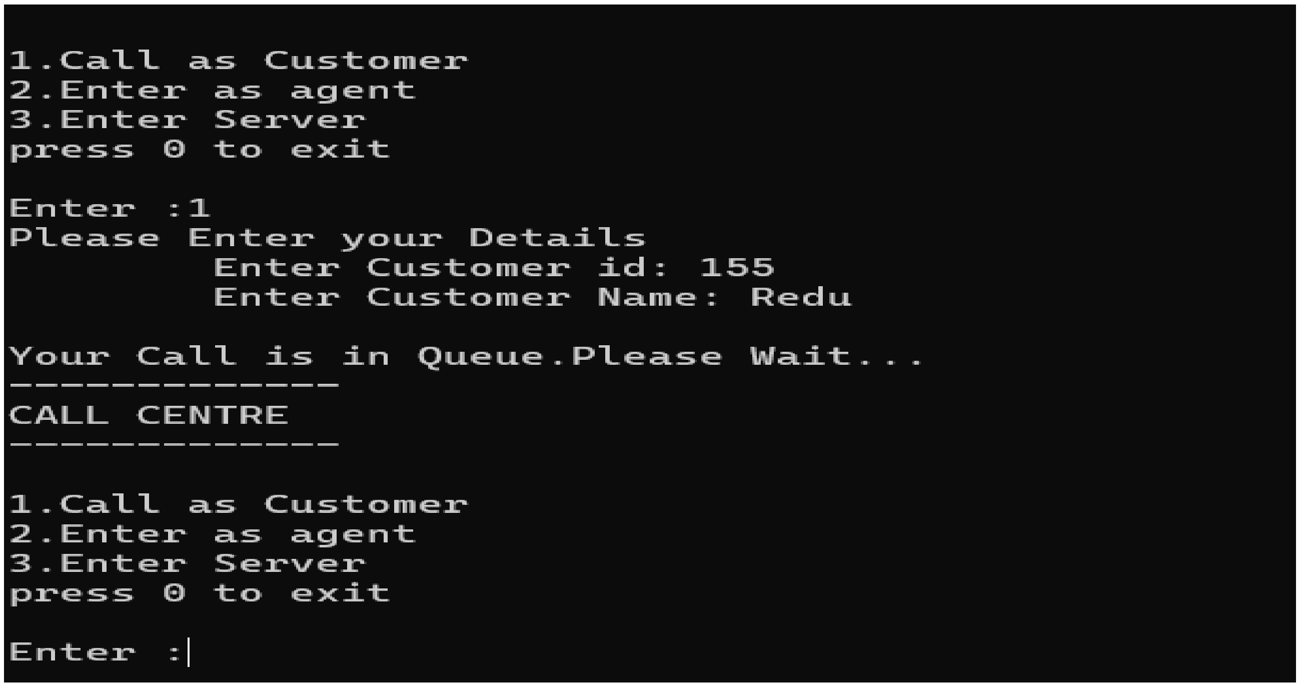


```
-------------
CALL CENTRE
-------------

1.Call as Customer
2.Enter as agent
3.Enter Server
press 0 to exit

Enter :
```

Now we will get into the features and see how those features works as an application.

Call As Customer:

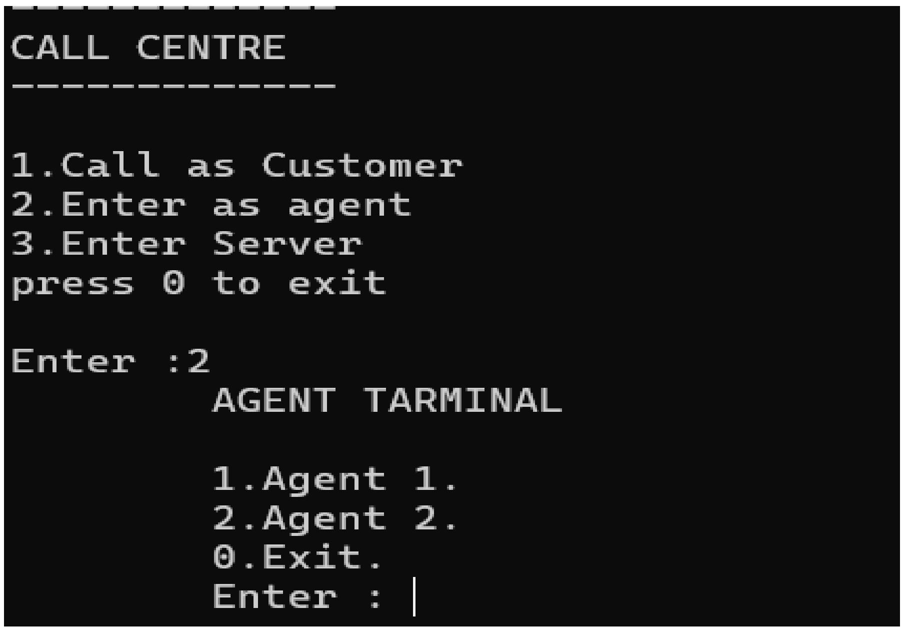
Here if we choose the option 1 we will enter the application as a customer .As a customer we will be asked to give some information like customer ID and name .Those information will be saved using the concept queue in a list. As a response the application will show a massage like this " Your call is now in queue .please wait ".Now that the call is given to the center we will be asked again if we want to enter the application as Customer ,agent or server . If the first option is choose which is "call as customer" than we can give another call to the queue as a customer.



```
1.Call as Customer
2.Enter as agent
3.Enter Server
press 0 to exit

Enter :1
Please Enter your Details
        Enter Customer id: 155
        Enter Customer Name: Redu

Your Call is in Queue.Please Wait...
-------------
CALL CENTRE
-------------

1.Call as Customer
2.Enter as agent
3.Enter Server
press 0 to exit

Enter :
```

Enter As Agent:

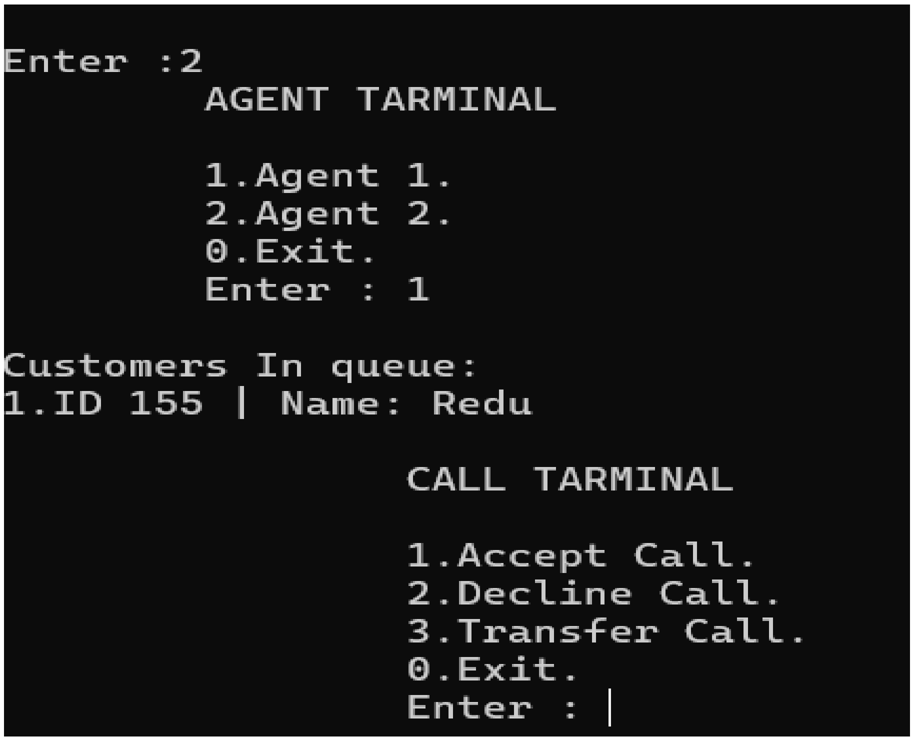
Our option 2 is " Enter as Agent" which is not for the use of common outside people .This features is for the inside worker of the company who will receive the call of customer whenever they like .As anyone who enter as an agent will be in agent terminal and he will be asked if he wants to enter as Agent -1or Agent-2 . The choice is upto if he wants to join as agent 1 or agent 2 or choose to exit .



```
-------------
CALL CENTRE
-------------

1.Call as Customer
2.Enter as agent
3.Enter Server
press 0 to exit

Enter :2
        AGENT TARMINAL

        1.Agent 1.
        2.Agent 2.
        0.Exit.
        Enter : |
```

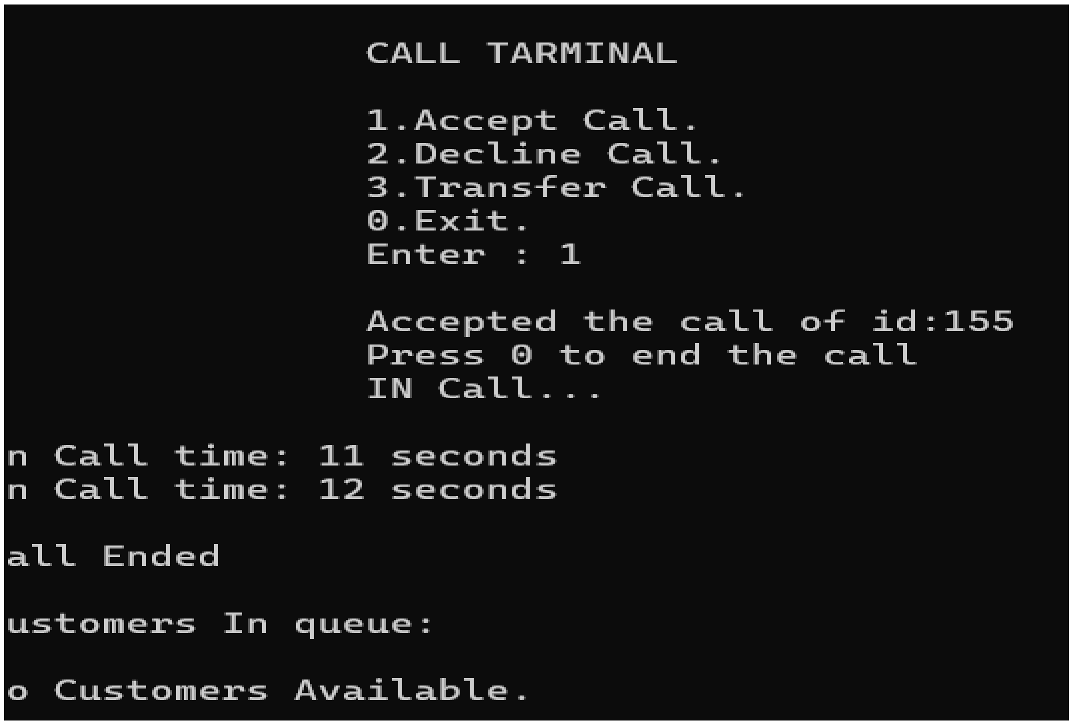
As we enter the terminal as agent 1 we will be shown the list of customer in queue for agent 1 with their name and ID . As the agent sees the customer in queue he will have 4 option in his which are

1-accept call
2-Decline Call
3-Transfer Call
4-Exit   which is the last option



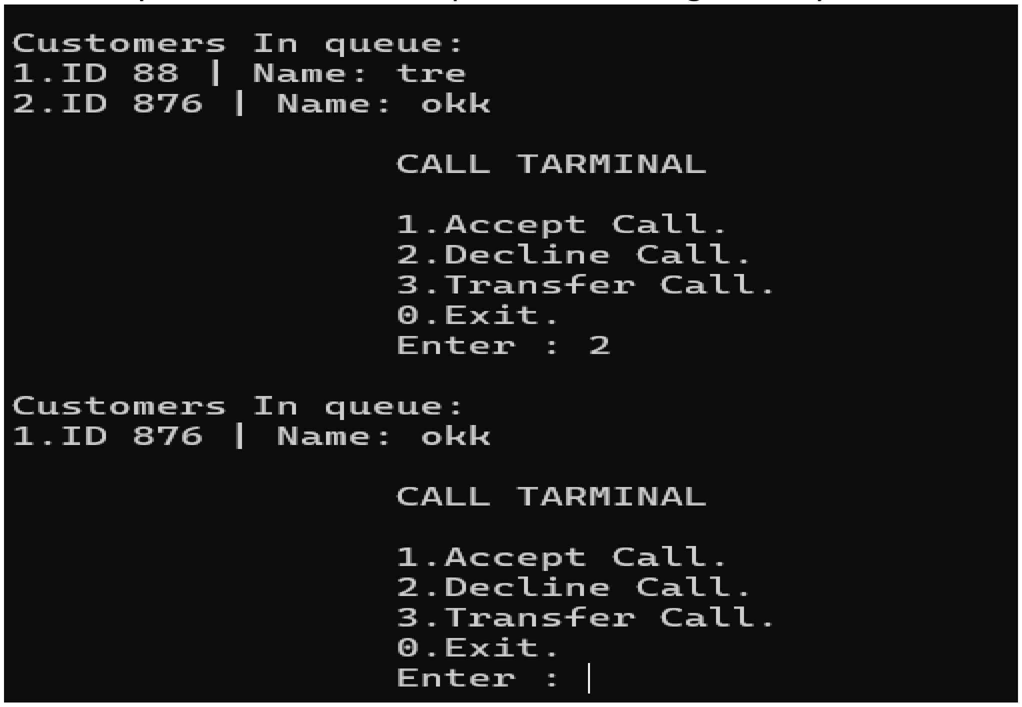
```
Enter :2
        AGENT TARMINAL

        1.Agent 1.
        2.Agent 2.
        0.Exit.
        Enter : 1

Customers In queue:
1.ID 155 | Name: Redu

                CALL TARMINAL

                1.Accept Call.
                2.Decline Call.
                3.Transfer Call.
                0.Exit.
                Enter : |
```

As we choose "1-accept call " first customer will be connected to agent 1 and ID of the customer will be shown along with that until the call is disconnected (which is by typing 0) the call time will be recorded with the computation function .



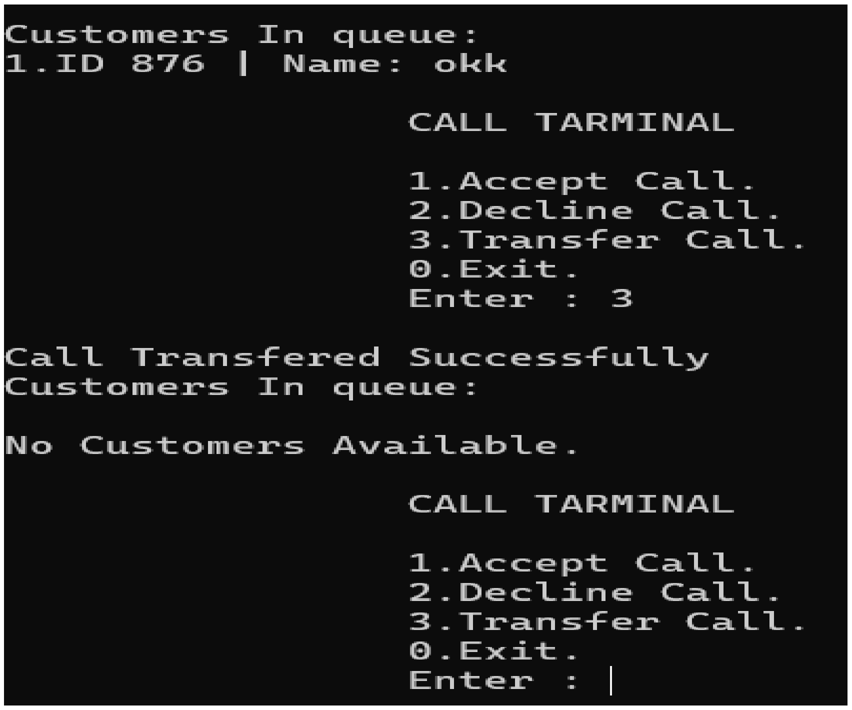
```
               CALL TARMINAL

               1.Accept Call.
               2.Decline Call.
               3.Transfer Call.
               0.Exit.
               Enter : 1

               Accepted the call of id:155
               Press 0 to end the call
               IN Call...

n Call time: 11 seconds
n Call time: 12 seconds

all Ended

ustomers In queue:

o Customers Available.
```

If we choose the option “2-Decline call’’ the first customer call will be declined and if any customer is in the queue for that agent they will shown .



```
Customers In queue:
1.ID 88 | Name: tre
2.ID 876 | Name: okk

                CALL TARMINAL

                1.Accept Call.
                2.Decline Call.
                3.Transfer Call.
                0.Exit.
                Enter : 2

Customers In queue:
1.ID 876 | Name: okk

                CALL TARMINAL

                1.Accept Call.
                2.Decline Call.
                3.Transfer Call.
                0.Exit.
                Enter : |
```

Now with the last option which is “3-transfer Call” if we choose this option the first customer in the queue list will be transferred to another available agent .After that if any customer is in the queue for this agent it will be shown or else the application will show “ No customer is available “.

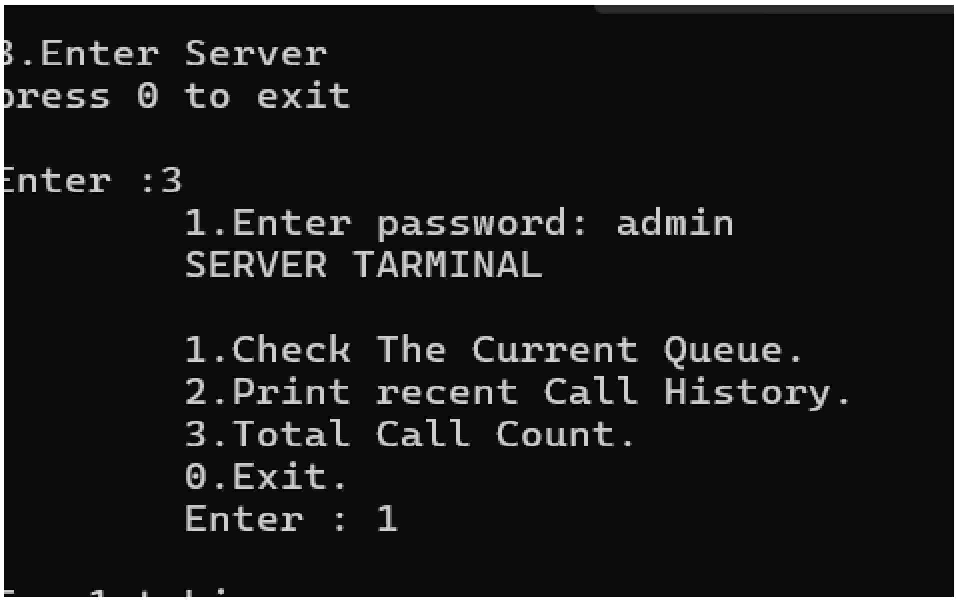


```
Customers In queue:
1.ID 876 | Name: okk

                CALL TARMINAL

                1.Accept Call.
                2.Decline Call.
                3.Transfer Call.
                0.Exit.
                Enter : 3

Call Transfered Successfully
Customers In queue:

No Customers Available.

                CALL TARMINAL

                1.Accept Call.
                2.Decline Call.
                3.Transfer Call.
                0.Exit.
                Enter : |
```

At last with the help of option ‘4-Exit’ we will take exit from the agent terminal. After coming back to the call center first page . will will see another important option.

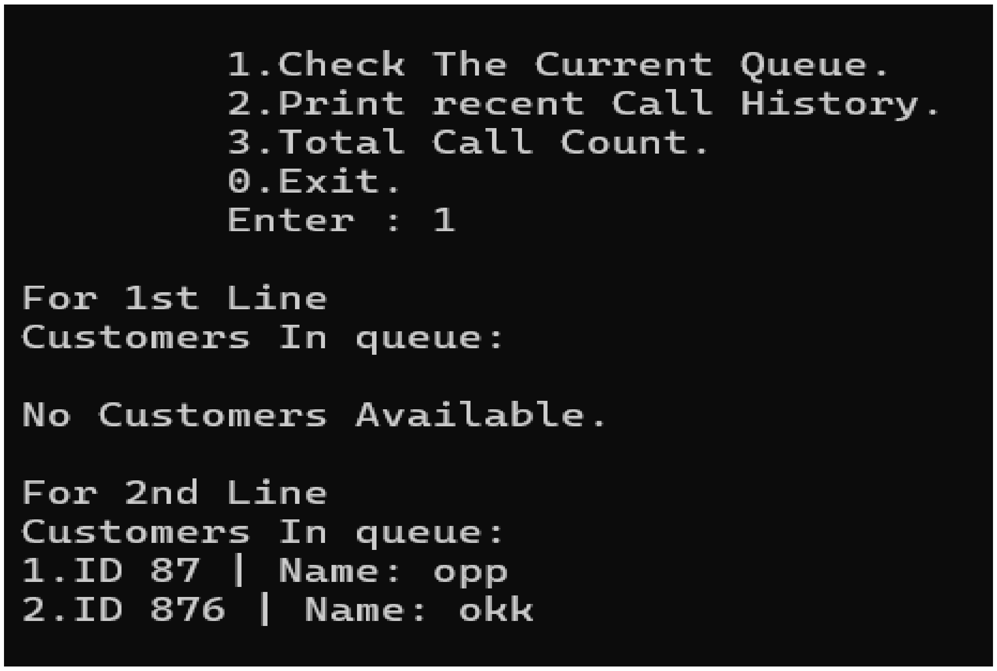
Enter Server:

Enter Server it is a very important and secured site of an application like call canter .Only few people from the organization can access this server for security option . If we choose the option “ 3-Enter Server” we will be asked to give the secured password to access this server . if the password is correct we will to go server terminal and This terminal has 3 features available .
1-check the current queue
2-print recent call history
3-Totall call count



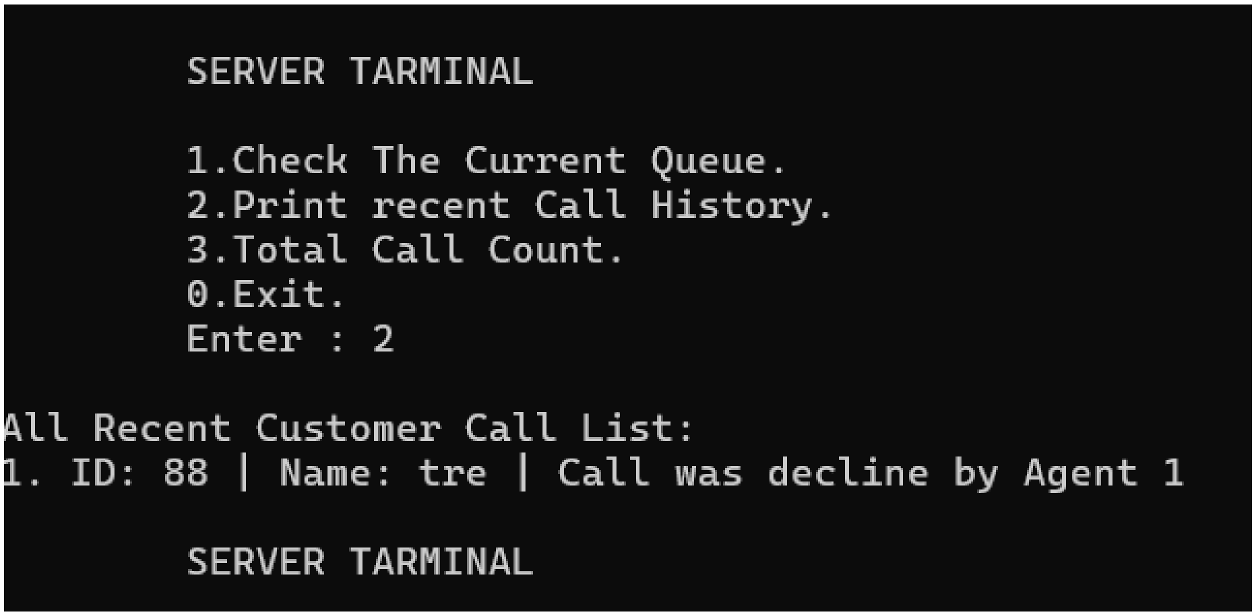
```
3.Enter Server
press 0 to exit

Enter :3
        1.Enter password: admin
        SERVER TARMINAL

        1.Check The Current Queue.
        2.Print recent Call History.
        3.Total Call Count.
        0.Exit.
        Enter : 1
```

If we choose the option 1 we will be showed the current queue for both agent or line , moreover the customers name and ID will also be shown for agents



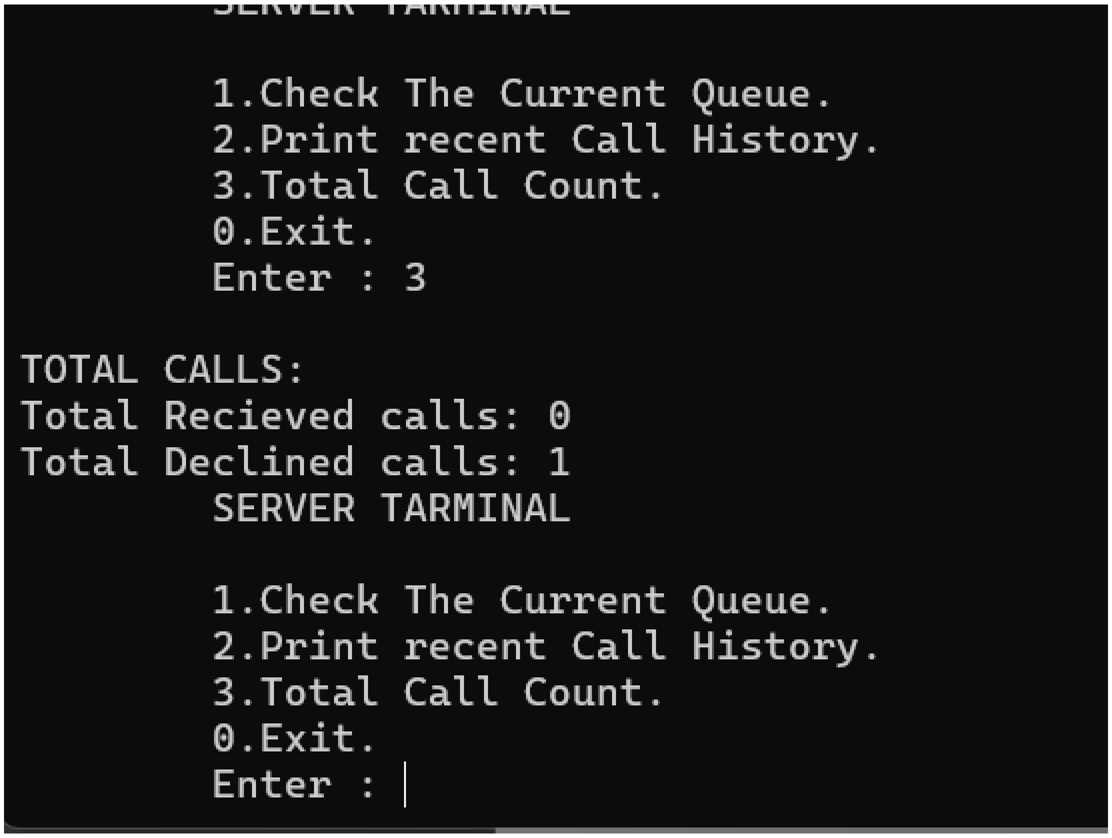
```
         1.Check The Current Queue.
         2.Print recent Call History.
         3.Total Call Count.
         0.Exit.
         Enter : 1

For 1st Line
Customers In queue:

No Customers Available.

For 2nd Line
Customers In queue:
1.ID 87 | Name: opp
2.ID 876 | Name: okk
```

After that coming to the 2nd option "print recent call history" by choosing this option we will see all recent customer list history moreover we can also see those customer name ID and their agent .



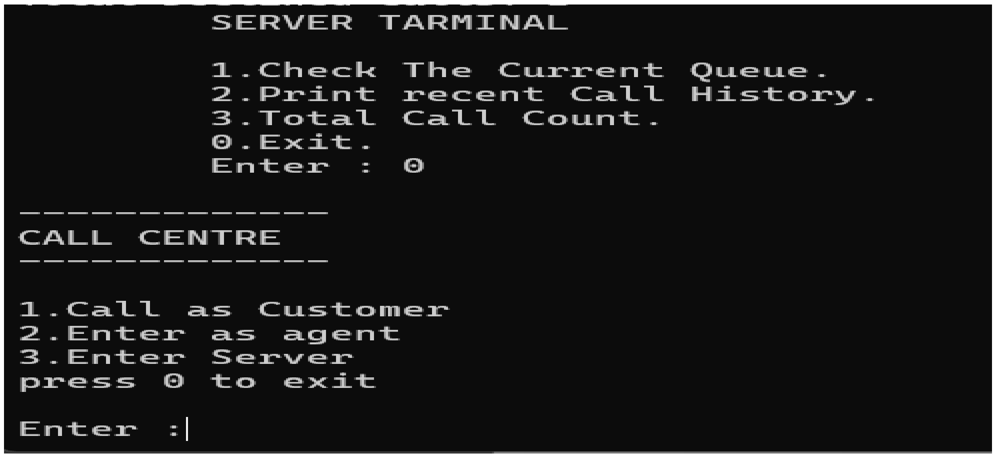
```
        SERVER TARMINAL

        1.Check The Current Queue.
        2.Print recent Call History.
        3.Total Call Count.
        0.Exit.
        Enter : 2

All Recent Customer Call List:
1. ID: 88 | Name: tre | Call was decline by Agent 1

        SERVER TARMINAL
```

At the very last coming to the last option which is “3-total call count” it’s an important feature in this application because this feature store
“Total received call “
“Total declined call”



```
          1.Check The Current Queue.
          2.Print recent Call History.
          3.Total Call Count.
          0.Exit.
          Enter : 3

TOTAL CALLS:
Total Recieved calls: 0
Total Declined calls: 1
          SERVER TARMINAL

          1.Check The Current Queue.
          2.Print recent Call History.
          3.Total Call Count.
          0.Exit.
          Enter :
```

After typing the “0-Exit “ we can take exit from those terminal .



```
          SERVER TARMINAL

          1.Check The Current Queue.
          2.Print recent Call History.
          3.Total Call Count.
          0.Exit.
          Enter : 0

-------------
CALL CENTRE
-------------

1.Call as Customer
2.Enter as agent
3.Enter Server
press 0 to exit

Enter :
```

Limitation:

In this project of Call Distributor in Call Canter we have used features link enter as customer, Enter as agent and lastly Enter as Server. we also used various function which made the application more applicable and useful But in real world where there are much more complexity we could have used more features to solve those problems also we could have use two terminal based system where one terminal will be used as customer interface and another terminal as agent so that we can make real time call simulation to make it more realistic. Also we could have add realtime chatbox to make the communication more efficient. Therefore this project lack the applicability for real world use .

Conclusion:

The project sets a solid foundation for future enhancements.
Overall, the Call Distributor project has successfully met its objectives, providing a robust solution to improve call center operations. The use of C++ has proven advantageous in achieving the desired performance and efficiency. This project not only addresses current needs but also positions the call center for future growth and technological advancements. The positive outcomes observed during the implementation phase encourage us to continue exploring innovative solutions to further enhance our call center operations.

Source Code:

Main:

```
#include "customer.h"
#include "server.h"
#include "basic.h"
using namespace std;
int main()
{

   customerL *Clist=new customerL();
   customerL *Clist2=new customerL();
   server *svr = new server();

   int m=-1;
   while(m!=0)
   {
      cout<<"-------------"<<endl;
      cout<<"CALL CENTRE"<<endl;
      cout<<"-------------"<<endl;
      cout<<endl;
      cout<<"1.Call as Customer"<<endl;
      cout<<"2.Enter as agent"<<endl;
      cout<<"3.Enter Server"<<endl;


      cout<<"press 0 to exit"<<endl<<endl;
      cout<<"Enter :";

      cin>>m;
      if(m==0)
      {
         break;
      }

      if(m==1)
      {
```

```
    cout<<"Please Enter your Details"<<endl;
    cout<<"\tEnter Customer id: ";
    int id;
    string name;
    cin>>id;
    cout<<"\tEnter Customer Name: ";
    cin>>name;
    if(Clist2->size()>=Clist->size())
        Clist->Cus_enQ(id,name);
    else
        Clist2->Cus_enQ(id,name);
    cout<<endl;
    cout<<"Your Call is in Queue.Please Wait...";

}


if(m==2)
{
    int a=-1;
    while(a!=0)
    {
        cout<<"\tAGENT TARMINAL"<<endl<<endl;
        cout<<"\t1.Agent 1."<<endl;
        cout<<"\t2.Agent 2."<<endl;
        cout<<"\t0.Exit."<<endl;
        cout<<"\tEnter : ";

        cin>>a;
        if(a==0)
            break;
        else if(a==1)
        {
            int a=-1;
            while(a!=0)
            {
                Clist->display();
                cout<<"\t\tCALL TARMINAL"<<endl<<endl;
                cout<<"\t\t1.Accept Call."<<endl;
                cout<<"\t\t2.Decline Call."<<endl;
                cout<<"\t\t3.Transfer Call."<<endl;
                cout<<"\t\t0.Exit."<<endl;
                cout<<"\t\tEnter : ";

                cin>>a;
                if(a==0)
                    break;
                else if(a==1)
                {
                    cout<<endl;
            cout<<"\t\tAccepted the call of id:"<<Clist->topd()<<endl;
            cout<<"\t\tPress 0 to end the call"<<endl;
            cout<<"\t\tIN Call..."<<endl<<endl;
            int t;
            t=calculateExecutionTime(exampleFunction);
            svr->push(Clist->topd(),t,Clist->topa()->name,"Agent 1");
            Clist->Cus_dQ();

        }
        else if(a==2)
        {
            svr->push(Clist->topd(),Clist->topa()->name,"Agent 1");
            Clist->Cus_dQ();
        }
        else if(a==3)
        {
            Clist2->Cus_enQ(Clist->topa()->id,Clist->topa()->name);
            Clist->Cus_dQ();
            cout<<endl;
            cout<<"Call Transfered Successfully";
        }

    }

}
else if(a==2)
{
    int a=-1;
    while(a!=0)
    {
        Clist2->display();
        cout<<"\t\tCALL TARMINAL"<<endl<<endl;
        cout<<"\t\t1.Accept Call."<<endl;
        cout<<"\t\t2.Decline Call."<<endl;
        cout<<"\t\t3.Transfer Call."<<endl;
        cout<<"\t\t0.Exit."<<endl;
        cout<<"\t\tEnter : ";

        cin>>a;
        if(a==0)
            break;
        else if(a==1)
        {
            cout<<endl;


            cout<<"\t\tAccepted the call of id:"<<Clist2->topd()<<endl;
            cout<<"\t\tPress 0 to end the call"<<endl;
            cout<<"\t\tIN Call..."<<endl<<endl;
            int t;
```

```
               t=calculateExecutionTime(exampleFunction);
               svr->push(Clist2->topd(),t,Clist2->topa()->name,"Agent 1");
               Clist2->Cus_dQ();

            }
            else if(a==2)
            {
               svr->push(Clist2->topd(),Clist2->topa()->name,"Agent 1");
               Clist2->Cus_dQ();
            }
            else if(a==3)
            {
               Clist->Cus_enQ(Clist2->topa()->id,Clist2->topa()->name);
               Clist2->Cus_dQ();
               cout<<endl;
               cout<<"Call Transfered Successfully";
            }

         }



      }

   }

}

if(m==3)
{
   cout<<"\t1.Enter password: ";
   string a;
   cin>>a;
   if(a=="admin")
   {


      int a=-1;
      while(a!=0)
      {

         cout<<"\tSERVER TARMINAL"<<endl<<endl;
         cout<<"\t1.Check The Current Queue."<<endl;
         cout<<"\t2.Print recent Call History."<<endl;
         cout<<"\t3.Total Call Count."<<endl;
         cout<<"\t0.Exit."<<endl;
         cout<<"\tEnter : ";

         cin>>a;
         if(a==0)
            break;
            else if(a==1)
            {
               cout<<endl;
               cout<<"For 1st Line";
               Clist->display();
               cout<<"For 2nd Line";
               Clist2->display();
            }
            else if(a==2)
            {
               svr->display();
            }
            else if(a==3)
            {
               cout<<endl;
               cout<<"TOTAL CALLS:"<<endl;
               cout<<"Total Recieved calls: "<<svr->R_call<<endl;
               cout<<"Total Declined calls: "<<svr->D_call<<endl;
            }

         }




      }
      else
         cout<<"\tInvalid Password";
   }


   if(m<0 || m>10)
      cout<<endl<<"Invalid Option"<<endl;



   cout<<endl;
}


}
```

Server

```
#include <iostream>
using namespace std;
class cus_s{
   public:
int id;
bool decline=false;
int Call_time;
cus_s *next=NULL;
string name;
string agent;
cus_s(){
}
cus_s (int i,string s){
id=i;
name=s;
}

};



class server{
public:
    cus_s *top=NULL,rear,*ptr,*temp;
   int R_call=0;
   int D_call=0;
    void push(int a,int t,string s,string agnt)
   {   R_call++;
       ptr=new cus_s(a,s);
       ptr->Call_time=t;
       ptr->agent=agnt;
       if(top==NULL)
   {
       top=ptr;
   }
   else
   {
       ptr->next=top;
       top=ptr;
    }
    }
    void push(int a,string s,string agnt)
    {   D_call++;
       ptr=new cus_s(a,s);
       ptr->decline=true;
       ptr->agent=agnt;
       if(top==NULL)
    {
       top=ptr;
    }
    else
    {
       ptr->next=top;
       top=ptr;
    }
    }

     void pop(){
       temp=top;
       top=top->next;
       delete temp;
    }



    int size(){
    int count=0;
    temp=top;
    while(temp!=NULL){
       count++;
       temp=temp->next;
    }
    return count;
    }


    void display(){
    temp=top;int i=0;
    cout<<endl;
    cout<<"All Recent Customer Call List: "<<endl;
    if(top==NULL){
          cout<<endl;
       cout<<"No Customers Available."<<endl;
    }
   while(temp!=NULL){

       i++;
   if(temp->decline)
       cout<<i<<". ID: "<<temp->id<<" | Name: "<<temp->name<< " | Call was decline by
"<<temp->agent;
```

```
   else
        cout<<i<<". ID: "<<temp->id<<" | Name: "<<temp->name<< " | Agent: "<<temp->agent<< "
| Call Duration: "<<temp->Call_time<<"s.";
        cout<<endl;
        temp=temp->next;
    }
    cout<<endl;
    }

};
```

Customer

```
#include <iostream>

using namespace std;

class customer{

   public:

int id;

string name;

customer *next=NULL;


customer(){

}

customer (int a,string s){

id=a;

name=s;

}


};
```

```
class customerL{
public:
     customer *frnt=NULL,rear,*ptr,*temp;



     void Cus_enQ(int a,string s)
    {
        ptr=new customer(a,s);
        if(frnt==NULL)
    {
        frnt=ptr;
//        rear=ptr;
    }
    else
    {
        temp=frnt;
        while(temp->next!=NULL)
            {
                temp=temp->next;
            }
            temp->next=ptr;
//            rear=ptr;
        }
    }

     void Cus_dQ(){
      temp=frnt;
        frnt=frnt->next;
        delete temp;
```

```
 }



 int size(){
int s=0;
temp=frnt;
while(temp!=NULL){
   s++;
   temp=temp->next;
}
return s;
}



int topd(){


   return frnt->id;


}
customer* topa(){
   return frnt;
}

void display(){
temp=frnt;int i=0;
 cout<<endl;
cout<<"Customers In queue: "<<endl;
if(frnt==NULL){
       cout<<endl;
```

```
        cout<<"No Customers Available."<<endl;

    }

   while(temp!=NULL){

        i++;

        cout<<i<<".ID "<<temp->id<<" | Name: "<<temp->name;

        cout<<endl;

        temp=temp->next;

    }

    cout<<endl;

    }

};
```

Time computation

```
#include<iostream>
#include<chrono>
#include<complex>
#include<thread>
#include <atomic>
using namespace std;
void exampleFunction() {
   int a;



     cin>>a;
  if(a==0){
     this_thread::sleep_for(chrono::seconds(1));



  }
}

// Function to calculate and return the execution time of a function
double calculateExecutionTime(void (*func)()) {
  atomic<bool> done(false);
  auto start = chrono::high_resolution_clock::now();
```

```
    // Start a thread to display the elapsed time
    thread timerThread([&]() {
        while (!done.load()) {
            auto now = chrono::high_resolution_clock::now();
            chrono::duration<double> elapsed = now - start;
            cout << "\t\rIn Call time: " << static_cast<int>(elapsed.count()) << " seconds" << flush;
            this_thread::sleep_for(chrono::seconds(1));
        }
    });

    // Call the function
    func();
    cout<<endl;
    cout<<endl;
        cout<<"\t\rCall Ended"<<flush;
        cout<<endl;
    // Get the ending time
    auto end = chrono::high_resolution_clock::now();

    // Indicate that the function is done
    done.store(true);
    timerThread.join(); // Wait for the timer thread to finish

    // Calculate the duration
    chrono::duration<double> duration = end - start;

    // Return the duration in seconds
    return duration.count();
}
```