<u>SPL-1 Project Report</u>

Room Raccoon:
Hotel Management System

Submitted by

Md. Shaikhul Islam

BSSE Roll No. : 1438

BSSE Session: 2021-2022

Submitted to

Assistant Prof. Abdus Satter

IIT, Dhaka University.

Supervisor' Approval: ______________________

Date: **(Signature)**

Institute of Information Technology

University of Dhaka

Date: 17.12.2023

Table of Contents

1. Introduction:

The whole world is being digitized through different systems. Inclusion of software is a big step to that. Such a system is built named **"Room Raccoon"**. This software based system, **"Room Raccoon"**, is a digital hotel management system. By this system all type of functionality can be executed and this is convenient to use. Here there are two type of login interface - one is for Manager, and the other one is for the Clients. After accessing the secured login system, the respective users will have their options of access.

The authoritative user is the manager interface, where the manager can conduct his/her duty and observation online without being present physically. The manager can see the current clients, the dues with their lists, working employee, the records of previous and current clients, also and if there are any complains or suggestions from the clients. The manager can communicate with the client if necessary and any query arises. The mutual connectivity will also be preserved.

This service is mainly for the clients in need of temporary residence and meal service. So this system contains all the necessary supports to regulate from remote place. They can book and get service through online process. The Clients can create a profile using their name, phone number and password which will be kept encrypted inside. After logging in, the clients can book any room as their choice depending on availability. They can see all the available packages with the corresponding prices and the food system while enrolling. The clients can check their current status of the packages, systems and also the payment status. They can also change their current packages and their password through this menu. While checking out, the client can see the payment and status and can give a review and rating. Even while staying, he/she can place any complains or suggestions. There is a feature to connect with the authority, in case of any query or any problem. The hotel authority will directly communicate with them.

Along with the modernization of the world, in every sector of life is needed to be upgraded. That's why digitization is needed everywhere. With a view to claiming that, all type of office work is taken under modern machines. Nevertheless, the social organizations are also doing the same. In the management sector of hotel, leaving the pen-paper manual method adapting with digital system is certainly difficult. But to do so, a convenient and user-friendly is needed. That's the motive of my work. Along with this, the security and privacy of the data is very important. That's why the matter of encryption-decryption has been initiated. To fulfill and manage the prerequisites of people, this project is being developed.

Mainly, this will be a software based management system with various functionality and the data will be preserved by a secured system produced by some complex encryption algorithm and multi device connection system.

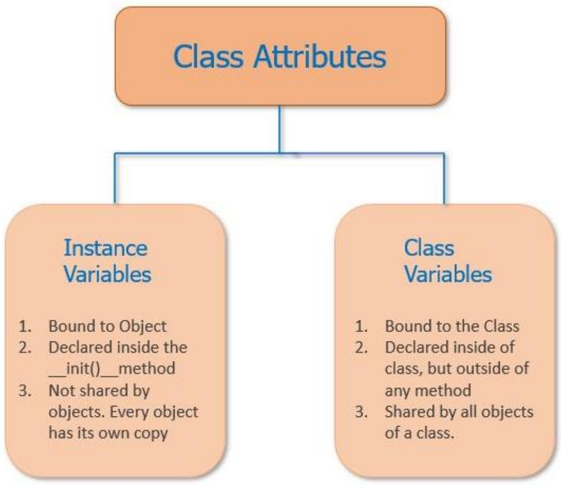
2. Background of the Project

The main topics that are related to my projects are:

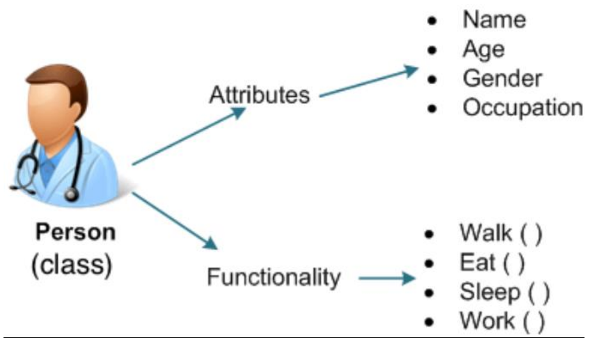
1. **Class-Object Manipulation.**
2. **AES Encryption Algorithm**
3. **Socket programming**
4. **Management System**

Class-Object Concept:

A Class is a user-defined data type that has data members and member functions. Data members are the data variables and member functions are the functions used to manipulate these variables together, these data members and member functions define the properties and behavior of the objects in a Class. An Object is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated. When a class is defined, only the specification for the object is defined; no memory or storage is allocated. To use the data and access functions defined in the class, objects need to be created.



Class Attributes
Instance Variables
1. Bound to Object
2. Declared inside the __init()__method
3. Not shared by objects. Every object has its own copy
Class Variables
1. Bound to the Class
2. Declared inside of class, but outside of any method
3. Shared by all objects of a class.



Introduction of class has emerged a great change in the structure in Object Oriented Programing. It made the code easily understandable and editable. It's uses is also easy and convenient, makes the code more efficient. The variables under the class is known as attributes and the functions of it is known as methods. Through this mainly the OOP has been implemented. Fundamentals of OOP are:

Inheritance: When one object acquires all the properties and behaviors of a parent object, it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.

Polymorphism: If one task is performed in different ways, it is known as polymorphism. Method overloading and method overriding is used to achieve polymorphism.

Abstraction: Hiding internal details and showing functionality is known as abstraction. For example phone no, addresses etc, we don't know the internal processing.

Encapsulation: Binding (or wrapping) code and data together into a single unit are known as encapsulation. A class is the example of encapsulation. An object is fully encapsulated because all the data members are private here.

Class-object or OOP concept in programming has lead the works much easier and modern to achieve the desired results.

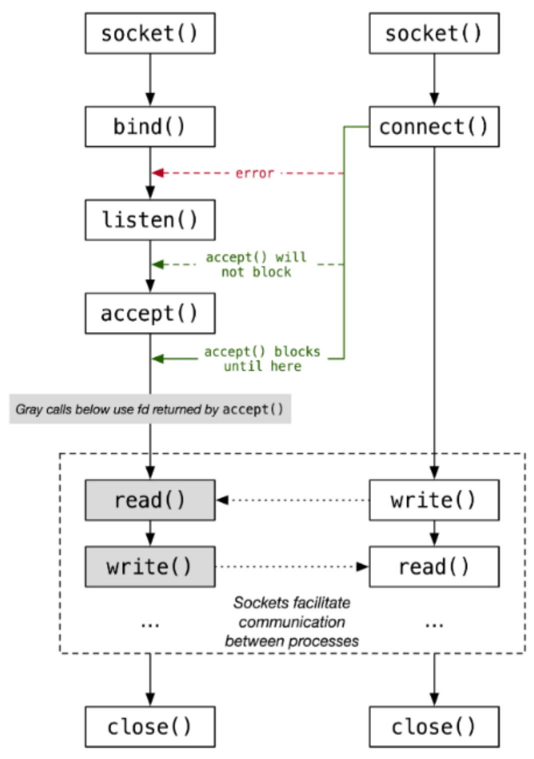
AES Algorithm:

For secured data preservation, different encryption algorithms have been created and are being used. Among them AES is currently the most effective and used one. The Advanced Encryption Standard (AES) is a symmetric encryption algorithm widely adopted for its robust security and performance. It was established as a replacement for the Data Encryption Standard (DES) by the National Institute of Standards and Technology (NIST) in 2001. AES operates on fixed-size blocks of data and supports key sizes of 128, 192, and 256 bits. The algorithm employs a combination of substitution, permutation, and bitwise operations to provide confidentiality, integrity, and authenticity of the transmitted data. AES encryption finds widespread use in secure communication, data storage, and cryptographic protocols. Some features of AES are as follows:

1. SP Network: It works on an SP network structure rather than a Festal cipher structure, as seen in the case of the DES algorithm.
2. Key Expansion: It takes a single key up during the first stage, which is later expanded to multiple keys used in individual rounds.
3. Byte Data: The AES encryption algorithm does operations on byte data instead of bit data. So it treats the 128-bit block size as 16 bytes during the encryption procedure.
4. Key Length: The number of rounds to be carried out depends on the length of the key being used to encrypt data. The 128-bit key size has ten rounds, the 192-bit key size has 12 rounds, and the 256-bit key size has 14 rounds. Here in the project 128 bit-key is used.

Socket Programming:

Socket Programming is a method to connect two nodes over a network to establish a means of communication between those two nodes. A node represents a computer or a physical device with an internet connection. A socket is the endpoint used for connecting to a node. The signals required to implement the connection between two nodes are sent and received using the sockets on each node respectively.



The nodes are divided into two types, **server** node and **client** node. The client node sends the connection signal and the server node receives the connection signal sent by the client node. The connection between a server and client node is established using the socket over the transport layer of the internet. After a connection has been established, the client and server nodes can share information between them using the read and write commands. After sharing of information is done, the nodes terminate the connection.

Mainly the communication build-up is done through socket-programming. Its uses is vast. A simple glimpse is used here in this project.

Management System:

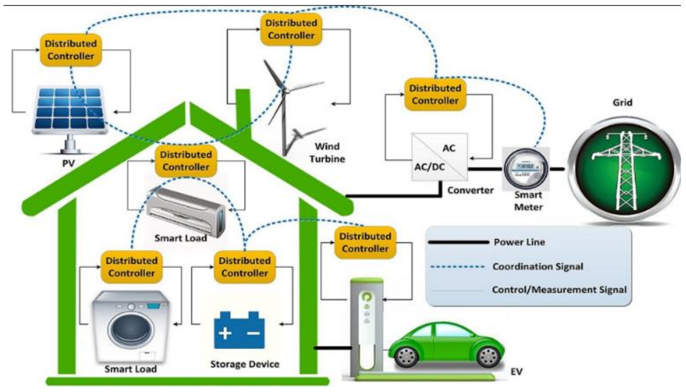
A management system describes the way in which companies organize themselves in their structures and processes in order to act systematically, ensure smooth processes and achieve planned results Modern management systems usually follow the PDCA cycle of planning, implementation, review and improvement (Plan-Do-Check-Act). This also implements in software production.

Management systems can be used in all areas - depending on where your company operates and what goals are to be achieved. This can be in a specific industry, such as transport and logistics, the automotive industry or healthcare, or even across industries. In software-farms this has a big role too. Without proper planning and management, the softwares will be vulnerable as well as the data. So it’s an undetachable part in procuring software.

The benefits of an effective management system to an organization includes:

- More efficient use of resources and improved financial performance
- Improved risk management and protection of people and the environment
- Increased capability to deliver consistent and improved services and products, thereby increasing value to customers and all other stakeholders
- Reduced cost and time consumption of production through error prevention
- Uniform structures and clear responsibilities
- Transparent and stable procedures and processes
- Pronounced customer and employee satisfaction
- Systematic achievement of set corporate goals
- Greater legal certainty through risk minimization
- Improved reputation and easier access to new markets

That’s why an efficient and effective management is needed in every type of production and system. It helps to get the desired product perfectly. And the proper planning takes the work in the right direction.

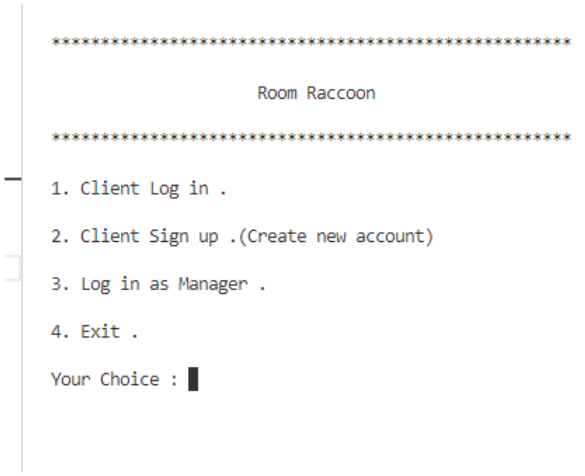


3. Description of the Project:

The project is a specimen of online Hotel System. So it starts with a menu-bar and login system. Here there will be different type of login. Client and Manager Login. There will be Sign up option for new clients.

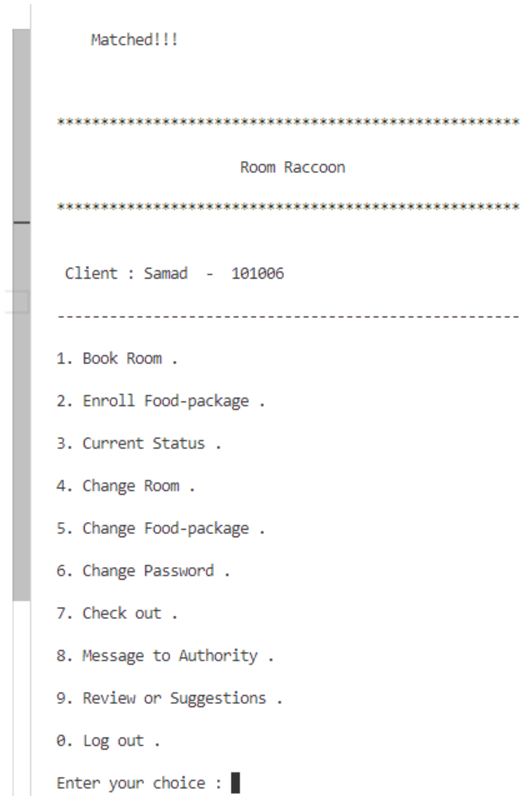
After the secured login process, the user will have different interfaces. There they can pursue the options available for them.

The snippets are given bellow:



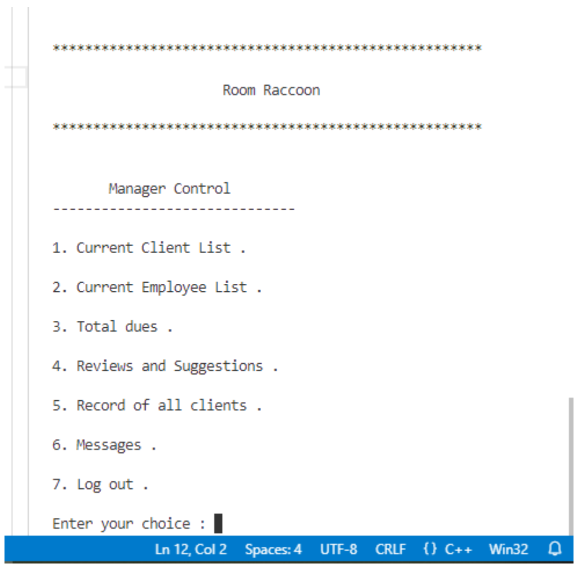
```
*******************************************************

                    Room Raccoon

*******************************************************

1. Client Log in .

2. Client Sign up .(Create new account)

3. Log in as Manager .

4. Exit .

Your Choice :
```

Client Interface:



```
   Matched!!!


*******************************************************

                    Room Raccoon

*******************************************************


 Client : Samad  -  101006

-------------------------------------------------------

1. Book Room .

2. Enroll Food-package .

3. Current Status .

4. Change Room .

5. Change Food-package .

6. Change Password .

7. Check out .

8. Message to Authority .

9. Review or Suggestions .

0. Log out .

Enter your choice :
```

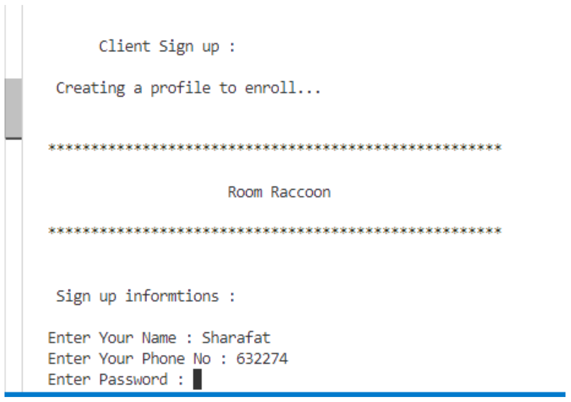
Manager Interface:



```
*******************************************************

                    Room Raccoon

*******************************************************


       Manager Control
-----------------------------

1. Current Client List .

2. Current Employee List .

3. Total dues .

4. Reviews and Suggestions .

5. Record of all clients .

6. Messages .

7. Log out .

Enter your choice :
```

Sign Up Menu:

From this menu, the users can choose their option of interest. The functionality will be done accordingly.

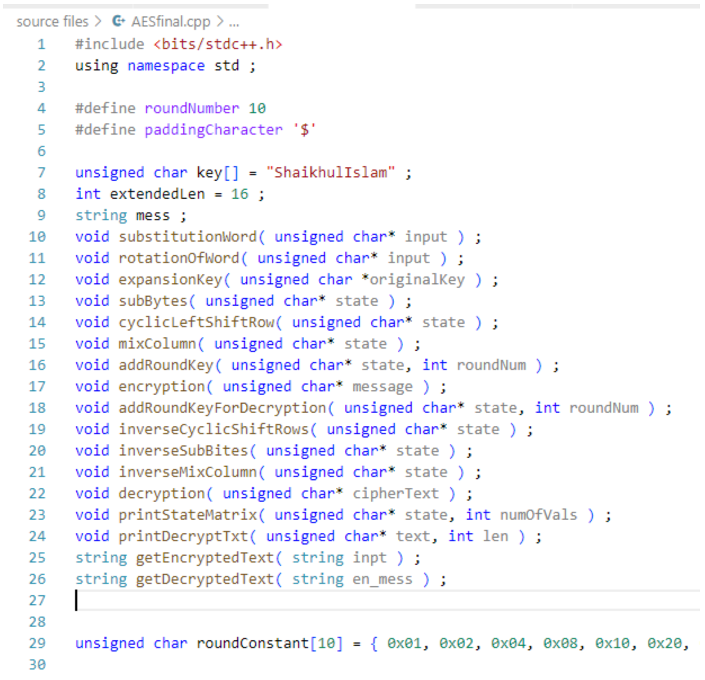


```
      Client Sign up :

  Creating a profile to enroll...


*******************************************************

                    Room Raccoon

*******************************************************


  Sign up informtions :

 Enter Your Name : Sharafat
 Enter Your Phone No : 632274
 Enter Password :
```

The sophisticated data of the Client and Manager is stored in separate files after encryption through AES algorithm. AES is simplified below:

AES ENCRYPTION:

- Generate Key and Perform Key Expansion
- Read file in 128 bit chunks to bytestream and 4x4 state array
- Perform Encryption in 10 rounds:
    - Substitute Byte
    - Shift Row
    - Mix Column
    - Add Round Key
- Write the encrypted contents back to file

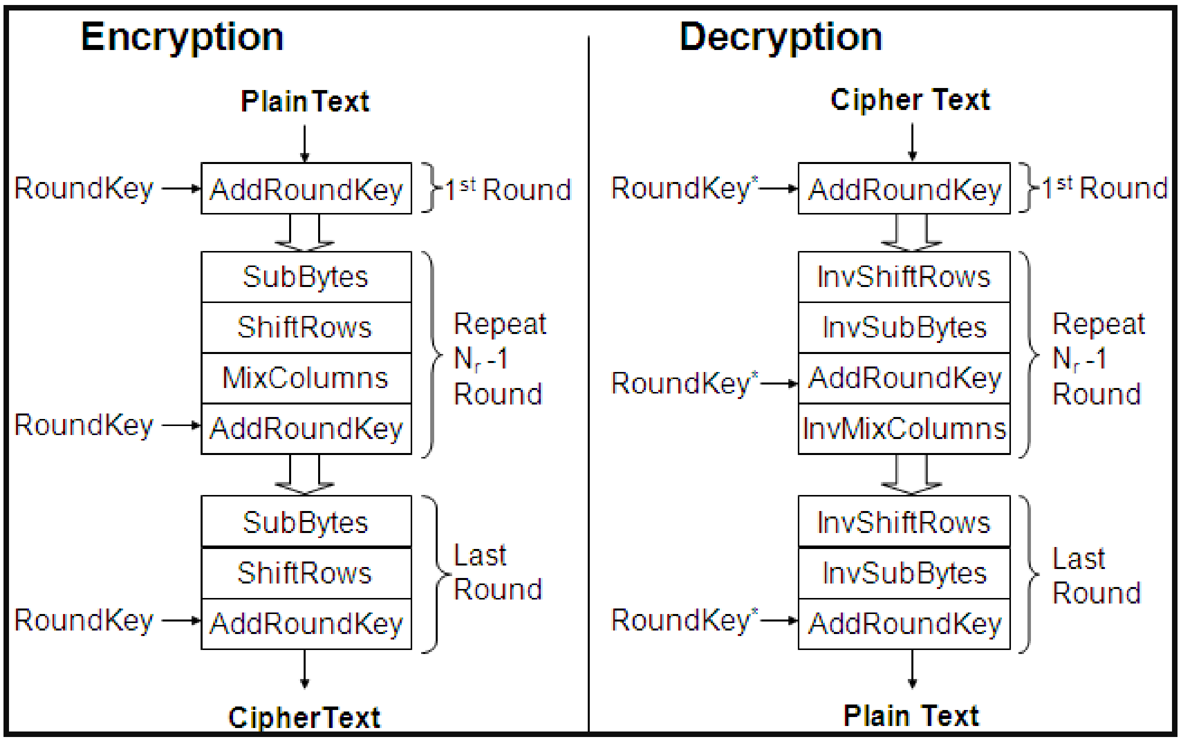


```
#include <bits/stdc++.h>
using namespace std ;

#define roundNumber 10
#define paddingCharacter '$'

unsigned char key[] = "ShaikhulIslam" ;
int extendedLen = 16 ;
string mess ;
void substitutionWord( unsigned char* input ) ;
void rotationOfWord( unsigned char* input ) ;
void expansionKey( unsigned char *originalKey ) ;
void subBytes( unsigned char* state ) ;
void cyclicLeftShiftRow( unsigned char* state ) ;
void mixColumn( unsigned char* state ) ;
void addRoundKey( unsigned char* state, int roundNum ) ;
void encryption( unsigned char* message ) ;
void addRoundKeyForDecryption( unsigned char* state, int roundNum ) ;
void inverseCyclicShiftRows( unsigned char* state ) ;
void inverseSubBites( unsigned char* state ) ;
void inverseMixColumn( unsigned char* state ) ;
void decryption( unsigned char* cipherText ) ;
void printStateMatrix( unsigned char* state, int numOfVals ) ;
void printDecryptTxt( unsigned char* text, int len ) ;
string getEncryptedText( string inpt ) ;
string getDecryptedText( string en_mess ) ;


unsigned char roundConstant[10] = { 0x01, 0x02, 0x04, 0x08, 0x10, 0x20,

```

AES DECRYPTION:

- Read back key from previous stored key
- Read encrypted file to state array
- Decrypt contents by reversing in 14 rounds
- Write decrypted contents back to file

Flowchart of AES Encryption-Decryption:



Socket programming in C++ allows for the implementation of network communication between different devices or processes. Here is a technical overview socket programming implementation in C++:

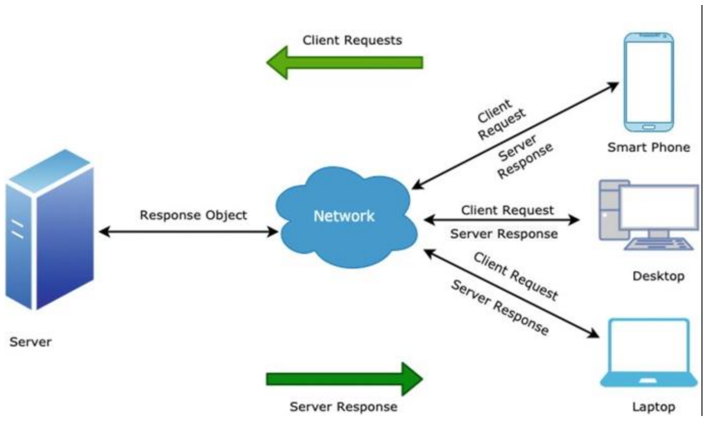
- We Begin by incorporating the essential libraries for C++ socket programming, such as <sys/socket.h>, <netinet/in.h>, and <arpa/inet.h>. These libraries furnish the requisite functions and data structures for tasks like socket creation, binding, listening, accepting connections, and data transmission.

- After that we initiate the socket creation process using the socket () function. Indicate the address family (IPv4 or IPv6) and socket type (TCP or UDP). The function yields a file descriptor representing the socket.

- We associate the socket with a specific address and port using the bind () function. Specify the socket file descriptor, address family, IP address (via sockaddr_in or sockaddr_in6 structure), and port number. This step is crucial for the server to listen on a designated network interface and port.

- Enable the server to listen for incoming connections by employing the listen () function. Provide the socket file descriptor and the maximum number of pending connections that can be queued. This step is pivotal for the server to accept client connections.

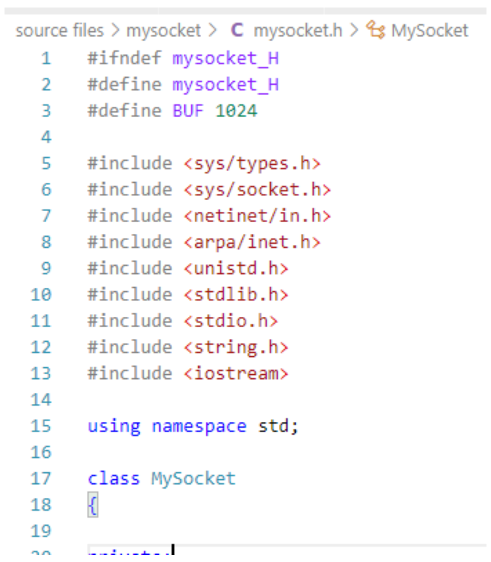
- For the server, use the accept () function to accept incoming client connections. Specify the socket file descriptor and an optional client address structure (sockaddr_in or sockaddr_in6) to store client details. The function returns a new file descriptor representing the accepted connection for further communication.



- For the client, employ the connect () function to establish a connection with the server. Specify the socket file descriptor and the server address structure (sockaddr_in or sockaddr_in6), including the server's IP address and port number. This facilitates bidirectional data transfer.

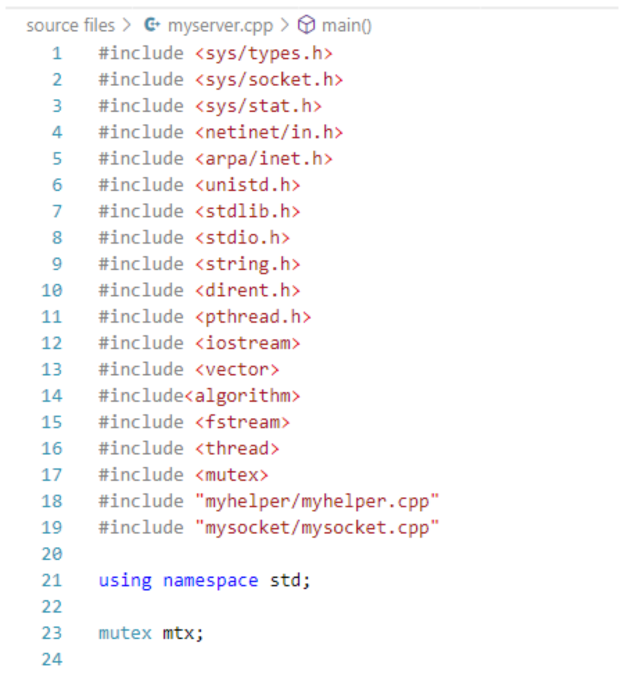
- Utilize the send() and recv() functions for sending and receiving data on sockets. Specify the socket file descriptor, a data buffer, data size, and optional flags for additional control. Data transmission can occur in chunks or through iterative loops until the entire message is sent or received.

- To conclude, release socket resources using the close() function. Specify the socket file descriptor to be closed. This action terminates the connection, freeing up system resources.



source files > mysocket > mysocket.h > MySocket

```cpp
#ifndef mysocket_H
#define mysocket_H
#define BUF 1024

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <iostream>

using namespace std;

class MySocket
{

```



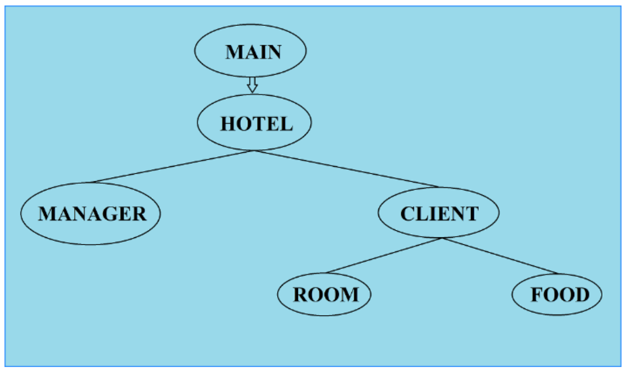
source files > myserver.cpp > main()

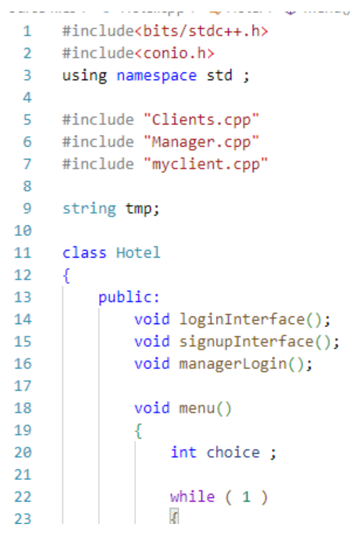
```cpp
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/stat.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <dirent.h>
#include <pthread.h>
#include <iostream>
#include <vector>
#include<algorithm>
#include <fstream>
#include <thread>
#include <mutex>
#include "myhelper/myhelper.cpp"
#include "mysocket/mysocket.cpp"

using namespace std;

mutex mtx;

```

This pictures above are example of the codebase for socket programming in C++.

4. Implementation and Testing:

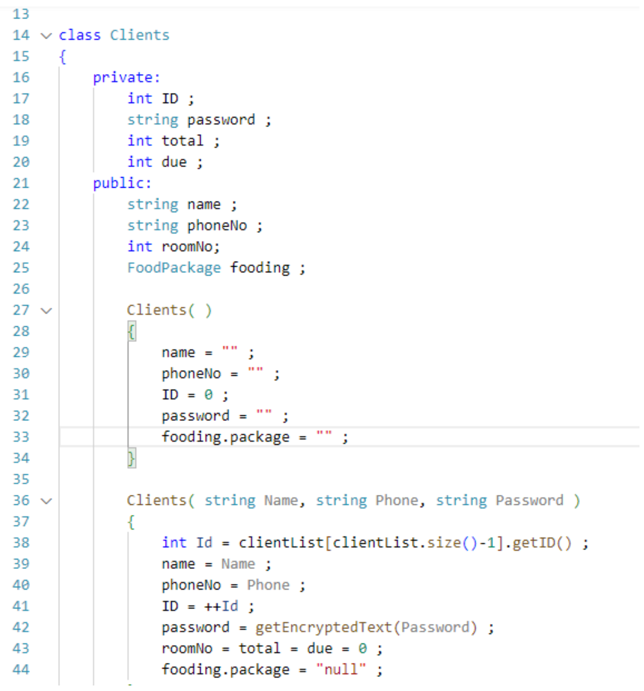
Firstly, the class and object management to conduct the hotel system is performed. The classes are aligned accordingly to manipulate the functions.



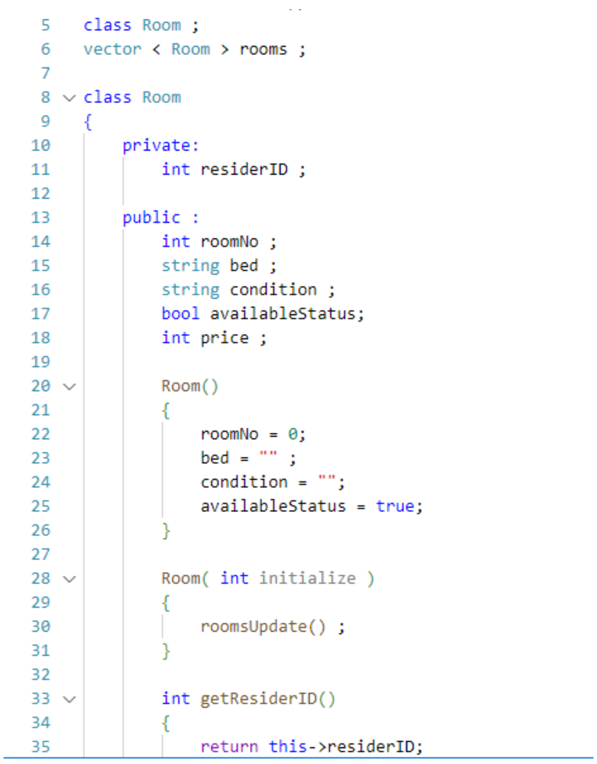


```cpp
#include<bits/stdc++.h>
#include<conio.h>
using namespace std ;

#include "Clients.cpp"
#include "Manager.cpp"
#include "myclient.cpp"

string tmp;

class Hotel
{
    public:
        void loginInterface();
        void signupInterface();
        void managerLogin();

        void menu()
        {
            int choice ;

            while ( 1 )
            {
```

Secondly, the classes are formed and merged. The objects are created to while necessary. But all of them are regulated from the main through the hotel class.



```
class Clients
{
    private:
        int ID ;
        string password ;
        int total ;
        int due ;
    public:
        string name ;
        string phoneNo ;
        int roomNo;
        FoodPackage fooding ;

        Clients( )
        {
            name = "" ;
            phoneNo = "" ;
            ID = 0 ;
            password = "" ;
            fooding.package = "" ;
        }

        Clients( string Name, string Phone, string Password )
        {
            int Id = clientList[clientList.size()-1].getID() ;
            name = Name ;
            phoneNo = Phone ;
            ID = ++Id ;
            password = getEncryptedText(Password) ;
            roomNo = total = due = 0 ;
            fooding.package = "null" ;
```



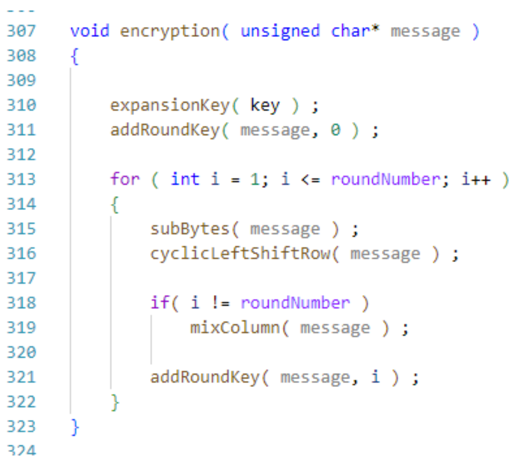
```
class Room ;
vector < Room > rooms ;

class Room
{
    private:
        int residerID ;

    public :
        int roomNo ;
        string bed ;
        string condition ;
        bool availableStatus;
        int price ;

        Room()
        {
            roomNo = 0;
            bed = "" ;
            condition = "";
            availableStatus = true;
        }

        Room( int initialize )
        {
            roomsUpdate() ;
        }

        int getResiderID()
        {
            return this->residerID;
```

Class - Client

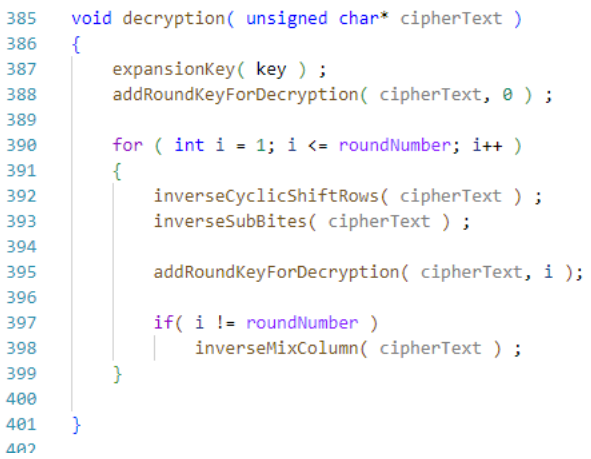
Class - Room

AES ENCRYPTION

The AES encryption performs following operations



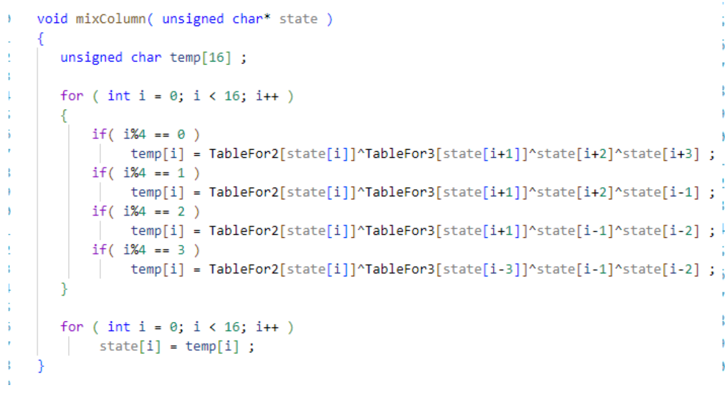
```
void encryption( unsigned char* message )
{

    expansionKey( key ) ;
    addRoundKey( message, 0 ) ;

    for ( int i = 1; i <= roundNumber; i++ )
    {
        subBytes( message ) ;
        cyclicLeftShiftRow( message ) ;

        if( i != roundNumber )
            mixColumn( message ) ;

        addRoundKey( message, i ) ;
    }
}
```



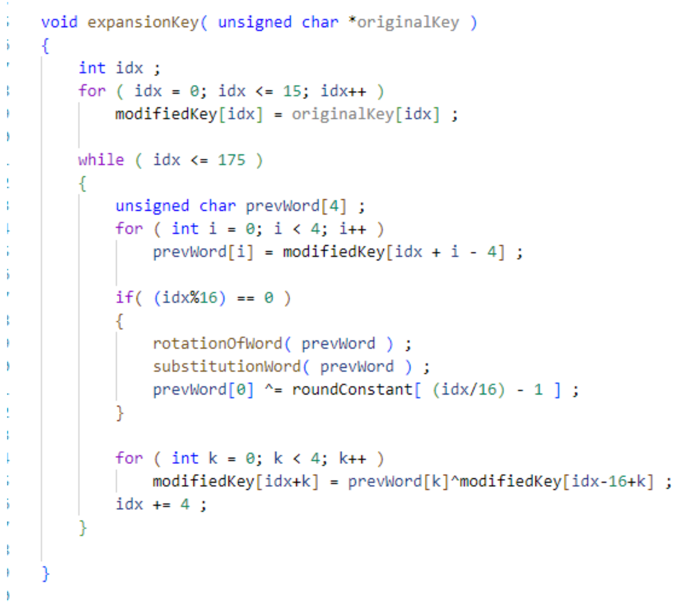
```
void decryption( unsigned char* cipherText )
{
    expansionKey( key ) ;
    addRoundKeyForDecryption( cipherText, 0 ) ;

    for ( int i = 1; i <= roundNumber; i++ )
    {
        inverseCyclicShiftRows( cipherText ) ;
        inverseSubBites( cipherText ) ;

        addRoundKeyForDecryption( cipherText, i );

        if( i != roundNumber )
            inverseMixColumn( cipherText ) ;
    }

}
```

Through these two functions, mainly the works are done. There are the supportive functions. Some are given bellow:

Mix_Column and Expansion_Key function is here :



```
void mixColumn( unsigned char* state )
{
    unsigned char temp[16] ;

    for ( int i = 0; i < 16; i++ )
    {
        if( i%4 == 0 )
            temp[i] = TableFor2[state[i]]^TableFor3[state[i+1]]^state[i+2]^state[i+3] ;
        if( i%4 == 1 )
            temp[i] = TableFor2[state[i]]^TableFor3[state[i+1]]^state[i+2]^state[i-1] ;
        if( i%4 == 2 )
            temp[i] = TableFor2[state[i]]^TableFor3[state[i+1]]^state[i-1]^state[i-2] ;
        if( i%4 == 3 )
            temp[i] = TableFor2[state[i]]^TableFor3[state[i-3]]^state[i-1]^state[i-2] ;
    }

    for ( int i = 0; i < 16; i++ )
        state[i] = temp[i] ;
}
```

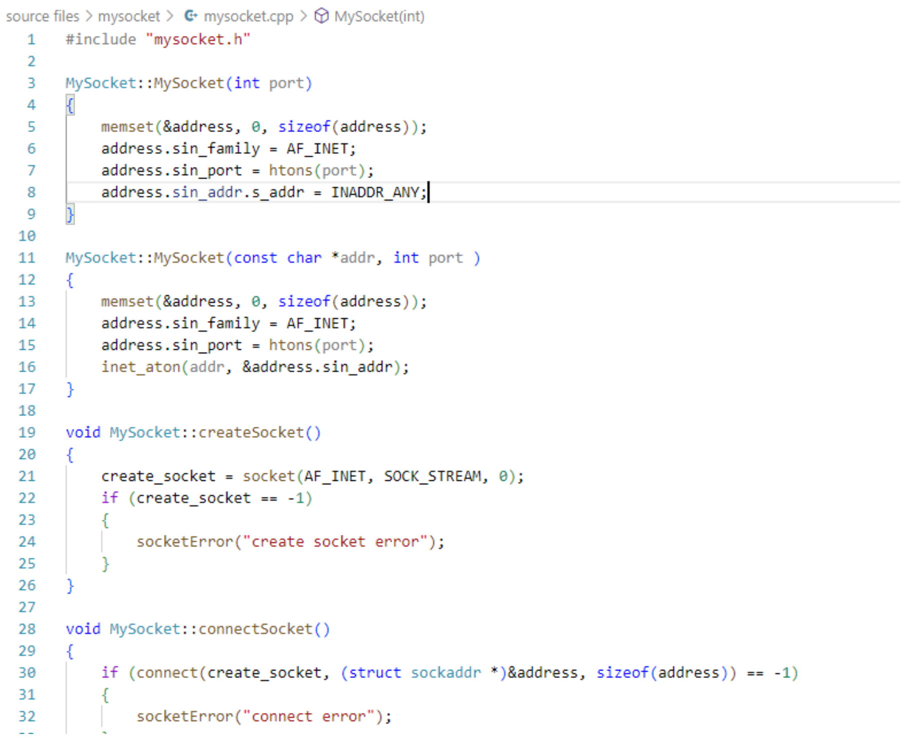


```
void expansionKey( unsigned char *originalKey )
{
    int idx ;
    for ( idx = 0; idx <= 15; idx++ )
        modifiedKey[idx] = originalKey[idx] ;

    while ( idx <= 175 )
    {
        unsigned char prevWord[4] ;
        for ( int i = 0; i < 4; i++ )
            prevWord[i] = modifiedKey[idx + i - 4] ;

        if( (idx%16) == 0 )
        {
            rotationOfWord( prevWord ) ;
            substitutionWord( prevWord ) ;
            prevWord[0] ^= roundConstant[ (idx/16) - 1 ] ;
        }

        for ( int k = 0; k < 4; k++ )
            modifiedKey[idx+k] = prevWord[k]^modifiedKey[idx-16+k] ;
        idx += 4 ;
    }

}
```

These generating of different keys are done by the expansion process, mixing with the column, row shifting etc.

Socket Usage:

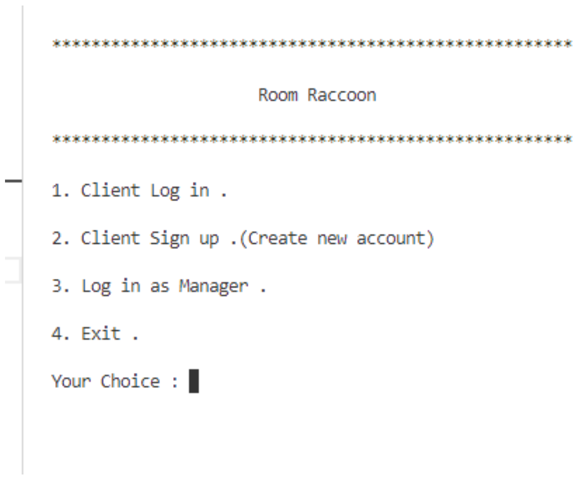
By the IP address and port number new socket is created and connected to the server. The server needs to be active always so that any client can be connected any time.



source files > mysocket > mysocket.cpp > MySocket(int)

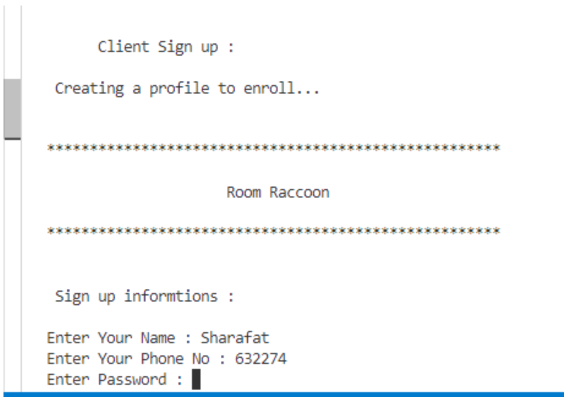
```
#include "mysocket.h"

MySocket::MySocket(int port)
{
    memset(&address, 0, sizeof(address));
    address.sin_family = AF_INET;
    address.sin_port = htons(port);
    address.sin_addr.s_addr = INADDR_ANY;
}

MySocket::MySocket(const char *addr, int port )
{
    memset(&address, 0, sizeof(address));
    address.sin_family = AF_INET;
    address.sin_port = htons(port);
    inet_aton(addr, &address.sin_addr);
}

void MySocket::createSocket()
{
    create_socket = socket(AF_INET, SOCK_STREAM, 0);
    if (create_socket == -1)
    {
        socketError("create socket error");
    }
}

void MySocket::connectSocket()
{
    if (connect(create_socket, (struct sockaddr *)&address, sizeof(address)) == -1)
    {
        socketError("connect error");
```

5. User Interfaces:



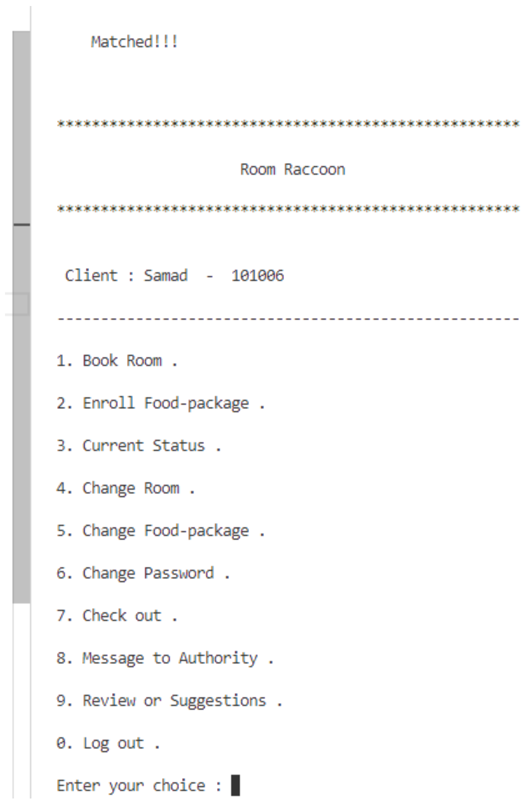
```
*******************************************************

                    Room Raccoon

*******************************************************

1. Client Log in .

2. Client Sign up .(Create new account)

3. Log in as Manager .

4. Exit .

Your Choice :
```

Sign Up Menu:



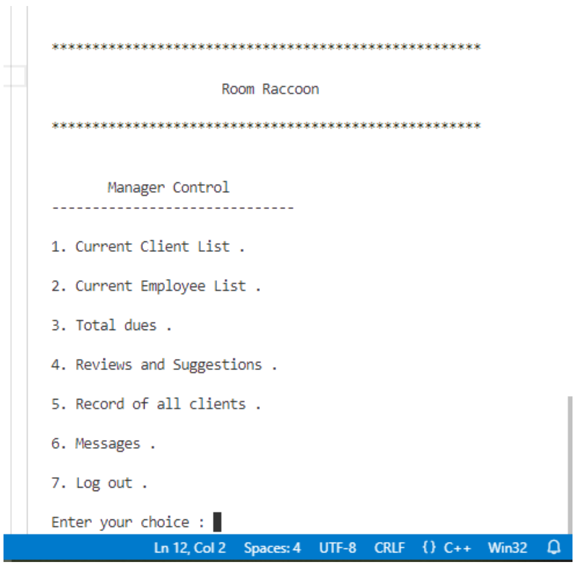
```
     Client Sign up :

 Creating a profile to enroll...

*******************************************************

                    Room Raccoon

*******************************************************


 Sign up informtions :

Enter Your Name : Sharafat
Enter Your Phone No : 632274
Enter Password :
```

Client Interface:



```
   Matched!!!


*******************************************************

                    Room Raccoon

*******************************************************


 Client : Samad  -  101006

-------------------------------------------------------

1. Book Room .

2. Enroll Food-package .

3. Current Status .

4. Change Room .

5. Change Food-package .

6. Change Password .

7. Check out .

8. Message to Authority .

9. Review or Suggestions .

0. Log out .

Enter your choice :
```

Manager Interface:



```
*******************************************************

                    Room Raccoon

*******************************************************


        Manager Control
------------------------------

1. Current Client List .

2. Current Employee List .

3. Total dues .

4. Reviews and Suggestions .

5. Record of all clients .

6. Messages .

7. Log out .

Enter your choice :
```

From this menu, the users can choose their option of interest. The functionality will be done accordingly.

6. Challenges Faced:

While doing this project, I faced several issues and difficulties as it is my first big project. It brought new technical issues as well as code handling skills in the test. The Challenges are as follows:

- AES algorithm was enough complex to implement correctly. Ensuring the proper understanding and accurate implementation of these algorithms posed a challenge for me as they use various sub-algorithms within it, making it larger and tough to implement correctly.

- Managing encryption keys securely is crucial for the security of the tool. Generating, storing, and handling keys in a secure manner was challenging for me as I had to deal with file read/writing in binary mode which I was not used to.

- Integrating different encryption algorithms and socket programing seamlessly in a single tool was tricky for me. I had to make sure that adding another feature does not break or bring a bug to the entire project. So, before implementing anything, I had to brainstorm how I am going to design a generalized form which can handle these algorithms.

- Working with header files in C++ for the first time. So there was enough problem initially.

- Reading and writing of files in C++.

- Establishing socket connection in C++.

- Working with multiple type of source files.

- Working with long lines of code and keeping track of everything.

- Integrating AES into the socket .

- Figuring out how to make the whole project work and explore to find the resources.

7. Conclusion:

This project is my first creation of a larger project. This is not much, but it is enriched with many features. Some are:

- Secured data management
- User-friendly system
- All services in a single system
- Machine calculated, less possibility of error
- Smart service
- Remote control and observation
- Many branch can be handled

To create this project I had to face many difficulties. The things which motivated me to create such a nice project are:

- To make easily understandable and adaptable .
- To confirm public data security .
- System must be optimized .
- Data overload or leakage should be checked .
- System must not be crashed .
- Command lineup and system testing should be ensured .
- File management should be checked several times .

The project taught me to work with socket programming in C++. Before working on the project, I had no knowledge about network programming. I also learned to incorporate security and better transmission facility in my codes. I have tried to apply some OOP principles such as increasing modularity and decreasing the coupling. I learnt file handling and working with header files in C++. Besides all these, I have had the opportunity to be supervised by a Professor in the technology field which gives me the confidence to take on bigger projects in future.

References

1. AES :- https://www.geeksforgeeks.org/advanced-encryption-standard-aes/ , Geeksforgeeks-> AES Encryption and Decryption, 13/10/2023

2. Socket programming Basics:- https://marketsplash.com/tutorials/cpp/cplusplus-scoket/