Rapport Mini-Projet C++

Intitulé du sujet :

Développement d'une chaine d'hotel

Encadré par:
Mr. Mohamed Chiheb BEN CHAABANE

Elaboré par:

- Mlle Yasmine Ameri
- Mlle Emna Belhareth
- Mlle Manar Ferah

Année Universitaire : 2021-2022

TABLE DES MATIÈRES

❖ Introduction :

La gestion hôtelière est une vitalité indispensable dans le déroulement des activités normale d'un hôtel. Notre travail consiste donc à la conception et l'implémentation de gestion de reservation hôtelière qui prendra en compte toutes les contraintes qui peuvent survenir lorsqu'un agent hôtelier établi des réservations. A travers notre projet , il est possibles de vérifier la liste des chambres disponible selon les critères souhaiter par le client ainsi de les réserver afin d'être occuper ultérieurement. Notre travail est présenté par deux chapitres : Le premier sera consacré à la présentation des besoins fonctionnels et non fonctionnels. Dans le second chapitre, nous nous intéressons à l'étude conceptuelle et nous détaillons les différents modèles adoptés ainsi les diagramme de sequence et de cas d'utilisation.

❖ Objectif du projet :

L'objectif du développement de cette application est de digitaliser la gestion d'une chaîne d'hôtel afin de mieux subvenir **aux besoins des clients**

Le processus de développement a été réalisé durant deux phases :

Nous avons récupéré les informations ainsi les différentes tâches du service au sein des hôtels.

La deuxième phase c'est l'implémentation de notre système, ou nous avons utilisé langage C++.

❖ Identification des Besoins:

Les besoins fonctionnels et non fonctionnels de notre application. Cette spécification nous permettra d'éclaircir notre objectif.
Le système comportera différentes fonctionnalités nécessaires pour une meilleure gestion.
L'application peut accomplir les traitements suivants :

- L'utilisateur peut se connecter avec son propre login et mot de passe, l'application est bien sécurisée.
- La gestion des réservations : L'ajout, annulation, modification de la réservation.
- La gestion des entités.

Ces informations seront stockées dans un fichier qui peut être mise à jour au fur et à mesure des besoins.

❖ Diagramme des Cas d'Utilisation :

Le diagramme de cas d'utilisation représente la structure des fonctionnalités nécessaires aux utilisateurs du système. Il est utilisé dans les deux étapes de capture des besoins fonctionnels et techniques. A partir de l'étude préliminaire, nous avons pu dégager le diagramme des cas d'utilisation général suivant :

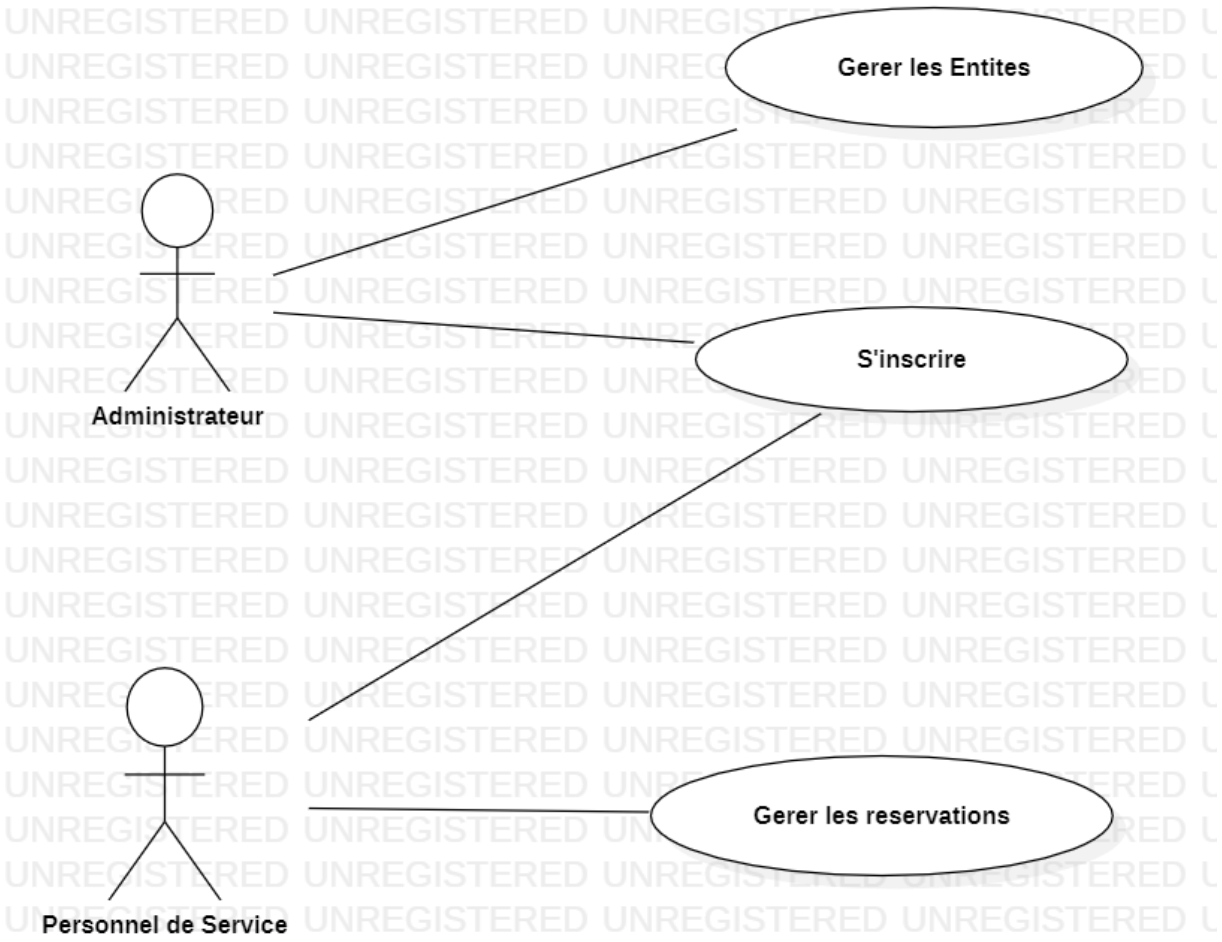


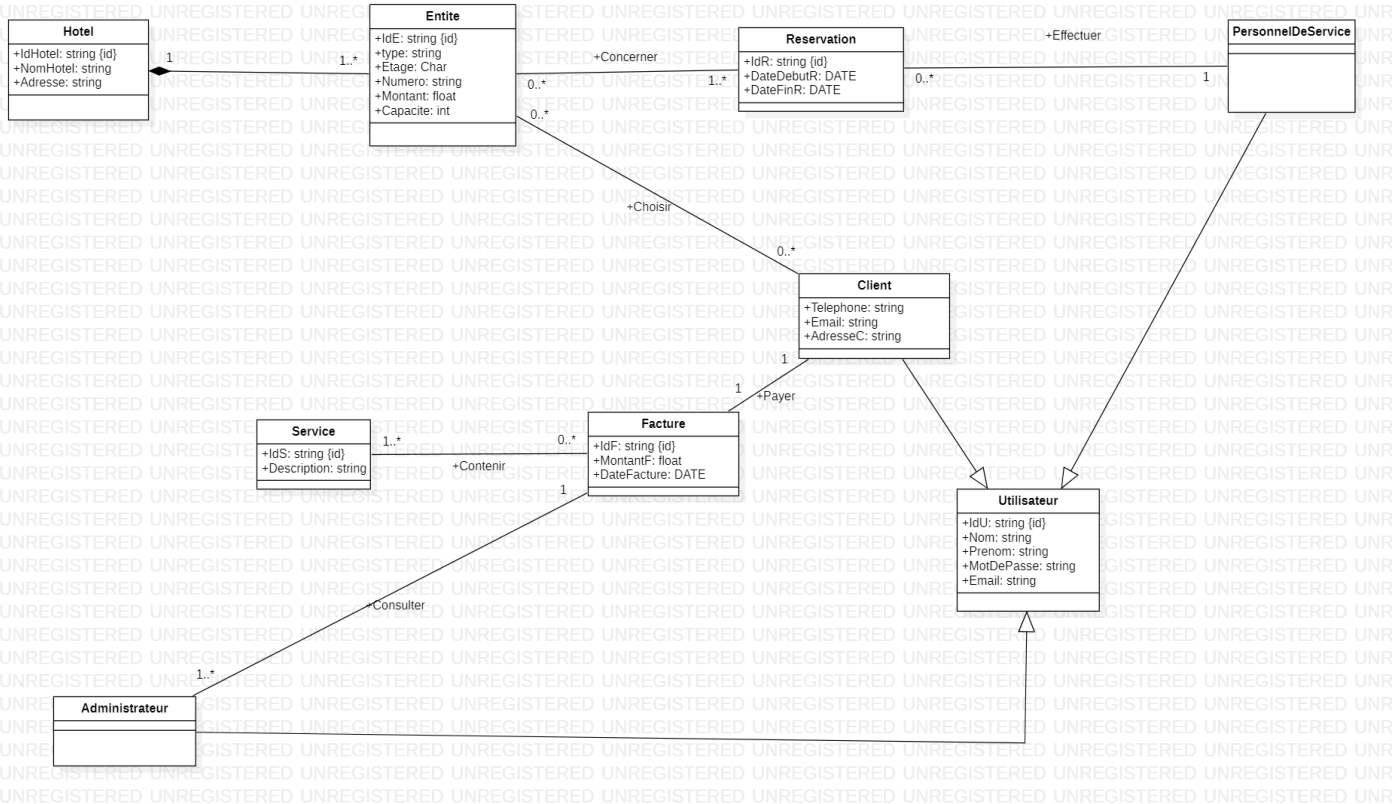
Diagramme de cas d'utilisation

On a 2 acteurs :

- Administrateur : gère les entités mais tout d'abord il faut avoir un compte

- Personnel de Service : gère les réservations, afin d'exploiter ce service, il doit avoir un compte.

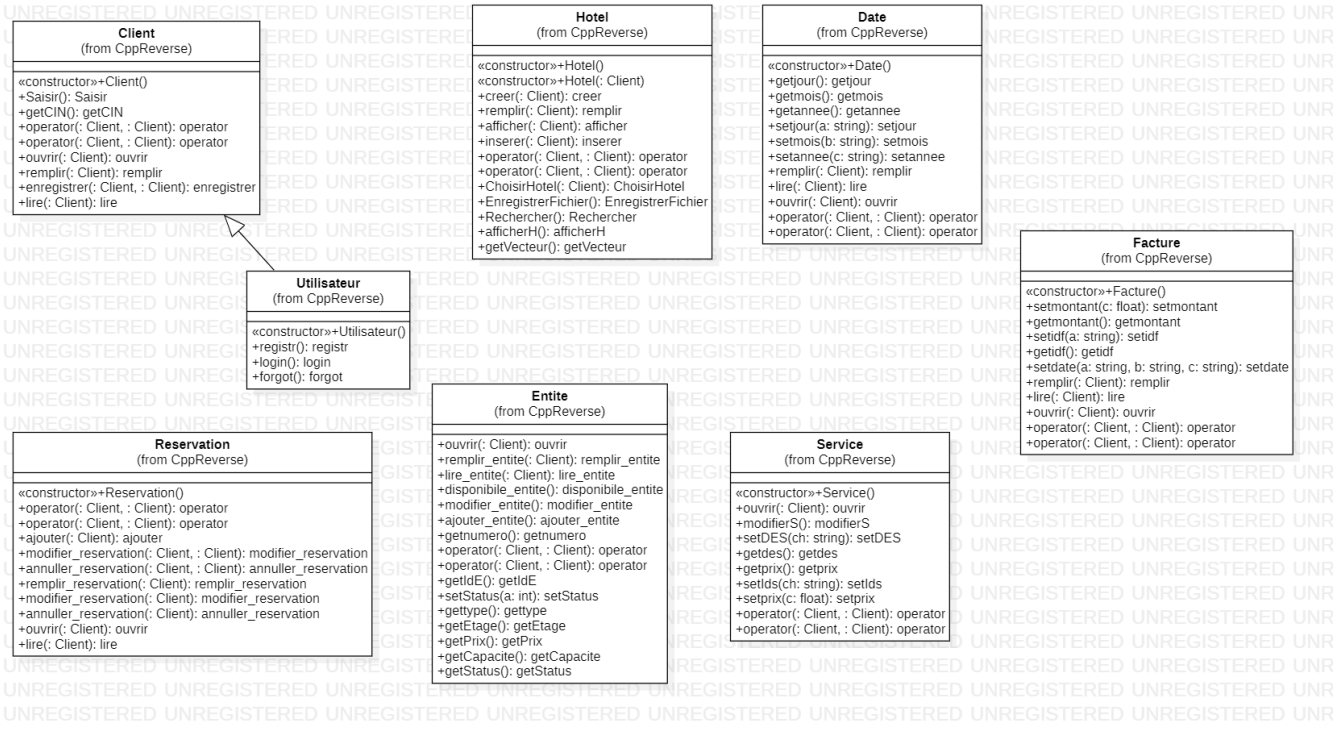
❖ Diagramme de classe participantes :



- On a relation de composition entre class<< Hotel >> et class<< Entite>>
- << Entite >> peut être choisi par plusieurs <<Client>> et est concerné par plusieurs <<Reservation>>
- Seuls les personnels services effectuent la <<Reservation>>
- <<PersonnelDeService >> est un << Utilisateur>>
- <<Client>> est un <<Utilisateur>>
- <<Client>> paie la <<Facture>> qui contient <<Service>>
- <<Facture>> est consulté par <<Administrateur>> qui est un <<Utilisateur>>

❖ Diagramme de classe d'implémentation :

Le diagramme est généré par **StarUML**



❖ Conclusion:

Les chaines hôtelières cherchent à se différencier par la digitalisation afin de mieux répondre les besoins de clients. Cette application essaie de satisfaire cette exigence, Si on continue à la développer, on peut arriver à mettre en place une application qui permet de gagner le temps, réduire les couts et qui permet de fournir des belles expériences pour les clients

❖ Annexes

Client.h

*client.h

```
#ifndef CLIENT_H_INCLUDED
#define CLIENT_H_INCLUDED
#pragma once
#include<iostream>
using namespace std;
#include<string>
#include<iomanip>
#include<fstream>
#include"utilisateur.h"

class Client:public Utilisateur
{
    string Telephone;           //Telephone client
    string AdresseC;            //Adresse client
public:
    Client():Utilisateur(){;}
    void Saisir(); //Saisir informations
    int getCIN(){return CIN;}
    friend ostream& operator<< (ostream & , const Client& ) ;   //Afficher informations
    friend istream& operator>> (istream & ,  Client& )  ;   //Afficher informations
    static void ouvrir(fstream &); //Ouvrir fichier
    void remplir(fstream& ); //Remplir fichier
    static void enregistrer(fstream &, Client &); //Enregistrer dans le fichier
    void lire(fstream& ); //Lecture a partir de fichier
};


#endif // CLIENT_H_INCLUDED

```

Client.cpp

client.cpp

```
#include"client.h"
#include<iostream>
#include<iomanip>
#include<fstream>
void menu();
using namespace std;
void Client::Saisir()       //saisir les informations
{
    cout<<"Entrer cin de client :"<<endl;
    cin>>CIN;
    cout<<"Entrer nom de client :"<<endl;
    cin>>Nom;
    cout<<"Entrer prenom de client:"<<endl;
    cin>>Prenom;
    cout<<"Entrer l address de client : "<<endl;
    cin>>AdresseC;
    test:
    cout<<"Entrer numero de telephone de client: "<<endl;
    cin>>Telephone;
    for(int i=0;i<(Telephone.length());i++) //Validation numero de telephone
     {

                                    //condition sur le numero de telephone : 8 chiffres
       if(!isdigit(Telephone[i]))
        {
            cout<<"Numero de telephone doit etre en chiffre\n"<<endl;
            goto test;
        }

    if(Telephone.length()!=8)       //validation longeur de telephone
    {
      cout<<"Numero de telephone sur 8 chiffres.\n";
      goto test;
     }
     }
     cout<<"Entrer adresse email: "<<endl;
    cin>>Email;
}
```

```
}
void  Client::ouvrir(fstream & f)
{
    f.open("reservation.txt",ios::in|ios::app);
    if(!f.is_open()) exit -1;
}
void Client::enregistrer(fstream & f,  Client & c)
{
    f<<c<<"\n";
}

ostream& operator<< (ostream & out ,const  Client& c)
{
    out<<"CIN :"<<endl;
    out<<c.CIN <<endl;
    out<<"Nom:"<<endl;
    out<<c.Nom <<endl;
    out<<"Prenom"<<endl;
    out<<c.Prenom <<endl;
    out<<"Telephone:"<<endl;
    out<<c.Telephone <<endl;
    out<<"Email:"<<endl;
    out<<c.Email<<endl;
    out<<"Adresse:"<<endl;
    out<<c.AdresseC<<endl;


    return out;
}
istream& operator>> (istream & in ,  Client& c)
{
    cout<<"Entrer CIN:"<<endl;
    in>>c.CIN;
    cout<<"Entrer Nom:"<<endl;
    in>>c.Nom;
    cout<<"Entrer Prenom:"<<endl;
    in>>c.Prenom;
    cout<<"Entrer Telephone:"<<endl;
    in>>c.Telephone;
    cout<<"Entrer Email:"<<endl;
    in>>c.Email;
    cout<<"Entrer Adresse:"<<endl;
    in>>c.AdresseC;



    return in;
}
void Client::remplir(fstream& f)
{

    f<<"CIN:"<<endl;
    f<<CIN<<endl;
   f<<"Nom:"<<endl;
    f<<Nom<<endl;
    f<<"Prenom:"<<endl;
    f<<Prenom<<endl;
    f<<"Telephone:"<<endl;
    f<<Telephone<<endl;
    f<<"Email:"<<endl;
    f<<Email<<endl;
   f<<"Adresse:"<<endl;
    f<<AdresseC<<endl;
```

```
}
void Client::lire(fstream& f)
{

    char ch[101];
      f.getline(ch,100,'\n');
     f>>CIN;
     f.getline(ch,100,'\n');
     f.getline(ch,100,'\n');
     f>>Nom;
      f.getline(ch,100,'\n');
      f.getline(ch,100,'\n');
     f>>Prenom;
      f.getline(ch,100,'\n');
      f.getline(ch,100,'\n');
     f>>Telephone;
      f.getline(ch,100,'\n');
      f.getline(ch,100,'\n');
     f>>Email;
      f.getline(ch,100,'\n');
      f.getline(ch,100,'\n');
     f>>AdresseC;
     f.getline(ch,100,'\n');




}
```

Date.h

date.h

```
#pragma once
#include<iostream>
using namespace std;
#include<string>
#include<iomanip>
#include<istream>
#include<fstream>
#ifndef DATE_H_INCLUDED
#define DATE_H_INCLUDED
/************************Class Date*****************************/
class Date
{
    string jour;
    string mois;
    string annee;
public:
    Date(){}; // Constructeur
    ~Date(){}; // Destructeur
    string getjour(){return jour;} // Retour jour
    string getmois(){return mois;} // Retour mois
    string getannee(){return annee;} // Retour annee
    void setjour(string a){jour=a; } // Set jour
    void setmois(string b){mois=b;} // Set mois
    void setannee(string c){annee=c;} // Set annee
    void remplir(fstream&);// Remplissage de fichier
    void lire(fstream&);// Remplissage de lecture de fichier
    void ouvrir(fstream&); //Ouverture de fichier
     friend ostream& operator<<(ostream&, Date&); // Surcharge
    friend istream& operator>>(istream&, Date&); // Surcharge
};


#endif // DATE_H_INCLUDED

```

Date.cpp

date.cpp

```cpp
#include"date.h"
#include<string>

 ostream& operator<<(ostream& out, Date& d)
 {

      out<<endl<<d.jour<<"/"<<d.mois<<"/"<<d.annee;

      return out;
 }
istream& operator>>(istream& in, Date& d)
{
{
     cout<<"Saisir jour : "<<endl;
     in>>d.jour;
     cout<<"Saisir mois:"<<endl;
     in>>d.mois;
      cout<<"Saisir annee:"<<endl;
     in>>d.annee;
     return in;
}
 void Date::remplir(fstream& f)
     {

       f<<jour<<"/"<<mois<<"/"<<annee<<endl;

     }
void Date::lire(fstream& f)
{

     char ch[101];

   f.getline(ch,100,'/');
   jour=ch;

 f.getline(ch,100,'/');
 mois=ch;

   f.getline(ch,100);
   annee=ch;


}
void  Date::ouvrir(fstream & f)
{
     f.open("reservation.txt",ios::in|ios::out|ios::app);
     if(!f.is_open()) exit -1;
}
```

Entite.h

entite.h

```cpp
#define ENTITE_H_INCLUDED
#include<iostream>
using namespace std;
#include<string>
#include"hotel.h"
#include<iomanip>
#include<istream>
#include<vector>
#include<fstream>
/*********Class Entite**************/
class Entite {
     string IdE; //Identifiant
     string type; //Type d entite : chambre,salle de marriage,...
     int Etage; // Numero d etage
     string Numero; // Numero d entite
     float Prix; // Prix
     int Capacite; // Capacite
     int status; // Status de disponibilite
public:
    static void ouvrir(fstream& ); // Ouverture de fichier
    void remplir_entite (fstream& ); //Remplissage de entite dans un fichier
    void  lire_entite(fstream&); // Lecture à partir de fichier
    void disponibile_entite();// Disponibilte d entite
    void modifier_entite();//Modification de prix ou status
    void ajouter_entite();//Ajout d entite
    string getnumero(){return Numero;} //retourner un numero
    friend ostream& operator<< (ostream& ,const Entite&);//Surcharge
    friend istream& operator>> (istream& , Entite&);//Surcharge
     string getIdE(){return IdE;}//retourner IdE
     void setStatus(int a){status=a;} // Modifier status
    string gettype(){return type;}//Retourner type
    char getEtage(){return Etage;} //Retourner etage
    float getPrix(){return Prix;} //Retourner prix
    int getCapacite(){return Capacite;} //Retourner capacite
    int getStatus(){return status;} // Retourner status
    friend class Hotel; //Class amie
};
#endif // ENTITE_H_INCLUDED
```

Entite.cpp

entite.cpp

```cpp
#include"entite.h"
void  Entite::ouvrir(fstream& f) // Ouverture de fichier
   {
      f.open("reservation.txt",ios::in|ios::app);
    if(!f.is_open()) exit -1;
   }
   void Entite::remplir_entite (fstream& f ) //Remplir l entite dans le fichier
   {


    f<<"identite : "<<endl;
    f<<IdE<<endl;
    f<<"type : "<<endl;
    f<<type<<endl;
    f<<"etage : "<<endl;
    f<<Etage<<endl;
    f<<"Numero : "<<endl;
    f<<Numero<<endl;
    f<<"Prix: "<<endl;
    f<<Prix<<endl;
    f<<"Capacite : "<<endl;
    f<<Capacite<<endl;
    f<<"status : "<<endl;
    f<<status<<endl;


   }
}
 void Entite::lire_entite (fstream& f ) // Lecture d entite à partir de fichier
{

 char ch[101];

f.getline(ch,100);
f.getline(ch,100);
 f>>IdE;
  f.getline(ch,100);
  f.getline(ch,100);
 f>>type;
  f.getline(ch,100);
  f.getline(ch,100);
 f>>Etage;
  f.getline(ch,100);
  f.getline(ch,100);
 f>>Numero;

  f.getline(ch,100);
  f.getline(ch,100);
 f>>Prix;
 f.getline(ch,100);
 f.getline(ch,100);
 f>>Capacite;
  f.getline(ch,100);
  f.getline(ch,100);
 f>>status;
  f.getline(ch,100);
  f.getline(ch,100);



}
```

```
  void Entite::disponibile_entite() // l entite est disponible ou pas
   {
      if (status==1) cout<<"Disponible\n";
      else cout<<"Non Disponible\n";

   }
  void Entite::modifier_entite() // Modification de l entite
   {
     cout<<"Modification de prix: "<<endl;
     cin>>Prix;   //setprix
     cout<<"Modification de status:"<<endl;
     cin>>status;
   }
void Entite::ajouter_entite() //Ajout de l entite
{
     cout<<"Entrer l'identificateur d' entite :\n";
     cin>>IdE;
     cout<<"Entrer le type d'entite: \n";
     cin>>type;
     cout<<"Entrer l' Etage d'entite :\n";
     cin>>Etage;
     cout <<"Entrer le Numero d' entite :\n";
     cin>>Numero;
     cout <<"Entrer le prix d' entite :\n";
     cin>>Prix;
     cout <<"Entrer la capacite d' entite :\n";
     cin>>Capacite;
     cout <<"Entrer le status d' entite :\n";
     cin>>status;

}
 ostream& operator<< (ostream& out   , const Entite& e) // Surcharge
    {
          out<<e.IdE<<endl;
          out<<e.type<<endl;
          out<<e.Etage<<endl;
          out<<e.Numero<<endl;
          out<<e.Prix<<endl;
          out<<e.Capacite<<endl;
          out<<e.status<<endl;
      return out;
    }
 istream& operator>> (istream& in   , Entite& e) //Surcharge
    {
          in>>e.IdE;
          in>>e.type;
          in>>e.Etage;
          in>>e.Numero;
          in>>e.Prix;
          in>>e.Capacite;
          in>>e.status;
      return in;
    }
```

Facture.h

facture.h

```
#ifndef FACTURE_H_INCLUDED
#define FACTURE_H_INCLUDED
#pragma once
#include<iostream>
#include<string>
#include<istream>
#include <ctime>
using namespace std;
#include"date.h"
#include"client.h"
/**********Class Facture*********/
class Facture {

    string IdF; // Identifiant de facture
    float montant; // Montant
    Date DateFacture; // Date de facture

public:
    Facture(){}; // Constructeur
    ~Facture(){}; // Destructeur
   void setmontant(float c){ montant=montant+c;} //Set montant
   float getmontant(){return montant;} // Get montant
  void setidf(string a){IdF=a;} // Set identite de facture
   string getidf(){return IdF;} // get Identite de facture
   void setdate(string a,string b,string c){DateFacture.setjour(a);DateFacture.setmois(b);DateFacture.setannee(c);} // Set date
    void remplir(fstream&);// Remplissage de fichier
    void lire(fstream&); // Lecture de fichier
    void ouvrir(fstream&); //Ouverture de fichier
    friend ostream& operator<<(ostream&, Facture&); // Surcharge
    friend istream& operator>>(istream&, Facture&); // Surcharge

};


#endif // FACTURE_H_INCLUDED
```

Facture.cpp

facture.cpp

```
#include"facture.h"
 ostream& operator<<(ostream& out, Facture& f) //Surcharge
 {

     out<<"Id de facture :  "<<endl<<f.IdF;
     out<<f.DateFacture<<endl;
     out<<"Montant :"<<endl<<f.montant<<endl;
     return out;
 }
istream& operator>>(istream& in, Facture& f) // Surcharge
{
    cout<<"saisir id de facture : "<<endl;
    in>>f.IdF;
    cout<<"saisir Date :"<<endl;
    in>>f.DateFacture;
     cout<<"saisir montant :"<<endl;
    in>>f.montant;
    return in;
}
  void Facture::remplir(fstream& f) // Remplissage de fichier
    {

      f<<"Id de facture : "<<endl<<IdF<<endl;
      f<<"Date de facture : "<<endl;DateFacture.remplir(f);
      f<<"Montant : "<<endl<< montant <<endl;

    }
```

```
void Facture::lire(fstream& f) // fonction permet la lecture de fichier
{

    char ch[101];

   f.getline(ch,100);
    f>>IdF;
    f.getline(ch,100);
    DateFacture.lire(f);
   f.getline(ch,100);
    f>>montant;


}
void  Facture::ouvrir(fstream & f) // Ouverture de fichier
{
    f.open("reservation.txt",ios::in|ios::out|ios::app);
    if(!f.is_open()) exit -1;
}
```

Hotel.h

hotel.h

```
#ifndef HOTEL_H_INCLUDED
#define HOTEL_H_INCLUDED
#include <string>
#include <fstream>
#include <iostream>
#include <istream>
#include<vector>
#include"entite.h"
using namespace std;
/*************Class Hotel*****************************/
class Entite;
class Hotel {

    string IdHotel;//Identifiant hotel
    string NomHotel;//Nom hotel
    string Adresse;//Adresse
    vector <Entite*> v;
public:

    Hotel();//Constructeur
    Hotel(const Hotel&);//Constructeur par recopie
    ~Hotel(void ); //Destructeur
    static void creer(fstream& ); //Creation fichier
    static void  remplir(fstream &); //Remplissage fichier
    static void  afficher(fstream& );//Affichage fichier
    void inserer (Entite*);//Insertion entite
    friend ostream& operator<< (ostream& ,const Hotel&);//Surcharge
    friend istream& operator>> (istream& ,Hotel& );//Surcharge
    int ChoisirHotel(vector<Hotel*> );//Choisir Hotel
    void EnregistrerFichier(); //Enregistrer dans un fichier
    int  Rechercher();//Rechercher
    void afficherH();//Affichage
     vector <Entite*> getVecteur(){return v;}//Retourner vecteur

};

#endif // HOTEL_H_INCLUDED
```

Hotel.cpp

hotel.cpp

```cpp
#include"hotel.h"
#include<vector>
Hotel::Hotel(){}


Hotel::Hotel(const Hotel& h)
{
     Entite*e;
     IdHotel=h.IdHotel;
     NomHotel=h.NomHotel;
    Adresse=h.Adresse;

    for (int i=0; i<h.v.size();i++)
    {
         e=new Entite(*h.v[i]);
         v.push_back(e);
    }
}
Hotel::~Hotel(void )
{
     for(int i=0;i<v.size();i++)
     delete v[i];
     v.clear();
}
void Hotel::creer(fstream& f)
{
      f.open("hotel.txt", ios::in | ios::out | ios::app );
       if (! f.is_open()) exit(-1);
}
```

hotel.cpp

```cpp
void Hotel::remplir(fstream& f)
{
  Hotel h;
  cout<<"Saisir Id hotel: "<<endl;
   cin>>h.IdHotel;
   f<<h.IdHotel<<endl;
    cout<<"Saisir Nom d'hotel: "<<endl;
   cin>>h.NomHotel;
   f<<h.NomHotel<<endl;
   cout<<"Saisir adresse  d'hotel: "<<endl;
   cin>>h.Adresse;
   f<<h.Adresse<<endl;
}
void Hotel:: afficher(fstream& f) // permet d afficher l hotel
   {
       cout<<"Affichage du fichier "<<endl;
        f.seekg(0);
        char ch[100];
        while  (1)
        {
         f.getline (ch,100);
         if (f.eof()) break;
         cout <<ch<<endl;
        }
   }

int Hotel::ChoisirHotel (vector<Hotel*> tab) // permet de choisir l'hotel
{
    cout<<"Saisir l Id de l hotel"<<endl;
    cin>>IdHotel;
    for (int i=0;i<tab.size();i++)
        if (tab[i]->IdHotel==IdHotel)
        return i;
}
```

*hotel.cpp

```
}
int Hotel:: Rechercher()              //fonction qui permet de retourner  l'indice d'un entite □ rechercher
        {
            int p=-1;
            string x;
            cout<<"Saisir le numero d entite □ rechercher"<<endl;
            cin>>x;
            cout<<"-------------------------------------------------------"<<endl;
            for(int i=0;i<v.size();i++)
            {
                cout<<"-55555-------------------------------------------------------"<<endl;
                if(x==v[i]->getnumero())
                {
                    p=i;
                    cout<<"valeur "<<i<<endl;
                    break;
                }
            }
            cout<<"-------------------------------------------------------"<<endl;
            cout<<"valeur2 "<<p<<endl;
            return p;
        }

void Hotel:: inserer ( Entite* e)// Permet d inserer entite
{
  v.push_back(e);
}

ostream & operator<<(ostream &out,const Hotel& h ) // Surcharge
{

     out<<setw(10)<<h.IdHotel<<" "<<setw(10)<<h.NomHotel<<" "<<setw(10)<<h.Adresse<<"\n";
     for (int i=0; i<h.v.size(); i++)
        out<<setw(10) <<*h.v[i]<<endl;
     return out ;
}
istream& operator>>(istream& in  ,Hotel& h) //Surcharge
  {
       in>>h.IdHotel;
       in>>h.NomHotel;
       in>>h.Adresse;
       for (int i=0; i<h.v.size(); i++)
          in>>*h.v[i];
     return in ;
  }
void Hotel::afficherH() //Afficher les informations Hotel+Entites
{
    cout<<"IdHotel :"<<IdHotel<<endl;
    cout<<"Nom Hotel:"<<NomHotel<<endl;
    cout<<"Adresse:"<<Adresse<<endl;
    for(int i=0;i<v.size();i++)
        cout<<*v[i]<<" ";

}
```

Service.h

service.h

```
#ifndef SERVICE_H_INCLUDED
#define SERVICE_H_INCLUDED
#pragma once
#include<iostream>
using namespace std;
#include<string>
#include<istream>
#include<iomanip>
#include<fstream>
#include<vector>
class Service {


     string IdS;//Identifiant
     string Description;//Description
     float prix;//Prix
    public:
         Service(){};//Constructeur
         ~Service(){};//Destructeur
         void ouvrir(fstream&);//Ouverture fichier
         void modifierS();//Modification service
        void setDES(string ch){Description=ch;}//Modification description
        string getdes(){return Description;}//Retourner description
        float getprix(){return prix;}//Retourner prix
         void setIds(string ch){IdS=ch;}//Modification  Identifiant
         void setprix(float c){prix=c;}//Modification de prix
         friend ostream& operator<<(ostream&, Service&);//Surcharge
    friend istream& operator>>(istream&, Service&);//Surcharge


};


#endif // SERVICE_H_INCLUDED
```

service.cpp

```cpp
#include "service.h"


ostream& operator<<(ostream& out, Service& s) //Surcharge
{
    out<<"id de service:  "<<s.IdS<<endl;
    out<<"Description de service:  "<<s.Description<<endl;
    out<<"prix de service:  "<<s.prix<<endl;

    return out;
}

istream& operator>>(istream& in, Service& s) //Surcharge
{
    cout<<"saisir id de service "<<endl;
    in>>s.IdS;
    cout<<"saisir description de service "<<endl;
    in>>s.Description;
    cout<<"saisir prix de service"<<endl;
    in>>s.prix;
    return in;
}
void Service::modifierS() //Modification
{
    cout<<"saisir id de service "<<endl;
    cin>>IdS;
    cout<<"saisir description de servide "<<endl;
    cin>>Description;
    cout<<"saisir prix de service"<<endl;
    cin>>prix;
}
```

Reservation.h

reservation.h

```cpp
#include<vector>
#include <ctime>
#include"date.h"
#include"entite.h"
#include"client.h"*/
/************class Reservation*****************/

class Reservation
{
    Date DateDebutR;//Date de debut de reservation
    Date DateFinR; // Date fin de reservation
    int nbjours;// Nombre de jour
    Facture fac;// facture
    Client cl;//Client
    Entite e;//Entite
    vector <Service>t;

public:
    Reservation() {};
    ~Reservation() {};

    friend ostream& operator<<(ostream&, Reservation&); //Surcharge
    friend istream& operator>>(istream&, Reservation& ); //Surcharge
    void ajouter(Entite ); //Ajouter Entite
    void remplir_reservation(fstream&  );//Remplissage dans le fichier
    void modifier_reservation(Client);//Modification
    void annuller_reservation(Client );//Annumlation
    void ouvrir(fstream&);//Ouverture
    void lire(fstream&);//Lecture



};


#endif // _RESERVATION_H
```

Reservation.cpp

*Reservation.cpp

```cpp
#include "reservation.h"
ostream& operator<<(ostream& out, Reservation& r) // surcharge
{

    out<<"Client  : "<<endl<<r.cl;
    out<<"Entite : "<<endl<<r.e;
    out<<"Nombre de jour : "<<r.nbjours<<endl;
    out<<"Date de d'arrivee :"<<r.DateDebutR<<endl;
    out<<"Date de depart :"<<r.DateFinR<<endl;

    out<<"Les services :    "<<endl;
    for (int i=0;i<r.t.size();i++)

    out<<r.t[i].getdes()<<endl;
    out<<"Facture :"<<endl<<r.fac;

    return out;
}
void Reservation::ajouter(Entite et) // fonction qui permet d ajouter une reservation
{

    Service tab[7];

        Service  massage , spa , gym , casino , salonDeBeaute ,sauna;
        massage.setDES("Massage");
        massage.setIds("1");
        massage.setprix(55);
        tab[0]=massage;
        gym.setDES("Gym");
        gym.setIds("2");
        gym.setprix(6);
        tab[1]=gym;
        casino.setDES("Casino");
        casino.setIds("3");
       casino.setprix(77);
       tab[2]=casino;
        sauna.setDES("Sauna");
        sauna.setIds("4");
        sauna.setprix(88);
        tab[3]=sauna;
        spa.setDES("Spa");
        spa.setIds("5");
        spa.setprix(88);
        tab[4]=spa;
        salonDeBeaute.setDES("SalonDeBeaute");
        salonDeBeaute.setIds("6");
        salonDeBeaute.setprix(99);
        tab[5]=salonDeBeaute;
```

```
49      cout<<"Saisir les donnees du client : "<<endl;
50      cin>>cl;
51      e=et;
52      cout<<"Saisir Date de d arrive :      "<<endl;
53      cin>>DateDebutR;
54      cout<<"Saisir Date de depart :        "<<endl;
55      cin>>DateFinR;
56      cout<<"Saisir nombre de jour :     "<<endl;
57      cin>>nbjours;
58      cout<<"Choisir les services :    "<<endl;
59     int nbrservice=0;
60     for(int i=1;i<7;i++)
61     {cout<<i<<":"<<tab[i-1].getdes()<<endl;}
62      cout<<6<<":"<<"Quitter"<<endl;
63
64     while(1)
65     {
66         int a;
67         cin>>a;
68         if(a==7) break;
69         nbrservice++;
70         t.push_back(tab[a]);
71          fac.setmontant(tab[a].getprix());
72     }
73     cout<<"Saisir date d'aujourdhui"<<endl;
74     Date d;
75     cin>>d;
76     fac.setdate(d.getjour(),d.getmois(),d.getannee());
77     cout<<"Saisir id facture : "<<endl;
78     string ch;
79     cin>>ch;
80     fac.setidf(ch);
81     cout<<"Montant a payer :"<<fac.getmontant()<<endl;
82     e.setStatus(1);
83
84 }
```

*Reservation.cpp

```
88  void  Reservation::ouvrir(fstream & f) // Ouverture de fichier
89  {
90      f.open("reservation.txt",ios::in|ios::app);
91      if(!f.is_open()) exit -1;
92  }
93  void Reservation::remplir_reservation(fstream& f) // remplissage dans le fichier
94  {
95       Reservation::ouvrir( f);
96       f<<"-----------------------------------------------------------------------------------------------"<<endl;
97      string ch;
98      f<<"Client : "<<endl;
99      cl.remplir(f);
100     f<<"Entite : "<<endl;
101     e.remplir_entite(f);
102     f<<"Nombre de jour: "<<endl<<nbjours<<endl;
103     f<<"Date d'arrivee : "<<endl<<DateDebutR<<endl;
104     f<<"Date de depart : "<<endl<<DateFinR<<endl;
105     f<<"services : " <<endl;
106     fac.setmontant(nbjours*e.getPrix());
107     for(int i=0;i<t.size();i++) {ch=t[i].getdes()/*service*/;f<<ch<<endl;}
108     f<<"Facture :"<<endl;
109      fac.remplir(f);
110      f.close();
111
112 }
113
```

*Reservation.cpp

```
114   void Reservation::lire(fstream& f) // Lecture à partir de fichier
115   {
116      Reservation::ouvrir( f);
117
118     char ch[101];
119     f.getline(ch,100);
120     f.getline(ch,100);
121      cl.lire(f);
122      e.lire_entite(f);
123      f>>nbjours;
124      f.getline(ch,100);
125      f.getline(ch,100);
126      f.getline(ch,100);
127      DateDebutR.lire(f);
128     f.getline(ch,100);
129     f.getline(ch,100);
130      DateFinR.lire(f);
131     for(int i=0;i<t.size(); i++) {
132               f.getline(ch,100);}
133      f.getline(ch,100);
134      f.getline(ch,100);
135
136      fac.lire( f);
137
138      f.close();
139   }
140
141   void Reservation::modifier_reservation(Client clt)
142   {
143       vector<Reservation*> vec;
144       fstream f;
145       Reservation::ouvrir(f);
146       while(1)
147       {
148
149           lire(f);
150          if(!f.eof()) break;
151           vec.push_back(this);
152       }
153       cout<<*this;
154       cout<<vec.size()<<endl;
155       cout<<*vec[0];
156       for(int i=0 ;i<vec.size();i++)
157         if(clt.getCIN()==(vec[i]->cl).getCIN()) {cout<<i;vec[i]->ajouter(vec[i]->e);};
158       for(int i=0 ;i<vec.size();i++) vec[i]->remplir_reservation(f);
159
160   }
161    void Reservation::annuller_reservation(Client clt)
162    {
163        vector<Reservation> vec;
164        fstream f;
165       Reservation::ouvrir(f);
166       while(1)
167       {
168           Reservation::lire(f);
169           if(!f.eof()) break;
170
171           vec.push_back(*this);
172
173       }
174       for(int i=0 ;i<vec.size();i++)
175         if(clt.getCIN()==(vec[i].cl).getCIN()) vec.erase(vec.begin(),vec.begin()+i);
176       for(int i=0 ;i<vec.size();i++) vec[i].remplir_reservation(f);
177    }
178
```

Utilisateur.h

utilisateur.h

```
#ifndef UTILISATEUR_H_INCLUDED
#define UTILISATEUR_H_INCLUDED
#pragma once
#include<iostream>
using namespace std;
#include<string>
#include<iomanip>
#include<istream>
#include<fstream>
void menu();
/*******************Class Utilisateur**********************/
class Utilisateur
{
    protected:
    int CIN; //Identite utlisateur
    string Nom; // Nom
    string Prenom;//Prenom
    string Email;//Email
public:
    Utilisateur(){};//Constructeur
    void registr();//Inscription
    void login(); //Login
    void forgot(); //Mot de passe oublié
   ~Utilisateur(){}//Destructeur
};


#endif // UTILISATEUR_H_INCLUDED

```

Utilisateur.cpp

*utilisateur.cpp

```
#include"utilisateur.h"
#include"reservation.h"
#include"hotel.h"
#include <vector>

void menu(); //Menu
void EnregistrerFichier();
void gererEntite(); //Gerer les entites
void GererReservation(); //Gerer les reservations
void Entete() // En tete
{
    cout<<"\n\n\n";
    cout<<"\t\t**********************************        MENU        ********************************\n\n";
}
void menu()
{
    int choice;
    Utilisateur u;
    string ch ;
    cout<<endl;
    cout<<"\n\t\t*****************************************************************************************************\n\n";
    cout<<"\n\t\t\t                              Welcome login page                                   \n\n";
    cout<<"\n\t\t*****************************************************************************************************\n\n";
    cin.get();
    system("cls");

    Entete();
    cout<<"\n\n\t\t\t\t\t\t\t1.Login"<<endl;
    cout<<"\t\t\t\t\t\t\t2.Inscription"<<endl;
    cout<<"\t\t\t\t\t\t\t3.Mot de passe oublie"<<endl;
    cout<<"\t\t\t\t\t\t\t4.Exit"<<endl;
    cout<<"Entrer votre choix :";
    cin>>choice;
    cout<<endl;
```

```
35
36          switch(choice)
37          {
38
39          case 1:
40               u.login();
41               break;
42          case 2:
43               u.registr();
44               break;
45          case 3:
46               u.forgot();
47               break;
48          case 4:
49               system("cls");
50               exit( EXIT_SUCCESS );
51               break;
52
53          default:
54               system("cls");
55               cout<<"Erreur! veuillez reessayer de nouveau"<<endl;
56               menu();
57
58
59          }
60
61     }
62     int menu2() // Menu 2
63     {
64          int choix2;
65          string ch;
66     again:
67          system("cls");
68          cout<<"\n";
69          cout<<"\n\n\t\t\t\t\t\t1.Administrateur"<<endl;
70          cout<<"\n\t\t\t\t\t\t2.Personnel de service"<<endl;
71          cout<<"\n\t\t\t\t\t\t3.Exit"<<endl;
72          cout<<"Entrer votre choix :";
73          cin>>choix2;
74          cout<<endl;
75
76          return choix2;
77     }
78     void Utilisateur::login() // Login
79     {
80          int count=0;
81          int choix2;
82          string ch;
83          system("cls");
84          Entete();
85          choix2=menu2();
86          switch(choix2)
87          {
88          case 1:
89               ch="database.txt" ;
90               break;
91          case 2:
92               ch="text.txt";
93               break;
94          case 3:
95               menu();
96               break;
97
```

```
    }
    string user,pass,u,p;
    system("cls");
    cout<<"Remplir les details suivants"<<endl;
    cout<<"Login :";
    cin>>user;
    cout<<"Mot de passe :";
    cin>>pass;
    ifstream input(ch);
    while(input>>u>>p)
    {
        if(u==user && p==pass)

        {
            count=1;
            system("cls");
        }
    }
    input.close();
    if(count==1)
    {
        cout<<"\n\t\t*****************************************************************************************************\n\n";
        cout<<"\n\t\t\t                              Bonjour\t"<< user<<"                                \n\n";
        cout<<"\n\t\t\t*****************************************************************************************************\n\n";
        cin.get();
        switch(choix2)
        {
        case 1:
            gererEntite();
            break;
        case 2:
            GererReservation();
            break;
        }


    }
    else
    {
        cout<<"LOGIN ERROR ,Veuillez verifier votre Login ou mot de passe";
    }
}
void Utilisateur::registr() // Inscription
{

    system("cls");
    string reguser,regpass,ru,rp,ch;


    int choix2;
    choix2=menu2();
    switch(choix2)
    {
    case 1:
        ch="database.txt" ;
        break;
    case 2:
        ch="text.txt";
        break;
    case 3:
        menu();
        break;


    }
    system("cls");
    cout<<"Entrer Login:";
    cin>>reguser;
    cout<<"Entrer Mot de passe :";
    cin>>regpass;

    ofstream reg(ch,ios::app);
    reg<<reguser<<' '<<regpass<<endl;
    system("cls");
```

*utilisateur.cpp

```
173         cout<<"Registration Success!!";
174         menu();
175
176
177
178     }
179
180     void Utilisateur::forgot() // En cas de mot de passe ou bien login oublie
181     {
182         int ch;
183         string ch2;
184         system("cls");
185         Entete();
186         ch2=menu2();
187         system("cls");
188         cout<<"oublie ? \n";
189         cout<<"1.Rechercher votre id par login"<<endl;
190         cout<<"2.Rechercher votre id par mot de passe"<<endl;
191         cout<<"3.Menu principal"<<endl;
192         cout<<"Entrer votre choix :";
193         cin>>ch;
194         switch(ch)
195         {
196         case 1:
197         {
198             int count=0;
199             string searchuser,su,sp;
200             cout<<"Entrer votre login :";
201             cin>>searchuser;
202
203             ifstream searchu(ch2);
204             while(searchu>>su>>sp)
205             {
206                 if(su==searchuser)
207                 {
208                     count=1;
209                 }
210             }
211             searchu.close();
212             if(count==1)
213             {
214                 cout<<"Hurray!, Compte trouve\n";
215                 cout<<"Votre mot de passe est "<<sp;
216                 cin.get();
217                 cin.get();
218                 system("cls");
219
220             }
221             else
222             {
223                 cout<<"Desole, Votre login n existe pas dans la base de donnees\n";
224                 cin.get();
225
226             }
227             break;
228         }
229         case 2:
230         {
231             int count=0;
232             string searchpass,su2,sp2;
233             cout<<"Entrer votre Mot de passe:";
234             cin>>searchpass;
235
236             ifstream searchp(ch2);
237             while(searchp>>su2>>sp2)
238             {
239                 if(sp2==searchpass)
```

```
240             {
241                 count=1;
242             }
243         }
244         searchp.close();
245         if(count==1)
246         {
247             cout<<"Votre mot de pass trouvee \n";
248             cout<<"Votre login esr : "<<su2;
249             cin.get();
250             cin.get();
251             system("cls");
252             menu();
253 
254         }
255         else
256         {
257             cout<<"Desole, votre mot de passe n'existe pas dans la base de donnee \n";
258             cin.get();
259             cin.get();
260 
261         }
262 
263         break;
264     }
265 
266     case 3:
267     {
268         cin.get();
269 
270     }
271     default:
272         cout<<"Veuillez ressayez de nouveau"<<endl;
273         forgot();
274     }
275 }
276 vector<Hotel*> copier_fichier_dans_var(vector<Hotel*>tab) //permet de copier contenu de fichier dans un variable
277 {
278     string ch;
279     fstream f("hotel.txt", ios::in|ios::out|ios::app);
280     if (!f) cout<<"Impossible de creer le fichier "<<endl;
281     while (1)
282     {
283         Hotel*h=new Hotel;
284         f>>*h;
285 //cout<<*h;
286         if(f.eof()) break;
287         tab.push_back(h);
288     }
289     f.close();
290 
291     return tab;
292 }
293 vector<Reservation*> copier_fichier_dans_var(vector<Reservation*>tab)// copier fichier dans variable
294 {
295     string ch;
296     int i=0;
297     fstream f("reservation.txt", ios::in|ios::out|ios::app);
298     if (!f) cout<<"Impossible de creer le fichier "<<endl;
299     while (1)
300     {
301         Reservation*r=new Reservation;
302         r->lire(f);
303         cout<<"-------------------------------------------------"<<endl;
304         cout<<*r;
305         cout<<"-------------------------------------------------"<<endl;
306         if(f.eof()) break;
307         tab.push_back(r);
308         cout<<"-------------------------------------------------"<<endl;
309 
310     }
311     f.close();
312     return tab;
313 }
```

```
313 }
314 void gererEntite() // gerer les entites
315 {
316
317     int x, i,choix;
318     vector<Hotel*>tab;
319     Entite e;
320     Hotel h,h1;
321     fstream f;
322     h.creer(f);
323     h.remplir(f);
324     f.close();
325     h1.creer(f);
326     f>>h1;
327     cout<<h1;
328     tab=copier_fichier_dans_var(tab);
329     //system("cls");
330
331     i=h.ChoisirHotel(tab);
332 flag:
333     cout<<"---------------------------------------"<<endl;
334     cout<<"1.Ajouter Entite"<<endl;
335     cout<<"2.Modifier Entite"<<endl;
336     cout<<"3.Disponibilite Entite"<<endl;
337     cout<<"4.Afficher les donnees de l hotel"<<endl;
338     cout<<"5.Quitter"<<endl;
339     cout<<"---------------------------------------"<<endl;
340     cin>>choix;
341     switch (choix)
342     {
343     case 1:
344         e.ajouter_entite();
345         cout<<*tab[i];
346         tab[i]->inserer(&e);
347         goto flag;
348         break;
349     case 2:
350         x=(*tab[i]).Rechercher();
351         tab[i]->getVecteur()[x]->modifier_entite();
352         goto flag;
353         break;
354     case 3:
355         e.disponibile_entite();
356         goto flag;
357         break;
358     case 4:
359         tab[i]->afficherH();
360         goto flag;
361         break;
362     case 5:
363         tab[i]->EnregistrerFichier();
364         menu();
365         break;
366     }
367
368 }
369 void Hotel:: EnregistrerFichier()
370 {
371     fstream f("hotel.txt",ios::in|ios::out|ios::app);
372     if (!f)
373     {
374         cout<<"\n Erreur fichier hotel";
375     }
376     for(int i=0; i<v.size(); i++)
377     {
378         f << *v[i];
379         f.close();
380     }
381 }
```

```
382  void GererReservation()
383  {
384      fstream f;
385      int x, i,choix;
386      Hotel h;
387      Entite e;
388      vector<Hotel*>tab;
389      Client cl;
390      tab=copier_fichier_dans_var(tab);
391
392
393      system("cls");
394      i=h.ChoisirHotel(tab);
395  flag:
396      system("cls");
397      cout<<"1.Ajouter reservation"<<endl;
398      cout<<"2.Modifier reservation"<<endl;
399      cout<<"3.annuler reservation"<<endl;
400      cout<<"4.afficher reservation"<<endl;
401      cout<<"5.quitter"<<endl;
```

*utilisateur.cpp

```
402      cin>>choix;
403      switch (choix)
404      {
405      case 1:
406      {
407          Entite e;
408          Reservation r;
409          e.ajouter_entite();
410          r.ajouter(e);
411          r.remplir_reservation(f);
412          goto flag;
413          break;
414      }
415      case 2:
416      {
417          Reservation r;
418          cl.Saisir();
419          r.modifier_reservation(cl);
420          goto flag;
421      }
422      break;
423      case 3:
424      {
425          Reservation r ;
426          cl.Saisir();
427          r.annuller_reservation(cl);
428          goto flag;
429
430          break;
431      }
432
433      case 4:
434          menu();
435          break;
436      }
437  }
```

Main.cpp

*main.cpp

```
1   #include<iostream>
2   #include<string.h>
3   #include<istream>
4   #include <ctime>
5   using namespace std;
6   #include"reservation.h"
7   #include"utilisateur.h"
8   #include"facture.h"
9   #include"date.h"
10  #include"Service.h"
11  #include"client.h"
12  #include"entite.h"
13
14  int main()
15  {
16      fstream f;
17      menu();
18
19      return 0;
20  }
21
```