Description

The goal of this project is to create a program in C++ language that represents the functionalities of a « hotel reservation » application. This project consists in developing a tool to manage the reservations of a hotel's rooms. The rooms and their characteristics are stored in a « **room.txt »** file (Figure below).

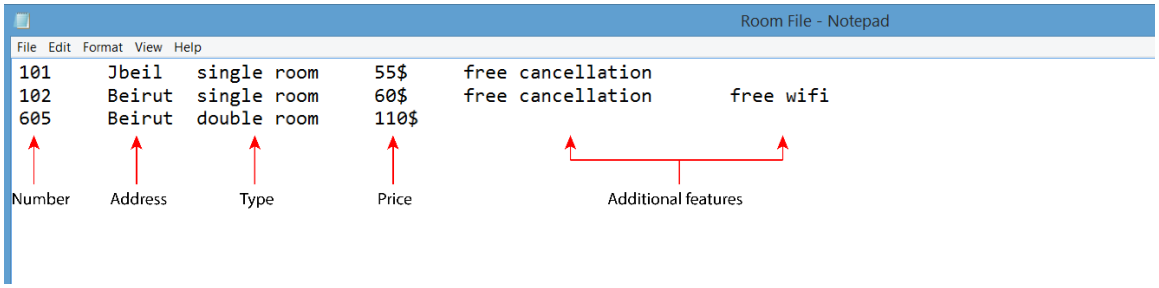

Figure 1 – room file

The application offers two services to the user: confirmation or cancellation of a reservation. After each confirmation, the characteristics of the reserved room must be stored in a structure called **« room »** while adding the data of the latter in the « **reservation.txt** » file (Figure 2).

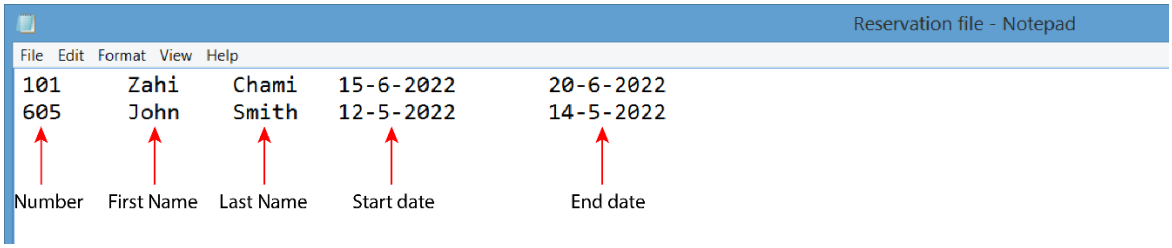

Figure 2 – reservation file

In case of cancellation, the characteristics of the room must be eliminated from the « **reservation.txt** » file and the « **room** » structure. At the end of the program, the reserved rooms must be displayed as well as the total to be paid (price with tax which is equal to 11%). An invoice is generated and saved in a PDF file containing the user's name along with the reserved rooms, sorted by rooms' price in ascending order.

In addition, the user information exists in a « **client.txt** » file (figure below) which contains: first name, last name, password (hashed *), e-mail address, and telephone number.

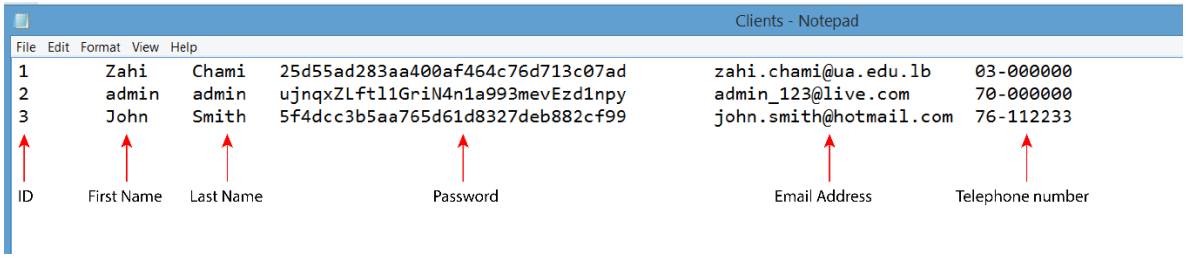

Figure 3 – client file

***: a hash function is a function that will calculate a unique signature from the data provided; in our case, the provided data represents the passwords. Some hash functions: MD5, SHA1, SHA2, etc.**

The program workflow

The program must, at the beginning, ask the user if he has an account or not. If he is a new user, then he must create an account (step 1 below) and store the data in the « **client.txt** » file (Figure 3), then continue the following steps. Otherwise, he must be authenticated by entering his email address and the password before continuing to step 2. The user data must also be stored in the « **Client** » structure in both cases.

1- Ask the user to enter the following information through the console:
- Last name, first name, and password (its length must be at least 8, while containing numbers, letters, and special characters)
- Email address (check and verify the address format)
- Phone number (verify the number format)

P.S: The identifier for each user must be unique and automatically generated by the program.

2- After completing the authentication process, the user could be either an administrator or a client.
- **a.** In the first case, he will then be responsible for the following tasks:
  - **i.** Add rooms to the « **room.txt** » file by entering the data through the console. You must first check if the room exists in the file to avoid data redundancy.
  - **ii.** Delete a room from the « **room.txt »** file through the console.
  - **iii.** Modify the data of a room that exists in the « **room.txt** » file.
- **b.** If the user is a client, he can then choose one of two options:
  - i. Reserve a room by entering the reservation date (starting date and ending date) through the console. The program should display the rooms that are available on this date. After choosing a room, the information must be stored in the « **room** » structure that exists in the « **client** » structure as well as in the « **reservation.txt »** file (Figure 2).
  - **ii.** Cancel a reservation through the console. The program must display the rooms reserved by this client to be able to choose the reservation that will be cancelled.

3- Calculate the total price to be paid and store the result as well as the user information in a PDF file.

Development

In this project, it is necessary to create:

- A **client** structure that contains: int ID, string firstName, string lastName, string password, string address, string tel, **room** *r.
- A **room** structure that contains: int num, string address, string type, double price, string *options (this field indicates if the room has additional features such as: free cancellation, free wifi, etc.), **date** start_date, **date** end_date.
- A **date** structure: int day, int month, int year.
- Files to read the data and write the result.
- Use functions, procedures, dynamic arrays, and pointers as much as possible

Bonus question

You can work on this project using **csv** files as a database (where the data are stored) instead of **txt** files.

Guidelines and information

1- The project must be worked individually or a group of two students.
2- The defense date will take place in the last week.
3- Beware of plagiarism!!!

Good Luck !!!