RMIT University

Software Engineering Design – Advanced Programming Techniques
Semester 2022A

FINAL GROUP ASSIGNMENT (30%)

OBJECTIVES

This assignment provides a chance for students to practice and apply all concepts learned so far. It is also an opportunity to get familiar with analysis, design and development of a software application with a practical project idea to get ready for life and work. For simplicity, all interaction with the application will be done via a simple text interface (no GUI is required). However, it will provide all basic functionalities of a practical application (may be further developed after completion).

DESCRIPTION

We will work with the following project idea (revised and developed from ISYS3416).

APP FOR VACATION HOUSE EXCHANGE

Assume that you are involved in a start-up who came up with an idea to make an app for people to exchange their houses on vacation. The app will allow registered members to use houses of other members when they go on vacation.

There is an initial entry fee of $500 (pay to the system) when **registering** as a member, which earns the new member *500* ***credit points***. Initial registrations must capture all the information necessary to be a member, including his/her personal info (*username, full name, phone number*), and the info of his/her house if available (*location*, *description*).

Each member *earns credit points* when the house is used by another member (e.g. who travels to other cities) and vice versa, can *use credit points* to occupy an available house.

The system will track two separate **score ratings** which can vary from **-10** to **+10**.

- The **house-rating score** will reflect the utility level derived based on average of ratings of the occupiers who have used the house in the past.
- The **occupier-rating score** is derived by averaging the ratings of all house owners who had let their houses to be used by that occupier (how well that occupier has taken care of the house).

Any member can **list his or her house** to be used in a particular **period** (start – end), **consuming points** (per day), and with or without specifying ***minimum required occupier rating*** for potential occupants. The *minimum occupier rating* can thus be used to prevent the house being rented to poor occupants or those for which no history is available. Of course, the member can **unlist the house** if needed.

Any member can **view all available houses** in a **specified time** (start – end), and **city**. The showed list should consist of only those houses for which the member has *adequate credit points and occupier rating*.

The information should also include the *house rating score* of the house. Members should also be given the option to **view the reviews** (score and comments) on any of the listed houses. Any member can **request to occupy** any property for which they are eligible.

The house owner can view all requests for his/her house, including *info and rating of the interested occupiers,* and may choose to *a request to accept (reject all other requests with overlapped time)*.

Non-members can **view all house details** (but not their reviews and availability), to encourage non-members to join.

CONSTRAINT

For the initial version of the app, it could have some limitations as below

1. Each member can only add one house to his/her account.
2. Each member can only occupy one house at a time.
3. Members are not allowed to cancel if the request is already accepted.
4. The application is early available to users in several cities: Hanoi, Saigon, Da Nang.

PROCESS AND GUIDE

A. Class Diagram: Identify classes, attributes, methods and relationships between classes. Sketch a class diagram for the application.

B. *Basic Features:*

Implement and test each feature. The application should have the following basic features:
Note: The program should validate user input accordingly. All attributes should be private for data encapsulation

1. A non-member can register to become a member (information is recorded).
2. A non-member can view all house details (but not their reviews and availability).
3. A member can login with the registered username and password, and can view all of his/her information.
4. An admin can login with predefined username and password, and can view information of all members and houses.
5. A member can list his/her house available to be occupied (with consuming points, and minimum required occupier rating), and unlist it if wanted.
6. A member can search for all available **suitable** houses for a particular **city** (suitable with his current credit points and rating score).
7. A member can request to occupy a house.
8. A member can view all requests to his listed house.
9. A member can accept a request (reject all other requests).
10. A member can occupy the successfully requested house.
11. A member can rate each of his/her occupied houses (*score and comment*).
12. A member can rate each of the occupiers who had used his/her house (*score and comment*).
13. All data must be saved into a **file** before the program is ended, and loaded into the program when it is started.

Help Note: *to manage all members, you can create a class (for the whole system) with attribute is vector of pointers of all member objects.*

C. ***Time Period Feature***: if you already completed all basic features, try to improve the application by supporting time period matter for the program as in its description, especially for features 5-9 *(e.g. list the house available for a time period; request to use the house in a period; only automatically reject non-accepted requests with overlapped time, etc.).*

UPDATED:

- *For **GROUPS OF THREE** (most of groups have 4 members, but several groups have three), features 10, 11, 12 are exempt (not required to implement).*
- *For **INDIVIDUAL WORKS** (for students did not work in groups in previous Group Activities), features 10, 11, 12 are exempt (not required to implement). Time Period Feature is required to implement with ONLY ONE DAY (list the house to be available in only one particular day, request the house for only one day, etc.)*

D. **Welcome Screen**

When completing your application, it should have a welcome screen with example content structure as below (you are free to adjust its format if prefer).

```
EEET2482/COSC2082 ASSIGNMENT
VACATION HOUSE EXCHANGE APPLICATION

Instructor: Mr. Linh Tran
Group: Group Name
sXXXXXXX, Student Name
sXXXXXXX, Student Name
sXXXXXXX, Student Name

Use the app as 1. Guest   2. Member   3. Admin
Enter your choice: 2
Enter username:……
……

This is your menu:
0. Exit
1. View Information
……

Enter your choice:……
……
```

SUBMISSION

1. Submit **source code** (zipped folder), compiled **executable file (.exe)** together with the **data file**.
2. Submit **report** with class diagram and explanation (*report template will be provided*).