**Course Work: Software Group Project**

Scenario:

We are a large multinational organization. We process data, analyze them and present them graphically. We make most of our software available as open source. We make most of our media available through Creative Commons. We make lots of money from support services and bespoke software enhancements. We expect our employees to be professional, ethical and compliant with the relevant laws, to manage themselves and their relation to others, and to deliver high quality products and services and deliver them on time.

The project is the following: A registered user will be able to access Hosting accommodation data, like AirBnb, Homeaway, VRBO, etc. Those rental websites list room or apartment based on location, price and review. Accommodations are administrated by host who pledge to meet conditions of those Website. Our goal is to use a data set which includes room’s criteria, create an application that will allow users to view data in specific forms, such as different graph types through a GUI.

To communicate, we will use a personal chat group and a version control, as a group we decided to use [WhatsApp](https://www.whatsapp.com/) to communicate and [GitHub](https://github.com/Khesnay/09-5SE2-1-1910_SoftwareProject) to manage the project versions. Hence, we set up different deadline which allows us to perform each task required to achieve such project. We agreed, as well, to split task to different parties, hence we can manage our time in a better way.

**Green annotations are subject to be removed as it only assign personal critic, raise issues encountered and possible suggestions.**

COMMENTS:

We divided the project in two parts, back-end and front-end. As we are assigned different task, we will tend to overlap to each others duty in the purpose of optimization. Thus, two of us will be working on the database and the other two are working on the GUI, and Java methods.

Database

(SQL)

Application

(Java)

Graph and statistics

(Excel and Java GUI)

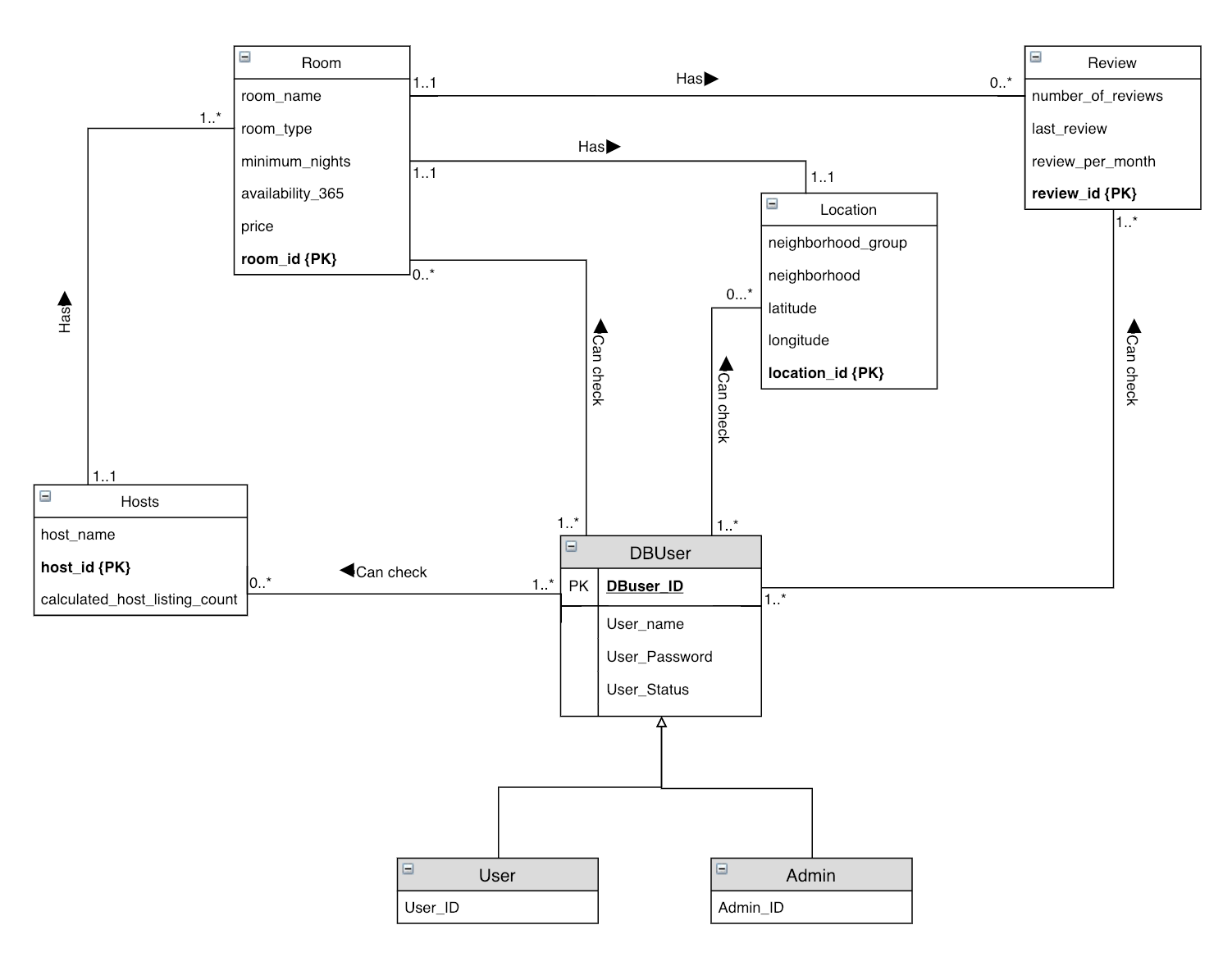
Send data

Display data

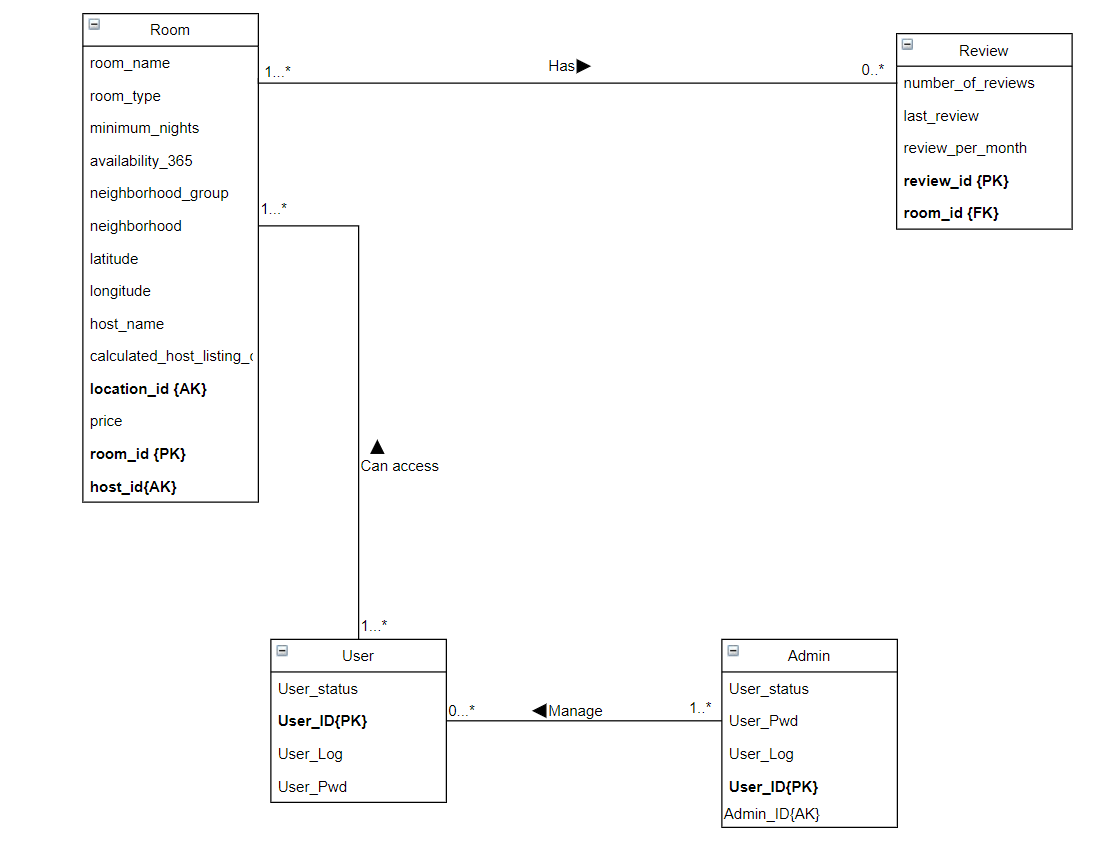
**Part A: Database**

As it is presented through the scenario, dataset and website, we need to decipher the status of each principal elements in our data. As we aim to process a fair amount of information through an application which will display it in a specific way. We need to identify each entities and their specifications through different dictionaries. Hence, we will use 4 dictionaries to justify our choice and process.

**Conceptual Entity Relationship Diagram**

****

**Logical Entity Relationship Diagram**

****

**Data dictionary: Entity**

|  |  |
| --- | --- |
| Entity | Justification |
| Room | Necessary entity has it will be the pinnacle of the DB. |
| Review | Added as an entity due to many attributions which can be related as a review. |
| Location | Added as an entity due to many attributions which can be related as a location. |
| Hosts | Referenced through websites, it can be |
| DBUser | Entity used to store differents users. |

**Data dictionary: Specialization**

|  |  |  |
| --- | --- | --- |
| Entity | Specialized entity | justification |
| DBUser | User | Classic user, that can simply be registered. |
| DBUser | Admin | User with capacity |

**Data dictionary: Attributes**

|  |  |  |
| --- | --- | --- |
| entity | Attributes (include PK) | justification |
| Room | room\_Name  room\_Type  minimum\_nights availability\_365  price  **room\_id{PK}** |  |
| Location | Neighbordhood\_group  Neighborhood  Latitude  Longitude  **Location\_id{PK}** |  |
| Review | Number\_of\_reviews  Last\_review  Review\_per\_month  **Review\_id{PK}** |  |
| Hosts | Host\_name  **Host\_id{PK}**  Calculated\_host\_listing\_count |  |
| DBUsers | User\_name  User\_password  User\_Status  **DBUser\_id{PK}** |  |

**Data dictionary: Multiplicities**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity | multiplicity | relationship | entity | multiplicity | justification |
| Room | 1..\*  1..1  1..1  0..\* | Has  Has  Has  Checked by | Hosts  Location  Review  DBUser | 1..1  1..1  0..\*  1..\* | -1 or many Room has 1 Host.  -1 Room has 1 Location  -1 Room has 0 or many reviews.  -0 o many rooms can be checked by one or many Users. |
| Review | 0..\*  1..\* | Can be  Checked by | Room  DBUser | 1..1  1..\* | -0 or many reviews can be one Room  -1 or many reviews can be checked by one or many Users. |
| Location | 1..1  0..\* | Has  Checked by | Room  DBUser | 0..\*  1..\* | 1 Location can have 0 or many room.  0 or man location can be checked by 1 or many Users. |
| Hosts | 1..1  0..\* | Has  Checked by | Room  DBUser | 1..\*  1..\* | 1 Hosts has one or man Rooms.  0 or many Hosts can be checked by 1 or many Users |
| DBUser | 1..\*  1..\*  1..\*  1..\* | Can check  Can check  Can check  Can check | Room  Location  Review  Hosts | 0..\*  0..\*  1..\*  0..\* |  |

COMMENTS:

As we decide on the Database structure, we need to think about how the application will fetch data through each topples. We could simplify the process with a table integrating all dataset and call them through specific queries.

Example: SELECT neighborhood FROM room WHERE neighborhood = “Ward 57”;

As neighborhood should be displayed through the DB, we could simply display (JAVA) it differently, for example through a pie chart or simple listing as only one type of data is called.

**Part B: UML**