Rakibul Islam

1271973

**Case Study**

Hotel Management System Use Case Diagram

This Use Case is a graphic depiction of the interaction among the elements of Hotel Management System it represents the methodology used in system analysis to identify, clarify, and organize system requirements of Hotel Management System. The main actors of Hotel Management System in this Use Case Diagram are : Super Admin, System Users, Employee, Anonymous Users, who perform the different type of use cases such as Manage Hotel, Manage Rooms, Manage Services, Manage Payments, Manage Booking, Manage Customers, Manage Users and Full Hotel Management System Operation. Major elements of the UML use case diagram of Hotel Management System are shown on the picture below.

The relationships between and among the actors the use case of Hotel Management System:

* **Super Admin Entity**: Use cases of Super Admin are Manage Hotel, Manage Rooms, Manage Services, Manage Payments, Manage Booking, Manage customers, Manage Users and full Hotel Management System Operation
* **System User Entity**: Use cases of System User are Manage Hotel, Manage Rooms, Manage Services, Manage Payments, Manage Booking, Manage Customers
* **Employee Entity**: Use cases of Employee are Manage Rooms, Manage Faculties, Search Rooms, Search Customers, Create Bills, Collect Payments
* **Anonymous Users Entity**: Use cases of Anonymous Users are view information, Fill Contact Us, Search content

Hotel Management System Sequence Diagram

This is the UML sequence diagrams of Hotel Management System which show the interaction between the object of Payments, Rooms, Booking, Services, Hotel. The instance of class objects involved in this UML Sequence Diagram of Hotel Management System are as follows:

* Payments Object
* Rooms Object
* Booking Object
* Services Object
* Hotel Object

Hotel Management System Activity Diagram

This is the Activity UML diagrams of Hotel Management System which show the flows between the activity of Hotel, Payments, Rooms, Booking, Customers. The main activity involved in this UML Activity Diagram of Hotel Management System are as follows:

* Hotel Activity
* Payments Activity
* Rooms Activity
* Booking Activity
* Customers Activity

Hotel Management System Class Diagram

A hotel room booking system requires customer information, room information, rent and  
booking process.  
Draw an UML class diagram from the above case study for hotel room booking. Use  
appropriate datatype, parameter and accessibility.

Hotel Management System State Machine

State machine diagram is a behavior diagram which shows discrete behavior of a part of designed system through finite state transitions. State machine diagrams can also be used to express the usage protocol of part of a system.

Hotel Management System CRC Card

* **Find classes**. Finding classes is fundamentally an analysis task because it deals with identifying the building blocks for your application. A good rule of thumb is that you should look for the three-to-five main classes right away, such as *Student*, *Seminar*, and *Professor* in Figure 4. I will sometimes include UI classes such as *Transcript* and *Student Schedule*, both are reports, although others will stick to just entity classes. Also, I'll sometimes include cards representing actors when my stakeholders are struggling with the concept of a student in the real world (the actor) versus the student in the system (the entity).
* **Find responsibilities**. You should ask yourself what a class does as well as what information you wish to maintain about it. You will often identify a responsibility for a class to fulfill a collaboration with another class.
* **Define collaborators**. A class often does not have sufficient information to fulfill its responsibilities. Therefore, it must collaborate (work) with other classes to get the job done. Collaboration will be in one of two forms: a request for information or a request to perform a task. To identify the collaborators of a class for each responsibility ask yourself "does the class have the ability to fulfill this responsibility?". If not then look for a class that either has the ability to fulfill the missing functionality or the class which should fulfill it. In doing so you'll often discover the need for new responsibilities in other classes and maybe even the need for a new class or two.
* **Move the cards around**. To improve everyone's understanding of the system, the cards should be placed on the table in an intelligent manner. Two cards that collaborate with one another should be placed close together on the table, whereas two cards that don't collaborate should be placed far apart. Furthermore, the more two cards collaborate, the closer they should be on the desk. By having cards that collaborate with one another close together, it's easier to understand the relationships between class.