Term Project Proposal

Hotel Reservation Manager

By: Phuc Nguyen, Junwei Ren, Yifan Dai, and David Ponder

1) Objective

We plan to develop a hotel reservation and management application. This application allows customers to search and make reservations, as well as make payments, with all the conveniences provided from a web applications. The system will also be in charge of ensuring that reservations will not conflict with one another. Finally, it will also monitor and keep track of each employee of the hotel.

2) Features

As discussed above, this system will have a lot of moving parts. As a brief outlook, here is a brief overview of some of the characteristics and pages that will be available to utilize within the system.

Users Page

This page allows customer to input personal information such as customer number, name, gender, address, email, phone number, username, and on-file credit card. All this information will be stored in a MySQL database.

Reservations Page

A page allowing users to place reservations on rooms of their choice. Attributes of both the customer and the requested room would create reservations, as well as some custom made for the reservation. Some attributes include the room number, customer number, check-in date, and check-out date.

Rooms

The database will maintain a record of all rooms available at the hotel. This will be the primary way to record and maintain rooms. Each room will have many attributes, such as room number, room type, and room condition.

Employees

Finally, the database will have a schema devoted to displaying information about the employees working at the hotel. The employee attributes will include name, address, work phone number, employee number, title, SSN, and hourly wage.

3) Proposed Technologies

We plan to use Play 2 for the web interface and MySQL to store data. This system will be implemented in Java.

4) Reason for MySQL

For our group, we have elected to use MySQL as the primary Relational Database Management Software. We decided upon this software due to its power and economic viability. It is also open source and easy to use. It also boasts a large amount of support from third party tools useful for GUI based implementation. Finally, it is more secure and scalable, making it a viable choice for this type of system.

5) Reason for Play 2

We have chosen Play 2 for a number of reasons. One of the primary reasons is due to our previous experience in using Play 2. During the semester, we were tasked with researching Play 2, and were able to familiarize ourselves with its toolset. Another reason is because of its simplicity. Play 2 makes it very easy to set up a basic restful API, which is very useful in a system such as this one. Finally, we've chosen it due to its flexibility with other Java libraries, making it easier to build the system.

6) Tentative Table Schema

Table Customer (GuestID int(6), Fname varchar(20), Lname varchar(20), Address varchar(100), primary key (GuestID));

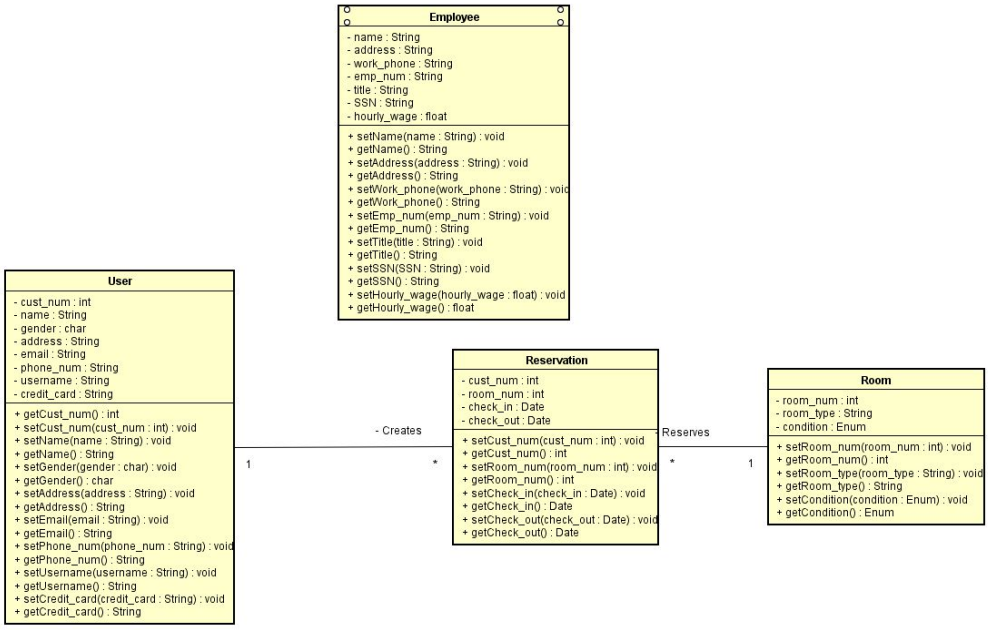
Table Room (RmNum int(4), RmType varchar(10), price double, RmCondition varchar(10), Availability boolean, primary key (RmNum));

Table Reservation (ReservationID int(20), GuestID int(6), moveindate int(8), moveroutdate int(8), numP int (2), primary key (ReservationID));

Table Employee (Fname varchar(10), Lname varchar(10), address varchar(50), tel_num varchar(15), employeeID int(3), SSN int(9), primary key SSN);

7) Class UML Diagrams

Below is a UML Diagram describing the classes and their relationships in this system. As this system is still in production, this is subject to change; however, the general architecture should remain consistent.


Employee
- name : String
- address : String
- work_phone : String
- emp_num : String
- title : String
- SSN : String
- hourly_wage : float
+ setName(name : String) : void
+ getName() : String
+ setAddress(address : String) : void
+ getAddress() : String
+ setWork_phone(work_phone : String) : void
+ getWork_phone() : String
+ setEmp_num(emp_num : String) : void
+ getEmp_num() : String
+ setTitle(title : String) : void
+ getTitle() : String
+ setSSN(SSN : String) : void
+ getSSN() : String
+ setHourly_wage(hourly_wage : float) : void
+ getHourly_wage() : float
User
- cust_num : int
- name : String
- gender : char
- address : String
- email : String
- phone_num : String
- username : String
- credit_card : String
+ getCust_num() : int
+ setCust_num(cust_num : int) : void
+ setName(name : String) : void
+ getName() : String
+ setGender(gender : char) : void
+ getGender() : char
+ setAddress(address : String) : void
+ getAddress() : String
+ setEmail(email : String) : void
+ getEmail() : String
+ setPhone_num(phone_num : String) : void
+ getPhone_num() : String
+ setUsername(username : String) : void
+ getUsername() : String
+ setCredit_card(credit_card : String) : void
+ getCredit_card() : String
- Creates
1
*
Reservation
- cust_num : int
- room_num : int
- check_in : Date
- check_out : Date
+ setCust_num(cust_num : int) : void
+ getCust_num() : int
+ setRoom_num(room_num : int) : void
+ getRoom_num() : int
+ setCheck_in(check_in : Date) : void
+ getCheck_in() : Date
+ setCheck_out(check_out : Date) : void
+ getCheck_out() : Date
Reserves
*
1
Room
- room_num : int
- room_type : String
- condition : Enum
+ setRoom_num(room_num : int) : void
+ getRoom_num() : int
+ setRoom_type(room_type : String) : void
+ getRoom_type() : String
+ setCondition(condition : Enum) : void
+ getCondition() : Enum