

HOTEL MANAGEMENT SYSTEM

```

class Hotel {
    String Name;
    Integer id;
    Location hotelLocation;
    List<Room> roomList;
}

class Location {
    Double longitude;
    Double latitude;
}

class Room {
    String roomNumber;
    RoomStyle roomStyle;
    RoomStatus roomStatus;
    Double bookingPrice;
    List<RoomKey> roomKeys;
    List<HouseKeepingLog> houseKeepingLogs;
}

public enum RoomStyle {
    STANDARD, DELUX, FAMILY_SUITE;
}

public enum RoomStatus {
    AVAILABLE, RESERVED, NOT_AVAILABLE, OCCUPIED, SERVICE_IN_PROGRESS;
}

class RoomKey {
    String keyId;
    String barCode;
    Date issuedAt;
    Boolean isActive;
    Boolean isMaster;

    public void assignRoom(Room room);
}

class HouseKeepingLog {
    String description;
    Date startDate;
    int duration;
    HouseKeeper houskeeper;

    public void addRoom(Room room);
}

abstract class Person {
    String name;
    Account accountDetail;
    String phone;
}

public class Account {
    String username;
    String password;

    AccountStatus accountStatus;
}

public enum AccountStatus {
    ACTIVE, CLOSED, BLOCKED;
}

class HouseKeeper extends Person {
    public List<Room> public getRoomsServiced(Date date);
}

class Guest extends Person {
    Search searchObj;
    Booking bookingObj;

    public List<RoomBooking> getAllRoomBookings();
    public RoomBooking createBooking();
    public RoomBooking cancelBooking(int bookingId);
}

class Receptionist extends Person {
    Search searchObj;
    Booking bookingObj;

    public void checkInGuest(Guest guest, RoomBooking bookingInfo);
    public void checkOutGuest(Guest guest, RoomBooking bookingInfo);
}

class admin extends Person {
    public void addRoom(Room roomDetail);
    public Room deleteRoom(String roomId);
    public void editRoom(Room roomDetail);
}

```



```

/
class search {
    public List<Room> searchRoom(RoomStyle roomStyle, Date startDate, int duration);
}

class RoomBooking {
    String bookingId;
    Date startDate;
    int durationInDays;
    BookingStatus bookingStatus;
    List<Guest> guestList;
    List<Room> roomInfo;
    BaseRoomCharge totalRoomCharges;
}

/**
 * Decorator pattern is used to decorate the prices here.
 */

interface BaseRoomCharge {
    Double getCost();
}

class RoomCharge implements BaseRoomCharge {
    double cost;

    Double getCost() {
        return cost;
    }

    void setCost(double cost) {
        this.cost = cost;
    }
}

class RoomServiceCharge implements BaseRoomCharge {
    double cost;
    BaseRoomCharge baseRoomCharge;

    Double getCost() {
        baseRoomCharge.setCost(baseRoomCharge.getCost() + cost);
        return baseRoomCharge.getCost();
    }
}

class InRoomPurchaseCharges implements BaseRoomCharge {
    double cost;
    BaseRoomCharge baseRoomCharge;

    Double getCost() {
        baseRoomCharge.setCost(baseRoomCharge.getCost() + cost);
        return baseRoomCharge.getCost();
    }
}

class Booking {
    public RoomBooking createBooking(Guest guestInfo);
    public RoomBooking cancelBooking(int bookingId);
}

```