HMS Reference Document

1. External libraries:

No external libraries are used with in this coursework project (I wrote all code myself).

2. A brief walk through:

Default guest account & staff account:

1) Guest:
    a) Username: Batman Password: iamveryrich
    b) Username: Superman Password: myunderwearisred
    c) Username: Kobe Password: 1234
    d) Username: Johnney Password: 1234Caesar

2) Staff:
    a) Username: JianJun Chen Password: 1234567890
    b) Username: Master Password: 1234567890

Insert Rooms (InsertRooms.php):

To insert all hotel rooms into the hotel room table. Notice that it could only be executed once, otherwise you will get duplicated room list.

Home page (index.php):

When a user firstly accesses the HMS webpage, there exists three routes for him to choose. First, click “Sign in” button to jump to “sign in” page. Second, click “Start Trip Now” to jump to “select room requirement” page to preview room information without sign in process. Third, wherever the client is, he could jump back to home page by clicking Sunny Isle’s logo on left-hand side of the navigation bar.

Sign in page (SigninPage.html):

This page takes client's username and password as input and jumps to the page client wants to access after he pass the database validation. And the "staff sign in" page could be accessed by clicking the "staff sign in button" on the right-hand side of the navigation bar. The user could register a new account by clicking "account creating link" below submission button.

Register page (RegisterAccount.html):

This page collects new clients' personal information to build new account for them. It will automatically check clients' username' duplication, email format, telephone format (only numbers and "-" are allowed) and password correctness, then remind user to correct their information. After clients pass the validation test, they will be sent to page they want to access.

Select room type page (Bookhotel.html):

This page permits clients to select time, room type and asks whether they want to select rooms themselves or not. And this page will check the correctness of time automatically. If clients do not want to select rooms their selves, they must input the number of rooms they want to book by select a number between 1 and 10, and he will be announced if this type of room is not enough. Otherwise client will be sent to "view order page". The client is not permitted to input number of rooms themselves to prevent malicious reservation. If the client does want to register an order contains more than ten rooms, he could use book again to make registration much more convenient. Else if the client chooses to select rooms themselves, this page will send them to "select room page".

If the client has not sign in yet, they will be asked to sign in or register first. And after they signed in, they will be sent to the page they chose to access. By the way, client could get back by using the navigation bar.

Select room page (SelectRoom.html):

This page permits clients to select rooms themselves and they could cancel the selected room by clicking the room buttons on map or order buttons within the order list. Once they clicked confirm, their order will be submitted, and they can view their orders in view order page. And if they clicked back, they will be sent to the room requirement selecting page and allowed to reselect their check in, check out date and room type. This page's interactions is well constructed.

View order page (ViewOrder.html):

In this page, the client could check the order they made and choose to book again or log out. He can access his own profile page as well, to modify or check his personal profile.

My profile page (Myprofile.html):

This page could be accessed by clicking on My profile button on left hand side of navigation bar. In the page, client could check and modify their personal information. This page will automatically check users' email and telephone format as well as the "Register Page".

Manage order page (ManageOrder.html):

This page could be accessed from My profile page. And it allows guests to view the orders he has made before and delete anyone of them if he wants. A guest will be asked to confirm their order if he wants to delete one of his orders. And the guest could log out directly from this page and jump to the home page. Or he could jump back to his profile page.

Staff sign in page (StaffSignin.html):

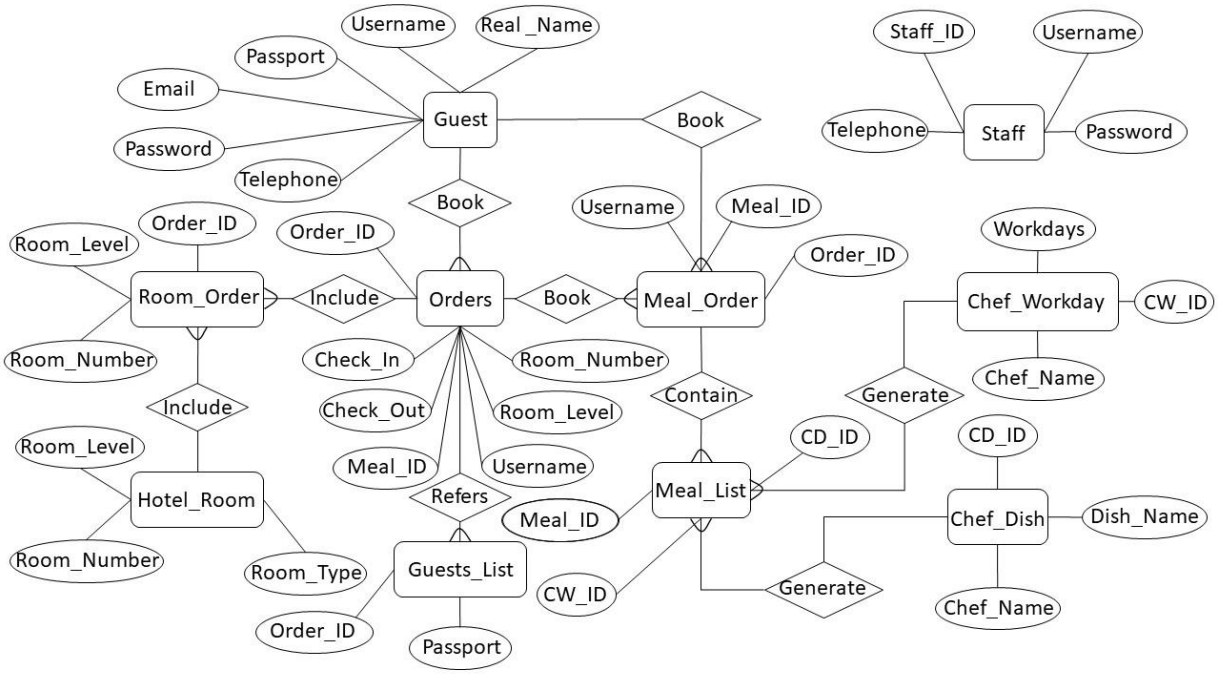
This page could be accessed from "Sign in page", and it checks staffs' usernames independently. After staff members passed the validation check, this page will send him to "staff view page".

Staff View page (StaffView.html):

In this page, staff member could view all booked hotel rooms. And they will be divided into 3 groups: Overdue, Booked and Checked in. Staffs could log off from this page as well.

3. Further study:

a. ER-Diagram:



b. SQL Statement:

```
CREATE TABLE Staff(
Staff_ID            VARCHAR(255)   NOT NULL   UNIQUE,
Username            VARCHAR(255)   NOT NULL   UNIQUE,
Telephone           VARCHAR(255)   NOT NULL,
Password            VARCHAR(255)   NOT NULL,

CONSTRAINT
        PRIMARY KEY (Staff_ID)
);


CREATE TABLE Guest(
    Username        VARCHAR(255)   NOT NULL   UNIQUE,
    Passport        VARCHAR(255)   NOT NULL   UNIQUE,
    Realname        VARCHAR(255)   NOT NULL,
    Telephone       VARCHAR(255)   NOT NULL,
    Email           VARCHAR(255)   NOT NULL,
    Password        VARCHAR(255)   NOT NULL,

    CONSTRAINT
```

```
        PRIMARY KEY (Username)
);


CREATE TABLE Chef_Workday(
    CW_ID           VARCHAR(255)  NOT NULL   UNIQUE,
    Chef_Name       VARCHAR(255)  NOT NULL,
    Workday         VARCHAR(255)  NOT NULL,

    CONSTRAINT
        PRIMARY KEY (CW_ID)
);


CREATE TABLE Chef_Dish(
    CD_ID           VARCHAR(255)  NOT NULL   UNIQUE,
    Chef_Name       VARCHAR(255)  NOT NULL,
    Dish_Name       VARCHAR(255)  NOT NULL,

    CONSTRAINT
        PRIMARY KEY (CD_ID)
);


CREATE TABLE Hotel_Room(
    Room_Level      VARCHAR(255)  NOT NULL,
    Room_Number     VARCHAR(255)  NOT NULL,
    Room_Type       VARCHAR(255)  NOT NULL,

    CONSTRAINT
        PRIMARY KEY (Room_Level, Room_Number)
);


CREATE TABLE Guest_List(
    Passport        VARCHAR(255)  NOT NULL,
    Order_ID        VARCHAR(255)  NOT NULL,

    CONSTRAINT
        PRIMARY KEY (Passport,Order_ID),
    CONSTRAINT
        FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID)
);
```

```
CREATE TABLE Orders(
    Order_ID        VARCHAR(255)   NOT NULL UNIQUE,
    Room_Level      VARCHAR(255)   NOT NULL,
    Room_Number     VARCHAR(255)   NOT NULL,
    Check_In        VARCHAR(255)   NOT NULL,
    Check_Out       VARCHAR(255)   NOT NULL,
    Username        VARCHAR(255)   NOT NULL,
    Meal_ID         VARCHAR(255)   NOT NULL,

    CONSTRAINT
        PRIMARY KEY (Order_ID),
    CONSTRAINT
        FOREIGN KEY (Username) REFERENCES Guest(Username)
);


CREATE TABLE Room_Order(
    Order_ID        VARCHAR(255)   NOT NULL,
    Room_Level      VARCHAR(255)   NOT NULL,
    Room_Number     VARCHAR(255)   NOT NULL,

    CONSTRAINT
        PRIMARY KEY (Order_ID,Room_Level, Room_Number),
    CONSTRAINT
        FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID),
    CONSTRAINT
        FOREIGN KEY (Room_Level, Room_Number) REFERENCES
Hotel_Room(Room_Level, Room_Number)
);


CREATE TABLE Meal_Order(
    Meal_ID         VARCHAR(255)   NOT NULL UNIQUE,
    Username        VARCHAR(255)   NOT NULL,
    Order_ID        VARCHAR(255),

    CONSTRAINT
        PRIMARY KEY (Meal_ID),
    CONSTRAINT
        FOREIGN KEY (Username) REFERENCES Guest(Username),
    CONSTRAINT
        FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID)
);
```

```
CREATE TABLE Meal_List(
    Meal_ID         VARCHAR(255)   NOT NULL,
    CW_ID           VARCHAR(255)   NOT NULL,
    CD_ID           VARCHAR(255)   NOT NULL,

    CONSTRAINT
        PRIMARY KEY (Meal_ID, CW_ID, CD_ID),
    CONSTRAINT
        FOREIGN KEY (Meal_ID) REFERENCES Meal_Order(Meal_ID),
    CONSTRAINT
        FOREIGN KEY (CW_ID) REFERENCES Chef_Workday(CW_ID),
    CONSTRAINT
        FOREIGN KEY (CD_ID) REFERENCES Chef_Dish(CD_ID)
);
```

c. Brief Discussion:

In this part of the report, the construction of entire database including 10 tables in total will be clear demonstrated one by one. And the order is from outer tables which are referenced by other tables' foreign key, to inner tables which apply foreign key to other tables to set up connections.

1) Staff table:

   The staff table is built first in that thought staffs are dealing with great number of issues in this hotel, they hardly could be joint into any of the other tables. So, the staff table turns out to be an isolated table which is a perfect choice for beginning. Because each staff's staff ID should be unique, it is chosen to be the primary key.

2) Guest table:

   Then the tables within outer cycle are going to be constructed. Start with Guest table, Username has been chosen to be the primary key. And it is refered Meal_Order in that a guest could directly book a meal with out booking a room if he is a registered guest.

3) Chef_Workday table:

   This table is built to store chefs' working schedules. And as a result, a CW_ID column is set to be this table's primary key in that duplications exist within the other two columns and by doing so other tables are much easier to refer to this one.

4) Chef_Dish table:

Similar to Chef_Workday table, this table consists of CD_ID, Chef_name and Dish_Name columns to store Chefs' dish sets. And the CD_ID columns is chosen to be the primary key as well. And both of these two tables are referred by Meal_List table in order to combine chefs' working schedule with their dish set while avoiding duplication.

5) Hotel_Room table:

This table is constructed to contain the information of all rooms within this hotel. And the Room_Level and Room_Number are chosen together to be this table's primary key.

6) Guest_List table:

This table is built to contain the checked in passengers' passport ID. Because for some types of room which could contain more than one person within, so the guest list part is spilt out and be referred by Orders table.

7) Orders table:

After the construction of Guest_List table, Order table could be built by apply a foreign key to Guest_list table. This is an important table which operates like a transfer station and it contains great amount of information. It is referred by two tables: Room_Order and Meal_Order. For the former one, because passenger should be enabled to book a meal as well and the cost could be paid by the registered guest who booked this room. And for the later one, because one room could be bonded with several orders and one order could contain several rooms, a transfer table is needed which is called Room_Order table.

8) Room_Order table:

As the document mentioned previously, this table is created to connect Orders table and Hotel_Room table. It contains two primary key elements from both tables, so there exists three columns called Order_ID, Room_Level and Room_Number. In this way, two foreign keys could be constructed to connect these two tables together.

9) Meal_Order table:

This table contains the meal orders form guests and passengers. It is connected with guest table and Orders table simultaneously. The Order_ID has not been constructed as a NOT NULL one because guest should be allowed to have meal in this hotel without staying in there for at least one night. In this way, the Order_ID could be NULL.

10) Meal_List table:

Eventually, Meal_List table is the last one to be constructed because it refers three different tables in total. And it could only be built after the constructions of the other three tables. This table works like a transfer station as well. It contain three primary key columns from other tables and joint them together to create its own primary key.