Hotel Reservations 1

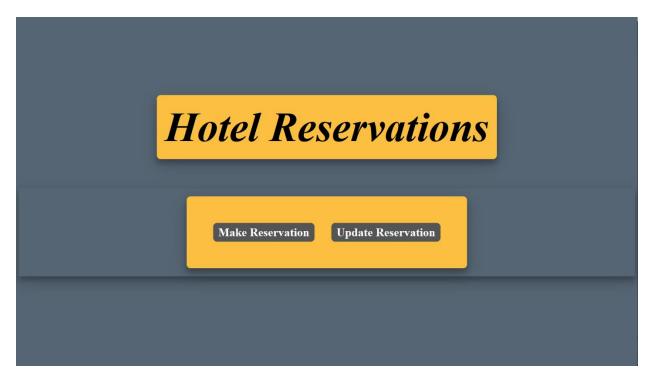**CST 438**

Final Report

Team 3

Jasper Kolp and Ramon Lucindo

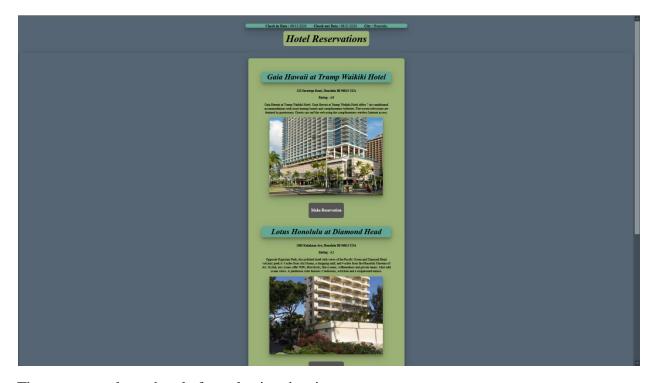**https://cst438-hotel-reservation.herokuapp.com**

This is the "Hotel Reservation" homepage. The user can either make a reservation or update an existing reservation. Each page subsequent has the "Hotel Reservations" banner as a link to the homepage. The colors of certain banners, headings and backgrounds change via CSS.

If the user chooses to make a reservation, this is the page where he/she can enter his/her first name, last name, and email address.
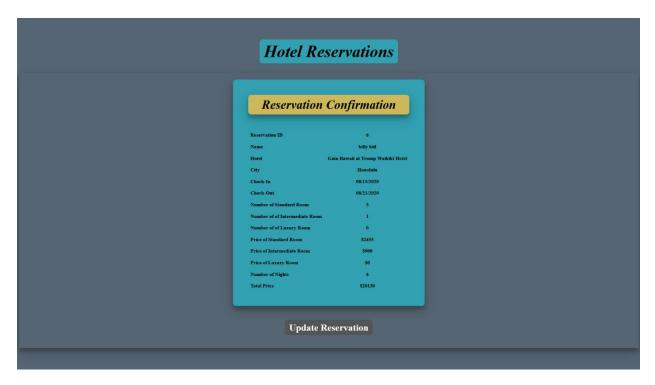
The user can then select the check-in and check-out dates via dropdown calendar options. He/she can also select from 3 cities from which to seek hotels: Honolulu, Monterey, and San Diego.

The user can select a hotel after selecting the city.

After selecting a hotel, the user can now select types and number of rooms per type: Standard, Intermediate or Luxury.
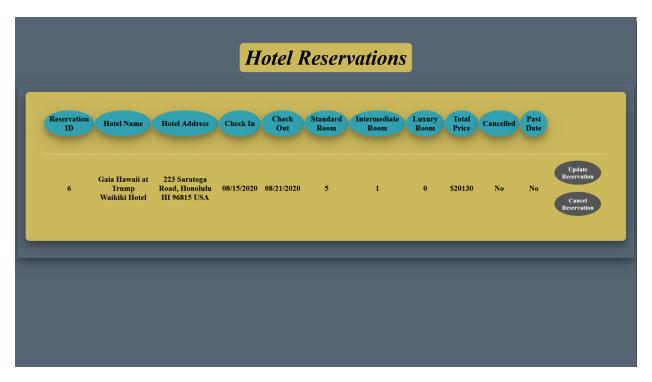

Once the user finishes making a reservation, this page serves as a reservation confirmation and online receipt providing a Reservation ID number and other information regarding the

reservation. The user can either go back to the homepage via the banner link or click on the button "Update Reservation," which also appears on the homepage.

If the user ever clicks on the "Update Reservation" buttons on the homepage, the reservation confirmation page or the display reservations page (shown below), he/she will land on this page and has the option to find one reservation based on the Reservation ID provided in the reservation confirmation page or possibly multiple reservations based on the user's email address.

Hotel Reservations

Reservation ID	Hotel Name	Hotel Address	Check In	Check Out	Standard Room	Intermediate Room	Luxury Room	Total Price	Cancelled	Past Date	
6	Gaia Hawaii at Trump Waikiki Hotel	223 Saratoga Road, Honolulu HI 96815 USA	08/15/2020	08/21/2020	5	1	0	$20130	No	No	Update Reservation / Cancel Reservation

Once the user selects a reservation by Reservation ID or email address, this page lets the user view information about the reservation and has the option to either update the reservation or cancel it.

Hotel Reservations

Cancel Complete

If the user chooses to cancel the reservation, he/she is taken to this page, notifying him/her that the reservation was cancelled successfully.
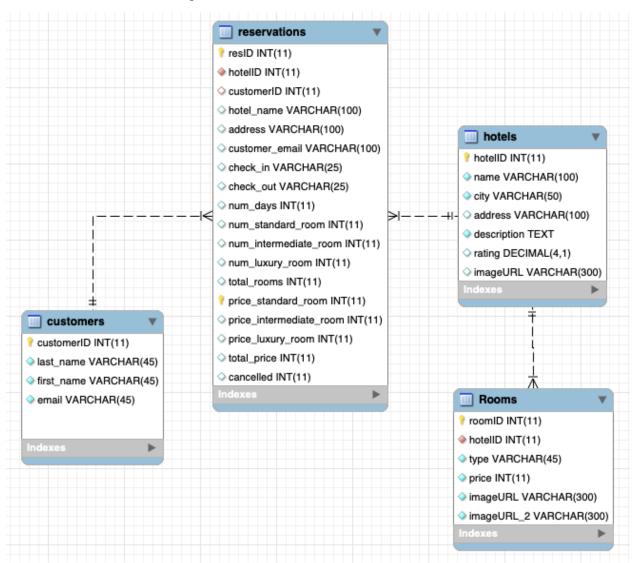
Database Design

For our database, we created 4 tables that record reservations, customer information, hotel information, and room information. In the beginning of making the reservation, customers are asked to provide their first name, last name, and email address. This information is stored within the customers table.

After the customer decides the check-in and check-out dates with their choice of city, our site will provide a list of hotels in the city of the customer's choice. All the information about the hotel is stored inside the hotels table. Columns include: hotelID, hotel name, city, address, short description, rating, and image URL.

After the customer chooses a hotel, our site will display the rooms that are available for that hotel. These rooms are related to the hotel table with hotelID as its foreign key.

Each table is created independently from each other, except the reservations table, to allow new records without breaking the code.

Project Experience

This project demonstrated a little bit of what it would be like to work at a professional environment. While working with teammates on the same service, we had to discuss code review often to make sure that we as a team are on the same page. By repeatedly reviewing each other's code, we had fewer misunderstandings of each other and allowed faster production than if only one person were to code the whole project. This project taught us teamwork. Secondly, this project also requires collaboration between different services. This forced us to think about the needs of other services and build our code and database based on their needs and our needs. The collective teamwork collaboration helped spread the production of the bigger system with different services by different teams. We overall did not have any issue collaborating with other teams.

One of the biggest mistakes we learned while doing the project is that we started testing after the app was deployed to Heroku. By the time we had deployed our program to Heroku, we had missed the timing of testing for functions and unit testing. We then had to create a different project to run on localhost to start testing the entire program from start to finish. In our experience, this took a good amount of our time and we realized that testing should be done while coding the program. Not only did we learn about the timing of testing, but we also learned about the importance of testing. Junit testing with Mockito allowed us to create stubs, expected values, to compare with real values. By doing so, we became more confident of the functions we created. Next time we build a larger project, we will make sure to start testing along with the coding and design of the program.