Julio Munoz

12/6/24

Final Project Documentation File

GitHub link: https://github.com/julio28226/FinalProject.git

In this documentation I will discuss the PHP files created to have an operating restaurant database. In this database, there will be options to view reservations, view preferences, add preferences, add reservation, and view customers. There are additional links to delete reservations and preferences, also the option to add a special request. I want the user to easily navigate the fields for simpler use, so I stuck to the basics. There is also a MySQL database that I will go over to describe how the information is set up. I will start with that. I will also cover some roadblocks that I encountered during my time making this project. A user guide will be created to show you how to access the restaurant portal and use the templates and links created.

Before the integration of the MySQL database, I first created a database named ‘restaurant_reservations’ in PhpMyAdmin. I could have created the database in the SQL code, but I decided to do it this way to learn how to use PhpMyAdmin a little bit more. The SQL code starts off with set commands and a start a transaction command. The following lines honestly, I am not sure what they do. I looked it up online and inserted it into the code. It runs correctly so I left it in there. Next, I created a ‘customers’ table. If the said table already exists, it will be dropped. In the ‘customers’ table, I created a ‘customerId’, ‘customerName’, ‘contactInfo’, and set ‘customerId’ as a primary key. I inserted entries into the ‘customers’ table through MySQL as well as using the add reservation link in the restaurant portal. For the ‘diningPreferences’, I created it the same way I did the ‘customers’ table. If a ‘diningpreferences’ table already exists, then it will be dropped. I created entries such as ‘preferenceId’, ‘customerId’, ‘favoriteTable’, and ‘dietaryRestrictions’. I set ‘preferenceId’ as the primary key and ‘customerId’ as the foreign key. I also inserted entries into the ‘diningPreferences’ table. There is also the option of adding entries using the restaurant portal as we did earlier. In the ‘reservations’ table, I created the ‘reservationId’, ‘customerId’, ‘reservationTime’, ‘numberOfGuests’, and ‘specialRequests’ entries. The primary key is ‘reservationId’ and the foreign key is ‘customerId’. I populated this table with three entries. I also used an alter table command for ‘diningPreferences’ which will allow me to add or delete information from the table.

Now I will discuss the templates for the final project. I will start with ‘addPreference’. This code will allow me to add a preference as the name implies. It starts with HTML code. The head of the code contains the title of the page along with the script and links of the CSS. Next, in the body, it contains a header named 'Add Dining Preferences'. The body has all the information that you see on the page such as the customer selection, the favorite table option, and the dietary restrictions information. After that, there is script detailing the customer, customer name, and contact information.

In the add reservation code, there is HTML code in the beginning followed by the head of the script. In the head, it contains the title which goes by 'Add Reservation'. A script and links are also embedded in the head. This format is similar to the 'addPreference' code so I will only cover the differences in each page and its functionality from now on to avoid redundancy.

On the add reservation page, you have the options of choosing a customer or creating a new one. If a customer already exists, it will be part of the selection in that field. Also, multiple reservations can be made for one customer. Next, we have the customer name to input followed by the contact information. A reservation time in the month, day and year format is applied including the time of the reservation. The number of guests and special requests are also information that needs to be entered.

In the view reservations link located in the portal, you will have access to view all of the reservations. This includes reservation ID, customer ID, customer name, reservation time, number of guests, and special requests. The reservations section allows you to individually view a reservation. You also have the option to change the special request or delete it.

Now I will take the time to talk about some of the problems that I ran across with the codes. For some reason unknown to me I can only make some of the templates work through the restaurant portal links. For example, to add a special request, you can do so through the add reservation link but when I input ‘addSpecialRequest.php’ in the localhost I get hit with a warning message. I am not sure why this is but to avoid this problem you can just use the links in the index or home PHP. It is better to have links in a project instead of having to type in each set of information in the address bar of the localhost. This allows for a smoother user interface and less keystrokes.

Another problem I encountered was with the MySQL database. I had to peruse the collation tab when creating the database because it was set to another which was different from our regular use. I found that problem and corrected it although it did take some time for me to figure it out. If you encounter this problem, just check the collation to see if it is set correctly on your end as well. It must be set to utf8mb4_general_ci. The occurrence of this problem might have been started accidentally by me when creating the database. It was corrected and now we can move on.

The ‘RestaurantDatabase’ PHP and the ‘RestaurantServer’ PHP hold all the public and private functions needed to make this project work. Functions such as ‘addReservation’, ‘addSpecialRequest’, or ‘deleteReservation’ are stored within these code files. Inside of the restaurant portal you can add or delete a reservation, add or delete a preference, or view reservations and preferences. The two PHP codes work in the background to ensure seamless fetching and inputting information from the restaurant portal and the database.

There are options to delete customers in the customers link from the restaurant portal. There are also delete reservation actions when you view the reservations that the restaurant has. Next to this, there is also the option to make a request for this reservation and to view a specific reservation. Another delete action can be found in the view preferences link. When you select the delete customer action, all of the information for that customer will be deleted. If that deleted customer had a preference or a reservation, they would be deleted as well.

I have created a simple to use restaurant reservations portal capable of many functions. A user can add reservations, view reservations, delete reservations, add or remove customers, add preferences, view or remove preferences all using the templates provided. Some CSS was used to create a template to view the restaurant portal. In that portal are links to access the information that is desired. I covered the details of this project through the use of MySQL, PhpMyAdmin, and the creation of HTML and PHP code. Although I ran into some trouble creating the platform, I found ways to correct those errors. Google search helped me through some of those problems.