**CSCE 4523 Database Management Systems**

**Homework 1**

**By: NAME(s)**

**Objectives**

Utilize the database that was created in HW3 to call query commands and edit the database from using code. Then build and application that allows a user to do multiple functions to the database, without having to write the SQL query’s every single time.

**Approach**

The language C++ was chosen and the code consists of a main loop which has a switch statement for each of the 6 functions. Then in each of the cases there will be user input depending on the functionality of the cases. This is handled by a get line. Then another function is called to make sure the inputted thing exists in the database before it continues to query the data. Finally a final function is called which completes the task of the case. For example in case 2 Ordering, the user inputs a dishName. It checks if the dishName exists then if the dishNo exists in MenuItems before it proceeds. It then shows all available dishes with the inputted name, and asks the user to input the item number they would like to order. Once the user inputs this number, a check is run on the database to see If that ItemNumber exists, if it doesn’t then it exits back to the main screen, else it uses cron to get the date and time and appends them to ItemNumber along with a OrderNum which is set the next highest integer in the FoodOrder table. Then this data is inserted into the array via a query() call, and then the FoodOrder table is displayed with another query() call. This is an example of one of the cases, but this is how the remainer of the cases were implemented.

A black screen with white text

Description automatically generated

Above is test of Case 2: ordering food.

Sample code was provided, which gave the basis to connect to the database, and some examples on how to query information.

**Results**

As mentioned above there was error handling on every level of the programming. Any time a user inputs some form of info a call to the database to see if that info exists would first be made, before any data was queried. The querying worked well for the initial setup, with some errors with inserting the data that was handled by just manually inserting it instead of using the function. The difficult part was making all the error checking functions, and making sure the user could not enter in the wrong info.

**Testing**

Testing the program went as follows. Implement a case with the basic features. Run the case and try different combinations of inputs to see what was allowed and what wasn’t. Find what was allowed but shouldn’t be allowed and note it for future reference. Then add a function to check for that case. Repeat until all cases are handled. Everything worked as intended in the typescript, and all the testcases were met and then some. One of the changes that had to be made was to Case2, where the error checking would check the Dish table and see if the dish existed, but wouldn’t check the MenuItem table to see if it was on the menu. Leading to a case where if Curry was inputted, then the error checking would not flag it. To fix this another function was created to check the Dish table to see if the dish existed, get the dishId, then check the MenuItem table to see if it exists in the menu.