# CS-355 Databases

# Fall 2019

# Final Project Report

**DBMS for a Restaurants Chain**

**Laiba Fatima Khan (lk04067)**

**Muhammad Shahzain (ms03977)**

**Kabir Kumar (kk03925)**

# *Submitted to*

**Dr. Ayaz ul Hassan Khan**

****

Habib University, Karachi

**PROPOSAL**

**Introduction**

For our project, we plan to build a database system to manage a [hypothetical] chain of restaurants in Karachi. While softwares do exist for this in real world, we want to implement our version of a database system to manage a restaurant chain, based on our own descriptions.

The database will hold information regarding the restaurants associated with the chain, the food items and drinks offered by these restaurants, the employees working there, and the customers and their orders placed at any restaurant of the chain.

The database management system will allow the user to enter new information and search from the existing information in the database.

**Modules of the System**

The important entities in our database include Restaurants, Food Items, Items Categories, Customers, Orders, Employees and Job Categories.

The system will be able to generate bills for all orders placed by customers at any restaurant of the chain. Each order will be assigned to a waiter in the case of dine-in and to a delivery person otherwise. Both waiters and delivery people can handle several orders at once. Total bill be calculated as the total amount of food ordered in the case of dine-in and will charge an extra fee for home delivery. System will also keep track of regularly visiting customers who will receive the options of deals and discounts.

The system will also generate pay slips for employees based on hourly wages rate for their particular job and their work hours per day.

**Tools & Technologies**

**Back-end:** Microsoft SQL Server

**Front-end:** Windows Forms App (language: C#)

**Front-end Development**

The DBMS Application for user will be a Windows Forms App developed in C#, with back-end connectivity to the actual Database through SQL Server queries.

Our goal is to have an app which mostly handles the billing of customer orders at the restaurants, by keeping track of the foods and drinks the restaurant offers along with their prices.

**Form 1**: User **Login** (only specific staff can access the database. For example, cashier, accountant, manager etc.) If user Id is in Staff table of the database and their position allows access, Id and password will be checked. If matches, login will be successful and they move to Form 2.

**Form 2**: User will choose ‘**Add Order’** to enter details of new order or **‘Search Order’** to search records of a previous order. There will be two other buttons **‘Add Staff’** and **‘Search Staff’** which will be enabled only if the database app is accessed by the staff whose position is Manager.

**Form 3**: If the **‘Add Order’** button is chosen from Form 2; this form will open and user will add the information of newly placed order. Pressing ‘Add’ button will insert the details in the database via SQL Query. The bill will be calculated on back-end and total amount will be filled in the ‘Total Bill Amount’ field by the system.

**Form 4**: If **‘Search Order’** is chosen from Form 2; this form will open and user will add some known details to find an order that fits the criteria. Pressing ‘Search’ will run a Select SQL Query to retrieve the orders and Project the relevant details in the List-box. Clicking an item in List-box and pressing ‘View’ will take user to Form 3, this time filled with the details and disabled for editing.

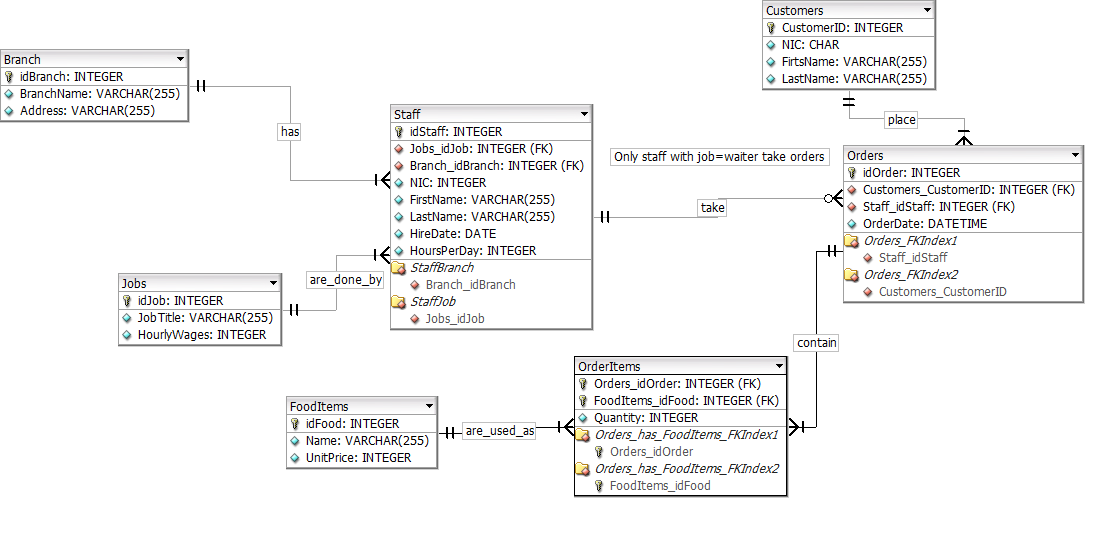
**Form 5**: If login Id is of a Manager and **‘Add Staff’** is chosen, the manager will be able to enter details of a newly appointed staff (same as adding order).

**Form 6**: If login Id is of a manager and **‘Search Staff’** is chosen, the manager will be able to retrieve all the staff members who fit the criteria in List-box, clicking on a specific item in list-box and pressing ‘View’ will show details of the respective staff in Form 5 (same as the search order) this time however the button ‘Staff Quit’ and ‘Generate Pay Slip’ will not be disabled like all the other fields. Pressing ‘Staff Quit’ will update the status of Staff in the database from ‘Active’ to ‘Quit’. Pressing ‘Generate Pay Slip’ will open a **new Form**, on selecting the ‘From Date’ and ‘To Date’ the system calculates total work hours and salary for the selected duration of time and fill it in the respective fields of the form.

**ADDITIONAL:**

In Form 2, we can also add an option to check popularity of a items or staff. ‘Check Popularity’ in Form 2 will take us to a new Form where we can choose from Foods/Drinks/Chefs/Waiters etc., select ‘From Date’ and ‘To Date’ and then select ‘Most Ordered’ or ‘Most Popular’ which gives us the Top 3 best results.

**ERD: DATABASE DIAGRAM**



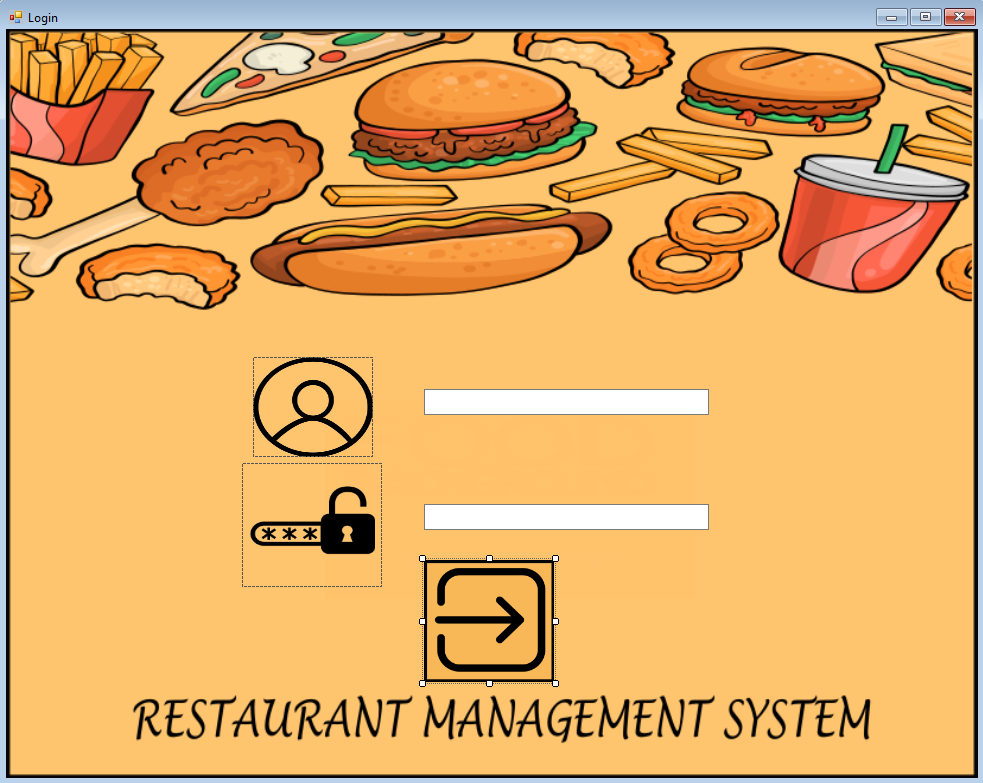
**ACHIEVED IMPLEMENTATION**

The final version of the project consists of most of the proposed functionalities. Below are the details for the database working within the management system and of the management system itself.

**Front-End Application**

**Login**

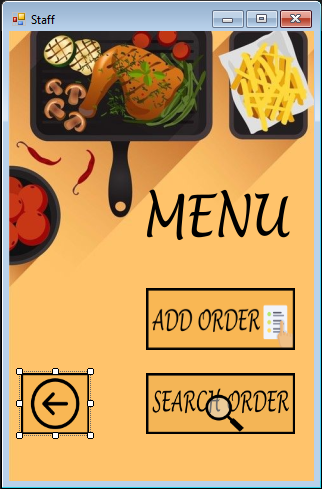
User inputs their staff ID and password. If ID is in Staff table of database and password matches, login is successful and they get access to further application features through Menu.



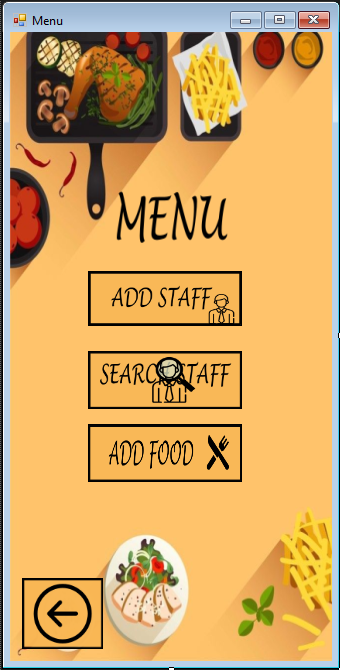
**Figure 1: Login window**

**Menu**

On logging in, the user will see a simple interface that shows the different abilities of the system. The Menu window will be shown according to the position of staff who just logged in, since staff members have different levels of access based on their job. Figure 2 shows the menu window for Cashiers, who can only access Orders related data in the database. Figure 3 shows the menu window for Managers, who have full access to all the Foods and Staff related data.



**Figure 2: Menu window for Cashiers**



**Figure 3: Menu window for Managers**

**Add Order**

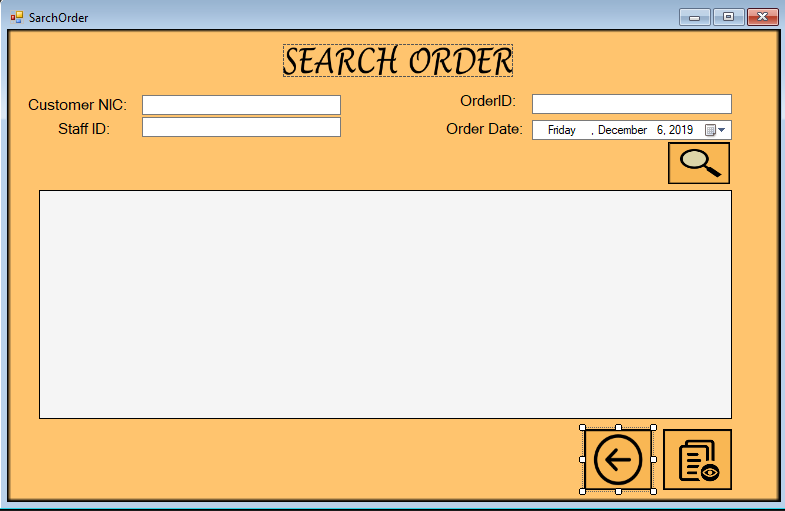
The interface to add orders is shown in Figure 4. As visible, adding an order will require the ID of the staff member taking and serving the order. After that, the food items, and their quantity included in the order, are put in the respective fields. On clicking Add button, these items are added and displayed in the List Box to give an overview of the order. A separate Group Box is dedicated for the fields to input Customer Details.



**Figure 4: Add Order Window**

**Search Order**

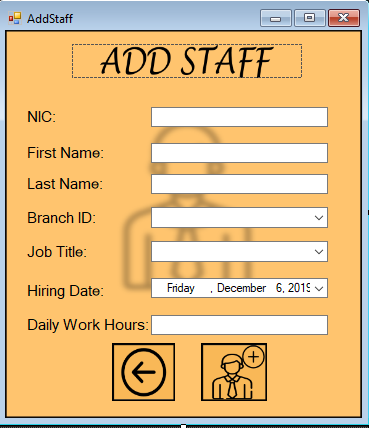
Figure 5 shows the window to search through the database of orders maintained by the system. There are four boxes to put in data related to an order and the relevant results are presented in the box below. It is not necessary for the user to put in all the details. The search functionality works as per whatever data the user provides.



**Figure 5: Search Order Window**

**Add Staff**:

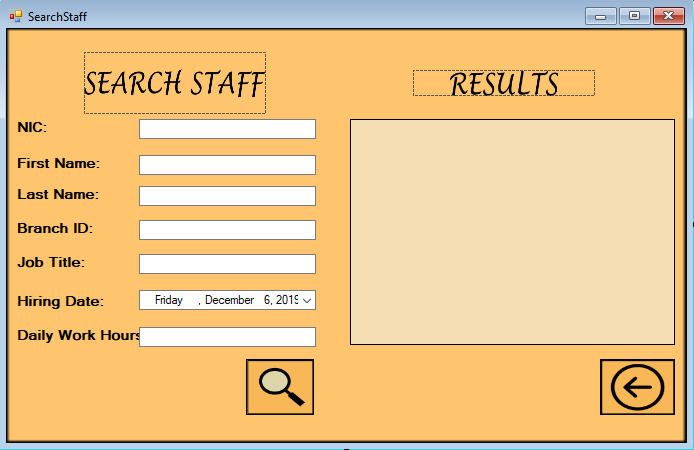
The Add Staff window is used to add new employees to the database. All the data can be put in the text boxes as shown in Figure 6. The fields specified with an asterisk are mandatory fields, without which the system would not advance to adding the employee. The functionality to modify already present records is hoped to be achieved in the final product.



**Figure 6: Add Staff Window**

**Search Staff**:

The window to search the staff records is shown in Figure 7. This window works in similar fashion to the window in Figure 5 (Search Orders), as the user does not have to necessarily put in all the fields and the search functionality will work with whatever is provided. This functionality accommodates the lapse of memory on the user’s side.



**Figure 7: Search Staff Window**

**Additional Functionality Achieved**

**Add / Delete Food Items**

This window will let the staff search if a certain food item is available in the restaurant or not. They can choose to delete the items from food menu, or add a new food item. Figure 8 shows the interface of this window.

This will allow the restaurant to keep their menu up-to-date with latest trends and availability of chefs.

****