Group Project   
Subject: Basic Cross-Platform Application Programming With .NET

*Restaurant Management Software*

**Version 1.0 approved**

**Prepared by Group 4-SE1504**

***Trần Mạnh Hùng***

***Nguyễn Thế Hiển***

***Lê Minh Khoa***

***Tôn Trọng Nghĩa***

**October 6, 2021**

**Table of Contents**

**Table of Contents**

[1.](#_30j0zll)Case Study **1**

[1.1.](#_1fob9te) Business opportunity and Problem 1

[1.2.](#_3znysh7) Business objective 1

[1.3.](#_2et92p0) Product overview 1

[1.4.](#_tyjcwt) Use Case 2

[**2.**](#_3dy6vkm) **Database Design 4**

[2.1.](#_1t3h5sf) Entity relationship diagram 4

[2.2.](#_4d34og8) DataBase Diagram 5

[**3.**](#_2s8eyo1) **System Architecture 6**

[**4.**](#_17dp8vu) **Implementation 10**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Group 4 | 01/10/2021 | initial draft | 1.0 draft 1 |
| Group 4 | 06/11/2021 | baseline following changes after inspection | 1.0 approved |

# Case Study

## Business opportunity and Problem

For a restaurant to run efficiently, a business owner or store manager must keep track of and handle a massive workload. Traditional manual restaurant management is not only prone to errors and mistakes, but also consumes a lot of money. Restaurant management software was born and has become an extremely effective "technology assistant" to support the management and control of business activities of managers.

## Business objective

To solve the problems that the restaurants are facing, it is necessary to put everything on the software and manage it all on the computer. This restaurant management system makes restaurant management easier than ever.

## Product overview

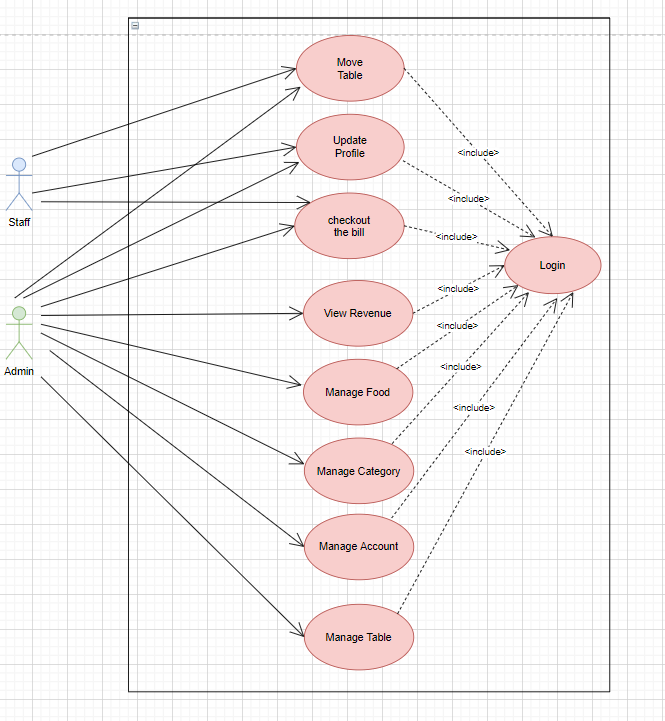
Restaurant management software is created to speed up the steps in the restaurant management process, it will replace paper paperwork or manual management, it also supports all table payment procedures, Manage revenue, inventory and restaurant staff more effectively. The system is expected to grow over several releases, eventually to a management system for the restaurant.

This system supports the following types of users:

Employees: They work in the restaurant, work with customers and perform table transfers, track and pay bills for dining tables.

Admin: They work with issues in the restaurant such as paying the table bill, tracking and managing sales, inventory, and restaurant staff.

## Use Case

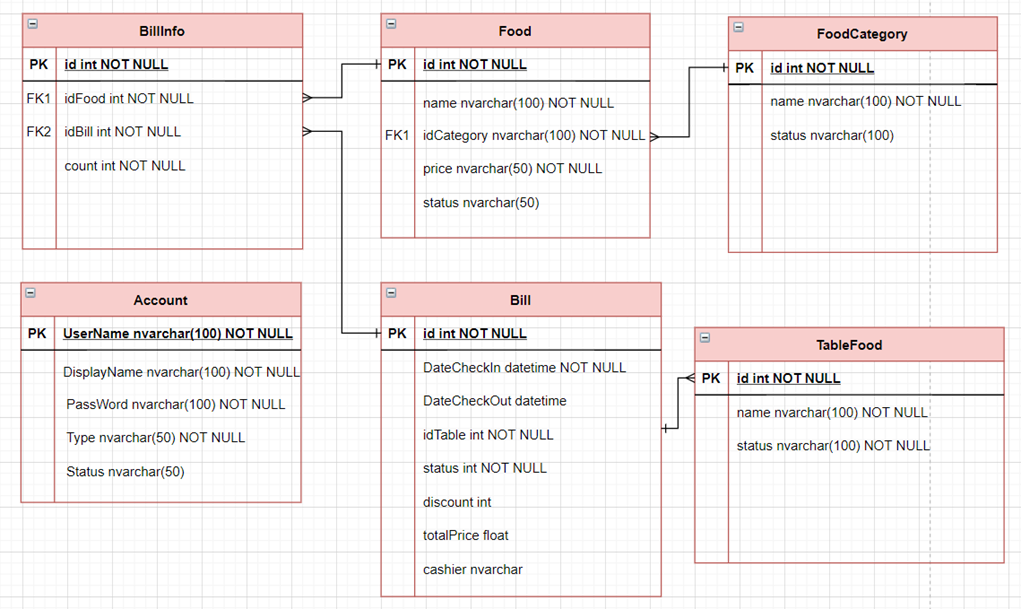


**Figure 1. Use Case diagram for release 1.0 of the Restaurant Management System.**

|  |  |
| --- | --- |
| ***Primary Actor*** | ***Use Cases*** |
| Manager | 1. Check and checkout the bill 2. Move table 3. View revenue table 4. Update Profile 5. Insert/ Delete Food 6. Update Food 7. View list of Food 8. Insert/ Delete Category 9. Update Category 10. View list of Category 11. Insert/ Delete Table 12. Update Table 13. View list of Table 14. Insert/Delete Account |
| Staff | 1. Check and checkout the bill 2. Move table 3. Update Profile |

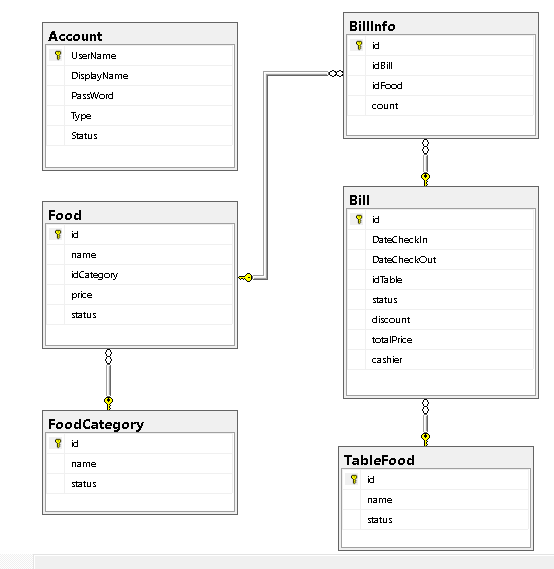
# Database Design

## Entity relationship diagram



**Figure 2. Entity Relationship Diagram for release 1.0 of the Restaurant Management System.**

## DataBase Diagram

**

# System Architecture

## 3.1 Environment

This software is build on .NET Core 5.0, required .NET Core 5.0 runtimes and Windows Form Libraries along with Windows Operating System to run it, database can run in localhost or on internet depend on customer’s requirement.

## 3.2 Architecture

Use 3 layers design pattern along with Repository design pattern, repository class will implementrepository interface, all repository methods will call from corresponding DAO classes. DAO classes methods is grouped by role. The system have 2 main roles, that is : staff, admin.

**System function for user:**

**Login:**

User input: username, password.

Description: User clicks Login button with username and password entered, if username and password exist in database, return User object, show main form.

**System function for staff:**

**Main form:**

User Input: N/A.

Description: When the main form is loaded, the tab shows the tables with the status "Available"/"Available" on the left and payment operations on the right.

**Add foods to the bill:**

User input: dish information and quantity

Description: When the main form is loaded, on the right payment tab, you can add dishes with information about dish name, quantity, unit price and amount. At the same time, the employee can enter the discount percentage on each order.

**Handling the table shifting operation:**

User input: Table information, table status

Description: At the bottom right of the main tab, the user selects the Switch table button to transfer the invoice as well as the status between the tables.

**Checkout the bill:**

User input: Invoice information

Description: Update revenue, final calculated price after entering more discount code.

**Update account profile:**

User Input: Display Name, Current Password, New Password and Confirm Password.

Description: Confirmation Password must match New Password and New Password must be different from Current Password.

**System function for manager: (Manipulation that can perform as Staff actions)**

**View revenue statistics:**

User input: Start date, End date.

Description: View the restaurant's sales statistics for the period from the start date to the end date.

**View food:**

User Input: N/A

Description: View all foods with their details.

**Insert food:**

User input: Food information

Description: New Food will be included in the database if the input meets the validation requirements (food name is a unique field).

**Update food:**

User input: Food information

Description: New food information will be updated based on their ID.

**Delete food:**

User Input: Information of food

Description: Select a food from the data grid view to delete. The delete operation will be performed after confirming the action in the message box.

**View Category:**

User Input: N/A

Description: View all Categories with their details.

**Insert Category:**

User Input: Category Information

Description: The new Category will be included in the database if the input meets the validation requirements (Category name is a unique field).

**Update Category:**

User Input: Category Information

Description: New Category information will be updated based on their ID.

**Delete Category:**

User Input: Information of Category

Description: Select a Category from the data grid view to delete. The delete operation will be performed after confirming the action in the message box.

**View Table:**

User Input: N/A

Description: View all Tables with their details.

**Insert Table:**

User Input: Table Information

Description: A new table will be included in the database if the input meets the validation requirements (Table name is a unique field).

**Update Table:**

User Input: Table Information

Description: New Table information will be updated based on their ID.

**Delete Table:**

User Input: Table Information

Description: Select a Table from the data grid view to delete. The delete operation will be performed after confirming the action in the message box.

**View Account:**

User Input: N/A

Description: View all Accounts with their details.

**Insert Account:**

User input: Account information

Description: The new account will be included in the database if the input meets the validation requirements (Account name is the only field).

**Update Account:**

User input: Account information

Description: New Account information will be updated based on their ID.

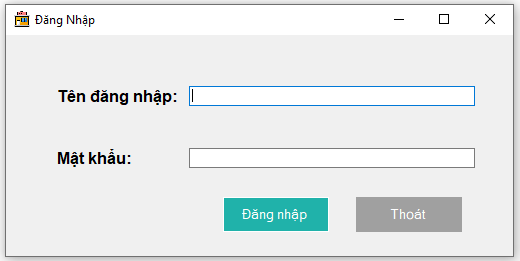
**Delete Account:**

User input: Account information

Description: Select an Account from the data grid view to delete. The delete operation will be performed after confirming the action in the message box.

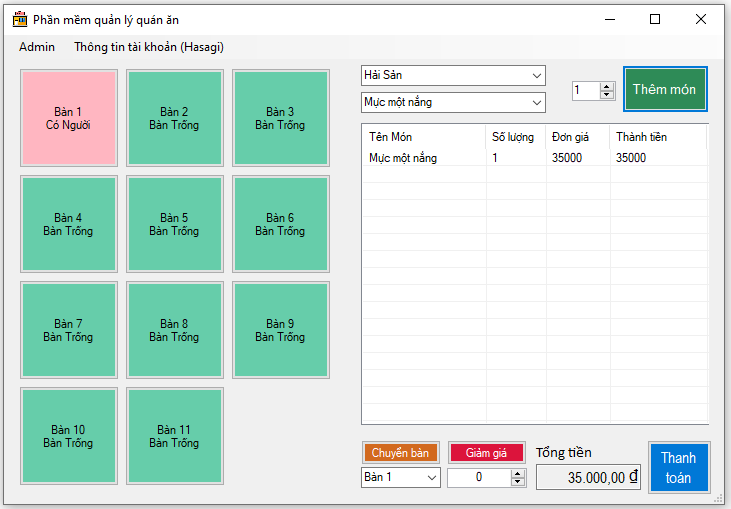
# Implementation

## Login



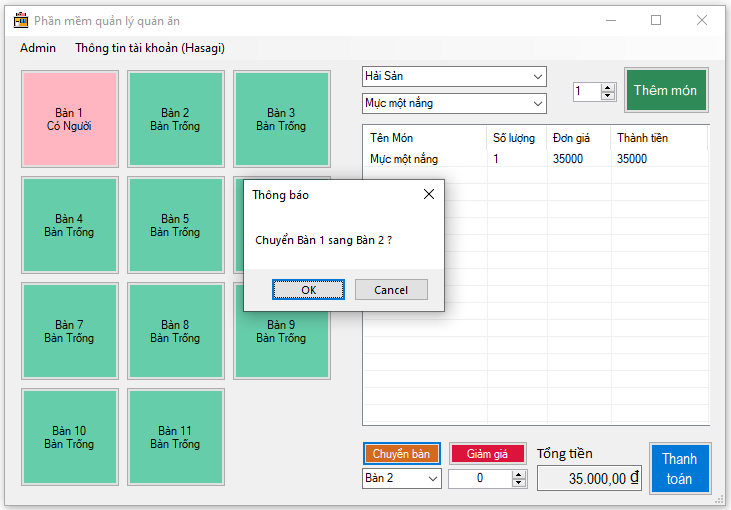
**Login Form**

## Check price and checkout the bill

****

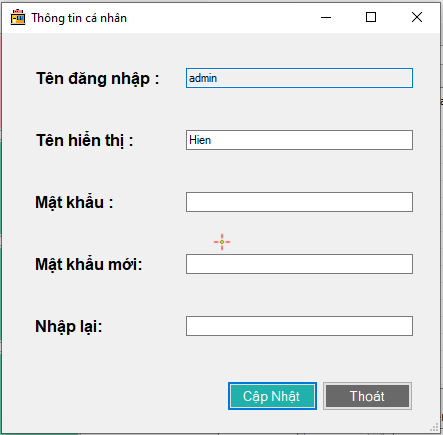
**Main Form**

## Handling the table shifting operation:



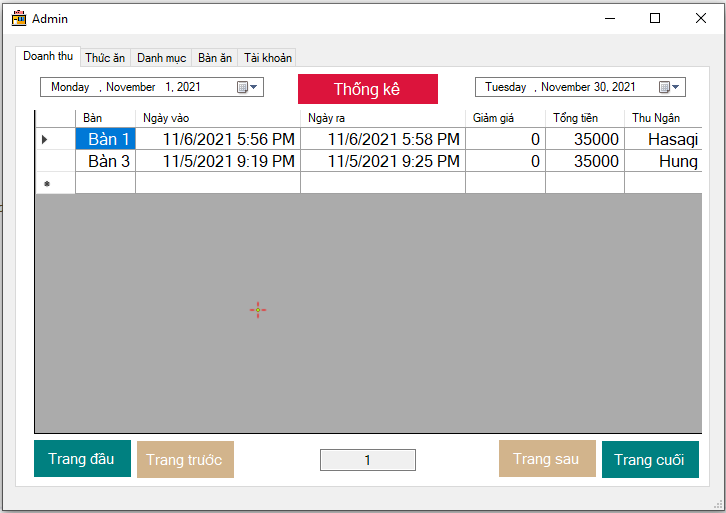
**Consumers Box**

## Update Profile



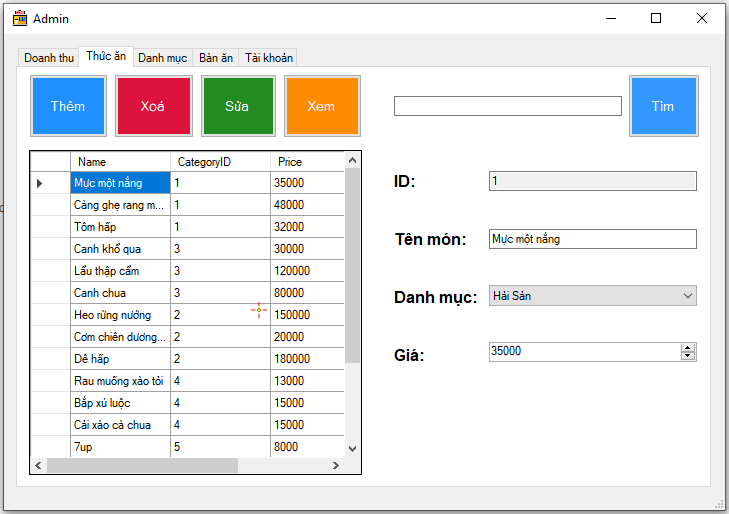
## Admin Form (Only Admin Role can access)

### View revenue table



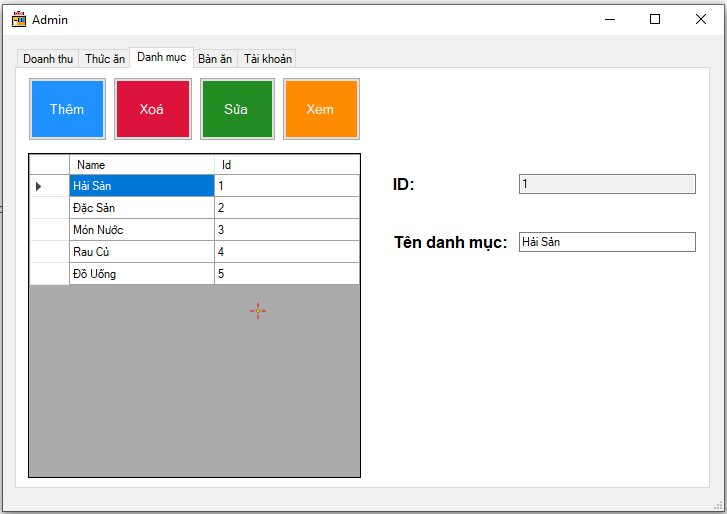
**View revenue table Box**

### Food Management



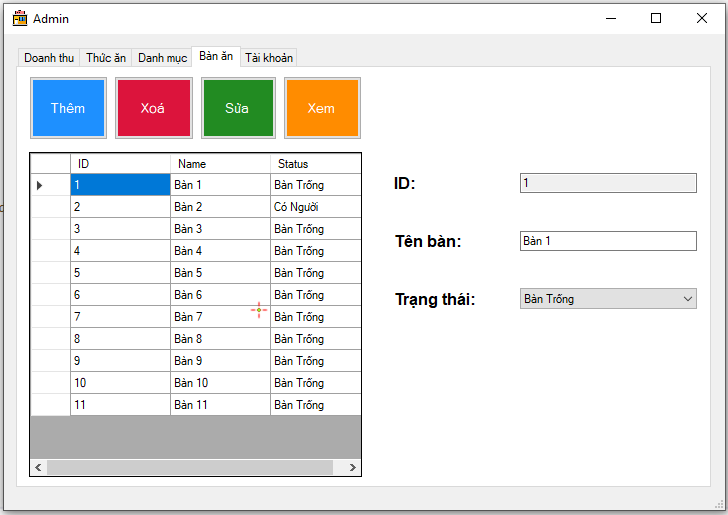
**Food Management Box**

### Category Management



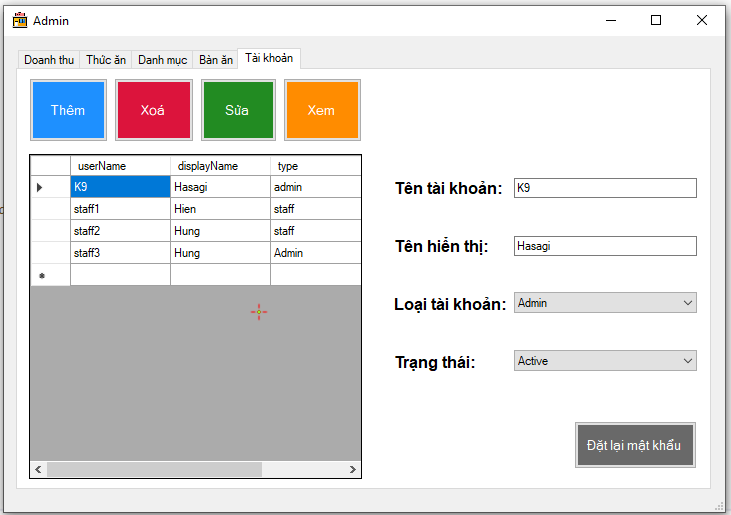
**Category Management Box**

### Table Management



**Table Management Box**

### Account Management



**Account Management Box**